Particle-Based Anisotropic Surface Meshing

Zichun Zhong* Xiaohu Guo* Wenping Wang† Bruno Lévy‡ Feng Sun† Yang Liu§ Weihua Mao¶

*University of Texas at Dallas †The University of Hong Kong ‡INRIA Nancy - Grand Est
§NVIDIA Corporation ¶UT Southwestern Medical Center at Dallas

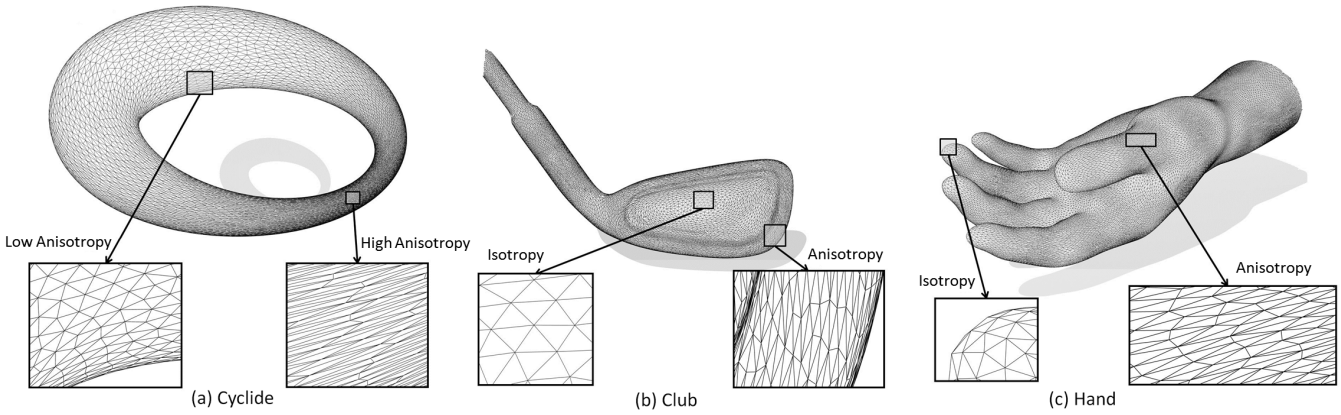

Figure 1: *Anisotropic meshing results generated by our particle-based method.*

Abstract

This paper introduces a particle-based approach for anisotropic surface meshing. Given an input polygonal mesh endowed with a Riemannian metric and a specified number of vertices, the method generates a metric-adapted mesh. The main idea consists of mapping the anisotropic space into a higher dimensional isotropic one, called "embedding space". The vertices of the mesh are generated by uniformly sampling the surface in this higher dimensional embedding space, and the sampling is further regularized by optimizing an energy function with a quasi-Newton algorithm. All the computations can be re-expressed in terms of the dot product in the embedding space, and the Jacobian matrices of the mappings that connect different spaces. This transform makes it unnecessary to explicitly represent the coordinates in the embedding space, and also provides all necessary expressions of energy and forces for efficient computations. Through energy optimization, it naturally leads to the desired anisotropic particle distributions in the original space. The triangles are then generated by computing the Restricted Anisotropic Voronoi Diagram and its dual Delaunay triangulation. We compare our results qualitatively and quantitatively with the state-of-the-art in anisotropic surface meshing on several examples, using the standard measurement criteria.

*{zichunzhong,xguo}@utdallas.edu, †{wenping,fsun}@cs.hku.hk,
‡bruno.levy@inria.fr, §thomasyoung.liu@gmail.com,
¶weihua.mao@utsouthwestern.edu

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling

Keywords: Anisotropic Meshing, Particle, and Gaussian Kernel.

Links: DL PDF

1 Introduction

Anisotropic meshing offers a highly flexible way of controlling mesh generation, by letting the user prescribe a direction and density field that steers the shape, size and alignment of mesh elements. In the simulation of fluid dynamics, it is often desirable to have elongated mesh elements with desired orientation and aspect ratio given by a Riemannian metric tensor field [Alauzet and Loseille 2010]. For surface modeling, it has been proved in approximation theory that the L_2 optimal approximation to a smooth surface with a given number of triangles is achieved when the anisotropy of triangles follows the eigenvalue and eigenvectors of the curvature tensors [Simpson 1994; Heckbert and Garland 1999]. This can be easily seen from the example of ellipsoid surface in Fig. 2 where the ratio of the two principal curvatures K_{max}/K_{min} is close to 1 near the two ends of the ellipsoid and is as high as 100 in the middle part. Anisotropic triangles stretched along the direction of minimal curvatures in the middle part of the ellipsoid provide best approximation, while isotropic triangles are needed at its two ends.

In this paper, we propose a new method for anisotropic meshing of surfaces endowed with a Riemannian metric. We rely on a particle-based scheme, where each pair of neighboring particles is equipped with a Gaussian energy. It has been shown [Witkin and Heckbert 1994] that minimizing this pair-wise Gaussian energy leads to a uniform isotropic distribution of particles. To compute the anisotropic meshing on surfaces equipped with Riemannian metric, we utilize the concept of a higher dimensional "embedding space" [Nash 1954; Kuiper 1955]. Our method optimizes the placement of the vertices, or particles, by uniformly sampling the higher dimensional embedding of the input surface. This embedding is designed in such a way that when projected back into the original space (usual-

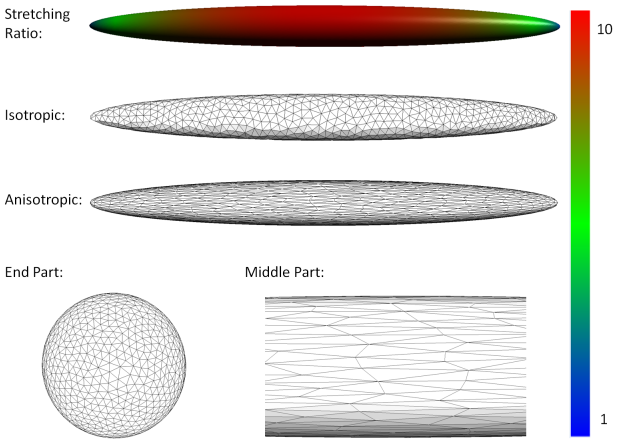

Figure 2: *Isotropic and anisotropic meshing with 1,000 output vertices of the ellipsoid surface. The stretching ratio (defined in Sec. 2.1) is computed as* $\sqrt{K_{max}/K_{min}}$*, where* K_{max} *and* K_{min} *are the two principal curvatures. Note that the "End Part" is rendered with orthographic projection along its long-axial direction, to better show the isotropy.*

ly 2D or 3D), a uniform sampling becomes anisotropic with respect to the input metric. Direct reference to the higher dimensional embedding is avoided by re-expressing all computations in terms of the dot product in the high-dimensional space, and the Jacobian matrices of the mappings that connect different spaces. Based on this re-expression we derive principled energy and force models for effective computation on the original manifold with a quasi-Newton optimization algorithm. Finally, the triangles are generated by computing a Restricted Anisotropic Voronoi Diagram and extracting the dual of its connected components.

This paper makes the following contributions for efficiently generating high-quality anisotropic meshes:

- It introduces a new particle-based formulation for anisotropic meshing. It defines the pair-wise Gaussian energies and forces between particles, and formulates the energy optimization in a higher dimensional "embedding space". We show further how anisotropic meshing can be translated into isotropic meshing in this higher dimensional embedding space (Sec. 3.1). The energy is designed in such a way that the particles are uniformly distributed on the surface embedded in this higher dimensional space. When the energy is optimized, the corresponding particles in the original manifold will achieve the anisotropic sampling with the desired input metric.

- It presents a computationally feasible and efficient method for our energy optimization (Sec. 3.2). The high-dimensional energy function and its gradient is mapped back into the original space, where the particles can be directly optimized. This computational approach avoids the need of computing the higher dimensional embedding space. Such energy optimization strategy shows very fast convergence speed, without any need for the explicit control of particle population (e.g., inserting or deleting particles to meet the desired anisotropy).

2 Background and Related Works

2.1 Definition of Anisotropy

Anisotropy denotes the way distances and angles are distorted. Geometrically, distances and angles can be measured by the dot product: $\langle \mathbf{v}, \mathbf{w} \rangle$, which is a bilinear function mapping a pair of vectors $\mathbf{v}, \mathbf{w}$ to $\mathbb{R}$. The dot product is symmetric, positive, and definite (SPD). If the dot product is replaced with another SPD bilinear form, then an anisotropic space is defined. We consider that a metric $\mathcal{M}(.)$, i.e. an SPD bilinear form, is defined over the domain $\Omega \subset \mathbb{R}^m$. In other words, at a given point $\mathbf{x} \in \Omega$, the dot product between two vectors $\mathbf{v}$ and $\mathbf{w}$ is given by $\langle \mathbf{v}, \mathbf{w} \rangle_{\mathcal{M}(\mathbf{x})}$. In practice, the metric can be represented by a symmetric $m \times m$ matrix $\mathbf{M}(\mathbf{x})$, in which case the dot product becomes:

$$\langle \mathbf{v}, \mathbf{w} \rangle_{\mathcal{M}(\mathbf{x})} = \mathbf{v}^T \mathbf{M}(\mathbf{x}) \mathbf{w}. \tag{1}$$

The metric matrix $\mathbf{M}(\mathbf{x})$ 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 orthogonal matrix $\mathbf{R}(\mathbf{x})$ contains its eigenvectors. We note that a globally smooth field $\mathbf{R}(\mathbf{x})$ may not exist for surfaces of arbitrary topology.

For the metric design, we use the following two options:
(1) In some of our experiments, we 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 those singularities, and 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 the same as Du et al. [2005]. They 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 $s_1 \leq s_2$. We simply call $\frac{s_2}{s_1}$ as the *stretching ratio*. This process will play a role later when the user specifies the desired input metric (Sec. 5).

(2) Note that if $\mathbf{M}(\mathbf{x})$ is given by users, the decomposition to $\mathbf{Q}(\mathbf{x})$ is non-unique. An equivalent decomposition $\mathbf{M}(\mathbf{x}) = \mathbf{Q}^O(\mathbf{x})^T \mathbf{Q}^O(\mathbf{x})$ is given by any matrix $\mathbf{Q}^O(\mathbf{x}) = \mathbf{O}(\mathbf{x})\mathbf{Q}(\mathbf{x})$, where $\mathbf{O}(\mathbf{x})$ is a $m \times m$ orthogonal matrix. In other words, $\mathbf{Q}(\mathbf{x})$ is unique *up to a rotation*.

However, it is easy to show that if a SPD metric $\mathbf{M}(\mathbf{x})$ is given, its square root $\mathbf{Q}'(\mathbf{x}) = \sqrt{\mathbf{M}(\mathbf{x})}$ is also a SPD matrix, and such decomposition is unique (Theorem 7.2.6 of [Horn and Johnson 1985]) and smooth (Theorem 2 of [Freidlin 1968]). $\mathbf{Q}'(\mathbf{x})$ is a symmetric affine mapping: $\mathbf{Q}'(\mathbf{x}) = \mathbf{R}(\mathbf{x})^T \mathbf{S}(\mathbf{x}) \mathbf{R}(\mathbf{x})$, and $\mathbf{M}(\mathbf{x}) = \mathbf{Q}'(\mathbf{x})\mathbf{Q}'(\mathbf{x})$. In Sec. 5.1, we use the "Mesh Font" example to show that $\mathbf{Q}'(\mathbf{x})$ can work well in our framework, given a user specified smooth metric field $\mathbf{M}(\mathbf{x})$.

It is interesting to note that if the metric tensor field is given as:

$$\mathbf{M}(\mathbf{x}) = \rho(\mathbf{x})^{\frac{2}{m}} \cdot \mathbf{I}, \tag{3}$$

where $\rho(\mathbf{x}) : \Omega \to \mathbb{R}$ and $\mathbf{I}$ is the identity matrix, then $\mathbf{M}(\mathbf{x})$ defines an isotropic metric graded with the density function $\rho(\mathbf{x})$.

Given the metric field $\mathbf{M}(\mathbf{x})$ and an open curve $\mathcal{C} \subset \Omega$, the length of $\mathcal{C}$ is defined as the integration of the length of tangent vector along the curve $\mathcal{C}$ with metric $\mathbf{M}(\mathbf{x})$. Then, the anisotropic distance $d_{\mathbf{M}}(\mathbf{x}, \mathbf{y})$ between two points $\mathbf{x}$ and $\mathbf{y}$ can be defined as the length of the (possibly non-unique) shortest curve that connects $\mathbf{x}$ and $\mathbf{y}$.

2.2 Previous Works

Anisotropic Voronoi Diagrams:
By replacing the dot product with the one defined by the metric, anisotropy can be introduced into the definition of the standard notions in computational geometry, e.g., Voronoi Diagrams and Delaunay Triangulations. The most general setting is given by Riemannian Voronoi diagrams [Leibon and Letscher 2000] that replace the distance with the anisotropic distance $d_{\mathbf{M}}(\mathbf{x}, \mathbf{y})$ defined above. Some theoretical results are known, in particular that Riemannian Voronoi diagrams admit a valid dual only in dimension 2 [Boissonnat et al. 2012]. However, a practical implementation is still beyond reach [Peyre et al. 2010]. For this reason, two simplifications are used to compute the Voronoi cell of each generator $\mathbf{x}_i$:

$$\begin{array}{lcl} \mathrm{Vor}_{\mathrm{Labelle}}(\mathbf{x}_i) & = & \{\mathbf{y} | d_{\mathbf{x}_i}(\mathbf{x}_i, \mathbf{y}) \leq d_{\mathbf{x}_j}(\mathbf{x}_j, \mathbf{y}), \forall j\} \\ \mathrm{Vor}_{\mathrm{Du}}(\mathbf{x}_i) & = & \{\mathbf{y} | d_{\mathbf{y}}(\mathbf{x}_i, \mathbf{y}) \leq d_{\mathbf{y}}(\mathbf{x}_j, \mathbf{y}), \forall j\} \end{array} \tag{4}$$

where:

$$d_{\mathbf{x}}(\mathbf{y}, \mathbf{z}) = \sqrt{(\mathbf{z}-\mathbf{y})^T \mathbf{M}(\mathbf{x})(\mathbf{z}-\mathbf{y})}.$$

The first definition $\mathrm{Vor}_{\mathrm{Labelle}}$ [Labelle and Shewchuk 2003] is easier to analyze theoretically. The bisectors are quadratic surfaces, known in closed form, and a provably-correct Delaunay refinement algorithm can be defined. The so-defined Anisotropic Voronoi Diagram (AVD) may be also thought of as the projection of a higher-dimensional power diagram [Boissonnat et al. 2008a]. The second definition $\mathrm{Vor}_{\mathrm{Du}}$ [Du and Wang 2005] is best suited to a practical implementation of Lloyd relaxation in the computation of Anisotropic Centroidal Voronoi Tessellations.

Centroidal Voronoi Tessellation and its Anisotropic Version:
A Centroidal Voronoi Tessellation (CVT) is a Voronoi Diagram such that each point $\mathbf{x}_i$ coincides with the centroid of its Voronoi cell. A CVT can be computed by either the Lloyd relaxation [Lloyd 1982] or a quasi-Newton energy optimization solver [Liu et al. 2009]. It generates a regular sampling [Du et al. 1999], from which a Delaunay triangulation with well-shaped isotropic elements can be extracted. In the case of surface meshing, it is possible to generalize this definition by using a geodesic Voronoi diagram over the surface [Peyre and Cohen 2004]. To make the computations simpler and cheaper, it is possible to replace the geodesic Voronoi diagram with the Restricted Voronoi Diagram (RVD) or Restricted Delaunay Triangulation (RDT), defined in [Edelsbrunner and Shah 1994] and used by several meshing algorithms, see [Dey and Ray 2010] and the references herein. Hence a Restricted Centroidal Voronoi Tessellation can be defined [Du et al. 2003]. With an efficient algorithm to compute the Restricted Voronoi Diagram, Restricted CVT can be used for isotropic surface remeshing [Yan et al. 2009].

CVT was further generalized to Anisotropic CVT (ACVT) by Du et al. [2005] using the definition $\mathrm{Vor}_{\mathrm{Du}}$ in Eq. (4). In each Lloyd iteration, an anisotropic Delaunay triangulation with the given Riemannian metric needs to be constructed, which is a time-consuming operation. Valette et al. [2008] proposed a discrete approximation of ACVT by clustering the vertices of a dense pre-triangulation of the domain. This discrete version is much faster than Du et al.'s continuous ACVT approach, at the expense of slightly degraded mesh quality. Sun et al. [2011] introduced a hexagonal Minkowski metric into ACVT optimization, in order to suppress obtuse triangles. Compared to these ACVT approaches, our particle-based scheme avoids the construction of AVD in the intermediate iterations of energy optimization, thus shows much faster performance as shown in Sec. 6.1.

Surface Meshing in Higher Dimensional Space:
Uniformly meshing surfaces embedded in higher dimensional space has also been studied in the literature [Cañas and Gortler 2006; Kovacs et al. 2010; Lévy and Bonneel 2012]. The work of Lévy and Bonneel [2012] is most related to ours, since both can be considered as using the framework of energy optimization in a higher dimensional embedding space. They extended the computation of CVT to a 6D space in order to achieve a curvature-adaptation. In particular, the anisotropic meshing on a 3D surface is transformed to an isotropic one on the surface embedded in 6D space, which can be efficiently computed by CVT equipped with Voronoi Parallel Linear Enumeration [Lévy and Bonneel 2012]. However, it does not provide users with the flexibility to control the anisotropy via an input metric tensor field. Our approach is designed to handle the more general anisotropic meshing scenario where a user-desired metric is specified.

Refinement-Based Delaunay Triangulation:
Anisotropic versions of point insertion in Delaunay triangulation has been successfully applied to many practical applications [Borouchaki et al. 1997a; Borouchaki et al. 1997b; Dobrzynski and Frey 2008]. Boissonnat et al. [2008b; 2011] introduced a Delaunay refinement framework, which is based on the goal to make 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 are proposed to add new vertices gradually to achieve the final anisotropic meshing. Our approach is different and consists in optimizing all the vertices of the mesh globally. Another difference is that we compute the dual of the connected components of the RVD [Yan et al. 2009] instead of the RDT. The results are compared in Sec 6.3.

Particle-Based Anisotropic Meshing:
Turk [1992] introduced repulsive points to sample a mesh for the purpose of polygonal remeshing. It was later extended by Witkin and Heckbert [1994] who used particles equipped with pair-wise Gaussian energy to sample and control implicit surfaces. Meyer et al. [2005] formulated the energy kernel as a modified cotangent function with finite support, and showed the kernel to be nearly scale-invariant as compared to the Gaussian kernel. It was later extended to handle adaptive, isotropic meshing of CAD models [Bronson et al. 2012] with particles moving in the parametric space of each surface patch. All these methods are only targeting isotropic sampling of surfaces.

To handle anisotropic meshing, Bossen and Heckbert [1996] incorporated the metric tensor into the distance function $d(\mathbf{x}, \mathbf{y})$, and use $f(\mathbf{x}, \mathbf{y}) = (1 - d(\mathbf{x}, \mathbf{y})^4) \cdot \exp(-d(\mathbf{x}, \mathbf{y})^4)$ to model the repulsion and attraction forces between particles. Shimada and his co-workers proposed physics-based relaxation of "bubbles" with a standard second-order system consisting of masses, dampers, and linear springs [Shimada and Gossard 1995; Shimada et al. 1997; Yamakawa and Shimada 2000]. They used a bounded cubic function of the distance to 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, to adaptively insert or delete particles/bubbles in certain regions. Thus if the initialization does not have a good estimation of the number of particles needed to fill the domain, it will take a long time to converge.

The method proposed in this paper is very similar to the idea of Adaptive Smoothed Particle Hydrodynamics (ASPH) [Shapiro et al. 1996] which uses inter-particle Gaussian kernels with an anisotropic smoothing tensor. However, as addressed in Sec. 3.3, ASPH directly formulates the energy in the original space without using the embedding space concept. To compute the forces between particles, the gradient of the varying metric tensor has to be ignored due to numerical difficulty. This treatment will lead to inaccurate

anisotropy in the computed mesh as shown in Fig. 4, when there are mild or significant variations in the metric.

Relation with the Theory of Approximation:
It has been studied in the theory of approximation [D'Azevedo 1991; Shewchuk 2002] that anisotropy is related to the optimal approximation of a function with a given number of piecewise-linear triangular elements. The anisotropy of the optimal mesh can be characterized, and optimization algorithms can be designed to best approximate the given function. The continuous mesh concept introduced by Loseille and Alauzet [2011a; 2011b] provides a relationship between the linear interpolation error and the mesh prescription, which has resulted in highly efficient anisotropic mesh adaptation algorithms. The relationship between anisotropic meshes and approximation theory has also been studied for higher-order finite elements [Mirebeau and Cohen 2010; Mirebeau and Cohen 2012], which leads to an efficient greedy bisection algorithm to generate optimal meshes.

Other Related Works:
This paper only focuses on anisotropic triangular meshing, which is different from other works handling anisotropic quad-dominant remeshing [Alliez et al. 2003; Kovacs et al. 2010; Lévy and Liu 2010; Zhang et al. 2010]. The notion of anisotropy has also been applied to the blue noise sample generation [Li et al. 2010].

3 The Particle Approach

Considering each vertex as a particle, the potential energy between the particles determines the inter-particle forces. When the forces applied on each particle become equilibrium, the particles reach the optimal balanced state with uniform distribution. To handle anisotropic meshing, we utilize the concept of "embedding space" [Nash 1954; Kuiper 1955]. In such high-dimensional embedding space, the metric is uniform and isotropic. When the forces applied on each particle reach equilibrium in this embedding space, the particle distribution on the original manifold will exhibit the desired anisotropic property.

Basic Framework: Given n particles with their positions $\mathbf{X} = \{\mathbf{x}_i | i = 1 \dots n\}$ on the surface Ω which is embedded in $\mathbb{R}^m$ space, we define the inter-particle *energy* between particles i and j as:

$$E^{ij} = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{4\sigma^2}}. \tag{5}$$

Here σ, called *kernel width*, is the fixed standard deviation of the Gaussian kernels. In Sec. 4.1 we will discuss how to choose an appropriate size of σ. Clearly, $E^{ij} = E^{ji}$.

The gradient of E^{ij} w.r.t. $\mathbf{x}_j$ can be considered as the force $\mathbf{F}^{ij}$ applied on particle j by particle i:

$$\mathbf{F}^{ij} = \frac{\partial E^{ij}}{\partial \mathbf{x}_j} = \frac{(\mathbf{x}_i - \mathbf{x}_j)}{2\sigma^2} e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{4\sigma^2}}. \tag{6}$$

Analogous to Newton's third law of motion, we have $\mathbf{F}^{ij} = -\mathbf{F}^{ji}$. We want to note that the formulation of Eq. (6) is similar to the particle repulsion/attraction idea of Witkin and Heckbert [1994].

By minimizing the total energy $E = \sum_i \sum_{j \neq i} E^{ij}$ with L-BFGS [Liu and Nocedal 1989], we can get a uniform isotropic sampling, where the forces applied on each particle reach equilibrium. It is shown in the supplementary Appendix that this particle-based energy formulation is fundamentally equivalent to Fattal's kernel-based formulation [2011], for the uniform isotropic case. However, Fattal's method does not handle anisotropic case. For non-uniform isotropic case, our analysis in Appendix shows the difference with respect to Fattal's approach, from both theoretical viewpoints and experimental results.

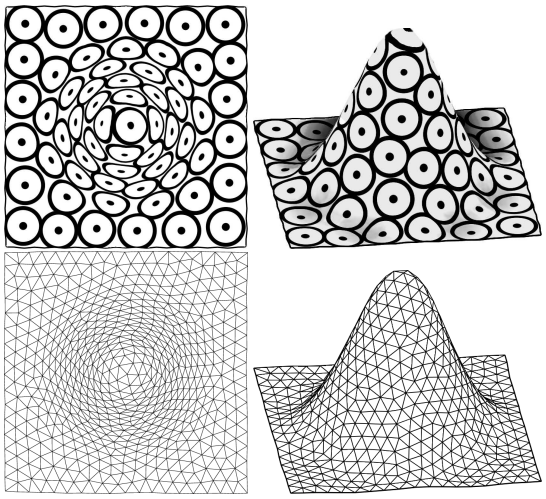

Figure 3: *A simple example of an embedding function that transforms an original 2D anisotropic surface Ω (left) into the surface $\overline{\Omega}$ (right) embedded in a higher dimensional space (3D in this example) where the metric is uniform and isotropic. In the general case a higher number of dimensions is required for $\overline{\Omega}$.*

3.1 Anisotropic Case

The top-left image of Fig. 3 shows a representation of a 2D metric field $\mathbf{M}$. The figure shows a set of points (black dots) and their associated unit circles (the bean-shaped curves, that correspond to the sets of points equidistant to each black dot). The bottom-left image of Fig. 3 shows the ideal mesh governed by such metric field: the length of the triangle edges, under the anisotropic distance, are close to be equal.

For this simple example of Fig. 3, one can see that the top-left image can be considered as the surface in the top-right image "seen from above". In other words, by embedding the flat 2D domain as a curved surface in 3D, one can recast the anisotropic meshing problem as the isotropic meshing of a surface embedded in higher-dimensional space.

In general, for an arbitrary metric $\mathbf{M}$, a higher-dimensional space will be needed [Nash 1954; Kuiper 1955]. We now consider that the surface Ω is mapped to $\overline{\Omega}$ that is embedded in a higher-dimensional space $\mathbb{R}^{\overline{m}}$. We simply call $\mathbb{R}^{\overline{m}}$ as the *embedding space* in this paper. Suppose the mapping function is ϕ: $\Omega \rightarrow \overline{\Omega}$, where $\Omega \subset \mathbb{R}^m$, $\overline{\Omega} \subset \mathbb{R}^{\overline{m}}$, and $m \leq \overline{m}$. Let us denote the particle positions on this surface $\overline{\Omega}$ by $\overline{\mathbf{X}} = \{\overline{\mathbf{x}}_i \mid \overline{\mathbf{x}}_i = \phi(\mathbf{x}_i), i = 1 \dots n\}$. A uniform sampling on $\overline{\Omega}$ can be computed by changing the inter-particle energy function E^{ij} of Eq. (5) as follows, hence defining $\overline{E}^{ij}$:

$$\overline{E}^{ij} = e^{-\frac{\|\overline{\mathbf{x}}_i - \overline{\mathbf{x}}_j\|^2}{4\sigma^2}}. \tag{7}$$

The gradient of $\overline{E}^{ij}$ w.r.t. $\overline{\mathbf{x}}_j$, i.e., the force $\overline{\mathbf{F}}^{ij}$ in the embedding space, can be defined similarly as:

$$\overline{\mathbf{F}}^{ij} = \frac{\partial \overline{E}^{ij}}{\partial \overline{\mathbf{x}}_j} = \frac{(\overline{\mathbf{x}}_i - \overline{\mathbf{x}}_j)}{2\sigma^2} e^{-\frac{\|\overline{\mathbf{x}}_i - \overline{\mathbf{x}}_j\|^2}{4\sigma^2}}. \tag{8}$$

3.2 Our Computational Approach

We show in this subsection how to optimize $\overline{E}^{ij}$ without referring to the coordinates of $\overline{\Omega}$ in the embedding space.

From the introduction of Sec. 2.1, we have seen that introducing anisotropy means changing the definition of the dot product. If we consider two small displacements $\mathbf{v}$ and $\mathbf{w}$ from a given location $\mathbf{x} \in \Omega$, then they are transformed into $\overline{\mathbf{v}} = \mathbf{J}(\mathbf{x})\mathbf{v}$ and $\overline{\mathbf{w}} = \mathbf{J}(\mathbf{x})\mathbf{w}$, where $\mathbf{J}(\mathbf{x})$ denotes the Jacobian matrix of ϕ at $\mathbf{x}$. The dot product between $\overline{\mathbf{v}}$ and $\overline{\mathbf{w}}$ is given by:

$$\langle \overline{\mathbf{v}}, \overline{\mathbf{w}} \rangle = \mathbf{v}^T \mathbf{J}(\mathbf{x})^T \mathbf{J}(\mathbf{x}) \mathbf{w} = \mathbf{v}^T \mathbf{M}(\mathbf{x}) \mathbf{w}. \tag{9}$$

In other words, given the embedding function ϕ, the anisotropy $\mathbf{M}$ corresponds to the first fundamental form of ϕ. If we now suppose that the anisotropy $\mathbf{M}(\mathbf{x})$ is known but not the embedding function ϕ, it is still possible to compute the dot product between two vectors in embedding space around a given point.

3.2.1 Computing the Energy Function

We now consider the inter-particle energy function in Eq. (7). Consider neighboring particles i and j. We use the Jacobian matrix evaluated at their middle point: $\frac{\mathbf{x}_i+\mathbf{x}_j}{2}$. In the following we denote $\mathbf{J}_{ij} = \mathbf{J}(\frac{\mathbf{x}_i+\mathbf{x}_j}{2})$, $\mathbf{M}_{ij} = \mathbf{M}(\frac{\mathbf{x}_i+\mathbf{x}_j}{2})$, $\mathbf{Q}_{ij} = \mathbf{Q}(\frac{\mathbf{x}_i+\mathbf{x}_j}{2})$, and $\mathbf{Q}'_{ij} = \mathbf{Q}'(\frac{\mathbf{x}_i+\mathbf{x}_j}{2})$ (see Sec. 2.1), for notational simplicity. Since the middle point is close to both $\mathbf{x}_i$ and $\mathbf{x}_j$, it is reasonable to make the following approximation:

$$\overline{\mathbf{x}}_i - \overline{\mathbf{x}}_j = \phi(\mathbf{x}_i) - \phi(\mathbf{x}_j) \approx \mathbf{J}_{ij}(\mathbf{x}_i - \mathbf{x}_j). \tag{10}$$

Thus the exponent in the term $\overline{E}^{ij}$ can be approximated as:

$$\begin{aligned} \|\overline{\mathbf{x}}_i - \overline{\mathbf{x}}_j\|^2 &= \langle \overline{\mathbf{x}}_i - \overline{\mathbf{x}}_j, \overline{\mathbf{x}}_i - \overline{\mathbf{x}}_j \rangle \\ &\approx (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{J}_{ij}^T \mathbf{J}_{ij} (\mathbf{x}_i - \mathbf{x}_j) \\ &= (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{M}_{ij} (\mathbf{x}_i - \mathbf{x}_j). \end{aligned} \tag{11}$$

Our inter-particle energy function can be approximated by:

$$\overline{E}^{ij} \approx e^{-\frac{(\mathbf{x}_i-\mathbf{x}_j)^T \mathbf{M}_{ij} (\mathbf{x}_i-\mathbf{x}_j)}{4\sigma^2}}. \tag{12}$$

The total energy is simply:

$$\overline{E} = \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} \overline{E}^{ij} \tag{13}$$

3.2.2 Computing the Force Function

Using Eq. (10) and Eq. (11), the inter-particle forces of Eq. (8) becomes:

$$\overline{\mathbf{F}}^{ij} \approx \frac{\mathbf{J}_{ij}(\mathbf{x}_i - \mathbf{x}_j)}{2\sigma^2} e^{-\frac{(\mathbf{x}_i-\mathbf{x}_j)^T \mathbf{M}_{ij} (\mathbf{x}_i-\mathbf{x}_j)}{4\sigma^2}}. \tag{14}$$

Here, for a particle i, different neighbors j have different $\mathbf{J}_{ij}$, which essentially encodes the variation of the metric.

The total force applied on each particle i is simply:

$$\overline{\mathbf{F}}^{i} = \sum_{j\neq i} \overline{\mathbf{F}}^{ji}. \tag{15}$$

Note that the expression in Eq. (14) still depends on the Jacobian matrix $\mathbf{J}_{ij}$. In our case, neither the embedding function ϕ nor its Jacobian is known. Therefore, we propose below an approximation of Eq. (14) that solely depends on the anisotropy field $\mathbf{M}(\mathbf{x})$.

We denote the set of particle i's neighbors as $N(i)$, and denote the vectors $\mathbf{v}_{ij} = \mathbf{x}_i - \mathbf{x}_j$, $j \in N(i)$. To better understand $\mathbf{J}_{ij}$, let us explore the relationship between the matrices $\mathbf{J}_{ij}$ and $\mathbf{Q}_{ij}$. $\mathbf{J}_{ij}$ is a $\overline{m} \times m$ matrix, where $\overline{m}$ is the dimension of the embedding space, and m is either 2 or 3, depending on whether Ω is a 2D domain or a 3D surface. Consider the QR decomposition: $\mathbf{J}_{ij} = \mathbf{U}_{ij} \begin{bmatrix} \mathbf{P}_{ij} \\ \mathbf{0} \end{bmatrix}$, where $\mathbf{U}_{ij}$ is a $\overline{m} \times \overline{m}$ unitary matrix (i.e. a rotation matrix in $\mathbb{R}^{\overline{m}}$), $\mathbf{P}_{ij}$ is a $m \times m$ matrix, and $\mathbf{0}$ is a $(\overline{m} - m) \times m$ block of zeros. Then:

$$\mathbf{M}_{ij} = \mathbf{J}_{ij}^T \mathbf{J}_{ij} = \mathbf{P}_{ij}^T \mathbf{P}_{ij}, \tag{16}$$

since $\mathbf{U}_{ij}^T \mathbf{U}_{ij} = \mathbf{I}$.

As mentioned in Sec. 2.1, if both $\mathbf{S}_{ij}$ and $\mathbf{R}_{ij}$ are given by users, then we can compose them and define $\mathbf{Q}_{ij} = \mathbf{S}_{ij}\mathbf{R}_{ij}$; if a smooth metric $\mathbf{M}_{ij}$ is given by users, we can use its square root $\mathbf{Q}'_{ij} = \sqrt{\mathbf{M}_{ij}}$. In the following derivation, both $\mathbf{Q}_{ij}$ and $\mathbf{Q}'_{ij}$ will lead to the same approximation technique. So we simply use $\mathbf{Q}_{ij}$ in the following discussion.

From Eq. (16) we can see that $\mathbf{P}_{ij}$ is exactly $\mathbf{Q}_{ij}$ up to a rotation, i.e., $\mathbf{P}_{ij} = \mathbf{O}_{ij}\mathbf{Q}_{ij}$ where $\mathbf{O}_{ij}$ is a $m \times m$ rotation matrix. We can simply represent $\mathbf{J}_{ij}$ as:

$$\mathbf{J}_{ij} = \mathbf{U}_{ij} \begin{bmatrix} \mathbf{O}_{ij}\mathbf{Q}_{ij} \\ \mathbf{0} \end{bmatrix} = \mathbf{W}_{ij} \begin{bmatrix} \mathbf{Q}_{ij} \\ \mathbf{0} \end{bmatrix}, \tag{17}$$

where $\mathbf{W}_{ij}$ is a rotation matrix in $\mathbb{R}^{\overline{m}}$:

$$\mathbf{W}_{ij} = \mathbf{U}_{ij} \begin{bmatrix} \mathbf{O}_{ij} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}. \tag{18}$$

If the metric field $\mathbf{M}(\mathbf{x})$ is smooth, then it is reasonable to approximate the rotation matrix $\mathbf{W}_{ij}$ with $\mathbf{W}_i$, where $\mathbf{W}_i$ is the rotation matrix of Eq. (18) evaluated at $\mathbf{x}_i$. Thus for $j \in N(i)$, the $\overline{m}$-dimensional vectors $\mathbf{J}_{ij}\mathbf{v}_{ij}$ in Eq. (14) can be approximated by:

$$\mathbf{J}_{ij}\mathbf{v}_{ij} = \mathbf{W}_{ij} \begin{bmatrix} \mathbf{Q}_{ij} \\ \mathbf{0} \end{bmatrix} \mathbf{v}_{ij} \approx \mathbf{W}_i \begin{bmatrix} \mathbf{Q}_{ij}\mathbf{v}_{ij} \\ \mathbf{0} \end{bmatrix}. \tag{19}$$

Then the force vector on particle i in Eq. (15) becomes:

$$\begin{aligned} \overline{\mathbf{F}}^{i} &= \textstyle\sum_{j\neq i} \frac{\mathbf{J}_{ij}\mathbf{v}_{ij}}{2\sigma^2} e^{-\frac{(\mathbf{x}_i-\mathbf{x}_j)^T \mathbf{M}_{ij} (\mathbf{x}_i-\mathbf{x}_j)}{4\sigma^2}} \\ &\approx \textstyle\sum_{j\neq i} \frac{1}{2\sigma^2} \mathbf{W}_i \begin{bmatrix} \mathbf{Q}_{ij}\mathbf{v}_{ij} \\ \mathbf{0} \end{bmatrix} e^{-\frac{(\mathbf{x}_i-\mathbf{x}_j)^T \mathbf{M}_{ij} (\mathbf{x}_i-\mathbf{x}_j)}{4\sigma^2}} \\ &= \mathbf{W}_i \textstyle\sum_{j\neq i} \frac{1}{2\sigma^2} \begin{bmatrix} \mathbf{Q}_{ij}\mathbf{v}_{ij} \\ \mathbf{0} \end{bmatrix} e^{-\frac{(\mathbf{x}_i-\mathbf{x}_j)^T \mathbf{M}_{ij} (\mathbf{x}_i-\mathbf{x}_j)}{4\sigma^2}}. \end{aligned} \tag{20}$$

If we define the m-dimensional forces:

$$\widetilde{\mathbf{F}}^{ij} = \frac{\mathbf{Q}_{ij}(\mathbf{x}_i - \mathbf{x}_j)}{2\sigma^2} e^{-\frac{(\mathbf{x}_i-\mathbf{x}_j)^T \mathbf{M}_{ij} (\mathbf{x}_i-\mathbf{x}_j)}{4\sigma^2}}, \tag{21}$$

and

$$\widetilde{\mathbf{F}}^{i} = \sum_{j\neq i} \widetilde{\mathbf{F}}^{ji}, \tag{22}$$

then the $\overline{m}$-dimensional force $\overline{\mathbf{F}}^{i}$ in Eq. (20) is simply:

$$\overline{\mathbf{F}}^{i} \approx \mathbf{W}_i \begin{bmatrix} \widetilde{\mathbf{F}}^{i} \\ \mathbf{0} \end{bmatrix} = \mathbf{V}_i \widetilde{\mathbf{F}}^{i}, \tag{23}$$

where $\mathbf{V}_i = \mathbf{W}_i \begin{bmatrix} \mathbf{I}_{m\times m} \\ \mathbf{0} \end{bmatrix}$, and $\mathbf{I}_{m\times m}$ is a $m \times m$ identity matrix.

Note that $\overline{\mathbf{F}}^{i}$ is the gradient in the higher-dimensional space $\mathbb{R}^{\overline{m}}$, while $\widetilde{\mathbf{F}}^{i}$ is in the original space $\mathbb{R}^{m}$. They are related by the matrix $\mathbf{V}_i$ in Eq. (23), which builds up a bijection between them. We can see that they can guide the optimization to arrive at the same equilibrium, since $\overline{\mathbf{F}}^{i} = 0 \Leftrightarrow \widetilde{\mathbf{F}}^{i} = 0$. Thus for the energy optimization purpose, we can simply replace Eq. (15) with Eq. (22) which can be computed directly on the original surface Ω.

The idea behind our force approximation can be interpreted as follows. At a given particle i, different neighboring pairs (i, j_1) and (i, j_2) may be equipped with different metrics $\mathbf{M}_{ij_1}$ and $\mathbf{M}_{ij_2}$ (as well as different Jacobians $\mathbf{J}_{ij_1}$ and $\mathbf{J}_{ij_2}$). The difference between $\mathbf{J}_{ij}$ encodes the variation of the metric locally around particle i. $\mathbf{J}_{ij}$ includes both "metric" part ($\mathbf{Q}_{ij}$) and "embedding rotation" part ($\mathbf{W}_{ij}$) (Eq. (19)). $\mathbf{W}_{ij}$ transforms the tangent plane at $\mathbf{x}_{ij}$ in the original space into the tangent plane in embedding space. Our approach uses the exact variation of neighboring metric $\mathbf{Q}_{ij}$, and approximates the embedding rotation $\mathbf{W}_{ij}$ with $\mathbf{W}_i$ in Eq. (19). Thus, the variation of embedding rotation is ignored in each particle's neighborhood, but the variation of metric is accounted.

In summary, we can optimize the uniform isotropic sampling on $\overline{\Omega}$ with the approximated energy of Eq. (12) and force of Eq. (21) using L-BFGS optimizer. They are both computed using the particle positions $\mathbf{X}$ on Ω, together with the metric $\mathbf{M}$. If $\mathbf{M}$ is given by users, we use its square root $\mathbf{Q}'$ instead of $\mathbf{Q}$ in Eq. (21). Although we utilize the elegant concept of "embedding space" to help develop our formulation for anisotropic meshing, we do NOT need to compute such an embedding space.

3.3 Importance of the Embedding Space

Anisotropic meshing is defined by the Riemannian metric $\mathbf{M}$, to locally affine-transform triangles into a "unit" space while enforcing the transformed triangles to be uniformly equilateral. Thus it is natural to directly define the energy optimization problem in this "unit" space. However, the metrics on each point can be different. Without establishing a coherent "unit" space, we cannot describe how these local affine copies of "unit" spaces can be "stitched" together. Our approach coherently considers all these local "unit" spaces by embedding the surface $\overline{\Omega}$ into high-dimensional space. Our energy in Eq. (7) is designed exactly by the definition of "anisotropy" – the affine-transformed triangles in $\overline{\Omega}$ should be uniformly equilateral (the particles should be uniformly distributed). This definition also leads to very efficient computations of forces in Eq. (21).

We want to emphasize that: without using this embedding space, the definition of energy function and the corresponding force formulation would be inconsistent with the definition of anisotropic mesh and thus lead to incorrect results. If we do not use this high-dimensional embedding space, the most intuitive formulation of energy will be Eq. (12). We elaborate on that and give some comparisons below.

Ignoring the Gradient of Metric (ASPH Method):
We need to note that the metric $\mathbf{M}_{ij}$ in Eq. (12) is dependent on the positions of particles $\mathbf{x}_i$ and $\mathbf{x}_j$. Therefore, the force formulation will involve the gradient of $\mathbf{M}_{ij}$ w.r.t. $\mathbf{x}_j$, which is numerically very difficult to compute. In the method of Adaptive Smoothed Particle Hydrodynamics (ASPH) [Shapiro et al. 1996], they use inter-particle Gaussian kernels and incorporated an anisotropic smoothing kernel to define the potential energy between particles, which is similar to Eq. (12). However, it is mentioned in their paper (Sec. 2.2.4 of [Shapiro et al. 1996]) that the gradient of metric term is ignored when computing the gradient of such inter-particle energy. Thus it leads to the following ASPH force formulation:

$$\widehat{\mathbf{F}}^{ij} \approx \frac{\mathbf{M}_{ij}(\mathbf{x}_i - \mathbf{x}_j)}{2\sigma^2} e^{-\frac{(\mathbf{x}_i-\mathbf{x}_j)^T \mathbf{M}_{ij}(\mathbf{x}_i-\mathbf{x}_j)}{4\sigma^2}}. \quad (24)$$

It is easy to see that Eq. (24) differs to Eq. (21) by only replacing $\mathbf{Q}_{ij}$ with $\mathbf{M}_{ij}$. Thus if the metric field is not constant, these two forces will lead to different local minima.

Our method in Eq. (21) only ignores the variation of embedding rotation in each particle's neighborhood, while the variation of metric is accounted. As confirmed in our experiments in Fig. 4, this has a measurable influence on the quality of generated meshes.

Ignoring the Variation of Jacobian Matrix:
Another approximation is to apply the pseudo-inverse of Jacobian matrix in the expression of Eq. (14). In Eq. (14), $\mathbf{J}_{ij}$ is different for different neighbors j. If we approximate $\mathbf{J}_{ij}$ with $\mathbf{J}_i$ in Eq. (14), and then apply the pseudo-inverse of $\mathbf{J}_i$, we arrive at the formulation (without the leading $\mathbf{M}_{ij}$ or $\mathbf{Q}_{ij}$) as follows:

$$\widehat{\mathbf{F}}^{ij} \approx \frac{(\mathbf{x}_i - \mathbf{x}_j)}{2\sigma^2} e^{-\frac{(\mathbf{x}_i-\mathbf{x}_j)^T \mathbf{M}_{ij}(\mathbf{x}_i-\mathbf{x}_j)}{4\sigma^2}}. \quad (25)$$

We emphasize the difference with our method: this variation is approximating $\mathbf{J}_{ij}$ with $\mathbf{J}_i$ in Eq. (14), while our method is approximating $\mathbf{W}_{ij}$ with $\mathbf{W}_i$ in Eq. (19). As mentioned above, $\mathbf{J}_{ij}$ contains both "metric" part $\mathbf{Q}_{ij}$ and "embedding rotation" part $\mathbf{W}_{ij}$. Thus the approximation of $\mathbf{J}_{ij}$ with $\mathbf{J}_i$ will potentially "erase" the variation of metric between neighboring particles.

To see their different effects on anisotropic mesh generation, we conduct the energy optimization in a 2D square domain using the following three choices of forces: (1) our force in Eq. (21); (2) the ASPH force in Eq. (24); and (3) the force in Eq. (25). As shown in Fig. 4, the 2D square domain is equipped with the background tensor field: $\mathbf{M}(\mathbf{x}) = diag\{Stretch(\mathbf{x})^2, 1\}$, where the field of $Stretch(\mathbf{x})$ is in the range of $[0.577, 9]$. In this experiment, we use a spatially nonuniform metric field – if $\mathbf{M}(\mathbf{x})$ is spatially uniform, then all the three forces will lead to the same particle configuration.

The comparative measurements of the quality of the generated anisotropic mesh are shown in Fig. 4, with triangle area quality G_{area}, angle histogram, G_{min}, G_{avg}, θ_{min}, θ_{avg}, and $\%_{<30^\circ}$, which are all defined in Sec. 4.5. The color-coded triangle area quality of our method shows that the areas of triangles computed using our force are uniform (all close to 1), which means the triangle sizes are conforming to the desired density defined by the metric tensor. From this experiment, we can see that performing energy optimization using our force in Eq. (21) generates the ideal anisotropic mesh, while optimizing the energy using the other two alternative forces in Eq. (24) and Eq. (25) cannot, which illustrates that formulating the energy optimization in the embedding space with our approximation leads to a principled formulation of inter-particle forces.

4 Implementation and Algorithm Details

Our particle-based method is summarized in Alg. 1 below. To help reproduce our results, we further detail each component of the algorithm and the implementation issues.

4.1 Kernel Width

The inter-particle energy as defined in Eq. (5) depends on the choice of the fixed kernel width σ. The slope of this energy peaks at distance of $\sqrt{2}\sigma$ and it is near zero at much smaller or much greater distances. If σ is chosen too small then particles will nearly stop spreading when their separation is about $5\sqrt{2}\sigma$, because there is almost no forces between particles. If σ is chosen too large then nearby particles cannot repel each other and the resulting sampling pattern will be poor. In this work, we choose σ to be proportional to the average "radius" of each particle when they are uniformly distributed on $\overline{\Omega}$: $\sigma = c_\sigma \sqrt{|\overline{\Omega}|/n}$, where $|\overline{\Omega}|$ denotes the area of the surface $\overline{\Omega}$ in the embedding space, n is the number of particles, and c_σ is a constant coefficient. Note that our goal is to let the particles

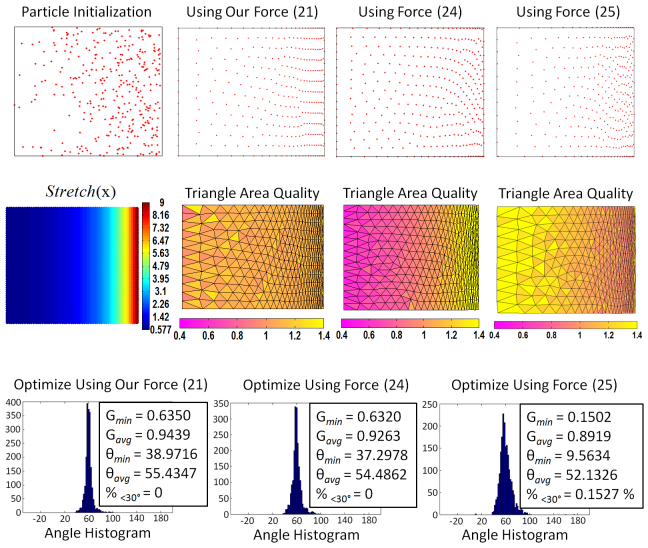

Figure 4: *Comparative experiments with 340 particles between (1) optimization using our force in Eq. (21); and (2) optimization using ASPH force in Eq. (24); and (3) optimization using force in Eq. (25). The color-coded triangle area quality shows our triangle areas in $\overline{\Omega}$ are uniform (all close to 1), which means the triangle sizes are conforming to the desired density defined by the metric tensor. Meanwhile, the triangle quality measurements show that the triangles are much closer to regular triangles after being transformed into embedding space.*

be uniformly and isotropically sampled on $\overline{\Omega}$. From our extensive experiments, we find out that the best isotropic mesh quality on $\overline{\Omega}$ can be achieved when $c_\sigma \approx 0.3$.

Given any input metric field $\mathbf{M}(\mathbf{x})$, the area of $\overline{\Omega}$ is: $|\overline{\Omega}| = \int_\Omega \sqrt{\det \mathbf{M}(\mathbf{x})} ds$. In this work the input surfaces are all triangular meshes, with metric defined on each vertex. For each triangle $\triangle_{abc}$ with vertices a, b, and c, we simply approximate its area in the embedding space as:

$$|\overline{\triangle}_{abc}| = \sqrt{\det(\frac{\mathbf{M}(\mathbf{x}_a) + \mathbf{M}(\mathbf{x}_b) + \mathbf{M}(\mathbf{x}_c)}{3})} \cdot |\triangle_{abc}|, \quad (26)$$

where $|\triangle_{abc}|$ is its area on the original surface. After summing all the areas of the triangles in the embedding space into $|\overline{\Omega}|$, we can set $\sigma = 0.3\sqrt{|\overline{\Omega}|/n}$ in our experiments.

In our implementation, for each particle we only compute the mutual effects from the particles within a neighborhood of five standard deviations (5σ), and use the Approximate Nearest Neighbor (ANN) library [Mount and Arya 1997] to quickly search such neighborhoods. Our experiments show that such a truncated Gaussian kernel (of 5σ range) works very well in practice, and generates regular hexagonal patterns of particles, similar to the results of CVT [Du et al. 1999; Liu et al. 2009]. For neighboring search in anisotropic space, we use ANN data structure to find Euclidean nearest neighbors within a larger range, and then we prune the spurious neighbors w.r.t. the prescribed metric.

4.2 Particle Optimization Algorithm

For any input metric field $\mathbf{M}(\mathbf{x})$, we use the adaptive initialization strategy of Valette et al. [2008], which distributes the initial sample positions based on the probability of the density $\sqrt{\det \mathbf{M}(\mathbf{x})}$.

Data: a surface Ω with metric $\mathbf{M}$, and the desired number of vertices n
Result: an anisotropic sampling $\mathbf{X}$ of Ω
Initialize particle locations $\mathbf{X}$;
while *stopping condition not satisfied* **do**
  Update the ANN data structure for the current sampling $\mathbf{X}$;
  for *each particle* i **do**
    Get particle i's neighbor $N(i)$ from ANN;
    for *each particle* $j \in N(i)$ **do**
      Compute $\overline{E}^{ij}$ using Eq. (12);
      Compute $\widetilde{\mathbf{F}}^{ij}$ using Eq. (21);
    end
    Sum the total force $\widetilde{\mathbf{F}}^i$ using Eq. (22);
    Project $\widetilde{\mathbf{F}}^i$ to the surface tangent using Eq. (27);
  end
  Sum the total energy $\overline{E}$ in Eq. (13);
  Run L-BFGS with $\overline{E}$ and $\{\widetilde{\mathbf{F}}^i\}$, to get updated locations $\mathbf{X}$;
  Project $\mathbf{X}$ onto the surface;
end

Algorithm 1: Anisotropic Particle Optimization with Metric $\mathbf{M}$.

We use the L-BFGS algorithm [Liu and Nocedal 1989] to optimize the sample positions. It is a quasi-Newton algorithm which can quickly find the minimum of the energy for our particle-based sampling. For each iteration of L-BFGS optimization, we update the total energy $\overline{E}$ in Eq. (13), and update its gradient by computing the total force $\widetilde{\mathbf{F}}^i$ applied on each particle i as in Eq. (22). The gradient in Eq. (22) used by the L-BFGS optimizer also needs to be projected onto the tangent space T_Ω of the surface:

$$\widetilde{\mathbf{F}}_i|_{T_\Omega} = \widetilde{\mathbf{F}}_i - [\widetilde{\mathbf{F}}_i \cdot \mathbf{n}(\mathbf{x}_i)]\mathbf{n}(\mathbf{x}_i), \quad (27)$$

where $\mathbf{n}(\mathbf{x}_i)$ is the unit normal of the surface at $\mathbf{x}_i$.

During L-BFGS optimization, the samples need to be constrained on the surface Ω. In each iteration, the updated sites $\mathbf{x}_j$ need to be projected to their nearest locations on Ω, if they are out of the boundary or out of the surface. This optimization process is iterated until convergence by satisfying a specified stopping condition, e.g., the magnitude of the gradient or the maximal displacement of particles is smaller than a threshold. Alg. 1 shows the details of our anisotropic particle optimization.

4.3 Mesh Generation

After the optimization of particle positions, the final output mesh is generated as the dual of the Anisotropic Voronoi Diagram (AVD) [Du and Wang 2005] restricted on the surface [Yan et al. 2009]. The dual of the connected components of Restricted Voronoi Diagram and topology control ensure that the components are discs (possibly by inserting points) [Yan et al. 2009; Lévy and Liu 2010]. This ensures homotopy equivalence in Nerve theorem [Borsuk 1948]. But we do not ensure homeomorphism since the conditions of [Edelsbrunner and Shah 1994] are not satisfied, which is considered as a limitation. Note that the mesh only needs to be computed once after the energy optimization. Thus it has significant advantage over all prior approaches based on ACVT, since they need to compute an AVD in each iteration of the optimization process. We compare both the computation speed and the quality of generated meshes with existing anisotropic meshing approaches in Sec. 6.

4.4 Extension to 6D Surfaces

Our particle-based optimization framework can be easily extended to handle the 6D embedding case as suggested by Lévy and Bonneel [2012]. The basic idea of Lévy and Bonneel's approach is to use the embedding $\phi : \Omega \rightarrow \mathbb{R}^6$ defined by:

$$\phi(\mathbf{x}) = [x, y, z, sn_x, sn_y, sn_z]^T, \tag{28}$$

where $\mathbf{x} = [x, y, z]^T$, $\mathbf{n}(\mathbf{x}) = [n_x, n_y, n_z]^T$ is the normal to Ω, and $s \in (0, \infty)$ is a user-defined parameter specifying the desired anisotropy of curvature-adaptation. Lévy and Bonneel solved the restricted CVT problem on such 6D surfaces, while in our framework we can simply perform optimization using the energy of Eq. (7) and the force of Eq. (8), since the embedding function ϕ is known. After energy optimization, the final mesh can be reconstructed by computing the Restricted Voronoi Diagram and its dual triangulation, using the Voronoi parallel linear enumeration.

4.5 Quality Measurements

To measure the isotropic triangular mesh quality, we use the criteria suggested by Frey and Borouchaki [1997]. 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. G_{min} is the minimal quality of the triangles, and G_{avg} is the average quality. θ_{min} is the smallest angle of the minimal angles of all triangles, and θ_{avg} is the average of the minimal angles of all triangles. $\%_{<30^\circ}$ is the percentage of triangles with their minimal angles smaller than 30°. In this paper, the angle histogram is also provided to show the angles of all generated triangles.

In anisotropic surface meshing, for each triangle $\triangle_{abc}$ we use its approximated metric $\mathbf{M}(\triangle_{abc}) = \frac{\mathbf{M}(\mathbf{x}_a)+\mathbf{M}(\mathbf{x}_b)+\mathbf{M}(\mathbf{x}_c)}{3}$. Then we use the corresponding $\mathbf{Q}(\triangle_{abc})$ or $\mathbf{Q}'(\triangle_{abc})$ matrix to affine-transform the triangle $\triangle_{abc}$, and use the above isotropic mesh quality measurements ($G_{min}, G_{avg}, \theta_{min}, \theta_{avg}, \%_{<30^\circ}$ and angle histogram) to check how close it is compared to a regular triangle in embedding space. We also define the following Area Quality G_{area} for each triangle $\triangle_i$, to evaluate how uniform are the areas of their affine-transformed copies:

$$G_{area}(\triangle_i) = \frac{|\overline{\triangle}_i|}{\sum_{j=1}^{n_t} |\overline{\triangle}_j|/n_t}, \tag{29}$$

where n_t is the total number of triangles and $|\overline{\triangle}_i|$ is the area of the affine-transformed triangle. In the optimal anisotropic meshing, each triangle will have the same area of $|\overline{\triangle}_i|$. Thus the best value of G_{area} for the generated anisotropic triangles is 1.

5 Results

We implement our algorithms using both Microsoft Visual C++ 2010 and Matlab R2010a. The experiments of meshing in 2D domains are coded in Matlab, while the meshing of surfaces are coded with C++. For the hardware platform, the experiments in Sec. 5.3 are tested with OpenMP parallel implementation on a laptop computer with Intel(R) Core(TM) i7-3720QM CPU with 2.60GHz, 8 threads (4 cores). The other experiments are tested with single-core implementation on a desktop computer with Intel(R) Xeon X5160 CPU with 3.00GHz. The users give their desired numbers of output vertices for all the experiments.

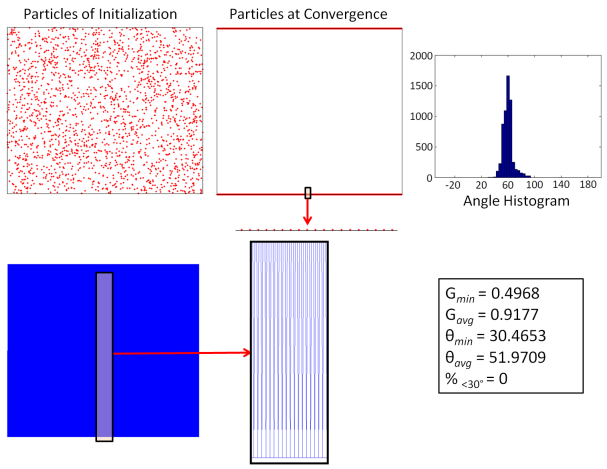

Figure 5: *Anisotropic meshing with 2,000 particles on a 2D domain with metric* $\mathbf{M}(\mathbf{x}) = diag\{Stretch(\mathbf{x})^2, 1\}$, *where* $Stretch(\mathbf{x}) = 1,000$.

5.1 Meshing in 2D Domains with Given Metrics

Fig. 5 shows the meshing result of a 2D square domain with $2,000$ samples, given a uniform metric $\mathbf{M}(\mathbf{x}) = diag\{Stretch(\mathbf{x})^2, 1\}$, where $Stretch(\mathbf{x}) = 1,000$. Fig. 6 shows the anisotropic meshing result of a 2D square domain with $3,000$ samples, with a varying metric tensor $\mathbf{M}(\mathbf{x}) = diag\{Stretch(\mathbf{x})^2, 1\}$, where $Stretch(\mathbf{x}) \in [1, 100]$.

Fig. 7 shows the meshing result with $20,000$ particles on a $2D$ domain of "Mesh Font" equipped with complex tensor fields, which includes both isotropic (graded with varying density) and anisotropic circular tensor field. The iso-contours of anisotropic distance to each black dot are given in red circles and ellipses in the two "zoom-in" parts. In Fig. 8, we use the "Mesh Font" example of Fig. 7 to show the comparison of meshing results between: (1) given $\mathbf{S}(\mathbf{x})$ and $\mathbf{R}(\mathbf{x})$, we use $\mathbf{Q}(\mathbf{x}) = \mathbf{S}(\mathbf{x})\mathbf{R}(\mathbf{x})$ in the force of Eq. (21); and (2) given $\mathbf{M}(\mathbf{x})$, we use $\mathbf{Q}'(\mathbf{x}) = \sqrt{\mathbf{M}(\mathbf{x})}$ in the force of Eq. (21). In the two "zoom-in" parts of the "Mesh Font" example, we can see that their mesh qualities are very close to each other. If $\mathbf{M}(\mathbf{x})$ is given, in the isotropic area, computing $\mathbf{Q}(\mathbf{x})$ from $\mathbf{M}(\mathbf{x})$ cannot give us a unique and smooth field of $\mathbf{Q}(\mathbf{x})$, since the two eigenvalues in the SVD decomposition of $\mathbf{M}(\mathbf{x})$ are equivalent. In this case, we should use $\mathbf{Q}'(\mathbf{x})$ instead of $\mathbf{Q}(\mathbf{x})$, as mentioned in Sec. 2.1.

5.2 Meshing on 3D Surfaces with Given Metrics

Fig. 9 shows the isotropic meshing of the Human Head surface controlled by a background density field $\rho(\mathbf{x}) = \frac{|K_{min}|+|K_{max}|}{2}$, where K_{min} and K_{max} are the principal curvatures. The original input surface has $53,696$ triangles, and is remeshed with $10,000$ particles running in 100 iterations (505.79 sec).

For testing anisotropic meshing on 3D surfaces, we use the following metric tensor:

$$\mathbf{M} = [\mathbf{v}_{min}, \mathbf{v}_{max}, \mathbf{n}] diag({s_1}^2, {s_2}^2, 0) [\mathbf{v}_{min}, \mathbf{v}_{max}, \mathbf{n}]^T, \tag{30}$$

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-controlled stretching factors along principal directions.

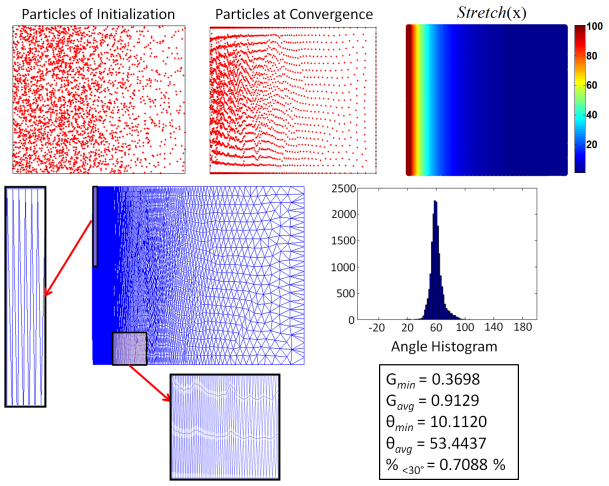

Figure 6: *Anisotropic meshing with 3,000 particles on a 2D domain with varying metrics* $\mathbf{M}(\mathbf{x}) = diag\{Stretch(\mathbf{x})^2, 1\}$*, where* $Stretch(\mathbf{x}) \in [1, 100]$.

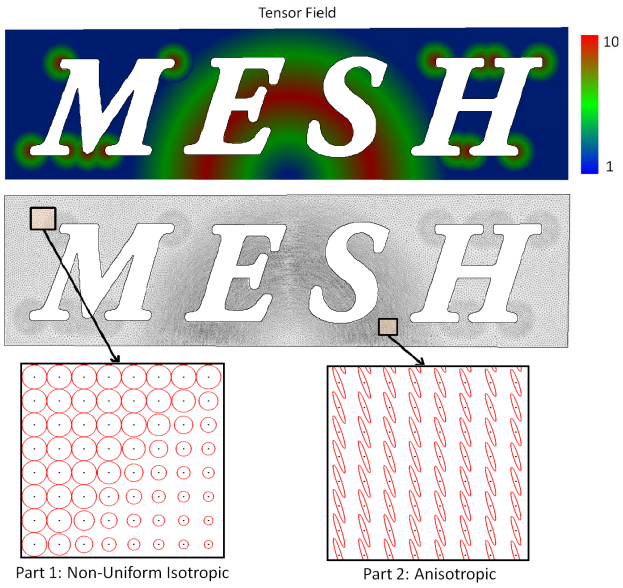

Figure 7: *Meshing with 20,000 particles on a 2D domain equipped with the combination of two metric fields: (1) the isotropic density metric* $\mathbf{M}(\mathbf{x}) = \rho(\mathbf{x})\mathbf{I}$*, where* $\rho(\mathbf{x}) \in [1, 100]$*; and (2) the anisotropic circular varying metric* $\mathbf{M}(\mathbf{x}) = \mathbf{R}(\mathbf{x})^T diag\{Stretch(\mathbf{x})^2, 1\}\mathbf{R}(\mathbf{x})$*, where* $Stretch(\mathbf{x}) \in [1, 10]$*, and* $\mathbf{R}(\mathbf{x})$ *is the rotation field shown in the middle part.*

The experiments on the Cyclide (Fig. 1) and Ellipsoid surfaces (Figs. 2 and 10) are computed by curvature-based metric tensor fields. We use the metric of Eq. (30) with $s_1 = \sqrt{K_{min}}$ and $s_2 = \sqrt{K_{max}}$, where K_{min} and K_{max} are the principal curvatures. The Cyclide surface in Fig. 1 has anisotropic stretching ratio $\frac{s_2}{s_1} \in [1, 18]$. Fig. 10 shows the anisotropic meshing of an ellipsoid surface with large stretching ratio $\frac{s_2}{s_1} \in [1, 100]$.

The Club and Hand surfaces (Fig. 1) and the Car surface (Fig. 11) are remeshed under user-specified stretching factors. As suggested by Alliez et al. [2003], Laplacian smoothing is applied to both the stretching factors and directions, to ensure smoothness of the input

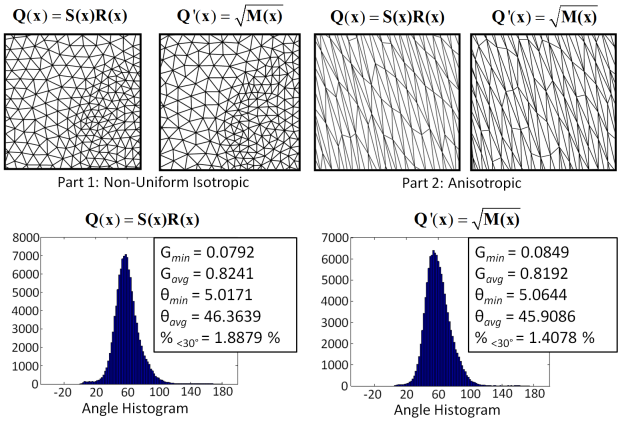

Figure 8: *For the two "zoom-in" parts in Fig. 7, we compare the meshing results of using* $\mathbf{Q}(\mathbf{x})$ *and* $\mathbf{Q}'(\mathbf{x})$ *in Eq. (21).*

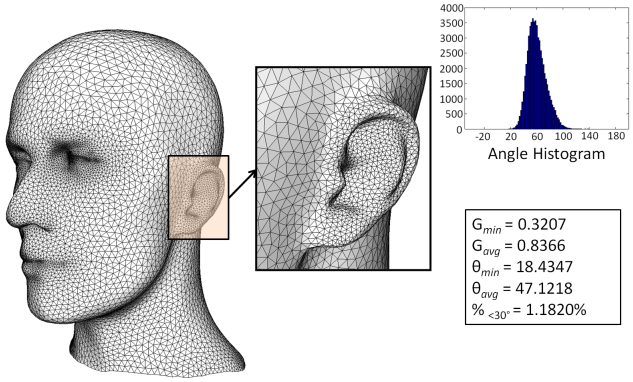

Figure 9: *Isotropic meshing with 10,000 output vertices of the Human Head surface with metric* $\mathbf{M}(\mathbf{x}) = \rho(\mathbf{x})\mathbf{I}$*, where the density* $\rho(\mathbf{x}) = \frac{|K_{min}|+|K_{max}|}{2}$*, with* K_{min} *and* K_{max} *being the principal curvatures.*

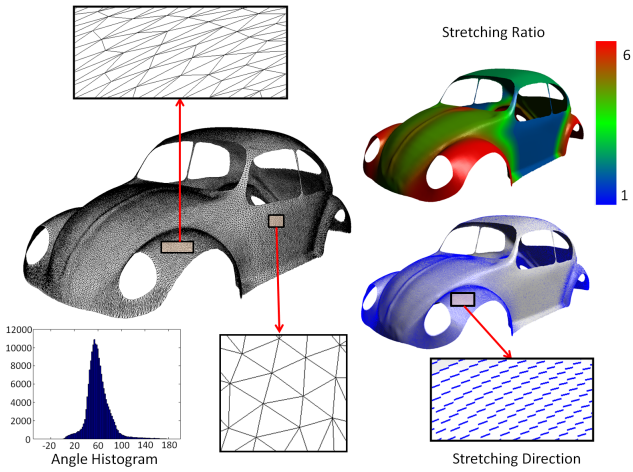

Figure 11: *The anisotropic meshing of the Car surface with the user-defined metric tensor field.*

metric field. Tab. 1 gives the statistics of the 3D surfaces meshed with given metrics.

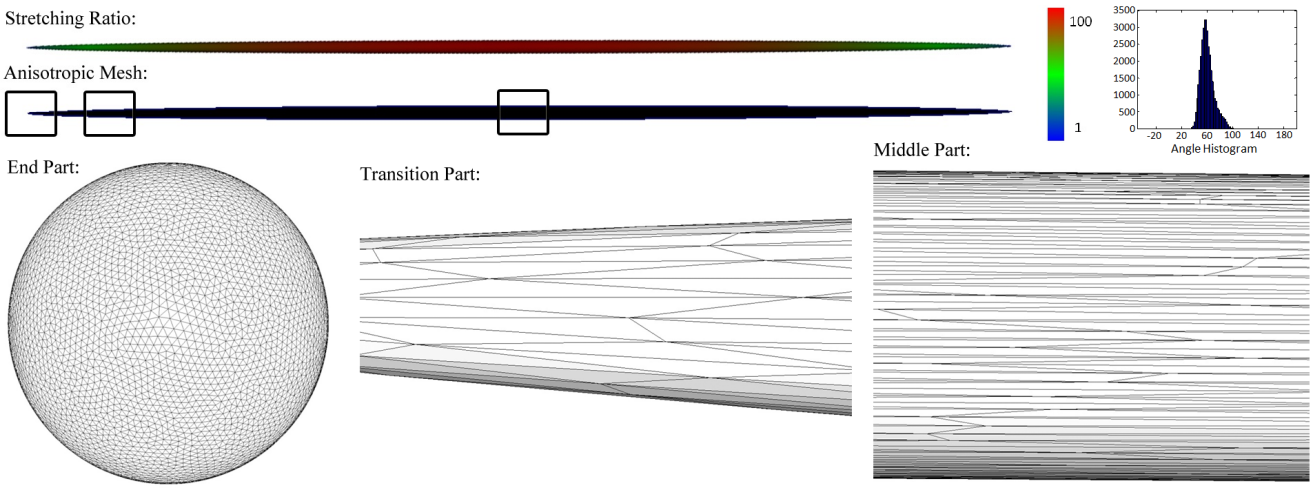

Figure 10: *The anisotropic meshing of Ellipsoid surface with large stretching ratio* $\frac{s_2}{s_1} \in [1, 100]$. *Note that the "End Part" is rendered with orthographic projection along its long-axial direction, to better show the isotropy.*

Table 1: *Statistics of the 3D surfaces with given metrics and their computation times.*

Model	Figure	Input # Vertex	Output # Vertex	Stretch	# Iteration	Time (sec)
Cyclide	1	129,600	8,000	[1, 18]	50	155.84
Hand	1	145,916	30,000	[1, 10]	100	1,263.55
Club	1	30,000	10,000	[1, 6]	100	219.60
Ellipsoid	10	40,962	6,000	[1, 100]	100	102.38
Car	11	197,948	30,000	[1, 6]	100	899.789

5.3 Meshing on 6D Surfaces for Curvature-Adaptation

As discussed in Sec. 4.4, we can easily extend our particle-based optimization to the 6D embedding space to achieve a curvature-adapted anisotropic meshing, as suggested by Lévy and Bonneel [2012]. In this experiment, we implement our isotropic 6D optimization algorithm with OpenMP parallel programming, since each particle's computation of energy and force is independent and can be easily parallelized. Our algorithm is running on a quad-core (8 threads) CPU as mentioned above, under the similar environmental setting as Lévy and Bonneel's work.

Fig. 12 shows the anisotropic remeshing of the Mechanism surface with $50,000$ particles. The input surface has $714,508$ triangles, and the computation time is $38.97sec$ for 100 iterations. Fig. 13 shows the anisotropic remeshing of the F1 surface with $60,000$ particles. The original input surface has $1,005,993$ triangles, and the computation time is $54.93sec$ for 100 iterations. The angle histogram shown in the figures are the quality of isotropic triangles in 6D space.

As a comparison, Lévy and Bonneel's Vorpaline algorithm is also parallelized with OpenMP. The same experiments in their paper are tested on a machine with Intel(R) Core(TM) i7-2720QM CPU 2.20GHz, 8 threads with hyperthreading activated. For these two surfaces they both run 5 iterations of Lloyd then 30 iterations of L-BFGS. The Mechanism surface takes $28.92sec$ and the F1 surface takes $35.93sec$ in their platform.

It can be seen that for each iteration, our particle-based computation is more efficient ($38.97sec/100iter < 28.92sec/35iter$), since it just need to sum up inter-particle energies and forces, instead of computing the restricted Voronoi diagram. However, the CVT-based energy optimization may converge faster with fewer number of iterations.

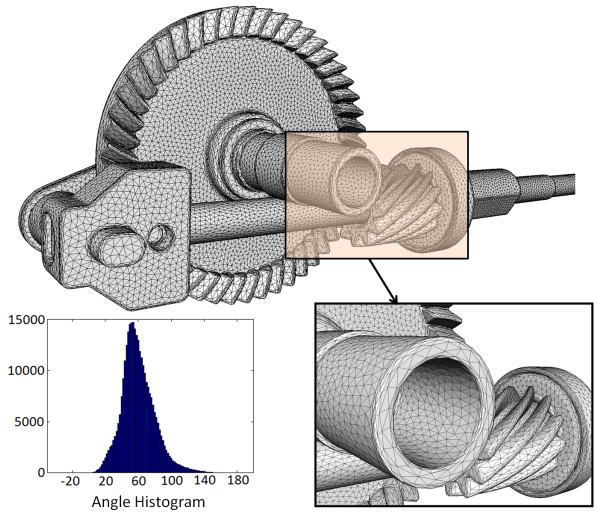

Figure 12: *Anisotropic mesh with 50,000 output vertices of the Mechanism surface generated by our particle-based optimization in 6D space.*

6 Comparisons

In this section, we show our comparative analysis and experiments with other anisotropic meshing approaches, including ACVT approaches (Sec. 6.1), other particle-based approaches (Sec. 6.2), and the Delaunay refinement approach (Sec. 6.3). To compare with other anisotropic triangulation methods, we use the same number of output vertices.

6.1 Comparison with ACVT

We compare the generated surface mesh quality and computational speed between our method and two ACVT methods: Du and Wang's method with triangle clipping strategy [Du and Wang 2005]

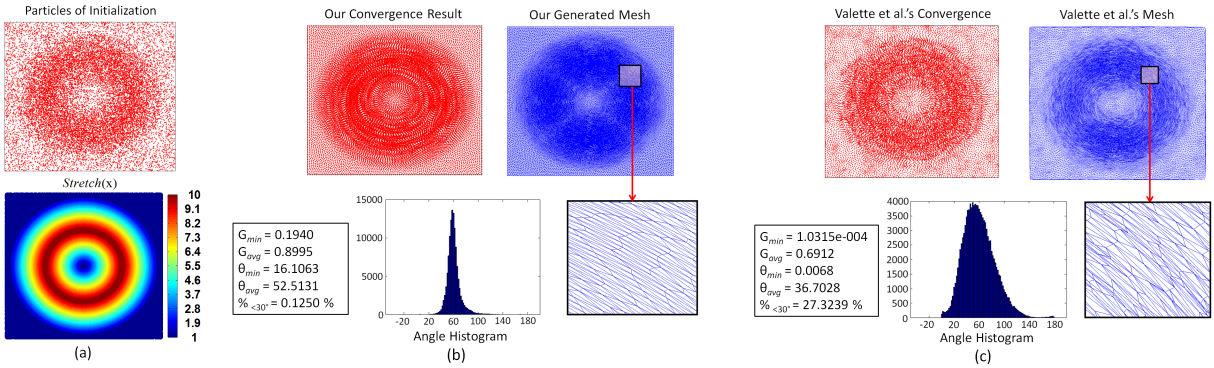

Figure 15: *Comparison with Valette et al.'s meshing result on the circular anisotropic tensor field (a) with the same initialization of* $20,000$ *samples. The unit square domain is tessellated by* $50,920$ *triangles as input. (b) Our meshing result. (c) Valette et al.'s meshing result.*

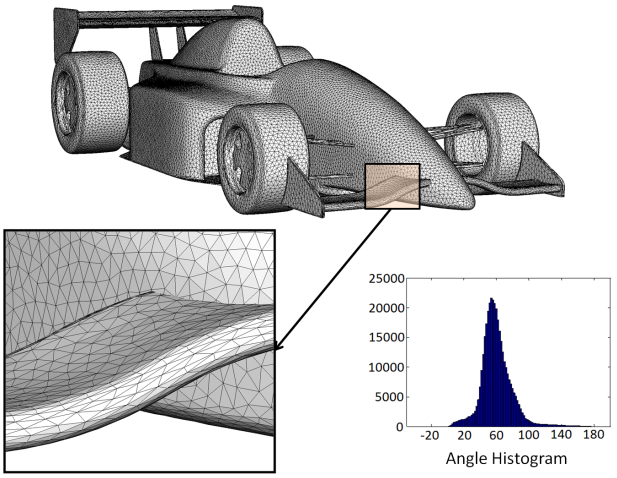

Figure 13: *Anisotropic mesh with 60,000 output vertices of the F1 surface generated by our particle-based optimization in 6D space.*

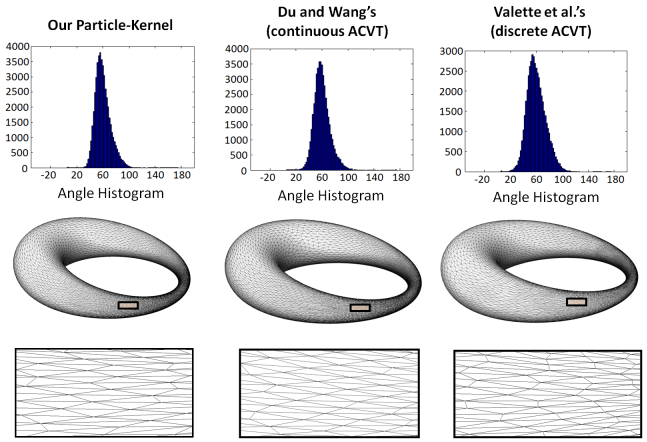

Figure 14: *Comparison with ACVT approaches with 8,000 output vertices.*

and Valette et al.'s discrete ACVT method [Valette et al. 2008]. All the three methods are implemented using Microsoft Visual C++ 2010. In the following comparison, we only run through 50 iterations, and re-sample $8,000$ vertices on the Cyclide surface with stretching ratio $\frac{s_2}{s_1} \in [1, 18]$.

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

Method	Time	G_{min}	G_{avg}	θ_{min}	θ_{avg}	$\%_{<30^\circ}$
Ours	$155.84s$	0.0930	0.8666	5.0327	49.6754	0.0446%
Du and Wang's	$19,538.86s$	0.1065	0.8590	5.2914	48.7769	0.1563%
Valette et al.'s	$863.15s$	0.0875	0.8286	5.2440	46.3001	0.2720%

Fig. 14 and Tab. 2 show the comparison results. On one hand, our approach provides comparable mesh quality with ACVT methods. On the other hand, our approach has much faster computational speed. From Tab. 2, we can see that our method is around 125 times faster than Du and Wang's continuous ACVT method, and around 5.5 times faster than Valette et al.'s discrete ACVT method. The input Cyclide surface is finely triangulated, thus Valette et al.'s method does not need to compute further subdivision as preprocessing. We want to note that due to its discrete nature, Valette et al.'s ACVT method does not work well for highly anisotropic stretching if the input triangulated mesh are not fine enough (See Fig. 15).

We perform further comparison with Valette et al.'s method on a unit square equipped with the circular anisotropic tensor field:

$$\mathbf{M}(\mathbf{x}) = \mathbf{R}(\mathbf{x})^T diag(Stretch(\mathbf{x})^2, 1)\mathbf{R}(\mathbf{x}), \qquad (31)$$

with the rotation field $\mathbf{R}(\mathbf{x})$ and the stretching field $Stretch(\mathbf{x})$ shown in Fig. 15(a). The square domain is discretized with $50,920$ triangles, and initialized with the same $20,000$ samples. Fig. 15(b) shows our converged anisotropic meshing result, and Fig. 15(c) shows the meshing computed by Valette et al.'s discrete ACVT method. When their method converges, we can see that their generated anisotropic mesh quality is much worse than ours: both G_{min} and θ_{min} of their resulting mesh are close to zero. To further test their method, we increase the number of domain triangles to $509,184$, but the quality of their generated mesh is still far from satisfactory: $G_{min} = 4.1851e-005$, $G_{avg} = 0.7245$, $\theta_{min} = 0.0017$, $\theta_{avg} = 38.6006$, $\%_{<30^\circ} = 21.75\%$. Using the domain tessellation of $509,184$ triangles, the time needed for Valette et al.'s method is $13,195.80$ sec, which is about 10 times slower than our method for computing the result in Fig. 15(b).

6.2 Comparison with Other Particle-Based Methods

In this subsection, we compare the convergence rate and the generated 2D anisotropic mesh quality between our method and two other particle-based methods: Bossen and Heckbert's method [1996]

and Shimada et al.'s method [1997]. All the three methods are implemented using Matlab R2010a.

Bossen and Heckbert's method is similar to Shimada et al.'s – they both consider repulsion and attraction forces. Bossen and Heckbert use $f(\mathbf{x},\mathbf{y}) = (1 - r(\mathbf{x},\mathbf{y})^4) \cdot \exp(-r(\mathbf{x},\mathbf{y})^4)$, where $r(x, y)$ is the distance function, to model the repulsion and attraction forces between particles. Shimada et al. use a bounded cubic function of the distance to model the inter-bubble forces. Both of their methods share the following same disadvantages: (1) Their methods are based on inter-particle forces without an explicit energy formulation. Thus it is hard to guarantee convergence and it needs much more iterations by using only forces to guide particle velocities. As a comparison, our energy-based optimization is based on a well-defined, smooth objective function, enabling efficient numerical optimization. (2) Both of their methods use adaptive population control – vertex insertions and deletions along with retriangulation or local bubble "population-check" according to the input metric tensor, which is very time-consuming. They define a desired triangle edge length or desired bubble size in accordance with the input metric. It is difficult to know beforehand the appropriate number of particles that is necessary and sufficient to fill the domain. Thus they have to insert and delete particles adaptively during iterations. If the initial number of particles is not close to the optimal number, it may take a longer time to converge. In contrast, the performance of our method is not affected by initial population, and the user is allowed to set their desired number of output particles.

To perform the comparative experiment, we estimate the optimal number of particles for both Bossen and Heckbert's and Shimada et al.'s methods using the concept of $\overline{\Omega}$ in embedding space: we compute the area of the embedding space by $|\overline{\Omega}| = \int_{\Omega} \sqrt{\det \mathbf{M}(\mathbf{x})} ds$, and divide it by the uniform area of each particle in the embedding space. Fig. 16 illustrates the meshing results of the three methods, on a low anisotropy field $Stretch(\mathbf{x}) \in [1, 3.95]$, using the same random initialization of 340 particles. Here the number of particles 340 is based on the above-mentioned population estimation. Both their methods show slower convergence rate than ours. If we run them on a slightly higher anisotropy field $Stretch(x) \in [2.53, 10]$ with $1,000$ particles, our method converged in 500 iterations (674.17 sec), while Bossen and Heckbert's method needs $10,000$ Iterations ($8,702.26$ sec) and Shimada et al.'s method needs $5,460$ Iterations ($12,702.26$ sec). All these experiments are shown in the supplementary videos.

6.3 Comparison with Delaunay Refinement Approach

In this subsection, we compare the result of Boissonnat et al.'s Delaunay refinement method [2011] with our particle-based optimization approach, with the simple 6D extension (Sec. 4.4). We conduct the following comparison on the Fertility surface with $13,971$ vertices in the input mesh. The output anisotropic meshes consists of $12,480$ vertices. From Fig. 17 we can tell visually that the mesh gradation of our result is smoother than theirs. Since both two methods are based on curvature-guided adaptation, we use Hausdorff distance to measure the approximation accuracy to the original input surface. The Hausdorff distance of our result is 0.002266% of the bounding-box's diagonal length, while the Hausdorff distance of their result is 0.005836%. From the triangle quality measurements in Fig. 17, we can see that our result is better than theirs as well.

In Tab. 3, we provide all the Hausdorff distances of our results, for those meshes that are generated with curvature-adaptation. The distances are measured using the percentages w.r.t. the bounding-box's diagonal length. The details of these experiments are described in Sec. 5.2 and Sec. 5.3.

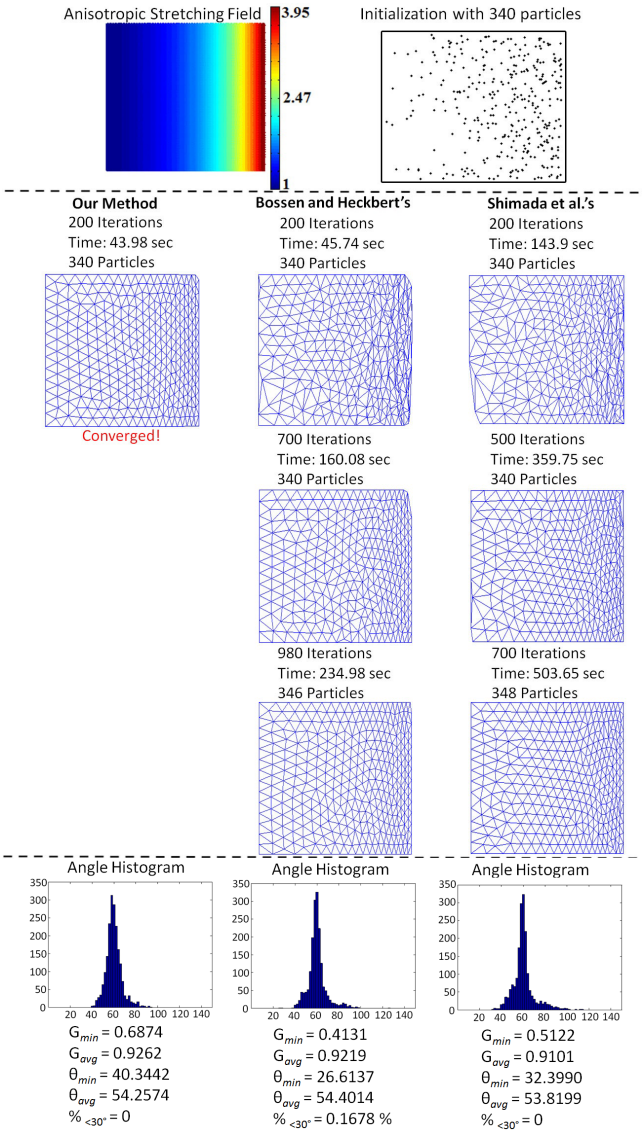

Figure 16: *Comparison with Bossen and Heckbert's and Shimada et al.'s methods, on 2D anisotropic meshing.*

Table 3: *Hausdorff distances between the input surfaces and our generated anisotropic meshes.*

Models	Cyclide	Ellipsoid (Fig. 2)	Ellipsoid (Fig. 10)	Mechanism	F1
Input #Vertices	$129,600$	$10,242$	$40,962$	$357,250$	$503,914$
Output #Vertices	$8,000$	$1,000$	$6,000$	$50,000$	$60,000$
Hausdorff Dist.	0.000819%	0.002307%	0.000529%	0.001439%	0.003315%

7 Conclusion and Future Work

The bottleneck of speed in our current framework is the search of neighboring particles. For isotropic meshing we can simply use the ANN library to search particles within five standard deviations (5σ). However, for anisotropic case such range of 5σ in the embedding space needs to be mapped back to the original space, and may result in very large range in Ω when the anisotropic stretching ratio is high. We plan to explore more efficient neighbor-search strategy for such high-anisotropy cases. In addition, we would like

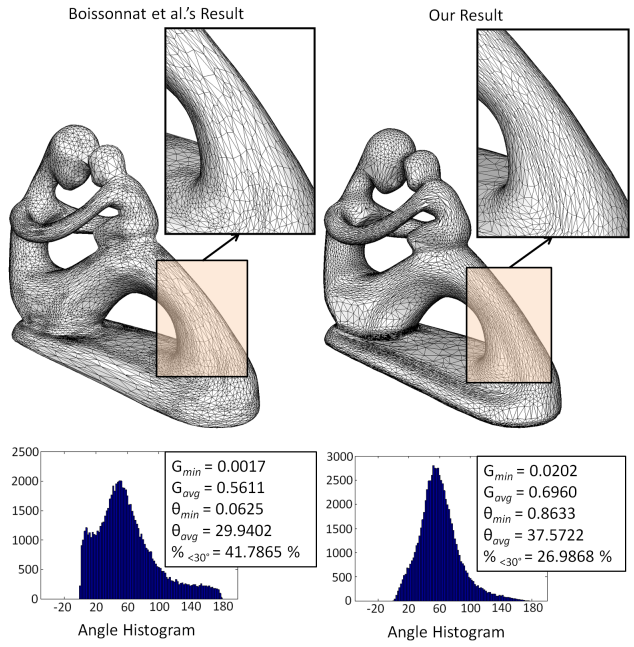

Figure 17: *Comparison of meshing results with 12,480 output vertices between Boissonnat et al.'s Delaunay refinement and our particle-based 6D optimization.*

to explore other kernel functions which have faster decay, and see how they affect the computational speed and mesh quality. Kernel functions that can produce stronger forces, when particles are close to each other, may also result in faster convergence rate. Our current work is only focusing on 2D domains and surfaces. It is still a challenging open problem to solve the 3D anisotropic meshing. Extending our particle-based framework to 3D volumes could potentially lead to some solutions which are both efficient and robust for 3D anisotropic meshing. We will investigate these interesting problems in the future.

Acknowledgements

The authors would like to thank the reviewers for their valuable comments. We also would like to thank UTD Computer Graphics & Animation (CGA) Lab members and Yufei Li for their help on model rendering, Pengbo Bo for providing some high quality input models, and Jean-Daniel Boissonnat for their results.

Zichun Zhong, Xiaohu Guo, Yang Liu, and Weihua Mao 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. The work of Wenping Wang was partially supported by the National Basic Research Program of China (2011CB302400), NSFC (61272019), and the Research Grant Council of Hong Kong (718209, 718010, 718311, 717012). Bruno Lévy was partially supported by the European Research Council (GOODSHAPE ERC-StG-205693) and the ANR (MORPHO and BECASIM).

References

ALAUZET, F., AND LOSEILLE, A. 2010. High-order sonic boom modeling based on adaptive methods. *Journal of Computational Physics 229*, 3, 561–593.

ALLIEZ, P., COHEN-STEINER, D., DEVILLERS, O., LÉVY, B., AND DESBRUN, M. 2003. Anisotropic polygonal remeshing. *ACM Transactions on Graphics 22*, 3, 485–493.

BOISSONNAT, J., WORMSER, C., AND YVINEC, M. 2008. Anisotropic diagrams: Labelle Shewchuk approach revisited. *Theoretical Computer Science 408*, 2–3, 163–173.

BOISSONNAT, J., WORMSER, C., AND YVINEC, M. 2008. Locally uniform anisotropic meshing. In *Proceedings of the 24th annual symposium on Computational geometry*, SCG '08, 270–277.

BOISSONNAT, J., WORMSER, C., AND YVINEC, M. 2011. Anisotropic Delaunay mesh generation. *Research Report RR-7712*.

BOISSONNAT, J.-D., DYER, R., AND GHOSH, A. 2012. Stability of Delaunay-type structures for manifolds. In *Symposium on Computational Geometry*, 229–238.

BOROUCHAKI, H., GEORGE, P. L., HECHT, F., LAUG, P., AND SALTEL, E. 1997. Delaunay mesh generation governed by metric specifications. part I. algorithms. *Finite Elements in Analysis and Design 25*, 1–2, 61–83.

BOROUCHAKI, H., GEORGE, P. L., AND MOHAMMADI, B. 1997. Delaunay mesh generation governed by metric specifications. part II. applications. *Finite Elements in Analysis and Design 25*, 1–2, 85–109.

BORSUK, K. 1948. On the imbedding of systems of compacta in simplicial complexes. *Fund. Math 35*, 217–234.

BOSSEN, F., AND HECKBERT, P. 1996. A pliant method for anisotropic mesh generation. In *5th International Meshing Roundtable*, 63–76.

BRONSON, J. R., LEVINE, J. A., AND WHITAKER, R. T. 2012. Particle systems for adaptive, isotropic meshing of CAD models. *Engineering with Computers 28*, 4, 331–344.

CAÑAS, G. D., AND GORTLER, S. J. 2006. Surface remeshing in arbitrary codimensions. *Visual Computer 22*, 9, 885–895.

D'AZEVEDO, E. F. 1991. Optimal triangular mesh generation by coordinate transformation. *SIAM Journal on Scientific and Statistical Computing 12*, 4, 755–786.

DEY, T. K., AND RAY, T. 2010. Polygonal surface remeshing with Delaunay refinement. *Engineering with Computers 26*, 3, 289–301.

DOBRZYNSKI, C., AND FREY, P. 2008. Anisotropic Delaunay mesh adaptation for unsteady simulations. In *17th International Meshing Roundtable*, 177–194.

DU, Q., AND WANG, D. 2005. Anisotropic centroidal Voronoi tessellations and their applications. *SIAM Journal on Scientific Computing 26*, 3, 737–761.

DU, Q., FABER, V., AND GUNZBURGER, M. 1999. Centroidal Voronoi tessellations: Applications and algorithms. *SIAM Review 41*, 4, 637–676.

DU, Q., GUNZBURGER, M. D., AND JU, L. 2003. Constrained centroidal Voronoi tessellations for surfaces. *SIAM Journal on Scientific Computing 24*, 5, 1488–1506.

EDELSBRUNNER, H., AND SHAH, N. R. 1994. Triangulating topological spaces. In *Symposium on Computational Geometry*, 285–292.

FATTAL, R. 2011. Blue-noise point sampling using kernel density model. *ACM Transactions on Graphics 30*, 4, 48:1–48:12.

FREIDLIN, M. 1968. On the factorization of non-negative definite matrices. *Theory of Probability and Its Applications 13*, 2, 354–356.

FREY, P. J., AND BOROUCHAKI, H. 1997. Surface mesh evaluation. In *6th International Meshing Roundtable*, 363–373.

HECKBERT, P. S., AND GARLAND, M. 1999. Optimal triangulation and quadric-based surface simplification. *Computational Geometry 14*, 1–3, 49–65.

HORN, R. A., AND JOHNSON, C. R. 1985. *Matrix Analysis*. Cambridge University Press.

KOVACS, D., MYLES, A., AND ZORIN, D. 2010. Anisotropic quadrangulation. In *Proceedings of the 14th ACM Symposium on Solid and Physical Modeling*, SPM '10, 137–146.

KUIPER, N. H. 1955. On C^1-isometric embeddings I. In *Proc. Nederl. Akad. Wetensch. Ser. A*, 545–556.

LABELLE, F., AND SHEWCHUK, J. R. 2003. Anisotropic Voronoi diagrams and guaranteed-quality anisotropic mesh generation. In *Proceedings of the 19th Annual Symposium on Computational Geometry*, ACM, 191–200.

LEIBON, G., AND LETSCHER, D. 2000. Delaunay triangulations and Voronoi diagrams for Riemannian manifolds. In *Proceedings of the Sixteenth Annual Symposium on Computational Geometry*, SCG '00, 341–349.

LÉVY, B., AND BONNEEL, N. 2012. Variational anisotropic surface meshing with Voronoi parallel linear enumeration. In *21st International Meshing Roundtable*, 349–366.

LÉVY, B., AND LIU, Y. 2010. Lp centroidal Voronoi tessellation and its applications. *ACM Transactions on Graphics 29*, 4, 119:1–119:11.

LI, H., WEI, L., SANDER, P. V., AND FU, C. 2010. Anisotropic blue noise sampling. *ACM Transactions on Graphics 29*, 6, 167:1–167:12.

LIU, D. C., AND NOCEDAL, J. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming 45*, 3, 503–528.

LIU, Y., WANG, W., LÉVY, B., SUN, F., YAN, D., LU, L., AND YANG, C. 2009. On centroidal Voronoi tessellation – energy smoothness and fast computation. *ACM Transactions on Graphics 28*, 4, 101:1–101:17.

LLOYD, S. 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory 28*, 2, 129–137.

LOSEILLE, A., AND ALAUZET, F. 2011. Continuous mesh framework part I: Well-posed continuous interpolation error. *SIAM Journal on Numerical Analysis 49*, 1, 38–60.

LOSEILLE, A., AND ALAUZET, F. 2011. Continuous mesh framework part II: Validations and applications. *SIAM Journal on Numerical Analysis 49*, 1, 61–86.

MEYER, M. D., GEORGEL, P., AND WHITAKER, R. T. 2005. Robust particle systems for curvature dependent sampling of implicit surfaces. In *International Conference on Shape Modeling and Applications*, 124–133.

MIREBEAU, J., AND COHEN, A. 2010. Anisotropic smoothness classes: From finite element approximation to image models. *Journal of Mathematical Imaging and Vision 38*, 1, 52–69.

MIREBEAU, J., AND COHEN, A. 2012. Greedy bisection generates optimally adapted triangulations. *Mathematics of Computation 81*, 278, 811–837.

MOUNT, D. M., AND ARYA, S. 1997. ANN: A library for approximate nearest neighbor searching. In *CGC Workshop on Computational Geometry*, 33–40.

NASH, J. 1954. C^1-isometric embeddings. *Annals of Mathematics 60*, 3, 383–396.

PEYRE, G., AND COHEN, L. 2004. Surface segmentation using geodesic centroidal tesselation. In *Proceedings of the 3D Data Processing, Visualization, and Transmission, 2nd International Symposium*, 3DPVT '04, 995–1002.

PEYRE, G., PECHAUD, M., KERIVEN, R., AND COHEN, L. 2010. Geodesic methods in computer vision and graphics. *Foundations and Trends in Computer Graphics and Vision 5*, 3-4, 197–397.

SHAPIRO, P. R., MARTEL, H., VILLUMSEN, J. V., AND OWEN, J. M. 1996. Adaptive smoothed particle hydrodynamics, with application to cosmology: Methodology. *Astrophysical Journal Supplement 103*, 269–330.

SHEWCHUK, J. R. 2002. What is a good linear element? interpolation, conditioning, and quality measures. In *11th International Meshing Roundtable*, 115–126.

SHIMADA, K., AND GOSSARD, D. C. 1995. Bubble mesh: Automated triangular meshing of non-manifold geometry by sphere packing. In *Proceedings of the 3rd ACM Symposium on Solid Modeling and Applications*, 409–419.

SHIMADA, K., YAMADA, A., AND ITOH, T. 1997. Anisotropic triangular meshing of parametric surfaces via close packing of ellipsoidal bubbles. In *6th International Meshing Roundtable*, 375–390.

SIMPSON, R. B. 1994. Anisotropic mesh transformations and optimal error control. *Applied Numerical Mathematics 14*, 1–3, 183–198.

SUN, F., CHOI, Y., WANG, W., YAN, D., LIU, Y., AND LÉVY, B. 2011. Obtuse triangle suppression in anisotropic meshes. *Computer Aided Geometric Design 28*, 9, 537–548.

TURK, G. 1992. Re-tiling polygonal surfaces. In *Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, SIGGRAPH '92, 55–64.

VALETTE, S., CHASSERY, J. M., AND PROST, R. 2008. Generic remeshing of 3D triangular meshes with metric-dependent discrete Voronoi diagrams. *IEEE Transactions on Visualization and Computer Graphics 14*, 2, 369–381.

WITKIN, A. P., AND HECKBERT, P. S. 1994. Using particles to sample and control implicit surfaces. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, ACM, SIGGRAPH '94, 269–277.

YAMAKAWA, S., AND SHIMADA, K. 2000. High quality anisotropic tetrahedral mesh generation via packing ellipsoidal bubbles. In *9th International Meshing Roundtable*, 263–273.

YAN, D., LÉVY, B., LIU, Y., SUN, F., AND WANG, W. 2009. Isotropic remeshing with fast and exact computation of restricted Voronoi diagram. *Computer Graphics Forum 28*, 5, 1445–1454.

ZHANG, M., HUANG, J., LIU, X., AND BAO, H. 2010. A wave-based anisotropic quadrangulation method. *ACM Transactions on Graphics 29*, 4, 118:1–118:8.