Variational implicit surface meshing

Arnaud Gelas, Sébastien Valette, Rémy Prost, Wieslaw L Nowinski

Variational Implicit Surface Meshing

Arnaud Gelas[a,c], Sébastien Valette[b], Rémy Prost[b], Wieslaw L. Nowinski[a]

[a] *Biomedical Imaging Laboratory, Singapore BioImaging Consortium, Biopolis, Singapore*
[b] *CREATIS-LRMN, University of Lyon, INSA-Lyon, CNRS UMR5220, Inserm U630, France*
[c] *Department of Systems Biology, Harvard Medical School, Boston, MA, USA*

Abstract

In this paper, we propose a new algorithm to mesh implicit surfaces which produces meshes both with a good triangle aspect ratio as well as a good approximation quality. The number of vertices of the output mesh is defined by the end-user. For this goal, we perform a two-stage processing : an initialization step followed by an iterative optimization step. The initialization step consists in capturing the surface topology and allocating the vertex budget. The optimization algorithm is based on a variational vertices relaxation and triangulation update. In addition a gradation parameter can be defined to adapt the mesh sampling to the curvature of the implicit surface. We demonstrate the efficiency of the approach on synthetic models as well as real-world acquired data, and provide comparisons with previous approaches.

Key words: implicit surface, meshing, variational approach, restricted Delaunay
PACS: code, code ‘
2008 MSC: code, code

1. Introduction

Implicit modeling is a geometric interface representation where the interface is expressed as the zero level-set of an implicit function. This representation is nowadays used to represent geometrically and topologically complex shapes (see Figure 1, 10). They are also used to represent *constructive solid geometric* objects (referred as CSG objects), *i.e.* objects defined as results of Boolean operations on geometrical shapes, *etc.*

Implicit modeling has received a lot of interest with the recent development of $3D$ acquisition technologies, such as range scanner and LIDAR, *etc*. Indeed, implicit surface based reconstruction methods, like [1, 2], are efficient to overcome topological difficulties in a reasonable time. A brief survey is given in [3]. Implicit modeling is also widely used in medical imaging to segment disconnected components via the *level-sets* formalism [4, 5, 6].

While implicit surfaces are nowadays common for surface modeling, most of the current hardware cannot directly display or process them. Moreover, it is difficult to make any measurement on the shape as long as the geometry is not explicitly known. For these reasons, linear approximations, *i.e.* polygonal meshes are largely preferred. Consequently the conversion from the implicit surface to polygonal meshes, *i.e.* polygonization, has received a lot of interest. The main challenges for a polygonizer are to (i) capture the topology of the implicit surface, (ii) to provide a good approximation of the geometry and normals of the implicit surface, (iii) have a good triangle aspect ratios even in presence of sharp edges. Our objective is so to develop a method which addresses all these problems simultaneously.

1.1. Contributions

We introduce a novel meshing algorithm of implicit surfaces with a user-defined number of points, where the contribution is twofold: (i) we propose a meshing algorithm which captures efficiently the topology and geometry of the given implicit surface, and (ii) an optimization algorithm, which improves the quality of the resulting mesh, both in terms of shape approximation quality and triangle aspect ratio. The initialization algorithm is a Delaunay refinement method guided by two straightforward tests that provides a manifold mesh where the accuracy is fixed via a user defined coefficient. The optimization algorithm, based on vertex relaxation, is driven by variational and approximation quality criteria. The relaxation process deals simultaneously with the problems of implicit surface approximation and mesh quality. The resulting mesh is an accurate approximation of the shape geometry, with good triangle aspect ratios, and is guaranteed to be locally Delaunay conform. Distribution of the mesh vertices can either be uniform or adaptive to the local curvature on the implicit surface via a gradation parameter.

1.2. Related Work

Implicit surface polygonization methods can be classified into four different categories according to the generation principle: *surface tracking*, *tessellation based*, *Morse theory based* and *Delaunay refinement* methods.

Figure 1: Starting from a given implicit function, we first take few random points on the implicit surface; To capture the topology and to construct a coarse approximation of the implicit surface we use a Delaunay refinement; To enhance geometry and triangles quality of the resulting mesh, we use a variational approach. Finally our method provides a mesh which is homeomorphic to the implicit surface, with a very good approximation of its geometry.

Surface tracking methods [7, 8] start from seeds (points or faces) and successively compute the resulting mesh by tracking the surface next to its tangent plane. In [9], Schreiner *et al.* proposed a tracking method guided by the maximum curvature with a minimal set of samples. They assume that the gradient and the Hessian of the implicit function are provided. However, the choice of the seeds is determinant with these tracking approaches. Indeed in the case of multi-component surface, if seeds are not taken carefully some parts may not be meshed at all.

Tessellation based methods first compute a tessellation of the domain, generally a grid based one, and then analyze intersections of the implicit surface with each cell. Following the work of Wyvill *et al.* [10], Lorensen and Cline introduced the very popular marching cubes [11]. It first samples the implicit function on a uniform grid and then analyzes the intersection of the grid edges with the implicit function. While this algorithm is fast and simple to implement, it faces several limitations regarding topological guarantees, difficulty to represent sharp edges, bad triangle aspect ratio, bad vertex degree distribution. For these reasons, some modified versions have been introduced in the last decade to represent sharp edges [12], remove topological ambiguities in the given configurations [13, 14]. Some post-processing methods have also been introduced to improve resulting mesh characteristics, by decimation [15], geometrical adaptation [16], and remeshing [17, 18, 19]. In [20], Karatasheva *et al.* proposed to first extract a base mesh using [21], then apply a similar optimization to [16] in order to first polygonize the implicit surface before generating a volume mesh suitable for finite element analysis. Some other methods use an octree based decomposition of the domain [22], and/or add one vertex per cells which intersects the implicit surface [23]. Note that the location of this vertex can be optimized to effectively represent sharp edges [24].

By computing intermediate level sets and changing the topology of the mesh when critical points are encountered, Stander and Hart [25] proposed a solution to the problem of homeomorphic polygonization of implicit surfaces. Later, Boissonnat *et al.* [26] provided an algorithm which updates an octree data structure (further subdivided into a tetrahedral mesh) according to smooth and discrete *Morse theory*, and extracts the complex from tetrahedra which have vertices with different signs. Under the assumption that critical points are provided, this algorithm ensures that the resulting mesh and the implicit surface are isotopic; however this last algorithm does not provide an accurate approximation of the surface.

Delaunay refinement methods, first introduced for meshing 2D domains, have been extended to $3D$ surfaces. Chew described in [27] a *furthest point* based algorithm to construct a *restricted Delaunay triangulation* of the surface by iteratively inserting the furthest intersection of all Voronoi edges and the surface. Cheng *et al.* [28] were the first one to add the *topological ball* criterion [29] in their refinement for a specific type of smooth surface called skin surface. Boissonnat and Oudot [30] introduced criteria to guarantee the resulting mesh to be homeomorphic to the surface, without using the topological ball property. They assume that the local feature size can be easily computed, or is provided. Unfortunately, the computation of the local feature size is not easy and remains computationally expensive. In [31], the authors ensure the homeomorphism between the surface and the resulting mesh via a Delaunay refinement method driven by the Morse theory, and the topological disk constraint. Note that these two last methods [30, 31] introduced and discussed the quality of the resulting mesh by the use of geometrical refinement. Even if these methods have theoretical guarantees, they can not efficiently capture sharp edges. Recently, Dey and Levine [32] presented a software called DelPSC which has the capability of meshing non-smooth surfaces when the feature edges are provided by the end-user.

2. Background

2.1. Restricted Delaunay Triangulation

Let $\mathcal{P}$ be a finite set of points in $\mathbb{R}^3$.

Voronoi Diagram. The *Voronoi cell* of the *site* $\mathbf{p}_i \in \mathcal{P}$ is given as $\mathcal{V}_i = \left\{\mathbf{x} \in \mathbb{R}^3 : \forall\, \mathbf{q} \in \mathcal{P},\ \|\mathbf{x}-\mathbf{p}_i\| \leqslant \|\mathbf{x}-\mathbf{q}\|\right\}$. The *Voronoi diagram* $\mathsf{Vor}\ \mathcal{P}$ of $\mathcal{P}$ is the collection of all Voronoi cells and their intersections (see Figure 2(b)). Note that faces shared by

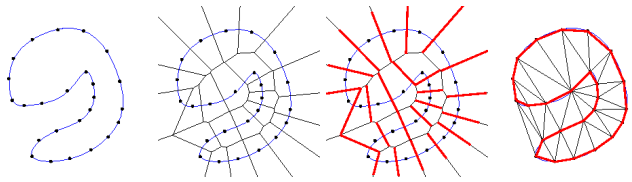


Figure 2: Restricted Delaunay triangulation for a curve in the plane. Given points on a curve in the plane (a). Voronoi diagram of this set of points (b). Restricted Voronoi edges to the blue curve, *i.e.* intersection of Voronoi edges and the curve, are represented by red segments (c). Corresponding restricted Delaunay triangulation and Voronoi diagram (d) represented by a thick red line.

two Voronoi cells are called *Voronoi faces*, and edges shared by three Voronoi cells are called *Voronoi edges*.

Delaunay Triangulation. The *Delaunay Triangulation* $\mathsf{Del}\ \mathcal{P}$ of a set of points $\mathcal{P}$ is dual to the Voronoi diagram $\mathsf{Vor}\ \mathcal{P}$. Note that the computational complexity of building a Delaunay triangulation of N points in 3D is of $O(N^2)$.

Restricted Delaunay Triangulation. Let $\mathcal{P}$ be a point set on a surface $\mathcal{S}$. Any Voronoi face $\mathcal{V}_\sigma \in \mathsf{Vor}\ \mathcal{P}$ which intersects the surface, *i.e.* $\mathcal{V}_\sigma \cap \mathcal{S} \neq \emptyset$, is called a *restricted Voronoi face* (see Figure 2(c)). The *restricted Voronoi diagram* $\mathsf{Vor}_\mathcal{P}|\mathcal{S}$ is the collection of all restricted Voronoi faces and the *restricted Delaunay triangulation*, $\mathsf{Del}_\mathcal{P}|\mathcal{S}$, consists of the dual Delaunay simplices of the restricted Voronoi faces, that is, $\mathsf{Del}_\mathcal{P}|\mathcal{S} = \{\sigma \in \mathsf{Del}_\mathcal{P} \mid \mathcal{V}_\sigma \cap \mathcal{S} \neq \emptyset\}$ (see Figure 2(d)).

Topological ball property. A restricted Voronoi diagram satisfies the *topological ball property* if each k dimensional Voronoi face either does not intersect the surface, *i.e.* $\mathcal{V}_\sigma \cap \mathcal{S} = \emptyset$, or intersects it in a $(k-1)$-ball. In [29], Edelsbrunner and Shah show that if $\mathsf{Vor}_\mathcal{P}|\mathcal{S}$ satisfies the topological ball property then $\mathsf{Del}_\mathcal{P}|\mathcal{S}$ is homeomorphic to $\mathcal{S}$. Note that if the sampling of $\mathcal{S}$ is "sufficiently dense", this condition is satisfied and the restricted Delaunay triangulation is an excellent approximation of the surface with theoretical bounds on the geometrical error and normal deviation (see [28, 30]).

2.2. Centroidal Voronoi Tessellation

2.2.1. Planar case

In the planar case, a *centroidal Voronoi tessellation* (CVT) is a particular tiling of the plane whose Voronoi sites are the centroids (centers of mass) of its corresponding Voronoi regions (see Figure 3). The centroid $\overline{\mathbf{p}}_i$ of a Voronoi region $\mathcal{V}_i$ is computed as:

$$\overline{\mathbf{p}}_i = \frac{\int_{\mathcal{V}_i} \mathbf{x} \cdot \chi(\mathbf{x})\, \mathrm{d}\mathbf{x}}{\int_{\mathcal{V}_i} \chi(\mathbf{x})\, \mathrm{d}\mathbf{x}} \tag{1}$$

where $\chi(\mathbf{x})$ is a given density function.

CVT construction is of great interest in many applications. One can refer to [33] for more details about CVT. There are various methods to achieve the goal of building such tessellation.

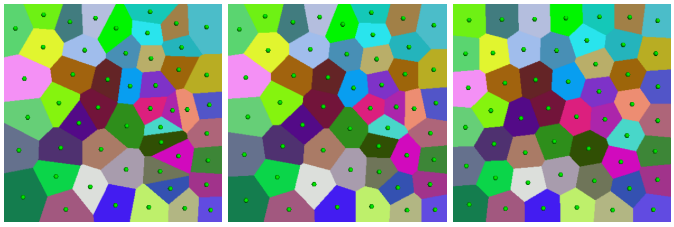


Figure 3: Lloyd algorithm on a square domain at various steps: initial Voronoi sites and Voronoi diagram (left), after 2 iterations (middle), after 50 iterations (right).

One can consider the minimization of the following functional:

$$E = \sum_{i=1}^{N} \int_{\mathcal{V}_i} \chi(\mathbf{x}) \cdot d(\mathbf{x}, \overline{\mathbf{p}}_i)^2\, \mathrm{d}\mathbf{x} \tag{2}$$

where d is a distance in the plane. It can easily be shown that for a given set of sites, the functional is minimized when each region $\mathcal{V}_i$ is the Voronoi cell of its site, and for a given set of regions, the sites minimize the functional when they are the centers of mass of their Voronoi region.

Several methods have been introduced to produce such tessellation, the most known and used is the *Lloyd algorithm* [34]. See Figure 3 for a simple example on the square. Given an initial configuration, *i.e.* initial sites $\{\overline{\mathbf{p}}_i\}_{i=1}^N$ and initial regions $\{\mathcal{V}_i\}_{i=1}^N$, the algorithm iteratively updates $\overline{\mathbf{p}}_i$ and $\mathcal{V}_i$.

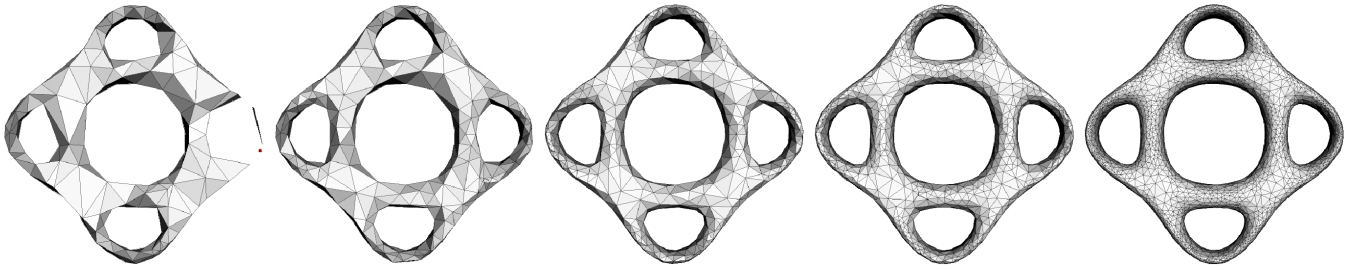
2.2.2. 3D surface mesh case

While CVTs are well-defined on the plane, various definitions and implementations have been proposed for $3D$ surface meshes, mainly for remeshing or surface segmentation purpose [17, 18, 19]. In [18], Alliez *et al.* use the Lloyd algorithm in parametric space. This method suffers from the need to compute a mesh parameterization at a first stage which creates distortions. In [17], the authors avoid the computation of the mesh parameterization by using local parameterization instead, *i.e.* for each vertex the relaxation is performed onto a parameterization of its 0-ring. In [35], Valette and Chassery introduced a fully discrete algorithm for CVT construction.

3. Variational Isotropic Meshing

For the sake of clarity, we consider an implicit function f, with $\mathcal{S}$ its corresponding implicit surface (a 2-manifold) on a given finite domain $\Omega \subset \mathbb{R}^3$. Note that we only consider implicit surface $\mathcal{S}$ without boundaries, and implicit function f where we can compute for any point $\mathbf{x} \in \Omega$ its value $f(\mathbf{x})$ and gradient $\nabla f(\mathbf{x})$.

We propose to compute high quality meshes from a given implicit surface $\mathcal{S}$ in two phases. First, we compute an initial mesh (see Section 3.1) which is homeomorphic to the implicit surface and approximates well its geometry via Delaunay refinement. Then, after a geometrical refinement to reach the number



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

Figure 4: Output of the Delaunay refinement process for PRETZEL5 implicit surface for decreasing face distance condition bounds $L = \lambda \cdot d$. Decreasing λ value, increases the number of points N_P and triangles N_T: {λ: 0.2, N_P: 234, N_T: 474} (a), {λ: 0.1, N_P: 398, N_T: 812} (b), {λ: 0.05, N_P: 750, N_T: 1516} (c), {λ: 0.025, N_P: 1380, N_T: 2776} (d), {λ: 0.01, N_P: 3831, N_T: 7678} (e). As expected, with too large bound L regarding to the local feature size, the output of the Delaunay refinement does not capture the right topology (a).

of points fixed by the user, we optimize the mesh by using a relaxation algorithm (see Section 3.2) which in the same time improves the sample distribution on the surface and the approximation quality of the resulting Delaunay conform mesh.

3.1. Initialization

The first step of our method is to build an initial mesh which is homeomorphic to the implicit surface and provides a good approximation of the surface. Our initialization can be decomposed into two different phases. Some random points are first projected onto the surface by a Newton descent method. Then, we compute the corresponding restricted Delaunay triangulation (see Section 2.1). We then attempt to capture the topology of the implicit surface on the given domain by applying Delaunay refinement (see Section 3.1.1), and to improve the geometrical approximation and characteristics of the resulting mesh by applying geometrical refinement adapted to the user requirements (see Section 3.1.2).

3.1.1. Delaunay refinement

We propose a straightforward refinement method with only two tests, in spirit with [36]. The first test is to check the topological disk condition, this proves that the resulting mesh is manifold. The second test is to measure the distance between the resulting mesh and the implicit surface, *i.e.* we refine the mesh until this distance is below a given parameter L for all facets. Note that this test is closely related to the geometrical approximation error, and will determine the topology of the resulting mesh (see Figure 4). The parameter L is directly linked to the size of the object to be meshed, via a user-defined coefficient λ. The refinement process is repeated until both conditions hold for all facets and all points of the resulting mesh.

Face distance condition. For any facet $\sigma \in \mathsf{Del}_{\mathcal{P}}|\mathcal{S}$, we define $DF(\sigma)$ as the distance between the circumcenter of σ and the furthest intersection $\mathbf{q}$ of the dual V_σ (Voronoi edge) with the implicit surface $\mathcal{S}$. In our refinement process, we fix a bound L on $DF(\sigma)$, *i.e.* we refine until all facets σ satisfy:

$$DF(\sigma) < L = \lambda \cdot d \tag{3}$$

where d is the shortest edge of the bounding box of the domain Ω, and $0 < \lambda < 1$ is a user-defined coefficient. If this condition is violated, we insert the corresponding intersection $\mathbf{q}$, *i.e.* intersection between the dual of σ and the surface into $\mathsf{Vor}_{\mathcal{P}}|\mathcal{S}$ and update it.

Topological disk condition. Since we consider manifold implicit surfaces, the output should satisfy the topological disk condition, *i.e.* all facets $\mathcal{N}_{\mathbf{p}}$ incident to a vertex $\mathbf{p}$ should form a topological disk. If this condition is not fulfilled, we find the facet $\sigma \in \mathcal{N}_{\mathbf{p}}$ which maximizes $DF(\sigma)$. Then we insert the corresponding intersection $\mathbf{q}$, *i.e.* intersection between the dual of σ and the surface into $\mathsf{Vor}_{\mathcal{P}}|\mathcal{S}$ and update it.

3.1.2. Geometrical Refinement

At this point, we assume that the resulting mesh $\mathcal{M}$ is homeomorphic to the implicit surface $\mathcal{S}$. If $\mathcal{M}$ has enough vertices regarding to the user requirements, we skip this part; else new vertices must be added. This can be achieved by using Delaunay refinement. However, this operation remains computationally expensive, because of the need to maintain a 3D Delaunay triangulation during the refinement process, thus we prefer to process the refinement directly with the surface mesh $\mathcal{M}$.

The geometrical refinement is guided by the user objective. In the uniform case the geometrical refinement should distribute the triangle area on the mesh as uniformly as possible. In the curvature-adapted case, the geometrical refinement should distribute the normal variation per triangle on the mesh as uniformly as possible. This is done by using one priority queue in which we push each triangle t with the priority P_t defined as:

$$P_t = |t| \cdot \left(\left(9 - \|\mathbf{n}_1 + \mathbf{n}_2 + \mathbf{n}_3\|^2\right)^\gamma + \epsilon\right) \tag{4}$$

where $|t|$ is the area of t, $\mathbf{n}_k$ is the estimated normal for the k^{th} vertex of t, *i.e.* $\mathbf{n}_k = -\frac{\nabla f(\mathbf{p}_k)}{\|\nabla f(\mathbf{p}_k)\|}$; γ is a gradation parameter allowing uniform sampling ($\gamma = 0$) and curvature-adaptive sampling ($\gamma > 0$). ϵ is an arbitrary small value used as an offset in order to deal correctly with perfectly flat regions (in our experiments, we set $\epsilon = 10^{-50}$). Intuitively, P_t gives an objective measure of normal variation inside the triangle, and then provides a good indicator of curvature.

Afterwards, we pop the triangles with the highest priority P_t from the queue. Each time a triangle t is popped:

- a new vertex $\mathbf{q}_t$ is added, located at the circumcenter of t,
- t is split into three new triangles, made of the three original vertices of t plus $\mathbf{q}_t$,
- the triangulation is locally updated by means of edge flips in order to restore the local Delaunay property (see section 3.2.3),
- the priority queue is updated in order to take into account the three new triangles of the mesh.

This way, new vertices are inserted until the desired number of vertices is obtained.

3.2. Relaxation

Our relaxation algorithm is driven by two objectives:

- Isotropic sampling of the surface: we want a uniform (possibly curvature-adapted) sampling of the surface.
- Approximation quality: we want a good approximation of the implicit surface in terms of the Haussdorf distance.

Those objectives are closely related: isotropic sampling involves control of the vertex positions in their respective parametric space (*i.e.* the local tangent plane) and approximation quality can be improved by moving the vertices in their normal direction.

We propose a two-step relaxation algorithm which takes into account both constraints. At each relaxation iteration, isotropic sampling is achieved by *approximating a geodesic CVT* with a Lloyd-like relaxation step, then approximation is enhanced by means of a *Quadric Error Metrics* (QEM)-based relocation strategy. After these two relaxation operations, a possible update of the triangulation is processed to maintain a local Delaunay-conforming mesh criterion, used for the computation of our geodesic Voronoi diagram approximation. The overall quality of the mesh increases with the number of relaxation iterations both in terms of shape approximation quality (geometry and normal approximation) and triangle aspect shape ratio distribution.

3.2.1. Isotropic sampling: CVT for Implicit Surfaces

CVT have been introduced for surfaces only for the case of 3D surface meshes (see Section 2.2.2). Note that Ohtake and Belyaev [16] proposed an approach to optimize the meshing of implicit surfaces, by means of vertices relaxation, but this approach operates at fixed connectivity. Instead we propose to approximate a geodesic CVT for implicit surfaces leading to an isotropic mesh.

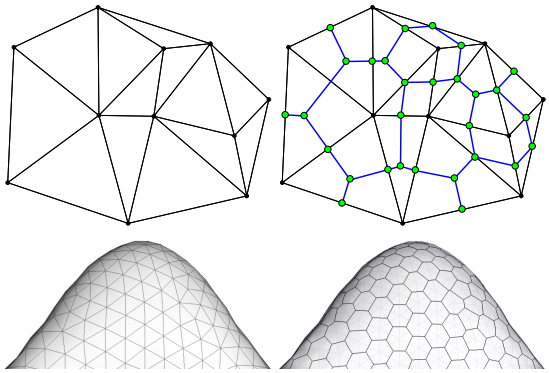


Figure 5: Approximation of Voronoi cells: the planar (top) and 3D mesh (bottom) are shown on the left, and their approximated Voronoi diagram on the right

Functional minimization. We adapt the minimization of the functional given in Equation 2 for the case of implicit surfaces. Following our statement that the implicit surface $\mathcal{S}$ and the mesh $\mathcal{M}$ are homeomorphic, the Euclidean distance must not be considered since it may change the topology of the resulting mesh. For this reason, we decide to deal with the geodesic distance instead and minimize the following functional:

$$E_i = \int_{\mathbf{x}\in\mathcal{V}_i} \chi(\mathbf{x}) \cdot d(\mathbf{x}, \overline{\mathbf{p}}_i)^2 \, \mathrm{d}\mathbf{x} \tag{5}$$

where $d(\mathbf{x}, \overline{\mathbf{p}}_i)$ is the geodesic distance between $\mathbf{x}$ and $\overline{\mathbf{p}}_i$, $\mathcal{V}_i$ is the geodesic Voronoi cell on $\mathcal{S}$.

Unfortunately, since the surface is implicitly defined, computation of the geodesic distance requires assumptions on the surface curvature, and explicit knowledge of the geometry. In practice, the computation of geodesic distance can be performed via finely meshing the implicit surface. But this method is obviously related to the size of the mesh and may be time and memory intensive. For this reason, we will not consider a dense mesh, but only the dual of the geodesic Voronoi diagram, *i.e.* a Delaunay triangulation, and consider an approximation of the the previous functional.

Approximation of geodesic Voronoi cell. The geodesic Voronoi cell $\mathcal{V}_i$ is theoretically defined by $\mathcal{V}_i = \{\mathbf{x} \in \mathcal{S} : \forall\, \mathbf{q} \in \mathcal{P} - \{\mathbf{p}\},\ d(\mathbf{x}, \mathbf{p}) \leqslant d(\mathbf{x}, \mathbf{q})\}$ with $d(\cdot, \cdot)$ is the geodesic distance on $\mathcal{S}$.

Let us consider a mesh $\mathcal{M}$ which satisfies the Delaunay criterion. Then, we approximate the geodesic Voronoi cell by piecewise linear approximations V_i (see Figure 5(a-b)). For a given vertex $\mathbf{p}$, its Voronoi cell is approximated by a set of triangles around this vertex. If the vertex has n neighboring vertices, the approximation will consist of $2n$ triangles. The boundary of this approximation is a polygon with vertices being alternatively $\mathbf{p}$'s neighbour triangle circumcenters and $\mathbf{p}$'s neighbour edge midpoints. Note that in cases where a given triangle t contains obtuse angles, the circumcenter $\mathbf{c}_t$ of t will be outside t. In this case, we constrain $\mathbf{c}_t$ to lie within the boundary of t to maintain consistency between neighboring Voronoi cells, and as a consequence the considered Voronoi cell approximation may contain less than $2n$ triangles.

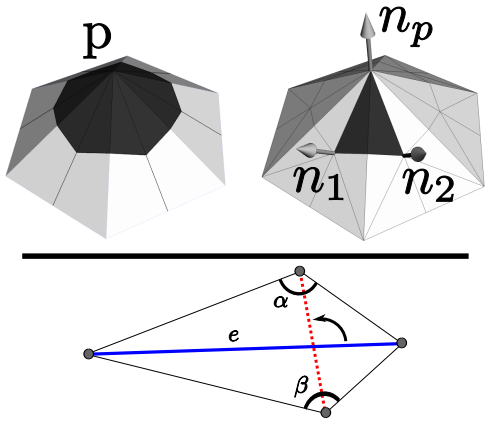


Figure 6: Top : Close-up view of a vertex ring and its approximated Voronoi cell (left), and the normals used in the weight χ computation of the two greyed triangles (right). Bottom : an edge e is non local Delaunay if the sum of the angles opposite to e on the adjacent faces to e is above π ($\alpha + \beta > \pi$). In such case e should then be flipped.

Functional Reformulation. The previous assumption, *i.e.* that the geodesic Voronoi cell is approximated by piecewise linear cells, naturally leads to simplification of the expression of the geodesic distance. Indeed in such a case, the geodesic distance becomes the Euclidean distance in each linear approximation cell. Finally, we can rewrite the functional in Equation 5 by:

$$E_i = \int_{\mathbf{x}\in V_i} \chi(\mathbf{x}) \cdot \left\|\mathbf{x} - \overline{\mathbf{p}}_i\right\|^2 \mathrm{d}\mathbf{x} \tag{6}$$

Vertex relaxation. During the relaxation of a vertex, the first step is to change its position in order to minimize the functional in Equation 6. This improves the local sampling quality with respect to sampling adaptivity and triangle shape aspect ratio. This is simply done by moving the vertex to its Voronoi cell barycenter. Moreover, we constraint each displacement to stay inside the local tangent plane, as moving the vertex in the normal direction is done with an approximation-aware relocation scheme (see next section). The following equation gives the new vertex position $\mathbf{p}_i^{t+1}$ according to the vertex previous position $\mathbf{p}_i^t$ and the corresponding Voronoi cell barycenter $\overline{\mathbf{p}}_i^t$:

$$\mathbf{p}_i^{t+1} = \overline{\mathbf{p}}_i^t - \langle \overline{\mathbf{p}}_i^t - \mathbf{p}_i^t, \mathbf{n}_i^t \rangle \cdot \mathbf{n}_i^t \tag{7}$$

where $\langle \cdot, \cdot \rangle$ denotes the usual dot product. The computation of the Voronoi cell barycenter is easily computed by splitting it into several triangles.

Curvature-adaptivity is easily achieved by associating individual weights to each triangle element inside the Voronoi cell. The bottom of Figure 6(top) shows two triangles adjacent to the same mesh edge. For the Voronoi cell barycenter computation, these two triangles will be given the weight χ defined as:

$$\chi = \left(9 - \left\|\mathbf{n}_\mathbf{p} + \mathbf{n}_1 + \mathbf{n}_2\right\|^2\right)^\gamma + \epsilon \tag{8}$$

where $\mathbf{n}_\mathbf{p} = -\frac{\nabla f(\mathbf{p})}{\|\nabla f(\mathbf{p})\|}$ is the estimated normal at $\mathbf{p}$ on the implicit surface, $\mathbf{n}_1$ and $\mathbf{n}_2$ are the estimated normals on the adjacent triangles barycenter projected on the surface (by means of Newton's algorithm); ϵ and γ have already defined in Equation 4.

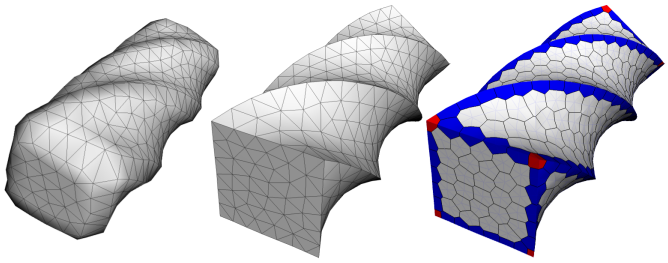


Figure 7: Simple projection onto the implicit surface (left) does not allow accurate features representation, whereas QEM-based relocation (center) allows to efficiently represent this twisted brick with few vertices with the approximation of Voronoi cells painted in white for planar regions, blue for sharp edges and red for corners (right). For the initialization step, we used $\lambda = 0.1$.

Note that here, considering the triangle circumcenters seems more natural. But in case of degenerate triangles, their circumcenter would be restricted to stay on the mesh edges, where sharp features are generally located. Normal estimation can lead to inconsistent results in the presence of sharp features. Then, Equation 8 would be inaccurate if it involved normal estimation on feature points. As a consequence, we use the triangles centers of mass as estimation location, as they are generally far from any sharp feature.

3.2.2. *Approximation-effective relocation in the normal direction*

While Lloyd relaxation is effective for relaxation in the planar case, it is much less efficient for our goal, since we operate in $3D$. Indeed, moving every vertex towards its Voronoi cell barycenter will result in poor approximation quality. We propose to relocate the vertex position using Quadric Error Metrics (QEM) [15]. We compute the QEM Matrix of each triangle surrounding the considered vertex, and relocate the vertex according to the sum of these matrices. Note that for the computation of one QEM matrix, one needs the position and the normal direction. For each considered triangle, the chosen position is the triangle barycenter projected on the surface (using Newton's algorithm), and the chosen normal direction is the gradient of f at this position. More importantly, QEM-based relocation allows us to easily detect whether a vertex is located on a feature or not. Indeed, during relocation, as explained in [37], the corresponding QEM matrix $M_\mathbf{p}$ is analyzed by means of Singular Value Decomposition (SVD). Its eigenvalues are sorted by increasing order of magnitude, and only the most significant values are considered. In this paper, we discarded any singular value smaller than one twentieth of the largest singular value. Intuitively, one can consider that when only one eigenvalue is kept, the vertex is on a relatively planar region. Two significant singular values indicate a sharp edge, and three significant eigenvalues indicate a corner. Figure 7 clearly show the advantages of QEM over a simple projection when meshing a twisted brick. The model on the left was created using only projections on the surface, while the model in the middle was created using QEM-based relocation. The right figure shows the approximation of the Voronoi diagram on the mesh, with regions colored in blue when two eigenvalues where used, and red when three

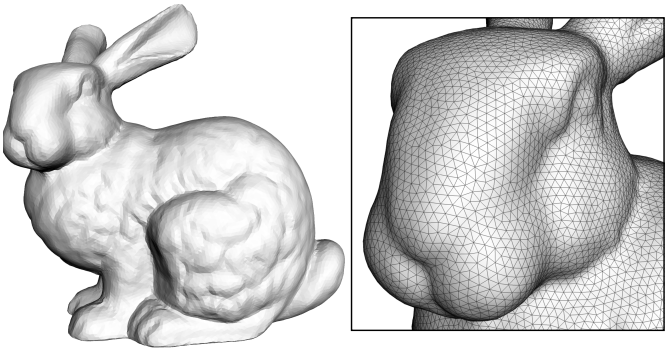


Figure 8: Uniform meshing with 27k vertices of the Stanford Bunny reconstructed with MPU [2].

eigenvalues were used. Note that when two or three eigenvalues are used, the vertex is not only displaced along the normal direction, but also along the tangent plane.

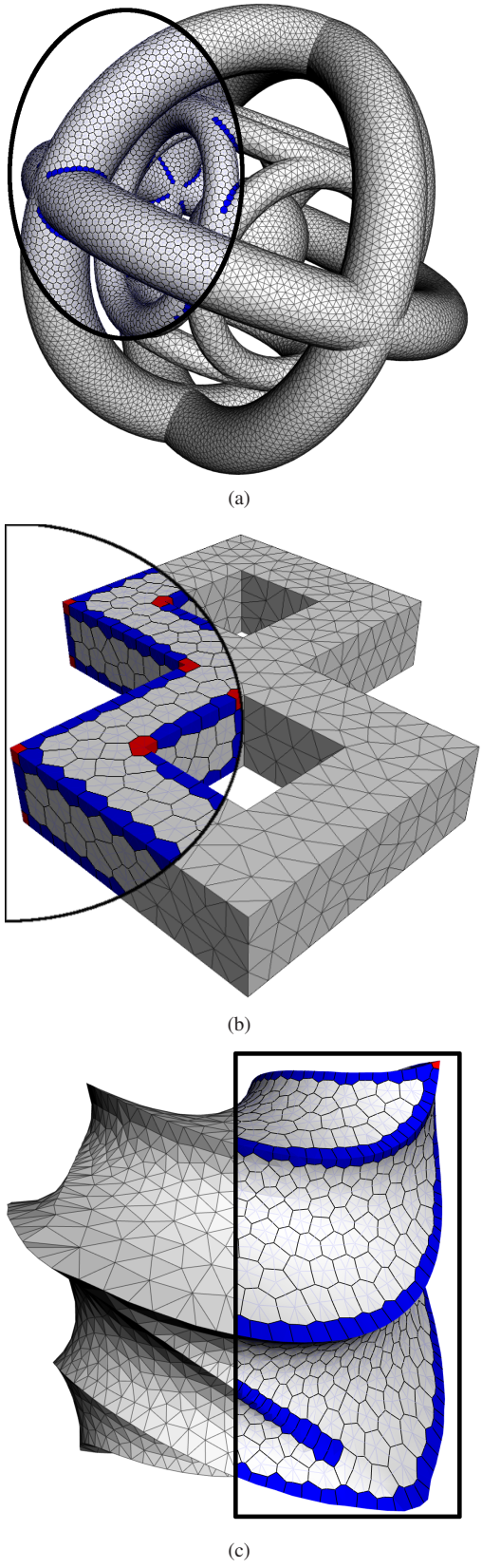
3.2.3. Triangulation update

The connectivity of the mesh resulting from the previous relaxation is updated to locally satisfy the Delaunay criterion by following [38], *i.e.* by removing all *non local Delaunay* edges. An edge e is *non local Delaunay* if the sum of the angles opposite to e on the adjacent faces to e is above π (see Figure 6(bottom)). The non local Delaunay edges are removed by flipping all edges according to a priority queue. The priority of an edge e is given by the sum of the angles opposite to e minus π. The process is then repeated until all the edges are Delaunay edges, and its convergence is guaranteed.

4. Results

We have tested our algorithm on a wide range of implicit surfaces, *i.e.* on typical algebraic surfaces used in the literature (see Figure 4, 11, 10), on CSG models available (see Figure 7, 9), and on some implicit surfaces generated from point sample models by using the Multilevel Partition of Unity (MPU) [2] (see Figure 1, 8).The user can control the meshing via few parameters: the number of points of the output mesh N, the coefficient λ in the initialization stage and the curvature gradation via the parameter γ. In our experiments, we used $\lambda = 0.01$ unless another value is clearly specified. The gradation parameter γ controls the sampling of the resulting mesh (see Figure 11). With $\gamma = 0$, the resulting mesh is uniformly sampled (see Figure 8); whereas higher γ values increase the samples density in high curvature region on the surface.

Figure 7 and 9 demonstrate the effectiveness of our method to represent sharp edges for several CSG models. Note that for the twisted octahedron (figure 9.(c)), we used $\gamma = 2$ to obtain a curvature-adapted mesh. The vertex budget is well distributed on the smooth parts, but the vertex distribution on sharp features is approximately twice as dense than on smooth regions. This artifact is caused by the ill-definition of the gradient for sharp features. We expect such problems to be solved with a high-level handling of the features, by locally evaluating the QEM



(a)

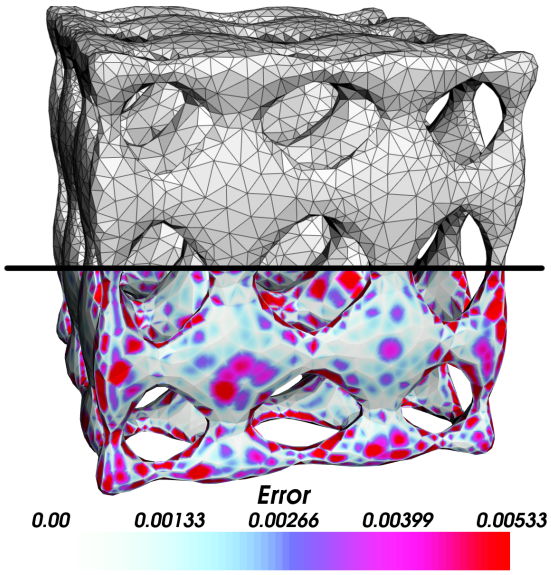
(b)

(c)

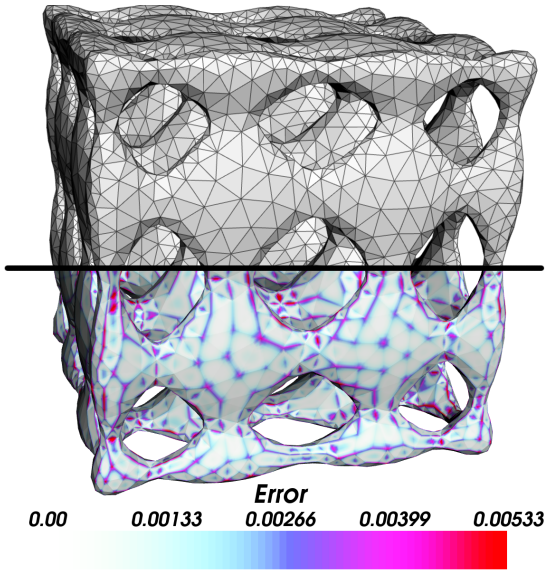
Figure 9: (a) the Earth surface, (b) 2Torus surface (with $\lambda = 0.1$) and (c) the twisted octahedron (with $\gamma = 2$) meshed with our approach, with some parts showing the underlying Voronoi diagram approximation

Surface	Method	$\|v\|$	e_{max} ($\times 10^{-2}$)	e_{RMS} ($\times 10^{-3}$)	$\angle_{min}$ (°)	$\angle_{min,Av}$ (°)	Initialization (s)	Optimization (s)	Total Time (s)
Twisted O.	[30]	24583	3.21	2.04	30	46	-	-	15.1
Twisted O.	[*Ours*]	24389	1.13	0.35	25	51	1.5	9	10.5
Fancy88	[30]	8861	2.64	3.04	30	46	-	-	4.5
Fancy88	[*Ours*]	8000	1.19	2.00	15.8	45	2.1	3.3	5.4
Fancy88	[16]	56298	0.16	0.33	11.5	41	-	-	-
Fancy88	[*Ours*]	54872	0.25	0.32	23.6	49	6.2	12.1	18.3
Earth	[*Ours*]	27000	3.15	2.6	4.5	51.8	2.9	19.9	22.8

Table 1: Objective measures : meshing errors (e_{max} and e_{RMS}), minimal ($\angle_{min}$) and average minimal ($\angle_{min,Av}$) angles and computation times, for Boissonnat and Oudot's approach [30], Ohtake and Belyaev's approach[16] and ours for the Twisted Octahedron (see Figure 13), the Fancy88 model model (see Figure 10 and 12), and the earth model (figure 9.(a)). The last column shows the global computing times, split into Initialization and optimization steps for our approach.



(a) Boissonat & Oudot



(b) Ours

Figure 10: Visual and Objective quality comparison between the approach of Boissonnat and Oudot [30] (top) and ours (bottom), for Fancy88.

projection relevance and by restricting edge flips for sharp features. To measure the approximation quality of our meshing method, we measured the approximation of the Euclidean distance $e(\mathbf{p})$ between the output mesh $\mathbf{p}$ and the implicit surface $\mathcal{S}$ [39] given by :

$$e(\mathbf{p}) \approx \frac{|f(\mathbf{p})|}{\|\nabla f(\mathbf{p})\|} \tag{9}$$

To globally increase the accuracy of the error computation, we subdivide each triangle three times, which increases the number of error sampling points. Table 1 shows numerical values of the maximum error e_{max}, RMS error e_{RMS}, the output meshes minimal and average minimal angles, and the computing times (for both the initialization step and the relaxation step), measured on a 4-core CPU running at 2.6 Ghz, compared with [16] and [30]. In these examples, 50 relaxation iterations were performed.

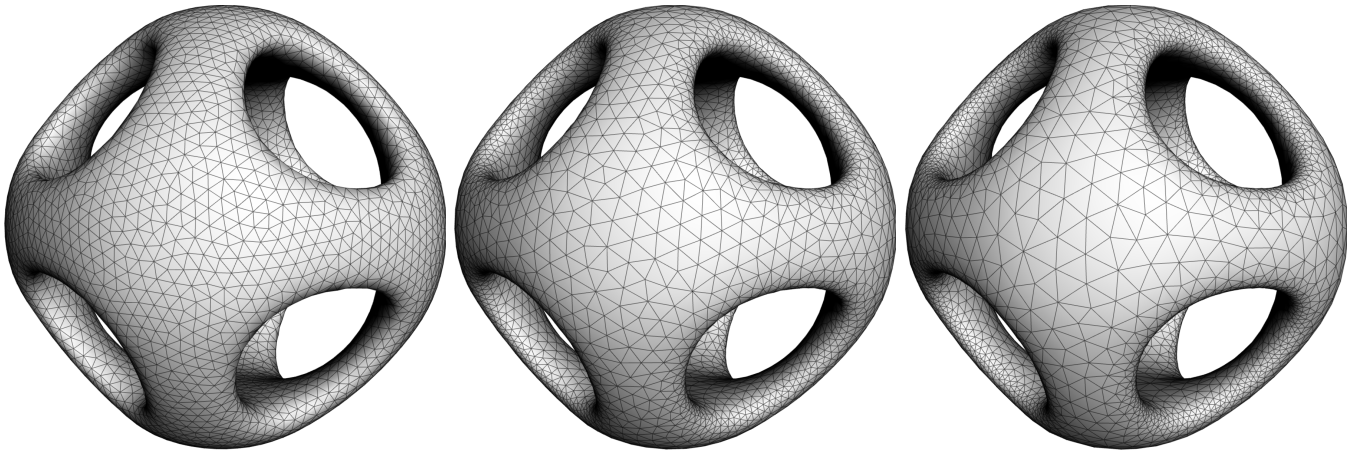
In Figure 13-10, we compare our method with [30] for surfaces with sharp edges and smooth surfaces, for an equivalent number of vertices. Our method provides much better approximation in high curvature regions and efficiently represents sharp edges. For the twisted octahedron (figure 13), the maximal error for our approach is in a region where a vanishing feature is present and was not detected by the QEM scheme (like the case shown in figure 9.(c)).

In Figure 12, we visually compare our method with [16]. As both approaches use quadrics to enhance approximation quality, they yield similar approximation errors, but the mesh created with our approach clearly exhibits a better isotropy.

5. Discussion & Conclusion

Our approach aims at creating high quality meshes and results indicate that our approach outperforms recent algorithm in terms of approximation quality and is of a greater interest for offline meshing applications.

An advantage of our approach is its very low memory footprint, as no specific data is stored during the mesh optimization, the processing operates directly on the output mesh. Moreover, this algorithm is easy to implement in a parallel fashion, to efficiently exploit workstations running with multi-core processors. In the future, we plan to extend this work to variational anisotropic meshing, to further enhance the approximation quality of the produced models.



(a) $\gamma = 0$ (b) $\gamma = 1$ (c) $\gamma = 2$

Figure 11: Orthocircle model with 5K points for various gradation values γ.

Acknowledgments

The Stanford Bunny and Lucy model are courtesy of the Stanford Computer Graphics Laboratory. Our algorithm was coded in c++, on top of the CGAL (www.cgal.org) and VTK (www.vtk.org) libraries.
The authors would like to thank Yutaka Ohtake and Steve Oudot for providing models for comparisons, Laurent Guigues for his help with the implementation, and Julien Dardenne for providing the images of 2D Lloyd relaxation.

Finally, the authors thank the anonymous reviewers both for their helpful comments and suggestions. Their criticisms helped to improve the quality of this paper.

This work was partly funded by a grant from the NHGRI (P50HG004071) to found the Center for in toto genomic analysis of vertebrate development.

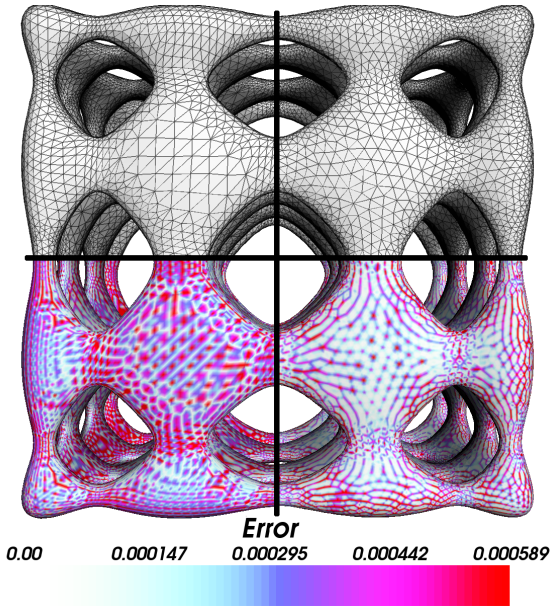


Figure 12: Visual comparison between the approach of Ohtake and Belyaev [16] (left part) and ours (right part) for $\approx 55k$ points.

References

[1] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, T. R. Evans, Reconstruction and representation of 3d objects with radial basis functions, in: ACM SIGGRAPH, 2001, pp. 67–76.

[2] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, H.-P. Seidel, Multi-level partition of unity implicits, ACM Transaction on Graphics 22 (3) (2003) 463–470.

[3] O. Schall, M. Samozino, Surface from scattered points: A brief survey of recent developments, in: In 1st International Workshop on Semantic Virtual Environments, Villars sur Ollon, Switzerland, 2005, pp. 138–147.

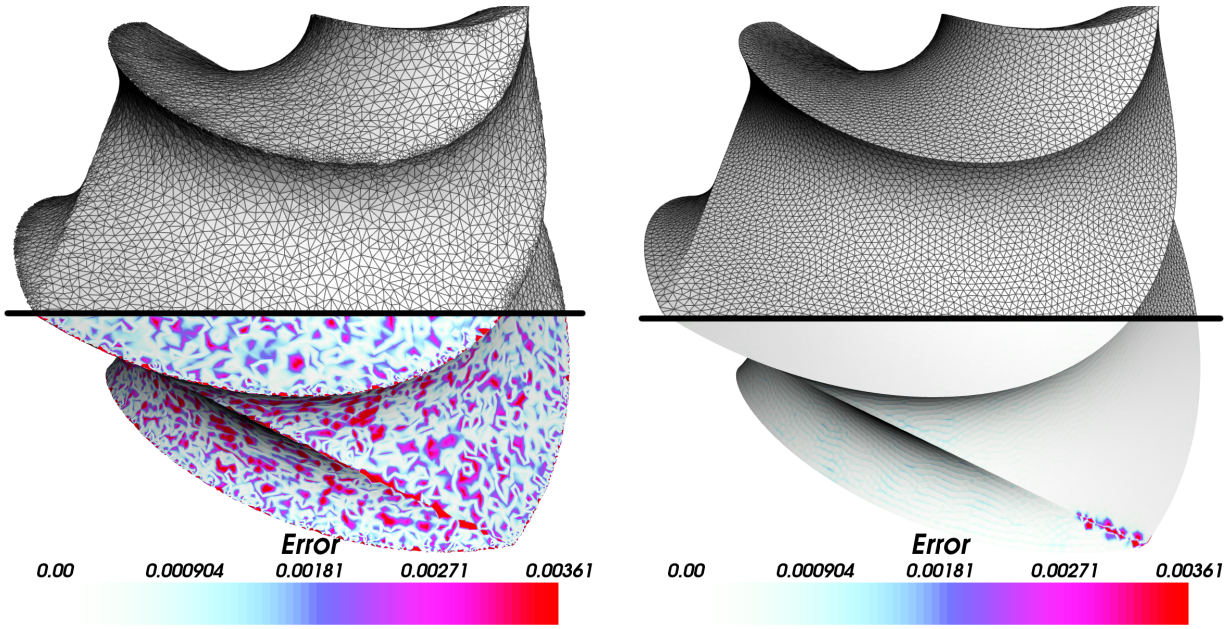
[4] S. Osher, R. Fedkiw, Level Set Methods and Dynamic Implicit Surfaces, Springer, 2002.

[5] A. Gelas, O. Bernard, D. Friboulet, R. Prost, Compactly supported radial basis functions based collocation method for level-set evolution in image segmentation, IEEE Transactions on Image Processing 16 (7) (2007) 1873–1887.

[6] K. Mosaliganti, L. Cooper, R. Sharp, R. Machiraju, G. Leone, K. Huang, J. Saltz, Reconstruction of cellular biological structures from optical microscopy data, IEEE Transactions on Visualization and Computer Graphics 14 (4) (2008) 863–876.

[7] A. Hilton, A. J. Stoddart, J. Illingworth, T. Windeatt, Marching triangles: Range image fusion for complex object modelling, in: International Conference on Image Processing, Vol. 2, 1996, pp. 381–384.

[8] S. Akkouche, E. Galin, Adaptive implicit surface polygonization using marching triangles, Computer Graphics Forum 20 (2) (2001) 67–80.



(a) Boissonat & Oudot (b) Ours

Figure 13: Visual and Objective quality comparison between the approach of Boissonnat and Oudot [30] (top) and ours (bottom), for the Twisted Octahedron.

[9] J. Schreiner, C. Scheidegger, C. Silva, High-quality extraction of isosurfaces from regular and irregular grids, IEEE Transactions on Visualization and Computer Graphics 12 (5) (2006) 1205–1212.

[10] G. Wyvill, C. McPheeters, B. Wyvill, Data Structure for Soft Objects, The Visual Computer 2 (4) (1986) 227–234.

[11] W. E. Lorensen, H. E. Cline, Marching cubes: A high resolution 3d surface construction algorithm, in: ACM SIGGRAPH, Vol. 21, 1987, pp. 163–169.

[12] L. P. Kobbelt, M. Botsch, U. Schwanecke, H.-P. Seidel, Feature sensitive surface extraction from volume data, in: ACM SIGGRAPH, 2001, pp. 57–66.

[13] E. V. Chernyaev, Marching cubes 33: Construction of topologically correct isosurfaces, Tech. Rep. CN 95–17, CERN (1995).

[14] T. Lewiner, H. Lopes, A. W. Vieira, G. Tavares, Efficient implementation of Marching Cubes cases with topological guarantees, Journal of Graphics Tools 8 (2) (2003) 1–15.

[15] M. Garland, P. S. Heckbert, Surface simplification using quadric error metrics, in: ACM SIGGRAPH, 1997, pp. 209–216.

[16] Y. Ohtake, A. Belyaev, Dual/primal mesh optimization for polygonized implicit surfaces, in: ACM Symposium on Solid Modeling and Applications, 2002, pp. 171–178.

[17] V. Surazhsky, P. Alliez, C. Gotsman, Isotropic remeshing of surfaces: a local parameterization approach, in: Proceedings of 12th International Meshing Roundtable, Santa Fe, New Mexico, USA, 2003, pp. 215–224.

[18] P. Alliez, E. Colin de Verdière, O. Devillers, M. Isenburg, Isotropic surface remeshing, in: SMI, IEEE Computer Society, 2003, pp. 49–58.

[19] S. Valette, J. Chassery, R. Prost, Generic remeshing of 3d triangular meshes with metric-dependent discrete voronoi diagrams, IEEE Trans Visu Comp Grap 14 (2) (2008) 369–381.

[20] E. Kartasheva, V. Adzhiev, A. Pasko, O. Fryazinov, V. Gasilov, Discretization of functionally based heterogeneous objects, in: SM '03: Proceedings of the eighth ACM symposium on Solid modeling and applications, ACM, New York, NY, USA, 2003, pp. 145–156.

[21] A. Pasko, V. Pilyugin, V. Pokrovskiy, Geometric modeling in the analysis of trivariate functions, Computer Graphics 12 (1988) 457–465.

[22] R. Shu, C. Zhou, M. S. Kankanhalli, Adaptive marching cubes, The Visual Computer 11 (4) (1995) 202–217.

[23] S. F. Frisken Gibson, Constrained elastic surface nets: Generating smooth surfaces from binary segmented data, in: MICCAI, Vol. 1496 of Lecture Notes in Computer Science, Springer, 1998, pp. 888–898.

[24] T. Ju, F. Losasso, S. Schaefer, J. D. Warren, Dual contouring of hermite data, in: ACM SIGGRAPH, 2002, pp. 339–346.

[25] B. T. Stander, J. C. Hart, Guaranteeing the topology of an implicit surface polygonization for interactive modeling, in: ACM SIGGRAPH, 1997, pp. 279–286.

[26] J.-D. Boissonnat, D. Cohen-Steiner, G. Vegter, Isotopic implicit surface meshing, in: ACM STOC '04, 2004, pp. 301–309.

[27] L. P. Chew, Guaranteed-quality mesh generation for curved surfaces, in: ACM SCG '93, 1993, pp. 274–280.

[28] H.-L. Cheng, T. K. Dey, H. Edelsbrunner, J. Sullivan, Dynamic skin triangulation, in: SODA '01: Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001, pp. 47–56.

[29] H. Edelsbrunner, N. R. Shah, Triangulating topological spaces, in: ACM SCG '94, 1994, pp. 285–292.

[30] J.-D. Boissonnat, S. Oudot, Provably good sampling and meshing of surfaces, Graphical Models 67 (5) (2005) 405–451.

[31] S.-W. Cheng, T. K. Dey, E. A. Ramos, T. Ray, Sampling and meshing a surface with guaranteed topology and geometry, in: ACM SCG '04, 2004, pp. 280–289.

[32] T. K. Dey, J. A. Levine, DelPSC: a delaunay mesher for piecewise smooth complexes, in: SCG '08: Proceedings of the twenty-fourth annual symposium on Computational geometry, ACM, New York, NY, USA, 2008, pp. 220–221.

[33] Q. Du, V. Faber, M. Gunzburger, Centroidal voronoi tesselations: applications and algorithms, SIAM Review (41(4)).

[34] S. P. Lloyd, Least squares quantization in pcm, IEEE Transactions on Information Theory 28 (2) (1982) 129–136.

[35] S. Valette, J.-M. Chassery, Approximated centroidal voronoi diagrams for uniform polygonal mesh coarsening, Computer Graphics Forum 23 (3) (2004) 381–390.

[36] T. K. Dey, J. A. Levine, Delaunay meshing of isosurfaces, The Visual Computer 24 (6) (2008) 411–422.

[37] P. Lindstrom, Out-of-core simplification of large polygonal models, in: ACM SIGGRAPH, 2000, pp. 259–262.

[38] R. Dyer, H. Zhang, T. Möller, Delaunay mesh construction, in: SGP '07, Eurographics Association, 2007, pp. 273–282.

[39] G. Taubin, F. Cukierman, S. Sullivan, J. Ponce, D. J. Kriegman, Parameterized families of polynomials for bounded algebraic curve and surface fitting, IEEE Transaction Pattern Analysis Machine Intelligence 16 (3) (1994) 287–303.