SIAM J. SCI. COMPUT.
Vol. 36, No. 3, pp. A930–A954

REVISITING OPTIMAL DELAUNAY TRIANGULATION FOR 3D GRADED MESH GENERATION*

ZHONGGUI CHEN†, WENPING WANG‡, BRUNO LÉVY§, LIGANG LIU¶, AND FENG SUN‡

Abstract. This paper proposes a new algorithm to generate a graded three-dimensional tetrahedral mesh. It revisits the class of methods based on optimal Delaunay triangulation (ODT) and proposes a proper way of injecting a background density function into the objective function minimized by ODT. This continuous/analytic point of view leads to an objective function that is continuous and Delaunay consistent, in contrast with the discrete/geometrical point of view developed in previous work. To optimize the objective function, this paper proposes a hybrid algorithm that combines a local search (quasi-Newton) with a global optimization (simulated annealing). The benefits of the method are both improved performances and an improved quality of the result in terms of dihedral angles. This results from the combination of two effects. First, the local search has a faster speed of convergence than previous work due to the better behavior of the objective function, and second, the algorithm avoids getting stuck in a poor local minimum. Experimental results are evaluated and compared using standard metrics.

Key words. mesh generation, optimal Delaunay triangulation, centroidal Voronoi tessellation, slivers, mesh optimization

AMS subject classifications. 42A10, 41A25, 41A50, 65N50

DOI. 10.1137/120875132

1. Introduction.

The quality of a tetrahedral mesh can tremendously affect the accuracy and efficiency of a finite element simulation [27]. A large body of work exists for generating tetrahedral meshes. Among these, the optimization-based methods—also called variational by some authors—have attracted much attention as they are capable of handling domains of complex shapes and topology in a consistent manner based on energy minimization. In this paper, we study the *optimal Delaunay triangulation (ODT)* for three-dimensional (3D) graded mesh generation.

ODT is defined as the minimizer of an objective function [2]. It has been shown [1] to be very effective in suppressing slivers, which are tetrahedra with near-zero volume but quite proper faces; for example, four evenly spaced vertices near a great circle of a sphere give rise to a sliver, as shown in Figure 1. Slivers have very small (near zero) and very large (near π) dihedral angles, which cause poor numerical conditions in simulation and are notoriously hard to remove. Hence, the ability of ODT to

*Submitted to the journal's Methods and Algorithms for Scientific Computing section April 27, 2012; accepted for publication (in revised form) February 21, 2014; published electronically May 13, 2014.

http://www.siam.org/journals/sisc/36-3/87513.html

†Department of Computer Science, Xiamen University, Xiamen, 361005 China, and University of Hong Kong, Hong Kong, China (chenzhonggui@xmu.edu.cn). Part of this work was done while the author was visiting the University of Hong Kong. This author was supported by NSFC 61100107, NSFC 61100105, and the Natural Science Foundation of Fujian Province of China (2012J01291, 2011J05007).

‡Department of Computer Science, University of Hong Kong, Hong Kong, China (wenping@cs.hku.hk, fsun@cs.hku.hk). The second author was supported by the National Basic Research Program of China (2011CB302400), NSFC 61272019, NSFC 61332015, and the Research Grant Council of Hong Kong (718209, 718010, 718311, 717012).

§Project ALICE, INRIA, Villers les Nancy, 54600 France (Bruno.Levy@inria.fr).

¶School of Mathematical Sciences, University of Science and Technology of China, Hefei, 230026 China (lgliu@ustc.edu.cn). This author was supported by NSF 61222206 and One Hundred Talent Project of the Chinese Academy of Sciences.

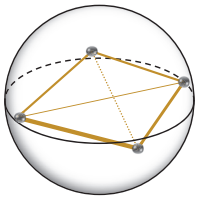


FIG. 1. *A sliver with four evenly spaced vertices near a circle.*

suppress slivers makes it a natural candidate for defining an objective function. Some modification of ODT allows us to generate a graded tetrahedral mesh, such that the size of the tetrahedra match a predefined background density function [32].

This paper introduces a method to improve both the speed and the quality of ODT-based mesh optimization, based on the following contributions:

- a proper formulation of the ODT objective function used to generate graded meshes, that satisfies more properties than previous work [32]: it is continuous and Delaunay consistent (section 3);
- linear quadratures for estimating the gradient of the ODT energy function (section 4.2) that improve the speed of convergence as compared with the one-point quadratures used in previous work [3] (see the comparison in Figure 9);
- a hybrid optimization algorithm that combines a quasi-Newton solver (L-BFGS) with a global optimization method (simulated annealing) to find a better minimum of the energy function (section 4.3);
- some hindsight about the sliver-suppressing properties of ODT (section 5.3).

2. Previous work. A comprehensive survey of mesh generation methods is beyond the scope of this paper; the reader is referred to the surveys in [13] and [29]. We restrict ourselves to the previous work directly related with our approach, based on the numerical optimization of an energy function and Delaunay triangulation (DT).

2.1. Optimization-based methods. Meshing methods of this class can be characterized according to three notions, *energy function*, *Delaunay consistence,* and *numerical optimization*, defined below.

Energy function. Given a compact domain $\Omega \subset \mathbb{R}^3$ and a set of vertices $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n \subset \Omega$, the quality of a tetrahedral mesh of Ω using $\mathbf{X}$ as the vertex set depends on the connectivity $\mathcal{T}$ of the vertices $\mathbf{X}$, as well as the positions of the vertices. Hence, an energy function (or a quality measure function) of a mesh has the form $F(\mathbf{X}, \mathcal{T})$. In general, the optimization of $F(\mathbf{X}, \mathcal{T})$ is nontrivial, since it is the combination of combinatorial optimization (over $\mathcal{T}$) and numerical optimization (over $\mathbf{X}$). A typical solution mechanism is to alternatively minimize $F(\mathbf{X}, \mathcal{T})$ over $\mathcal{T}$ with $\mathbf{X}$ fixed, and then minimize $F(\mathbf{X}, \mathcal{T})$ over $\mathbf{X}$ with $\mathcal{T}$ fixed.

Delaunay consistence. Due to its shape optimality, the Delaunay triangulation is widely adopted to determine the connectivity $\mathcal{T}$ for fixed $\mathbf{X}$. An energy function $F(\mathbf{X}, \mathcal{T})$ is said to be *Delaunay consistent* if for any fixed $\mathbf{X}$, $F(\mathbf{X}, \mathcal{T})$ attains its minimum when $\mathcal{T}$ is the Delaunay triangulation of $\mathbf{X}$. An energy function $F(\mathbf{X}, \mathcal{T})$ that is Delaunay consistent is preferred, since it decouples the combinatorial variables from the numerical ones. The combinatorial optimization only needs to compute a Delaunay triangulation, for which fast methods are available (e.g., CGAL, QHULL).

Numerical optimization. Several methods can be used to minimize $F(\mathbf{X}, \mathcal{T})$ over the continuous positions of the vertex $\mathbf{X}$ with fixed $\mathcal{T}$. Besides the selection of an appropriate method, one critical issue is to obtain accurate gradients of $F(\mathbf{X}, \mathcal{T})$ to

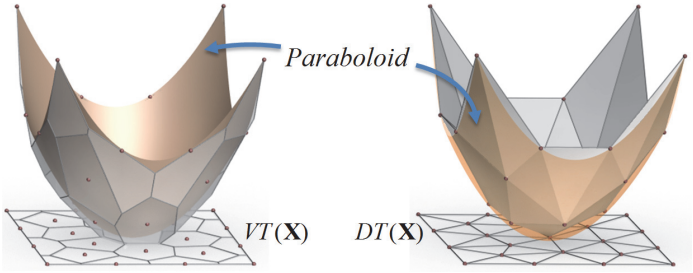


FIG. 2. *Illustrations of CVT (left) and ODT (right) energies.*

ensure fast convergence. For a Delaunay consistent function $F(\mathbf{X}, \mathcal{T})$, we note that the convergence of its minimization also depends on the smoothness of $F(\mathbf{X}, \mathcal{T})$ when the connectivity $\mathcal{T}$ changes due to the continuous change of the vertices $\mathbf{X}$ (more on this later).

Several energy functions were proposed, such as angles [12], tetrahedral condition number [11], simplicial element Jacobian norm [21], and mean-ratio metric [6]. A main problem with the majority of these methods is that their energy is not Delaunay consistent. Moreover, the resulting objective functions can be highly nonconvex and nonsmooth, making it hard to minimize.

Centroidal Voronoi tessellation (CVT) [7] has been well studied and successfully applied to isotropic meshing [8] and anisotropic meshing [9], mainly computed by Lloyd's method. Liu et al. [22] showed that the CVT energy has C^2 continuity and proposed to use the L-BFGS method, a variant of the quasi-Newton method, that outperforms the Lloyd's method for CVT computation. However, when applied to 3D mesh generation, CVT may generate a large number of slivers.

ODT was introduced by Chen and Xu [5]. Alliez et al. [1] observed that the ODT method performs better than CVT in terms of removing slivers. This is because the ODT energy directly measures the quality of a triangulation, while the CVT energy is formulated in terms of its dual structure (Voronoi diagram) and therefore does not penalize slivers. We now review the uniform and weighted cases.

2.2. Uniform ODT.

Definition. Let Ω denote the convex hull of a set of points $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ in $\mathbb{R}^d$. Let $\mathcal{T}$ be a triangulation of $\mathbf{X}$. Lifting up the $\mathbf{x}_i$ to the paraboloid $f(\mathbf{x}) = \|\mathbf{x}\|^2$, we obtain the set of projected vertices $\mathbf{X}' = \{\mathbf{x}_i'\}_{i=1}^n \subset \mathbb{R}^{d+1}$, where $\mathbf{x}_i' = (\mathbf{x}_i, \|\mathbf{x}_i\|^2)$ (see Figure 2). Let $f_I(\mathbf{x})$ be the piecewise linear interpolant of the vertices $\mathbf{x}_i'$ over the triangulation $\mathcal{T}$. Then the ODT energy $E_{\mathrm{ODT}}(\mathbf{X}, \mathcal{T})$: $\Omega^n \to \mathbb{R}$ is defined as the L^1 interpolation error of $f(\mathbf{x})$ by $f_I(\mathbf{x})$, that is,

$$E_{\mathrm{ODT}}(\mathbf{X}, \mathcal{T}) = \sum_{\tau \in \mathcal{T}} \int_\tau |f_I(\mathbf{x}) - f(\mathbf{x})| \mathrm{d}\mathbf{x}. \tag{2.1}$$

ODT is the global minimizer of $E_{\mathrm{ODT}}(\mathbf{X}, \mathcal{T})$. In practice, since one usually can only compute a local minimizer of the nonconvex function $E_{\mathrm{ODT}}(\mathbf{X})$, we will still use the term "ODT" to refer to a triangulation given by a local minimizer of $E_{\mathrm{ODT}}(\mathbf{X})$. Then, our goal is to minimize $E_{\mathrm{ODT}}(\mathbf{X}, \mathcal{T})$ to compute a triangulation of the domain Ω, with the positions of the sites $\mathbf{X}$ and their connectivity subject to optimization.

Property. $E_{\mathrm{ODT}}(\mathbf{X}, \mathcal{T})$ is Delaunay consistent [5]. This follows from the observation that the volume $E_{\mathrm{ODT}}(\mathbf{X}, \mathcal{T})$ is minimized when the polyhedron of $f_I(\mathbf{x})$ is

the lower convex hull $\mathrm{CH}(\mathbf{X}')$ of $\mathbf{X}' = \{\mathbf{x}'_i\}_{i=1}$ and the fact that the Delaunay triangulation of $\mathbf{X}$ is the projection of $\mathrm{CH}(\mathbf{X}')$ onto $\mathbb{R}^d$ (see Figure 2). Consequently, we will omit the triangulation $\mathcal{T}$ in $E_{\mathrm{ODT}}(\mathbf{X}, \mathcal{T})$ and always use the DT of $\mathbf{X}$ by default, since we are primarily interested in the minimization of $E_{\mathrm{ODT}}(\mathbf{X}, \mathcal{T})$.

Alternative formulations. $E_{\mathrm{ODT}}(\mathbf{X})$ is also given by

$$E_{\mathrm{ODT}}(\mathbf{X}) = \frac{1}{d+3} \sum_{i=1}^{n} \int_{\Omega_i} \|\mathbf{x} - \mathbf{x}_i\|^2 \mathrm{d}\mathbf{x}, \tag{2.2}$$

where Ω_i is the first-ring neighborhood of $\mathbf{x}_i$ [2].[1]

Existing solution mechanisms. The stationary point of (2.2) suggests a vertex update formula, proposed in [2] and modified in [1, 32]. The method iteratively applies the following two steps: (1) (*connectivity update*) compute the DT of the current vertices; (2) (*position update*) pick a vertex $\mathbf{x}_i$ and move it to the new location:

$$\mathbf{x}_i^* = \frac{1}{|\Omega_i|} \sum_{\tau \in \Omega_i} |\tau| \mathbf{c}_k, \tag{2.3}$$

where $|\tau|$ is the volume of τ. That is, $\mathbf{x}_i^*$ is the volume-weighted average of the circumcenters of all the simplices incident to $\mathbf{x}_i$. We will refer to this method as the *averaged Voronoi vertex (AVV) method*.

2.3. Weighted ODT. Graded meshes can be generated by considering the interpolation error to an arbitrary convex function f in (2.1) with the Hessian of f serving as a metric [4]. However, this is rather restrictive, because in general there does not exist a function whose Hessian matches a given arbitrary Riemannian metric. We now consider using a weighted extension of the ODT energy that takes a density function $\rho(\mathbf{x})$ into account.

A first possibility [2] is obtained by inserting the density $\rho(\mathbf{x})$ into (2.2), which yields

$$\tilde{E}^{\rho}_{\mathrm{ODT}}(\mathbf{X}, \mathcal{T}) = \frac{1}{d+3} \sum_{i=1}^{n} \int_{\Omega_i} \rho(\mathbf{x}) \|\mathbf{x} - \mathbf{x}_i\|^2 \mathrm{d}\mathbf{x}. \tag{2.4}$$

To minimize this energy, Alliez et al. [1] used the *discrete* point of view, which makes a geometrical interpretation of the AVV vertex update rule (equation (2.3)). With this point of view in mind, they multiply the weights by the background density function, thus yielding the vertex update formula

$$\mathbf{x}_i^* = \frac{1}{\sum_{\tau \in \Omega_i} |\tau|_\rho} \sum_{\tau \in \Omega_i} |\tau|_\rho \mathbf{c}_k,$$

which is obtained by replacing in the AVV update formula (2.3) the volume $|\tau|$ of the simplex τ in (2.3) by its weighted volume $|\tau|_\rho = \int_\tau \rho(\mathbf{x}) \mathrm{d}\sigma$. A variant called NODT was introduced in [32] that optimizes both the interior vertices and the vertices on the boundary.

The second extension [3], which we study in this paper, is obtained by inserting the density $\rho(\mathbf{x})$ into the ODT expression in (2.1):

$$E^{\rho}_{\mathrm{ODT}}(\mathbf{X}, \mathcal{T}) = \sum_{\tau \in \mathcal{T}} \int_{\tau} \rho(\mathbf{x}) |f_I(\mathbf{x}) - \mathbf{x}^2| \mathrm{d}\mathbf{x}. \tag{2.5}$$

[1] Note that the coefficient $\frac{1}{d+1}$ in [2] is incorrect.

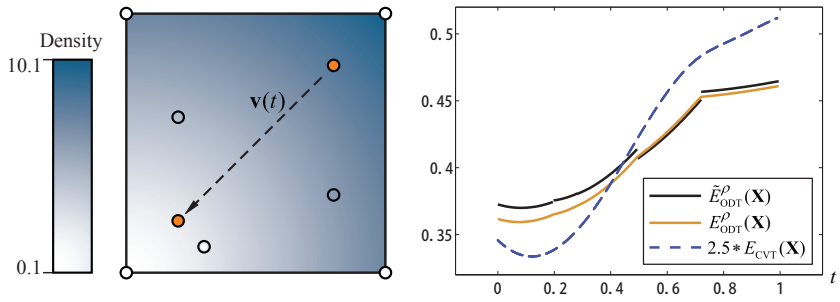


FIG. 3. *Empirical comparison of the continuity of CVT (C^2 but generates slivers), $\tilde{E}^{\rho}_{\mathrm{ODT}}$ (discontinuous), and energy E^{ρ}_{ODT} (C^0 but not C^1). The Delaunay triangulation is used by default in $\tilde{E}^{\rho}_{\mathrm{ODT}}$ and E^{ρ}_{ODT}.*

The gradient and Hessian matrix of this energy function are difficult to compute. For this reason, Chen and Holst [3] proposed a one-point quadrature to estimate the gradient and the Hessian matrix, and they minimize the energy with Newton-type iterations. We propose a better approximation of the gradient, leading to a more efficient solution mechanism.

3. Revisiting weighted ODT. The two extensions $\tilde{E}^{\rho}_{\mathrm{ODT}}$ (2.4) and E^{ρ}_{ODT} (2.5) are identical if the density function $\rho(\mathbf{x})$ is constant. For general density, there are some important differences.

As shown in Figure 3, we conducted a simple numerical experiment with a two-dimensional (2D) triangulation of eight vertices. One of the vertices moves along a straight line parameterized by t, and we plot the different energies (CVT, $\tilde{E}^{\rho}_{\mathrm{ODT}}(\mathbf{X}, DT)$, and $E^{\rho}_{\mathrm{ODT}}(\mathbf{X}, DT)$) as a function of t to compare their continuity. The $\tilde{E}^{\rho}_{\mathrm{ODT}}$ in (2.4), in general, is not Delaunay consistent. Moreover, it is not continuous when the Delaunay triangulation of the variable point set $\mathbf{X}$ changes its connectivity. Besides these empirical observations, we now further characterize the properties of E^{ρ}_{ODT} and prove the following properties.

THEOREM 3.1. *The ODT function E^{ρ}_{ODT} in* (2.5) *is Delaunay consistent.*

Proof. Let $f^*_I(\mathbf{x})$ be the piecewise linear interpolant of $f(\mathbf{x}) = \mathbf{x}^2$ based on the Delaunay triangulation $DT(\mathbf{X})$ of $\mathbf{X}$. Let $f_I(\mathbf{x})$ be a piecewise linear interpolation of $f(\mathbf{x})$ based on any triangulation $\mathcal{T}$ of $DT(\mathbf{X})$. Since $f^*_I(\mathbf{x})$ is given by the lower convex hull of the lifted points $\mathbf{X}'$ on the paraboloid, for any point $\mathbf{x}$ in the domain, we have

$$f_I(\mathbf{x}) - f(\mathbf{x}) \geq f^*_I(\mathbf{x}) - f(\mathbf{x}) \geq 0.$$

It follows that

$$\rho(\mathbf{x})|f_I(\mathbf{x}) - f(\mathbf{x})| \geq \rho(\mathbf{x})|f^*_I(\mathbf{x}) - f(\mathbf{x})|.$$

Hence,

$$DT(\mathbf{X}) = \arg\min_{\mathcal{T}} E^{\rho}_{\mathrm{ODT}}(\mathbf{X}, \mathcal{T}).$$

This completes the proof. ∎

In other words, for a fixed $\mathbf{X}$, the optimal triangulation that minimizes the ODT energy function among all triangulations is the Delaunay triangulation. This determines the combinatorial parameters $\mathcal{T}$, and we can now consider that E^{ρ}_{ODT} only depends on $\mathbf{X}$.

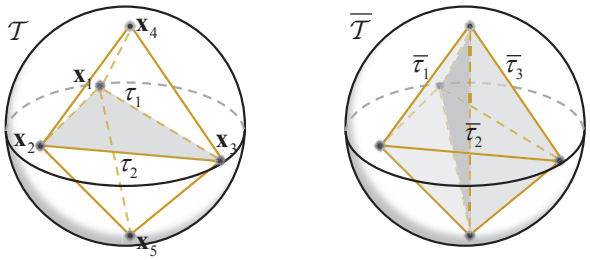


FIG. 4. *Co-spherical configuration.*

We shall now confirm that replacing the (discontinuous) $\tilde{E}^{\rho}_{\mathrm{ODT}}$ energy used in previous work with E^{ρ}_{ODT} gains one order of continuity.

THEOREM 3.2. *Suppose that the density function $\rho(\mathbf{x})$ is C^2 in Ω. Then the ODT function $E^{\rho}_{\mathrm{ODT}}(\mathbf{X}, \mathcal{T})$ in (2.5) is a C^0 function. Note that the ODT function is in general not C^1. Specifically, it is a piecewise C^2 function in Ω^n and has only C^0 but not C^1 contact between the pieces.*

Proof. Because the ODT function $E^{\rho}_{\mathrm{ODT}}(\mathbf{X})$ has different expressions for Delaunay triangulations $DT(\mathbf{X})$ with different ways of connectivity, it is defined in Ω^n in a piecewise way. Each piece corresponds to a subset of Ω^n whose points have Delaunay triangulations of the isomorphic connectivity.

First consider the smoothness of the $E^{\rho}_{\mathrm{ODT}}(\mathbf{X})$ at an interior point $\mathbf{X}_0$ of a piece. The expression of $E^{\rho}_{\mathrm{ODT}}(\mathbf{X})$ does not change within a sufficiently small neighborhood of $\mathbf{X}_0$. Each term of $E^{\rho}_{\mathrm{ODT}}(\mathbf{X}_0)$ defined over a tetrahedron τ which is

$$E^{\tau}(\mathbf{X}_0) = \int_{\tau} \rho(\mathbf{x}) |f_I(\mathbf{x}) - \mathbf{x}^2| \mathrm{d}\mathbf{x}$$

has an integrand that is a C^2 function in $\mathbf{x}$; furthermore, the integration domain τ changes smoothly with respect to the vertex positions. Hence, $E^{\tau}(\mathbf{X})$ is C^2 continuous at $\mathbf{X}_0$. So is $E^{\rho}_{\mathrm{ODT}}(\mathbf{X})$.

Now consider the smoothness of $E^{\rho}_{\mathrm{ODT}}(\mathbf{X})$ at the interface between different pieces, where the connectivity of the Delaunay triangulation $DT(\mathbf{X})$ changes. Without loss of generality, we consider the case when five vertices shared by two or three adjacent tetrahedra (as shown in Figure 4) come to lie on the same sphere in three dimensions. The Delaunay triangulation in three dimensions changes its connectivity, therefore admitting multiple Delaunay triangulations.

Let $\{\mathbf{x}_i\}_{i=1}^5$ denote these five co-spherical points. Let $\mathbf{c}$ and r denote the center and radius of the sphere, respectively. Any point $\mathbf{x}$ on the circle satisfies

$$\|\mathbf{x}\|^2 - 2\mathbf{x} \cdot \mathbf{c} + \|\mathbf{c}\|^2 = r^2.$$

The five points $\{\mathbf{x}_i\}_{i=1}^5$ are lifted up to the five points $\{\mathbf{x}'_i\}_{i=1}^5$ on the paraboloid $(\mathbf{x}, z) \in \mathbb{R}^4$ that lies on the same hyperplane defined by

$$z - 2\mathbf{x} \cdot \mathbf{c} + \|\mathbf{c}\|^2 - r^2 = 0$$

since $z = \|\mathbf{x}\|^2$.

Let $\mathcal{T}$ (consisting of two tetrahedra τ_1 and τ_2) and $\overline{\mathcal{T}}$ (consisting of three tetrahedra $\overline{\tau_1}$, $\overline{\tau_2}$, and $\overline{\tau_3}$) be the two Delaunay triangulation of $\{\mathbf{x}'_i\}_{i=1}^5$, as shown in

Figure 4. Clearly, the function $f_I(\mathbf{x})$ evaluated using these two triangulations has the same value, because the lifted points $\{\mathbf{x}'_i\}_{i=1}^5$ are coplanar in the space. It follows that

$$E^{\tau_1}(\mathbf{X}_0) + E^{\tau_2}(\mathbf{X}_0) = E^{\overline{\tau}_1}(\mathbf{X}_0) + E^{\overline{\tau}_2}(\mathbf{X}_0) + E^{\overline{\tau}_3}(\mathbf{X}_0).$$

That is, the different expressions of the ODT function given by two adjacent pieces have equal values at their interface. Hence, the ODT function is a C^0 function. The fact that it is not C^1 is easily established by numerical verification using a specific example. We will skip the details here. ∎

The solution mechanism described in the next section needs the expression of the gradient of $E^{\rho}_{\mathrm{ODT}}(\mathbf{X})$. If $\mathbf{X}$ has co-cyclic points, then the gradient is undefined; else it is given by the following theorem.

THEOREM 3.3. *The gradient of the ODT energy function $E^{\rho}_{\mathrm{ODT}}(\mathbf{X})$ with density $\rho(\mathbf{x})$ is given by*

$$\frac{\partial E^{\rho}_{\mathrm{ODT}}(\mathbf{X})}{\partial \mathbf{x}_i} = \sum_{\tau\in\Omega_i} \int_{\tau} \rho(\mathbf{x}) \frac{\partial f_I(\mathbf{x})}{\partial \mathbf{x}_i} \mathrm{d}\mathbf{x}, \tag{3.1}$$

where Ω_i is the first-ring neighborhood of the vertex $\mathbf{x}_i$.

Proof. First recall the gradient in the case of uniform density $\rho(\mathbf{x}) = 1$ [1],

$$\frac{\partial E_{ODT}(\mathbf{X})}{\partial \mathbf{x}_i} = \frac{2|\Omega_i|}{d+1}(\mathbf{x}_i - |\Omega_i|^{-1} \sum_{\tau\in\Omega_i} |\tau|\mathbf{c}_k), \tag{3.2}$$

where $\mathbf{c}_k$ is the circumcenter of the triangle τ.

With a density $\rho(\mathbf{x})$, the gradient of ODT function in (2.5) can be derived as follows. Consider the ODT energy on the 1-ring neighborhood Ω_i of vertex $\mathbf{x}_i$. By (2.5), we have

$$E^{\rho}_{ODT}(\mathbf{x}_i) \equiv \sum_{\tau\in\Omega_i} \int_{\tau} \rho(\mathbf{x}) f_I(\mathbf{x}) \mathrm{d}\mathbf{x} - \int_{\Omega_i} \rho(\mathbf{x}) f(\mathbf{x}) \mathrm{d}\mathbf{x},$$

which contains all the terms in $E^{\rho}_{\mathrm{ODT}}(\mathbf{X})$ involving $\mathbf{x}_i$. It follows that

$$\frac{\partial E^{\rho}_{\mathrm{ODT}}(\mathbf{X})}{\partial \mathbf{x}_i} = \sum_{\tau\in\Omega_i} \frac{\partial}{\partial \mathbf{x}_i} \int_{\tau} \rho(\mathbf{x}) f_I(\mathbf{x}) \mathrm{d}\mathbf{x}. \tag{3.3}$$

Here, $\mathbf{x}_i$ is involved in both the integrand and the integral domain of the integral terms. So we need to take into account the variation of the integral domain in differentiation as well. To simplify this formula, we need to recall the general Leibniz rule. Suppose that D_t is a time-varying domain. The velocity vector of a point on the domain boundary ∂D_t is $\mathbf{v} = \partial\mathbf{x}/\partial t$. Let $\mathbf{n}$ be the outward unit normal vector of ∂D_t. Then for a smooth function $g(\mathbf{x}, t), \mathbf{x} \in D_t$, the general Leibniz rule [10] states

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{D_t} g(\mathbf{x}, t) \mathrm{d}\mathbf{x} = \int_{\partial D_t} g(\mathbf{x}, t) \mathbf{v} \cdot \mathbf{n} \mathrm{d}\sigma + \int_{D_t} \frac{\partial g(\mathbf{x}, t)}{\partial t} \mathrm{d}\mathbf{x},$$

where $\mathrm{d}\sigma$ is the area element on ∂D_t.

Applying the general Leibniz rule to (3.3), we see that the integral terms over the boundaries of the τ will cancel out due to opposite orientations. Let Ω_i be the complete 1-ring neighborhood of $\mathbf{x}_i$. We have

$$\frac{\partial E^{\rho}_{\mathrm{ODT}}(\mathbf{X})}{\partial \mathbf{x}_i} = \sum_{\tau\in\Omega_i} \int_{\tau} \rho(\mathbf{x}) \frac{\partial f_I(\mathbf{x})}{\partial \mathbf{x}_i} \mathrm{d}\mathbf{x}. \tag{3.4}$$

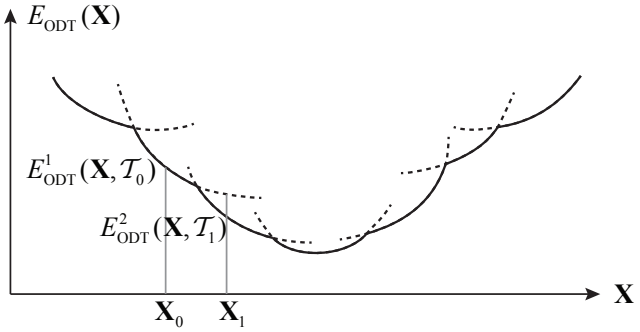


FIG. 5. *Behavior of L-BFGS applied to* E^{ρ}_{ODT}.

Note that $f_I(\mathbf{x})$ is a quadratic function of $\mathbf{x}_i$ on τ, so its derivative can easily be obtained. This completes the proof. □

4. Solution mechanism.

4.1. Reaching a local minimum with L-BFGS. Minimizing the ODT energy function is difficult because the function is nonconvex and nonsmooth (i.e., merely C^0). A common method is the local relaxation method which moves only one vertex at a time, like the AVV method mentioned in section 2.2. Alliez et al. [1] proposed to move all the vertices simultaneously and globally update the connectivity in every iteration. This treatment, though making computation faster, does not guarantee decreasing of the ODT energy. Chen and Holst [3] applied Newton's method to the minimization of ODT energy with the positions of vertices subject to optimization. They called it the "global" method, and showed that it significantly accelerates the convergence [3].

Newton's method, though converging quadratically, is not suitable for a large-scale problem due to the prohibitive cost of computing and storing the inverse Hessian matrix. We propose to apply a quasi-Newton method, specifically the L-BFGS method [24], to computing the ODT. Briefly, the L-BFGS method reduces greatly the time of each iteration by using accumulated gradient information to approximate the inverse Hessian, while maintaining fast convergence.

The L-BFGS method, like other Newton-type methods, is applied to functions with sufficient smoothness (i.e., C^2). The E^{ρ}_{ODT} function that we minimize is theoretically not smooth enough (C^0). However, a specific property of E^{ρ}_{ODT} (Delaunay consistence) results in good performances of L-BFGS.

The Delaunay consistence of the ODT function means that its graph is the lower envelope of the set of all expressions with different triangulation of $\mathbf{X}$. The one-dimensional analogue of such a function is illustrated in Figure 5 (see also Figure 3), with its graph characterized by nonconvex C^0 kinks; this is a property not possessed by arbitrary C^0 functions. Now suppose that the L-BFGS method is applied to the expression $E^{\rho}_{\mathrm{ODT}}(\mathbf{X}) = E^1_{\mathrm{ODT}}(\mathbf{X},\mathcal{T}_0)$ at the current point $\mathbf{X}_0$, where $\mathcal{T}_0 = DT(\mathbf{X}_0)$, and update it to $\mathbf{X}_1$ with a smaller value $E^1_{\mathrm{ODT}}(\mathbf{X}_1,\mathcal{T}_0)$. Suppose that $DT(\mathbf{X}_1) \neq DT(\mathbf{X}_0)$, i.e., there has been a connectivity change, for otherwise the nonsmoothness is not an issue. Then, the new location $\mathbf{X}_1$ can still be kept as a feasible, energy decreasing configuration, because, due to the Delaunay consistency property, the true ODT energy at $\mathbf{X}_1$, which is $E^{\rho}_{\mathrm{ODT}}(\mathbf{X}_1) = E^2_{\mathrm{ODT}}(\mathbf{X}_1, DT(\mathbf{X}_1))$, is smaller than $E^1_{\mathrm{ODT}}(\mathbf{X}_1,\mathcal{T}_0)$ and therefore also smaller than $E^{\rho}_{\mathrm{ODT}}(\mathbf{X}_0)$. That is, the

L-BFGS method can safely update the variable mesh vertices $\mathbf{X}$ across pieces of the domain Ω^n with different connectivities and still ensure an energy decrease and fast convergence.

The following algorithm optimizes a 3D mesh by minimizing E^{ρ}_{ODT}. The input is a background density function ρ and a 3D domain Ω, represented by its boundary $\partial\Omega$.

Algorithm 1. Quasi-Newton algorithm for ODT.

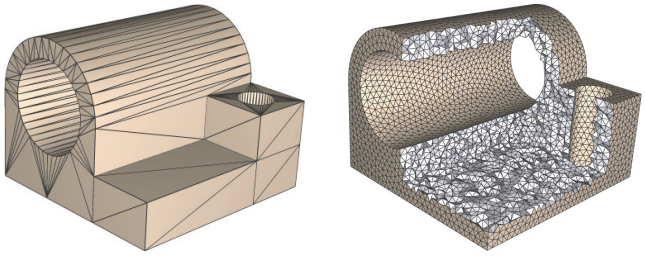
(1) Compute a remesh $\mathcal{S}$ of $\partial\Omega$ with density ρ
(2) initialize $\mathbf{X}^{(0)}$
while $\|\nabla E^{\rho}_{\mathrm{ODT}}(\mathbf{X}^{(k)})\| > \epsilon$ **do**
  (3) Compute the constrained Delaunay triangulation of $\mathbf{X}^{(k)}$
  (4) Compute $\nabla E^{\rho}_{\mathrm{ODT}}$
  (5) Compute $\mathbf{p}^{(k)}$ using the L-BFGS update rule
  (6) $\mathbf{X}^{(k+1)} \leftarrow \mathbf{X}^{(k)} + \mathbf{p}^{(k)}$
end

To facilitate reproducing our results, we further detail each step of the algorithm:

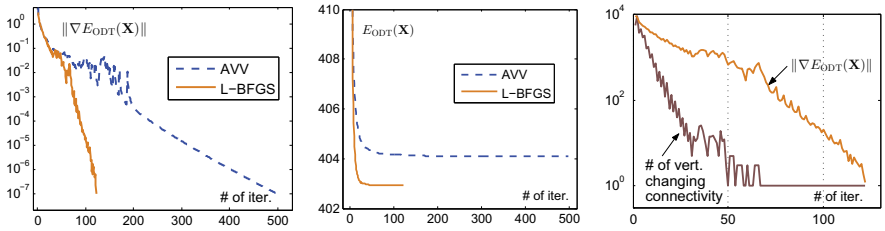
(1) We use the method in [33].
(2) $\mathbf{X}^{(0)}$ is initialized with the vertices of $\mathcal{S}$ and additional points randomly sampled in the interior of $\mathcal{S}$ with probability ρ.
(3) $E^{\rho}_{\mathrm{ODT}}(\mathbf{X}, \mathcal{T})$ is Delaunay consistent with respect to the CDT. Namely, the CDT of Ω with vertices in $\mathbf{X}$ minimizes the $E^{\rho}_{\mathrm{ODT}}(\mathbf{X}, \mathcal{T})$ among all constrained triangulations of $\mathbf{X}$ [28]. To compute the constrained Delaunay triangulation, we use the method in [30].
(4) See (3.1) and the complete formula in section 4.2. The vertices of the boundary $\mathcal{S}$ are "locked" by zeroing the components of the gradient that correspond to their coordinates.

Now we compare the efficiency of the AVV method and the L-BFGS method. Refer to the mesh model with uniform density in Figure 6. Here the input is a closed mesh enclosing a 3D domain. Additional points are randomly inserted for initialization. The two methods are run starting from the same random initialization. The norms of the gradients and the energies by these two methods are plotted in Figure 6(d), (e), which show that the L-BFGS method is stable and converges faster than the AVV method. With the stopping criterion $\|\nabla E(\mathbf{X})\| \leq 10^{-7}$, the L-BFGS method finishes in 122 iterations using 80.02 seconds, while the AVV method finishes in 499 iterations using 265.67 seconds.

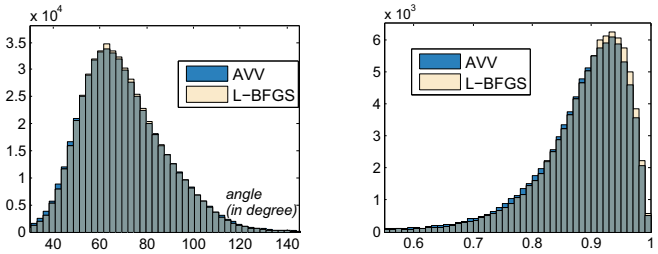
The next set of data shows that the L-BFGS method also tends to converge to an energy level lower than the one obtained with the AVV method. Figure 7 shows the distributions of the local minima obtained by the AVV method and the L-BFGS method applied to the mesh model in Figure 6, each 200 times, with 200 random initializations. In each of these 200 tests, the L-BFGS method yields a lower energy than the AVV method does. Thus, we may say that L-BFGS usually produces triangulations of slightly better quality than the AVV method; this is illustrated by the distributions of the dihedral angles and radius ratios of the triangulations, as shown in Figure 6(f), (g) for the mesh in Figure 6(a). Here the radius ratio, which is the ratio of the inscribed sphere radius to the circumscribed sphere radius of a tetrahedron, has been multiplied by 3 for normalization.



(a) Input boundary (b) Triangulation by L-BFGS optimization



(c) $\|\nabla E_{ODT}\|$ versus # of iter. (d) E_{ODT} versus # of iter. (e) connectivity change in L-BFGS



(f) Dihedral angle distributions (g) Radius ratio distributions

FIG. 6. *Convergence comparison.* (a) *Input boundary mesh;* (b) *tetrahedral mesh (with* 18,083 *vertices and* 91,025 *tetrahedra) by the L-BFGS method (*122 *iterations in* 80.02*s with final energy* 4.0296×10^2*);* (c) $\|\nabla E_{ODT}\|$ *versus number of iterations of selected method; AVV method finishes in* 499 *iterations using* 265.67 *seconds with final energy* 4.0411×10^2*;* (d) E_{ODT} *versus number of iterations of selected method;* (e) *number of vertices changing connectivity versus number of iterations of the L-BFGS method; (f) dihedral angle distribution; and (g) radius ratios distribution.*

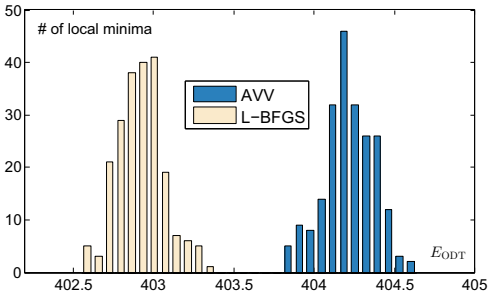
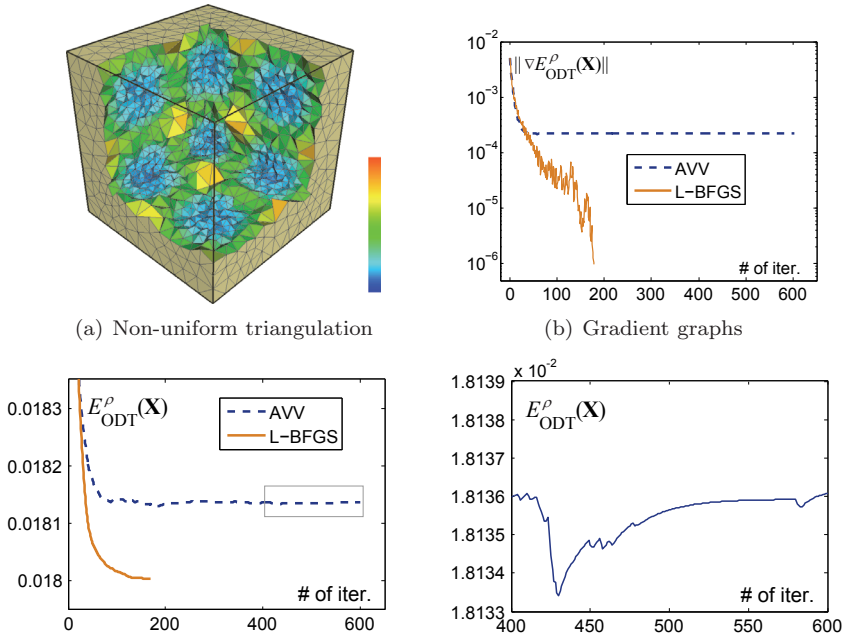


FIG. 7. *Distributions of local minima obtained by the AVV method and the L-BFGS method with* 200 *random initializations. The average values of local minima obtained by the AVV method and the L-BFGS method are* 4.0421×10^2 *and* 4.0292×10^2*, respectively.*



(a) Non-uniform triangulation (b) Gradient graphs

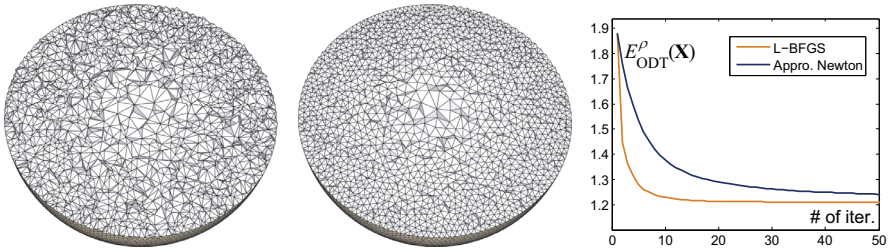
(c) Energy graphs (d) Zoom-in view

FIG. 8. *Weighted ODT energies.* (a) *Triangulation (4k vertices) generated by the L-BFGS method (*178 *iterations in* 194.4 *seconds). (The tetrahedra are color-coded in accordance to their average edge lengths.)* (b) *Gradient of energy versus number of iterations of selected method.* (c) *Energy value versus number of iterations of selected method. (d) Zoom-in view of the energy graph of AVV method in* (c).

Besides efficiency improvement, the L-BFGS method ensures monotonic decrease of the ODT energy in (2.5). Figure 8 shows a mesh obtained by minimizing the ODT energy with the density $\rho(\mathbf{x}) = e^{-20(x^2+y^2+z^2)} + 0.05\sin^2(\pi x)\sin^2(\pi y)\sin^2(\pi z)$. Here, the domain is a cube centered at the origin with an edge length of 2. The tetrahedra are color-coded in accordance to their average edge lengths. The norm of gradient and the energy are plotted in Figure 8(b)–(d). We see that the L-BFGS method decreases the energy value monotonically and is terminated with its gradient satisfying $\|\nabla E_{\mathrm{ODT}}\| \leq 10^{-6}$, while the energy value of the AVV exhibits oscillation, as shown in the zoom-in view in Figure 8(d). Here the numerical integration method over simplices from [14] is used to evaluate the gradient in (3.1) and the energy in (2.5).

The fast convergence of Newton-type methods is also confirmed in [3]. They used the approximate gradient and the Hessian matrix in Newton iterations to speed up the optimization. The gradient is evaluated by one-point numerical quadrature, and the Hessian matrix is approximated by a graph Laplacian. The time-consuming effort of this method is the iterative computation of the inverse of the Hessian. In Figure 9, we run the L-BFGS iteration and the approximated Newton [3] iteration 50 times, respectively, in a sphere with quadratic density $\rho(\mathbf{x}) = \mathbf{x}^2$. The gradient used in the L-BFGS method is exactly computed by (3.1). We can see that L-BFGS converges faster, as shown in Figure 9(c). And for each iteration, the approximated Newton method takes twice as long as L-BFGS takes.

4.2. Approximate computation of E^{ρ}_{ODT} and $\nabla E^{\rho}_{\mathrm{ODT}}$. If the density function $\rho(\mathbf{x})$ is simple, like the quadratic density in Figure 9, we can compute the energy



(a) Initial mesh (60k vert.) (b) L-BFGS (c) Energy graphs

Method	E^{ρ}_{ODT}	$\|\nabla E^{\rho}_{ODT}\|$	avg. radius ratio	time (sec.) per iter.
L-BFGS	1.21037	1.369e-5	0.8910	3.04
Appro. Newton	1.23069	7.294e-4	0.8685	6.13

(d) Statistics of comparison

FIG. 9. *Comparing L-BFGS and the approximated Newton method in* [3]. *The initial points are randomly distributed in the sphere according to the density function* $\rho(\mathbf{x}) = \mathbf{x}^2$.

E^{ρ}_{ODT} and its gradient $\nabla E^{\rho}_{\mathrm{ODT}}$ precisely. For general density, the numerical integration method in [14] can be used to approximate the integral with sufficient precision. (See the example in Figure 8.) If the density varies smoothly in a tetrahedron, we can approximate it by linear interpolation, thus achieving fast computation.

Noting that $f_I(\mathbf{x}) \geq \mathbf{x}^2$, the ODT energy in (2.5) can be written as

$$E^{\rho}_{\mathrm{ODT}}(\mathbf{X}) = \sum_{\tau \in DT(\mathbf{X})} \int_{\tau} \rho(\mathbf{x}) f_I(\mathbf{x}) \mathrm{d}\mathbf{x} - \int_{\Omega} \rho(\mathbf{x}) \mathbf{x}^2 \mathrm{d}\sigma.$$

We omit the computation of the second term in the above formula which is a constant for a fixed domain Ω.

THEOREM 4.1. *Suppose a tetrahedron* τ *is specified by its four vertices as* $\mathbf{x}_i$, *where* $i = 0, \ldots, 3$. *Approximating* $\rho(\mathbf{x})$ *by linear interpolation in each tetrahedron, one has*

$$\sum_{\tau \in DT(\mathbf{X})} \int_{\tau} \rho(\mathbf{x}) f_I(\mathbf{x}) \mathrm{d}\mathbf{x} \approx \frac{1}{20} \sum_{\tau \in DT(\mathbf{X})} |\tau| \left(\sum_{i,j=0}^{3} \rho_i w_j + \sum_{i=0}^{3} \rho_i w_i \right), \tag{4.1}$$

where $w_i = \|\mathbf{x}_i\|^2$, $\rho_i = \rho(\mathbf{x}_i)$.

Proof. Let $\lambda = (\lambda_0, \lambda_1, \lambda_2, \lambda_3)^T$ denote the barycentric coordinate of a point $\mathbf{x}$ in τ; then we have

$$\mathbf{x} = \sum_{i=0}^{3} \lambda_i \mathbf{x}_i, \qquad f_I(\mathbf{x}) = \sum_{i=0}^{3} \lambda_i w_i, \qquad \rho(\mathbf{x}) \approx \sum_{i=0}^{3} \lambda_i \rho_i. \tag{4.2}$$

Plugging the above formula into the integral in (4.1), we have

$$\begin{aligned} \int_{\tau} \rho(\mathbf{x}) f_I(\mathbf{x}) \mathrm{d}\mathbf{x} &\approx 6|\tau| \sum_{i,j=0}^{3} \rho_i w_j \int_{\hat{\tau}} \lambda_i \lambda_j \mathrm{d}\lambda \\ &= \frac{|\tau|}{20} \left(\sum_{i,j=0}^{3} \rho_i w_j + \sum_{i=0}^{3} \rho_i w_i \right). \end{aligned} \tag{4.3}$$

The last equation is obtained by using the formula for integrating a polynomial on the *canonical* tetrahedron [18],

$$\int_{\hat{\tau}} \lambda_1^{\alpha_1}\lambda_2^{\alpha_2}\lambda_3^{\alpha_3}\mathrm{d}\lambda = \frac{\alpha_1!\alpha_2!\alpha_3!}{(3+\sum_{i=1}^3\alpha_i)!},$$

where $\hat{\tau} = \{(\lambda_1,\lambda_2,\lambda_3)\in\mathbb{R}^3 | \sum_{i=1}^3\lambda_i \leqslant 1; \lambda_i \geqslant 0, i=1,2,3\}$. Summing (4.3) from all tetrahedra gives the result. ∎

Let $(x_i,y_i,z_i), i=0,\ldots 3$, denote the coordinates of vertices $\mathbf{x}_i$'s of a tetrahedron τ. For any point $\mathbf{x}=(x,y,z)$ in tetrahedron τ, $f_I(\mathbf{x})$ satisfies the following equation:

$$\det(\mathbf{M})=0, \qquad \text{where } \mathbf{M}=\begin{pmatrix} x & x_0 & x_1 & x_2 & x_3\\ y & y_0 & y_1 & y_2 & y_3\\ z & z_0 & z_1 & z_2 & z_3\\ f_I(\mathbf{x}) & w_0 & w_1 & w_2 & w_3\\ 1 & 1 & 1 & 1 & 1\end{pmatrix}. \tag{4.4}$$

Thus we get

$$f_I(\mathbf{x}) = \left(\frac{M_1}{M_4}, -\frac{M_2}{M_4}, \frac{M_3}{M_4}\right)\cdot\mathbf{x} + \frac{M_5}{M_4}, \tag{4.5}$$

where M_i is the $(i,1)$ minor of matrix $\mathbf{M}$. We then compute $\nabla E^{\rho}_{\mathrm{ODT}}$ by the following formula.

THEOREM 4.2. *Approximating $\rho(\mathbf{x})$ by linear interpolation in τ, one has*

$$\frac{\partial E^{\rho}_{\mathrm{ODT}}(\mathbf{X})}{\partial x_0} \approx \frac{1}{120}\sum_{\tau\in\Omega_0}(12x_0|\tau| - M_1)\left(\sum_{i=0}^3\rho_i + \rho_0\right), \tag{4.6}$$

where Ω_0 is the 1-ring neighborhood of vertex $\mathbf{x}_0$.

Proof. We first compute the derivative of f_I with respect to the first coordinate x_0 of the vertex $\mathbf{x}_0$. From (4.5), we have

$$\frac{\partial f_I}{\partial x_0} = (m_1,-m_2,m_3)\cdot\mathbf{x} + m_5, \tag{4.7}$$

where $m_i = \frac{M_i' M_4 - M_i M_4'}{M_4^2}$ and $M_i' = \frac{\partial M_i}{\partial x_0}$. On the other hand, according to (4.5), it obviously holds that

$$\|\mathbf{x}_i\|^2 = \left(\frac{M_1}{M_4}, -\frac{M_2}{M_4}, \frac{M_3}{M_4}\right)\cdot\mathbf{x}_i + \frac{M_5}{M_4}, i=0,\ldots 3.$$

Differentiating both sides of the above equations by x_0, we get

$$\begin{cases} 2x_0 = (m_1,-m_2,m_3)\cdot\mathbf{x}_i + \frac{M_1}{M_4} + m_5, & i=0,\\ 0 = (m_1,-m_2,m_3)\cdot\mathbf{x}_i + m_5, & i=1,2,3.\end{cases}$$

Substituting $\mathbf{x} = \sum_{i=0}^3\mathbf{x}_i\lambda_i$ into (4.7), we get

$$\frac{\partial f_I}{\partial x_0} = \left(2x_0 - \frac{M_1}{M_4}\right)\lambda_0.$$

Recalling that $\rho(\mathbf{x}) \approx \sum_{i=0}^{3} \lambda_i \rho_i$, we obtain

$$\int_\tau \rho(\mathbf{x}) \frac{\partial f_I(\mathbf{x})}{\partial x_0} \mathrm{d}\mathbf{x} \approx 6|\tau| \left(2x_0 - \frac{M_1}{M_4}\right) \int_{\hat{\tau}} \sum_{i=0}^{3} \rho_i \lambda_i \lambda_0 \mathrm{d}\lambda$$
$$= \frac{1}{120}(12x_0|\tau| - M_1) \left(\sum_{i=0}^{3} \rho_i + \rho_0\right).$$

Equation (4.6) is then obtained by using formula (3.1). ∎

For the ODT function with constant density, say, $\rho(\mathbf{x}) \equiv 1$, the formula (4.6) provides another efficient way to compute the exact derivative of the ODT function,

$$\frac{\partial E_{\mathrm{ODT}}(\mathbf{X})}{\partial x_0} = \frac{1}{24} \sum_{\tau \in \Omega_0} (12x_0|\tau| - M_1).$$

4.3. Global optimization. As the ODT energy function is nonconvex and nonsmooth, any local search scheme, including the fast L-BFGS method presented above, is susceptible to being stuck is a relatively poor local minimum. To address this issue, we shall next present a global optimization method, called the *global ODT method*, based on the principle of simulated annealing [19]. This global optimization process starts each round with a given local minimizer $\mathbf{X}_0$, typically produced by some local search scheme. Then by random perturbation and local optimization, a nearby local minimizer $\mathbf{X}$ with lower energy value is found as the new starting point for the next round. The perturbation to $\mathbf{X}_0$ is set to make the optimization jump out the basin of the current local minimum. An appropriate magnitude of random perturbation is crucial—if it is too small, the search will roll back to the same local minimum, while a too-large perturbation would amount to restarting optimization all over with a random initialization. With extensive experiments, we found that it is effective to set the perturbation magnitude of a vertex to be 0.2 times the average length of its incident edges.

The pseudocode of our global search scheme is given in Algorithm 2, followed by more details about each step.

Algorithm 2. Global ODT method.

(1) $m \leftarrow 0, \mathbf{X} \leftarrow \mathbf{X}_0, E = E^{\rho}_{\mathrm{ODT}}(\mathbf{X})$, temperature $T \leftarrow 10^{-4}$
(2) **while** $m < M$ **do**
  (3) $\mathbf{X}^* \leftarrow perturb(\mathbf{X})$
  (4) $\mathbf{X}^* \leftarrow L\text{-}BFGS(\mathbf{X}^*)$
  (5) $E^* = E^{\rho}_{\mathrm{ODT}}(\mathbf{X}^*)$
  (6) **if** $E^* < E$ or $\mathrm{Prob}(E, E^*, T) > Rand()$ **then**
    $\mathbf{X} \leftarrow \mathbf{X}^*$; $E \leftarrow E^*$
  else
    $T \leftarrow 0.9T$
  end
  (7) $m \leftarrow m + 1$
end
(8) $\mathbf{X}^* \leftarrow L\text{-}BFGS(\mathbf{X})$

The following hold:
(2) The optimization is stopped after M iterations ($M = 20$ in the results herein).

(3) Each vertex is perturbed along a random direction with a maximum magnitude of 0.2 times the average length of its incident edges.
(4) We run only 20 iterations of L-BFGS (Algorithm 1, steps (3) to (6), since a highly accurate minimizer is not necessary at this stage.
(6) $\mathbf{X}^*$ is accepted as the new starting point of the next round if its ODT energy E^* is lower than E; otherwise $\mathbf{X}^*$ is accepted with the probability $\text{Prob}(E, E^*, T) = \exp((E - E^*)/T)$.
(8) Finally, 30 iterations of L-BFGS are applied again to further minimize $\mathbf{X}$.

We first use a 2D mesh example to allow visual appreciation of the improvement brought about by the global ODT method. As shown in Figure 10, starting from the same random initialization, the 2D mesh obtained by the global ODT method (Figure 10(c)) is much better than the one obtained by the AVV method (Figure 10(a)), in terms of angle distributions (Figure 10(b)). The ODT energies of the sequence of local minima obtained in consecutive rounds are shown in Figure 10(d).

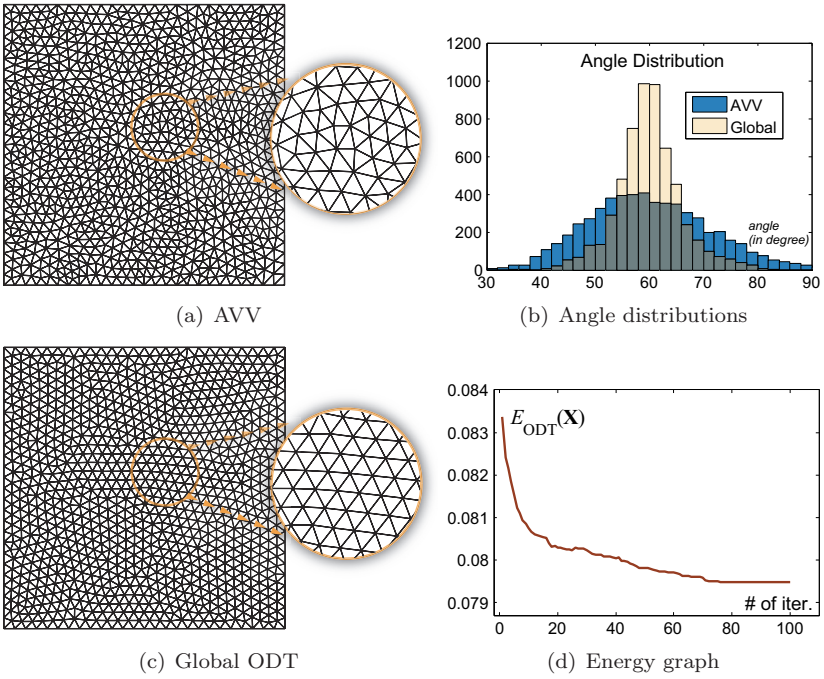
Next we consider the 3D meshing example shown in Figure 11. Here 20 rounds of the global ODT method are applied to the mesh model in Figure 6, finishing in 246.21 seconds with energy value 3.9799×10^2, which is much lower than the range $[4.0341 \times 10^2, 4.0432 \times 10^2]$ of the local minima produced by 200 repeated applications of the AVV method, as shown in Figure 7. In comparison, the AVV result (the same as in Figure 6) takes 499 iterations using 265.67 seconds with final energy 4.0411×10^2. Accordingly, the mesh quality has improved remarkably in terms of both the radius ratio and dihedral angle distributions. Furthermore, the number of slivers is further reduced by global optimization, even though it is already quite small by the AVV method.

5. Implementation and evaluation. Our algorithm is implemented in C++. We use the TetGen library [30] for generating 3D Delaunay triangulation. All the experiments are conducted on a laptop computer with a 2.66 GHz Intel processor and 4 GB memory.

5.1. Boundary handling. A tetrahedral mesh needs to conform with the domain boundary, which is a triangle mesh surface. Boundary handling refers to the way that those mesh vertices on the domain boundary are determined. In the NODT method by Tournois et al. [32], the positions of the boundary vertices are computed by maintaining the ODT circumsphere property and then projected onto the domain boundary. The biggest problem with this treatment is that the ODT energy, in general, is not decreased by an NODT iteration. To resolve this issue, we propose to first compute a good initial meshing of the domain boundary before minimizing the ODT energy. We use the restricted CVT method [33] to obtain a high-quality remeshing of the boundary. With the boundary mesh fixed, the constrained Delaunay triangulation is used in the ODT computation.

Figure 12 shows a comparison of the NODT method, the L-BFGS method, and the global ODT method. The global ODT method produces a triangulation of the best quality with the smallest number of slivers. As mentioned, the NODT method does not ensure the monotonic decrease of the ODT energy, as shown in Figure 12(b), and stops with the energy 0.5436, which is higher than those of the L-BFGS method and the global ODT method, which are 0.5347 and 0.5228, respectively.

5.2. Sliver-free graded mesh. Mesh gradation is controlled by a density function of the ODT energy function. We adopt the following method proposed in [1] for the automatic design of density functions. First, a sizing function $\mu(\mathbf{x})$ is defined in



(a) AVV (b) Angle distributions

(c) Global ODT (d) Energy graph

FIG. 10. *Comparison in 2D triangulations of 1k vertices.* (a) *The triangulation by the AVV (907 iterations in 8.748 seconds, final* $E_{\mathrm{ODT}} = 8.336 \times 10^{-2}$*);* (c) *angle distributions;* (b) *the mesh by the global ODT method (100 global iterations in 19.351 seconds, final* $E_{\mathrm{ODT}} = 7.946 \times 10^{-2}$*);* (d) *energy plot of the global ODT method.*

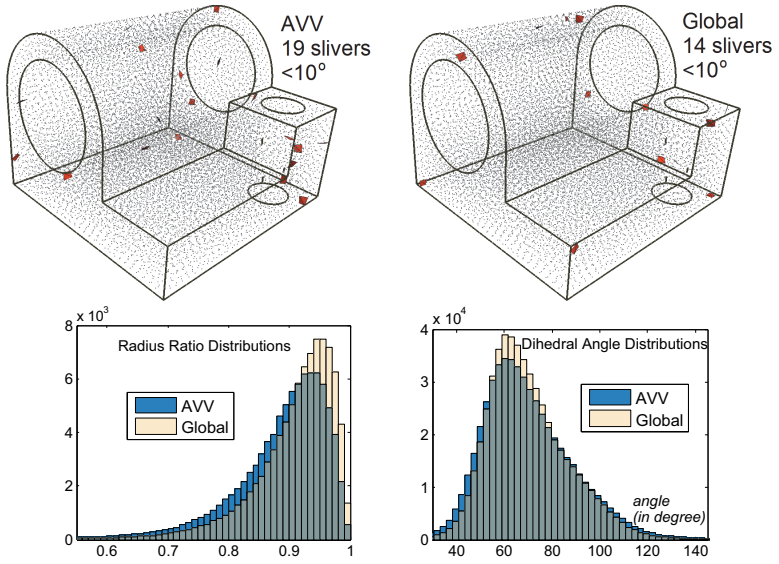
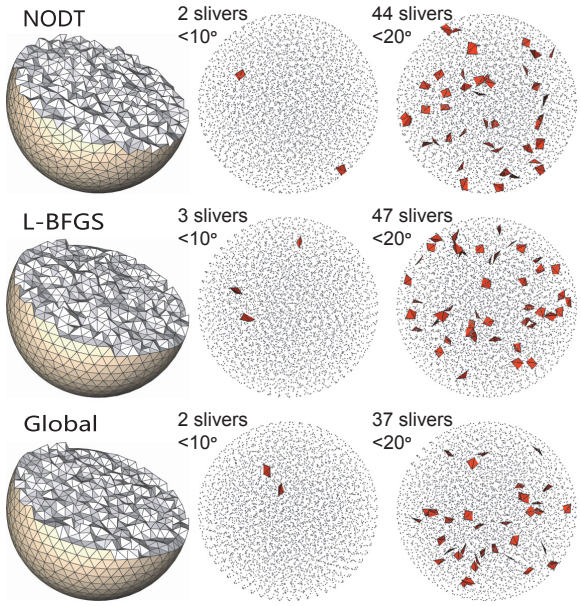
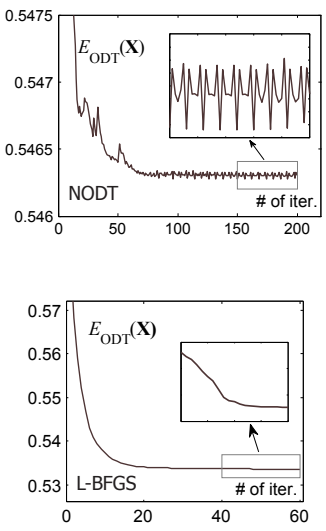


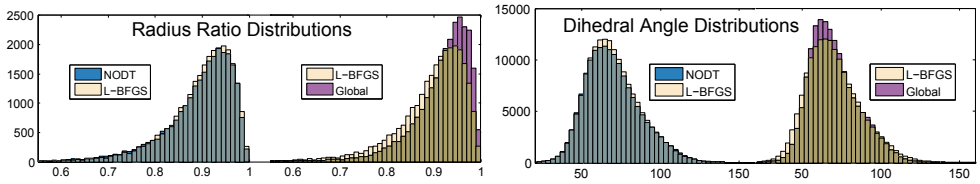
FIG. 11. *Comparison of the local AVV method and the global ODT method.* Top: *Slivers in the meshes by the two methods.* Bottom: *Radius ratio distributions and dihedral angle distributions of the two meshes.*



(a) Triangulations by NODT, L-BGFS and global ODT



(b) Energy graphs of NODT (top) and L-BGFS (bottom)

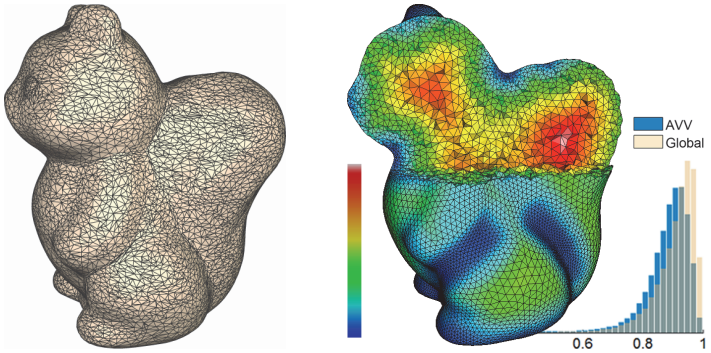


(c) Distributions

FIG. 12. *Comparing NODT* [32], *L-BFGS (Algorithm* 1*), and global ODT (Algorithm* 2*).* (a) 3*D triangulations by NODT, L-BFGS, and global ODT; slivers are shown for dihedral angle bounds of* 10° *and* 20°*, respectively;* (b) *energy graphs of NODT and L-BFGS;* (c) *radius ratio distributions and dihedral angle distributions.*

the domain based on the *local feature size*. An example of the sizing function is shown in Figure 13, where blue and red indicate the minimum and maximum values of the sizing function. Then a density function is defined as $1/\mu^5$, according to the fact that ODTs tend to equidistribute the weighted volume $\int_\tau \rho^{\frac{d}{d+2}} \mathrm{d}\mathbf{x}$ [4]. The density function used in the restricted CVT method for boundary surface remeshing is set to $1/\mu^4$. We first assign the density function values at the grid points of a uniform grid covering the domain and reconstruct the function by trivariate cubic spline interpolation. For fast computation of the ODT energy in (2.5) and the gradient in (3.1), we approximate the density function in a tetrahedral by linear interpolation. Figure 13 shows a mesh obtained by the global ODT method. The smooth gradation of the mesh is in agreement with the density function designed automatically by the above method.

Despite the use of the global ODT method, the slivers (<10 degrees) still exist, because of the difficulty in computing the true global minimizer. In [32, 31], the *sliver perturbation* method is proposed. It can effectively remove slivers, though it may not



(a) (b)

FIG. 13. *Automatic design of density function from the sizing field.* (a) *Input boundary mesh;* (b) *triangulation by global ODT with distribution of vertices consistent with density function. Distributions of radius ratios are shown on the right.*

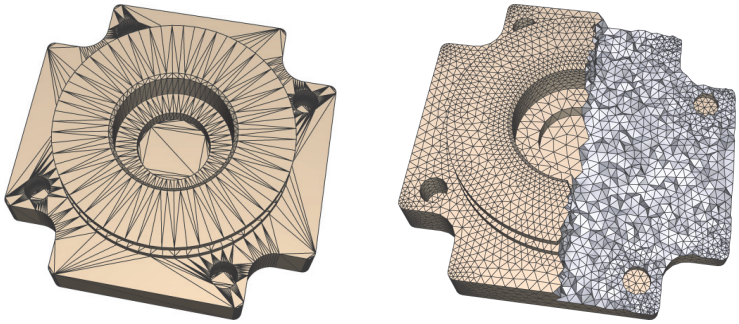


FIG. 14. *Remeshing a CAD model. Left: input boundary mesh; right: triangulation by global ODT and sliver perturbation. Minimal dihedral angle*= 17.6*, maximal dihedral angle* = 155.31*, average radius ratio* = 0.8868.

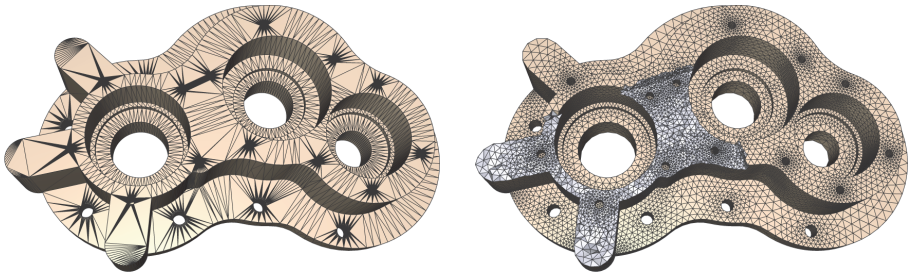


FIG. 15. *Input mesh model (left) and the tetrahedral mesh with* 277k *tetrahedra (right) generated by our hybrid local/global optimization; minimal dihedral angle*= 15.22*, maximal dihedral angle* = 157.45*, average radius ratio* = 0.8715.

decrease the ODT energy. This method can be integrated with our ODT method by using it as a postprocessing step. Figures 14 and Figure 15 show two meshes obtained by the global ODT method and further optimized by sliver perturbation. This combination produces meshes with improved angle distribution dihedral and that is free of slivers. (All the dihedral angles are between 15 degrees and 158 degrees.)

(a) Interleaved (b) Global ODT (c) Stellar

FIG. 16. *Moai model. Meshes by* (a) *interleaved method* [32] *and* (b) *our method. Red tetrahedra and blue tetrahedra are slivers with angle bounds* 10° *and* 20°*, respectively.* (c) *The sliced view of the mesh by Stellar* [20].

Method	Dihedral angle distribution	Radius ratio distribution	Avg. radius ratio	Slivers <10°	Slivers <20°	Time (s)
Interleaved	0.78 178.31	0.02	0.8818	21	214	186.66
Interleaved +Sliver perturbation	20.00 150.82	0.30	0.8685	0	0	+85.58
Global ODT	0.71 178.87	0.01	0.9110	17	115	236.24
Global ODT +Sliver perturbation	20.03 151.45	0.33	0.9007	0	0	+50.45
Stellar	37.21 139.22	0.29	0.7648	0	0	446.61
Global ODT +Stellar	41.40 136.13	0.30	0.8630	0	0	+421.42
	0 50 100 150	0 0.2 0.4 0.6 0.8 1				

FIG. 17. *Moai model: statistics of different methods.*

We now compare the global ODT method with the interleaved method [32] and the Stellar method [20], using a model with uniform density in Figure 16. Stellar can drastically increase the smallest dihedral angles in a mesh without paying much attention to improving angle distribution. Statistics are shown in Figure 17. The global ODT method and Stellar take as input the optimized boundary mesh and random initialization of the interior vertices. The interleaved method operates by alternating Delaunay refinement and ODT optimization. Sliver perturbation is applied as a postprocessing with an angle bound of 20 degrees. Again, the global ODT method yields the best triangulation in terms of the dihedral angle and radius ratio distributions. When postprocessed by sliver perturbation, the global ODT method produces a high-quality mesh free of slivers. Although Stellar improves the meshes and results in all dihedral angles between 30° and 150°, it does little to help produce evenly spaced mesh vertices, thus resulting in many needle-shaped tetrahedra (see Figure 16(c)).

5.3. Comparison of ODT and CVT. It has been observed [1, 32] that the ODT method is more effective than the CVT method in suppressing slivers. That conclusion is based on two observations: (1) The CVT method, with its energy defined

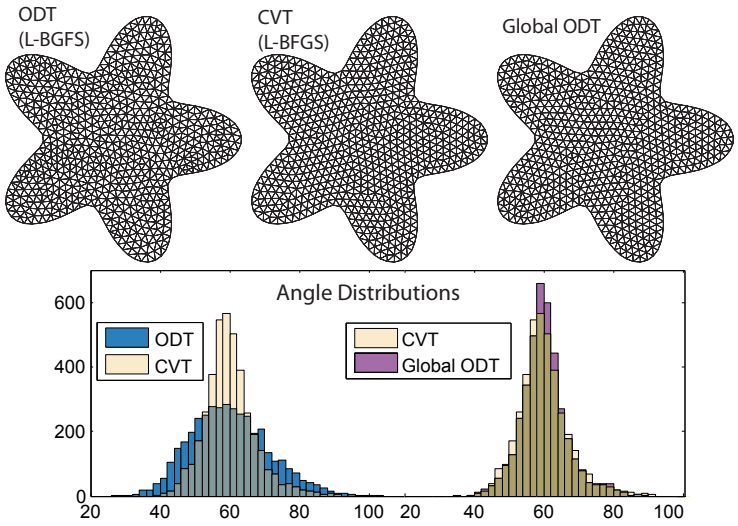


FIG. 18. *Comparing CVT and ODT on 2D triangulations.*

in the dual Voronoi domain, only tries to make the mesh vertices evenly spaced, which is not sufficient for avoiding slivers; that is, the CVT energy does not penalize slivers. (2) On the other hand, the ODT energy function, formulated in the prime space of triangulation, can be shown to optimize certain shape measurements of tetrahedral elements (see equation (7) in [1]). Those works [1, 32] also present experimental comparisons to confirm this conclusion using the Lloyd method for CVT computation and the AVV method for ODT computation.

Given the popularity of the CVT method in mesh generation and the superior performance of the ODT method in sliver suppression, it would be instructive to provide further analysis and understanding to supplement the above observations. First, we note that for 2D meshes, well-spaced mesh vertices *are* sufficient to guarantee triangle elements of good quality. Hence, the ODT method does not have a real advantage over the CVT method in 2D mesh generation or surface remeshing, as shown by the results in Figures 18(a) and (b). Since the ODT method and the CVT method handle boundary vertices in different ways, for a fair comparison, evenly spaced boundary vertices are provided before optimization and fixed during optimization. The better quality of the CVT result seen here can be attributed to the C^2 smoothness of the CVT energy (versus the nonsmoothness of the ODT energy), which makes it easier for the CVT method to produce a local minimizer of good mesh quality.

The situation with 3D triangulations is more complicated. According to the celebrated Gersho conjecture [16], in theory the tetrahedral mesh given by the global minimizer of the CVT energy should also be free of slivers. Therefore, the experimental observation that the CVT is inferior to the ODT in sliver suppression implies that the CVT energy function is rather insensitive to the presence of slivers, in the sense that it allows local minimizers that contain a large number of slivers. This is in line with the conclusion in [1] that the CVT energy does not penalized slivers. Furthermore, the ability of the CVT method in sliver suppression is clearly dependent on the particular computational method used. In this regard, we compare in Figure 19 the performance of the ODT method and the CVT method for 3D meshing, both using the fast L-BFGS method and simulated annealing-based global search. The results again confirm that the ODT is much better at sliver suppressing.

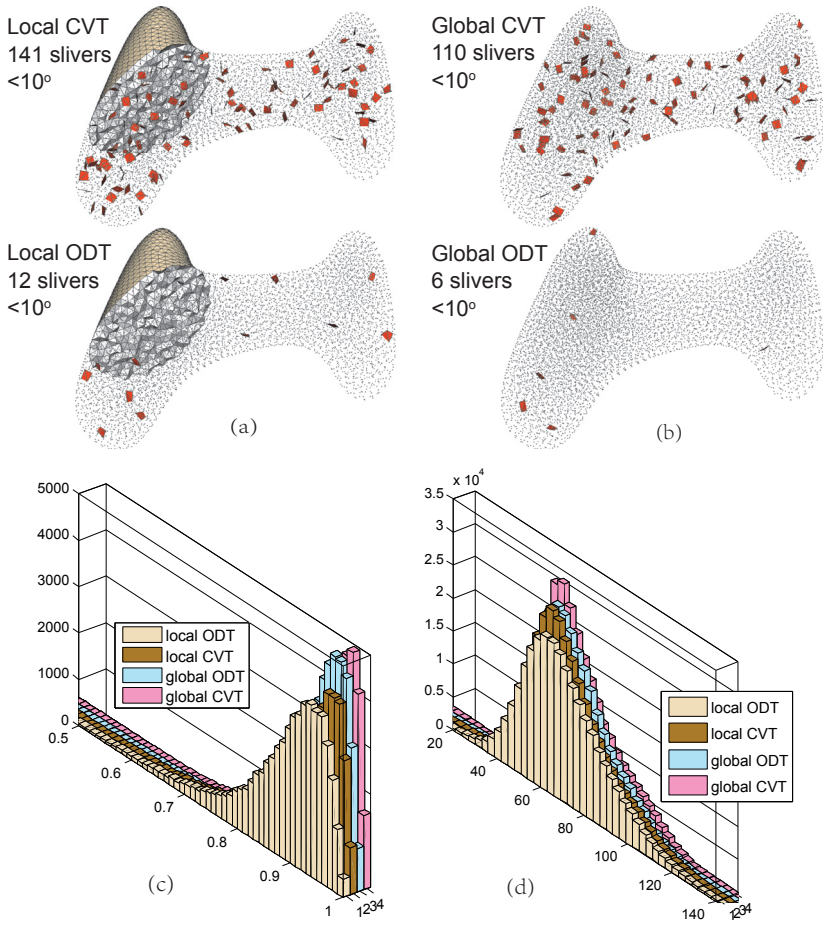


FIG. 19. *Comparing CVT and ODT. Slivers are shown for a dihedral angle bound of* 10°. (a) *Triangulations by local CVT (top) and local ODT (bottom);* (b) *triangulations by global CVT (top) and global ODT (bottom);* (c) *radius ratio distributions; (d) dihedral angle distributions.*

However, we note that the tetrahedral mesh by the CVT method has better distributions of dihedral angles than that by the ODT method, as shown in Figures 19(c) and (d). This suggests that it would pay off to apply the following scheme that combines the strength of both ODT and CVT. Given an initial set of random vertices in a 3D domain, first run the CVT method to obtain a tetrahedra mesh with good angle distribution, and then run the ODT method on it to reduce its slivers, followed by sliver perturbation to remove all the remaining slivers. Figure 20 shows the intermediate results of such a procedure.

5.4. Numerical examples. In the process of solving a partial different equation by finite element methods, the shape of the elements has strong influence on the finite element solutions. We here consider the 3D Poisson problem, as it has a large number of applications in practice. Let Ω be a 3D domain with boundary $\partial\Omega$. We look for an approximate solution of the following Poisson equation by using the linear finite

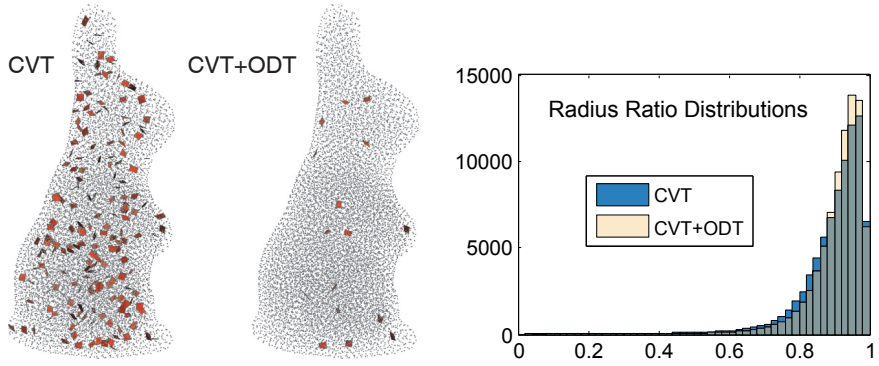


FIG. 20. *Combining CVT and ODT methods. After applying ODT optimization to the result of the CVT method, the number of slivers is reduced from* 195 *to* 16*, and the average radius ratio rises from* 0.8930 *to* 0.9085*.*

TABLE 1
Comparison of error norms for Poisson problem with different meshes.

Method	Mesh generation (#vert. = 9009)				Error norm	
	Avg. rad. ratio	min$\angle$	max$\angle$	Time (s)	L^2	H^1
Structured	0.596	30.00^o	135.00^o	0.017	$2.729e{-3}$	$2.011e{-1}$
TetGen	0.781	7.82^o	164.99^o	0.572	$2.159e{-3}$	$1.499e{-1}$
Gmsh	0.808	16.49^o	150.29^o	2.011	$1.953e{-3}$	$1.468e{-1}$
GHS3D	0.845	22.882	136.71^o	0.483	$1.919e{-3}$	$1.298e{-1}$
NetGen	0.872	25.98^o	138.02^o	11.026	$1.805e{-3}$	$1.318e{-1}$
Ours	0.904	15.13^o	152.31^o	92.127	$1.601e{-3}$	$1.271e{-1}$

element method:

$$-\nabla^2 u(\mathbf{x}) = f(\mathbf{x}) \text{ in } \Omega,$$
$$u(\mathbf{x}) = g(\mathbf{x}) \text{ on } \partial\Omega.$$

We create triangulations of Ω with different methods and compare the approximate errors of the corresponding finite element solutions. To compare the approximate solution u_h, we solve the Poisson equation with the exact solution $u_{exc}(\mathbf{x}) = xyz$ on $\Omega = (-1, 1)^3$. The approximate error is computed in both the L^2-norm and the H^1-seminorm:

$$\|u_{exc} - u_h\|_{L^2} = \sqrt{\int_\Omega (u_{exc} - u_h)^2 \mathrm{d}\mathbf{x}}, \quad \|u_{exc} - u_h\|_{H^1} = \sqrt{\int_\Omega \|\nabla(u_{exc} - u_h)\|^2 \mathrm{d}\mathbf{x}}.$$

For tetrahedral mesh generation, we explore a number of widely used tools, in particular, TetGen [30], GHS3D [15], Gmsh [17], and NetGen [26]. Table 1 shows the statistics of the qualities of the meshes generated by different methods and the error norms of the solutions. The number of vertices is prespecified and a uniform sizing function is used in all the methods. The structured mesh is constructed by connecting the regular grid points in the cube. We can see that the accuracy of the solution increases along with the improvement of the mesh quality indicated mainly by the average value of radius ratios. Our method is capable of generating high-quality meshes benefiting from the global optimization involved in the algorithm. On the other side, our method is less efficient than the most efficient state of the art (e.g., the software

TABLE 2
Running times.

Model	# vert.	# tet.	Time (min.)		
			Initialization	Global ODT	Sliver perturbation
Figure 6	18k	91k	1.5	4.3	0.25
Figure 13	43k	206k	5.2	16.1	1.21
Figure 14	17k	82k	1.3	3.9	0.23
Figure 15	75k	404k	6.3	26.7	1.42

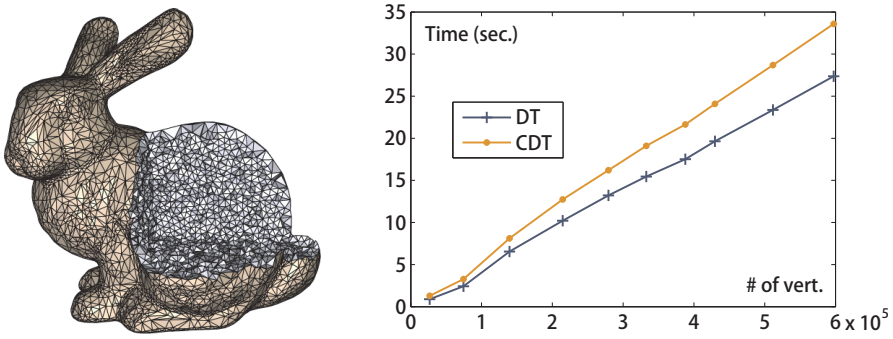


FIG. 21. *Comparing the time of computing CDT and DT.*

GHS3D from Distene [15] or the advancing front methods in [25, 23]), due to the high computational complexity involved in the global optimization.

5.5. Timing. The running time for each example discussed is given in Table 2. It consists of three parts. The initialization part includes the time for calling the restricted CVT method [33] to compute the optimized boundary mesh. The angle bound in sliver perturbation is set to 15 degrees. In all these examples, at most 20 rounds of the global ODT method are applied; in each round, 20 iterations of the L-BFGS method are applied. The most time-consuming part of the ODT method is building the constrained Delaunay triangulation. We test the algorithms of CDT and DT implemented in TetGen [30] with different numbers of points, sampled in a bunny model (Figure 21). The CDT algorithm is about 20% slower than computing DT.

6. Conclusion. In this paper, we have discussed some variations of ODT-based mesh generation when a density function is used to specify a graded mesh. From the combinatorial point of view, as compared with the initial formulation in [2], the energy function E^{ρ}_{ODT} that we used is optimized by the standard Delaunay triangulation, which can be constructed effectively using existing software packages. From the numerical point of view, the objective function that we minimize and the quasi-Newton method show faster convergence than in previous work [1] due to more regularity in the objective function. As compared with [3], which uses the same objective function as here, we show that a more accurate estimation of the gradient improves the performances. In addition, based on the observation that the ODT energy is nonsmooth and nonconvex, we show that global optimization further optimizes the ODT energy and achieves high-quality resulting meshes with fewer slivers and better dihedral angle and radius ratio distributions.

Considering the limitations of our method, some further research possibilities are suggested below:

(1) The objective function E^{ρ}_{ODT} introduced here together with the hybrid optimization algorithm improves the performance and quality of graded mesh generation. We cannot ensure that the global minimum is obtained, but we give experimental results that show the practical benefit of our mixed Newton/global optimization framework. However, slivers are not totally eliminated, thus requiring a postprocessing with sliver perturbation [31].
(2) Our method needs to fix the vertices on the boundary. Note that a graded meshing of the boundary can be precomputed (e.g., using [33]). We use the constrained Delaunay triangulation in the ODT optimization. However, the constrained Delaunay triangulation may not exist in some extreme cases. Since the points are nearly evenly distributed in the domain during the ODT optimization, we never failed to build the constrained Delaunay triangulations in our experiments.
(3) The objective function is C^0 only, which is better than the discontinuous functions used in previous work but still does not fulfill the C^2 theoretical requirement of the L-BFGS algorithm. To further improve both the speed and the quality of the results, a unified, C^2 sliver-eliminating objective function remains to be found.

In another perspective, efficient optimization methods for anisotropic mesh generation are also an important research topic.

Acknowledgment. The surface models are courtesy of Aim@Shape.

REFERENCES

[1] P. Alliez, D. Cohen-Steiner, M. Yvinec, and M. Desbrun, *Variational tetrahedral meshing*, ACM Trans. Graphics, 24 (2005), pp. 617–625.
[2] L. Chen, *Mesh smoothing schemes based on optimal delaunay triangulations*, in Proceedings of the 13th International Meshing Roundtable, 2004, pp. 109–120.
[3] L. Chen and M. Holst, *Efficient mesh optimization schemes based on optimal Delaunay triangulations*, Comput. Methods Appl. Mech. Engrg. 200 (2011), pp. 967–984.
[4] L. Chen, P. Sun, and J. Xu, *Optimal anisotropic meshes for minimizing interpolation errors in L^p norm*, Math. Comp., 76 (2007), pp. 179–204.
[5] L. Chen and J. Xu, *Optimal Delaunay triangulations*, J. Comput. Math., 22 (2004), pp. 299–308.
[6] L. F. Diachin, P. Knupp, T. Munson, and S. Shontz, *A comparison of two optimization methods for mesh quality improvement*, Eng. Comput., 22 (2006), pp. 61–74.
[7] Q. Du, V. Faber, and M. Gunzburger, *Centroidal Voronoi tessellations: Applications and algorithms*, SIAM Rev., 41 (1999), pp. 637–676.
[8] Q. Du and D. Wang, *Tetrahedral mesh generation and optimization based on centroidal Voronoi tessellations*, Internat. J. Numer. Methods Engrg., 56 (2003), pp. 1355–1373.
[9] Q. Du and D. Wang, *Anisotropic centroidal Voronoi tessellations and their applications*, SIAM J. Sci. Comput., 26 (2005), pp. 737–761.
[10] H. Flanders, *Differentiation under the integral sign*, Amer. Math. Monthly, 80 (1973), pp. 615–627.
[11] L. A. Freitag and P. M. Knupp, *Tetrahedral element shape optimization via the Jacobian determinant and condition number*, in Proceedings of the 8th International Meshing Roundtable, 1999, pp. 247–258.
[12] L. A. Freitag and C. Ollivier-gooch, *A comparison of tetrahedral mesh improvement techniques*, in Proceedings of the 5th International Meshing Roundtable, 1996, pp. 87–100.
[13] P. J. Frey and P.-L. George, *Mesh Generation: Application to Finite Elements*, Wiley, London, 2000.
[14] A. Genz and R. Cools, *An adaptive numerical cubature algorithm for simplices*, ACM Trans. Math. Softw., 29 (2003), pp. 297–308.

[15] P.-L. George, *Tetmesh-GHS3D, tetrahedral mesh generator*, in INRIA User's Manual, INRIA, Paris, 2004.
[16] A. Gersho, *Asymptotically optimal block quantization*, IEEE Trans. Inform. Theory, 25 (1979), pp. 373–380.
[17] C. Geuzaine and J.-F. Remacle, *Gmsh: A 3-d finite element mesh generator with built-in pre- and post-processing facilities*, Internat. J. Numer. Methods Engrg., 79 (2009), pp. 1309–1331.
[18] A. Grundmann and H. M. Möller, *Invariant integration formulas for the n-simplex by combinatorial methods*, SIAM J. Numer. Anal., 15 (1978), pp. 282–290.
[19] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, *Optimization by simulated annealing*, Science, 220 (1983), pp. 671–680.
[20] B. M. Klingner, *Improving Tetrahedral Meshes*, Ph.D. thesis, EECS Department, University of California, Berkeley, 2008.
[21] P. Knupp, *Achieving finite element mesh quality via optimization of the Jacobian matrix norm and associated quantities. Part* II*—a framework for volume mesh optimization and the condition number of the Jacobian matrix*, Internat. J. Numer. Methods Engrg., 48 (2000), pp. 1165–1185.
[22] Y. Liu, W. Wang, B. Lévy, F. Sun, D.-M. Yan, L. Lu, and C. Yang, *On centroidal Voronoi tessellation—energy smoothness and fast computation*, ACM Trans. Graphics, 28 (2009), pp. 1–17.
[23] R. Löhner, *A parallel advancing front grid generation scheme*, Internat. J. Numer. Methods Engrg., 51 (2001), pp. 663–678.
[24] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed., Springer, New York, 2006.
[25] J. Peraire and K. Morgan, *Automatic generation of unstructured meshes for Navier Stokes flows*, in Proceedings of AIAA, Albuquerque, NM, 1998, pp. 98–3010.
[26] J. Schöberl, *Netgen—an advancing front* 2*d/*3*d-mesh generator based on abstract rules*, Comput. Vis. Sci., 1 (1997), pp. 41–52.
[27] J. R. Shewchuk, *What is a good linear element? Interpolation, conditioning, and quality measures*, in Proceedings of the 11th International Meshing Roundtable, 2002, pp. 115–126.
[28] J. R. Shewchuk, *General-dimensional constrained delaunay and constrained regular triangulations* I*: Combinatorial properties*, Discrete Comput. Geom., 39 (2008), pp. 580–637.
[29] J. R. Shewchuk, *Unstructured mesh generation*, in Combinatorial Scientific Computing, U. Naumann and O. Schenk, eds., CRC Press, Boca Raton, FL, 2011, pp. 259–298.
[30] H. Si, *TetGen: A Quality Tetrahedral Mesh Generator and Three-Dimensional Delaunay Triangulator*, http://tetgen.berlios.de (2007).
[31] J. Tournois, R. Srinivasan, and P. Alliez, *Perturbing slivers in* 3*D delaunay meshes*, in Proceedings of the 18th International Meshing Roundtable, 2009, pp. 157–173.
[32] J. Tournois, C. Wormser, P. Alliez, and M. Desbrun, *Interleaving Delaunay refinement and optimization for practical isotropic tetrahedron mesh generation*, ACM Trans. Graphics, 29 (2009), pp. 1–9.
[33] D.-M. Yan, B. Lévy, Y. Liu, F. Sun, and W. Wang, *Isotropic remeshing with fast and exact computation of restricted Voronoi diagram*, Comput. Graphics Forum, 28 (2009), pp. 1445–1454.