Mesh Smoothing Algorithms for Complex Geometric Domains

Hale Erten, Alper Üngör*, and Chunchun Zhao

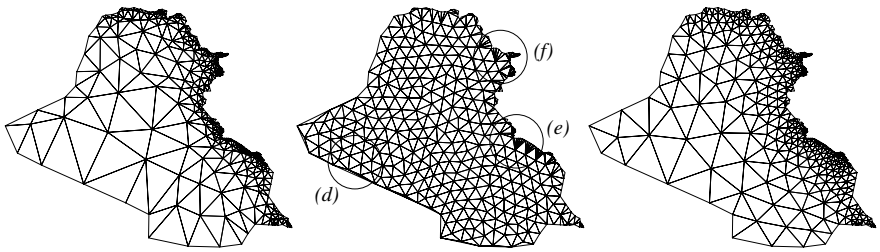
University of Florida, Dept. of Computer & Info. Sci. & Eng.
{herten,ungor,czhao}@cise.ufl.edu

Abstract. Whenever a new mesh smoothing algorithm is introduced in the literature, initial experimental analysis is often performed on relatively simple geometric domains where the meshes need little or no element size grading. Here, we present a comparative study of a large number of well-known smoothing algorithms on triangulations of complex geometric domains. Our study reveals the limitations of some well-known smoothing methods. Specifically, the optimal Delaunay triangulation smoothing and weighted centroid of circumcenter smoothing methods are shown to have difficulty achieving smooth grading and adapting to complex domain boundary. We propose modifications and report significant improvements and behavior change in the performance of these algorithms. More importantly, we propose three new smoothing strategies and show their effectiveness in computing premium quality triangulations for complex geometric domains. While the proposed algorithms give the practitioners additional tools to chose from, our comparative study of over a dozen algorithms should guide them selecting the best smoothing method for their particular application.

1 Introduction

Mesh smoothing is an important research problem for scientific simulation applications where high quality elements are desired for accurate numerical calculations. While mesh generation [5, 6, 7, 22] focuses on computing a subdivision (e.g. triangulation) of a given input domain from scratch, mesh smoothing [1, 2, 4, 8, 9, 10, 15, 19, 18, 21, 23, 26, 29, 28, 27] aims to improve an existing subdivision (e.g., for instance by relocating its vertices). Effectiveness of proposed smoothing methods are often shown on meshes of relatively simple domains, e.g., points distributed inside a square box. When the input mesh models a complex geometric domain, the existing smoothing strategies may struggle. For instance, Chen [10] proposed a smoothing method called optimal Delaunay triangulation (ODT) and showed that it is effective for smoothing meshes of simple geometric domains. Figure 1 illustrates

* This research is partially supported by NSF CAREER Award CCF-0846872.



(a) `Initial` $(9.49°, 145.95°)$ (b) `ODT` $(0.59°, 177.84°)$ (c) `Modified ODT` $(12.25°, 131.53°)$

Fig. 1. Smoothing on complex geometry. (a) Initial Iraq mesh; (b) ODT smoothing output with regions depicting suppressed (d) and stretched elements ((e) and (f)); (c) Output of our modified ODT smoothing method. Smallest and largest angles in each mesh are also shown.

the poor performance ODT smoothing when the input geometry is complex. The smoothed mesh contains very many bad quality triangles (suppressed or stretched) near the domain boundary. We later present a modification of ODT smoothing that can handle complex geometric domains (Figure 1 (c) illustrates the output of this improved version).

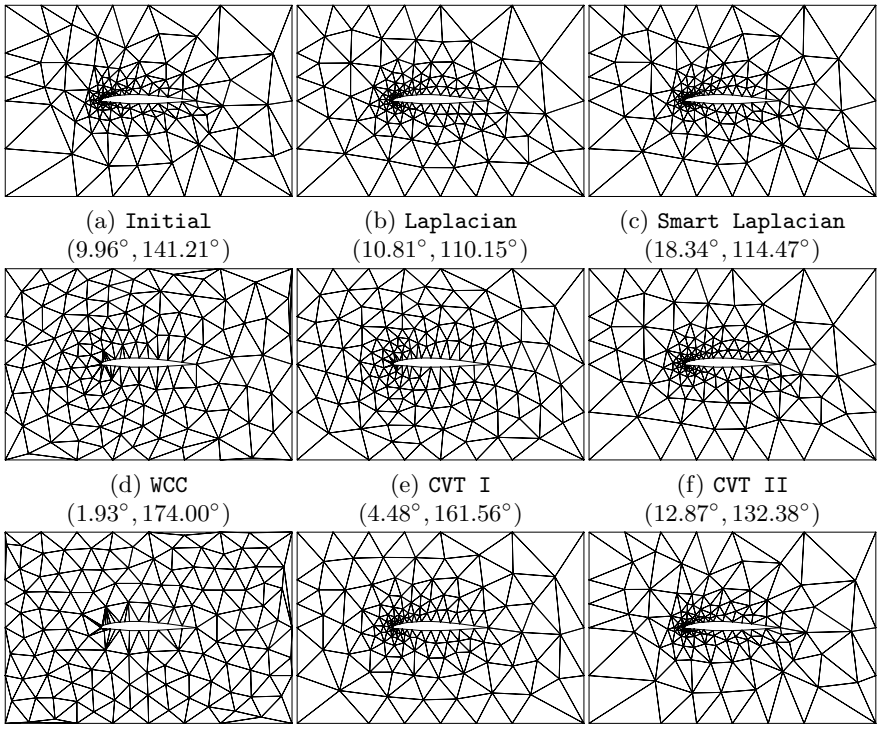
In this paper, we compare the performance of a number of smoothing methods when the input is a complex geometric domain. We propose modifications for some of these techniques to handle complex geometry. In addition, we introduce three new smoothing methods, which are shown to be successful to obtain high quality triangulations on complex domains.

2 Review of Smoothing Algorithms

In this section, we examine a number of smoothing methods, each of which has been shown to be easy to implement, fast and/or effective. Here, we review each approach to better understand their similarities, differences, strengths and drawbacks regarding complex input domains. Note that, in this study we mainly focus on smoothing methods which are based on geometric concepts.

First, we define a vertex $\mathbf{x}_i$ to be *free* if it is allowed to be relocated. The *star* of $\mathbf{x}_i$ is denoted by Ω_i which consists of all triangles incident to $\mathbf{x}_i$. Similarly, the *link* of $\mathbf{x}_i$ consists of all surrounding edges of triangles in the star that are not incident to $\mathbf{x}_i$.

In general, smoothing algorithms iteratively relocate free vertices of a triangulation until a threshold value is satisfied or a specified number of iterations are completed. This is also incorporated with potential edge flipping operations to obtain higher quality triangulations. Now, we describe how to compute a new location for a given vertex based on given algorithms.



(a) `Initial` $(9.96^\circ, 141.21^\circ)$ (b) `Laplacian` $(10.81^\circ, 110.15^\circ)$ (c) `Smart Laplacian` $(18.34^\circ, 114.47^\circ)$

(d) `WCC` $(1.93^\circ, 174.00^\circ)$ (e) `CVT I` $(4.48^\circ, 161.56^\circ)$ (f) `CVT II` $(12.87^\circ, 132.38^\circ)$

(g) `ODT` $(0.76^\circ, 178.11^\circ)$ (h) `Angle-based` $(8.44^\circ, 117.37^\circ)$ (i) `WCT` $(13.10^\circ, 133.64^\circ)$

Fig. 2. (a) Initial airfoil mesh. (b)-(i) Output of existing smoothing algorithms. Smallest and largest angles in each mesh are listed.

2.1 Laplacian Smoothing

In Laplacian smoothing [18], a free vertex is simply relocated to the centroid of the vertices connected to that vertex (see Equation 1). This technique is widely used due to its simplicity and effectiveness. However, it does not guarantee quality improvement where inverted elements might be generated. Therefore, many studies introduced different versions of Laplacian smoothing or combined with other methods. An overview of those can be found in [9].

$$\mathbf{x}^* = \frac{1}{k} \sum_{\mathbf{x}_j \in \Omega_i, \mathbf{x}_j \neq \mathbf{x}_i} \mathbf{x}_j, \tag{1}$$

where Ω_i is the star of the vertex $\mathbf{x}_i$ having k points and $\mathbf{x}^*$ is the new location. Notice that, this formulation can also be interpreted as a torsion spring system [29], where a central node in a star polygon is located at the centroid of the polygon balancing out the system to stay in equilibrium (see also Equation 10).

2.2 Smart Laplacian Smoothing

One of the variations of the standard Laplacian smoothing algorithm is the smart Laplacian smoothing [9, 19] which is also called constrained Laplacian smoothing. The main difference between the original and this variant is relocating points only if there is an improvement in the neighborhood. Briefly, a vertex is relocated when the new candidate location improves the quality of the elements in the star. Otherwise, the node remains in its current location. Note that, the candidate location is the centroid of the surrounding points as before. Cost of evaluating the quality is also added to overall computation, however this method is still quite inexpensive and avoids inverted elements. The quality metric can be chosen based on the application, where we use minimum angle as our quality criterion.

2.3 Centroidal Voronoi Tessellation (CVT)-Based Smoothing

A Voronoi tessellation is called a centroidal Voronoi tessellation (CVT) when its generating points are the centroids of the corresponding Voronoi regions [13]. This special structure has been shown to have applications in many fields, such as image processing, clustering, cell division, and others. Du *et al.* discussed many properties and applications of CVT in [13, 14, 15, 27]. They also showed that the Lloyd iteration [20] which is a minimizer for the energy function given in Equation 2, converges to centroidal Voronoi tessellation.

$$\Psi_{CVT} = \sum_{i=1}^{N} \int_{V_i} \rho(\mathbf{x}) \|\mathbf{x} - \mathbf{x}_i\|^2 d\mathbf{x}, \tag{2}$$

where V_i is the Voronoi region generated by the vertex $\mathbf{x}_i$ and ρ is the density function.

Hence, an immediate algorithm is to apply the Lloyd iteration by iteratively computing the Voronoi regions for each point and updating their locations to the centroid of each region until convergence. In general, the new location can be defined as follows:

$$\mathbf{x}^* = \frac{\int_{V_i} \mathbf{x}\rho(\mathbf{x}) d\mathbf{x}}{\int_{V_i} \mathbf{x} d\mathbf{x}} \tag{3}$$

Note that, the Lloyd iteration is not the only technique to determine CVTs. A discussion on other deterministic and probabilistic methods can be found in [13]. However, this method is naturally suitable for mesh smoothing, although the Lloyd iteration converges slow and hard to analyze [12]. Thus, researchers proposed different smoothing techniques based on centroidal Voronoi tessellation concept [15, 27, 1, 10]. Now, we overview a few which mainly differ in density function and regions leading to less computation time.

Weighted Centroid of Circumcenters (WCC). In this method, Alliez *et al.* [1] proposed to relocate each vertex to a weighted average of corresponding circumcenters, where the weights are based on the physical size of each

simplex. This is same as computing a weighted centroid of the corresponding Voronoi region, where density function is represented by the weights. Then, the new location is represented as follows:

$$\mathbf{x}^* = \frac{1}{|\Omega_i|} \sum_{\tau_j \in \Omega_i} |\tau_j| \mathbf{c}_j, \tag{4}$$

where Ω_i is the star of the point of interest, $\mathbf{c}_j$ is the circumcenter of the simplex τ_j and $|.|$ denotes area in two-dimensions. The circumcenter can be defined as the intersection point of the bisectors of a triangle, which is also the center of the circle that passes through all three vertices of the triangle.

CVT I. In this variant, Chen [10] used the same density function as above, but instead of circumcenters, he proposed to use the centroid of each simplex. This allows to compute the surrounding points, i.e., the region, faster. This smoother can be represented as the following expression:

$$\mathbf{x}^* = \frac{1}{|\Omega_i|} \sum_{\tau_j \in \Omega_i} |\tau_j| \mathbf{x}_{\tau_j}, \tag{5}$$

where $\mathbf{x}_{\tau_j}$ denotes the centroid of the simplex τ_j, which is equal to $\frac{\sum_{\mathbf{x}_k \in \tau_j} \mathbf{x}_k}{n+1}$, while n is the dimension.

CVT II. For non-uniform domains, Chen [10] proposed to incorporate the mesh density function to the calculations. Hence, the smoother can generate an appropriate grading for modulation regions. So, in general by the addition of a density function ρ, Equation 5 becomes

$$\mathbf{x}^* = \frac{\sum_{\tau_j \in \Omega_i} |\tau_j| \mathbf{x}_{\tau_j} \rho_{\tau_j}}{\sum_{\tau_j \in \Omega_i} |\tau_j| \rho_{\tau_j}}. \tag{6}$$

In two-dimensions, when the density function is chosen as $\rho_{\tau_j} = |\tau_j|^{-\frac{n}{2}}$, the above expression is equal to

$$\mathbf{x}^* = \frac{2}{3} \frac{\sum \mathbf{x}_j}{k} + \frac{1}{3} \mathbf{x}_i. \tag{7}$$

Chen described this as a "lumped Laplacian Smoothing", which partially explains the reason behind the effectiveness of the Laplacian smoothing [10].

2.4 Optimal Delaunay Triangulation (ODT)-Based Smoothing

Chen and Xu [11] studied Delaunay triangulations in terms of linear interpolation error for a given function. They showed that when $f(\mathbf{x}) = \|\mathbf{x}\|^2$, the Delaunay triangulation can be referred as optimal in terms of minimizing the interpolation error among all the triangulations for a given number of vertices. This helped Chen to design a new mesh smoothing strategy based

on the optimal Delaunay triangulation (ODT) concept [10], which aims to equally distribute the edge lengths based on the function to be approximated. Basically, the energy function from the interpolation error is exactly solvable, which results in the new location for each query point (see Equation 8).

$$\Psi_{ODT} = \frac{1}{1+n} \sum_{i=1}^{N} \int_{\Omega_i} \|\mathbf{x} - \mathbf{x}_i\|^2 d\mathbf{x}, \tag{8}$$

where Ω_i is the star of the point of interest $\mathbf{x}_i$, while n represents the dimension. For a regular-shaped uniform mesh generation, the function to be interpolated can be chosen as $f(\mathbf{x}) = \|\mathbf{x}\|^2$. Then, the new location can be computed as in Equation 9. Note that, convergence rate is slower while the time for each relocation calculation is increased compared to the others.

$$\mathbf{x}^* = -\frac{1}{2|\Omega_i|} \sum_{\tau_j \in \Omega_i} (\nabla|\tau_j(\mathbf{x})| \sum_{\mathbf{x}_k \in \tau_j, \mathbf{x}_k \neq \mathbf{x}_i} \|\mathbf{x}\|^2). \tag{9}$$

2.5 Angle-Based Smoothing (AB)

Zhou and Shimada [29] proposed a smoothing method which aims to improve the geometric quality of a mesh in a computationally easy way. Their method has been shown to provide better results than Laplacian smoothing in quality and to avoid creating inverted elements. They formulated local neighborhood of a vertex as a torsion spring system similar to Laplacian smoothing. (See Equations 10 and 11.)

$$\Psi_{LAPLACIAN} = \sum_{j=1}^{k} \frac{1}{2} K \|\mathbf{v}_j\|^2, \tag{10}$$

$$\Psi_{ANGLE} = \sum_{j=1}^{2k} \frac{1}{2} K {\theta_j}^2, \tag{11}$$

where k is number of nodes in the star of the vertex $\mathbf{x}_i$, K is the spring constant, $\mathbf{v}_j$ is the vector from $\mathbf{x}_i$ to every surrounding node and θ_j is the angle formed by each side of the polygon and the central point $\mathbf{x}_i$.

However, their spring system is based on angles instead of distances. The system has its minimum potential energy when the new location for the central node yields to an optimum angle distribution on the angles formed by the each side of the star. In order to find the optimum location, following two approaches have been introduced.

Original Version. Zhou and Shimada [29] introduced a heuristic method which gives an approximate position for the optimum location. A four-step procedure, summarized below, is followed to find the new location for each free vertex. Note that, this method is more expensive than Laplacian smoothing, but easier and faster than optimization techniques.

Given a vertex $\mathbf{x}_i$ and its corresponding neighborhood (star) and link, the vertices of the link is denoted by $\mathbf{y}_j$, $j = \{1, 2, ..., k\}$ in counterclockwise order.

1. Compute the side angles $\alpha_{j_1} = \angle \mathbf{x}_i \mathbf{y}_j \mathbf{y}_{j+1}$ and $\alpha_{j_2} = \angle \mathbf{y}_{j-1} \mathbf{y}_j \mathbf{x}_i$.
2. Compute the difference between each adjacent angles: $\beta_j = (\alpha_{j_2} - \alpha_{j_1})/2$.
3. Compute the new location $\mathbf{x}_i^j$ for the central node $\mathbf{x}_i$ by a vector rotation based on β_j, $\mathbf{y}_j$ and $\mathbf{x}_i$, while keeping the size of the vector from $\mathbf{y}_j$ to $\mathbf{x}_i$.
4. Compute the average of the new location suggestions for the final location: $\mathbf{x}^* = \frac{1}{k} \sum_{j=1}^{k} \mathbf{x}_i^j$

Optimization Version. Xu and Newman [28] approached to the same idea of angle-based smoothing from an optimization point of view. They sacrificed computation time to have better quality results. Instead of heuristically calculating an approximate new location, they used optimization techniques to find a close optimum. The same torsion spring system is formulated as an optimization problem and the Gauss-Newton approximation has been used to solve the least-squares formulation of the following objective function:

$$s = \sum_{j=1}^{k} [distance(\mathbf{x}^*, l_j)], \tag{12}$$

where $\mathbf{x}^*$ is the optimized location of the query point $\mathbf{x}_i$, and l_j is the angle bisector line for each side angle of the star Ω_i. This method has been shown to produce slightly better quality meshes than original angle-based method and also shown to converge faster. However, overall computation can take more than the heuristic version, although the number of iterations is less. Therefore, we will use the original version of this method in our experiments due to its simplicity, efficiency and availability.

2.6 Well-Centered Triangulation (WCT) Smoothing

Another iterative method has been proposed by Vanderzee *et al.* in [26], mainly focusing on providing well-centered triangulations (WCT), i.e., acute triangulations in the plane. They introduced a global energy function, which aims to minimize the maximum angle of a mesh M. They also claim that their energy function penalizes small angles as well.

$$\Psi_{WCT} = \sum_{\theta \in M} |cos(\theta) - 1/2|^p, \tag{13}$$

where p is a finite number (it is sufficient to set $p \in \{4, 6, 8\}$) and θ represents each angle in the triangulation M. This algorithm is computationally more expensive, however considering this approach concentrates on maximum angle, it would be interesting to observe its behavior among others. We should

also note that, this algorithm preserves the topological connections unlike the above mentioned methods where they incorporate edge flip operations into smoothing.

3 New Smoothing Algorithms

We first give modifications to two of the methods described in the previous section, so that they handle complex geometric input domains. Then, we propose three new smoothing algorithms in Sections 3.2, 3.3 and 3.4.

3.1 Modifying Existing Algorithms for Complex Geometry

As we mentioned, when the input domain has a complex geometry, most of the known smoothing methods struggle and generate bad quality elements (see Figure 1 and 2). In particular, although the output triangulation overall contains good quality elements, the elements which are close to the boundary, are problematic. We observed this boundary problem especially in ODT and WCC methods. Here, we propose simple modifications for each algorithm to handle boundary issues to an extend.

ODT with boundary fix via incenters. ODT smoothing approach [10] tends to create bad quality elements close to input features (see Figures 1, 2 (g) and 3). The original ODT algorithm relocates a vertex $\mathbf{x}_i$, by exactly solving an optimization problem within the star of $\mathbf{x}_i$ and trying to achieve uniformity in the edge lengths of the triangles. Since we consider the boundary vertices as input features representing the complex domain and chosen not to be relocated, this approach causes some triangles near the boundary to be stretched or suppressed (see Figure 3). As a modification, we now perform the same relocation strategy within the star-shaped polygon formed by the incenters of the triangles in the star of $\mathbf{x}_i$. This way, we aim to make the distances of the relocated vertex to the star vertices proportional to the length of the corresponding edges on the star. Experiments indicate that this idea works fairly well in producing graded and quality triangulations. Therefore, we suggest to use a patch formed by the incenters of the connected elements instead of elements themselves (see Figures 1, 3, 5 and 7).

WCC with boundary push. WCC smoothing approach [1] has the similar boundary issues (see Figure 2 (g)). This is partly because the algorithm tries to iteratively reach a regular-shaped centroidal Voronoi tessellation, where in case of complex geometric domains it fails to consider the input features. In order to alleviate this problem, while calculating the new location, we propose to use incenters of the triangles whose circumcenters are located outside the boundary. Note that, by definition an incenter is always located inside of the triangle. This idea reasonably eliminates suppressed elements and increase the quality, however stretched elements can remain near the boundary (see Figures 5 and 7). Therefore, we explore other smoothing ideas as we propose next.

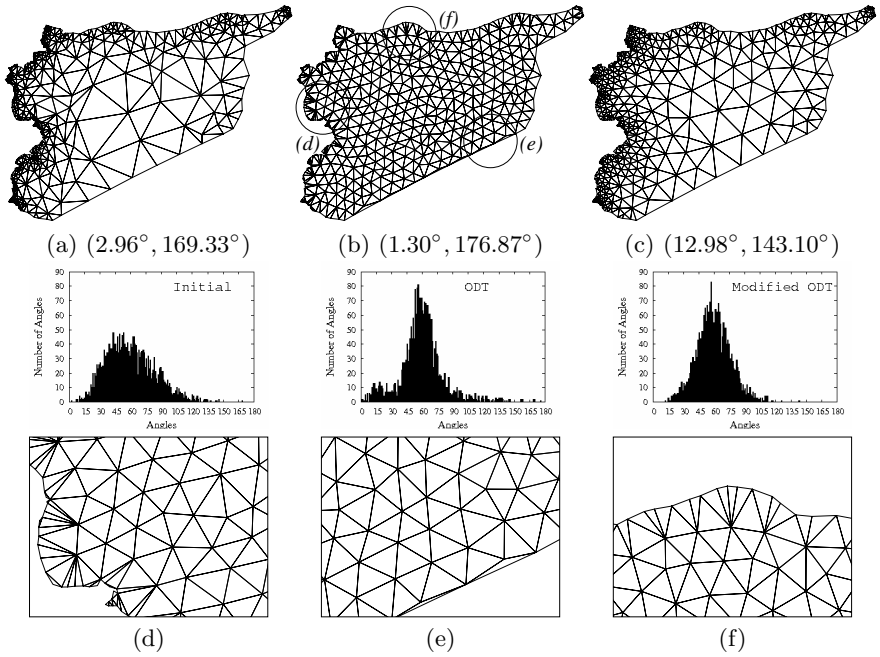


Fig. 3. (a) Initial Syria mesh and histogram of angles. (b) Output and histogram of angles after 50 iterations of ODT applied on (a). (c) Output and histogram of angles after applying 10 iterations of modified ODT on (a). ODT output is zoomed in for stretched ((d) and (f)) and suppressed (e) elements.

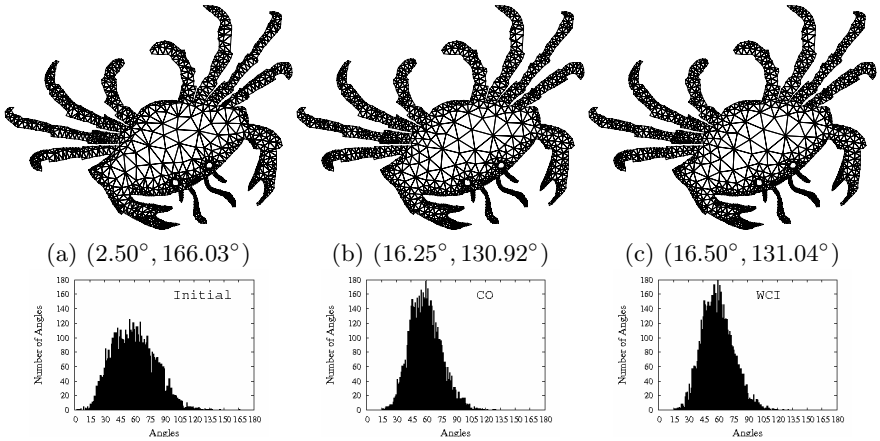


Fig. 4. (a) Initial crab mesh and histogram of angles. (b) Output and histogram of angles after 10 iterations of CO smoothing applied on (a). (c) Output and histogram of angles after applying 10 iterations of WCI on (a).

3.2 Centroid of Off-Centers (CO)

Off-centers [25] have been introduced to improve Delaunay refinement algorithms as another type of Steiner point alternative to circumcenters. Given a triangle with the shortest edge pq, the off-center is a point o on the bisector of pq, which is furthest from p (or q) such that the angle $\angle poq$ is α.

Here, we introduce a new method to benefit from off-centers for smoothing. Given a point to be relocated, similar to WCC method, with an additional quality constraint, we compute the off-center of each simplex connected to that point. Then, we calculate the centroid of this region formed by the computed centers and relocate our point to that location. In our experiments, we chose a reasonably large α value for computing the off-centers and obtained good results (see Figures 4, 5 and 7). Note that, here the uniformity constraint is lifted, however a weighted version of this method (WCO) is also explored and observed to have a similar behavior to WCC. WCO is also prone to boundary problems, but off-centers help to reduce.

3.3 Distance Weighted Centroid of Incenters (WCI)

The incenter of a triangle is the intersection point of the angle bisectors, which is the center of the largest circle that fits inside the triangle. Hence, the location of the incenter of a triangle is determined by the angles and edge lengths of that element, while remaining always inside of the element. Here, we relocate free vertices to the centroid of the incenters of the elements in their stars. A weighted version of this idea have been introduced as part of an hybrid smoothing approach for mixed meshes which classifies elements based on their local neighborhoods and apply different smoothing algorithms accordingly [21]. Hence, WCI is used only for specific types of elements. However, it has not been explored as a sole triangular mesh smoother. As in some other smoothing methods (see Section 2) a central node and its star is formulated as a torsion spring model, where relocation point is computed based on energy minimization.

$$\Psi_{INCENTER} = \sum_{j=1}^{k} \frac{1}{2} K \|\mathbf{z}_j\|^2, \tag{14}$$

where $\mathbf{z}_j$ denotes the vector from the central node $\mathbf{x}_i$ to the incenter of each connected triangle. Then, the new location is calculated as follows:

$$\mathbf{x}^* = \sum_{\tau_j \in \Omega_i} w_{\tau_j} \mathbf{p}_{\tau_j}, \tag{15}$$

where $\mathbf{p}_{\tau_j}$ is the incenter of the simplex τ_j and weight $w_{\tau_j} = \frac{\|\mathbf{z}_j\|}{\sum_{t=1}^{k} \|\mathbf{z}_j\|}$. Note that, here the weight function is different than the previously mentioned methods. Our experiments show that this smoothing method achieves good quality bounds while generating proper grading (see Figure 4, 5 and 7).

3.4 Sliced-Petal Smoothing

Among many mesh quality criteria [24], minimum angle quality is widely popular due to its direct influence on applications [3, 24]. Here, we designed a new smoothing method to obtain large minimum angles for given triangulations. In our method, we guarantee to preserve and improve (if possible) the initial minimum angle while smoothing the mesh elements. As can be seen from Figure 2, this is not always true for other smoothing methods. They potentially reduce the minimum angle, in particular, for complex geometric domains.

Our method relies on the geometric concepts that have been introduced in [17, 16], where a similar relocation strategy has been used as part of the Delaunay refinement process.

Consider a bad quality triangle pqr (where its minimum angle does not satisfy the angle constraint α) in a triangulation with its shortest edge pq. Let α-$petal(pq)$ be the disk bounded by the circle that goes through p, q and a third point y such that y and r are on the same side of pq and $\angle pyq = \alpha$ [17, 16]. Then, α-$slice(pq)$ is defined as the intersection of the α-$petal(pq)$ and the region between the two lines one going through p, the other q where both making an angle α to line segment pq. Note that, the α-slice is the feasible region, where r needs to be relocated in order to make all the angles of pqr greater than or equal to α. Here, due to space restriction we omit descriptive figures, however the reader can refer to the figures in [17, 16].

Given a vertex to be relocated $\mathbf{x}_i$ and its star, in order to have minimum angles of all elements inside the star to be at least α, the intersection of the α-slices of the edges on the link must be non-empty. Based on this observation, first we search for approximately the best possible α value. We start our search with an initial α value which is determined according to the current minimum angle of the star. After the search operation, we construct the feasible relocation region. In order to find the best location inside this region for $\mathbf{x}_i$ which strictly improves the local quality, we check sufficiently large number of sample points. If such a location could not be found, simply the point location is kept the same. Thus, every local relocation definitely improves the overall quality of the mesh.

Compared to the other smoothing methods, our results indicate a considerable amount of improvement in quality (see Figures 5, 6, 7 and 8). Clearly, this method requires more computation time than many of the algorithms mentioned, however it successfully handles complex geometric domains with proper grading and has high convergence rate. Although our method obtains small maximum angle values, our ultimate goal is to incorporate the maximum angle constraint (γ) to our relocation strategy as in [16].

4 Results and Discussions

Results of our experiments have been given throughout the paper, however in this section we compare all smoothing algorithms that are reviewed and

introduced. We generated several experimental set-ups which include various input domains in different complexities. We discuss the results mostly based on the distribution of angles and edge lengths, uniformity, minimum and maximum angle and time.

4.1 Implementation and Data Sets

The algorithms discussed in this paper are all implemented in the same platform and using the C programming language. The examples and data sets presented in this limited space are only a small but representative subset of our extensive experimental study. The output triangulations (Figures 5 and 7) are accompanied by the histogram of angles in each case (Figures 6 and 8) and the time performance is summarized in Table 2.

4.2 Experiments

Our experimental study is conducted on a large number of data sets including the representative China and Hawaii maps, which have 1158 and 3216 points and 1876 and 5024 triangles, respectively.

In Table 1, we give an overall qualitative comparison of smoothing methods experimented in this study. These classifications have been obtained based on the results of our qualitative visual experiments. In particular, ease of implementation and speed incorporate our experimental observations and the main idea of each approach, where the speed levels are determined according to have almost an order of magnitude difference. Minimum and maximum

Table 1. A qualitative comparison of the known and the proposed smoothing methods, where we label the performance of each method on a scale of 3, as 1. Poor, 2. Medium, or 3. Good.

Smoothing Method	Ease of Implementation	Speed	Quality α	Quality γ	Grading
Laplacian [18]	Good	Good	Medium	Medium	Good
Smart Lap. [9, 19]	Good	Good	Medium	Good	Good
WCC [1]	Medium	Good	Poor	Poor	Medium
CVT I [10]	Good	Good	Medium	Medium	Good
CVT II [10]	Good	Good	Medium	Medium	Good
ODT [10]	Medium	Medium	Poor	Poor	Poor
AB [29]	Medium	Good	Medium	Good	Medium
WCT [26]	Poor	Poor	Poor	Medium	Good
M. ODT	Medium	Good	Medium	Medium	Good
M. WCC	Medium	Good	Medium	Medium	Medium
WCO	Medium	Good	Medium	Medium	Medium
CO	Medium	Good	Medium	Medium	Good
WCI [21]	Medium	Good	Medium	Medium	Good
Sliced-petal	Medium	Medium	Good	Good	Good

Table 2. Time comparison between the known and the proposed methods.

Smoothing Method	China Quality (α, γ)	China Total Time (msec)	Hawaii Quality (α, γ)	Hawaii Total Time (msec)
Initial	(8.82°,161.47°)	N/A	(5.80°,164.76°)	N/A
Laplacian [18]	(22.02°,117.83°)	377	(6.88°,151.59°)	1447
Smart Lap. [9, 19]	(23.60°,121.84°)	420	(13.92°,148.32°)	1988
WCC [1]	(10.54°,138.30°)	382	(0.14°,179.72°)	1866
CVT I [10]	(12.88°,138.30°)	449	(0.70°,166.35°)	1872
CVT II [10]	(21.13°,115.58°)	415	(8.27°,145.28°)	1849
ODT [10]	(0.72°,177.96°)	2073	(0.01°,179.97°)	18669
AB [29]	(18.54°,114.92°)	407	(5.44°,154.95°)	2053
WCT [26]	(15.24°,118.58°)	8129	(1.96°,168.18°)	21534
M. ODT	(17.15°,117.00°)	415	(2.12°,161.30°)	1984
M. WCC	(10.54°,138.30°)	996	(0.58°,168.71°)	13112
WCO	(10.14°,144.60°)	483	(0.40°,171.51°)	2225
CO	(22.02°,117.83°)	443	(6.88°,151.59°)	1938
WCI [21]	(21.88°,118.09°)	421	(7.50°,148.32°)	1839
Sliced-petal	(30.48°,113.42°)	1550	(18.98°,133.73°)	5347

angle (α and γ respectively) quality is categorized based on the consistency of the methods. Average performance has been chosen as medium level while the others indicate notably better or worse performance. Grading values reflect the overall appearence of the output triangulations, where three different classes naturally emerge.

We observed that the complexity of geometric domains plays a significant role in the performance of the smoothing methods. For complex geometric domains, methods incorporating input features into their relocation strategy perform well. On the other hand, algorithms that aim to converge to lattice structures have trouble computing graded meshes. Our proposed sliced-petal method is shown to be performing considerably better than others in quality, while smart Laplacian method obtains good results as well. However, its behavior is not as stable as our sliced-petal method.

Quality. Figures 5 and 7 together with the histograms in Figures 6 and 8 show the performance of smoothing algorithms on complex geometric domains. Here, Laplacian and smart Laplacian methods are shown to be performing well on complex geometric domains, however ODT and WCC methods have clear boundary problems. Our modified versions alleviate the problem considerably for ODT and reduce it for WCC. The CO and WCI methods behave similar to the angle-based smoothing method, whereas our sliced-petal smoothing method outperforms them all. Both the smallest and the largest angle values observed in the output of the sliced-petal smoothing method is significantly better than those of the other methods. Histograms also support our observations by showing the angle quality of each element.

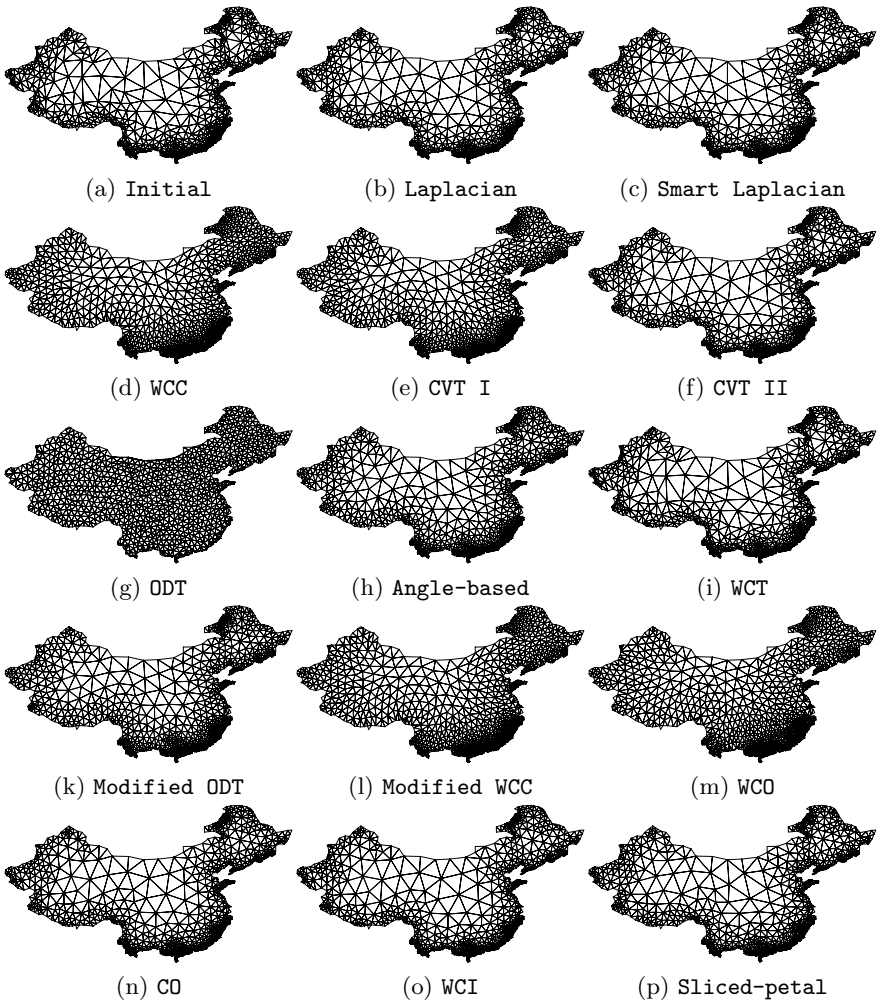
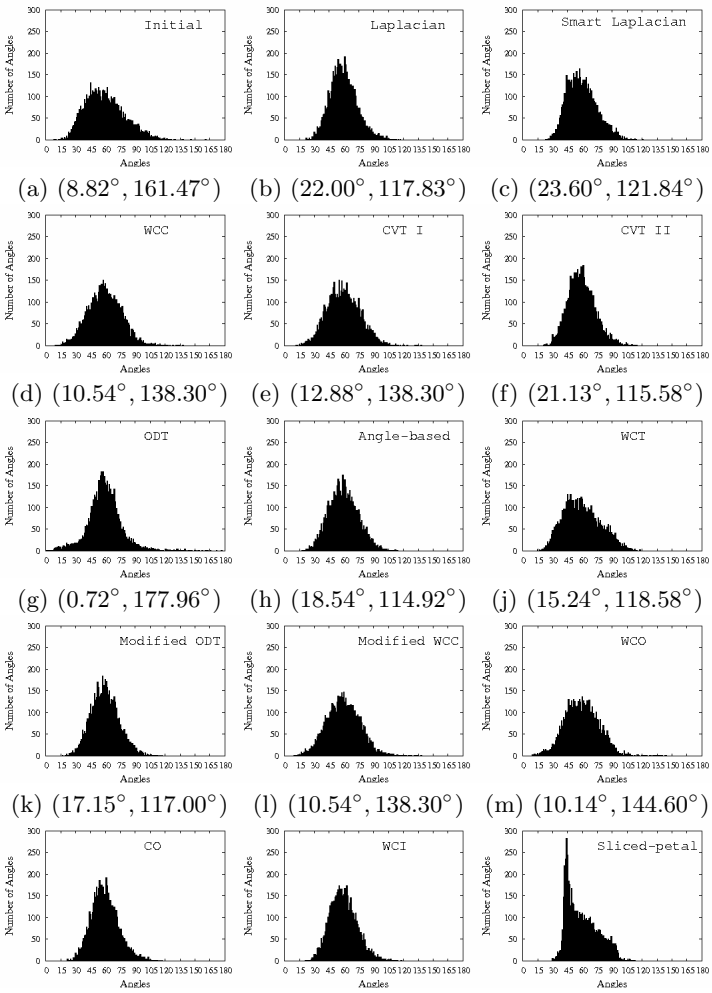


Fig. 5. (a) Initial China mesh. (b)-(j) Output of the existing smoothing methods. (k)-(p) Output of the proposed smoothing methods.

Time. In Table 2, we report the total execution time for each smoothing method. As can be seen, WCT and ODT take significantly more time than the others. In case of WCT, computing the energy function is costly, whereas for ODT convergence takes time. Mainly, due to the current implementation of the feasible region calculation, our sliced-petal method is slower than the remaining. As we expected, the Laplacian smoothing is the fastest, whereas others perform similar to each other.



(a) $(8.82°, 161.47°)$ (b) $(22.00°, 117.83°)$ (c) $(23.60°, 121.84°)$

(d) $(10.54°, 138.30°)$ (e) $(12.88°, 138.30°)$ (f) $(21.13°, 115.58°)$

(g) $(0.72°, 177.96°)$ (h) $(18.54°, 114.92°)$ (j) $(15.24°, 118.58°)$

(k) $(17.15°, 117.00°)$ (l) $(10.54°, 138.30°)$ (m) $(10.14°, 144.60°)$

(n) $(22.02°, 117.83°)$ (o) $(21.88°, 118.09°)$ (p) $(30.48°, 113.42°)$

Fig. 6. (a) Initial China mesh histogram. (b)-(j) Output histograms of the existing smoothing methods. (k)-(p) Output histograms of the proposed smoothing methods.

Grading. Experiments show that smoothing on complex geometric domains mostly requires proper grading in order to achieve high quality triangulations. As we pointed out, smoothing methods designed to reach regular meshes fails to conform to the boundary (see Figures 5 and 7). While others achieve proper grading as well, our sliced-petal method converges to significantly higher quality meshes than the existing methods.

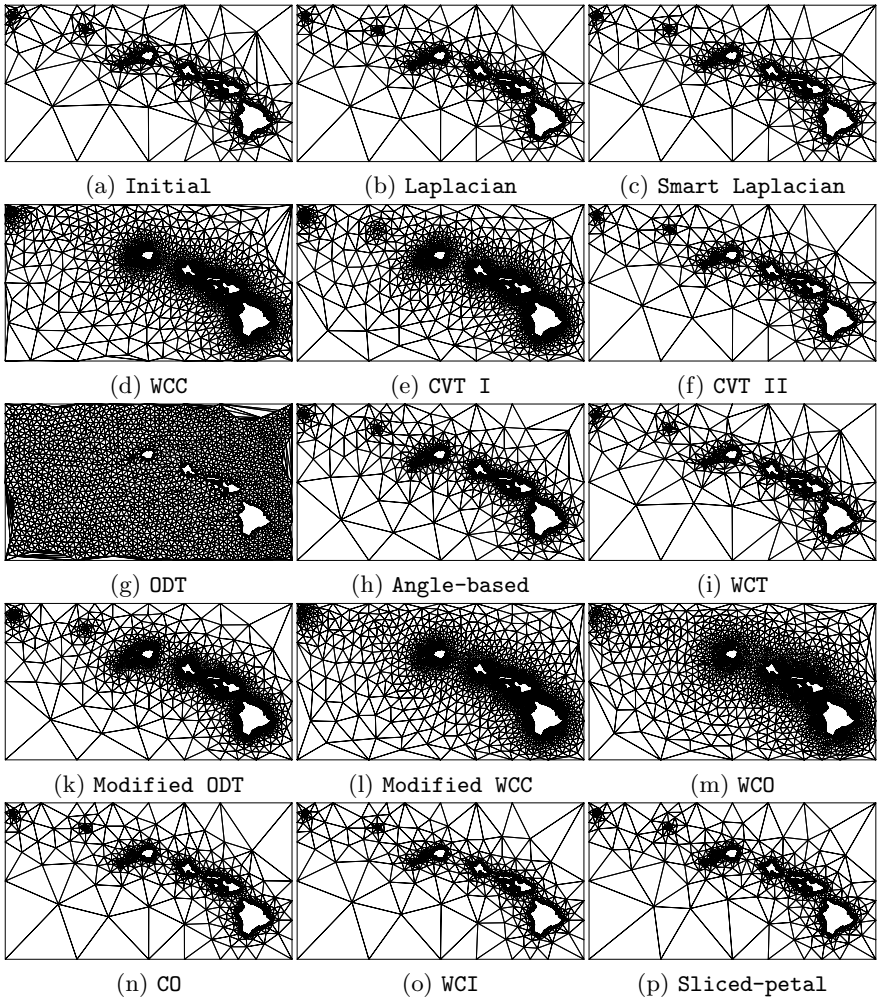
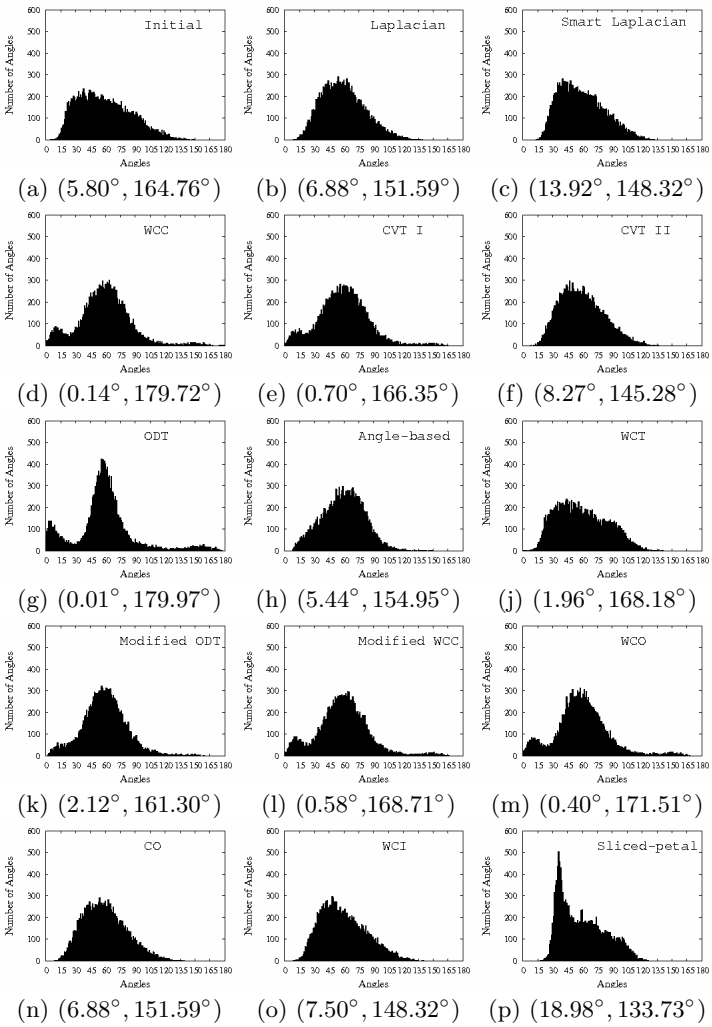


Fig. 7. (a) Initial Hawaii mesh. (b)-(j) Output of the existing smoothing methods. (k)-(p) Output of the proposed smoothing methods.

4.3 Future Work

Combining smoothing methods to benefit from their properties remains as future work. A thorough convergence study on these algorithms and a similar study on quadrilateral/three-dimensional meshes would be helpful for future studies. Our sliced-petal smoothing performance can easily be improved by careful implementation and search strategies. We believe this study will be useful for applications involving complex geometric domains, especially in choosing the appropriate smoothing method.



(a) $(5.80°, 164.76°)$ (b) $(6.88°, 151.59°)$ (c) $(13.92°, 148.32°)$

(d) $(0.14°, 179.72°)$ (e) $(0.70°, 166.35°)$ (f) $(8.27°, 145.28°)$

(g) $(0.01°, 179.97°)$ (h) $(5.44°, 154.95°)$ (j) $(1.96°, 168.18°)$

(k) $(2.12°, 161.30°)$ (l) $(0.58°, 168.71°)$ (m) $(0.40°, 171.51°)$

(n) $(6.88°, 151.59°)$ (o) $(7.50°, 148.32°)$ (p) $(18.98°, 133.73°)$

Fig. 8. (a) Initial Hawaii mesh histogram. (b)-(j) Output histogram of the existing smoothing methods. (k)-(p) Output histogram of the proposed smoothing methods.

References

1. Alliez, P., Cohen-Steiner, D., Yvinec, M., Desbrun, M.: Variational tetrahedral meshing. ACM Transactions on Graphics 24, 617–625 (2005)
2. Amenta, N., Bern, M.W., Eppstein, D.: Optimal point placement for mesh smoothing. In: ACM-SIAM Symp. Discrete Algorithms, pp. 528–537 (1997)
3. Babuška, I., Aziz, A.: On the angle condition in the finite element method. SIAM J. Numer. Analysis 13, 214–227 (1976)
4. Bank, R.E., Smith, R.K.: Mesh smoothing using A posteriori error estimates. SIAM Journal on Numerical Analysis 34(3), 979–997 (1997)

5. Bern, M., Eppstein, D.: Mesh generation and optimal triangulation. Computing in Euclidean Geometry, 23–90 (1992)
6. Bern, M., Eppstein, D., Gilbert, J.R.: Provably good mesh generation. J. Comp. System Sciences 48, 384–409 (1994)
7. Bern, M., Plassmann, P.: Mesh generation. In: Sack, J.-R., Urrutia, J. (eds.) Handbook of Computational Geometry, pp. 291–332. Elsevier, Amsterdam (1998)
8. Canann, S.A., Tristano, J.R., Blacker, T.: Opti-smoothing: An optimization-driven approach to mesh smoothing. Finite Elements in Analysis and Design 13, 185–190 (1993)
9. Canann, S.A., Tristano, J.R., Staten, M.L.: An approach to combined laplacian and optimization-based smoothing for triangular, quadrilateral, and quad-dominant meshes. In: 7th Int. Meshing Roundtable, pp. 479–494 (1998)
10. Chen, L.: Mesh smoothing schemes based on optimal Delaunay triangulations. In: 13th Int. Meshing Roundtable, pp. 109–120 (2004)
11. Chen, L., Xu, J.: Optimal Delaunay triangulation, Journal of Computational Mathematics 22, 299–308 (2004)
12. Du, Q., Emelianenko, M., Ju, L.: Convergence of the Lloyd algorithm for computing centroidal Voronoi tessellations. SIAM. J. Numer. Anal. 44(1), 102–119 (2006)
13. Du, Q., Faber, V., Gunzburger, M.: Centroidal Voronoi tessellations: Applications and algorithms. SIAM Review 41, 637–676 (1999)
14. Du, Q., Gunzburger, M.: Grid generation and optimization based on centroidal Voronoi tessellations. Appl. Math. Comp. 133, 591–607 (2002)
15. Du, Q., Wang, D.: Tetrahedral mesh generation and optimization based on centroidal Voronoi tessellations. Inter. J. Numer. Meth. Eng. 56(9), 1355–1373 (2002)
16. Erten, H., Üngör, A.: Computing no small no large angle triangulations, To be appear in Proc. Int. Symp. Voronoi Diagrams (2009)
17. Erten, H., Üngör, A.: Quality triangulations with locally optimal steiner points. SIAM Journal of Scientfic Computing 31(3), 2103–2130 (2009)
18. Field, D.A.: Laplacian smoothing and Delaunay triangulations. Comm. in Applied Numer. Analysis 4, 709–712 (1988)
19. Freitag, L.A.: On combining laplacian and optimization-based mesh smoothing techniques. In: Trends in Unstructured Mesh Generation, pp. 37–43 (1997)
20. Lloyd, S.: Least square quatization in PCM. IEEE Trans. Inform. Theory 28, 129–137 (1982)
21. Mukherjee, N.: A hybrid, variational 3D smoother for orphaned shell meshes. In: 11th Int. Meshing Roundtable, pp. 379–390 (2002)
22. Owen, S.: A survey of unstructured mesh generation technology. In: 7th Int. Meshing Roundtable, pp. 239–267 (1998)
23. Parthasarathy, V.N., Kodiyalam, S.: A constrained optimization approach to finite element mesh smoothing. Finite Elements in Analysis and Design 9, 309–320 (1991)
24. Shewchuk, J.R.: What is a good linear element? interpolation, conditioning, and quality measures. In: 11th Int. Meshing Roundtable, pp. 115–126 (2002)
25. Üngör, A.: Off-centers: A new type of steiner points for computing size-optimal quality-guaranteed Delaunay triangulations. Comput. Geom. Theory Appl. 42(2), 109–118 (2009)

26. VanderZee, E., Hirani, A.N., Guoy, D., Ramos, E.: Well-centered planar triangulation–an iterative approach. In: 16th Int. Meshing Roundtable, pp. 121–138 (2007)
27. Wang, D., Du, Q.: Mesh optimization based on the centroidal Voronoi tessellation. Int. J. Numer. Anal. Mod. 2, 100–113 (2005)
28. Xu, H., Newman, T.S.: An angle-based optimization approach for 2d finite element mesh smoothing. Finite Elem. Anal. Des. 42(13), 1150–1164 (2006)
29. Zhou, T., Shimada, K.: An angle-based approach to two-dimensional mesh smoothing. In: 9th Int. Meshing Roundtable, pp. 373–384 (2000)