# Size Competitive Meshing without Large Angles [*]

Gary L. Miller Todd Phillips Donald Sheehy
Computer Science Department,
Carnegie Mellon University, Pittsburgh, PA 15213
{glmiller,tp517,dsheehy}@cs.cmu.edu

February 2, 2007

**Abstract**

We present a new meshing algorithm for the plane, Overlay Stitch Meshing (OSM), that accepts as input an arbitrary Planar Straight Line Graph and produces a triangulation with all angles smaller than $170°$. The output triangulation has size that is competitive with any optimal size mesh having bounded largest angle. The competitive ratio is $O(\log(L/s))$ where $L$ and $s$ are respectively the largest and smallest features in the input. OSM runs in $O(n \log(L/s) + m)$ time/work where $n$ is the input size and $m$ is the output size. The algorithm first uses Sparse Voronoi Refinement to compute a quality mesh of the input points alone. This triangulation is then combined with the input edges to give the final mesh.

## 1 Introduction

The meshing problem is to take as input a domain containing a collection of features and return a triangulation of the domain. In 2D, the features are simply points and non-crossing edges; a planar straight line graph. The design of a meshing algorithm involves analysis of four fundamental properties of the algorithm and its outputs. First, the output mesh must be **conforming**, so that all the vertices and edges should appear as a union of simplices in the final mesh. Secondly, all the triangular elements should have some guarantee of **element quality**. Third, the number of output triangles should be asymptotically competitive with any optimal triangulation that is conforming and good quality, so that we have an output size **approximation algorithm**. Finally the algorithm should be fast, **work efficient**. This paper will only be concerned with the 2D meshing problem.

The 2D meshing problem was first posed by Bern, Eppstein, and Gilbert[BEG94] who proposed a quadtree algorithm. Ruppert[Rup95a] gave an $O(n^2)$ time constant size approximation algorithm for the meshing problem using Delaunay refinement. Mitchell and Vavasis[MV92] extended the quadtree algorithm to 3D and proved that the Bern, Eppstein, and Gilbert algorithm was in fact also a constant size approximation algorithm. In order for these algorithms to be constant factor approximation algorithms, two critical assumptions are made. First, element quality is defined by the absence of arbitrarily small angles in any triangle. Secondly, an input PSLG with small angles is not permitted. Our main goal is to develop algorithms and techniques that will allow an arbitrary PSLG as input, relaxing the latter assumption. Note that this also forces us to abandon the former assumption, since any small input angle will require a small output angle to conform. Other small output angles will also be necessary; Shewchuk [She02] gives an extensive discussion of how small input angles force small output angles.

An alternative setting employs a definition of element quality that allows small angles, but bounds all angles strictly away from $180°$, prohibiting arbitrarily large angles. Many meshing applications require only this weaker quality guarantee. Babuška and Aziz showed that only large angles effect interpolation error [BA76, GMW99], while Boman, Hendrickson, and Vavasis showed that only large angles effect their reduction of a elliptic problem to the solution of a Laplacian problem [BHV04, MV05]. An algorithm operating under weaker quality constraints is accordingly able to produce meshes with fewer added vertices (in theory and practice).

[*]This work was supported in part by the National Science Foundation under grants CCR-0122581.

In this work, we present and analyze the Overlay Stitch Meshing algorithm (OSM), a new no-large-angle meshing algorithm for conforming to an arbitrary PSLG in 2D. For practical inputs, OSM achieves the best known competitive guarantees on output size relative to the optimal no-large-angle conforming mesh. The algorithm outputs a mesh with no angle larger than $170°$. It is also noteworthy as a new paradigm for Delaunay-type meshing algorithms that are able to accept inputs with arbitrarily small angles. In addition to having good theoretical guarantees, the algorithm is remarkably straightforward and runs in time $O(n \log(L/s) + m)$.

## 1.1 Preliminaries

Some analytic results will be dependent on the geometric conditioning of the input features. In keeping with the literature, we will use the **spread** $(L/s)$ of the input as a condition. The spread is the ratio of the diameter of the input $(L)$ to the smallest distance between any two disjoint features of the input $(s)$. The definition of the spread intuitively captures the total amount of geometric grading in the input that must be resolved by a quality mesh.

We will also use the standard **gap-ratio** $(\Gamma_M(x))$ at a location $x$ relative to a point set $M$. The gap-ratio can be written as $R(x)/r(x)$, where $R(x)$ is the radius of the largest disc containing $x$ but not intersecting $M \backslash x$, and $r(x)$ is the nearest neighbor of $x$ in $M \backslash x$. At the boundary of our geometric domain, we require that $x$ and the center of the largest disc be contained in the convex closure of $M$. For shorthand, we say a mesh $M$ has gap-ratio quality $\Gamma$ if:

$$\forall x, \Gamma_M(x) \leq \Gamma$$

Note that if a mesh $M$ has some constant gap-ratio quality $\Gamma$, then it has no arbitrarily small or large angles. The output of overlay meshing will subsequently *not* have gap-ratio quality (since it may contain small angles), but some intermediates will have bounded $\Gamma$.

Also of crucial importance in analyzing meshing algorithms is the **local feature size** (lfs). The local feature size at a point $x$ relative to a PSLG $M$, denoted $\mathrm{lfs}_M(x)$, is given by the radius of the smallest disc centered at $x$ that intersects two disjoint features (segments or vertices) of $M$. When $M$ is suppressed, $\mathrm{lfs}(x)$ will be with reference to the *input* PSLG. Also of note is $\mathrm{lfs}_0(x)$, the radius of an identically defined disc intersecting two *vertices* of the input, thus ignoring the presence of segments.

**Theorem 1.** *On input a PSLG Algorithm Overlay Stitch Meshing generates a conforming triangulation $\mathcal{T}$ with no angle greater than $170^0$. The size of $|\mathcal{T}| \leq c \log(L/S) \cdot OPT$ where $OPT$ is the size of an optimal conforming triangulation with all angle bound away from $180^o$ and c is some fixed constant.*

## 1.2 Related Work

The present work derives from two disjoint lines of past research, well-graded meshes and no-large-angle triangulation. Both lines of research trace their lineage back to the same motivating problem, that of producing quality meshes for finite-element simulations. However, the methods, guarantees, and final output provided by each field are drastically different.

Well-graded meshing algorithms attempt to produce meshes in which the length of each edge is proportional to the lfs at its end points. These algorithms generally fall into two categories, structured and unstructured, as typified by Quadtree methods [BEG94] and Delaunay refinement methods [Rup95b, She02] respectively.

One significant property of most well-graded meshing algorithms is that they provide guarantees regarding both the largest and the smallest angles in the resulting mesh. A bound on the smallest angle is a sufficient but not necessary condition for a mesh to have no large angles. However, many Delaunay refinement algorithms are only guaranteed to terminate for inputs where all input angles are at least $60°$. Several algorithms avoid this restriction by severely weakening the smallest output angle guarantee in order to guarantee termination [Pav03].

Significant research has been done on how to extend well-graded meshing algorithms to efficiently handle small input angles, most of which is based on extending the "concentric shelling" method proposed by Ruppert in [Rup95a]. The main idea requires that if two edges share a vertex, then the splitting of these edges should

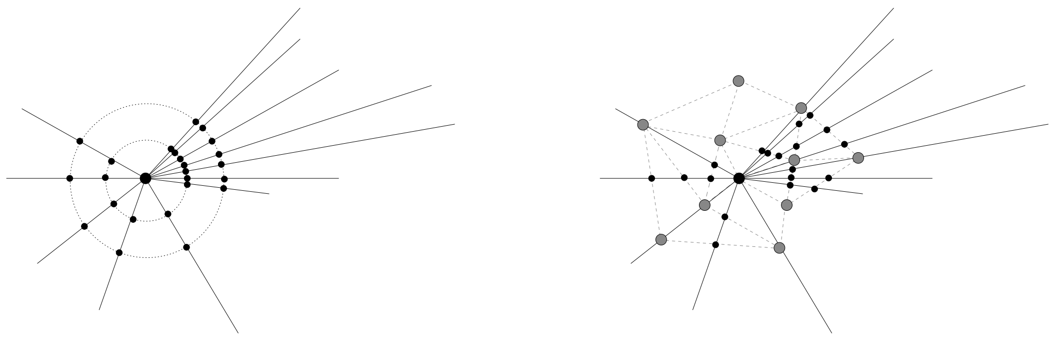

Figure 1: **Left:** Prior work involves cutting intersecting edges on concentric shells to "protect" the central region. We avoid this approach in order to obtain strong guarantees on the output mesh. **Right:** OSM uses an overlay mesh (shown dashed) to determine where input constraints will be subdivided, producing fewer output triangles in general.

be identical, effectively "protecting" the region inside the small angle. See Figure 1 Left. Extensions of the protective region approach to 3D have so far been quite involved, and output size guarantees fairly weak [CP03, PW04, CDRR04]. The immediate consequence of concentric shelling is that the size and grading of the output mesh depends on the smallest angle in the input PSLG.

Even in the absence of small input angles, well-graded meshing techniques often produce meshes whose size is linearly dependent on the spread ($L/s$) of the underlying vertex input set. Generally, the spread is assumed to be some polynomial in the size of the input. Many common meshing applications model domains with layers of thin sheets. In these cases, the increased spread can make well-graded methods entirely infeasible.

Because of the blowup in size associated with well-graded meshing, much research has focused on the problem of eliminating large angles. Bern, Dobkin, and Eppstein give an algorithm that produces a no-large-angle mesh of a simple polygon using $O(n \log n)$ triangles; or $O(n^{3/2})$ triangles for polygons with holes [BDE95]. This result was later improved by Bern, Mitchell, and Ruppert who gave an algorithm that produces a nonobtuse triangulation of a polygon with holes using $O(n)$ triangles [BMR95]. For arbitrary planar straight line graphs, there is a lower bound attributed to Paterson that says $\Omega(n^2)$ triangles may be necessary. The best known algorithm for producing a no-large-angle triangulation of an arbitrary PSLG is due to Mitchell and uses $O(n^2 \log n)$ triangles [Mit93].

In his paper, Mitchell poses two questions that are addressed in this paper: Is there an algorithm that produces a no-large-angle triangulation that is within a constant factor of worst case optimal? Secondly, is there an algorithm that, for any given input, gives a no-large-angle triangulation competitive in size with the optimal? We settle the first question in the affirmative for inputs with $L/s \in O(n)$. We answer the second question with an $O(\log(L/s))$ competitive factor.

## 1.3 Overlay Stitch Meshing

In his 1993 paper, Mitchell posed two questions that are addressed in this paper: Is there an algorithm that produces a no-large-angle riangulation that is within a constant factor of worst case optimal? Secondly, is there an algorithm that, for any given input, gives a no-large-angle triangulation competitive in size with the optimal? We settle the first question in the affirmative for inputs with $L/s \in O(n)$. We answer the second question with an $O(\log(L/s))$ competitive factor.

The main results of this paper may be viewed as an improvement in both the fields of well-graded meshing and no-large-angle meshing. The OSM algorithm provides a new Delaunay-refinement based method that is guaranteed to terminate on inputs with small angles *without* the output size depending heavily on the smallest feature size. Furthermore, as a no-large-angle meshing algorithm, this paper presents the first $\log(L/s)$-approximation to the optimal no-large-angle mesh for a given input PSLG. As it is reasonable to assume that

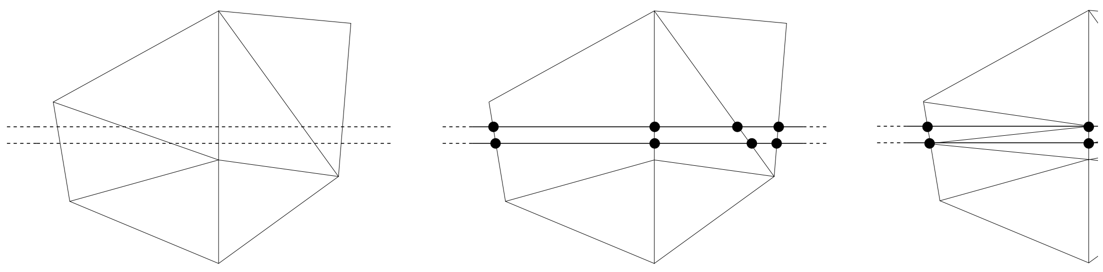

Figure 2: The basic stages of overlay meshing: At left, an overlay mesh is generated that does not yet conform to input segments (shown dashed). Center, the input segments are stitched into the mesh by adding some intersections and discarding some overlay edges. At right, the triangulation is completed.

$L/s \in O(poly(n))$, OSM may be viewed as a $\log(n)$-approximation.

## 2 The OSM Algorithm

The algorithm has three phases: the Overlay mesh phase, the Stitching phase, and the Completion phase. In the first phase, a standard point set meshing algorithm is run on the input vertices to form the *overlay mesh*. In the second phase, the input edges are *stitched* into the mesh, , carefully choosing to add vertices at some intersections of the overlay mesh and the input segments. The Stitching phase will leave some non-triangular faces. In the last phase, the Completion phase, these leftover faces are triangulated to minimize the largest angle. See Figure 2

### 2.1 Phase 1: The Overlay Mesh

The overlay mesh is constructed on the input vertices using the Voronoi Refinement meshing algorithm for point sets [HMP06]. The output is a mesh conforming to the input vertices of gap-ratio quality $\Gamma$, for a constant parameter $\Gamma > 1$. No angle in the overlay mesh will be smaller than $\theta_1 = \arcsin(1/2\Gamma)$.

As written, this phase of OSM is blind to the input segments. Heuristics that are segment-aware can be used here to tune some properties of the final output (see Section 5). In the absence of such heuristics, the usual method is to set $\Gamma = 1 + \varepsilon$ for some small constant $\varepsilon$, yielding a $\theta_1$ slightly less than 30°.

### 2.2 Phase 2: Stitching in Edges

We now wish to begin conforming to the input segments. We can look at all the intersections between overlay edges and input segments, and we will classify every intersection as good or bad. A good intersection is approximately perpendicular, meaning that the segment and overlay edge meet at angles larger than $\theta_1$. A bad intersection is approximately parallel (angles smaller than $\theta_1$).

If an overlay edge crosses no segment, or crosses any segment with a good intersection, it will be kept. Overlay edges that cross segments at solely bad intersections will be discarded. The intuition here is that if an edge of the overlay mesh intersects input segments only in a parallel fashion, then we can use the input segments instead, so we throw out the overlay edge.

We have kept some overlay edges that cross input segments. We will then add Steiner points that subdivide input segments where they intersect these crossing edges (at good or bad intersections). We also now add all the subdivided input segments to the mesh. The result is not, in general, a triangulation, although it does now conform to the input.

### 2.3 Phase 3: Completing the Mesh

After the edges are stitched in, all that remains is to add enough edges to get back to a triangulation. Lemma 1 shows that this last step can be done efficiently because all of the non-triangular faces have at most six sides. We will show that each face can be triangulated with no small angles. In this phase of the algorithm, each remaining non-triangular face is passed to a subroutine that returns a triangulation of that face minimizing the

**Stitching in edges**

```
1: for all input edges $e$ do
2:     Compute intersections of $e$ with overlay mesh edges
3:     if some overlay edge $e'$ intersects $e$ at an angle greater than $\theta_1$ then
4:         mark $e'$ as kept.
5:     end if
6: end for
7: /* Insert the 'good' intersections */
8: for all intersections do
9:     if the overlay edge at the intersection is marked as kept then
10:        insert the intersection point into the mesh splitting corresponding input edge.
11:    else
12:        Remove the overlay edge at the intersection from the mesh.
13:    end if
14: end for
15: /* Recover the input */
16: for all input subsegments $e$ do
17:     insert $e$ into the mesh
18: end for
```

maximum angle. Since, the faces are so small, it is possible to find this triangulation in constant time with a naive algorithm.

**Lemma 1.** *After the input edges are stitched in, all faces have at most 6 sides.*

*Proof.* The non-triangular faces have at most 3 faces from the overlay mesh and at most 3 faces from input segments that were stitched. Details omitted for brevity. □

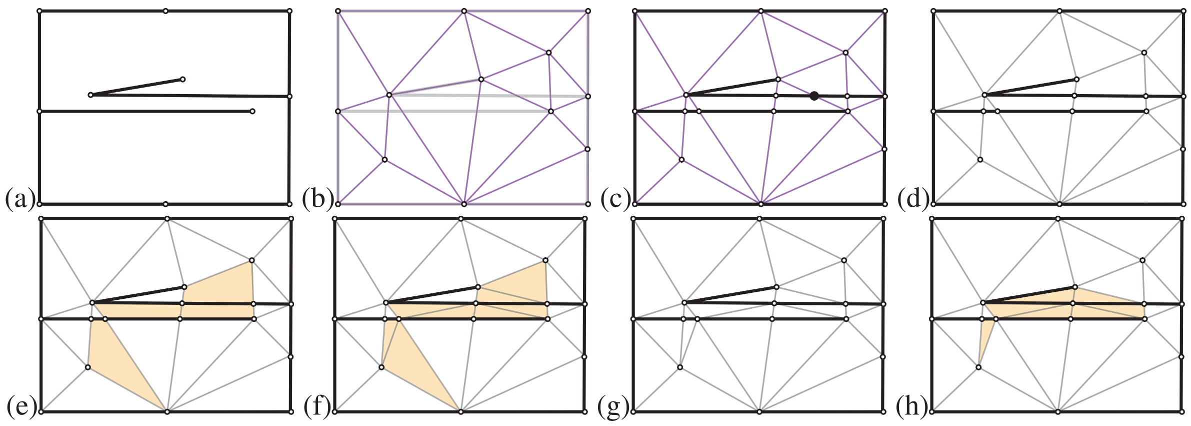

Figure 3: A sample run of OSM. In box (a), an input PSLG is given. The input contains both a small angle and a narrow pair of edges. In (b), the first phase of OSM constructs a quality overlay mesh on the input points, ignoring the input edges. In (c), the input is intersected with the overlay mesh and one bad intersection is identified. Box (d) shows the (not yet triangular) mesh after the stitching phase. The remaining non-triangular faces highlighted in (e) are triangulated in the completion phase (f). The final mesh is shown in (g) and the remaining small angle triangles are shown in (h).

## 3 Output Angle Guarantees

The purpose of this section is to prove that the maximum angle in the output mesh is bounded from above by a constant $\pi - \theta_2$ that is independent of the smallest input angle. Here, $\theta_2$ is a constant depending only on $\theta_1$, the smallest angle in the overlay mesh. This result is stated formally below as Theorem 2. If our overlay mesh has $\Gamma = 1 + \varepsilon$, then we find that $\theta_1 \geq 29.9°$, and $\theta_2 = \arcsin(1/(4 + 2\cos\theta_1)) \geq 10°$. First, we prove an important lemma regarding the angles at vertices inserted during the stitching phase.

**Lemma 2.** *If an edge $e$ intersects an input edge $f_1$ at an angle greater than $\theta_1$ then no input edge $f_2$ can intersect $e$ at an angle less than $\theta_2 = \arcsin\left(\frac{1}{2\Gamma(2\Gamma+\cos\theta_1)}\right)$.*

*Proof.* Let $F_1$ and $F_2$ be the lines containing edges $f_1$ and $f_2$ respectively, and let $v = F_1 \cap F_2$. The statement is trivial if $F_1, F_2$ are parallel or identical, so we may assume that $v$ is a single point.

Without loss of generality, let us assume that the edge $e$ is horizontal and $v$ lies below the line containing $e$. Also, by symmetry, we may assume that the small angle intersections we are worried about are on the left.

Figure 4 (Left) shows the edge $e$ with the circumcircle $C_1$ of a triangle containing it. The gap ratio guarantees that there is an empty ball $C_2$ centered at $a$ with radius at least $1/\Gamma$ times the radius of $C_1$. The point $d$ is the lower intersection of these two circles. In the figure $\theta_1 = \angle abd$. The line $L_2$ contains $\overline{bd}$ and the line $L_1$ is parallel to $L_2$ passing through $a$. The point $c$ is the lower intersection of $L_1$ and $C_2$.

Since $f_1$ passes through $e$ at an angle greater than $\theta_1$, $v$ must lie below $L_1$. In order for $f_2$ to intersect $e$ at an angle less than $\theta_1$, $v$ must lie above $L_2$. Thus we see that the intersection $v$ must lie somewhere in the shaded region. Therefore the smallest possible angle between $f_2$ and $e$ occurs if $v = c$ and the angle is

$$\angle abc \geq \arcsin\left(\frac{\sin\theta_1}{2\Gamma + \cos\theta_1}\right) = \arcsin\left(\frac{1}{2\Gamma(2\Gamma + \cos\theta_1)}\right)$$

as desired. □

The preceding Lemma ensures that if OSM chooses not to discard an edge of the overlay mesh, then that edge will not create any large angles. We can now proceed to the main theorem about the output angle guarantees.

**Theorem 2.** *All angles in the final mesh are bounded from above by $\pi - \theta_2$, where $\theta_2 = \arcsin\left(\frac{1}{2\Gamma(2\Gamma+\cos\theta_1)}\right)$ is the lower bound on the angle of intersection between a kept edge of the overlay mesh and an input edge. For $\Gamma = 1 + \varepsilon$, this gives a largest angle of $170°$.*

*Proof.* We need to consider the angles at two types of vertices, those vertices that appear in the overlay mesh and those that are added during the stitching phase. In the latter case, we were careful only to add vertices when the angles of intersection met this criterion. Thus, although we may have added more edges incident to such vertices in order to get a triangulation, the maximum angle at such vertices clearly achieves the desired bound.

In the case of a vertex $v$ from the overlay mesh, we have to be a little more careful. The overlay mesh itself satisfied this property so all large angles at overlay vertices must arise from discarded overlay edges. The completion phase of the algorithm returns the max-min angle triangulation of the non-triangular phases. It will suffice to show that there exists some triangulation that guarantees no large angles.

Observe that any input edge $e$ crossing a triangle $t$ of the overlay mesh can cause at most one edge of $t$ to be discarded. This is because $\theta_1 = \arcsin\left(\frac{1}{2\Gamma}\right)$ is a lower bound on the smallest angle in the overlay mesh, and thus the largest angle is at most $\pi - 2\theta_1$. It follows that at least one of these edges must intersect $e$ at an angle greater than $\theta_1$ and therefore will not be discarded. This fact implies that when an edge gets discarded, we can replace it with one that has been rotated by at most $\theta_1$. This is illustrated in Figure 4 (Right). The resulting largest angle at $v$ is at most $\pi - \theta_1$. Now, if we look at the edges ordered radially around $v$, we see that no two adjacent edges can be rotated apart from each other. This is because if two edge of the triangle $t$ are discarded then they both get replaced with edges that terminate on the third edge and thus lie entirely within the triangle. Thus the angle at $v$ is strictly smaller in this case. □

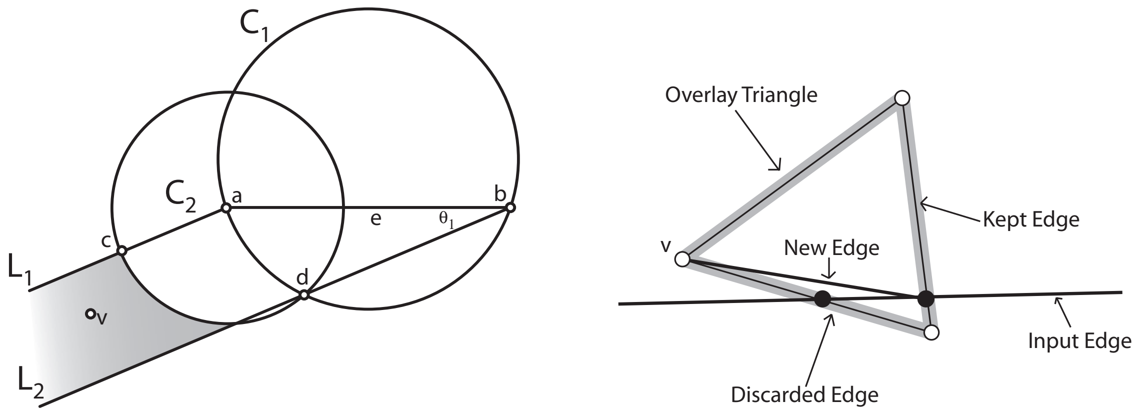

Figure 4: **Left.** Circumball $C_1$ and gap radius ball $C_2$ for an edge $e$ and its endpoint $a$ respectively. The lines $L_1$ and $L_2$ are parallel. The point $v$ must be below $L_1$ to keep $e$ from being discarded and above $L_2$ in order to allow $f_2$ to form a bad angle with $e$. **Right.** The input edge causes exactly one edge of the overlay triangle to be discarded. This edge is replaced with a new edge whose angle with the discarded edge is less than $\theta_1$. The angles at $v$ are changed by at most $\theta_1$.

## 4 Size of the Triangulation

In this section we show that the size of the output mesh is determined only by the local feature size of the input PSLG.

**Lemma 3.** *For any input edge $e$, the number of triangles in the overlay mesh intersecting $e$ is*

$$O\left(\int_{z\in e} \frac{1}{\mathrm{lfs}_0(z)}dz\right). \tag{4.1}$$

*Proof.* Let $t_1, \ldots, t_k$ be a minimal ordered sequence of adjacent triangles in the overlay mesh that covers $e$. Assign heights to the vertices of $t_2, \ldots, t_k$ so that the highest point of $t_i$ is one more than the highest point of $t_{i-1}$. Set the heights for $t_1$ to be $0, 0, 1$. Let $e_i = e \cap t_i$ be the subsegment of $e$ contained in triangle $t_i$. We consider the lifted version of $e$, call it $e^+$, to be the polygonal chain in $\mathbb{R}^3$ on the surface of these lifted triangles whose segments project down onto $e$.

Observe that the gradient of a lifted triangle $t$ cannot be too steep because both the smallest angle of $t$ and the maximum height difference between vertices of $t$ are bounded by constants. The maximum difference between the height of the vertices of a triange is bounded by the degree of vertices. Thus the gradient of $t$ is bounded by $\frac{\gamma}{r}$ where $r$ is the radius of the circumcircle of $t$ and $\gamma > 0$ is a constant.

Let $e_i$ be the subsegment of $e$ lying in triangle $t_i$. The change in height along $e_i^+$ is at most $|e_i|\frac{\gamma}{r_i}$ where $r_i$ is the circumradius of triangle $t_i$. Thus, the total change in height $k$ along $e^+$ is bounded as follows.

$$k \leq \sum_{i=1}^{k} |e_i| \frac{\gamma}{r_i} \tag{4.2}$$

$$\leq \gamma \sum_{i=1}^{k} \int_{e_i} \frac{1}{r_i} dx \tag{4.3}$$

The triangles $t_i$ are part of a well graded mesh and therefore, $\mathrm{lfs}(x) = cr_i$ for all $x \in t_i$ and some constant $c$.

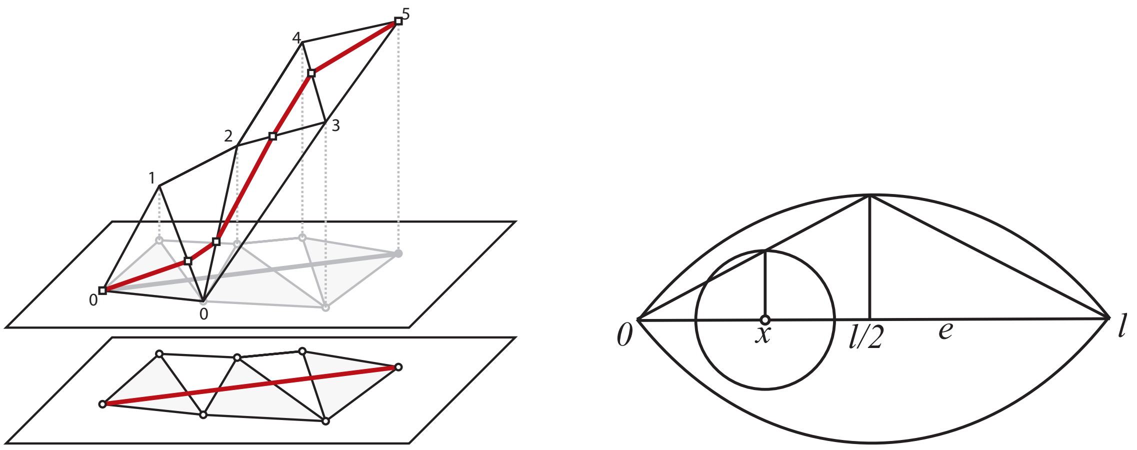

Figure 5: **Left.** An edge cuts through the overlay triangulation. Heights are assigned to the vertices so that the maximum height is exactly the number of triangles cut. **Right** An edge with an empty lens around it. The radius of the circle centered at $x$ is the height of the triangle at $x$. This guarantees that the circle is contained entirely within the lens.

So, we can rewrite the above inequality as follows to complete the proof.

$$k \leq \gamma \sum_{i=1}^{k} \int_{e_i} \frac{c}{\mathrm{lfs}_0(x)} dx \tag{4.4}$$

$$= c\gamma \int_{E} \frac{1}{\mathrm{lfs}_0(x)} dx \tag{4.5}$$

□

**Theorem 3.** *The number of Steiner points added during the course of the algorithm is*

$$O\left(\int_{\Omega} \frac{1}{(\mathrm{lfs}_0(z))^2} dz + \int_{E} \frac{1}{\mathrm{lfs}_0(z)} dz\right).$$

*where $E$ is the set of input edges and $\Omega$ is the input domain (*i.e.* the plane).*

*Proof.* We look at the two phases of OSM where vertices are added. First, in the construction of the overlay mesh, the number of points added is $O\left(\int_{\Omega} \frac{1}{(\mathrm{lfs}_0(z))^2} dz\right)$. This is a standard size guarantee for point set meshing [Rup95a, HMP06].

Second, in the stitching phase, we choose a subset of the intersections along each edge with the overlay mesh. It follows from Lemma 3 that the total number of intersections is $O\left(\int_{E} \frac{1}{\mathrm{lfs}_0(z)} dz\right)$. Therefore, it follows that the subset of these that we keep also achieves this bound. The statement of the theorem follows directly from summing the Steiner points added in each phase.

□

### 4.1 Competitive results

The $\alpha$-lens is the main tool we will use to analyze the optimal mesh for a given maximum angle guarantee. Recall that an $\alpha$-lens on a line segment $\overline{xy}$ is the set of all points $z$ such that $\angle xzy \geq \alpha$. A lens is the intersection of two circles with the same radius. We note one important fact about $\alpha$-lenses: If $\overline{ab}$ is a subsegment of $\overline{xy}$ then the $\alpha$-lens around $\overline{ab}$ is strictly contained in the $\alpha$-lens around $\overline{xy}$.

**Theorem 4.** *The output of OSM is a mesh with no large angles that is at most $O(\log(L/s))$ times the size of any mesh achieving the same maximum angle guarantee for some fixed constant c.*

*Proof.* Just as in Theorem 3, we will consider the Steiner points added during the Overlay phase separate from those added during the Stitching phase. The Overlay phase only adds $O(n \log(L/s))$ and any mesh mesh conforming to the input has size $\Omega(n)$ so we need only worry about Steiner points added during the Stitching phase.

Suppose we have an optimal size mesh $M_{OPT}$ with the property that no angle is greater than some constant $\alpha$. For any edge $e$ in $M_{OPT}$, the $\alpha$-lens around $e$ contains no other vertices of $M_{OPT}$. This is just another way of stating the no large angle property. In particular, the edges $e$ that are subsegments of input edges have $\alpha$-lenses that contain no input vertices.

In order to prove that OSM is $\log(L/s)$-competitive, it will suffice to prove that OSM stitches at most $\log(L/s)$ Steiner points on any input subsegment of $M_{OPT}$.

Recall that $\mathrm{lfs}_0(x)$ is the radius of the smallest circle centered at $x$ containing two input vertices. In general, $\mathrm{lfs}_0$ is lower bounded by $s$, the distance between the two closest vertices in the mesh. We get a better lower bound on $\mathrm{lfs}_0$ near input edges from the fact that the lenses around the input subsegments of $M_{OPT}$ contain no input vertices. We will use both of these lower bounds on $\mathrm{lfs}_0$ to upper bound the integral from Theorem 3.

For a particular input subsegment $e$ in $M_{OPT}$, we parameterize $e$ on the interval $[0, l]$ where $l$ is the length of $e$ as in Figure 5.

To compute the lower bound on $\mathrm{lfs}_0(x)$ for $x \in [0, l/2]$ it suffices to show that there is an circle centered at $x$ that contains no input vertices. We first inscribe an isosceles triangle into the top half of the $\alpha$-lens around $e$. The altitude of the triangle at $x$ is $\tan^{-1}(\alpha/2)x$.

Consider the circle $C$ centered at $x$ with radius $\tan^{-1}(\alpha/2)x$. Observe that the top edge of the inscribed triangle cuts off some half $\alpha'$-lens from the circle $C$ and some half $\alpha''$-lens from the original lens around $e$. Observe that $\alpha' = \alpha''$ and thus the smaller lens is entirely contained in the larger. It follows that the circle $C$ is contained entirely within the $\alpha$-lens around $e$ and thus, $C$ contains no input vertices. Therefore, $\mathrm{lfs}_0(x) \geq \tan^{-1}(\alpha/2)x$ for all $x \in e$.

We can now use our bound on $\mathrm{lfs}_0$ to bound the size integral from Theorem 3 as follows.

$$\int_0^l \frac{1}{\mathrm{lfs}_0(x)} dx \leq 2 \left( \int_0^s \frac{1}{s} dx + \int_s^{\frac{l}{2}} \frac{1}{\mathrm{lfs}_0(z)} dx \right) \tag{4.6}$$

$$\leq 2 + 2 \tan^{-1} \alpha/2 \int_s^{\frac{l}{2}} \frac{1}{x} dx \tag{4.7}$$

$$\in O(\log \frac{l}{s}) \tag{4.8}$$

$$\in O(\log \frac{L}{s}) \tag{4.9}$$

□

## 5 Snapping Heuristic

In this section, we discuss a heuristic that can be added to the Overlay phase of OSM. When generating the overlay mesh using Voronoi Refinement, Steiner points are incrementally inserted to gradually improve the gap-ratio quality of the mesh.

One obvious way to make this process segment-aware is to *snap* these Steiner points onto input segments in cases where they are relatively close. This shows some similarity to standard well-graded meshing with input segments [Rup95a]. It differs however in that we to do not force the overlay mesh to conform to these segments yet, so if a Steiner point is close to multiple input segments (such as those meeting at a small angle), the snapping heuristic would only be subdividing one of the input segments.

In principle, it is not necessary to do any snapping in order to obtain any of the theoretical guarantees regarding output angles and mesh size that we have proved in this paper. However, in practice, it is beneficial to do some snapping to avoid placing Steiner points arbitrarily close to input edges.

Another useful heuristic is *pinching*, the contraction of some artificially short edges of the final output mesh as a post-process. If snapping and pinching are combined, it is possible to achieve lower bounds on mesh angles that are outside some narrow regions dictated by the input PSLG. This involved result is not strictly relevent to the no-large-angle problem, but it does give intuition on the aesthetic structure of the output.

## 6 Conclusions

### 6.1 Size bounds in terms of n

To better understand the size guarantees of OSM in relation to previous results that only analyze worst case performance, we can compute two coarse upper bounds on the mesh size.

First, we see that the mesh size is $O(n^2 \log(L/s))$. This follows from the fact that the overlay mesh is of size $O(n \log(L/s))$. There are at most $O(n)$ edges so the stitching phase adds at most $O(n^2 \log(L/s))$ points. When $L/s \in O(poly(n))$, this bound exactly matches the $O(n^2 \log n)$ of Mitchell on triangulating with no large angles from [Mit93].

Alternatively, one could recompute the sizing integral from Theorem 3 using $s$ as a lower bound on $lfs_0$. The result is an $O(n(L/s))$ upper bound on the mesh size. So, when $L/s \in O(n)$, the output size is $O(n^2)$. Certain pathological examples such as Paterson's example (see [BDE95]) requiring $\Omega(n^2)$ Steiner points can be drawn so that $L/s \in O(n)$. Thus, the OSM algorithm is worst case optimal when the input vertices have linear spread.

For reasonable inputs where $L/s \in O(poly(n))$, Theorem 4 implies that the output of OSM is $O(\log n)$-competitive. For inputs that admit linear size no-large-angle meshes, this is a factor of $n$ better than the previous guarantee.

### 6.2 Work Efficiency

Overlay Stitch Meshing can be easily implemented to run in time and space $O(n \log(L/s)+m)$. The majority of the work is spent in generating the overlay mesh. Simple arguments can show that the stiching and completion phases can be implemented as $O(m)$ post-processes. In the case of input with $O(poly(n))$ spread, this is asymptotically optimal work.

### 6.3 Extensions

The first obvious extension is into three and higher dimensions. Obtaining good guarantees on output size for meshing algorithms in three dimensions remains an interesting open problem [CP03, PW04, CDRR04]. When the input (now containing facets as well) has no small angles, obvious extensions of traditional algorithms work well. However, for the general input case, concentric shelling techniques for unstructured meshing have proven complicated and yielded no strong sizing results.

## References

[BA76] Ivo Babuška and A. K. Aziz. On the Angle Condition in the Finite Element Method. *SIAM Journal on Numerical Analysis*, 13(2):214–226, April 1976. 1

[BDE95] Marshall W. Bern, David P. Dobkin, and David Eppstein. Triangulating polygons without large angles. *Int. J. Comput. Geometry Appl*, 5:171–192, 1995. 3, 10

[BEG94] Marshall Bern, David Eppstein, and John Gilbert. Provably good mesh generation. *J. Comput. Syst. Sci.*, 48(3):384–409, 1994. 1, 2

[BHV04] Erik Boman, Bruce Hendrickson, and Stephen Vavasis. Solving elliptic finite element systems in near-linear time with support preconditioners, 2004. 1

[BMR95] Marshall W. Bern, Scott A. Mitchell, and Jim Ruppert. Linear-size nonobtuse triangulation of polygons. *Discrete & Computational Geometry*, 14(4):411–428, 1995. 3

[CDRR04] Siu-Wing Cheng, Tamal K. Dey, Edgar A. Ramos, and Tathagata Ray. Quality Meshing for Polyhedra with Small Angles. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry*, pages 290–299, Brooklyn, New York, June 2004. Association for Computing Machinery. 3, 10

[CP03] Siu-Wing Cheng and Sheung-Hung Poon. Graded Conforming Delaunay Tetrahedralization with Bounded Radius-Edge Ratio. In *Proceedings of the Fourteenth Annual Symposium on Discrete Algorithms*, pages 295–304, Baltimore, Maryland, January 2003. Society for Industrial and Applied Mathematics. 3, 10

[GMW99] Stephen Guattery, Gary L. Miller, and Noel Walkington. Estimating interpolation error: A combinatorial approach. In *Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 406–413, Baltimore, January 1999. ACM and SIAM. 1

[HMP06] Benoît Hudson, Gary Miller, and Todd Phillips. Sparse Voronoi Refinement. Technical Report CMU-CS-06-132, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, June 2006. 4, 8

[Mit93] Scott A. Mitchell. Refining a triangulation of a planar straight-line graph to eliminate large angles. In *34th Annual Symposium on Foundations of Computer Science*, pages 583–591, Palo Alto, California, 3–5 November 1993. IEEE. 3, 10

[MV92] S.A. Mitchell and S. Vavasis. Quality mesh generation in three dimensions. In *Proc. 8th ACM Symp. Comp. Geom.*, pages 212–221, 1992. 1

[MV05] Gary Miller and Steve Vavasis. Only large angles mater. Private communications, 2005. 1

[Pav03] Steven Elliot Pav. *Delaunay Refinement Algorithms*. PhD thesis, Department of Mathematics, Carnegie Mellon University, Pittsburgh, Pennsylvania, May 2003. 2

[PW04] Steven E. Pav and Noel J. Walkington. Robust Three Dimensional Delaunay Refinement. In *Thirteenth International Meshing Roundtable*, pages 145–156, Williamsburg, Virginia, September 2004. Sandia National Laboratories. 3, 10

[Rup95a] Jim Ruppert. A Delaunay refinement algorithm for quality 2-dimensional mesh generation. *J. Algorithms*, 18(3):548–585, 1995. 1, 2, 8, 9

[Rup95b] Jim Ruppert. A Delaunay refinement algorithm for quality 2-dimensional mesh generation. *J. Algorithms*, 18(3):548–585, 1995. Fourth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA) (Austin, TX, 1993). 2

[She02] Jonathan Richard Shewchuk. Delaunay Refinement Algorithms for Triangular Mesh Generation. *Computational Geometry: Theory and Applications*, 22(1–3):21–74, May 2002. 1, 2