Biting: Advancing Front Meets Sphere Packing *

Xiang-Yang Li Shang-Hua Teng Alper Üngör

Abstract. *A key step in the finite element method is to generate a high quality mesh that is as small as possible for an input domain. Several meshing methods and heuristics have been developed and implemented. Methods based on advancing front, Delaunay triangulations, and quadtrees/octrees are among the most popular ones. Advancing front uses simple data structures and is efficient. Unfortunately, in general, it does not provide any guarantee on the size and quality of the mesh it produces. On the other hand, the circle-packing based Delaunay methods generate a well-shaped mesh whose size is within a constant factor of the optimal. In this paper, we present a new meshing algorithm, the* biting method, *which combines the strengths of advancing front and circle packing. We prove that it generates a high quality 2D mesh, and the size of the mesh is within a constant factor of the optimal.*

Keywords. unstructured mesh generation, advancing front, paving, circle packing, biting.

1 Introduction

An essential step in numerical simulation of physical and engineering problems is to find a proper discretization of a continuous domain. This is the problem of *mesh generation* [4, 22, 11, 12, 24, 25]. For problems with complex geometry boundaries and with solutions that change rapidly, we need to use an *unstructured mesh* with a varying local topology and spacing in order to reduce the problem size. A good unstructured meshing algorithm uses elements of properly chosen size and shape that adapt to the complex geometry and solution accuracy. In doing so, it generates meshes that are numerically sound and that are also as small as possible. Several meshing methods and heuristics have been developed, implemented, and applied to various applications such as steady state and transient compressible inviscid flow simulations.

Over the years, several meshing methods such as those based on advancing front, Delaunay triangulations, and quadtrees/octrees have become popular due to their effectiveness in practical applications. However, these methods do not come with equal strengths. For example, advancing front [6, 14, 15] uses simple data structures and is efficient and relatively easy to implement. It offers a high quality of point placement strategy and the integrity of the boundary. Unfortunately, it does not provide any general guarantee on the size and quality of the mesh it produces. On the other hand, more sophisticate methods such as quadtree/octree refinement [4, 22, 32] and Delaunay methods [7, 8, 9, 19, 24, 25] generate a well-shaped mesh whose size is within a constant factor of the optimal. Our objective is to develop a new 2D meshing algorithm that combines the strengths of advancing front and these provably good meshing methods.

The particular type of Delaunay method that we will use in conjunction with advancing front is the circle packing method. It first constructs a *well-spaced* point set by computing a circle packing of the domain and then uses the Delaunay triangulation of this point set as the final mesh. Two methods have been developed to generate the well-spaced point set. The first one applies *particle simulation* [26, 27, 28] to find a stable

*Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801. Supported in part by an NSF CAREER award (CCR-9502540), an Alfred P. Sloan Research Fellowship, and the U.S. Department of Energy through the University of California under Subcontract number B341494 (DOE ASCI). The emails of authors: xli2@cs.uiuc.edu, steng@cs.uiuc.edu, ungor@uiuc.edu.

configuration of a set of energetic circles. The second one uses quadtree/octree refinement to obtain an oversample of the input domain, and then applies a properly defined maximal independent set to create the circle packing [3, 5, 12, 21, 30]. Both in theory and in practice, the second approach is faster.

In this paper, we show that the advancing front method can be used to efficiently construct a quality circle-packing. At a high level, this new advancing front based packing algorithm first finds a circle packing of the boundary of the domain and then grows the packing towards the interior of the domain. Each time when a new circle is added to the interior, a larger protection circle is removed (bitten away) from the domain so that no future circle will overlap with this one. By doing this, it builds the circle packing by adding circles one at a time, or a layer at a time, in the same spirit as the standard advancing method; our new method uses advancing front to construct a circle packing instead of the mesh elements themselves. We show that this advancing front based method does generate a well-spaced point set, whose Delaunay triangulation is well-shaped. We will refer this new method as the *biting method* and show that it can be made as practical as the standard advancing front meshing methods.

The paper is organized as follows. Section 2 discusses the control space which specifies the element sizes and point densities of a mesh. Section 3 reviews the standard advancing front methods. Section 4 covers the circle packing method and introduces definitions that will be used in this paper and related theorems concerning the connection between the quality of the Delaunay triangulation and the quality of the circle packing. Section 5 presents the the biting method and proofs about the quality of the circle packing that it generates. Section 6 gives the details of boundary protection to complete the proof of our main theorem.

2 Control Space

Each domain Ω and a differential equation u defines a desired local spacing within a domain to specify, for example, the expected element size in a given neighborhood or point densities near a point. In this section, we discuss how to determine the local spacing from the geometry of Ω and the numerical condition of u.

2.1 Geometric Features

The geometry of the boundary of Ω also contributes to the local spacing of a well-shaped mesh. In two dimensions, we assume that Ω is given as a *planar-straight-line graph* (PSLG), which is a collection of line segments and points in the plane, closed under intersection. Suppose Ω is described by a PSLG S. Ruppert [24] introduced the following concept called *local feature size*.

Definition 2.1 *Given a PSLG S, the* local feature size *at a point $\boldsymbol{x}$, $lfs_S(\boldsymbol{x})$, or simply $lfs(\boldsymbol{x})$, is the radius of the smallest disk centered at $\boldsymbol{x}$ that intersects two non-incident vertices or segments of S.* [1]

Note that adding new Steiner vertices does not change the value of $lfs()$ function, since it is determined by the input. Ruppert has observed that $lfs()$ changes slowly within the domain. Formally, a function $f()$ is *Lipschitz with a coefficiency* α if for any two points $\boldsymbol{x}$, $\boldsymbol{y}$ in the domain, $|f(\boldsymbol{x}) - f(\boldsymbol{y})| \leq \alpha||\boldsymbol{x} - \boldsymbol{y}||$. Then the Lipschitz coefficiency of $lfs()$ is bounded from above by 1 [24]. In addition, $lfs()$ is the maximum in the following sense.

Lemma 2.1 *If f is a 1-Lipschitz function over a domain Ω such that for each point $\boldsymbol{x}$ on $\partial\Omega$ $f(\boldsymbol{x}) \leq lfs_\Omega(\boldsymbol{x})$, then for every $\boldsymbol{x} \in \Omega$, $f(\boldsymbol{x}) \leq lfs_\Omega(\boldsymbol{x})$.*

A common shape criterion for the mesh elements is the condition that the angles of each element are not too small, or the aspect ratio of each element is bounded [1, 4, 29]. In this paper, we measure the quality of a triangular element by the *radius-edge ratio* as defined in [19, 20]. Based on this quality measure, we can define well-shaped meshes as the following.

[1] Ruppert also gave a modified definition by using the geodesic distance to the 2 nearest non-incident portions of the input to handle the *two arms* situation [24]. The geodesic distance is measured along the shortest path that stays within the domain to be triangulated.

Definition 2.2 (**α-well-shaped mesh**) *A mesh M is* α-well-shaped *for a constant $\alpha > 1$ if the minimum radius-edge ratio over all of its elements is bounded from above by α.*

There are several ways to describe the spacing function of a well-shaped mesh M over a domain Ω:

- **[Edge-length function,** el_M**]** for each point $x \in \Omega$, $el_M(x)$ is equal to the length of the longest edges of all mesh simplex elements that contain x (note that points on the lower dimensional faces of a simplex are contained in more than one element).
- **[Nearest-neighbor function,** nn_M**]** Let x be a point in Ω, there are two cases. (1) if x is a mesh point, then $nn_M(x)$ is equal to the distance of x to the nearest mesh point in M. (2) if x is not a mesh point, then $nn_M(x)$ is equal to the distance to the second closest mesh point in M.

Lemma 2.2 (**[19]**) *If M is an α-well-shaped, then there exists constants c_1 and c_2 depending only on α such that for all point $x \in \Omega$,*

$$c_1 el_M(x) \leq nn_M(x) \leq c_2 el_M(x).$$

2.2 Numerical Spacing

The numerical condition is usually obtained from an *a priori* error analysis, or an *a posteriori* error analysis based on an initial numerical simulation. It defines a *numerical spacing functions*, denoted by $nsf(\boldsymbol{x})$, for each point $\boldsymbol{x}$ in the domain Ω. The value of $nsf(\boldsymbol{x})$, from the interpolation viewpoint, is determined by the eigenvalues of the Hessian matrix of u [29]. Locally at point $\boldsymbol{x}$, u can be approximated by a quadratic function

$$u(\boldsymbol{x} + d\boldsymbol{x}) = \frac{1}{2}(d\boldsymbol{x}^T H d\boldsymbol{x}) + d\boldsymbol{x} \bigtriangledown u(\boldsymbol{x}) + u(\boldsymbol{x}),$$

where H is the *Hessian matrix* of u, the matrix of the second partial derivatives. The spacing of the mesh points, required by the accuracy of the discretization near $\boldsymbol{x}$ should depend on the reciprocal of the square root of the largest eigenvalues of H at $\boldsymbol{x}$.

For example, in adaptive numerical simulation, we estimate the eigenvalue of the Hessian matrix at a certain set of points in Ω based on the solution of the previous iteration, and then expand the spacing requirement from these points to the entire domain. From the new spacing and the old spacing function deduced from the previous mesh, we can get the *refinement* or *coarsening* factor for mesh points. We can then use the simultaneous refinement and coarsening method of Li *et al.* [12] to generate the mesh that satisfies the new control space requirement.

2.3 Control Spacing Function

The local feature size $lfs()$ and the numerical condition nsf together defines the global control spacing function. Notice however, that the Lipschitz coefficiency of nsf may not be bounded by a constant. Using the technique of Miller, Talmor, and Teng [16], we can define a new numerical spacing function $\underline{nsf}()$ as the following: for each point $\boldsymbol{x}$,

$$\underline{nsf}(\boldsymbol{x}) = min(nsf(\boldsymbol{x}), min_{\boldsymbol{y} \in \Omega}(nsf(\boldsymbol{y}) + ||\boldsymbol{x} - \boldsymbol{y}||)).$$

The Lipschitz coefficiency of $\underline{nsf}()$ is at most 1. In addition, $\underline{nsf}()$ is the best possible in the sense that for any 1-Lipschitz function g over the domain Ω, if $g(\boldsymbol{x}) \leq \overline{nsf(\boldsymbol{x})}$ point-wise in Ω, then $g(\boldsymbol{x}) \leq \underline{nsf}(\boldsymbol{x})$ point-wise.

The global control spacing function $gns()$ can then be defined as

$$gns(\boldsymbol{x}) = min(lfs(\boldsymbol{x}), \underline{nsf}(\boldsymbol{x})).$$

Where gns stands for Geometric and Numerical Spacing[31]. The function $gns()$ captures both the numerical and the geometric requirements for a well-shaped adaptive mesh.

Lemma 2.3 *If $f()$ and $g()$ are α_1 and α_2-Lipschitz respectively over Ω, then $f()+g()$ is $\alpha_1+\alpha_2$-Lipschitz, and* $\min(f(), g())$ *and* $\max(f(), g())$ *are $max(\alpha_1, \alpha_2)$-Lipschitz.*

Therefore, gns is 1-Lipschitz.

For mesh generation, we do not need to compute these spacing functions exactly. A common approach to approximate $gns()$ is to store discrete values on the vertices of a background mesh such as a quadtree/octree decomposition of the domain. When we need to evaluate the function at an arbitrary point in the domain, we simply interpolate these discrete values.

3 Advancing Front Methods

Advancing front methods construct a mesh of a domain by moving a front from its boundary towards its interior. It first generates an initial front typically by constructing a surface mesh for the boundary of the domain. It then creates new elements one at a time or a layer at a time and updates the front with these created faces [23, 10, 6, 14, 15]: In the one element-at-a-time model, it chooses a face of the current front and introduces a new mesh element with it as the base face. It can use another vertex on the front or insert a new Steiner point in the interior as the additional vertex of the new element. The base face and potentially some other faces on the front (if the additional vertex is an existing one) are removed from the front, and some faces of the new element are added to the front. This process is repeated until the front is empty, i.e., all fronts have merged upon each other and the domain is fully meshed.

Note that initial front does not have to be a single component. For example, for a domain with holes, the initial front can be built for the boundary of each hole as well.

The selection of the base face and the placement of the new mesh vertex are the two key steps of any advancing front method. These two steps must ensure that the new mesh element is valid and well-shaped and keep the front in good condition to allow the creation of graceful future elements. The faces of the clefts and the small faces are given priority to be picked as the base faces to satisfy these requirements.

Hence, once the base face is chosen, we need to decide where to place the new vertex. Recall that in a well-shaped triangular mesh, points must be well-spaced [30, 17], which implies that for each base face, we can only place the Steiner point in a particular region near the base face so that the new element is well-shaped. Call this region the *feasible region*. Some points in the feasible region will make the new element slightly larger (by a constant factor) than some other points do. This is where the control spacing function can be used. It helps us to decide whether we should go for a larger new element or a smaller one.

Paving [6] is one of the popular advancing front methods. It uses a number of operations to tightly controll the moving front to ensure the mesh validity and quality. These operations include *row choice*, *closure check*, *row generation*, *smooth*, *seam*, *row adjustment*, *intersection*, and *cleanup* [6]. The size of the elements in the mesh is determined by the spacing of the nodes on the paving boundary as it propagates. The spacing on the paving boundary is initially defined by the fixed node spacing on the corresponding exterior boundary.

Advancing front methods can be combined with Delaunay or quadtree/octree refinements. For instance, these refinement techniques can be used to generate a pretty-good domain decomposition of the input domain and then advancing front can be applied to get a mesh for each subdomain. We can also use quadtree/octree refinement to generate the set of points for the creation of the new elements.

In Section 5, we show how to use advancing front methods to help Delaunay based mesh generation. In particular, we present a method to construct a high quality circle-packing using the advancing front methods. We will prove that a well-shaped mesh can then be generated by the Delaunay triangulation of the centers of the packing circles.

4 Circle Packing Methods

At a high level, the circle-packing method fills an input domain with a set of circles whose centers provide a good vertex set for a quality Delaunay mesh. It can be used to generate meshes for various quality conditions.

For example, Bern, Mitchell, and Ruppert [5] use circle packing to triangulate a n-vertex polygonal region (potentially with holes) so that no element has angle larger than $\pi/2$. They show that one can do so with $O(n)$ triangles, improving a previous result that uses $O(n^2)$ triangles [2].

The algorithm first packs a set of circles within the domain such that the gaps between them are surrounded by at most four tangent circles. It then defines the mesh points as the union of centers of these circles, the tangency points, and one point within each gap, and locally triangulates these points. Notice that for nonobtuse triangulation, one does not need to consider the control spacing function. Therefore, their mesh may have elements with very bad aspect ratio. A similar circle-packing based method has been developed by Bern and Eppstein [3] for quadrilateral meshes.

Shimada and Gossard [27] have developed a circle-packing method called *bubble mesh* to generate triangular meshes for two and three dimensions. Their packing scheme is based on the simulation of the particles that interact each other with repulsive/attractive forces. They first define a proximity based force among the circles, and then find a stable configuration by moving or deleting circles. However, their method does not provide a theoretical bound on the time of the algorithm nor the quality of the mesh that they generate.

Miller *et al.* [19, 20] have designed a circle-packing based meshing method which combines two well-known methods, quadtree and Delaunay refinements. First, they apply a balanced quadtree refinement to approximate the spacing function $f()$. Second, they oversample a set of points in the domain to define a set of overlapping circles. Then, they compute a maximal set of non-overlapping circles from this set to obtain a circle packing. Finally, they compute the Delaunay triangulation of the centers of these circles.

Suppose $f()$ is the desired edge-length or nearest-neighbor function of a well-shaped mesh for a domain Ω. We now introduce some definitions to capture the quality of circle packing. Let $B(\boldsymbol{x}, r)$ denote the circle centered at point $\boldsymbol{x}$ with radius r.

Definition 4.1 (β-Packing) *Let β be a positive real constant. A set S of circles is a β-packing with centers P of Ω with respect to a spacing function $f()$ if*

- *For each point p of P, $B(p, f(p)/2) \in S$;*
- *The interiors of any two circles s_1 and s_2 in S do not overlap; and*
- *For each point $q \in \Omega$, there is a circle in S that overlaps with $B(q, \beta f(q)/2)$.*

The following structure theorem of Miller, Talmor, and Teng [18] states an equivalence relationship between β-circle packing and well-shaped meshes.

Theorem 4.1 (Circle Packing and Well-Shaped Meshes)

1. *For any positive constant β, there exists a constant α depending only on β such that if $f()$ is a spacing function of Lipschitz constant 1 over a domain Ω and S is a β-packing with respect to $f()$, then the Delaunay triangulation M of the centers of S is an α well-shaped mesh; in addition, for each point $\boldsymbol{p}$ in Ω, $nn_M(\boldsymbol{p}) = \Theta(f(\boldsymbol{p}))$, where the constant in Θ depends only on β.*

2. *For any positive constant α, there exists a constant β depending only on α such that if M is an α well-shaped mesh, then the set of circles*

$$S = \{B(\boldsymbol{p}, nn_M(\boldsymbol{p})/2) : \text{for all mesh point } \boldsymbol{p} \in M\},$$

is a β-packing with respect to $nn_M/2$.

5 Biting: Advancing Front Meets Circle Packing

Advancing front is known to be a practical mesh generation method. Unfortunately, in general, it does not provide any guarantee on the size and quality of the mesh it produces. On the other hand, the circle-packing based methods can generate a well-shaped mesh whose size is within a constant factor of the optimal. The cost of packing the circles using particle simulation or MIS of oversampled circles might be large. In this section, we present a new scheme, called the *biting method*, that combines the strengths of advancing front and circle packing. It uses advancing front to generate a quality circle packing rather than the mesh itself. The Delaunay triangulation of the centers of the circles is then used to define the final mesh. Using the equivalence between the well-shaped meshes and circle packings, we show that the biting method construct a well-shaped Delaunay mesh whose size is optimal up to a constant factor.

5.1 Biting Scheme

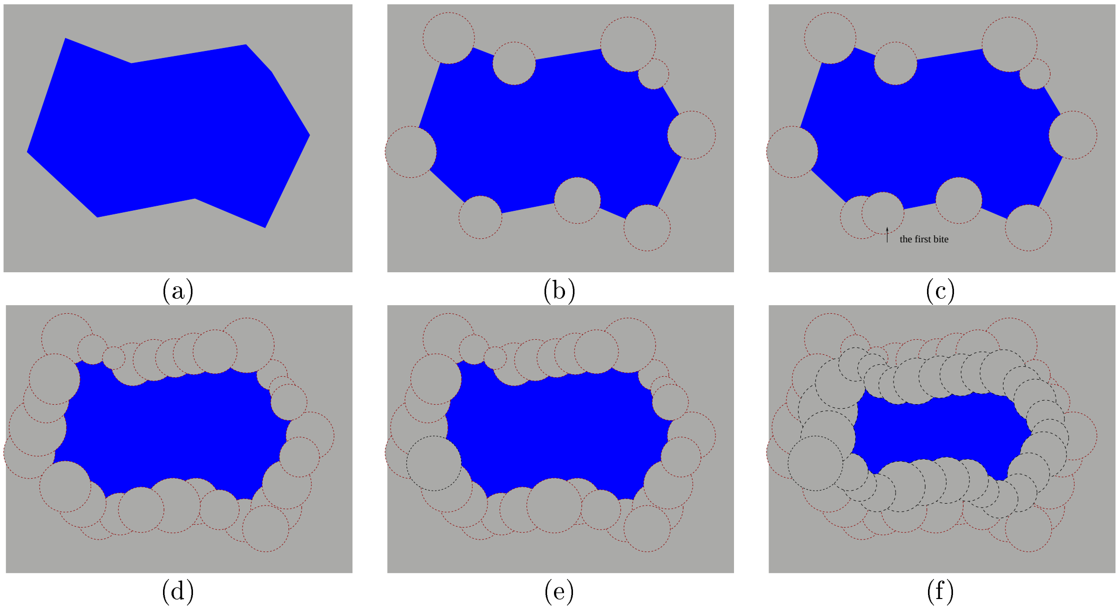

Figure 1: A snapshot of the *biting* scheme: (a) initial PSLG domain; (b) bites only on the vertices of the polygon; (c) first bite of a non-original vertex; (d) biting a layer of the boundary; (e) the first biting in the interior of the domain; (f) biting a layer in the interior of the domain.

The basic idea of the biting method is to first compute the control spacing function f of the mesh. We then try to find a point set by constructing a circle packing with respect to the spacing function. The circle packing is constructed using the advancing front method. The input domain boundary is set as the initial advancing front. The biting method moves a front from the boundary of the domain to the interior and adds new mesh points in the process. These mesh points are chosen such that we can define a circle packing with them as the centers. At each step, we place a new point on the current front rather than place it in the interior of the remain domain. Each time we add a point, we remove a circle, the *biting circle*, from the remaining interior domain. The boundary between the union of biting circles and the remaining interior domain defines the new front. This process is called *biting*.

A biting circle at a point $\boldsymbol{x}$ is $B(\boldsymbol{x}, c_b h(\boldsymbol{x}))$, where c_b is a constant which will be decided later. Here the subscribe b of constant c_b denotes *b*iting. Note that the biting circles are different from the packing circles due to some technical reasons that will be discussed in section 5.2. For making sure that the biting process results in a circle packing, we use the following simple idea: at every step, we choose a center on the advancing front and remove its biting circle. The removal of its biting circle ensures that the future packing circles will not intersect with the packing circle of this center. The Delaunay triangulation is then used to generate the mesh from the resulting circle packing.

If the input domain has a small angle, we can cut off that small angle first to approximate the domain. From now on, we assume that the input domain does not have an acute input angle. The following is a formal description of the biting method:

Algorithm `Biting`

1. Compute the control spacing function $h()$ of Ω by combining the local feature size and the numerical condition;
2. Let the boundary of the domain be the initial front, see Figure 1 (a);
3. [**Vertex Protection**]: Bite all the input vertices by removing their biting circles from the interior of the domain, see Figure 1 (b);

   Modify the front which becomes a set of segments and arcs. Segments are represented by the endpoints and arcs are represented by the center of the biting circle.
4. [**Edge Protection**]: Bite circles centered on the input boundary: choose a vertex $\boldsymbol{x}$ on the front and remove its biting circle. Whenever possible, we choose $\boldsymbol{x}$ on the intersection of some bitten circles with the initial boundary, see Figure 1 (c) and (d).

   Assume that $\boldsymbol{x}$ is on the bitten circle of $\boldsymbol{p}_1$. If the biting circle $B(\boldsymbol{x}, c_b h(\boldsymbol{x}))$ intersects with a bitten circle $B(\boldsymbol{p}_2, c_b h(\boldsymbol{p}_2))$, for a point $\boldsymbol{p}_2 \neq \boldsymbol{p}_1$ on the same boundary, then let $\boldsymbol{q}_1, \boldsymbol{q}_2$ be the closest intersection points of $B(\boldsymbol{p}_1, c_b h(\boldsymbol{p}_1))$ with the boundary and $B(\boldsymbol{p}_2, c_b h(\boldsymbol{p}_2))$ with the boundary, respectively. Note that $\boldsymbol{q}_1 = \boldsymbol{x}$. Let $\boldsymbol{q}$ be the middle point of segment $\boldsymbol{q}_1\boldsymbol{q}_2$.

   Check whether $B(\boldsymbol{q}_2, c_b h(\boldsymbol{q}_2))$ intersects with $B(\boldsymbol{p}_1, c_b h(\boldsymbol{p}_1))$. If it does not, we remove $B(\boldsymbol{q}_2, c_b h(\boldsymbol{q}_2))$ from the interior, otherwise we remove $B(\boldsymbol{q}, c_b h(\boldsymbol{q}))$. see Figure 2.

   Modify the front by introducing the arc of the new biting circles and removing the intersection of it with the front.

   Repeat until all initial input boundaries are bitten;
5. [**Interior Biting**]: Choose a vertex $\boldsymbol{x}$ on the front and remove its biting circle, see Figure 1 (e) and (f);

   Modify the front by introducing the arc of the new biting circle and removing the intersection of it with the front.

   Repeat until the advancing front is empty, i.e., the input domain is all covered by the biting circles.
6. Construct the Delaunay triangulation of the centers of the biting circles as the final mesh.

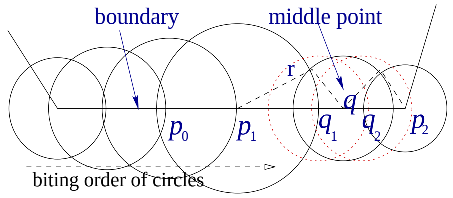

Figure 2: Edge Protection: the center of the biting circle is the intersection of the previous biting circle with the boundary, as the point $\boldsymbol{p}_1$ of the example: it is the intersection of circle $B(\boldsymbol{p}_0, c_b f(\boldsymbol{p}_0))$ with the boundary; or it is the middle point of the segment formed by the intersection of the two other bitten circles, as the point $\boldsymbol{q}$ of the Figure: it is the middle point of segment $\boldsymbol{q}_1\boldsymbol{q}_2$ (because $B(\boldsymbol{q}_1, c_b f(\boldsymbol{q}_1))$ intersects with $B(\boldsymbol{p}_2, c_b f(\boldsymbol{p}_2))$, and $B(\boldsymbol{q}_2, c_b f(\boldsymbol{q}_2))$ intersects with $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$).

Usually, the advancing front is represented as a circular list of already placed points. In our method, it is represented as a set of arcs and boundary segments. We always choose the next Steiner point on the front itself. In other words, the front itself is a subset of the feasible region for the selection of new mesh vertices, making it easier to choose the next point.

The intersection of two arcs or an arc and a boundary segment provides a good candidate for a new Steiner point, whose biting circle will reduce the interior. *Isn't this the way we take a bite on a biscuit or an apple? We center our bite more or less around a sharpest nose. Then bite after bite, we eat away the boundary of the food and move to its interior.*

Also noting that the packing circle is never appeared in our biting method. We will prove that the centers of all biting circles define a circle packing by carefully choose the size of the packing circle.

5.2 Quality Guarantee of the Biting Scheme

In this section we show that the biting method generates a well-shaped mesh. Moreover, the size of this mesh is within a constant factor of the optimal. For the first statement we prove that the points placed by the biting method is well-spaced, i.e., they are centers of a β-packing with respect to a 1-Lipschitz spacing function. The size optimality then follows from the fact that the spacing function is maximal.

For each point that the biting scheme generates, we would like to define a *packing circle*. Because the biting circles defined by our scheme overlap among themselves, the packing circle of a point is chosen to be smaller than its biting circle. Let us focus on a particular point $\boldsymbol{x}$. From the specification of the biting scheme, the biting circle at $\boldsymbol{x}$ is $B(\boldsymbol{x}, c_b h(\boldsymbol{x}))$, where $c_b < 1$ is a positive constant. We now choose another positive constant $c_p < c_b$, and define the packing circle at $\boldsymbol{x}$ to be $B(\boldsymbol{x}, c_p h(\boldsymbol{x}))$. Where the subscribe p of constant c_p denotes $p$acking. See Figure 3 (a).

In principle, in the biting scheme, c_b is the largest constant that satisfies the constraint of the control spacing function so that the resulting mesh has an optimal size. Similarly, c_p is the largest constant to ensure no two packing circles overlap so that the gaps among the packing circles are minimized. The following lemma of Ruppert [24] suggests that c_p should be smaller than 1/2.

Lemma 5.1 *For each input vertex $\boldsymbol{x}$ of an input PSLG Ω, $nn(\boldsymbol{x})$, the distance to its nearest input vertex, is at least $lfs(\boldsymbol{x})$.*

Proof: The circle centered at $\boldsymbol{x}$ with radius $nn(\boldsymbol{x})$ contains at least two non-incident input features of the PSLG, and hence $nn(\boldsymbol{x}) \geq lfs(\boldsymbol{x})$. □

As defined in the biting scheme $h(\boldsymbol{x}) \leq lfs(\boldsymbol{x})$ and thus $h(\boldsymbol{x}) \leq nn(\boldsymbol{x})$. If $c_p \leq \frac{1}{2}$, then Lemma 5.1 ensures that the packing circles of any two input vertices of Ω do not intersect.

The fact that $f()$ is an α-Lipschitz spacing function implies the following local similarity among the close neighborhood of point $\boldsymbol{x}$.

Lemma 5.2 *Let $f()$ be an α-Lipschitz spacing function over a domain Ω. For any two points $\boldsymbol{x}, \boldsymbol{y} \in \Omega$ and $c_0 \geq 0$, if $||\boldsymbol{x} - \boldsymbol{y}|| \leq c_0 f(\boldsymbol{x})$, then*

$$(1 - \alpha c_0) f(\boldsymbol{x}) \leq f(\boldsymbol{y}) \leq (1 + \alpha c_0) f(\boldsymbol{x}). \tag{1}$$

The following lemma specifies the relation between the biting constant c_b and the packing constant c_p.

Lemma 5.3 *If a spacing function $f()$ is α-Lipschitz, and $||\boldsymbol{x}-\boldsymbol{y}|| \geq \frac{2\gamma}{1-\alpha\gamma} \min(f(\boldsymbol{x}), f(\boldsymbol{y}))$, where $1-\alpha\gamma > 0$, then the interior of the two circles $B(\boldsymbol{x}, \gamma f(\boldsymbol{x}))$ and $B(\boldsymbol{y}, \gamma f(\boldsymbol{y}))$ do not intersect.*

Proof: Without loss of generality, we assume that $f(\boldsymbol{y}) = \min(f(\boldsymbol{x}), f(\boldsymbol{y}))$. Because $f()$ is α-Lipschitz, $f(\boldsymbol{x}) \leq f(\boldsymbol{y}) + \alpha||x - y||$. Then $\gamma f(\boldsymbol{x}) + \gamma f(\boldsymbol{y}) \leq 2\gamma f(\boldsymbol{y}) + \alpha\gamma||\boldsymbol{x} - \boldsymbol{y}|| \leq (1 - \alpha\gamma)||\boldsymbol{x} - \boldsymbol{y}|| + \alpha\gamma||\boldsymbol{x} - \boldsymbol{y}||$. The lemma follows from $\gamma f(\boldsymbol{x}) + \gamma f(\boldsymbol{y}) \leq ||\boldsymbol{x} - \boldsymbol{y}||$. □

The biting scheme works for any control spacing function $f()$ with Lipschitz condition. Assume that the spacing function $h()$ used in the biting scheme is α-Lipschitz over the domain Ω. Let S_b be the set of biting circles generated by the biting scheme, and S_p be the set of corresponding packing circles defined by a constant c_p. Lemma 5.3 implies that if $c_b \geq \frac{2c_p}{1-\alpha c_p}$ and $1-\alpha c_p > 0$, then no two packing circles overlap.

Lemma 5.4 *If the packing constant $c_p = \frac{c_b}{2+\alpha c_b}$, then there is no overlap between the packing circles S_p defined by c_p.*

Proof: Let $B(\boldsymbol{x}, c_p f(\boldsymbol{x}))$ and $B(\boldsymbol{y}, c_p f(\boldsymbol{y}))$ be any two of the packing circles defined by the biting vertices. Then from the biting scheme, we know that either $\boldsymbol{x}$ is bitten before $\boldsymbol{y}$, or $\boldsymbol{y}$ is bitten before $\boldsymbol{x}$. It implies that $||\boldsymbol{x}-\boldsymbol{y}|| \geq c_b \min(f(\boldsymbol{x}), f(\boldsymbol{y}))$. From $c_p = \frac{c_b}{2+\alpha c_b}$, we have $c_b = \frac{2c_p}{1-\alpha c_p}$ and $1-\alpha c_p = 2/(2+\alpha c_b) > 0$. The lemma follows from Lemma 5.3. □

Thus the biting circle $B(\boldsymbol{x}, c_b h(\boldsymbol{x}))$ is like a protecting circle of $\boldsymbol{x}$: it prevent any point that potentially conflicts with $\boldsymbol{x}$ from being chosen. The following theorem shows that the biting method generates a good circle packing.

Theorem 5.5 *The circle packing S_p is a $\beta = \frac{1+\alpha c_b}{1-\alpha c_b}$-packing with respect to the spacing function $\frac{2c_b}{2+\alpha c_b} f()$.*

Proof: The biting scheme guarantees that each point $\boldsymbol{y}$ in the domain Ω is covered by at least one biting circle of S_b. Let $B(\boldsymbol{x}, c_b f(\boldsymbol{x}))$ be the biting circle that covers $\boldsymbol{y}$. Then $||\boldsymbol{y}-\boldsymbol{x}|| \leq c_b f(\boldsymbol{x})$. Because $f()$ is α-Lipschitz, $f(\boldsymbol{y}) \geq (1-\alpha c_b) f(\boldsymbol{x})$. Noting $c_p = 2c_b/(2+\alpha c_b)$, we have

$$\begin{aligned}
\beta c_p f(\boldsymbol{y}) + c_p f(\boldsymbol{x}) &= \frac{1+\alpha c_b}{1-\alpha c_b} c_p f(\boldsymbol{y}) + c_p f(\boldsymbol{x}) \\
&= \frac{c_b}{2+\alpha c_b}\left(\frac{1+\alpha c_b}{1-\alpha c_b} f(\boldsymbol{y}) + f(\boldsymbol{x})\right) \\
&\geq \frac{c_b}{2+\alpha c_b}\left(\frac{1+\alpha c_b}{1-\alpha c_b}(1-\alpha c_b) f(\boldsymbol{x}) + f(\boldsymbol{x})\right) \\
&= \frac{c_b}{2+\alpha c_b}((1+\alpha c_b) f(\boldsymbol{x}) + f(\boldsymbol{x})) \\
&= c_b f(\boldsymbol{x}) \\
&\geq ||\boldsymbol{x}-\boldsymbol{y}||.
\end{aligned}$$

Hence, for point $\boldsymbol{y}$ there is a circle in S_p that overlaps with $B(\boldsymbol{y}, \beta c_p f(\boldsymbol{y}))$, i.e., there is no large gap at $\boldsymbol{y}$. From the β-packing definition 4.1, S_p is β-packing with respect to spacing function $2c_p f()$. □

The control spacing $h()$ used in the biting scheme is a 1-Lipschitz function. Hence $2c_p h()$ is a $2c_p$-Lipschitz function. Because $c_p \leq \frac{1}{2}$, $2c_p h()$ is also a 1-Lipschitz function. Then it follows from Theorems 4.1 and 5.5 that the Delaunay triangulation of the centers of S_p is a well-shaped mesh.

Theorem 5.6 *The biting method generates a well-shaped mesh.*

5.3 The Spacing Conformation and the Size Guarantee

In this section, we show that the nearest neighbor value of any point in the domain is related to the control spacing function $f()$ by a constant factor. This relation enables us to show that the biting scheme generates a mesh whose size is optimal with a constant factor.

Assume that the required spacing function $f()$ is α-Lipschitz. Let $c_e = 2c_b/(1-\alpha c_b)$, where the subscribe e denotes the *e*xtension of the biting circle. From Lemma 5.3, we know that the two biting circles $B(\boldsymbol{x}, c_b f(\boldsymbol{x}))$ and $B(\boldsymbol{y}, c_b f(\boldsymbol{y}))$ do not intersect if $||\boldsymbol{x}-\boldsymbol{y}|| \geq c_e f(\boldsymbol{x})$ and $\alpha c_b < 1$. In other words, there is at least one other mesh vertex generated in $B(\boldsymbol{x}, c_e f(\boldsymbol{x}))$ other than $\boldsymbol{x}$. We obtain the following lemma about the nearest neighbor of the mesh vertices.

Lemma 5.7 *For each vertex $\boldsymbol{x}$ of the mesh generated by the biting method, $nn(\boldsymbol{x})$ satisfies*

$$c_b(1-\alpha c_b) f(\boldsymbol{x}) \leq nn(\boldsymbol{x}) \leq \frac{2c_b}{1-\alpha c_b} f(\boldsymbol{x}). \tag{2}$$

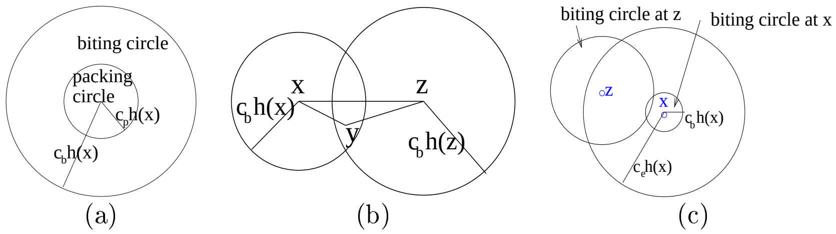

Figure 3: The biting circles and packing circles: (a) the biting circle and the packing circle centered at point $\boldsymbol{x}$; (b) a point $\boldsymbol{y}$ is covered by at least two biting circles, for example, $B(\boldsymbol{x}, c_b f(\boldsymbol{x}))$ and $B(\boldsymbol{z}, c_b f(\boldsymbol{z}))$; (c) for any biting circle $B(\boldsymbol{x}, c_b f(\boldsymbol{x}))$, there must exist a biting circle centered at $\boldsymbol{z}$, such that $\boldsymbol{z}$ is contained in $B(\boldsymbol{x}, c_e f(\boldsymbol{x}))$, where $c_e = 2c_b/(1-\alpha c_b)$.

Proof: First, if there is no other mesh vertex other than $\boldsymbol{x}$ that lies in the interior of the biting circle $B(\boldsymbol{x}, c_b f(\boldsymbol{x}))$. Then, $c_b f(\boldsymbol{x}) \leq nn(\boldsymbol{x})$. If there is at least one mesh vertex $\boldsymbol{y}$ inside $B(\boldsymbol{x}, c_b f(\boldsymbol{x}))$, then vertex $\boldsymbol{y}$ must be bitten before $\boldsymbol{x}$. Then we have $c_b f(\boldsymbol{y}) \leq ||\boldsymbol{y} - \boldsymbol{x}|| \leq c_b f(\boldsymbol{x})$. Because $f()$ is α-Lipschitz, $f(\boldsymbol{y}) \geq (1-\alpha c_b) f(\boldsymbol{x})$. Hence, $||\boldsymbol{y} - \boldsymbol{x}|| \geq c_b(1-\alpha c_b) f(\boldsymbol{x})$. By Lemma 5.3, for any point $\boldsymbol{y}$ in the exterior of the circle $B(\boldsymbol{x}, c_e f(\boldsymbol{x}))$, the biting circle of $\boldsymbol{y}$ does not overlap with that of $\boldsymbol{x}$. Hence, if $nn(\boldsymbol{x}) > c_e f(\boldsymbol{x})$, then the boundary of $B(\boldsymbol{x}, c_b f(\boldsymbol{x}))$ is not covered by any other biting circles. See Figure 3 (c). The lemma then follows from property that every point in the domain is covered by at least one biting circle. □

We now show that the nearest neighbor function of each non-mesh point in the domain is linearly related to the control spacing function.

Lemma 5.8 *Assume point $\boldsymbol{y} \in \Omega$ is not a mesh vertex. Then $nn(\boldsymbol{y})$ defined by the mesh generated by the biting method satisfies*

$$\frac{c_b f(\boldsymbol{y})}{2+2\alpha c_b} \leq nn(\boldsymbol{y}) \leq \frac{(3-\alpha c_b) c_b f(\boldsymbol{y})}{(1-\alpha c_b)^2}. \tag{3}$$

Proof: First, there is at least one biting circle, say $B(\boldsymbol{x}, c_b f(\boldsymbol{x}))$, that covers point $\boldsymbol{y}$. There must be at least one mesh vertex $\boldsymbol{z}$ other than $\boldsymbol{x}$ in $B(\boldsymbol{x}, c_e f(\boldsymbol{x}))$. See Figure 3 (c). Then $||\boldsymbol{z} - \boldsymbol{y}|| \leq ||\boldsymbol{z} - \boldsymbol{x}|| + ||\boldsymbol{x} - \boldsymbol{y}|| \leq c_e f(\boldsymbol{x}) + c_b f(\boldsymbol{x}) = (c_b + c_e) f(\boldsymbol{x})$. So $nn(\boldsymbol{y}) \leq max(||\boldsymbol{x} - \boldsymbol{y}||, ||\boldsymbol{z} - \boldsymbol{y}||) \leq (c_b + c_e) f(\boldsymbol{x})$. Then the inequality $f(\boldsymbol{y}) \geq f(\boldsymbol{x}) - \alpha ||\boldsymbol{x} - \boldsymbol{y}|| \geq (1-\alpha c_b) f(\boldsymbol{x})$ implies that

$$nn(\boldsymbol{y}) \leq (c_b + c_e) f(\boldsymbol{y})/(1-\alpha c_b).$$

We have two cases: (1) only one biting circle contains $\boldsymbol{y}$, and (2) two or more biting circles contain $\boldsymbol{y}$.

In the first case, let us assume that $\boldsymbol{y} \in B(\boldsymbol{x}, c_b f(\boldsymbol{x}))$. Thus $||\boldsymbol{y} - \boldsymbol{z}|| > c_b f(\boldsymbol{z})$ holds for any other mesh vertex $\boldsymbol{z}$, i.e., $\boldsymbol{y} \notin B(\boldsymbol{z}, c_b f(\boldsymbol{z}))$. Because $f()$ is α-Lipschitz, $f(\boldsymbol{y}) \leq f(\boldsymbol{z}) + \alpha ||\boldsymbol{y} - \boldsymbol{z}|| \leq (\alpha + 1/c_b) ||\boldsymbol{y} - \boldsymbol{z}||$. Then for any mesh vertex $\boldsymbol{z}$, if $\boldsymbol{y} \notin B(\boldsymbol{z}, c_b f(\boldsymbol{z}))$, then $||\boldsymbol{y} - \boldsymbol{z}|| \geq \frac{c_b}{1+\alpha c_b} f(\boldsymbol{y})$. Therefore, by the definition of the nearest neighbor function, $nn(\boldsymbol{y}) \geq min_{\boldsymbol{z} \neq \boldsymbol{x}}(||\boldsymbol{y} - \boldsymbol{z}||)$, which implies that

$$nn(\boldsymbol{y}) \geq \frac{c_b}{1+\alpha c_b} f(\boldsymbol{y}).$$

In the second case, assume that $B(\boldsymbol{x}, c_b f(\boldsymbol{x}))$ and $B(\boldsymbol{z}, c_b f(\boldsymbol{z}))$ are two of the biting circles that contain $\boldsymbol{y}$. See Figure 3 (b). Further assume that $B(\boldsymbol{x}, c_b f(\boldsymbol{x}))$ is bitten before $B(\boldsymbol{z}, c_b f(\boldsymbol{z}))$. Hence $||\boldsymbol{x} - \boldsymbol{z}|| \geq c_b f(\boldsymbol{x})$. Because $f()$ is α-Lipschitz and $\boldsymbol{y} \in B(\boldsymbol{x}, c_b f(\boldsymbol{x}))$, $f(\boldsymbol{y}) \leq (1+\alpha c_b) f(\boldsymbol{x})$. By the triangle inequality, $max(||\boldsymbol{y} - \boldsymbol{x}||, ||\boldsymbol{y} - \boldsymbol{z}||) \geq \frac{1}{2} ||\boldsymbol{x} - \boldsymbol{z}|| \geq c_b/2 f(\boldsymbol{x}) \geq c_b/(2+2\alpha c_b) f(\boldsymbol{y})$.

Therefore, the second smallest distance from $\boldsymbol{y}$ to the set of mesh vertices whose biting circles contain $\boldsymbol{y}$ is at least $c_b/(2+2\alpha c_b) f(\boldsymbol{y})$. In addition, from the analysis of the first case, the smallest distance from $\boldsymbol{y}$ to the set of mesh vertices whose biting circles do not contain $\boldsymbol{y}$ is at least $c_b/(1+\alpha c_b) f(\boldsymbol{y})$, Thus,

$$nn(\boldsymbol{y}) \geq c_b/(2+2\alpha c_b) f(\boldsymbol{y}).$$

Consequently,

$$\frac{c_b}{2+2\alpha c_b} f(\boldsymbol{y}) \le nn(\boldsymbol{y}) \le \frac{c_b + c_e}{1-\alpha c_b} f(\boldsymbol{y}),$$

and the lemma follows from $c_e = 2c_b/(1-\alpha c_b)$. $\square$

We will use the following lemma of Miller *et al.* [18] to prove the size optimality of the generated mesh.

Lemma 5.9 (Size of a Well-shaped Mesh [18]) *If M is an α-well-shaped mesh of n elements, then*

$$n = \Theta\left(\int_\Omega \frac{dA}{nn_M^2}\right). \tag{4}$$

Lemma 5.10 (Size of Mesh Respect to the Space Control [31]) *There exists a constant c such that if M is a well-shaped mesh of n elements over a domain Ω that satisfies the control spacing function $f()$, then*

$$n \ge c \int_\Omega \frac{dA}{f^2}. \tag{5}$$

The $nn()$ function deduced from the mesh generated by the biting method is within a constant factor of $f()$ implies the following theorem.

Theorem 5.11 *Size of the mesh generated by the* biting *method is within a constant factor of the optimal possible.*

6 The Minimal Angle of the Mesh and the Boundary Protection

Theorems 4.1 and 5.5 show that the biting scheme generates a well shaped mesh, see Theorem 5.6. However, the constant bound on the minimal angle so derived may be too small. In this section, we provide a better analysis of the lower bound of the minimal angle.

In our analysis, we divide the triangle elements into two subsets: the first subset contains all elements that are visible from their circumcenters and the second subset contains all elements that are not completely visible from their circumcenters, i.e., elements that are close to the boundary. See Figure 4 (a) (b) (c).

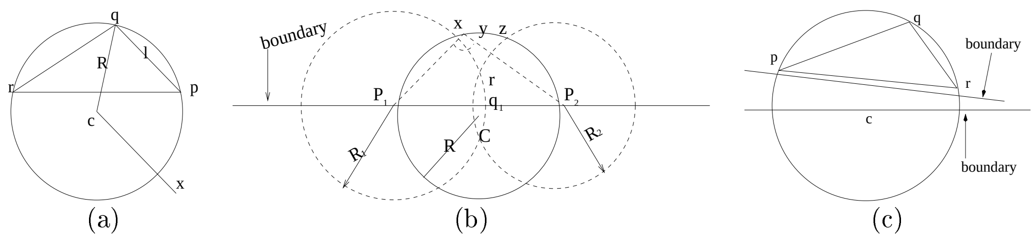

Figure 4: The triangles of the mesh: (a) a triangle $\triangle_{\boldsymbol{pqr}}$ of the first subset. (b) a triangle $\triangle_{\boldsymbol{xyz}}$ of the second subset. (c) a triangle $\triangle_{\boldsymbol{pqr}}$ of the second subset, even the circumcenter $\boldsymbol{c}$ is in the domain.

Assume that the spacing function $f()$ used by the biting scheme is α-Lipschitz. Let $t = \alpha c_b$. We use t to control the quality and the size of the mesh generated by the biting scheme. In our proof, we assume $t \le 1/3$, although for a practical scheme, we can set t to $1/2$.

6.1 The Minimal Angle of Elements of the First Subset

The following lemma bounds the minimal angle of the triangles in the first subset.

Lemma 6.1 *Let $\triangle\boldsymbol{pqr}$ be a triangle element of the first subset. Let l be the length of the shortest edge of $\triangle\boldsymbol{pqr}$. Let R be the radius of the circumcircle $\mathcal{C}$ of $\triangle\boldsymbol{pqr}$. Then*

$$\frac{R}{l} \le \frac{1}{1-2t}. \tag{6}$$

Proof: Let $\boldsymbol{c}$ be the circumcenter of $\triangle\boldsymbol{pqr}$. Assume $\boldsymbol{c} \in B(\boldsymbol{x}, c_b f(\boldsymbol{x}))$. See Figure 4 (a). Then $||\boldsymbol{c}-\boldsymbol{x}|| \le c_b f(x)$. Because the mesh is a Delaunay triangulation, $\boldsymbol{x}$ is not in the interior of circumcircle of $\triangle\boldsymbol{pqr}$, i.e., $||\boldsymbol{x}-\boldsymbol{c}|| \ge R$. Thus $f(\boldsymbol{x}) \ge R/c_b$. Because $f()$ is α-Lipschitz, $f(\boldsymbol{c}) \ge f(\boldsymbol{x}) - \alpha||\boldsymbol{c}-\boldsymbol{x}||$. Also because $||\boldsymbol{c}-\boldsymbol{x}|| \le c_b f(x)$, $f(\boldsymbol{c}) \ge (1-\alpha c_b) f(\boldsymbol{x}) \ge (1-\alpha c_b)R/c_b$.

Without loss of generality, assume that $||\boldsymbol{p}-\boldsymbol{q}|| = l$, and $B(\boldsymbol{q}, c_b f(\boldsymbol{p}))$ is bitten before $B(\boldsymbol{q}, c_b f(\boldsymbol{q}))$, implying $l \ge c_b f(\boldsymbol{p})$. Because $f()$ is α-Lipschitz, $f(\boldsymbol{c}) \le f(\boldsymbol{p}) + \alpha R \le l/c_b + \alpha R$. Using $f(\boldsymbol{c}) \ge (1-\alpha c_b)R/c_b$, we have $(1-t)R/c_b \le (l+tR)/c_b$. Then the lemma follows. □

Consequently, we have the following lemma.

Lemma 6.2 *The minimal angle θ of any triangle of the first subset satisfies*

$$\sin(\theta) = \frac{l}{2R} \ge \frac{1}{2} - t. \tag{7}$$

6.2 Boundary Protection

We show that the mesh vertices generated by the biting scheme are not too close to the boundary with respect to its control spacing. Note that our scheme first removes the biting circles centered at the input vertices of the domain. It then progressively removes the biting circles centered at some points on the boundary of the domain.

There are two cases for selecting the center $\boldsymbol{q}$ of the new biting circle for protecting the boundary. The first type of the choices is the intersection point of a previously selected biting circle $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$ with the input boundary. In this case, q is on the boundary of biting circle $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$. We label this point $\boldsymbol{q}$ with $B(\boldsymbol{p}_1)$. We call $B(\boldsymbol{q}, c_b f(\boldsymbol{q}))$ the *progressive circle*. See Figure 5 (a).

The second type of choices is a vertex $\boldsymbol{q}$ that is the middle point of a segment $\boldsymbol{q}_1\boldsymbol{q}_2$, where $\boldsymbol{q}_1$, $\boldsymbol{q}_2$ is the intersection of $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$ and $B(\boldsymbol{p}_2, c_b f(\boldsymbol{p}_2))$ with the boundary respectively. We label this point $\boldsymbol{q}$ with $M(\boldsymbol{p}_1, \boldsymbol{p}_2)$. We call $B(\boldsymbol{q}, c_b f(\boldsymbol{q}))$ the *middle circle*. See Figure 5 (b).

Let $\boldsymbol{p}_1$,$\boldsymbol{p}_2$ and $\boldsymbol{p}_3$ be three consecutive mesh vertices on an input boundary generated by the biting scheme. From above description, we know that $\boldsymbol{p}_2$ is labeled by $B(\boldsymbol{p}_1)$ or $B(\boldsymbol{p}_3)$ or $M(\boldsymbol{p}_1, \boldsymbol{p}_3)$.

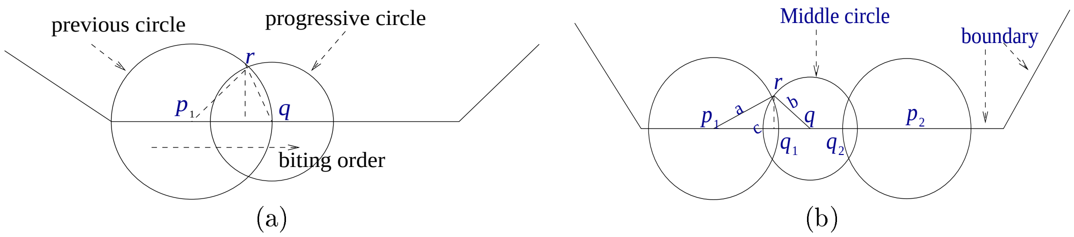

Figure 5: The biting circles on the boundary: (a) the center $\boldsymbol{q}$ of the new biting circle is labeled by $B(\boldsymbol{p}_1)$. $\boldsymbol{p}_1$ is chosen prior to $\boldsymbol{q}$. (b) the center $\boldsymbol{q}$ of the new biting circle is labeled by $M(\boldsymbol{p}_1, \boldsymbol{p}_2)$, i.e., it is the middle point of segment $\boldsymbol{q}_1\boldsymbol{q}_2$ formed by the intersection of $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$ and $B(\boldsymbol{p}_2, c_b f(\boldsymbol{p}_2))$ with the boundary.

Let *the boundary biting circles* be the circles centered on the input boundary generated by the biting scheme. It contains either progressive biting circles or middle circles. The following lemma guarantees that the progressive biting circles do not generate vertices that are too close to the input boundary.

Lemma 6.3 *Assume $\boldsymbol{q}$ is labeled by $B(\boldsymbol{p}_1)$. Let $\boldsymbol{r}$ be one of the intersection points of $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$ with $B(\boldsymbol{q}, c_b f(\boldsymbol{q}))$. Then the distance $h_{\boldsymbol{r}}$ of point $\boldsymbol{r}$ to the boundary $\boldsymbol{p}_1\boldsymbol{q}$ satisfies*

$$\frac{h_{\boldsymbol{r}}}{c_b f(\boldsymbol{p}_1)} \geq (1-t)\sqrt{(3-t)(1+t)}/2. \tag{8}$$

Proof: Because $\boldsymbol{q}$ is on circle $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$, $||\boldsymbol{q}-\boldsymbol{p}_1|| = c_b f(\boldsymbol{p}_1)$, and $||\boldsymbol{q}-\boldsymbol{r}|| = c_b f(\boldsymbol{q})$. See Figure 5 (a). Let θ be the angle formed by segments $\boldsymbol{p}_1\boldsymbol{r}$ and $\boldsymbol{p}_1\boldsymbol{q}$. From $f(\boldsymbol{q}) \geq (1-\alpha c_b) f(\boldsymbol{p}_1)$, we have

$$\begin{aligned} \sin(\theta/2) &= ||\boldsymbol{q}-\boldsymbol{r}||/(2c_b f(\boldsymbol{p}_1)) \\ &\geq c_b(1-\alpha c_b) f(\boldsymbol{p}_1)/(2c_b f(\boldsymbol{p}_1)) \\ &= (1-t)/2. \end{aligned}$$

Hence $\theta/2 \geq \sin^{-1}((1-t)/2)$. Similarly, from $f(\boldsymbol{q}) \leq (1+\alpha c_b) f(\boldsymbol{p}_1)$, we have $\theta/2 \leq \sin^{-1}((1+t)/2)$. Hence,

$$\begin{aligned} \sin(\theta) &\geq min((1-t)\sqrt{(3-t)(1+t)}/2, (1+t)\sqrt{(3+t)(1-t)}/2) \\ &= (1-t)\sqrt{(3-t)(1+t)}/2. \end{aligned}$$

The lemma then follows from $\frac{h_{\boldsymbol{r}}}{c_b f(\boldsymbol{p}_1)} = \sin(\theta)$. □

The following lemma guarantees that the middle circles do not generate vertices that are too close to the input boundary.

Lemma 6.4 *Assume mesh vertex $\boldsymbol{q}$ is labeled with $M(\boldsymbol{p}_1, \boldsymbol{p}_2)$. Let $\boldsymbol{r}$ be one of the intersection points of $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$ with $B(\boldsymbol{q}, c_b f(\boldsymbol{q}))$. Then the distance $h_{\boldsymbol{r}}$ of point $\boldsymbol{r}$ to the boundary $\boldsymbol{p}_1\boldsymbol{p}_2$ satisfies*

$$\frac{h_{\boldsymbol{r}}}{c_b f(\boldsymbol{p}_1)} \geq (1-t)\sqrt{(t^2-4t+7)(1-t^2)}/4. \tag{9}$$

Proof: By the assumption of the lemma, $\boldsymbol{q}$ is labeled with $M(\boldsymbol{p}_1, \boldsymbol{p}_2)$, i.e., $\boldsymbol{q}$ is the middle point of segment $\boldsymbol{q}_1\boldsymbol{q}_2$, where $\boldsymbol{q}_1$, $\boldsymbol{q}_2$ are the intersection points of circle $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$ and $B(\boldsymbol{p}_2, c_b f(\boldsymbol{p}_2))$ with boundary, respectively. See Figure 5 (b). From the biting scheme, we know that $B(\boldsymbol{q}_1, c_b f(\boldsymbol{q}_1))$ intersects with $B(\boldsymbol{p}_2, c_b f(\boldsymbol{p}_2))$, and $B(\boldsymbol{q}_2, c_b f(\boldsymbol{q}_2))$ intersects with $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$. Otherwise, we will not select $\boldsymbol{q}$ as the mesh vertex. See Figure 2. Hence, $||\boldsymbol{q}_1-\boldsymbol{q}|| \leq c_b f(\boldsymbol{q}_1)/2$. Let $a = c_b f(\boldsymbol{p}_1)$, $b = c_b f(\boldsymbol{q})$, $c = ||\boldsymbol{p}_1-\boldsymbol{q}|| = a + ||\boldsymbol{q}_1-\boldsymbol{q}||$, and $d = c_b f(\boldsymbol{q}_1)$. In the analysis, we assume a and d are fixed.

Because $f()$ is α-Lipschitz, $f(\boldsymbol{q}) \geq f(\boldsymbol{q}_1) - \alpha||\boldsymbol{q}_1-\boldsymbol{q}||$. Hence,

$$\begin{aligned} b &= c_b f(\boldsymbol{q}) \\ &\geq c_b f(\boldsymbol{q}_1) - \alpha c_b ||\boldsymbol{q}_1-\boldsymbol{q}|| \\ &\geq c_b f(\boldsymbol{q}_1) - \alpha c_b c_b f(\boldsymbol{q}_1)/2 \\ &= (1-t/2)d. \end{aligned}$$

Similarly, $b \leq (1+t/2)d$. Because $B(\boldsymbol{q}_1, c_b f(\boldsymbol{q}_1))$ intersects with $B(\boldsymbol{p}_2, c_b f(\boldsymbol{p}_2))$, $B(\boldsymbol{q}_1, c_b f(\boldsymbol{q}_1))$ contains segment $\boldsymbol{q}_1\boldsymbol{q}_2$. Thus, $||\boldsymbol{q}_1-\boldsymbol{q}|| \leq d/2$, i.e., $a \leq c \leq a + d/2$.

It is easy to show that $h_{\boldsymbol{r}}$ get the smallest distance value when setting b to the smallest possible value, and setting c to the largest possible value. We would like to minimize b and maximize c, i.e., $b = (1-t/2)d$, $c = a + d/2$. Let $s = (a+b+c)/2$. Let $S_\triangle$ be the area of the triangle $\triangle_{\boldsymbol{p}_1\boldsymbol{q}\boldsymbol{r}}$. Note that $h_{\boldsymbol{r}} = 2S_\triangle/c$, and

$$\begin{aligned} S_\triangle &= \sqrt{s(s-a)(s-b)(s-c)} \\ &= \sqrt{(a+\frac{3-t}{4}d)(\frac{3-t}{4}d)(a-\frac{1-t}{4}d)(\frac{1-t}{4}d)}. \end{aligned}$$

Let $x = a/d$. Because $f()$ is α-Lipschitz, from Lemma 5.2, we have $(1-t)f(\boldsymbol{p}_1) \leq f(\boldsymbol{q}_1) \leq (1+t)f(\boldsymbol{p}_1)$. In other words, the value of x satisfies $1/(1+t) \leq x \leq 1/(1-t)$.

Therefore,

$$\begin{aligned} h_{\boldsymbol{r}} &\geq 2\sqrt{(a+\frac{3-t}{4}d)(\frac{3-t}{4}d)(a-\frac{1-t}{4}d)(\frac{1-t}{4}d)}/(a+d/2) \\ &= \sqrt{(3-t)(1-t)}a\sqrt{(x+\frac{3-t}{4})(x-\frac{1-t}{4})}/(2x^2+x). \end{aligned}$$

Because $\frac{1}{1+t} \leq x \leq \frac{1}{1-t}$, and $g(x) = \sqrt{(x+\frac{3-t}{4})(x-\frac{1-t}{4})}/(2x^2+x)$ is minimized when $x = \frac{1}{1-t}$,

$$\frac{h_{\boldsymbol{r}}}{a} \geq (1-t)\sqrt{(t^2-4t+7)(1-t^2)}/4.$$

□

Similarly, the ratio $h_{\boldsymbol{r}}/c$ achieves its minimum when $b = (1-t/2)d$ and $c = a + d/2$. Hence

$$\frac{h_{\boldsymbol{r}}}{c} \geq (1-t)(3-t)\sqrt{(t^2-4t+7)(1-t^2)}/8. \tag{10}$$

6.3 The Minimal Angle of Elements of the Second Subset

We now consider elements in the second subset. Let $\triangle_{\boldsymbol{xyz}}$ be a mesh element from the second subset. Let $\mathcal{C}$ be the circumcircle of $\triangle_{\boldsymbol{xyz}}$. Let $\boldsymbol{c}$ be the center of $\mathcal{C}$. Let R be the radius of $\mathcal{C}$. Assume $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ are the two closest mesh vertices on the boundary that separates $\triangle_{\boldsymbol{xyz}}$ and $\boldsymbol{c}$. Note that $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ may be one of the vertices among $\{\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}\}$. Let $\boldsymbol{r}$ be one of the intersection points of $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$ with $B(\boldsymbol{p}_2, c_b f(\boldsymbol{p}_2))$. Let $h_{\boldsymbol{r}}$ be the distance of point $\boldsymbol{r}$ to boundary segment $\boldsymbol{p}_1\boldsymbol{p}_2$. Let $c = ||\boldsymbol{p}_1 - \boldsymbol{p}_2||$. Let L be the half of the length of the chord on that boundary cut by the circumcircle. Then $L \leq c/2$. See Figure 4 (b) and Figure 5 (b). Then there exists an input boundary that separates $\triangle_{\boldsymbol{xyz}}$ from $\boldsymbol{c}$. We only need to consider such a boundary that is the closest to $\triangle_{\boldsymbol{xyz}}$.

First, $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ must be the two consecutive boundary mesh vertices generated by the biting scheme. Otherwise, there would be at least one mesh vertex in the interior of the circumcircle of the triangle $\triangle_{\boldsymbol{xyz}}$, which is a contradiction to the Delaunay property. Second, neither $\boldsymbol{p}_1$ is on $B(\boldsymbol{p}_2, c_b f(\boldsymbol{p}_2))$, nor $\boldsymbol{p}_2$ is on $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$. Otherwise, assume that $\boldsymbol{p}_1$ is on $B(\boldsymbol{p}_2, c_b f(\boldsymbol{p}_2))$, and $\boldsymbol{x}$ is not on that boundary. Because $\boldsymbol{p}_1\boldsymbol{p}_2$ separates the center $\boldsymbol{c}$ from the element $\triangle_{\boldsymbol{xyz}}$, the angle formed by $\boldsymbol{p}_1\boldsymbol{x}$ and $\boldsymbol{x}\boldsymbol{p}_2$ is obtuse. Then $||\boldsymbol{p}_1 - \boldsymbol{x}|| < ||\boldsymbol{p}_1 - \boldsymbol{p}_2|| = c_b f(\boldsymbol{p}_1)$. See Figure 4 (b). Hence $\boldsymbol{x}$ is in the biting circle $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$. It is a contradiction to the assumption that $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$ is removed before $B(\boldsymbol{x}, c_b f(\boldsymbol{x}))$. Hence, without loss of generality, we only need to analyze the case that $\boldsymbol{p}_2$ is labeled with $M(\boldsymbol{p}_1, \boldsymbol{p}_3)$, where $\boldsymbol{p}_3$ is the other neighbor vertex of $\boldsymbol{p}_2$ on the boundary.

The circumcircle $\mathcal{C}$ must contain $\boldsymbol{r}$. If it does not contain $\boldsymbol{r}$, then points $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{z}$ will be in the biting circle centered at $\boldsymbol{p}_1$ or $\boldsymbol{p}_2$, which is a contradiction to the biting property. Thus we have the following lemma about the radius R of $\mathcal{C}$.

Lemma 6.5

$$R \leq (h_{\boldsymbol{r}}^2 + L^2)/(2h_{\boldsymbol{r}}).$$

Proof: Because $\mathcal{C}$ contains $\boldsymbol{r}$, $R - \sqrt{R^2 - L^2} \geq h_{\boldsymbol{r}}$. Then $R^2 \geq (\sqrt{R^2 - L^2} + h_{\boldsymbol{r}})^2$, which implies that $R \leq (h_{\boldsymbol{r}}^2 + L^2)/(2h_{\boldsymbol{r}})$. □

The following lemma gives an upper bound on R/c.

Lemma 6.6 *If $t \leq 1/2$, then*

$$R \leq \frac{5}{8}c. \tag{11}$$

Proof: Let $u = c/2$. From lemma 6.5, and $L \leq c/2$, we have $R \leq (h_{\boldsymbol{r}}^2 + u^2)/(2h_{\boldsymbol{r}})$. Notice that $R(x) = (x^2+u^2)/(2x)$ increases monotonically when $x \geq u$, and decreases monotonically when $0 < x \leq u$. From formula (10), we have

$$h_{\boldsymbol{r}}/c \geq (1-t)(3-t)\sqrt{(t^2-4t+7)(1-t^2)}/8.$$

Let $g_c(t) = (1-t)(3-t)\sqrt{(t^2-4t+7)(1-t^2)}/8$. Then $1/4 < 15\sqrt{7}/128 \leq g_c(t) \leq \sqrt{63}/8 < 1$, if $0 \leq t \leq 1/2$. Hence $h_{\boldsymbol{r}} \geq u/2$. Because $h_{\boldsymbol{r}} \leq c_b f(\boldsymbol{p}_1) \leq c$, i.e., $h_{\boldsymbol{r}} \leq 2u$, we have

$$R \leq \max(((u/2)^2 + u^2)/u), ((2u)^2+u^2)/(4u)) = 5c/8.$$

□

We now give a bound on the radius-edge ratio for the triangles of the second subset.

Lemma 6.7 *Let l be the length of the shortest edge of $\triangle_{\boldsymbol{xyz}}$. If $t < (\sqrt{17}-3)/2$, then*

$$\frac{R}{l} \leq \frac{5(3+t)}{8(2-3t-t^2)}. \tag{12}$$

Proof: Assume that $\boldsymbol{p}_2$ is labeled with $M(\boldsymbol{p}_1, \boldsymbol{p}_3)$. Let $a = c_b f(\boldsymbol{p}_1)$, and further assume that the shortest edge of $\triangle_{\boldsymbol{xyz}}$ is edge $\boldsymbol{xy}$, and that $\boldsymbol{x}$ is bitten before $\boldsymbol{y}$. Then $l \geq c_b f(\boldsymbol{x})$. Notice that $\boldsymbol{x}$ and $\boldsymbol{y}$ could be either $\boldsymbol{p}_1$ or $\boldsymbol{p}_2$.

If $\boldsymbol{x}$ is $\boldsymbol{p}_1$, then $l = ||\boldsymbol{x}-\boldsymbol{y}|| = ||\boldsymbol{p}_1 - \boldsymbol{y}|| \geq c_b f(\boldsymbol{p}_1) = a$, because the boundary circle $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$ is bitten before $B(\boldsymbol{y}, c_b f(\boldsymbol{y}))$. Otherwise, notice that the angle formed by segment $\boldsymbol{p}_1\boldsymbol{x}$ and $\boldsymbol{x}\boldsymbol{p}_2$ is obtuse. Hence $||\boldsymbol{p}_1 - \boldsymbol{x}|| \leq ||\boldsymbol{p}_1 - \boldsymbol{p}_2||$. Because $f()$ is α-Lipschitz, $f(\boldsymbol{x}) \geq f(\boldsymbol{p}_1) - \alpha||\boldsymbol{p}_1 - \boldsymbol{x}|| \geq f(\boldsymbol{p}_1) - \alpha||\boldsymbol{p}_1 - \boldsymbol{p}_2||$. Then $l \geq c_b f(\boldsymbol{x}) \geq a - t||\boldsymbol{p}_1 - \boldsymbol{p}_2||$. Combining both cases, we have

$$l \geq a - t * c.$$

From lemma 6.6, we have $R/l \leq 5c/(8(a - t*c))$, if $a - t*c > 0$.

Note that circle $B(\boldsymbol{p}_2, c_b f(\boldsymbol{p}_2))$ is the middle biting circle. Let $\boldsymbol{q}_1$ be the intersection point of $B(\boldsymbol{p}_1, c_b f(\boldsymbol{p}_1))$ with the boundary, which is between $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$, Then we have $c = a + ||\boldsymbol{q}_1 - \boldsymbol{p}_2|| \leq a + c_b f(\boldsymbol{q}_1)/2 \leq a + (1+t)a/2$. Hence, if $2 - 3t - t^2 > 0$, i.e., $t < (\sqrt{17}-3)/2$, then

$$\frac{R}{l} \leq \frac{5(3+t)}{8(2-3t-t^2)},$$

□

From Lemmas 6.1 and 6.7, we have the following theorem to bound the minimal angle of the mesh generated by the biting scheme.

Theorem 6.8 *The minimal angle θ of the mesh generated by the biting scheme satisfies*

$$\sin(\theta) \geq \frac{1}{2} - t. \tag{13}$$

Proof: From lemma 6.1 and 6.7, we have $\sin(\theta) \geq min(\frac{1}{2} - t, \frac{4(2-3t-t^2)}{5(3+t)})$ Then the theorem follows from the fact that $\frac{1}{2} - t < \frac{4(2-3t-t^2)}{5(3+t)}$. □

If control spacing $f()$ is 0-Lipschitz, then the minimal angle of the mesh is at least 30^o. Given the positive α-Lipschitz control spacing $f()$, there are two goals in generating the mesh. One goal is to bound the value $nn(\boldsymbol{x})/f(\boldsymbol{x})$ from upper and below, for every point $\boldsymbol{x}$ in the domain. The other goal is to bound the minimal angle θ of all mesh elements from below. Unfortunately, this two goals are at odds with each other. Hence there must be a tradeoff. For example, if we set $t = 1/4$, then the mesh generated by the biting scheme has the following property: for any point $\boldsymbol{y} \in \Omega$, $\frac{f(\boldsymbol{y})}{10\alpha} \leq nn(\boldsymbol{y}) \leq \frac{11 f(\boldsymbol{y})}{9\alpha}$; and the minimal angle θ is at least $asin(1/4) \simeq 14.48^o$. If we set $t = 1/3$, then for any point $\boldsymbol{y} \in \Omega$, $\frac{f(\boldsymbol{y})}{8\alpha} \leq nn(\boldsymbol{y}) \leq \frac{2f(\boldsymbol{y})}{\alpha}$; and the minimal angle θ is at least $asin(1/6) \simeq 9.59^o$.

7 Conclusion

In this paper, we present a new mesh generation scheme which combines the merits of the advancing front and the circle packing methods. Our scheme, the biting method, first applies some variations of an advancing front method to generate a quality circle packing; it then constructs the final mesh by Delaunay triangulation. Therefore, it is as simple and as practical as the advancing front methods. It is different to the previous approaches such as those that use the particle simulation [27] or maximal independent set (MIS) of oversampled points [19].

The biting scheme is theoretically efficient than the paving method because it explicitly maintains the set of candidates for new mesh points, and it does not have to handle the case when fronts meet each other or itself. By using circle packings, the new scheme resolves this difficulty that occurs at the end of the standard advancing front method. The standard advancing front methods, however, does not provide a quality guarantee, especially in the region where the fronts meet. The biting method is also provably good and the size of generated mesh is within a constant factor of the optimal.

Note that the biting method can be extended to generate 3D mesh: by replacing the biting circle as the biting sphere. The main difficulty for 3D version of biting is how to protect the input boundary face, such that there are no bad tetrehedra near the boundary face. One possible approach is to use the classic advancing front method to generate one layer of elements incident to the boundary face, then use our biting method to generate point set for the remaining interior domain. For other approaches to solve this problem in 3D, the reader is refered to [13].

References

[1] I. Babuška and A. K. Aziz. On the angle condition in the finite element method. *SIAM J. Numer. Anal.*, 13(2):214–226, 1976.

[2] M. Bern and D. Eppstein. Polynomial-size non-obtuse triangulation of polygons. *Int. J. Comp. Geom. Appl.*, 2:241–255, 1992.

[3] M. Bern and D. Eppstein. Quadrilateral meshing by circle packing. In *6th International Meshing Roundtable*, pages 7–20, 1997.

[4] M. Bern, D. Eppstein, and J. R. Gilbert. Provably good mesh generation. In *the 31st Annual Symposium on Foundations of Computer Science, IEEE*, pages 231–241, 1990.

[5] M. Berna, S. Mitchell, and J. Ruppert. Linear-size nonobtuse triangulation of polygons. In *Proc. of 10th Symp. on Computational Geometry*, pages 221–230, New York, 1994.

[6] T. D. Blacker. Paving: a new approach to automated quadrilateral mesh generation. *Int. Jour. for Numerical Methods in Eng*, 32:811–847, 1991.

[7] S. W. Cheng, T. K. Dey, H. Edelsbrunner, M. A. Facello, and S.-H. Teng. Silver exudation. In *Proc. 15th ACM Symposium on Computational Geometry*, 1999. to appear.

[8] L. P. Chew. Guaranteed-quality delaunay meshing in 3d (short version). In *13th ACM Sym. on Comp. Geometry*, pages 391–393, 1997.

[9] L. Paul Chew, N. Chrisochoides, and F. Sukup. Parallel constrained delaunay meshing. In S. A. Canann and S. Saigal, editors, *Trends in Unstructured Mesh Generation*, pages 89–96, 1997.

[10] P. J. Frey and H. Borouchaki. surface mesh evaluation. In *6th International Meshing Roundtable*, pages 363–374, 1997.

[11] X. Y. Li and S. H. Teng. Dynamic load balancing for parallel adaptive mesh refinement. In *5th International Symposium on Solving Irregularly Structured Problems in Parallel*, pages 144–155, Berkeley, 1998.

[12] X. Y. Li, S. H. Teng, and A. Üngör. Simultaneous refinement and coarsening: adaptive meshing with moving boundaries. In *7th International Meshing Roundtable*, pages 201–210, Dearborn, Mich., 1998.

[13] X. Y. Li, S. H. Teng, and A. Üngör. Biting spheres in 3d. *Submitted to the 8th International Meshing Roundtable*, 1999.

[14] R. Lohrer. Progress in grid generation via the advancing front technique. *Engineering with Computers*, 12:186–210, 1996.

[15] R. Lohrer and P. Parikh. Three dimensional grid generation by the advancing-front method. *Int. J. Numer. Meth. Fluids*, 8:1135–1149, 1988.

[16] G. L. Miller, D. Talmor, and S.-H. Teng. Optimal good aspect ratio coarsening for unstructured meshes. In *8th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 538–547. ACM-SIAM, 1997.

[17] G. L. Miller, D. Talmor, and S. H. Teng. Data generation for geometric algorithms on non-uniform distributions. *International Journal of Computational Geometry and Applications*, 1998. accepted and to appear.

[18] G. L. Miller, D. Talmor, and S.-H. Teng. Optimal coarsening of unstructured meshes. *Journal of Algorithms*, 1998. invited and accepted to a special issue for SODA 97.

[19] G. L. Miller, D. Talmor, S.-H. Teng, and N. Walkington. A delaunay based numerical method for three dimensions: generation, formulation, and partition. In *Proc. 27^{th} Annu. ACM Sympos. Theory Comput.*, pages 683–692, 1995.

[20] G. L. Miller, D. Talmor, S.-H. Teng, and N. Walkington. On the radius–edge condition in the control volume method. *SIAM J. on Numerical Analysis*, 1998. accepted and to appear.

[21] G. L. Miller, D. Talmor, S.-H. Teng, N. Walkington, and H. Wang. Control volume meshes using sphere packing: generation, refinement, and coarsening. In *5th International Meshing Roundtable*, pages 47–61. Sandia National Laboratories, 1996.

[22] S. A. Mitchell and S. A. Vavasis. Quality mesh generation in three dimensions. In *ACM Symposium on Computational Geometry*, pages 212–221, 1992.

[23] H.Borouchaki P.J.Frey and P.L.George. Delaunay tetrahedralization using an advancing-front approach. In *6th International Meshing Roundtable*, 1997.

[24] J. Ruppert. A new and simple algorithm for quality 2-dimensional mesh generation. In *Third Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 83–92, 1992.

[25] J. R. Shewchuk. Tetrahedral mesh generation by delaunay refinement. In *14th Annual ACM Symposium on Computational Geometry*, pages 86–95, 1998.

[26] K. Shimada. *Physically-based Mesh Generation:Automated Triangulation of surfaces and Volumes via Bubble Packing*. PhD thesis, MIT, Cambridge, 1993.

[27] K. Shimada and D. C. Gossard. Bubble mesh: automated triangular meshing of non-manifold geometry by sphere-packing. In *third Symp. on Solid Modeling and Appl.*, pages 409–419, 1995.

[28] K. Shimada, A. Yamada, and T. Itoh. Anisotropic triangular meshing of parametric surfaces via close packing of ellipsoidal bubbles. In *6th International Meshing Roundtable*, pages 375–390, 1997.

[29] G. Strang and G. J. Fix. *An Analysis of the Finite Element Method*. Prentice-Hall, 1973.

[30] D. Talmor. *Well-Spaced Points for Numerical Methods*. PhD thesis, Carnegie Mellon University, 1997.

[31] S. H. Teng and C. W. Wong. Unstructured mesh generation: Theory, practice, and perspectives. *International Journal of Computational Geometry and Applications*, 1999.

[32] M. A. Yerry and M. S. Shephard. A modified quadtree approach to finite element mesh generation. In *IEEE Computer Graphics and Applications*, volume 3, pages 39–46, 1983.