Linear-Size Meshes*

Gary L. Miller† Todd Phillips‡ Donald Sheehy§

Abstract

Most modern meshing algorithms produce asymptotically optimal size output. However, the size of the optimal mesh may not be bounded by any function of n. In this paper, we introduce *well-paced* point sets and prove that these will produce linear size outputs when meshed with any "size-optimal" meshing algorithm. This work generalizes all previous work on the linear cost of balancing quadtrees. We also present an algorithm that uses well-paced points to produce a linear size Delaunay mesh of a point set in $\mathbb{R}^d$.

1 Introduction

The goal of meshing is to discretize a geometric domain. Such discretizations are necessary for a variety of applications, notably including the finite element method.

We consider the case of meshing a point set $P \subset \mathbb{R}^d$ of size n to produce a "quality" simplicial complex, where quality is a technical condition we describe in Section 3. The vertices of the output include the input set P and some number of Steiner points added to achieve quality.

The most powerful theoretical tool for analyzing meshing algorithms comes from Ruppert[9] in his work on Delaunay refinement meshing in $\mathbb{R}^2$. Define $\mathrm{lfs}_P(x)$, the *local feature size* at a point x in the domain, to be the distance to the second nearest point in P. The following theorem is the standard generalization of Ruppert's results to d-dimensions.

Theorem 1 *The number of number of vertices in any optimal-size quality mesh of a domain* $\Omega \subseteq \mathbb{R}^d$ *is* $\Theta(\int_{x\in\Omega} \frac{1}{\mathrm{lfs}(x)^d} dx)$.

Several known meshing algorithms ([9, 2, 10, 3] to mention a few) witness to the upper bound in Theorem 1, terminating with meshes that are $O(1)$-competitive.

There is a marked absence of n in Theorem 1. In fact, the size of the optimal mesh may not be bounded by any function of n. A guarantee of optimal size output is not a guarantee that output will even be polynomial size. This leads one to believe that perhaps optimality should not be the last word in mesh size analysis.

In this paper, we attack this problem from two directions. First, we show a general condition on point sets for which it is possible to show that the optimal mesh will have linear size. Second, we present an algorithm called LinearMesh, that produces a linear size mesh of any point set by weakening the quality guarantees in regions where the Ruppert lower bound requires a superlinear number of Steiner points.

2 Previous Work

Previous work in simplicial meshing can be roughly divided into two categories, structured and unstructured as typified by quadtree methods and Delaunay refinement respectively. Structured meshing algorithms are characterized by three main properties: a fixed coordinate system, strict control over where Steiner points are added, and predefined mesh templates for filling boxes or other common shapes. These three properties simplify the implementation and analysis of the algorithms, but at a cost. Unstructured meshing algorithms produce meshes that are independent of the coordinate system, allow complete freedom for Steiner point insertion, and have well-defined topology without predefined templates.

Much recent work has sought to bridge the gap between these two paradigms[4, 3, 5].

In this paper, we present two unstructured generalizations of previous results from quadtree meshing. The cost of balancing a quadtree with n boxes is $O(n)$[8]. We present a general class of point sets, of which quadtree vertices are a special case, and for which any quality meshing algorithm will only use a linear number of Steiner points. The algorithm we present in this paper is an unstructured generalization of a quadtree algorithm of Bern et al [2]. The key to both results is a powerful new analytic technique that allows us to analyze optimal mesh size in terms of n without relying on the fixed structure of the quadtree.

3 Well-paced and well-spaced points

We present some standard definitions and introduce two new ones, namely *θ-medial* points and *θ-well-paced ex-*

*This work was partially supported by the National Science Foundation under grant number CCF-0635257.

†Department of Computer Science, Carnegie Mellon University, glmiller@cs.cmu.edu

‡Department of Computer Science, Carnegie Mellon University, tp517@cs.cmu.edu

§Department of Computer Science, Carnegie Mellon University, dsheehy@cs.cmu.edu

tensions of point sets.

Let $\Omega \subseteq \mathbb{R}^d$ be some compact, convex set representing the domain to be meshed. For a point p, denote the distance of p to its nearest neighbor by r_p. The *gap ball* of a point p is the largest empty ball with center in Ω and p on its surface. Let R_p denote the radius of the gap ball of p. A point set P is *ρ-well-spaced* if for every point $p \in P$, $\frac{R_p}{r_p} \leq \rho$.

Say that a point x is *θ-medial* with respect to a point set P if $NN_P(x) \geq \theta \ \mathrm{lfs}_P(x)$ where $NN_P(x)$ denotes the nearest neighbor of x in P. A 1-medial point is equidistant from both nearest neighbors and is thus on the medial axis of P. In general, medial points are near the medial axis where *near* is defined in terms of θ.

An ordered point set $p_1, \ldots, p_n \subset \Omega$ is a *θ-well-paced extension* of a set Q if each p_i is θ-medial with respect to $\{p_1 \ldots, p_{i-1}\} \cup Q$. We call the ordered points p_i *θ-well-paced*. The term "well-paced" is motivated by the way large changes in the local feature are paced out over the sequence of insertions, because any one insertion can only change lfs by a constant factor depending on θ.

In unstructured Delaunay meshing, the topology of the mesh is determined by the location of the points so it is customary to speak of a mesh and a set of points interchangeably. Moreover, properties of point sets have a natural correlation with properties of meshes. For this paper, we will say that a *quality* mesh is simply the triangulation of a well-spaced point set. Usually, mesh quality is defined in terms of some properties of the mesh triangles, but for all of the meshing algorithms we are considering, quality and well-spaced are equivalent notions.

4 Two examples of well-paced point set extensions

The two classic methods for adding points to a mesh are splitting quadtree cells and adding circumcenters of Delaunay triangles. Both methods fit neatly in the theory of well-paced points. In fact, both methods produce 1-well-paced extensions of a constant sized mesh. In Section 5, we show how to bound the cost of meshing such point sets.

Consider the following very simple quadtree construction. Start with a single box. At each step, pick some box and split it in half along each axis. When viewed as a cell-complex, the vertices or 0-faces in this construction form a 1-well-paced extension of the vertices of the initial box. Each time we split a box, we split the corresponding faces in increasing order by dimension. The edge bisectors are 1-medial because the two nearest neighbors must be the endpoints. Likewise, the points splitting higher dimensional faces have nearest neighbors on each lower dimensional face and all are equidistant, so every insertion is 1-medial.

Our second example of a well-paced point set is the case of circumcenter meshes. Start with some Delaunay triangulation of a point set $Q \subset R^d$. At each step, pick a Delaunay triangle and add its circumcenter. The $d+1$ nearest neighbors of a circumcenter at the time of insertion are all the same distance away (they are on the circumsphere), so the circumcenter is 1-medial. Thus, any sequence of circumcenter insertions forms a 1-well-paced extension of Q.

5 The cost of going from well-paced to well-spaced

Running a meshing algorithm on a point set P will add Steiner points until the resulting set P' is well-spaced. The *cost of cleaning* a point set P, denoted by $Cost(P)$ is defined as $|P'|$, the size of the well-spaced output. In this section, we prove that adding an n point well-paced extension of Q will only increase the cost of cleaning by $O(n)$. In Section 4, we showed that inserting points in a quadtree a special case of well-paced points. Balancing a quadtree involves splitting cells to achieve well-spaced vertices. Thus, the result of this section generalizes previous work on the linear cost of balancing quad trees [8].

In particular, if the cost of cleaning Q is $O(1)$ (as is the typical case when Q is a well-spaced bounding box,) then the output mesh will have size $O(n)$.

Theorem 2 *If P is a θ-well-paced extension of Q, then* $Cost(Q \cup P) = O(Cost(Q) + |P|)$.

Proof. The proof will be by induction on $n = |P|$. Let $\mathrm{lfs}^{(i)}$ be the local feature size function induced by $Q \cup \{p_1, \ldots, p_i\}$. Let $\Psi_i = c_1 \int_{x\in\Omega} \frac{1}{\mathrm{lfs}^{(i)}(x)^d} dx$, where c_1 is the constant from the upper bound in Theorem 1. In general, c_1 will depend on the particular meshing algorithm used. Theorem 1 says that $Cost(Q \cup \{p_1, \ldots, p_i\}) \leq \Psi_i$ and $\Psi_0 = O(Cost(Q))$, the base of our induction.

By induction, we assume $\Psi_{n-1} \leq Cost(Q) + c_2(n-1)$ for some constant c_2. It will suffice to show that $\Psi_n - \Psi_{n-1} < c_2$. We can split the Ruppert sizing integral as follows.

$$\Psi_n = c_1 \int\limits_{x\in\Omega} \frac{1}{\mathrm{lfs}^{(n)}(x)^d} dx \tag{1}$$

$$\leq \Psi_{n-1} + c_1 \int\limits_{x\in U} \frac{1}{\mathrm{lfs}^{(n)}(x)^d} - \frac{1}{\mathrm{lfs}^{(n-1)}(x)^d} dx \tag{2}$$

where $U \subseteq \Omega$ is the set of all points for which the local feature size was changed by the insertion of p_n. Let $R = r_{p_n}$. The following two inequalities hold for all $x \in U$, the first is trivial and the second follows from the definition of well-paced points.

$$\mathrm{lfs}^{(n)}(x) \geq |p_n - x|, \text{ and} \tag{3}$$

$$\mathrm{lfs}^{(n-1)}(x) \leq |p_n - x| + \frac{R}{\theta}. \tag{4}$$

We use these inequalities to compute the integral above using spherical coordinates. Since the integrand is positive everywhere, we can upper bound the integral by integrating over all of $\mathbb{R}^d$ instead of just U:

$$\Psi_n - \Psi_{n-1} \leq c_1 \int\limits_{x \in U} \frac{1}{(|x|)^d} - \frac{1}{(|x| + \frac{R}{\theta})^d} dV, \quad (5)$$

$$\leq c_1 \int_0^\infty \int\limits_{S_r} \left(\frac{1}{r^d} - \frac{1}{(r + \frac{R}{\theta})^d} \right) dA dr, \quad (6)$$

$$\leq c_1 s_d \int_0^\infty \left(\frac{1}{r^d} - \frac{1}{(r + \frac{R}{\theta})^d} \right) r^{d-1} dr, \quad (7)$$

where S_r is the sphere of radius r and s_d is the surface area of the unit d-sphere. In the ball of radius $\frac{R}{2}$ around p_n the lfs is at least $\frac{R}{2}$, so the contribution of this region is to Ψ_n at most some constant c_3.

$$\Psi_n - \Psi_{n-1} \leq c_3 + c_1 s_d \int_{\frac{R}{2}}^\infty \left(\frac{1}{r^d} - \frac{1}{(r + \frac{R}{\theta})^d} \right) r^{d-1} dr \quad (8)$$

By the change variable $yR/\theta = r$ and simplifying we get:

$$\Psi_n - \Psi_{n-1} \leq c_3 + c_1 s_d \int_{\frac{\theta}{2}}^\infty \left(\frac{(y+1)^d - y^d}{y(y+1)^d} \right) dy \quad (9)$$

$$\leq c_3 + c_1 s_d \sum_{i=0}^{d-1} \binom{d}{i} \int_{\frac{\theta}{2}}^\infty \frac{y^i}{y^{d+1}} dy \quad (10)$$

$$\leq c_3 + c_1 s_d d^2 \binom{d}{d/2} (2/\theta)^d \quad (11)$$

The last inequality follows from the fact that each integral is bounded by $d(2/\theta)^d$. Choosing c_2 larger than this constant completes the proof. □

One interpretation of this theorem is that the amortized increase in the cost of cleaning a point set is constant if you add a θ-medial point.

Corollary 3 *If Q is a well-spaced point set and P is a well-paced extension then $Cost(Q \cup P) = O(|Q| + |P|)$.*

Proof. Follows from the above theorem and the linear cost of cleaning points that are already well-spaced. □

6 Algorithm LinearMesh

Psuedocode for the algorithm LinearMesh is shown in Figure 1. The algorithm takes a set of points I as input and outputs a superset of points L, such that the Delaunay triangulation of L is linear in the size of I.

The first call is to a simple routine BoundingBox that will calculate the diameter of I and place a bounding box around I that is a constant factor β larger in size. The bounding box is constructed with a constant number of vertices N_β. The bounding box controls the area where new points will be added, and controls interaction with recursive sub-calls.

The `WHILE` loop then selects a subset of I and adds it to P. The selection of θ-medial points makes sure that P is well-paced. The call to DelaunayRefine can invoke any Delaunay refinement algorithm that will accept as input points in a bounding box and produce an optimal quality mesh S. Acceptable algorithms are prevalent in the literature [2, 9, 10, 3].

The `FOREACH` loop now partitions the remaining points from $I - P$ into clusters I_v around the vertices $v \in S$. Each cluster is then meshed recursively, and these points L_v (along with S) are all added together to form L. This is illustrated in Figure 2(d).

6.1 Linearity of LinearMesh

To show that the output of LinearMesh is only linear in the input I, we must first show that it generates only a linear number of vertices.

First, the well-paced pointset P has size $N_\beta + |I \cap P|$. Considering S, by Theorem 2, $|S| \in O(|P|) \in O(|I \cap P|)$. Now consider the recursive partition. Inductively, the cluster submeshes have size $L_v \in O(I_v)$, so their union has total size $O(|I - P|)$. It follows that the final answer has $|L| \in O(|I|)$.

Recall that in three or more dimensions, the number of edges in $Del(L)$ may be $\Omega(|L|^{\lceil d/2 \rceil})$, so we must argue that $|Del(L)|$ is only linear in $|L|$. We claim that the degree of every vertex in $Del(L)$ will be constant. We make use of an established theorem about well-spaced points [7, 3]:

Theorem 4 *If S is a well-spaced point set, then every vertex of $Del(S)$ has constant degree, so that $|Del(S)| \in O(|S|)$ with constants depending on dimension and quality of the well-spacing.*

This theorem guarantees us that every vertex in $Del(S)$ has constant degree (see Figure 2(e)). Next, L was constructed by substituting each L_v for its parent v. Define G as a contraction of $Del(L)$ obtained by collapsing all the vertices of L_v into v (for every choice of v). Intuitively, this contracted graph G should be almost exactly the same as $Del(S)$. (Contracting the graph in Figure 2(f) happens to give exactly Figure 2(e)).

G is not precisely $Del(S)$; its edge structure could be slightly perturbed. The standard "gap-ratio" analysis technique employed in [7, 10, 3] can show that G still has constant degree.

Furthermore, it is straightforward to see the degree of a vertex in $Del(L)$ is at most N_β times as large as its degree in G. Thus $Del(L)$ has constant degree, and so $|Del(L)| \in O(|L|)$.

LinearMesh(I)
  RETURN {} IF $I == \{\}$
  Initialize P =BoundingBox(I)
  WHILE $\exists p \in I - P$ such that p is θ-medial w.r.t. P
    Add p to P
  ENDWHILE
  Initialize $L = S$ =DelaunayRefine(P)
  FOREACH $v \in S$
    $I_v = \{p \in I$ such that $NN_S(p) = v\}$
    $L_v =$ LinearMesh(I_v)
    Add L_v to L
  ENDFOR
  RETURN L

Figure 1: Pseudocode for LinearMesh.

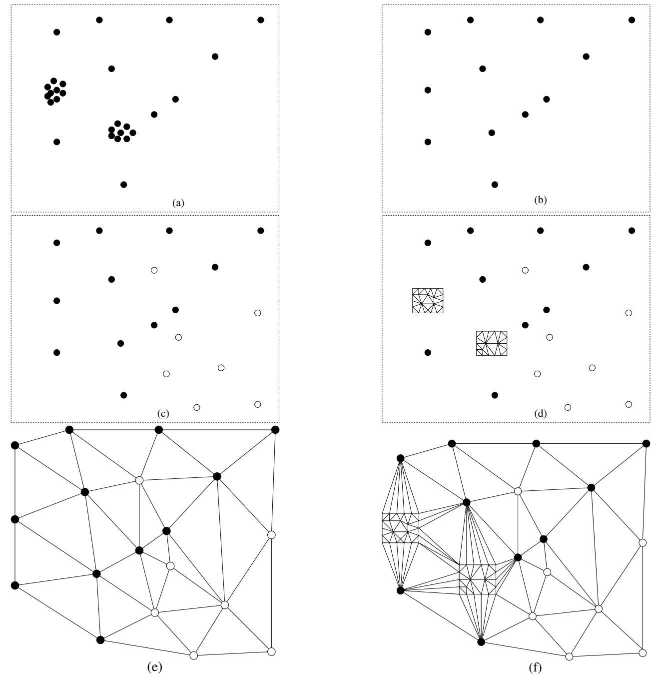

Figure 2: Stages of LinearMesh: (a) Input points I. (b) A well-paced subset P is constructed (bounding box not shown). (c) A well-spaced superset S is constructed. (d) L is S augmented with recursive clusters L_v. (e) The quality mesh of S. (f) Final mesh of L where degree could be larger by a factor of N_β.

6.2 Output Quality

Besides linearity, we can also guarantee that simplices in the the output triangulation have bounded circumradius to longest edge ratio (R/E). In two dimensions, this is equivalent to bounding the largest mesh angle away from π, which guarantees the quality of the mesh with regards to interpolation [1]. One might hope that in higher dimensions, this condition would guarantee no large dihedral angles, and indeed it does come close, with only the unfortunate exception of allowing sliver tetrahedra. We could imagine a variant involving techniques from [6] that might achieve this guarantee while adding only linearly many new vertices.

7 Conclusions

We have presented a powerful new tool for analyzing quality simplicial meshing algorithms.

In addition our algorithm, LinearMesh, is a fully unstructured method for producing linear size meshes of point sets in $\mathbb{R}^d$. Two potential extensions to this work are to conform to more complex inputs and to use common post processing procedures to improve quality guarantees such as in [6].

It is our hope that this work will fuel new research into optimal (or mostly optimal) quality meshing algorithms with polynomial size output.

References

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

[2] M. Bern, D. Eppstein, and J. R. Gilbert. Provably Good Mesh Generation. *Journal of Computer and System Sciences*, 48(3):384–409, June 1994.

[3] B. Hudson, G. Miller, and T. Phillips. Sparse Voronoi Refinement. In *Proceedings of the 15th International Meshing Roundtable*, pages 339–356, Birmingham, Alabama, 2006. Long version available as Carnegie Mellon University Technical Report CMU-CS-06-132.

[4] F. Labelle. Sliver removal by lattice refinement. In N. Amenta and O. Cheong, editors, *Symposium on Computational Geometry*, pages 347–356. ACM, 2006.

[5] F. Labelle and J. R. Shewchuk. Isosurface stuffing: fast tetrahedral meshes with good dihedral angles. *ACM Trans. Graph.*, 26(3):57, 2007.

[6] X.-Y. Li and S.-H. Teng. Generating well-shaped Delaunay meshed in 3D. In *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, pages 28–37. ACM Press, 2001.

[7] G. L. Miller, D. Talmor, S.-H. Teng, and N. Walkington. On the radius–edge condition in the control volume method. *SIAM J. Numer. Anal.*, 36(6):1690–1708, 1999.

[8] D. Moore. The cost of balancing generalized quadtrees. In *SMA '95: Proceedings of the Third Symposium on Solid Modeling and Applications*, pages 305–312, 1995.

[9] J. 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).

[10] A. Üngör. Off-centers: A new type of steiner points for computing size-optimal guaranteed-quality delaunay triangulations. In *Proceedings of LATIN*, 2004.