Point Placement for Delaunay Triangulation of Polygonal Domains

Lee R. Nackman* Vijay Srinivasan*

1 Introduction

In some applications of triangulation, such as finite element mesh generation, the aim is to triangulate a given domain, not just a set of point sites. One approach to meeting this requirement, while maintaining the desirable properties of Delaunay triangulation, has been to enforce the empty circumcircle property of Delaunay triangulation, subject to the additional constraint that the edges of a polygon (or, more generally, a graph) be covered by edges of the triangulation. This leads to the *constrained Delaunay triangulation* (also called the *generalized Delaunay triangulation*), in which the usual empty circumcircle property that defines Delaunay triangulation is modified to require that the interior of the circumcircle of every triangle be devoid of points that are visible from all three of the triangle's vertices [FFP85, LL86, Che89, BEG90].

In finite element mesh generation, it is usually necessary to include additional points besides the vertices of the domain boundary. This leads us to ask whether it is possible to triangulate a domain by introducing additional points in such a manner that the Delaunay triangulation of the points includes the edges of the domain boundary. In other words, instead of modifying the definition of Delaunay triangulation to constrain the triangulation of a given set of points to cover the edges of a polygon, we ask whether we can add additional points to achieve the same aim.

We present an algorithm that given a polygonal domain with N vertices on its boundary, adds K additional points in $O(N \log N + K)$ time such that the domain boundary is covered by the edges of the Delaunay triangulation of the $N + K$ points. Furthermore, K is the minimum number of additional points such that there exists a circle, passing through the endpoints of each boundary edge segment, that does not contain in its interior any other part of the domain boundary.

2 Problem Statement

A *multiply-connected polygonal domain* (hereafter, *domain*) D is a region of $\Re^2$ whose boundary, ∂D, consists of one or more simple polygons. A set of points P is *admissible* only if the edges of the Delaunay triangulation of P covers the edges of ∂D. A circle is *point-free* if none of the given points is contained in the circle's interior. A triangulation of $n \geq 2$ points is Delaunay if and only if every edge has a point-free circle passing through its endpoints [GS85]. Starting with P being the vertices of ∂D, a set of points can be constructed by adding points to P until there exists a point-free circle passing through consecutive points on the edges of ∂D. Suppose the points are added sequentially and the resulting point-free circles on edge e_1 are allowed to overlap another edge, e_2. Then, adding a point later on e_2 might destroy the "point-freeness" of one of the point-free circles of e_1. This would then require another pass over the edges to add more points, and this process would be repeated until no more points are added. Although it can be argued that this process will eventually terminate, a very large number of points can be added and the ultimate number added depends on the order and the position in which the points are added. Therefore, we seek an admissible set of points having the additional property that there exists an *edge-free* circle passing through each pair of adjacent points

*Manufacturing Research Department, IBM Research Division, Thomas J. Watson Research Center, P. O. Box 218, Yorktown Heights, NY 10598

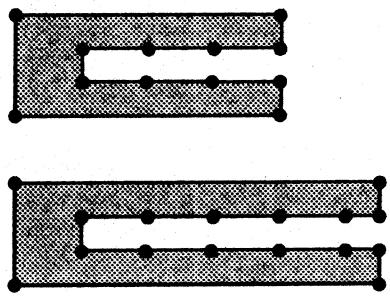

Figure 1: Each example shows a smallest possible admissible set of points having the edge-free circle property. By altering the shape of the input, without altering the size of the input, the output size can be made arbitrarily large.

---

on the edges of ∂D. By "edge-free circle," we mean that the open disk bounded by the circle does not intersect any other edge of ∂D. Finding edge-free circles also brings an additional benefit: it can be seen trivially that after obtaining the admissible set satisfying the edge-free circle criterion, new points can be added anywhere on ∂D without destroying the edge-free circle property.

The problem we solve in this paper is to find a smallest possible admissible set of points having the edge-free circle property. We note at the outset that the best we can hope for is an algorithm with time complexity that is linear in the output size, for as Figure 1 illustrates, the output size can be independent of the input size.

3 Related Work

Boissonnat et al. have sketched two algorithms for finding an admissible set of points P. The analysis presented in [Boi88] claims that only $O(N)$ points are added. Unfortunately, the claim is incorrect, being based on the incorrect implication that if the number of points added is independent of N (which it is), the number of points added is O(1). A similar algorithm (and incorrect analysis) is given in [SP]. Boissonnat's other algorithm [BFBM88] finds an admissible set of points that also has the edge-free circle property. It computes for each edge the minimum distance d to all other non-adjacent boundary entities and then places additional points on the edge so that the distance between consecutive points is no greater than d. This is sufficient to obtain the edge-free circle property, but an excessive number of additional points are unnecessarily placed. Here, too, the claim that only $O(N)$ additional points are added is incorrect.

4 Algorithm Overview

The algorithm described in this paper processes each edge independently. Starting at one endpoint of the edge e, it finds an edge-free circle that passes through the endpoint and cuts off the largest possible portion of e. This is repeated until the entire edge is consumed. In the example of Figure 2, the algorithm finds an edge-free circle passing through $\mathbf{v}_1$ and $\mathbf{v}_2$ and therefore does not add any points to that edge. However, in processing the next edge, it does not find an edge-free circle passing through $\mathbf{v}_2$ and $\mathbf{v}_3$. Therefore, it adds the point $\mathbf{p}$ to that edge and proceeds further.

5 Maximal Circles and Medial Involutes

A *maximal disk* with respect to a domain D is an open disk that is contained in D but not contained in any other open disk contained in D. The circle that bounds a maximal disk is called a *maximal circle*. A maximal circle with respect to D must intersect ∂D, for if it did not, it would be possible

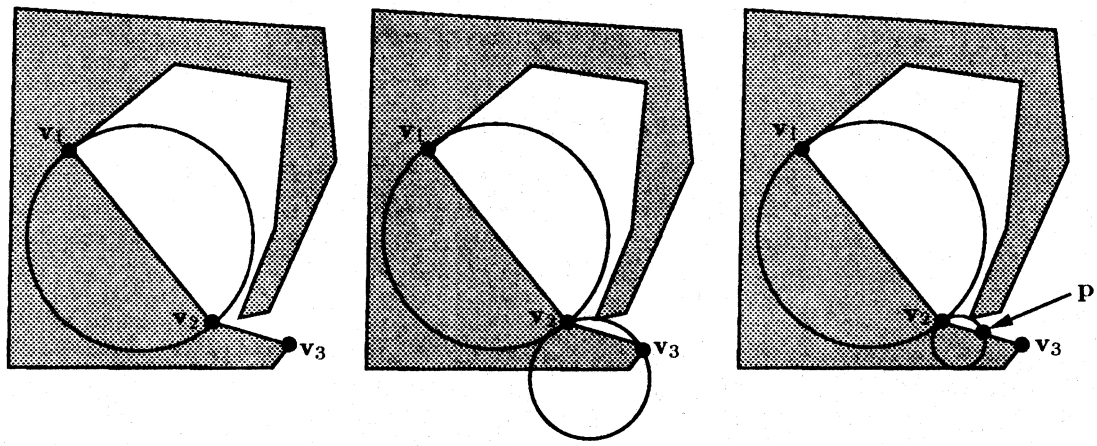

Figure 2: Algorithm overview

to construct a concentric circle and associated open disk in D that contains the maximal circle. The points in the intersection are called the *touching points* of the maximal circle.

Maximal disks can also be defined with respect to two sets using the notion of a bisector. The *bisector* $B(S_1, S_2)$ of two sets S_1 and S_2 is the set of points equidistant from S_1 and S_2. Let $\mathbf{q}$ be a point on $B(S_1, S_2)$. A *maximal disk* $\mathcal{D}$ with respect to S_1 and S_2 is the open disk centered at $\mathbf{q}$ with radius $r = d(\mathbf{q}, S_1) = d(\mathbf{q}, S_2)$. The boundary of $\mathcal{D}$ is called the maximal circle.

Let $\mathbf{p} \in \partial D$. The *medial involute* [Boo79] of $\mathbf{p}$ with respect to D, denoted $\mathrm{MI}(\mathbf{p}, D)$, is the set of points $\mathbf{q} \in \partial D$ such that there exists a maximal circle with respect to D having $\mathbf{p}$ and $\mathbf{q}$ as touching points. The medial involute of a set $S \subset \partial D$ with respect to D, denoted $\mathrm{MI}(S, D)$, is $\bigcup_{\mathbf{p} \in S} \mathrm{MI}(\mathbf{p}, D)$. The *interior medial involute* of a set S with respect to a domain D, denoted $\mathrm{IMI}(S, D)$, is $\mathrm{MI}(S, D)$ and the *exterior medial involute* of S with respect to D, denoted $\mathrm{EMI}(S, D)$, is $\mathrm{MI}(S, \overline{D})$.

6 Algorithm

Algorithm PlacePoints:

Input: directed edge e from vertex $\mathbf{v}_1$ to vertex $\mathbf{v}_2$, $\mathrm{IMI}(e, D)$, and $\mathrm{EMI}(e, D)$
Output: additional points $\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_{K_e}$ on e

1. Compute the bisector $B(\mathrm{IMI}(e, D), \mathrm{EMI}(e, D))$ and label its segments b_i, $i = 1 \ldots m_e$, consecutively, where m_e is the number of segments.
2. Find the maximal circle that passes through $\mathbf{v}_1$ and "cuts off" the largest possible portion of e:
   (a) Find the last member b of $b_1, b_2, \ldots, b_{m_e}$ that is also a bisector of $\mathbf{v}_1$ and some other boundary entity. The desired maximal circle is the circle passing through $\mathbf{v}_1$ and having the rightmost endpoint $\mathbf{q}_0$ of b as center.
   (b) Let $\mathbf{p}_1$ be the other intersection of this circle with e and set $i = 1$. Set b to the bisector segment that is incident from the right at $\mathbf{q}_0$.
3. **do while** $\mathbf{p}_i \neq \mathbf{v}_2$
   Find the maximal circle that first intersects e at $\mathbf{p}_i$:
   (a) Let ϵ_1 and ϵ_2 be the two boundary vertices or edges such that $b \subset B(\epsilon_1, \epsilon_2)$.
   (b) Let β be the bisector between $\mathbf{p}_i$ and either ϵ_1 or ϵ_2.
   (c) If $\mathbf{q}_i = b \bigcap \beta$ is not empty, then

i. The desired maximal circle is centered at $\mathbf{q}_i$ and passes through $\mathbf{p}_i$.

ii. Set $\mathbf{p}_{i+1}$ to the other intersection of this circle with e. Set b to the bisector segment that is incident from the right at $\mathbf{q}_i$.

iii. $i = i + 1$.

Else set b to the next bisector segment.

To place points on all edges in ∂D, the algorithm is executed independently on each edge. This can be shown to take $\Sigma_{e \in \partial D} O(K_e + N_e \log N_e)$ time, which is $O(K + N \log N)$, where $K = \Sigma_{e \in \partial D} K_e$ is the output size. It can be shown that Algorithm PlacePoints places the smallest possible number of points while maintaining the edge-free circle property.

7 Summary

We have presented an algorithm that takes $O(N \log N + K)$ time to place the smallest possible number K of additional points on the boundary of a polygonal domain (having N edges) so that each pair of consecutive points on the domain boundary has passing through it an edge-free circle. Finding the minimum number of additional points to obtain an admissible set of points to Delaunay triangulate a polygonal domain remains an open problem. Our algorithms solve a special case of this problem by constraining the circumcircles of Delaunay triangles to contain at most one connected portion of an edge on the boundary of the domain.

References

[BEG90] M. Bern, D. Eppstein, and J. Gilbert. Provably good mesh generation. In *Proc. 31st Annual Symposium on the Foundations of Computer Science*. IEEE Computer Society, October 1990.

[BFBM88] J.D. Boissonnat, O.D. Faugeras, and E. Le Bras-Mehlman. Representing stereo data with the Delaunay triangulation. Technical Report 788, INRIA, Domaine de Voluceau Rocquencourt, B.P. 105, 78153 Le Chesnay Cedex, France, February 1988.

[Boi88] J.D. Boissonnat. Shape reconstruction from planar cross sections. *Computer Vision, Graphics, and Image Processing*, 44(1):1–29, October 1988.

[Boo79] F.L. Bookstein. The line-skeleton. *Computer Graphics and Image Processing*, 11:123–137, October 1979.

[Che89] L.P. Chew. Constrained Delaunay triangulations. *Algorithmica*, 4:97–108, 1989.

[FFP85] L. De Floriani, B. Falcidieno, and C. Pienovi. Delaunay-based representation of surfaces defined over arbitrarily shaped domains. *Computer Vision, Graphics, and Image Processing*, 32:127–140, 1985.

[GS85] L. Guibas and J. Stolfi. Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams. *ACM Trans. Graphics*, 4(2):74–123, 1985.

[LL86] D. T. Lee and A. K. Lin. Generalized Delaunay triangulation for planar graphs. *Discrete and Computational Geometry*, 1:201–217, 1986.

[SP] N. Sapidis and R. Perucchio. Delaunay triangulation of arbitrarily shaped planar domains. Preprint, Mechanical Engineering Department, University of Rochester, to appear in *Computer Aided Geometric Design*.