Discrete Comput Geom 1:201–217 (1986)

Discrete & Computational Geometry
© 1986 Springer-Verlag New York Inc.

Generalized Delaunay Triangulation for Planar Graphs*

D. T. Lee and A. K. Lin

Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60201, USA

Abstract. We introduce the notion of generalized Delaunay triangulation of a planar straight-line graph $G = (V, E)$ in the Euclidean plane and present some characterizations of the triangulation. It is shown that the generalized Delaunay triangulation has the property that the minimum angle of the triangles in the triangulation is maximum among all possible triangulations of the graph. A general algorithm that runs in $O(|V|^2)$ time for computing the generalized Delaunay triangulation is presented. When the underlying graph is a simple polygon, a divide-and-conquer algorithm based on the polygon cutting theorem of Chazelle is given that runs in $O(|V| \log |V|)$ time.

1. Introduction

A triangulation of a set of points is a straight-line maximally connected planar graph $G = (V, E)$, whose vertices are the given set of points and whose edges do not intersect each other except at the endpoints. Each face, except the exterior one, of the graph is a triangle. Triangulations of a set of points in the plane have been extensively studied and have applications in closest point problem [6, 14, 21, 28], finite element method [1]–[3], stress analysis of two-dimensional continua [8], and interpolation [15, 16, 23, 25, 26]. In this paper we shall consider the Delaunay triangulation [23], [28] which has the property that the circumcircle of any triangle does not contain any other point in its interior. It has been shown [23], [28] that the Delaunay triangulation of a set of N points in the plane can be constructed in $O(N \log N)$ time, which is asymptotically optimal by a result

* Supported in part by the National Science Foundation under Grants DCR 8420814 and ECS 8340031.

of Shamos and Hoey [28] who have shown that any triangulation algorithm can also sort. In a sense, a triangulation embeds the given set of points by containing it as its set of vertices. It is natural to see if it can be generalized so as to embed an arbitrary planar straight-line graph (PSLG) which may or may not be connected. Indeed, a triangulation of a simple polygon with N vertices can be found in $O(N \log N)$ time [4], [5], [9]. With slight modifications, the approach given in [9] can be extended to find a triangulation which embeds a PSLG with N vertices in $O(N \log N)$ time. Along this line we shall address the problem of finding a Delaunay-like triangulation which embeds a given PSLG and satisfies a certain property similar to the circumcircle property mentioned earlier. For any given PSLG G, there are a number of possible triangulations $\mathrm{T}(G)$. We are interested in the following two criteria based on which triangulations $\mathrm{T}(G)$ are constructed: (i) the circle criterion as given in Definition 2, and (ii) the maxmin angle criterion, i.e., the minimum measure of angles of all the triangles in a triangulation is maximized. As will be shown later, these two criteria are equivalent and the resulting triangulation is the generalized Delaunay triangulation, to be defined later.

What motivates the consideration of the generalized Delaunay triangulation stems from a problem in terrain interpolation. Given a terrain surface $z = f(x, y)$ for which some of the function values are known at irregularly scattered points, we want to find a triangular faceted surface to approximate on the set of points whose functional values are known. Each triangle of the triangulation will correspond to a triangular faceted plane in three-dimensional space. The collection of triangular faceted planes is then used to approximate the terrain surface. The functional value of a point $p = (x, y)$ is then linearly interpolated by the functional values of the vertices of the triangle in the triangulation which contains p in its interior.

The performance of this interpolation depends heavily on the choice of the triangular grid. Intuitively, the Delaunay triangulation can be viewed as one in which the triangles look more like equilateral triangles and is believed to provide the best triangular grid for this purpose. However, information other than just a set of points is often available, especially in geographical interpolation. For example, the boundary of a lake and the ridge of a mountain range are also located. Then it would be nice if the triangular grid could preserve the additional information as well. This leads us to consider the construction of a "nice" triangulation for a set of points and line segments. The generalized Delaunay triangulation defined below can be viewed as a triangular grid that retains the edges of the original graph and, whenever it is not obstructive to the original graph, tries to capture the flavour of the Delaunay triangulation as much as possible.

This paper is organized as follows. In the next section we give the definition and some characterizations of the generalized Delaunay triangulation (GDT). A quadratic algorithm is given in Section 3 that computes the $\mathrm{GDT}(G)$ of any PSLG G. In Section 4 we consider a special case when the edges in G form a simple polygon and provide an $O(|V| \log |V|)$ algorithm for computing the GDT. Finally, we discuss possible directions for further research.

2. Preliminaries

We first give a formal definition of the generalized Delaunay triangulation of a PSLG $G = (V, E)$ [17] and then derive certain interesting properties of the generalized Delaunay triangulation.

Definition 1. For any PSLG $G = (V, E)$, a triangulation $\mathrm{T}(G)$ of G is a PSLG $G' = (V, E')$, where $E \subseteq E'$, such that no edges can be added without intersecting an existing edge.

Definition 2. For any PSLG $G = (V, E)$ the *generalized Delaunay triangulation* (GDT) of G, denoted by $\mathrm{GDT}(G)$ is a triangulation $\mathrm{T}(G) = (V, E')$ in which the *circumcircle* of each face or triangle $\triangle v_i v_j v_k$, denoted by $O(v_i, v_j, v_k)$ does not contain in its interior any other vertex which is visible from the vertices v_i, v_j, and v_k of the triangle. The edges of the set $E' - E$ are called *Delaunay edges*, and the edges of E are called *sides*. The vertices u and v, $u, v \in V$ are *visible* from each other if the line segment u, v does not intersect an edge of E at an interior point.

Figure 1 shows the GDT of a graph G with Delaunay edges shown in dotted line. Note that for graphs $G = (V, \phi)$, the $\mathrm{GDT}(G)$ becomes the conventional Delaunay triangulation of a set V of points. The following lemma, which relates the numbers of triangles and edges to the number of vertices in V, can be established fairly easily.

Lemma 1. *Given any* PSLG $G = (V, E)$, *any triangulation* $\mathrm{T}(G)$ *has* $2(|V| - 1) - B$ *triangles, and* $3(|V| - 1) - B$ *edges, where* B *is the number of vertices that are on the convex hull of the set* V *of points and* $|V|$ *denotes the cardinality of* V.

We now establish the relationship between the circle criterion and the maxmin angle criterion for the Delaunay triangulation. Given a strictly convex quadrilateral *abcd* so that the four vertices are not cocircular, there are two possible

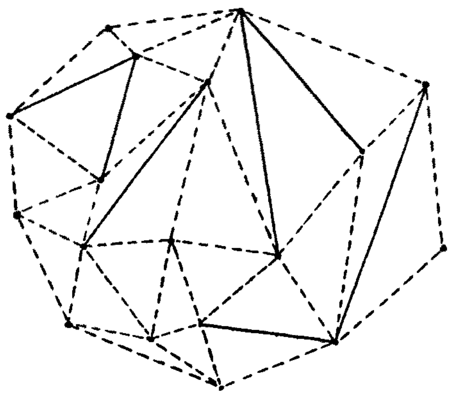


Fig. 1. Generalized Delaunay Triangulation of a PSLG.

triangulations as shown in Fig. 2(a). If the quadrilateral is not convex or one of the diagonals is a side, i.e., an edge of the given graph, then only one triangulation is possible (Fig. 2(b)). If we use the circle criterion, i.e., if the circumcircle of $\triangle abc$ contains vertex d, then vertex d is connected to vertex b, otherwise vertices a and c are connected. The resulting triangulation is the Delaunay triangulation of the quadrilateral. However, if the maxmin angle criterion is used, i.e., if the minimum angle of the two triangles is to be maximized, the resulting triangulation must also satisfy the circle property, as shown below.

Lemma 2. *Let abcd be a convex quadrilateral such that vertex c is outside the circle $O(a, b, d)$. The minimum angle of the triangulation obtained by adding diagonal $\overline{b, d}$ is strictly larger than the minimum angle obtained by adding diagonal $\overline{a, c}$.*

Proof. As shown in Fig. 3 let θ_1 be the smallest angle and let edge $\overline{b, c}$ intersect $O(a, b, d)$ at c'. It is clear that angle $\measuredangle acd < \measuredangle ac'b = \measuredangle adb$. Thus, the minmum angle of the triangles in the triangulation obtained by adding diagonal $\overline{a, c}$ is smaller than the minimum angle in the triangulation obtained by adding diagonal $\overline{b, d}$. The other cases in which the smallest angle is not θ_1 can be shown similarly. □

Indeed, based on this result, Lawson [16] devised a *local optimization procedure* (LOP) to construct the Delaunay triangulation for a set of N points. The LOP works as follows. Suppose e is an internal edge, in contrast to the edges on the convex hull, of a triangulation and Q is the quadrilateral formed by two triangles having e as the common edge. Consider the circumcircle of one of the triangles

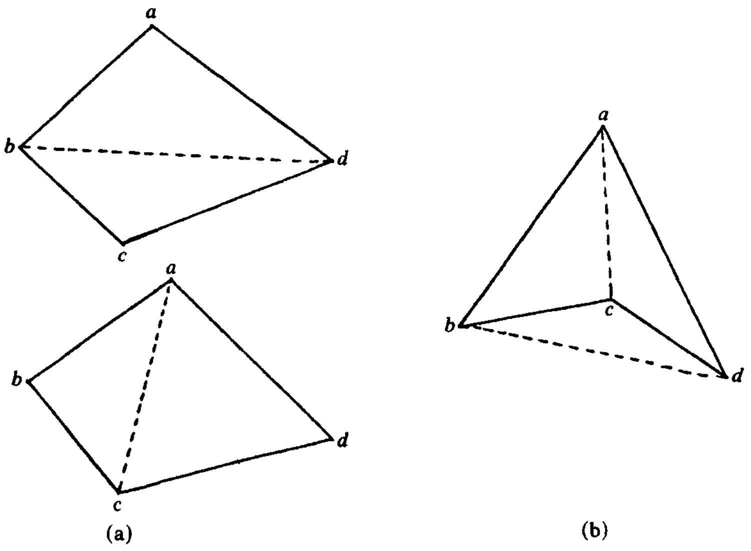


Fig. 2. Triangulation of quadrilateral.

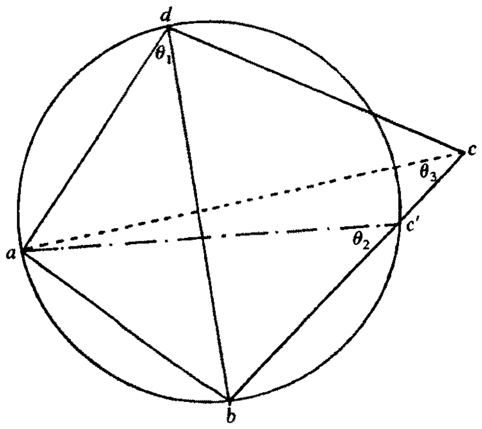


Fig. 3. Illustration for the proof of Lemma 2.

in Q. If the circle contains the other vertex, replace e by the other diagonal of Q, otherwise leave e as it is. An edge of the triangulation is said to be *locally optimal* if the application of the LOP to it would not swap it. The edges on the convex hull are locally optimal by default. Initially the convex hull of the set of points is obtained and an initial triangulation of the convex hull is constructed either by the procedure described below or by other methods. (See, e.g. [22].) The points not on the convex hull are added to the triangulation one at a time. Suppose that the new point p is contained in triangle $\triangle abc$. First p is connected to the vertices of the triangle a, b, and c to form three new triangles. Each of these triangles, in general, will form a quadrilateral with a neighboring triangle. The LOP is then applied to each of these quadrilaterals. Whenever a new triangle is created by swapping the two diagonals of the quadrilateral on which the LOP is applied, the same LOP will be applied to the newly created quadrilateral. In this manner the LOP is repeatedly applied until all the edges in the triangulation are locally optimal. It has been shown [16], [17] that this process terminates and the resultant triangulation after all N points have been added is the Delaunay triangulation of the set of points. The overall time required is $O(N^2)$, since for each point one needs to spend linear time for locating the triangle containing the point and for applying the LOP to the edges of the triangulation.

Note that the above LOP can be applied to any triangulation as we elaborate below. Consider any triangulation $T(G)$ of a PSLG $G = (V, E)$. Assume by default that the edges in E are *locally optimal*. We examine first the effect of each application of LOP to an internal edge of $T(G)$. Let N_t denote the number of triangles in $T(G)$. Recall that the number of triangles in any triangulation of G is a constant when the graph G is given (Lemma 1). For each triangulation $T(G)$ we define a characteristic vector C_T with N_t components, each of which corresponds to a triangle and is the measure of the minimum angle of the triangle. These values are sorted in nondecreasing order. Given triangulations $T(G)$ and $T'(G)$, we define $T(G) < T'(G)$ if and only if the associated characteristic vector of T, C_T, is lexicographically less than $C_{T'}$.

Lemma 3. *Given a triangulation* T(G), *if an application of the* LOP *to an edge* e *results in a swapping of the edge with another edge* e' *and thus producing a new triangulation* T$'(G)$, *then* T$(G)<$T$'(G)$.

Proof. Let the two triangles of T sharing edge e have the minimum angles appear as two components of C_T, say C_{T_j} and C_{T_k}, $j<k$. Thus $C_{T_j} \leq C_{T_k}$. Since a swap was made, from Lemma 2 the smaller of the two smallest angles of the two new triangles resulting from the swap is strictly greater than C_{T_j}. Thus, it follows that C_T must be lexicographically less than $C_{T'}$ and hence T$(G)<$T$'(G)$. □

Lemma 4. *The edges of a triangulation* T(G) *of a* PSLG $G=(V, E)$ *are locally optimal if and only if each triangle of* T(G) *satisfies the circle property, i.e., circumcircle of any triangle* $\triangle abc$ *of* T(G) *does not contain in its interior any vertex of* V *visible from all three vertices* a, b, *and* c.

Proof. Suppose that all triangles of T(G) satisfy the circle property. Since by definition of local optimality, we need only to consider internal edges that are not in E. Consider $\triangle abc$ such that $\overline{b, c}$ is an internal edge and is shared by $\triangle bcd$. If both $\triangle abc$ and $\triangle bcd$ satisfy the circle property, i.e., vertex $d \notin O(a, b, c)$ and vertex $a \notin O(b, c, d)$, application of LOP to $\overline{b, c}$ will not swap it. Thus, all edges are locally optimal. To show the converse suppose that all edges are locally optimal and that the circumcircle K of $\triangle abc$ contains a point p visible from a, b, and c. Let δ be the distance from p to its nearest edge, say $\overline{a, c}$ (Fig. 4). Assume that among all triangles of T whose circumcircles contain p as an interior point, none has an edge which is at a distance less than δ from p. Since p is on the opposite side of $\overleftrightarrow{a, c}$ from b, the edge $\overline{a, c}$ must be shared with another triangle $\triangle acq$, and vertex q cannot be interior to K, as this would contradict the hypothesis that $\overline{a, c}$ is locally optimal. The vertex q cannot be in the cross-lined region as shown in Fig. 4, or $\triangle acq$ would contain p in its interior. Thus, one of the edges,

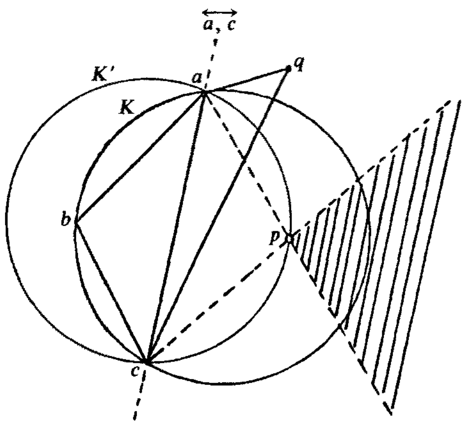


Fig. 4. Illustration for the proof of Lemma 4.

$\overline{a, q}$ and $\overline{c, q}$ is at a distance less than δ from p. If we can show that the circumcircle of $\triangle acq$ also contains p in its interior, then we would have a contradiction that $\triangle abc$ is the triangle with an edge at the smallest distance from p. Note that the circumcircle K' of $\triangle acp$ must contain b in its interior and the quadrilateral $abcp$ lies entirely in the intersection of K and K'. Since the portion of K' that lies on the same side of $\overleftrightarrow{a, c}$ as p, lies totally in K, the vertex q, which is outside of K, must be exterior to K'. Thus, the circumcircle of $\triangle acq$ must contain p in its interior, since for any convex quadrilateral $abcd$ if $O(a, b, c)$ does not contain vertex d, then $O(a, b, d)$ must contain vertex c. □

Theorem 1. *A triangulation* T(G) *of a* PSLG $G = (V, E)$ *is a generalized Delaunay triangulation if and only if its characteristic vector is lexicographically maximum.*

Proof. If the characteristic vector of T(G) is lexicographically maximum, application of LOP to any edge of T(G) will not swap it and hence all the edges must be locally optimal. This implies from Lemma 4 that T(G) is the GDT(G). Conversely, if T(G) is a GDT(G) then it must satisfy the circle property and from Lemma 4 it follows that the edges in a T(G) are locally optimal. Suppose now that there exists a trangulation T′(G) which is lexicographically maximum, but T′(G) ≠ T(G). Since T′(G) is lexicographically maximum, no edges will be swapped when the LOP is applied to any internal edge. Hence all edges in T′(G) must be locally optimal. Since T′(G) ≠ T(G), there must exist a Delaunay edge $\overline{a, d}$ in T′(G) that intersects $\triangle abc$ of T(G). We may assume that $\overline{a, d}$ is the edge closest to vertex c among those that intersect $\triangle abc$. Let the intersection of $\overline{a, d}$ and $\overline{b, c}$ be denoted p (Fig. 5). Note that the edge $\overline{b, c}$ of T(G) intersected by $\overline{a, d}$ must be a Delaunay edge, and vertices b and c and vertices a and d are visible from each other. We shall show that $\overline{a, d}$ cannot be locally optimal and

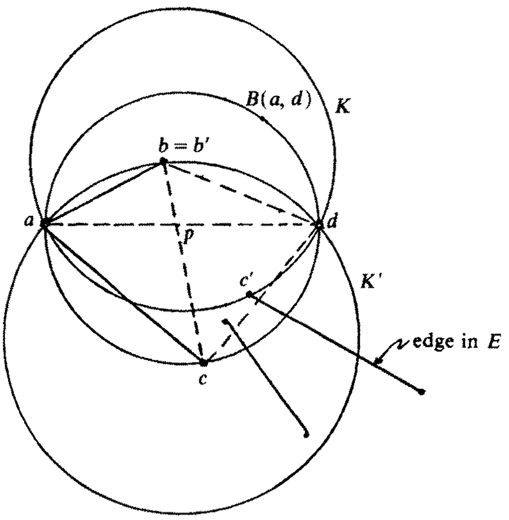


Fig. 5. Both edges $\overline{a, d}$ and $\overline{b, c}$ cannot be locally optimal.

hence cannot be a Delaunay edge as claimed. We claim that there exist two vertices b' and c' in G that lie on different sides of $\overline{a,d}$ such that vertices a, c', d and b' form a convex quadrilateral and they are all visible from one another. If vertex c is visible from d then $c' = c$, otherwise we can find in $\triangle cdp$ a vertex c' which is visible from both a and d, since vertices b and c and vertices a and d are visible and no side of E intersects $\overline{a,d}$ or $\overline{b,c}$. For example, vertex c' can be chosen from the endpoints of sides of E that lie in $\triangle cdp$ so that it is closest to $\overline{a,d}$. Similarly, vertex b' can be found. Consider $K = O(a, c', d)$ and $K' = O(a, b', d)$. It is obvious that $K \cup K'$ contains the circle $B(a, d)$ with $\overline{a,d}$ as the diameter, and $B(a, d)$ contains the quadrilateral $ac'db'$. That is, any circle that passes through vertices a and d will either contain vertex c' or vertex b'. Note that $\overline{a,d}$ must be an edge shared by two triangles in $T'(G)$. In fact, one of these two triangles must be $\triangle ac'd$, since other choices of triangles would contain vertex c' in the interior or violate the assumption that $\overline{a,d}$ is the closest edge to vertex c. (Recall that vertex c' is closest to $\overline{a,d}$.) Therefore $\overline{a,d}$ cannot be locally optimal, a contradiction. Thus, $\mathrm{T}(G)$ must be lexicographically maximum. □

Corollary 1. *If no four points of V in a* PSLG $G = (V, E)$ *are cocircular, the generalized Delaunay triangulation* $\mathrm{GDT}(G)$ *is* unique, *and furthermore, it satisfies the maxmin angle property and the smallest angle of the triangles in* $\mathrm{GDT}(G)$ *is maximum among all possible triangulation* $\mathrm{T}(G)$.

3. A General Algorithm

So far we have concentrated on the characterizations of the generalized Delaunay triangulation. Let us now turn our focus on the problem of constructing $\mathrm{GDT}(G)$ of a given PSLG $G = (V, E)$. We shall present below an $O(|V|^2)$ algorithm for computing the $\mathrm{GDT}(G)$ of a given PSLG $G = (V, E)$. The main idea is to compute for each vertex $v \in V$, the Delaunay edges incident with it. We begin by proving the following lemma, on which the algorithm is based.

Lemma 5. *For any* PSLG $G = (V, E)$, *an edge* $\overline{v,t}$ *is a Delaunay edge in* $\mathrm{GDT}(G)$ *if and only if v and t are visible from each other and there exists a circle passing through v and t that does not contain any vertex visible from both v and t.*

Proof. If $\overline{v,t}$ is a Delaunay edge, then v and t must be visible from each other and there exists a triangle $\triangle tuv$ in $\mathrm{GDT}(G)$ with $\overline{v,t}$ as an edge. Since the triangle satisfies the circle property, the claim follows. Conversely, suppose there exists a circle K passing through vertices v and t and K does not contain any vertex visible from both v and t. We now move the center of K along the perpendicular bisector of $\overline{v,t}$ until the first vertex u visible from both v and t is on the circle. Note that the size of the circle changes as its center moves. The circle $O(t, u, v)$ obtained by enlarging K certainly satisfies the circle property, for if it contained a vertex u' visible from all three vertices t, u, and v in the interior, it would not be the first circle found in this manner. Hence $\overline{v,t}$ is a Delaunay edge, and so are edges $\overline{t, u}$ and $\overline{v, u}$. The existence of vertex u can be argued simply by the fact that the set of vertices visible from vertex, say v, must contain at least one vertex other than t and that among the vertices in the set we can always find one which is also visible from t and satisfies the claimed property. □

We now proceed to find for each vertex $v \in V$ the set S_v of vertices visible from v, i.e., $S_v = \{u \mid u \text{ is visible from } v\}$. The graph obtained by connecting v to all $u \in S_v$ for each $u \in V$ is called the *visibility graph* and it has been shown that the visibility graph can be computed in $O(|V|^2)$ time [12]. Once the visibility graph is obtained, we shall eliminate those edges that are not Delaunay edges based on the above lemma.

We note that the vertices in S_v for v are ordered by angles around v when they are computed [12]. We first find $u \in S_v$ such that the edge $\overline{u, v}$ is the shortest. Since the circle K with $\overline{u, v}$ as the diameter is totally contained in the circle of radius $d(u, v)$ and centered at v, no vertex in S_v is interior to K. (Fig. 6(a)). It follows from Lemma 5 that $\overline{u, v}$ must be a Delaunay edge. We then scan the vertices in S_v around vertex v in counterclockwise order starting with the vertex after u. The procedure is similar to the Graham scan [10], [27] for computing the convex hull of the set S_v. We take three consecutive vertices x, y, z at a time, and these three vertices along with vertex v form a quadrilateral $\underline{vxyz}$. At each step we apply the LOP to the edge $\overline{v, y}$ of the quadrilateral, if $\overline{v, y} \notin E$. If $z \in O(v, x, y)$, the vertex y is deleted from S_v, since edge $\overline{v, y}$ is not a Delaunay edge (Fig. 6(b)). Once vertex y is deleted from S_v, we need to backtrack to consider vertices w, x, and z, since deletion of y may make edge $\overline{v, x}$ non-Delaunay. We summarize it as follows.

Procedure Scan (S_v, u)

```
(Comment: Let the vertices be ordered as $u_1 = u, u_2, \ldots, u_k, u_{k+1} = u_1$, where
      $k = |S_v| > 2$.)
(Comment: $\mathrm{SUCC}(u_i) = u_{i+1}$, and $\mathrm{PRED}(u_i) = u_{i-1}$.)
(Comment: Note that $\overline{v, u}$ is a Delaunay edge.)
1. $x \leftarrow u$; $y \leftarrow \mathrm{SUCC}(x)$; $z \leftarrow \mathrm{SUCC}(y)$
2. while $z \neq x$ do
3.    begin if $\overline{v, y} \in E$ or $z \notin O(v, x, y)$
             then $x \leftarrow y$; $y \leftarrow z$; $z \leftarrow \mathrm{SUCC}(y)$
             else if $\overline{v, y} \notin E$
                 then begin delete $y$
                            if $x \neq u$ then $y \leftarrow x$; $x \leftarrow \mathrm{PRED}(y)$
                                   else $y \leftarrow z$; $z \leftarrow \mathrm{SUCC}(y)$
                      end
      end
```

It is easy to see that in a single scan the edges in S_v that remain are Delaunay edges and the total time spent is $O(|S_v|)$. Since $\Sigma_v |S_v| = O(|V|^2)$, we thus have the following.

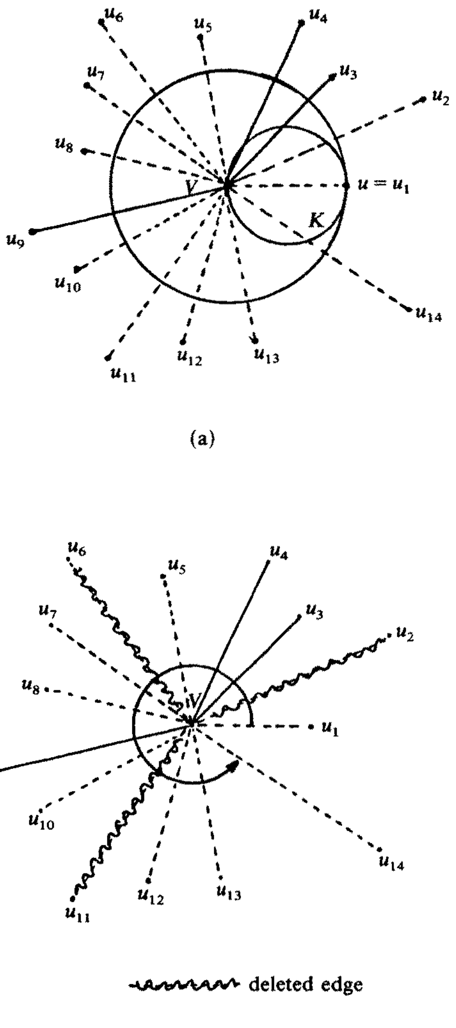


Fig. 6. Computing Delaunay edges incident with vertex v.

Theorem 2. *Given a* PSLG $G = (V, E)$, *the generalized Delaunay triangulation* GDT(G) *can be computed in* $O(|V|^2)$ *time.*

We remark that the above algorithm actually takes $O(|V|^2)$ time and space due to the computation of the visibility graph [12]. But we can easily obtain an $O(|V|^2 \log |V|)$ time and $O(|V|)$ space algorithm by using a straightforward approach (see, e.g., [17]) to computing the visibility set S_v for each vertex $v \in V$ and then apply procedure **Scan** to each set S_v.

4. A Special Case—Simple Polygons

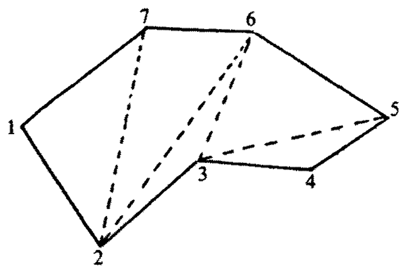
We consider the problem of finding the generalized Delaunay triangulation of a simple polygon, i.e., finding GDT(G) where $G=(V, E)$ is a simple cycle. When the polygon is monotone, Yeung [30] provides an $O(N \log N)$ time algorithm where $N=|V|$, using a straightforward divide-and-conquer scheme. In this section we show that the generalized Delaunay triangulation of an arbitrary simple polygon can also be computed by divide-and-conquer paradigm, except that it rests on a crucial result, the *polygon cutting theorem*, of Chazelle [4].

In [4] Chazelle has shown that given a simple polygon P with N vertices, two vertices a and b can be found in $O(N)$ time such that $\overline{a, b}$ lies entirely in P and each of the two simple subpolygons of P resulting from the "cut" by $\overline{a, b}$, has at least $N/3$ vertices. With this polygon cutting theorem it is possible to triangulate (the interior of) a simple polygon in $O(N \log N)$ time [4]. This triangulation procedure can be used to construct a *decomposition tree* for the simple polygon. The decomposition tree is a binary tree in which the root represents the simple polygon P and the leaves represent triangles in the interior of P. Each node, escept the root, represents a polygon which is a proper subset of its parent and has at least one-third of its vertices. At each node we store the "cut," called *diagonal*, which results in decomposing the corresponding polygon to the smaller polygons represented by its sons. Note that the tree has height $O(\log N)$. A decomposition tree of the polygon shown in Fig. 7(a) is depicted in Fig. 7(b), where the internal nodes represent subpolygons specified as circular lists and associated with each internal node is a diagonal; the leaves correspond to triangles in the decomposition.

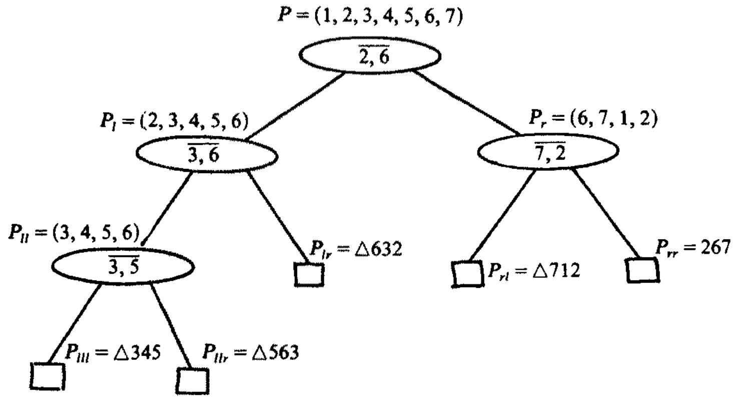
Assume that a simple polygon decomposition tree is constructed by using the polygon cutting theorem. Suppose the simple polygon represented by the root is divided into two subpolygons Q_l and Q_r by the diagonal. We recursively construct the GDT(Q_l) and GDT(Q_r) for the two subpolygons Q_l and Q_r, respectively. If these two triangulations can be merged to form GDT(P) in linear time, the overall time required for the construction of GDT(P) is $O(N \log N)$.

We now describe below the merge process which is similar to the merge algorithm described in [23].

```
Procedure Merge (GDT($Q_l$), GDT($Q_r$), $\overline{v_i, v_j}$)
(Comment: GDT($G$) is maintained as a doubly connected edge list (DCEL) [27]
        so that the edges incident with any vertex can be traversed in either
        counterclockwise or clockwise order.)
(Comment: $\overline{v_i, v_j}$ is the diagonal shared by GDT($Q_l$) and GDT($Q_r$).)
(Comment: It computes, given GDT($Q_l$) and GDT($Q_r$), the GDT($Q$) where $Q$
        is the union of the polygons $Q_l$ and $Q_r$.)
1. Find $\triangle v_i v_j v_l \in$ GDT($Q_l$) and $\triangle v_i v_j v_r \in$ GDT($Q_r$).
2. If there is no vertex in $Q_l$ visible from $v_r$ that lies in $O(v_i, v_j, v_r)$, then $\overline{v_i, v_j}$
   is a Delaunay edge.
   Return (GDT($Q$)), where GDT($Q$) is just the union of GDT($Q_l$) and
   GDT($Q_r$).
```



(a)



(b)

Fig. 7. Decomposition tree of a simple polygon based on polygon cutting theorem.

3. Let $T_{v_r} = \{v_{k_1}, v_{k_2}, \ldots, v_{k_t}\}$ be the set of vertices of Q_l that are visible from v_r and are in $O(v_i, v_j, v_r)$. Let $B(v_t, v_r)$ be the smallest circle that passes through $v_t \in T_{v_r}$ and v_r and that is totally contained in and tangent to $O(v_i, v_j, v_r)$ at v_r.
(*Comment*: No vertices in T_{v_r} are inside $B(v_t, v_r)$.)
(*Comment*: Note that $\overline{v_t, v_r}$ called the *base*, is a Delaunay edge and it intersects $\overline{v_i, v_j}$ and possibly many others in GDT(Q_l), and cuts polygon Q into two subpolygons Q_i and Q_j, where $v_i \in Q_i$ and $v_j \in Q_j$.)
4. Insert the base $\overline{v_t, v_r}$ into the edge lists of v_t and v_r in GDT(Q_l) and GDT(Q_r), respectively. Delete those edges of GDT(Q_l) that intersect $\overline{v_t, v_r}$ from corresponding edge lists accordingly.
5. Assume now that edge $\overline{v_t, v_r}$ is positioned horizontally and v_t and v_r are at the left and right ends respectively. We now concentrate on the problem of computing GDT(Q_j), since GDT(Q_i) can be computed similarly. Let $\overline{v_r, v_{r_1}}, \overline{v_r, v_{r_2}}, \ldots, \overline{v_r, v_{r_m}}$ be Delaunay edges in GDT(Q_r) incident on v_r in clockwise direction and they are above the line $\overleftrightarrow{v_t, v_r}$. Similarly, let $\overline{v_t, v_{t_1}}, \overline{v_t, v_{t_2}}, \ldots, \overline{v_t, v_{t_n}}$ be Delaunay edges in GDT(Q_l) incident on v_t in counterclockwise direction and above $\overleftrightarrow{v_t, v_r}$.

6. Scan the edges incident with v_r in clockwise direction, and let $O(v_r, v_{r_q}, v_{r_{q+1}})$ be the first circle which does not contain v_t, $1 \leq q \leq m-1$. If no such circle exists, let $v_{r_q} = v_{r_m}$. Delete the edges $\overline{v_r, v_{r_w}}$, $w = 1, 2, \ldots, q-1$, from GDT(Q_r) since they are *not* Delaunay edges.
7. Scan the edges incident with v_t in counterclockwise direction, and let $O(v_t, v_{t_z}, v_{t_{z+1}})$ be the first circle which does not contain v_r, $1 \leq z \leq n-1$. If no such circle exists, let v_{t_z} be v_{t_n}. Delete the edges $\overline{v_l, v_{l_w}}$, $w = 1, 2, \ldots, z-1$, from GDT(Q_l) since they are *not* Delaunay edges.
8. If $v_{t_z} = v_j$ and $v_{r_q} = v_j$, then we are done, i.e., the computation of GDT(Q_j) is completed. Otherwise, we have obtained a quadrilateral $v_t v_r v_{t_z} v_{r_q}$ as shown in Fig. 8. Use the maxmin angle criterion on the quadrilateral to decide if $\overline{v_r, v_{t_z}}$ or $\overline{v_t, v_{r_q}}$ is a Delaunay edge. If $\overline{v_r, v_{t_z}}$ is a Delaunay edge, $v_t = v_{t_z}$. If $\overline{v_t, v_{r_q}}$ is a Delaunay edge, $v_r = v_{r_q}$. In either case we obtain a new base $\overline{v_r, v_{t_z}}$ or $\overline{v_t, v_{r_q}}$. Go to Step 4, with either $\overline{v_r, v_{t_z}}$ or $\overline{v_t, v_{r_q}}$ replacing the old base $\overline{v_t, v_r}$.

We now prove a few lemmas that establish the correctness of the above procedure.

Lemma 6. *If no vertex of Q_l visible from v_r lies in $O(v_i, v_j, v_r)$, then $\overline{v_i, v_j}$ is a Delaunay edge.* (*Step* 2.)

Proof. Consider the circle $O(v_i, v_j, v_l)$. Since it must satisfy the circle property in GDT(Q_l), it cannot contain any vertex $v_l \in Q_l$ that is visible from v_i, v_j and v_l. Suppose now that $\overline{v_i, v_j}$ were not a Delaunay edge. Then there must exist a vertex $v_q \in Q_r$ visible from v_i, v_j and v_l that is contained in $O(v_i, v_j, v_l)$. Referring to Fig. 9, vertex v_q must lie on different sides of $\overline{v_i, v_j}$ from v_l, for otherwise v_q is, by simplicity of the polygon, invisible from either v_i or v_j. Furthermore vertex v_q, together with $\triangle v_i v_j v_l$, must form a convex quadrilateral, and hence it must be contained in $O(v_i, v_j, v_r)$. Since $O(v_i, v_j, v_r)$ satisfies the circle property in GDT(Q_r), v_q must be invisible from v_r. That is, $\overline{v_q, v_r}$ must intersect a *side* of the polygon. Since no side of the polygon can intersect $\overline{v_i, v_r}$, we can find a vertex $v_u \in Q_r$ visible from all three vertices v_i, v_j and v_r such that it lies in $O(v_i, v_j, v_r)$, a contradiction. □

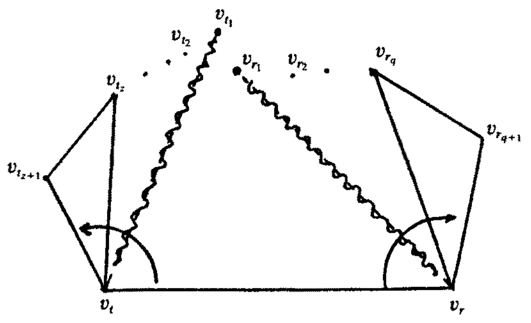


Fig. 8. Illustration of the scanning procedure in Step 8.

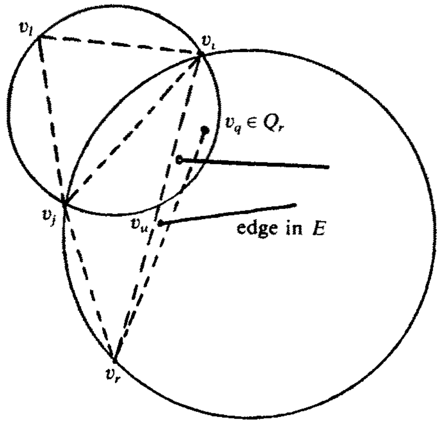


Fig. 9. Illustration for the proof of Lemma 6.

Lemma 7. *The base $\overline{v_t, v_r}$ found in Step 4 is a Delaunay edge.*

Proof. Since the circle $B(v_t, v_r)$ is the smallest circle, it cannot contain any vertex $v_t \in Q_l$ visible from v_r. Moreover, $B(v_t, v_r)$ is totally contained in $O(v_i, v_j, v_r)$ and so it does not contain any vertex $u \in Q_r$ visible from v_r. We therefore conclude that $B(v_t, v_r)$ does not contain any vertex of Q visible from v_r and that $\overline{v_t, v_r}$ is a Delaunay edge by Lemma 5. □

Lemma 8. *Let v_w be a vertex above $\overleftrightarrow{v_t, v_r}$. If $\overline{v_w, v_t}$ and $\overline{v_w, v_r}$ are Delaunay edges in* GDT(Q), *then $v_w \in \{v_{r_1}, v_{r_2}, \ldots, v_{r_m}, v_{t_1}, v_{t_2}, \ldots, v_{t_n}\}$, i.e., v_w must be adjacent to either v_t or v_r in* GDT(Q_l) or GDT(Q_r) *respectively.*

Proof. Since $\triangle v_w v_t v_r$ is a triangle in GDT(Q), $O(v_w, v_t, v_r)$ does not contain any vertex visible from v_t, v_w and v_r. Therefore, if $v_w \in Q_r$, $\overline{v_w, v_r}$ is a Delaunay edge in GDT(Q_r), otherwise, $\overline{v_w, v_t}$ is a Delaunay edge in GDT(Q_l). The claim follows. □

Lemma 9. *One of the two edges $\overline{v_r, v_{t_z}}$ and $\overline{v_t, v_{r_q}}$ obtained in Step 8 is a Delaunay edge.*

Proof. We first show that $O(v_t, v_r, v_{r_q})$ cannot contain any vertex in Q_r visible from v_r and v_{r_q}. Since $v_t \in O(v_r, v_{r_{q-1}}, v_{r_q})$ and $v_t \notin O(v_r, v_{r_q}, v_{r_{q+1}})$ by assumption and $O(v_r, v_{r_{q-1}}, v_{r_q}) \cup O(v_r, v_{r_q}, v_{r_{q+1}})$ contains $O(v_t, v_r, v_{r_q})$, no vertex in Q_r visible from v_r and v_{r_q} is contained in $O(v_t, v_r, v_{r_q})$. Similarly, we can show that $O(v_r, v_t, v_{t_z})$ cannot contain any vertex in Q_l visible from v_t and v'_{t_z}. Next we show if $\overline{v_t, v_{r_q}}$ is selected in Step 8 when the maxmin angle criterion is applied, $O(v_t, v_r, v_{r_q})$ does not contain any vertex in Q_l visible from v_t, and hence $\overline{v_t, v_{r_q}}$ is a Delaunay edge. Since $\overline{v_r, v_t}$ is a Delaunay edge by Lemma 7, $v_{r_q} \notin B(v_r, v_t)$. By assumption, $O(v_r, v_t, v_{t_z})$ contains v_{r_q}. Thus, since $O(v_r, v_t, v_{t_z}) \cup B(v_r, v_t)$ contains $O(v_t, v_r, v_{r_q})$, no vertex in Q_l visible from v_t can be contained in $O(v_t, v_r, v_{r_q})$. The other case when $\overline{v_r, v_{t_z}}$ is selected in Step 8 can be shown similarly. □

Now, let us analyze the running time of the algorithm. In the merge process, assume that GDT(Q_l) and GDT(Q_r) are the two Delaunay triangulations and have one edge in common. The vertices in Q_l visible from v_r can be found in $O(m)$ time when m is the number of vertices in Q_l [7], [19]. All edges in GDT(Q_l) or GDT(Q_r) are checked at most once. And, those deleted edges are not considered again. Therefore, since the total number of edges in GDT(Q_l) and GDT(Q_r) is $O(N)$ and the number of edges added is also $O(N)$, the time for the merge process is $O(N)$. The polygon cutting theorem [4] guarantees that the depth of the simple polygon decomposition tree is $O(\log N)$. Based on the divide-and-conquer technique, we can construct the Delaunay triangulation of a simple polygon with N points in $O(N \log N)$ time.

Theorem 3. *Given a simple polygon P with N vertices, the generalized Delaunay triangulation* GDT(P) *can be completed in $O(N \log N)$ time.*

Proof. The above algorithm computes the Delaunay triangulation of the interior of the polygon P. The exterior of the polygon can be triangulated in a similar manner as follows. We first compute the convex hull of the simple polygon in $O(N)$ time [11], [18], [24]. Each new hull edge $\overline{v_i, v_j}$ and a portion of the boundary of P will define a new simple polygon. These new simple polygons can then be triangulated in a total of $O(N \log N)$ time. The theorem follows. □

5. Conclusion

We have given characterizations of the generalized Delaunay triangulation of a planar straight-line graph $G=(V, E)$ and presented a general algorithm for computing the generalized Delaunay triangulation which runs in $O(|V|^2)$ time. If the graph G is a simple polygon with N vertices, an $O(N \log N)$ algorithm based on the polygon cutting theorem of Chazelle [4] is given, providing yet another triangulation algorithm for simple polygons. The latter algorithm can be applied to the case where the underlying graph is connected.

Recall that the Delaunay triangulation of a set of points in the plane is normally considered as the dual graph of the Voronoi diagram of the set of points [28]. Efficient algorithms for constructing the Voronoi diagram can then be used to obtain the Delaunay triangulation. However, the connection between the notion of generalized Delaunay triangulation for a straight-line planar graph and the notion of generalized Voronoi diagram for line segments [13], [20], [29] is not clear. The crux in computing the generalized Delaunay triangulation for disconnected planar graph is that of converting the graph to a connected one by introducing "new" Delaunay edges thereby the above algorithm can be applied. With that in mind one needs to consider the Voronoi polygon associated with the end vertices of the graph in the Voronoi diagram to see if any Delaunay edge incident with the end vertices can be found. Since the generalized Voronoi algorithm for n line segments can be constructed in $O(n \log n)$ time [13], [29], it may render an $O(|V| \log |V|)$ algorithm for computing the generalized Delaunay triangulation of an arbitrary planar straight-line graph $G = (V, E)$. We intend to carry on the investigation in the future.

References

1. I. Babuska and A. K. Aziz, On the angle condition in the finite element method, *SIAM J. Numer. Anal.* **13** (1976), 214-226.
2. J. H. Bramble and M. Zlamal, Triangular elements in the finite method, *Math. Comp.* **24** (1970), 809-820.
3. J. C. Cavendish, Automatic triangulation of arbitrary planar domains for the finite method, *Internat. J. Numer. Methods Engrg.* **8** (1974), 679-696.
4. B. M. Chazelle, A theorem on polygon cutting with applications, *Proceedings of the 23rd IEEE Annual Symposium on Foundations of Computer Science*, 339-349, 1982.
5. B. M. Chazelle and J. Inceipi, Triangulating a polygon by divide-and-conquer, *Proceedings of the 21st Allerton Conference on Communications Control and Computers*, 447-456, October, 1983.
6. H. Edelsbrunner, L. J. Guibas, and J. Stolfi, Optimal location in a monotone subdivision, *SIAM J. Comput.*, to appear.
7. H. ElGindy and D. Avis, A linear algorithm for determining the visibility polygon from a point, *J. Algorithms* **2** (1981), 186-197.
8. C. O. Frederick, Y. C. Wong, and F. W. Edge, Two-dimensional automatic mesh generation for structural analysis, *Internat. J. Numer. Methods Engrg.* **2** (1970), 133-144.
9. M. R. Garey, D. S. Johnson, F. P. Preparata, and R. E. Tarjan, Triangulating a simple polygon, *Inform. Process. Lett.* **7** (1978), 175-179.
10. R. L. Graham, An efficient algorithm for determining the convex hull of a finite planar set, *Inform. Process. Lett.* **1** (1972), 132-133.
11. R. L. Graham and F. F. Yao, Finding the convex hull of a simple polygon, *J. Algorithms* **4** (1983), 324-331.
12. T. Asano, T. Asano, L. Guibas, J. Hershberger, and H. Imai, Visibility of disjoint polygons, *Algoritnmica* **1** (1986), 49-63.
13. D. G. Kirkpatrick, Efficient computation of continuous skeletons, *Proceedings of the 20th IEEE Annual Symposium on Foundations of Computer Science*, 18-27, October, 1979.
14. D. G. Kirkpatrick, Optimal search in planar subdivisions, *SIAM J. Comput.* **12** (1983), 28-35.
15. C. L. Lawson, C^1-Compatible interpolation over a triangle, JPL Technical Memo. 33-770, May, 1976.
16. C. L. Lawson, Surface for C^1 surface interpolation, JPL Publication 77-30, August, 1977.
17. D. T. Lee, Proximity and reachability in the plane, Ph.D. dissertation, Univ. Illinois, October, 1978.
18. D. T. Lee, On finding the convex hull of a simple polygon, *Internat. J. Comput. Inform. Sci.* **12** (1983), 87-98.
19. D. T. Lee, Visibility of a simple polygon, *Comput. Vision, Graphics, Image Process.* **22** (1983), 207-221.
20. D. T. Lee and R. L. Drysdale III, Generalized Voronoi diagram in the plane, *SIAM J. Comput.* **10** (1981) 73-87.
21. D. T. Lee and F. P. Preparata, Location of a point in a planar subdivision and its applications, *SIAM J. Comput.* **6** (1977), 594-606.
22. D. T. Lee and F. P. Preparata, Computational geometry—a survey, *IEEE Trans. Comput.* **33**, (1984), 1072-1101.
23. D. T. Lee and B. Schachter, Two algorithms for constructing a Delaunay triangulation, *Internat. J. Comput. Inform. Sci.* **9** (1980), 219-242.
24. D. McCallum and D. Avis, A linear algorithm for finding the convex hull of a simple polygon, *Inform. Process. Lett.* **9** (1979), 201-206.
25. D. H. McLain, Two-dimensional interpolation from random data, *Comput. J.* **9** (1976), 178-181.

26. M. J. D. Powell and M. A. Sabin, Pairwise quadratic approximation on triangles, *ACM Trans. Math. Software* **3** (1977), 316–325.
27. F. P. Preparata and M. I. Shamos, *Computational Geometry*, Springer-Verlag, New York–Berlin–Heidelberg, 1985.
28. M. I. Shamos and D. Hoey, Closest point problems, *Proceedings of the 16th IEEE Symposium on Foundations of Computer Science*, 151–162, October, 1975.
29. C. K. Yap, An $O(n \log n)$ algorithm for the Voronoi diagram of a set of simple curve segments, Technical Report of the Courant Institute, New York University, October, 1984.
30. T. M. Yeung, Generalization of Delaunay triangulation, M.Sc. thesis, Department of EE/CS, Northwestern University, May, 1980.

Received April 18, 1985, and in revised form October 28, 1985.