Open access • Journal Article • DOI:10.1137/S0097539795285916

Finding the Constrained Delaunay Triangulation and Constrained Voronoi Diagram of a Simple Polygon in Linear Time — Source link

Francis Y. L. Chin, Cao An Wang

Institutions: University of Hong Kong, Memorial University of Newfoundland

Published on: 15 Feb 1999 - SIAM Journal on Computing (Society for Industrial and Applied Mathematics)

Topics: Constrained Delaunay triangulation, Bowyer–Watson algorithm, Pitteway triangulation, Voronoi diagram and Polygon triangulation

SIAM J. COMPUT.
Vol. 28, No. 2, pp. 471–486

FINDING THE CONSTRAINED DELAUNAY TRIANGULATION AND CONSTRAINED VORONOI DIAGRAM OF A SIMPLE POLYGON IN LINEAR TIME*

FRANCIS CHIN† AND CAO AN WANG‡

Abstract. In this paper, we present an $\Theta(n)$ time worst-case deterministic algorithm for finding the constrained Delaunay triangulation and constrained Voronoi diagram of a simple n-sided polygon in the plane. Up to now, only an $O(n \log n)$ worst-case deterministic and an $O(n)$ expected time bound have been shown, leaving an $O(n)$ deterministic solution open to conjecture.

Key words. algorithm, computational geometry, constrained Delaunay triangulation, polygon, Voronoi diagram

AMS subject classifications. 68P05, 68Q20, 68Q25, 68U05

PII. S0097539795285916

1. Introduction.

Delaunay triangulation and *Voronoi diagram*, duals of one another, are two fundamental geometric constructs in computational geometry. These two geometric constructs for a set of points as well as their variations have been extensively studied [17, 3, 4]. Among these variations, Lee and Lin [15] considered two problems related to *constrained Delaunay triangulation*:[1]

(i) the Delaunay triangulation of a set of points constrained by a set of noncrossing line segments, and
(ii) the Delaunay triangulation of the vertices of a simple polygon constrained by its edges.

They proposed an $O(n^2)$ algorithm for the first problem and an $O(n \log n)$ algorithm for the second one. While the $O(n^2)$ upper bound for the first problem was later improved to $\Theta(n \log n)$ by several researchers [6, 21, 18], the upper bound for the second has remained unchanged and the quest for an improvement has become a recognized open problem [1, 3, 4].

Recently, there have been some results related to this open problem on the Delaunay triangulation of simple polygons. Aggarwal et al. [2] showed that the constrained Delaunay triangulation of a convex polygon can be constructed in linear time. Chazelle [5] presented a linear-time algorithm for finding an "arbitrary" triangulation of a simple polygon. Klein and Lingas showed that the aforementioned open problem for L_1 metrics can be solved in linear time [12], and this problem for the Euclidean metrics can be solved in expected linear time by a randomized algorithm [13]. These efforts all seem to point toward a linear solution to the Delaunay triangulation of simple polygons and support the intuition that the simple polygon problem is easier than the noncrossing line segment problem.

In this paper, we settle this open problem by presenting a deterministic linear-time worst-case algorithm. Our approach follows that of [13]:

*Received by the editors May 5, 1995; accepted for publication (in revised form) August 22, 1996; published electronically July 28, 1998. This work was partially supported by the RGC research grant HKU 439/94E and the NSERC grant OPG0041629.
http://www.siam.org/journals/sicomp/28-2/28591.html
†Department of Computer Science, The University of Hong Kong, Pokfulam Road, Hong Kong (chin@cs.hku.hk).
‡Department of Computer Science, Memorial University of Newfoundland, St. John's, NFLD, Canada A1C 5S7 (wang@garfield.cs.mun.ca).
[1]Same as *generalized Delaunay triangulation* as defined in [15]

(i) first decomposing the given simple polygon into a set of simpler polygons, called *pseudonormal histograms* (*PNHs*),
(ii) constructing the constrained Delaunay triangulation of each normal histogram (NH), and
(iii) merging the constrained Delaunay triangulations of all these NHs to get the result.

In this three-step process, the first and third steps were shown to be possible in linear time, but the second step was done in expected linear time by a randomized algorithm. Our contribution is to show how this second step can be done in linear worst-case time deterministically.

The organization of the paper is as follows. In section 2, we review some definitions and known facts, which are related to our method. In section 3, we concentrate on how to construct the constrained Delaunay triangulation or constrained Voronoi diagram of a normal histogram in linear time. We conclude the paper in section 4.

2. Preliminaries. In this section,
(i) we define the constrained Delaunay triangulation and its dual, the constrained Voronoi diagram,
(ii) we define PNHs, and
(iii) to put our solution of how to construct the constrained Voronoi diagram of a PNH into perspective, we explain the approach taken to first partition any simple polygon into PNHs and then merge constrained Voronoi diagrams of these pseudodiagrams for the solution of the original polygon.

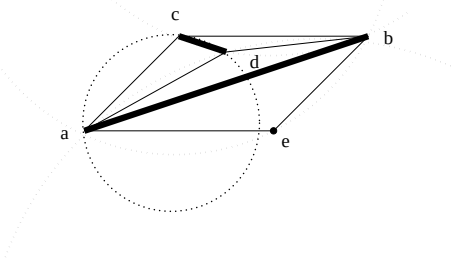
2.1. Constrained Delaunay triangulations and constrained Voronoi diagrams. The *constrained Delaunay triangulation* [15, 6, 21, 18] of a set of noncrossing line segments L, denoted by $CDT(L)$, is a triangulation of the endpoints S of L satisfying the following two conditions:
(i) the edge set of $CDT(L)$ contains L, and
(ii) the line segments in L are treated as obstacles and the interior of the circumcircle of any triangle of $CDT(L)$, say $\Delta ss's''$, does not contain any endpoint in S visible[2] to all vertices s, s', and s''.

Essentially, the constrained Delaunay triangulation is the Delaunay triangulation with the further constraint that the triangulation must contain a set of designated line segments. Figure 1a shows the constrained Delaunay triangulation of two obstacle line segments and a point (a degenerated line segment). In particular, if L forms a nonintersecting chain C, monotone with respect to a horizontal line l, we are only interested in the portion of $CDT(C)$ between C and l. If L forms a simple polygon P, we only consider the portion of $CDT(P)$ internal to P.

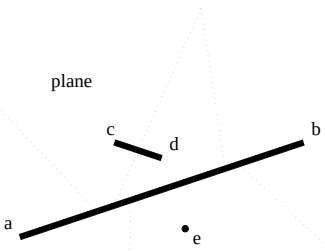
Given a set of line segments L, we can define the Voronoi diagram with respect to L as a partition of the plane into cells, one for each endpoint set S of L, such that a point p belongs to the cell of an endpoint v if and only if v is the closest endpoint visible from p. Figure 1b illustrates the corresponding Voronoi diagram for the set of line segments given in Figure 1a. Unfortunately, this Voronoi diagram is not the complete dual diagram of $CDT(L)$ [3]; i.e., some of the edges in $CDT(L)$ may not have a corresponding edge in this Voronoi diagram.

In [18, 16, 9], the proper dual for the constrained Delaunay triangulation has been defined as the *constrained (or bounded) Voronoi diagram* of L, denoted by $V_c(L)$.

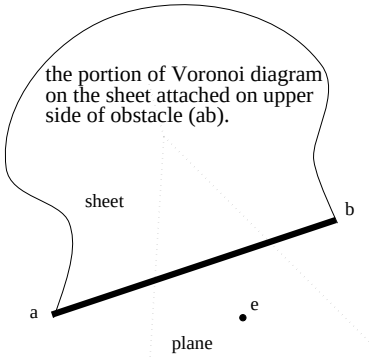
[2]Two points are *visible* to each other if the straight line joining them does not intersect any line segments in L.

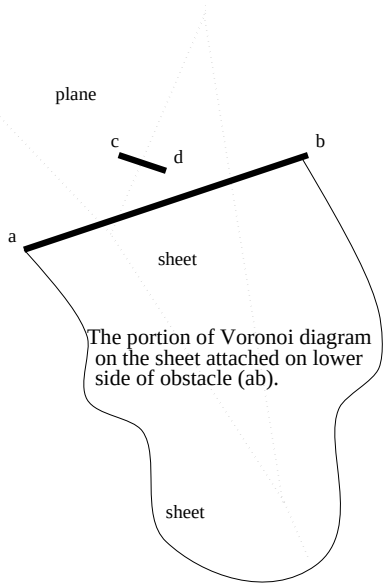


(a) the constrained Delaunay triangulation of line segments (ab) and (cd), and point e, (L).



(b) Voronoi diagram w.r.t. L

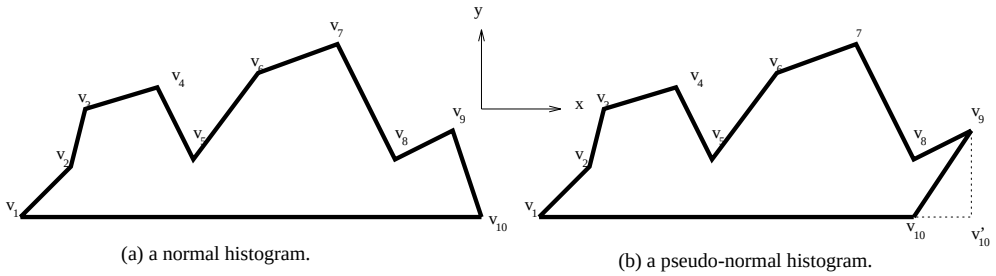




(c) Constrained Voronoi diagram on 'extended' space.

FIG. 1. *Constrained Delaunay triangulation and constrained Voronoi diagram.*

Imagine two sheets or half-planes attached to each side of the obstacle line segments; for each sheet, there is a well-defined Voronoi diagram that is induced only by the endpoints on the other side of the sheet excluding the obstacle line segment attached to the sheet. The constrained Voronoi diagram extends the standard Voronoi diagram by including the Voronoi diagrams induced by the sheets, i.e., the extended Voronoi diagrams beyond both sides of each line segment in L. Figure 1c gives an example of $V_c(L)$, the Voronoi diagrams on the plane and on the two sheets of the obstacle line segment $\overline{ab}$. Note that the Voronoi diagrams on the two sheets of the obstacle line segment $\overline{cd}$ happened to be the same as the Voronoi diagram on the plane. With this definition of $V_c(L)$, there is a one-to-one duality relationship between edges in $V_c(L)$ and edges in $CDT(L)$. It was further proved in [18, 9] that the dual diagrams, $CDT(L)$ and $V_c(L)$, can be constructed from each other in linear time. For simplicity, we omit the word "constrained" over Voronoi diagrams in this paper as all the Voronoi diagrams are deemed to be constrained unless they are explicitly stated to be standard Voronoi diagrams.



(a) a normal histogram. (b) a pseudo-normal histogram.

FIG. 2. *NH and PNH.*

2.2. PNHs. A NH [8] is a monotone polygon with respect to one of its edges, called the *bottom edge*, such that all the vertices of the polygon lie on the same side of the line extending the bottom edge (Figure 2a gives an example). A PNH [13] with a bottom edge e is a simple polygon which, by adding at most one right-angle triangle flush with e, can be transformed into a NH whose bottom edge is the extension of e by the colinear edge of the triangle. Intuitively, a PNH can be viewed as a NH missing one of its bottom corners; i.e., a PNH can be transformed into a NH by adding a right-angle triangle at its bottom[3] (Figure 2b).

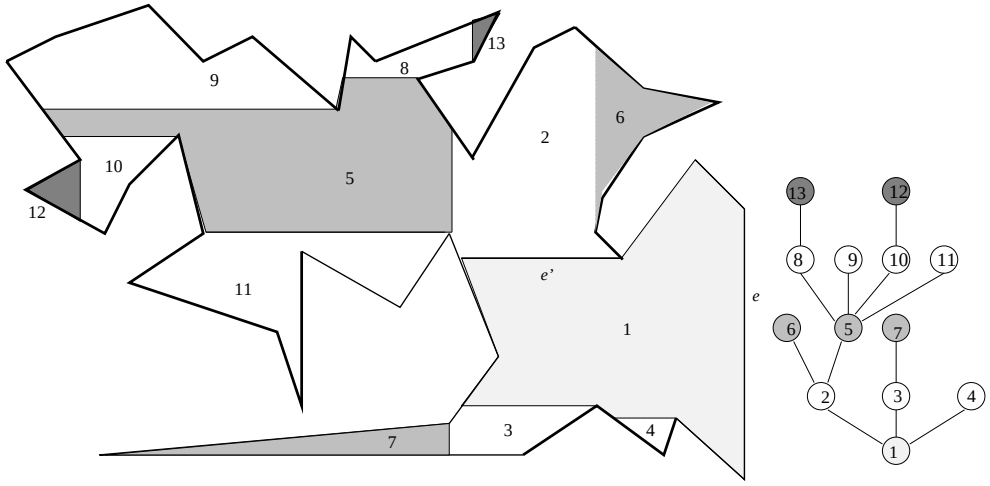


FIG. 3. *A decomposition of P into a tree of PNHs.*

2.3. Decomposition of a simple polygon into PNHs. Figure 3 illustrates how a polygon P is decomposed into 13 PNHs. PNH_1 is associated with the vertical bottom edge e missing its upper bottom corner; PNH_2, associated with the horizontal bottom edge e', is missing its left bottom corner, etc.

A simple polygon P with n vertices can be decomposed into PNHs in $O(n)$ time according to [13] when provided with what are known as the *horizontal* and *vertical visibility maps* of P (Figure 4), which in turn can be obtained in linear time according to [5]. A *diagonal* of P is a line segment joining two vertices of P and lying entirely inside P, while a *chord* of P is a line segment which

[3]Our definition of PNH is different from that given in [13], in which a PNH might be missing both bottom corners. Following the same approach as given in [13], decomposition of a simple polygon into PNHs is also possible with our definition.

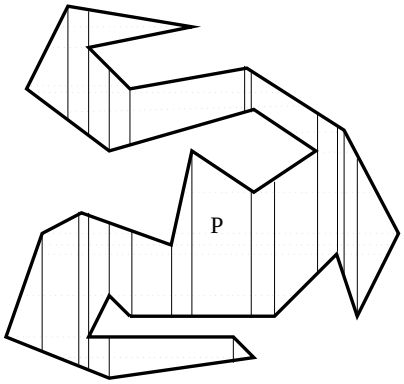


FIG. 4. *Horizontal and vertical visibility maps of simple polygon P.*

(i) lies entirely inside P,
(ii) is parallel to the designated bottom edge, and
(iii) joins a vertex and a boundary point of P (such a boundary point is called a *pseudovertex*).

The *horizontal visibility map* of a simple polygon P is the set of horizontal chords associated with every vertex of P. Note that every vertex of P is associated with at most two chords, and the horizontal visibility map partitions P into a number of horizontal trapezoids (*horizontal trapezoidal decomposition* or *horizontal trapezoidation*). The *vertical visibility map* can be defined similarly.

The decomposition of P into PNHs starts with an arbitrary edge e of P as the bottom edge of the first PNH. The interior of the PNH refers to the part of P that is illuminated by the parallel light emanating into PNH perpendicular to e from its pseudobottom edge $e \cup e_s$, where e_s is the edge (if any) incident to e at an interior angle between 90° and 180°, i.e., the hypotenuse of the missing right-angle triangle. The boundary edges of the PNH that are not edges of P will be the bottom edges in the next step.

The decomposition of P can then be represented by a tree such that each tree node is a PNH and each tree edge represents the adjacency of two PNHs sharing a chord. Consider the example as given in Figure 3: PNH_1, with an edge of P as its bottom edge, is classified as the root. For each edge in PNH_1 which is not an edge of P, we regard it as the bottom edge for a son of PNH_1. PNH_2, PNH_3, and PNH_4 are sons of PNH_1 whose bottom edges are all horizontal, whereas PNH_2 is on one side facing PNH_3 and PNH_4, which are on the other. Similarly, the grandsons of PNH_1 are those with vertical bottom edges adjacent to sons of PNH_1, etc.

2.4. Merging the Voronoi diagrams of PNHs. It has been proved [13] that a Voronoi cell in $V_c(P)$ of a vertex in a PNH would not share any boundary edge with a Voronoi cell of a vertex in another PNH as long as these two vertices are not shared by these two PNHs, and these two PNHs are
(i) at the same depth not facing each other, or
(ii) with their corresponding depths at least two apart.

The Voronoi diagram of a PNH is first merged with the extended Voronoi diagram of its parent, then with those of its sons on one side, and finally with those of the remaining sons on the other. Condition (ii) ensures that the extended Voronoi diagram of its parent will not share any boundary with those of its sons, and thus, only those of

its neighbors (sons and parent) have to be considered in the construction of the part of the Voronoi diagram $V_c(P)$ in a PNH. Condition (i) ensures that merging the Voronoi diagram of a PNH with the extended ones of its sons will trace the bisectors between sites of the PNH at most twice (once for sites on each side) [10, 11, 14, 17, 19]. So, the merging of Voronoi diagrams at each PNH can be done in time linearly proportional to the total size of the PNH and all its sons. Thus, the Voronoi diagram $V_c(P)$ can be obtained in time linearly proportional to the size of P by merging the Voronoi diagram of each PNH together with the extended ones of its neighbors.

In order to find $V_c(P)$ in deterministic linear time, what remains to be solved is the efficient construction of the constrained Voronoi diagram of a PNH. In [13], a randomized algorithm is introduced to find the Voronoi diagram of a NH in expected linear time. The Voronoi diagram of the corresponding PNH can then be obtained by removing the bottom vertex from the Voronoi diagram of this NH, and this can be done in time linearly proportional to the size of the NH. In the next section, we shall concentrate our effort to design a linear-time deterministic algorithm for constructing the Voronoi diagram of a NH.

3. Finding the constrained Voronoi diagram of an NH. Given a normal histogram H with a horizontal bottom edge e, H is decomposed recursively into a tree, say T_I, of smaller normal histograms called *influence normal histograms* (INH), where a node of T_I corresponds to an INH and an edge of T_I indicates an adjacency between two INHs. A formal definition of INH with an algorithmic method of construction will be given in section 3.2. In Figure 5, node 0 (the root INH) is $(v_1, v_3', v_3, v_3'', v_5, v_6, v_7, v_8, v_9,\ v_{10}, v_{12}', v_{12}, v_{13}, v_{13}',\ v_{25}, v_{25}',\ v_{28}, v_{28}', v_{33}, v_{34}, v_{35})$.

Nodes 1–6 form the second level and are sons of node 0.

Node 1 $= (v_3', v_2, v_3)$,
Node 2 $= (v_3, v_4, v_3'')$,
Node 3 $= (v_{12}', v_{11}, v_{12})$,
Node 4 $= (v_{13}, v_{14}, v_{14}', v_{13}')$,
Node 5 $= (v_{25}, v_{26}, v_{27}, v_{25}')$, and
node 6 $= (v_{28}, v_{30}', v_{30}, v_{32}', v_{32}, v_{28}')$.

Nodes 7–9 form the third level with

Node 7 $= (v_{14}, v_{15}, v_{16}, v_{17}, v_{17}',\ v_{19}, v_{20},\ v_{21}, v_{22}, v_{23}, v_{24}, v_{14}')$,
Node 8 $= (v_{30}', v_{29}, v_{30})$,
Node 9 $= (v_{32}', v_{31}, v_{32})$.

Node 10 $= (v_{17}, v_{18}, v_{17}')$ is on the fourth level.

The decomposition ensures that the portion of Voronoi diagram $V_c(H)$ in each INH can only be affected by its own vertices and the vertices of its sons and nothing beyond. In general, the Voronoi cells of $V_c(H)$ associated with vertices of an INH might cross its bottom edge and share edges with Voronoi cells associated with vertices of its parent, but not with those of its brothers or its grandparents. Similarly, the Voronoi cells of an INH would not share any boundary with those of its grandsons. This property implies that, should the Voronoi diagrams of the INHs ($V_c(INH)$) be given, the repeated merging of the Voronoi diagrams of the adjacent INHs can be done in time linearly proportional to the sum of their sizes.

Let $V(p)$ denote the Voronoi cell associated with vertex p in a Voronoi diagram. A point p in a normal histogram H is called an *influence point* if the Voronoi cell $V(p)$ in $V_c(H \cup \{p\})$ will cross H's bottom edge e. The set of influence points is called the *influence region* (IR) with respect to bottom edge e. Consider Figure 5: the IR of H with respect to $\overline{v_1 v_{35}}$ (the bottom edge e) is the region enclosed by $\widehat{v_1 v_5}$, $\overline{v_5 v_6}$, $\overline{v_6 v_7}$, $\overline{v_7 v_8}$, $\overline{v_8 v_9}$, $\overline{v_9 v_{10}}$, $\widehat{v_{10} v_{25}}$, $\widehat{v_{25} v_{28}}$, $\widehat{v_{28} v_{34}}$, $\overline{v_{34} v_{35}}$, and $\overline{v_{35} v_1}$, where $\overline{xy}$ and $\widehat{xy}$ represent,

respectively, the straight line and the arc joining vertices x and y. The *root* (or *root INH*) of T_I is defined as the NH enclosing all influence points of H and consisting of all horizontal trapezoids that intersect the IR. In other words, the root INH would contain all its horizontal chords which will intersect the IR of H, i.e., the *smallest* NH containing IR in the sense that the INH is bound above by the lowest horizontal chords which do not intersect IR. As an example, the root INH is indicated by the unshaded region in Figure 5. Let us now consider the part of H excluding the root INH, which consists of zero or more disjoint polygons. Each polygon is also a NH with a chord as its bottom edge. As given in Figure 5, H is decomposed into a root INH and six other NHs, i.e., the NHs above chords $\overline{v_3'v_3}$, $\overline{v_3v_3''}$, $\overline{v_{12}'v_{12}}$, $\overline{v_{13}v_{13}'}$, $\overline{v_{25}v_{25}'}$, and $\overline{v_{28}v_{28}'}$. For example, the NH above chord $\overline{v_{28}v_{28}'}$ is $(v_{28}, v_{29}, v_{30}, v_{31}, v_{32}, v_{28}')$. The decomposition can be recursively applied to each of these NHs.

Since any node of T_I does not contain the influence points of its parent by the definition of INH, the Voronoi cell associated with a vertex in any node of T_I could not cross the bottom edge of its parent. Thus, the part of $V_c(H)$ within the root can be formed by merging the Voronoi diagram of the root INH with those of its sons. As the Voronoi cells associated with the internal vertices of an INH never share any edges with the Voronoi cells of its brother INH (Theorem 1), the merging can be performed in $O(m_0 + \sum_{i=1}^{s} m_i)$ time, where m_0 is the number of vertices of the root, s is the number of its sons, and m_i is the number of vertices of its ith son.

THEOREM 1. *Let v_1 be a vertex of INH_1 with bottom edge $\overline{u_1w_1}$ and v_2 be a vertex of INH_2 with bottom edge $\overline{u_2w_2}$. Assume that INH_1 and INH_2 are brothers in T_I, $v_1 \neq u_1$, $v_1 \neq w_1$, $v_2 \neq u_2$, and $v_2 \neq w_2$. Then, the Voronoi cell of v_1 will never share any point with the Voronoi cell of v_2.*

Proof. Without loss of generality, assume that $\overline{u_1w_1}$ is on the left-hand side of $\overline{u_2w_2}$; i.e., $u_1 < w_1 \leq u_2 < w_2$ according to their x-coordinates. We show that there does not exist a point p in H

(i) that is equidistant to v_1 and v_2, and
(ii) for which there exists no other vertex in H closer to p than v_1 and v_2.

Assume p exists. Since p is in H, p has to lie directly under $\overline{u_1w_1}$ in order to be closer to v_1 than to u_1 or w_1, i.e., $u_1 < p < w_1$. Similarly, p has to lie directly under $\overline{u_2w_2}$, i.e., $u_2 < p < w_2$. Obviously, p cannot simultaneously satisfy both conditions. □

For example, the Voronoi cell of v_{14} in INH_4 never shares any point with the Voronoi cell of v_{26} or v_{27} in INH_5. Let $M(n)$ denote the merging time for constructing $V_c(H)$ with $\mid H \mid = n$ when provided with the Voronoi diagram of every INH in T_I. Then, we have $M(n) = k(m_0 + \sum_{i=1}^{s} m_i) + \sum_{i=1}^{s} M(n_i)$, where k is a constant and n_i is the number of vertices of the ith subtree. As $n = m_0 + \sum_{i=1}^{s} n_i$, we can show that $M(n) = k(2n - m_0)$ by induction. Thus, the total merging time is $O(n)$. Note that in the above calculation, the pseudovertices are also counted. As n is at most thrice the actual number of vertices of H (as each vertex of H might associate with at most two pseudovertices), the total merging time is still linearly proportional to the actual number of vertices of H.

In the following sections, we shall prove the properties of the IR and the INH which allow us to do efficient merging and identification.

3.1. IR. Let H_V be a subpolygon of NH H, consisting of the bottom edge of H and all those vertices of H with the property that their associated Voronoi cells in $V_c(H)$ cross the bottom edge of H. H_V can also be viewed as the maximum subsequence of the vertices of H having this property. As H is a NH, H_V will also be a

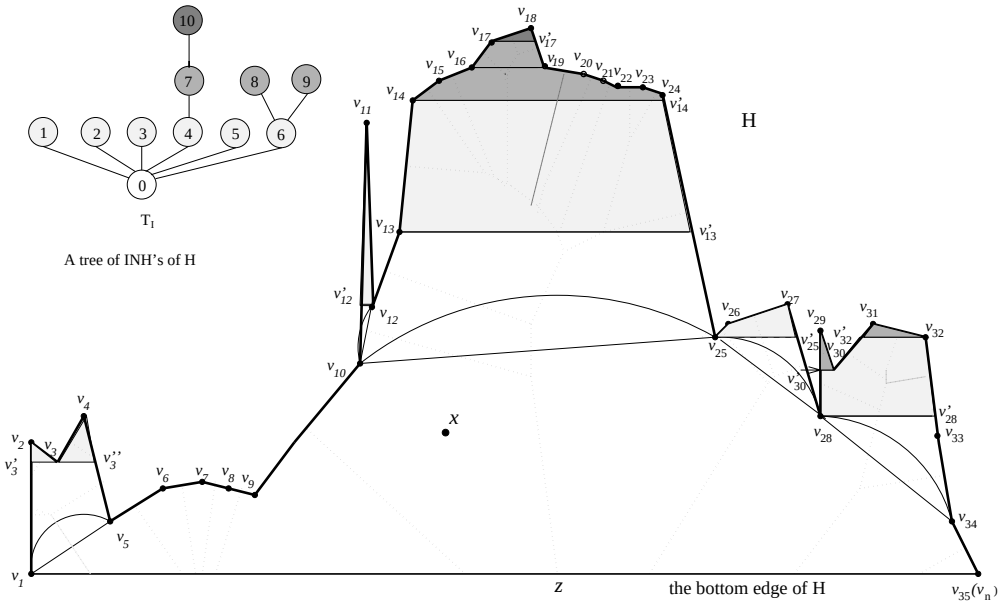


FIG. 5. *Decomposition of H into $INHs$ and T_I.*

NH sharing the same bottom edge as H. Let us consider the example given in Figure 5 again, in which H_V is indicated by the sequence of vertices $(v_1, v_5, v_6, v_7, v_8, v_9, v_{10}, v_{25}, v_{28}, v_{34}, v_{35})$.

LEMMA 1. *All points in H_V are influence points.*

Proof. By the definition of H_V, the bisector of two adjacent vertices (except the two vertices of the bottom edge) of H_V, which forms part of the $V_c(H)$, always crosses the bottom edge of H. In other words, H_V are partitioned by these bisectors into cells, each of which is associated with one of its vertices. These cells resemble the Voronoi cells of $V_c(H)$. In fact, each of these cells in $V_c(H_V)$ always includes its corresponding Voronoi cell in $V_c(H)$. These bisectors also partition the bottom edge into segments according to their closest vertices in H or H_V. It is sufficient to prove this lemma by showing that given any point x in H_V, there always exists a point on the bottom edge which is closer to x than to any vertex in H; i.e., $V(x)$, the Voronoi cell of x, in $V_c(H \cup \{x\})$ would cross the bottom edge. Let x be a point in H_V, in particular, in a Voronoi cell $V(u)$, corresponding to vertex u in $V_c(H_V)$. Furthermore, let the extended line of $\overline{ux}$ intersect the boundary of this Voronoi cell $V(u)$ at y, which may be a point on the bottom edge or a point on a bisector. If y is on the bottom edge, let z be y; otherwise let z be the intersection point of that bisector and the bottom edge. An example is given in Figure 5, where x is in $V(v_{10})$, i.e., $u = v_{10}$. As $\angle uxz > 90°$, z is always closer to x than to u by the triangular property. It is easy to see that z is closer to x than to any other vertices in H, thus z is a point on the bottom edge that belongs to $V(x)$; i.e., $V(x)$ crosses the bottom edge. Hence, x belongs to the IR. □

In general, the IR includes some regions not belonging to H_V. Let b be a boundary edge of H_V. If b is also an edge of H, then b must be an edge of the IR, e.g., $\overline{v_5v_6}$, $\overline{v_6v_7}$, $\overline{v_7v_8}$, etc. in Figure 5. However, if $b = \overline{uw}$ is a diagonal of H, then the IR must include some region of H above b and below the circular arc $\widehat{uw}$, where $\widehat{uw}$ is part of

the semicircle above the bottom edge. (The semicircle is uniquely defined by boundary points u and w and center c, where c is the intersection point of the bottom edge and the perpendicular bisector of u and w.) Let O_{uw} denote the region in H above b (i.e., outside H_V) and below the circular arc $\widehat{uw}$, and let $O_{v_1v_5}$, $O_{v_{10}v_{25}}$, $O_{v_{25}v_{28}}$, and $O_{v_{28}v_{34}}$ be such examples in Figure 5. Then we have the following theorem.

THEOREM 2. *$IR = (\cup_{\overline{uw}\in D} O_{uw}) \cup H_V$, where D is the set of edges of H_V which are diagonals of H.*

Proof. By Lemma 1, we need to prove $IR - H_V = \cup_{\overline{uw}\in D} O_{uw}$ (as $H_V \cap O_{uw} = \phi$). Consider a point p in H, but not in H_V. Then, point p must lie above an edge $\overline{uw}$ of H_V which is a diagonal of H. On one hand, if point $p \in O_{uw}$, then b_{up} and b_{pw} will cross the bottom edge before intersecting each other, where b_{xy} denotes the perpendicular bisector of vertices x and y, and thus p belongs to the IR; i.e., $IR - H_V \supseteq \cup_{\overline{uw}\in D} O_{uw}$. On the other hand, if $p \notin O_{uw}$, then b_{up} and b_{pw} will intersect each other above the bottom edge, and thus p does not belong to the IR; i.e., $IR - H_V \subseteq \cup_{\overline{uw}\in D} O_{uw}$. □

COROLLARY. *Assume a NH H and let $H_V = (v_0, v_1, \ldots, v_n)$. Then the IR with respect to H can be defined by keeping the sequence of vertices of H_V and by replacing all diagonals $\overline{v_iv_{i+1}}$ of H in the sequence of H_V by an arc $\widehat{v_iv_{i+1}}$.* □

3.2. INH. As the root INH is the smallest NH containing the IR, an INH would contain all the edges of the IR, in particular, those edges of H_V (Theorem 2) which are also edges of H (e.g., $\overline{v_5v_6}$, $\overline{v_6v_7}$, $\overline{v_7v_8}$, etc. in Figure 5). As O_{uw} is part of the IR for every $\overline{uw} \in D$ (Theorem 2), the remaining edges of an INH above $\overline{uw}$ would be those chords and edges of H enclosing O_{uw}. Thus, we define H_B *above* $\overline{uw}$ as the smallest NH containing O_{uw}, which consists of edges (or parts of edges) of H and the lowest horizontal chord which do not intersect the IR. For example, as in Figure 5, the H_B's are $(v_1, v_3', v_3, v_3'', v_5)$, $(v_{10}, v_{12}', v_{12}, v_{13}, v_{13}', v_{25})$, $(v_{25}, v_{25}', v_{28})$, and $(v_{28}, v_{28}', v_{33}, v_{34})$.

Now, we can have a precise description of an INH. There are two types of vertices in an INH, the vertices of H_V and the vertices of H_B's, with one H_B for each edge in D. Thus, any vertex in an INH that is not in H_V will be in H_B, and the endpoints of any edge in D will be vertices in both H_V and H_B. In the following, we shall describe the properties of H_B and H_V and show that the Voronoi diagram of an INH can be constructed in linear time.

A *monotonic histogram* is an NH such that if the bottom edge is on the x-axis, then the x-coordinates of the vertices along the boundary are monotonically nondecreasing, and the y-coordinates of the vertices (except the last vertex) along the boundary are monotonically nondecreasing or nonincreasing. A *bitonic histogram* is a composition of two monotone histograms such that the x-coordinates of the vertices along the boundary are monotonically nondecreasing, and the y-coordinates of the vertices along the boundary are first monotonically nondecreasing on one side and then monotonically nonincreasing on the other.

LEMMA 2. *H_B is bitonic.*

Proof. Since H_B is the smallest NH enclosing O_{uw}, all its internal horizontal chords will intersect with O_{uw}; i.e., all vertices of H_B, except possibly the top vertex and its associated pseudovertex (vertices), should be horizontally visible from O_{uw}. As H_B consists of only edges (or parts of edges) and chords of H, all edges of H_B should be monotonically nondecreasing in the x- and y-coordinates on one side and monotonically nondecreasing in the x-coordinate but monotonically nonincreasing in the y-coordinate on the other. Thus, H_B is bitonic. □

LEMMA 3. *The Voronoi diagrams of H_B and H_V can be constructed in linear time.*

Proof. It is shown in [8] that the Voronoi diagram of a monotonic histogram can be constructed in linear time. Because H_B can be partitioned into two monotonic histograms by the vertical line through its highest vertex or edge (Lemma 2), the Voronoi diagrams of two such monotonic polygons can be merged in linear time [20, 13]. Thus, $V_c(H_B)$ can be found in linear time. As far as H_V is concerned, because the vertices on the boundary of H_V are in sorted order according to their x-coordinates (property of the NH), the extended Voronoi diagram below the bottom edge can be found in linear time [2]. Since all the Voronoi cells in H_V must cross the bottom edge, the Voronoi diagram of H_V can be constructed in linear time from its extended Voronoi diagram below the bottom edge. □

Note that in the construction of the Voronoi diagrams of H_B and H_V, all the pseudovertices are ignored. Thus, the resulting Voronoi diagrams do not contain any Voronoi cell of pseudovertices. This approach is different from that proposed in [13], which requires the removal of the Voronoi cells of pseudovertices.

The following lemma shows that the Voronoi diagrams of two H_B's cannot affect each other.

LEMMA 4. *Assume an INH with its attached H_B's, and let x and y be two vertices not belonging to H_V but in two different H_B's. Then the Voronoi cells, $V(x)$ and $V(y)$, cannot share any point in $V_c(H_V)$.*

Proof. As x and y are vertices in two different H_B's but not in H_V, x and y must be separated by some vertex z in H_V. By the definition of H_V, the Voronoi cell $V(z)$ must cross the bottom edge. Thus, $V(x)$ cannot share any point with $V(y)$ above the bottom edge. □

THEOREM 3. *The Voronoi diagram of an INH can be constructed in time linearly proportional to its size.*

Proof. By Lemma 3, the Voronoi diagrams of H_V and H_B's can be constructed in time linearly proportional to their sizes. Since each H_B shares an edge with H_V, the Voronoi diagrams of each H_B and H_V can be merged in time proportional to the number of Voronoi edges shared by them [20, 13]. As different H_B's do not interfere with each other (Lemma 4), the total merging time is linearly proportional to the number of Voronoi edges shared by H_B's and H_V, i.e., the size of the INH. □

3.3. Region identification. In this section, we shall present an algorithm which identifies the INH in a NH in time linearly proportional to the size of the INH. Chazelle's linear-time algorithm [5] is first applied to the NH to obtain its horizontal visibility map (Figure 6). By the property of a normal histogram, H can be further represented by a *partition tree* T_P, in which each tree node represents a chord in the map and each tree edge represents the adjacency of two chords. Let $n(v)$ denote the chord(s) associated with vertex v of H. If there are two chords in $n(v)$, $n^L(v)$ and $n^R(v)$ denote the left chord and right chord, respectively. With this partition tree T_P, the INH to be identified can be represented as a rooted subtree of T_P.[4] For example, the INH indicated by the shaded area can be represented by the rooted subtree as marked in Figure 6. The algorithm to identify the INH is based on tree traversal. In order to achieve linear time complexity, only those tree nodes relevant to the INH will be traversed. Thus, one of the key steps in the tree traversal is the pruning condition, i.e., under what conditions the traversal of a subtree can be terminated. The other key step is the identification of the vertices of H_V so that we can partition the INH

[4] A rooted subtree of T has the property that the root of T is also the root of the subtree.

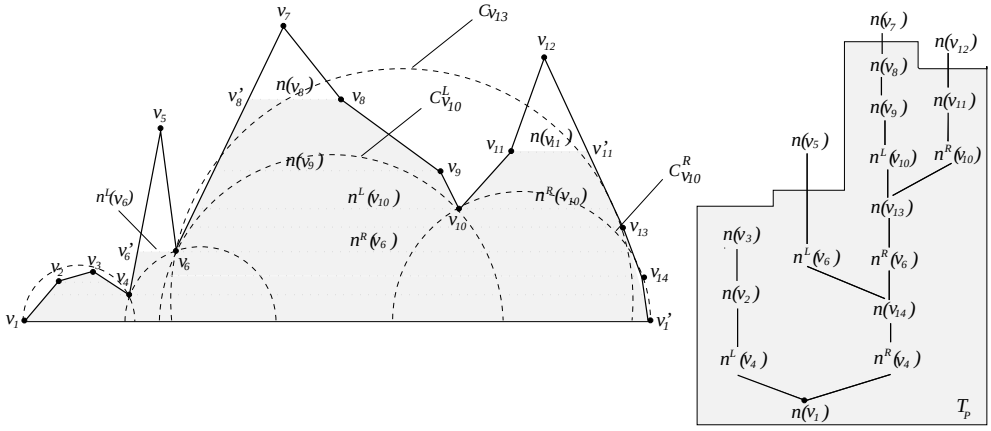


FIG. 6. *An INH and its tree* T_P.

into H_B's and H_V for further constructing of Voronoi diagrams, as described in the previous section.

The following lemma gives a necessary and sufficient condition for a vertex v of H to be a vertex of H_V. Based on Theorem 2 and the definition of H_B, we can ensure that a visited vertex, that is not a vertex of H_V, will be a vertex of H_B.

LEMMA 5. *For any vertex v of H, v is a vertex of H_V if and only if v can be touched by a circle centered at the bottom edge and empty of other vertices of H.*

Proof. The center of such a circle is a point closer to vertex v than to other vertices of H. As Voronoi cells are simply connected, the Voronoi cell $V(v)$ will intersect or cross the bottom edge. The lemma follows directly from the definition of H_V. □

Without loss of generality, assume the parent of $n(v)$ intersects the IR and $n(v)$ is being visited on the traversal of T_P. Based on Lemma 5, vertex v is tested and classified into one of the following three types: (i) a vertex in H_V, (ii) a vertex in H_B (if not in H_V), or (iii) a potential vertex.

A *potential vertex* is such a vertex that can be touched by a circle centered at the bottom edge and empty of any vertices of H *on or below* $n(v)$; i.e., the Voronoi cell of a potential vertex would extend across the bottom edge if no vertex above $n(v)$ will affect this Voronoi cell. However, since we have not examined any vertex above $n(v)$ yet, we cannot rule out the possibility that v is in H_B.

Assume that all circles in the following discussion will be centered at the bottom edge. Let C_v denote the largest circle that crosses chord $n(v)$ and whose interior does not contain any vertex on or below $n(v)$. If vertex v is associated with two chords, then C_v^L and C_v^R denote such circles that cross $n^L(v)$ and $n^R(v)$, respectively. Let us first study some properties of C_v, which can be used to determine the IR above $n(v)$ by considering only the histogram studied so far, i.e., the histogram below $n(v)$ (note that C_v^L and C_v^R also have these properties).

LEMMA 6. *If C_v exists, then*

(i) *C_v must touch vertices to the left and right of its centers, or is centered at a nonvertex endpoint of the bottom edge, i.e., pseudovertex, and*

(ii) *if v is the left (right) endpoint of $n(v)$, then the center of C_v must be to the right (left) of v.*

Proof. (i) If C_v exists, then C_v is unique. This is because, by the x-monotonic property of H, there is no vertex below the bottom edge (e) and between the intervals

of $n(v)$; i.e., the largest circle centered at e cannot be bounded by such vertices. Then, either the largest circle centered at e must touch only one vertex above e and hence will be centered at the (nonvertex) endpoint of e, or this circle must touch two or more vertices above e on both sides of its center. In either case, this largest circle is the C_v if it crosses $n(v)$. (ii) By contradiction, assume that the center of C_v is to the left of v, which is the left endpoint of $n(v)$; then C_v would not cross $n(v)$ and cannot exist. □

The following lemma will give a sufficient condition for a vertex to be in H_V.

LEMMA 7.

(i) *If both C_v^L and C_v^R exist, then vertex v must be in H_V.*

(ii) *If C_v^L (C_v^R) does not exist, then the traversal of the subtree above $n^L(v)$ ($n^R(v)$) is terminated.*

Proof. (i) The existence of both C_v^L and C_v^R implies the existence of an empty circle with center at the bottom edge and lying entirely below $n(v)$. Thus v must be in H_V by Lemma 5. (ii) Since there does not exist any vertex above $n^L(v)$ that can affect the IR, the tree traversal can be terminated. □

Let us consider an example as given in Figure 6 to show how these properties can be applied to classify v. Initially C_{v_1} is the circle that touches v_1 and is centered at v_1'. The next vertex to be visited is v_4. Since both $C_{v_4}^L$ and $C_{v_4}^R$ exist, v_4 is in H_V (Lemma 7). When the tree traversal of T_P at $n^R(v_4)$ is continued, vertices v_{14}, v_6, and v_{13} will be visited and classified as potential vertices.

In order to construct C_v for each v during the tree traversal T_P, the potential vertices previously identified are kept in two stacks, L and R. Except possibly for their bottom vertices, L contains the "left" potential vertices (i.e., left endpoints of the corresponding chords) while R contains the "right" ones. For example, after v_6 and v_{13} are visited, L and R contain $[v_4, v_6)$ and $(v_{13}, v_{14}]$, respectively, with v_6 and v_{13} being their top vertices. The following lemma gives the properties of stacks L and R.

LEMMA 8. *With respect to the histogram on or below the chord $n(v)$ studied so far,* (i) *stacks L and R contain the vertices whose Voronoi cells cross the bottom edge in order, and* (ii) *the largest empty circle C_v is determined by the top vertex of L and the top vertex of R.*

Proof. (i) First, we have to show that the y-coordinates of the vertices in L are monotonically increasing while those in R are monotonically decreasing. Without loss of generality, assume the contrary, that L contains a vertex v whose y-coordinate is lower than that of its precedent vertex t. Then it is impossible for t to be the left endpoint of a chord. As the vertices in L and R are potential vertices visited according to their y-coordinates, their Voronoi cells must cross the bottom edge in order.

(ii) As L and R contain the "left" and "right" potential vertices (i.e., left and right endpoints of the chords), the largest circle C_v must touch the top vertices of L and R, which are the rightmost left endpoint and the leftmost right endpoints, respectively. □

We shall describe the construction of $C_{v'}$ and the tree traversal algorithm of T_P. Assume v' is the next vertex to be visited after v.

Case 1. $n(v')$ does not exist or does not intersect C_v; the tree traversal is terminated/pruned at $n(v')$ and all vertices in L and R become vertices in H_V. Subtrees rooted at such $n(v')$ (if they exist) are pruned because their corresponding INHs do not contain the IR. The pruned subtrees represent smaller NHs needed to be processed recursively.

For example, in Figure 6, the tree traversal is terminated at $n(v_8)$ and $n(v_{11})$ as they do not intersect C_{v_9} and $C^R_{v_{10}}$, respectively. Vertex v_6, previously in L, becomes a vertex in H_V. In Figure 5 the pruned portion $(v_{28}, v_{29}, v_{30}, v_{31}, v_{32}, v_{28'})$ is an NH with $\overline{v_{28}v_{28'}}$ (the pruned chord) as the bottom edge to be processed recursively.

Case 2. $n(v')$ intersects C_v; the tree traversal will continue to visit $n(v')$'s son(s).

(a) v' is outside C_v (i.e., $b_{uv'}$ and $b_{v'w}$ intersect each other above the bottom edge where u and w are top vertices of L and R, respectively). $-$ v' will not be in H_V and must be in H_B; stacks L and R remain unchanged. Vertex v_9 in Figure 6 is such an example.

(b) v' is inside C_v (i.e., $b_{uv'}$ and $b_{v'w}$ cross the bottom edge before intersecting each other), and $-$ v' may be closer to some point of the bottom edge than vertices in L and R. The Voronoi cell of v' crosses the bottom edge and may crowd out the Voronoi cells of some vertices in L and R. If the largest circle determined by the next-to-top vertex of L and v' does not contain the top vertex of L (i.e., $b_{u'u}$ intersects $b_{uv'}$ above the bottom edge where u' is the next-to-top vertex of L), then pop the top vertex of L and assign it as a vertex in H_B. Vertices of L are popped until its top vertex remains in L; Lemma 8 guarantees that all vertices beneath also remain in L. Stack R is handled similarly. For example, in Figure 6, v_{13} is popped when v_{10} is visited.

  (i) If either $C^L_{v'}$ or $C^R_{v'}$ exists but not both, v' is pushed onto stack $L(R)$ if v' is the left (right) endpoint of that chord. The tree traversal is continued at $n(v')$ using the new empty circle $C_{v'}$, new stack $L(R)$, and old stack $R(L)$. Vertices v_6 and v_{13} are such examples which are pushed onto their corresponding stacks L and R when $n(v_6)$ and $n(v_{13})$ are visited.

  (ii) If both $C^L_{v'}$ and $C^R_{v'}$ exist, v' must be a vertex in H_V by Lemma 7. The tree traversal will continue at $n^L(v')$, where $C^L_{v'}$, old stack L, and new stack R' containing v' alone will be used for further vertex classification. Similarly, the traversal at $n^R(v')$ will use $C^R_{v'}$, old stack R, and a new stack L' containing only v'. Vertex v_{10} in Figure 6 is such an example.

For visualizing the above algorithm, Figure 7 gives a walk-through of the example in Figure 6.

3.4. Complexity analysis. Our method for constructing the constrained Voronoi diagram of a simple polygon P mainly relies on the efficiency of the identification of the INHs from an NH. Since the identification for different INHs is executed recursively, we shall only consider the root INH of an NH.

As described previously, when we traverse tree T_P of an NH to identify an INH, we visit each vertex of the INH exactly once. Those vertices which have not been visited in the traversal of T_P cannot belong to the root INH. Therefore, we only need to show that each visited vertex is tested in constant time in order to classify it as a vertex in H_V or in H_B.

Let us consider a vertex v. In the test, v can be classified into one of the following three types: (i) $v \epsilon H_V$, (ii) $v \epsilon H_B$, and (iii) v is a potential vertex.

For type (i), v is stored in the list of vertices representing H_V.

For type (ii), v is stored in the list of vertices corresponding to a particular H_B and vertex v will never be tested again. Note that each vertex in H_V or potential vertex is associated with a separate list H_B of vertices. If the potential vertex, separating the two lists of vertices corresponding to two H_B's, has been determined to be in H_B, then these two lists of vertices will be concatenated together.

For type (iii), v is stored in the left or right stack and could be repeatedly tested

Initially $L = [v_1)$, $R = (\ \ ]$, $H_V = (v_1)$
visiting v_4: insert v_4 into H_V $\quad /* case2b(ii) */$
left chord $n^L(v_4)$, $L = [v_1)$, $R = (v_4]$
visiting v_2: chord $n(v_2)$, $L = [v_1, v_2)$, $R = (v_4]$ $\quad /* case2b(i) */$
visiting v_3: traversal terminated, insert v_2, v_3 into H_V $\quad /* case1 */$
right chord $n^R(v_4)$, $L = [v_4)$, $R = (\ \ ]$
visiting v_{14}: chord $n(v_{14})$, $L = [v_4)$, $R = (v_{14}]$ $\quad /* case2b(i) */$
visiting v_6: $\quad /* case2b(i) */$
left chord $n^L(v_6)$, traversal pruned, process NH above $n^L(v_6)$ recursively
$/* case1 */$
right chord $n^R(v_6)$, $L = [v_4, v_6)$, $R = (v_{14}]$
visiting v_{13}: chord $n(v_{13})$, $L = [v_4, v_6)$, $R = (v_{13}, v_{14}]$ $\quad /* case2b(i) */$
visiting v_{10}: insert v_{10} in H_V $\quad /* case2b(ii) */$
left chord $n^L(v_{10})$, $L = [v_4, v_6)$, $R = (v_{10}]$
visiting v_9: $H_B(-, v_{10}) = (v_9, v_{10})$ $\quad /* case2a */$
visiting v_8: traversal pruned, insert v_6 into H_V $\quad /* case1 */$
$H_B(v_6, v_{10})$ = concatenate (v_6, v'_8, v_8) and $H_B(-, v_{10}))$
$= (v_6, v'_8, v_8, v_9, v_{10})$
process NH above $n(v_8)$ recursively
right chord $n^R(v_{10})$, $L = [v_{10})$, pop v_{13} from R and assign v_{13}
as an element in H_B,
$R = (v_{14}]$, H_B(-, v_{14})=(v_{13}, v_{14})
visiting v_{11}: traversal pruned, $\quad /* case1 */$
$H_B(v_{10}, v_{14})$ = concatenate $(v_{10}, v_{11}, v'_{11})$ and $H_B(-, v_{14}))$
$= (v_{10}, v_{11}, v'_{11}, v_{13}, v_{14})$
process NH above $n(v_{11})$ recursively
Finally $H_V = (v_1, v_2, v_3, v_4, v_6, v_{10}, v_{14})$

FIG. 7. *Walk-through of the example in Figure* 6.

when the descendants of v are visited. However, once vertex v is identified to be a vertex in H_V or H_B, v will never be tested again. Thus, we can argue that the time for visiting a vertex is constant when amortized over a sequence of tests. To see this, our analysis assumes that one unit credit should have been assigned to each potential vertex in L and R. For each vertex v of H in the bottom-up sweep of T_P, two unit credits of work are needed for each test: one for carrying the test itself, i.e., either assigning v as a vertex in H_V, H_B or a potential vertex in L or R; the other unit credit is assigned to the vertex should it be identified as a potential vertex. The test on a vertex in L or R to determine whether or not it has to be reassigned to H_B or H_V will be paid by the unit credit associated with the vertex. This either happens once for the checked vertex (which is accounted to it) or this test stops at a vertex which still cannot be reassigned and the test stop condition is accounted to the vertex again.

It is not difficult to see that linked lists can be used to keep track of the vertices in H_V and H_B's. In particular, insertion and concatenation operations on H_V and H_B's can be executed in constant time. The time complexity analysis for the construction of Voronoi diagrams of INH, NH, and P is obvious, as described in the previous sections. We shall conclude the above analysis by the following theorem.

THEOREM 4. *$CDT(P)$ can be found in $\Theta(|\ P\ |)$ time for simple polygon P.*

4. Concluding remarks. In this paper, we presented a deterministic algorithm for finding the constrained Delaunay triangulation of a simple polygon with n sides in $\Theta(n)$ time in the worst case. This may be one of the few linear-time algorithms for nonarbitrary triangulation of a simple polygon.

In the definition of Delaunay triangulation, we can check whether a triangulation is Delaunay by studying vertices within local proximity. It should not be surprising that the Delaunay triangulation and the constrained Voronoi diagram of a simple polygon can be done in linear time after being given Chazelle's horizontal visibility map, which links vertices within proximity together. The horizontal visibility maps are helpful to decompose the polygon into components such that the "divide and conquer" approach can be applied. However, if the decomposition of the polygon into components is not carefully done, "interaction" of the Voronoi diagrams of the components may be more than linear (even quadratic time). From Theorem 1, the partition of the polygon into components by chords has the advantage that the Voronoi diagrams of the components at the same level would not interact with each other; i.e., horizontal interaction can be reduced. Moreover, because of the property of H_V, interaction of Voronoi diagrams of components at different levels can also be confined; i.e., vertical interaction can be eliminated.

With our linear-time algorithm, the following related problems can also be solved efficiently:

(1) all nearest (mutual visible) neighbors of the vertices of a simple polygon [13],

(2) a shortest diagonal of a simple polygon [13],

(3) a largest inscribing circle of vertices of a simple polygon [12],

(4) the nearest vertex from a query point [13],

(5) finding $DT(S)$ if the Euclidean minimum spanning tree for a point set S is given [1],

(6) finding standard Voronoi diagram for S' if the Voronoi diagram of a point set S is known [1], where $S' \subset S$.

By treating edges and vertices of a single polygon as sites for the Voronoi diagram, we can apply ideas similar to those given in this paper to find the medial axis of a simple polygon in linear time [7].

Acknowledgment. The authors would like to thank Bethany Chan, Siu-Wing Cheng, and the anonymous referees for their patience in reading the first draft of this paper and their comments in improving the readability of the paper.

REFERENCES

[1] A. Aggarwal (1988), *Computational Geometry*, MIT Lecture Notes 18.409, MIT, Cambridge, MA.

[2] A. Aggarwal, L. Guibas, J. Saxe, and P. Shor (1989), *A linear time algorithm for computing the Voronoi diagram of a convex polygon*, Discrete Comput. Geom., 4, pp. 591–604.

[3] A. Aurenhammer (1991), *Voronoi diagrams: A survey*, ACM Computing Surveys, 23, pp. 345-405.

[4] M. Bern and D. Eppstein (1992), *Mesh Generation and Optimal Triangulation*, Technical Report, Xerox PARC, Palo Alto, CA.

[5] B. Chazelle (1991), *Triangulating a simple polygon in linear time*, Discrete Comput. Geom., 6, pp. 485–524.

[6] P. Chew (1987), *Constrained Delaunay triangulation*, in Proc. 3rd ACM Symposium on Comp. Geometry, pp. 213–222.

[7] F. Chin, J. Snoeyink, and C. A. Wang (1995), *Finding the medial axis of a simple polygon in linear time*, in Proc. 6th International Symposium ($ISAAC'95$), Lecture Notes in Comput. Sci. 1004, Springer-Verlag, New York, pp. 382–391.

[8] H. Djidjev and A. Lingas (1991), *On computing the Voronoi diagram for restricted planar figures*, in Lecture Notes in Comput. Sci. 519, Springer-Verlag, New York, pp. 54–64.
[9] B. Joe and C. Wang (1993), *Duality of constrained Delaunay triangulation and Voronoi diagram*, Algorithmica, 9, pp. 142–155.
[10] D.G. Kirkpatrick (1979), *Efficient computation of continuous skeletons*, in Proc. 20th IEEE Symposium on Foundations of Computer Science, pp. 18–27.
[11] R. Klein (1989), *Concrete and Abstract Voronoi Diagrams*, Lecture Notes in Comput. Sci. 400 Springer-Verlag, New York.
[12] R. Klein and A. Lingas (1992), *A linear time algorithm for the bounded Voronoi diagram of a simple polygon in L_1 metrics*, in Proc. 8th ACM Symposium on Comp. Geometry, pp. 124–133.
[13] R. Klein and A. Lingas (1993), *A linear time randomized algorithm for the bounded Voronoi diagram of a simple polygon*, in Proc. 9th ACM Symposium on Comp. Geometry, pp. 124–133.
[14] D. T. Lee (1982), *Medial axis transformation of a planar shape*, IEEE Trans. Pat. Anal. Mach. Int., PAMI-4(4), pp. 363–369.
[15] D. Lee and A. Lin (1986), *Generalized Delaunay triangulations for planar graphs*, Discrete Comput. Geom., 1, pp. 201–217.
[16] A. Lingas (1987), *A space efficient algorithm for the constrained Delaunay triangulation*, in Lecture Notes in Control and Inform. Sci. 113, Springer-Verlag, New York, pp. 359–364.
[17] F. P. Preparata and M. I. Shamos (1985), *Computational Geometry - An Introduction*, Springer-Verlag, New York.
[18] R. Seidel (1988), *Constrained Delaunay Triangulations and Voronoi Diagrams with Obstacles*, Rep. 260, IIG-TU Graz, Austria, pp. 178–191.
[19] M. I. Shamos and D. Hoey (1975), *Closest point problems*, in Proc. 16th IEEE Symposium on the Foundations of Computer Science, pp. 151–162.
[20] C. Wang (1993), *Efficiently updating the constrained Delaunay triangulations*, BIT, 33, pp. 176–181.
[21] C. Wang and L. Schubert (1987), *An optimal algorithm for constructing the Delaunay triangulation of a set of line segments*, in Proc. 3rd ACM Symposium on Comp. Geometry, pp. 223–232.