An Improved Algorithm for Constructing kth-Order Voronoi Diagrams

BERNARD CHAZELLE AND HERBERT EDELSBRUNNER

Abstract—**The kth-order Voronoi diagram of a finite set of sites in the Euclidean plane E^2 subdivides E^2 into maximal regions such that all points within a given region have the same k nearest sites. Two versions of an algorithm are developed for constructing the kth-order Voronoi diagram of a set of n sites in $O(n^2 \log n + k(n-k)\log^2 n)$ time, $O(k(n-k))$ storage, and in $O(n^2 + k(n-k)\log^2 n)$ time, $O(n^2)$ storage, respectively.**

Index Terms—**Arrangements of lines and planes, computational geometry, Euclidean and projective space, geometric transforms, maintenance of convex hulls, Voronoi diagrams.**

I. Introduction

LET S be a set of n points (called *sites*) in the Euclidean plane E^2. For any subset S' of S, the set of points

$$\text{dom}(S') = \{x \in E^2 \mid d(x, s) < d(x, t), \quad s \in S', t \in S - S'\}$$

is called the *Voronoi region* of S'. Intuitively, dom (S') is the set of points closer to all sites in S' than to any site in $S - S'$. For two sites s and t, define

$$h(s, t) = \{x \in E^2 \mid d(x, s) < d(x, t)\};$$

$h(s, t)$ is the open half-plane bounded by the perpendicular bisector of s and t that contains s. With this definition, we have

$$\text{dom}(S') = \bigcap_{s \in S, t \in S - S'} h(s, t),$$

which implies that dom (S') is an open convex polygon. Obviously, dom $(S') \cap$ dom $(S'') = \emptyset$ if $|S'| = |S''|$ and $S' \neq S''$. In fact, for any value of k between 1 and $n-1$ the Voronoi regions of all subsets of cardinality k induce a subdivision of E^2, called the *kth-order Voronoi diagram of S* which we denote as $\mathcal{V}_k(S)$. No two regions of $\mathcal{V}_k(S)$ intersect, and the closures of all regions cover E^2. The closure of two regions intersect in a (possibly empty or degenerate) line segment. If this line segment is neither empty nor degenerate, we call its relative interior an *edge* of $\mathcal{V}_k(S)$. The nonempty intersections of the closures of three or more regions are the *vertices of* $\mathcal{V}_k(S)$. Fig. 1 shows the first- and the second-order Voronoi diagrams of 11 sites. Note that not every pair of points defines a region in the second-order diagram.

Each vertex of $\mathcal{V}_k(S)$ is equally distant from at least three sites. Consequently, each vertex is incident to precisely three edges, unless there are four sites that lie on a common circle which we consider to be a degenerate case. In [13] it is shown that the number of regions, edges, and vertices of $\mathcal{V}_k(S)$ do not exceed $O(k(n-k))$.

First-order Voronoi diagrams have appeared throughout the scientific literature (see [6], [17], [1], [10], [2], [12], [15], [16], [3]). In computer science, these diagrams were christened after Voronoi, a Russian mathematician who investigated the geometry of numbers (see [17]). Algorithms for constructing $\mathcal{V}_1(S)$ in $O(n \log n)$ time and $O(n)$ storage can be found in [16], [15], [3], [11]. In [13] an algorithm for computing $\mathcal{V}_k(S)$ is presented; the method requires $O(k^2 n \log n)$ time and $O(k^2 n)$ storage. Another method that can also be used to construct diagrams of a more general nature and that takes $O(k(n-k)\sqrt{n}\log n)$ time and $O(n(n-k))$ storage is reported in [7]. Finally, it is shown in [8] that $O(n^3)$ time and storage suffice for computing all kth-order Voronoi diagrams of S, for $k = 1, 2, \cdots, n-1$. A host of applications of Voronoi diagrams can be found in [15].

The main result of this paper consists of two versions of a new method for constructing $\mathcal{V}_k(S)$: one requires $O(n^2 \log n + k(n-k)\log^2 n)$ time and $O(n(n-k))$ storage, while with additional $O(n^2)$ preprocessing and storage, the other version speeds up the construction to $O(n^2 + k(n-k)\log^2 n)$ time. Fig. 2 displays the requirements in time and storage of the two algorithms and of the algorithms in [13] and [7] using logarithmic scales. This display does not show differences by factors of $\log n$. To distinguish the four algorithms, we use label "L" for the algorithm in [13], label "E" for the algorithm in [7], and labels "CE1" and "CE2" for the first and the second algorithm to be presented in this paper. Of course, one can combine the algorithms such that the time needed is the minimum of the requirements of any of the displayed methods.

The organization of this paper is as follows. Section II discusses a geometric transformation that maps S to a set of planes in three dimensions; Voronoi diagrams can be obtained by vertical projection of certain "skeletons of edges" in the three-dimensional arrangement defined by these planes. Sec-

Manuscript received August 21, 1985; revised January 29, 1987 B. Chazelle was supported in part by NSF Grants MCS-83-03925 under Contract N000 14-83-K-0146, and the Office of Naval Research and the Defense Research Projects Agency under Contract N000 14-83-K-0146 and ARPA Order 4786.

B. Chazelle was with the Department of Computer Science, Brown University, Providence, RI. He is now with the Department of Computer Science, Princeton, University, Princeton, NJ 08544.

H. Edelsbrunner was with the Institutes for Information Processing of the Technical University of Graz, Graz, Austria. He is now with the Department of Computer Science, University of Illinois, Urbana-Champaign, IL 61801.

IEEE Log Number 8715435.

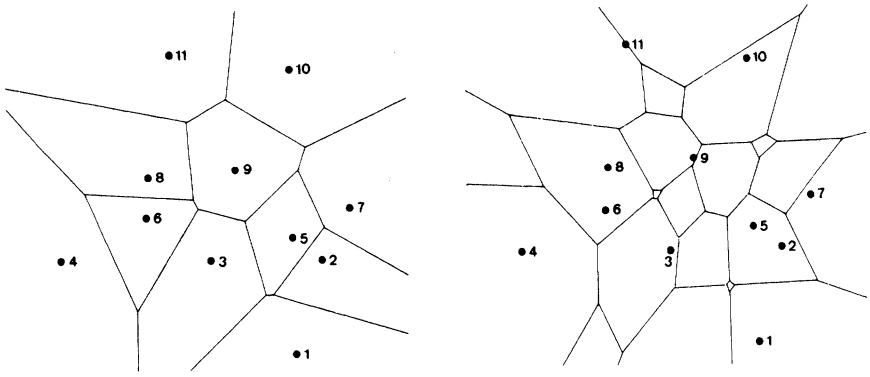


Fig. 1. First- and second-order Voronoi diagrams.

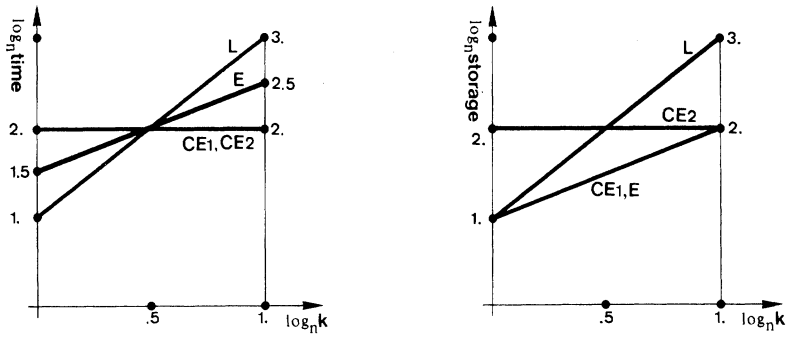


Fig. 2. The complexity of constructing kth-order Voronoi diagrams.

tion III puts this geometric background to use by describing a new algorithm for constructing $\mathcal{V}_k(S)$. Section IV exploits extra storage for speeding up the construction of $\mathcal{V}_k(S)$. Finally, Section V discusses the contributions of the paper and mentions some extensions of the methods described.

II. The Geometric Backdrop

First, we introduce a geometric transform $\mathcal{E}$ to embed kth-order Voronoi diagrams in three-dimensional Euclidean space E^3:

$\mathcal{E}$ maps a site $s = (\sigma_1, \sigma_2)$ in E^2 to the plane $\mathcal{E}(s)$ given by the relation $x_3 = 2\sigma_1 x_1 + 2\sigma_2 x_2 - (\sigma_1^2 + \sigma_2^2)$.

Intuitively, $\mathcal{E}(s)$ is the plane tangent to the paraboloid $U{:}x_3 = x_1^2 + x_2^2$ at the point obtained by projecting s onto U vertically. Straightforward analytic calculations show the following result.

Observation 2.1: Let $s = (\sigma_1, \sigma_2)$ be a site and $p = (\pi_1, \pi_2)$ a point in E^2; write $\pi_{3,U} = \pi_1^2 + \pi_2^2$ and $\pi_{3,s} = 2\sigma_1\pi_1 + 2\sigma_2\pi_2 - (\sigma_1^2 + \sigma_2^2)$ for the x_3 coordinates of the vertical projections of p onto U and $\mathcal{E}(s)$, respectively. Then $d^2(p, s) = \pi_{3,U} - \pi_{3,s}$.

To measure the distance between site s and point p we can thus take the root of the vertical distance between the vertical projections of p onto U and onto $\mathcal{E}(s)$. Now let S be a set of n sites in E^2; identify E^2 with the plane $x_3 = 0$ in E^3 and define $H = \mathcal{E}(S)$, that is, $H = \{h | h = \mathcal{E}(s), s \in S\}$. An immediate consequence of Observation 2.1 gives an intuitively appealing picture that will help us to understand kth-order Voronoi diagrams.

For a point $p = (\pi_1, \pi_2)$ in E^2 let $I_p = (i_1, i_2, \cdots, i_n)$ be a sequence of indexes such that $d(p, s_{i_l}) \leq d(p, s_{i_m})$ if $l < m$. Then the vertical line L_p through the point $(\pi_1, \pi_2, 0)$ does not intersect plane $\mathcal{E}(s_{i_l})$ below plane $\mathcal{E}(s_{i_m})$.

We develop this idea more formally. By construction, any plane $h \in H$ is nonvertical, that is, h intersects the x_3 axis in a unique point. Thus, h can be described by a relation of the form $x_3 = \eta_1 x_1 + \eta_2 x_2 + \eta_3$. Plane h bounds two half-spaces, namely

$$h^+ : x_3 > \eta_1 x_1 + \eta_2 x_2 + \eta_3 \text{ and } h^- : x_3 < \eta_1 x_1 + \eta_2 x_2 + \eta_3.$$

A point x is said to be *above, on,* or *below* plane h if $x \in h^+$, $x \in h$, or $x \in h^-$, respectively. We let $a(x)$, $o(x)$, and $b(x)$ denote the numbers of planes in H with point x below, on, and above, respectively. Obviously, $a(x) + o(x) + b(x) = n$, for any point $x \in E^3$.

The set of planes, H, defines a cell complex in E^3 called the *arrangement* $\mathcal{A}(H)$ of H whose *faces* are equivalence classes of the following equivalence relation: two points x and y in E^3 are *equivalent* if, for every plane $h \in H$, h either contains both x and y or the two points lie both above or both below h. Obviously, the functions a, o, and b are invariant over all points of a single face which allows us to extend them to faces of $\mathcal{A}(H)$ in the natural way. A face f is called a *cell, facet, edge,* or *vertex* depending on whether it is three-, two-, one-, or zero-dimensional. Obviously, we have $o(f) = 0$ for each cell f and $o(f) = 1$ for each facet f. Furthermore, $o(f) = 2$ for each edge f since a line that avoids U allows exactly two planes that contain it and are tangent to U. It follows that $o(f) \geq 3$ if and only if f is a vertex.

By Observation 2.1, we have the following relationship between a set of sites in E^2 and the corresponding arrangement of planes in E^3.

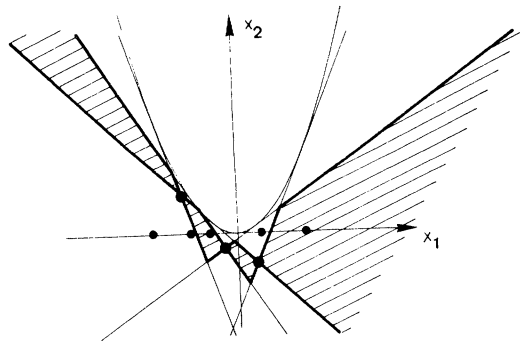


Fig. 3. Illustration of the one-to-one-correspondence.

Observation 2.2: Let S be a set of n sites in E^2, define $H = \mathcal{E}(S)$, and let k be an integer number with $1 \leq k \leq n-1$. There exists a one-to-one correspondence between

1) the regions in $\mathcal{V}_k(S)$ and the cells c in $\mathcal{A}(H)$ with $a(c) = k$,

2) the edges in $\mathcal{V}_k(S)$ and the edges e in $\mathcal{A}(H)$ with $a(e) = k-1$, and

3) the vertices in $\mathcal{V}_k(S)$ and the vertices v in $\mathcal{A}(H)$ with $a(v) \leq k-1$ and $a(v) + o(v) \geq k+1$

such that f' is the vertical projection of f onto the plane $x_3 = 0$, where f' is a face of $\mathcal{V}_k(S)$ and f is the corresponding face of $\mathcal{A}(H)$.

Fig. 3 illustrates Observation 2.2 in one dimension lower: S is a set of five sites in E^1 identified with the line $x_2 = 0$ in E^2. The transform $\mathcal{E}$ maps a site $s = (\sigma_1)$ to the line tangent to the paraboloid $x_2 = x_1^2$ at point (σ_1, σ_1^2). All cells c with $a(c) = 2$ are shaded in Fig. 3.

The algorithms in Section III construct $\mathcal{V}_k(S)$ by computing the kth *skeleton* $\mathcal{S}_k(H)$ of H that consists of all points $x \in E^3$ with

$$a(x) \leq k-1 \text{ and } a(x) + o(x) \geq k+1.$$

Notice that $\mathcal{S}_k(H)$ consists only of edges and vertices of $\mathcal{A}(H)$ and that the vertical projection of $\mathcal{S}_k(H)$ onto the plane $x_3 = 0$ yields exactly the edges and vertices, and thus also the regions, of $\mathcal{V}_k(S)$ (see Observation 2.2). In order to reduce the dimensionality of the problem from three to two, the parts of $\mathcal{S}_k(H)$ in the various planes of H are computed separately. Thus, to construct $\mathcal{S}_k(H)$ in pieces, the projective view of the arrangement defined by the planes in H turns out to be more convenient than the Euclidean interpretation.

The faces of the *projective arrangement* $\mathcal{A}^*(H)$ of H are defined as the equivalence classes of the following equivalence relation: points x and y in E^3 are *equivalent* if either

1) x is above, on, or below h if and only if y, is respectively above, on, or below h, for all planes $h \in H$, or

2) x is above, on, or below h if and only if y is, respectively below, on, or above h, for all planes $h \in H$.

With this definition, the *projective* kth *skeleton*

$$\mathcal{S}_k^*(H) = \mathcal{S}_k(H) \cup \mathcal{S}_{n-k}(H)$$

of H is connected in the three-dimensional projective space. The main difference between the projective and the Euclidean interpretation of the three-dimensional arrangement is that in the projective interpretation opposite pairs of (in the Euclidean sense) unbounded faces are thought of as one face. This interpretation will turn out to be natural when we map the arrangement into dual space.

For any plane $h \in H$, we define

$$\mathcal{S}_k^*(H, h) = \mathcal{S}_k^*(H) \cap h,$$

the *h part of* $\mathcal{S}_k^*(H)$, and

$$\mathcal{A}(H, h) = \mathcal{A}(H) \cap h,$$

the *subarrangement of h*. All edges in $\mathcal{A}(H, h)$ are also edges in $\mathcal{A}(H)$, and an edge e in $\mathcal{A}(H, h)$ is contained in $\mathcal{S}_k^*(H)$ if and only if $a(e) = k-1$ or $a(e) = n-k-1$. Fig. 4 shows the vertical projections of $\mathcal{A}(H, h)$ and $\mathcal{S}_2^*(H)$, for the set H of planes corresponding to the 11 sites shown in Fig. 1; $h = \mathcal{E}(9)$ in Fig. 4(a) and $h = \mathcal{E}(11)$ in Fig. 4(b).

The value $a(p)$ of a point p in h can be expressed in terms of the following distance function δ. Define $h = \mathcal{E}(s)$ and let s_s be the vertical projection of s onto h. For any point $p \in h$, we let $\delta(p)$ designate the number of lines in $\{l | l = h \cap g, g \in H - \{h\}\}$ that intersect the relatively open line segment with endpoints p and s_s. Function $\delta(p)$ is invariant over all points of an edge or facet in h; thus, δ can be extended to faces in h in the natural way. Notice that $\delta(e) = 1$ or $\delta(e) = 8$ holds for all solid edges shown in Fig. 4; these edges indicate the edges of $\mathcal{S}_2^*(H)$.

Lemma 2.3: Let $H = \mathcal{E}(S)$, for a set S of n sites in E^2, let h be a plane in H, and let p be a point on h. Then $a(p) = \delta(p)$.

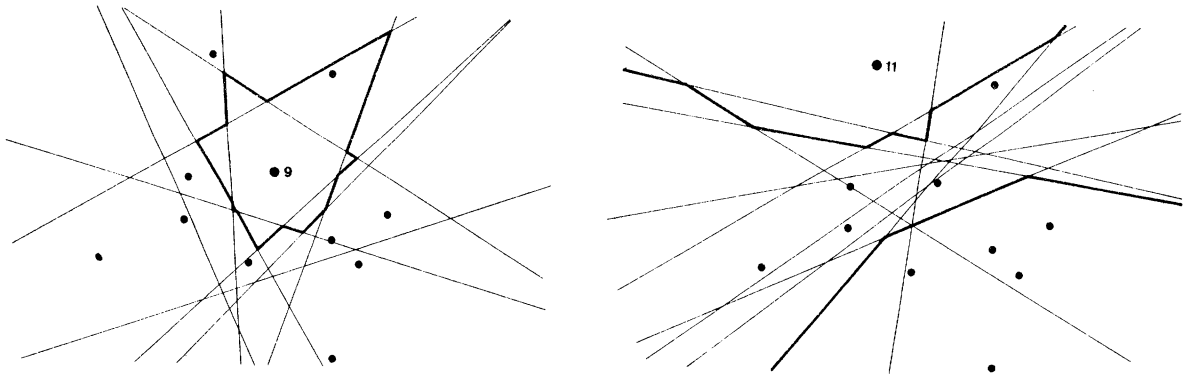
Proof: Let $h = \mathcal{E}(s)$, and let s_s be the vertical projection of s onto h (or, equivalently, onto U). By definition of $\mathcal{E}$, s_s is above all planes in $H - \{h\}$. Let g be a plane in $H - \{h\}$ and define $l = h \cap g$. A point $p \in h$ lies above g if and only if p lies on the same side of l as s_s, and $p \in h$ lies below g if and only if p and s_s lie on different sides of l. □

The vertical projection of $\mathcal{A}(H, h)$ onto $x_3 = 0$ is a two-dimensional arrangement $\mathcal{A}(G)$ in E^2, alternatively defined as follows. Let S and s be such that $H = \mathcal{E}(S)$ and $h = \mathcal{E}(s)$. For any site $t \in S - \{s\}$ G contains the perpendicular bisector of s and t. So if $|S| = n$, then $|G| = n-1$ (see Fig. 4). Arrangement $\mathcal{A}(G)$ will be used to construct the h part of $\mathcal{S}_k^*(H)$. The remainder of this section describes a transformation of the lines in G to points in the plane. The obtained point set will be dual to G.

Given an origin O in E^2, any line g not containing O can be written as $\gamma_1 x_1 + \gamma_2 x_2 = 1$.

The *central duality* $\mathcal{C}$ maps g to the point $\mathcal{C}(g) = (\gamma_1, \gamma_2)$, and vice versa, that is, it maps a point $p = (\pi_1, \pi_2)$ different from O to the line $\mathcal{C}(p): \pi_1 x_1 + \pi_2 x_2 = 1$.

Let q be the unique point of line g closest to O. Then O, q, and $\mathcal{C}(g)$ are collinear and we have $d(O, q) = 1/d(O, \mathcal{C}(g))$. We write g^{+O} for the open half-plane bounded by g that contains O, and we write g^{-O} for the open half-plane bounded by g that does not contain O. The fundamental property of $\mathcal{C}$ presented below implies the more complicated relations described in Lemma 2.5 which will be essential in the algorithmic sections of this paper.



(a) (b)

Fig. 4. Projections of h-parts of $\mathcal{S}_2^*(H)$.

Observation 2.4: Let g be a line in E^2 that does not contain O and let $p \neq O$ be a point in E^2. Then p belongs to g^{+O}, g, or g^{-O} if and only if point $\mathcal{C}(g)$ lies in $\mathcal{C}(p)^{+O}$, $\mathcal{C}(g)$, or $\mathcal{C}(g)^{-O}$, respectively.

There is a useful relationship between the edges of $\mathcal{S}_k^*(H)$ and certain collections of lines dissecting $\mathcal{C}(G)$, the set of points $\mathcal{C}(g)$, $g \in G$. Note that $\mathcal{E}(O)$ is the plane defined by the relation $x_3 = 0$.

Lemma 2.5: Let S be a set of n sites in E^2, O be a site in S, and let G be the set of perpendicular bisectors defined by O and all sites in $S - \{O\}$. As usual, H and h denote $\mathcal{E}(S)$ and $\mathcal{E}(O)$, respectively; note that $\mathcal{A}(G) = \mathcal{A}(H, h)$. Define $P = \mathcal{C}(G)$ and let p be any point in E^2.

a) $|P \cap \mathcal{C}(p)^{-O}| = \delta(p) (= a(p))$.

b) Point p is contained in an edge of $\mathcal{S}_k(H, h)$ if and only if $|P \cap \mathcal{C}(p)^{-O}| = k - 1$ and $|P \cap \mathcal{C}(p)| = 1$.

c) Point p is a vertex of $\mathcal{S}_k(H, h)$ if and only if $|P \cap \mathcal{C}(p)| \geq 2$, $|P \cap \mathcal{C}(p)^{-O}| \leq k - 1$, and $|P \cap \mathcal{C}(p)| + |P \cap \mathcal{C}(p)^{-O}| \geq k$.

d) Point p is contained in an edge of $\mathcal{S}_k^*(H, h)$ if and only if $|P \cap \mathcal{C}(p)| = 1$ and $|P \cap \mathcal{C}(p)^{-O}|$ or $|P \cap \mathcal{C}(p)^{+O}|$ is equal to $k - 1$.

e) Point p is a vertex of $\mathcal{S}_k^*(H, h)$ if and only if $|P \cap \mathcal{C}(p)| \geq 2$, $|P \cap \mathcal{C}(p)^{-O}| \leq k - 1$, and $|P \cap \mathcal{C}(p)^{+O}| \leq n - k - 1$, or vice versa, that is, $|P \cap \mathcal{C}(p)^{-O}| \leq n - k - 1$, and $|P \cap \mathcal{C}(p)^{+O}| \leq k - 1$.

Proof: Assertion a) is a direct consequence of Observation 2.4 and the definition of δ. b) and c) follow from a) and Observation 2.2. Notice that the lower bound, k, for $|P \cap \mathcal{C}(p)| + |P \cap \mathcal{C}(p)^{-O}|$ in c) differs by one from the lower bound, $k - 1$, for $a(v) + o(v)$ in Observation 2.2 c). This can be explained by the fact that point p belongs to plane h which is counted in Observation 2.2 but neglected by term $|P \cap \mathcal{C}(p)|$ in c). Finally, d) and e) follow from b), c), and the definition of $\mathcal{S}_k^*(H)$ as the union of $\mathcal{S}_k(H)$ and $\mathcal{S}_{n-k}(H)$. □

Section III develops an algorithm for constructing all edges and vertices characterized by Lemma 2.5 d) and e), and hence for computing $\mathcal{S}_k^*(H, h)$.

III. The Algorithm

Let S be a set of n sites in E^2 and define $H = \mathcal{E}(S)$, as usual. The kth-order Voronoi diagram of S can be constructed as follows.

1) Compute $\mathcal{S}_k^*(H)$ by constructing its h-parts for all $h \in H$.

2) Compute $\mathcal{S}_k(H)$ from $\mathcal{S}_k^*(H)$ by eliminating all edges and vertices of $\mathcal{S}_{n-k}(H)$, unless they also belong to $\mathcal{S}_k(H)$.

3) Project $\mathcal{S}_k(H)$ vertically onto the plane $x_3 = 0$; this yields $\mathcal{V}_k(S)$.

To distinguish edges and vertices of $\mathcal{S}_k(H)$ from those of $\mathcal{S}_{n-k}(H)$ we follow the connected sequence of edges of $\mathcal{S}_k^*(H)$. Whenever we pass the point at infinity, we switch from $\mathcal{S}_k(H)$ to $\mathcal{S}_{n-k}(H)$ or vice versa. It is thus sufficient to determine the status of one particular vertex of $\mathcal{S}_k^*(H)$ in order to carry out the classification in time linear in the number of edges encountered. The intriguing part of the algorithm is the construction of an h part of $\mathcal{S}_k^*(H)$, $h \in H$. To this end, we exploit Lemma 2.5 which reduces the task to a point set problem in E^2.

Let P be a set of $n - 1$ points and let g be a directed line in E^2. We define $l(g)$, $o(g)$, and $r(g)$ as the numbers of points in P that lie to the left of, on, and to the right of g, respectively. Call g a k*-line if $r(g) \leq k - 1$ and $o(g) + r(g) \geq k$ and, for some value of k, compute all k*-lines of P [compare Lemma 2.5 d) and e)].

We have the following result.

Observation 3.1: Let P be a set of $n - 1$ points in E^2 and let k be an integer number with $1 \leq k \leq n - 1$. For any angle α in $[0, 2\pi)$, there is a unique k*-line $g(\alpha)$ that can be obtained by rotating the x_1-axis through α radians.

To compute all k*-lines of P efficiently, we use a result of [14] about maintaining two-dimensional convex hulls through a sequence of insertions and deletions of points. The same technique has be used in [5] and [9] to construct k-hulls and k-belts which are structures that bear close relationship with what we are after. We include the result of [14] and its application to our problem since we will make use of a specific property of the technique in Section IV.

Proposition 3.2: There is a data structure that maintains the convex hull of a set of at most n points in E^2 in $O(\log^2 n)$ time per insertion and deletion of a point. The structure requires $O(n)$ storage and $O(n \log n)$ time for its construction ($O(n)$ time if the points are given sorted in x_1 direction); for any point p outside the convex hull, the two lines passing through p that are tangent to the convex hull can be computed in $O(\log n)$ time.

The algorithm for constructing all k*-lines follows the three steps below. It uses the data structure of [14] to perform a sequence of operations, where each operation is either the insertion of a point, the deletion of a point, or the determination of a tangent line passing through a point.

1) Sort the points in P from left to right (and bottom-up if points share the same x_1 coordinate).

2) Take the vertical line that passes through the kth point $p \in P$ and is directed downward as the initial position of a directed line g which will rotate through all k*-lines. Call p the current anchor (that is, point of contact) of g, and construct the convex hulls of sets R and L containing the first $k-1$ and last $n-k-1$ points of P, respectively.

3) Repeat the following steps until g reaches its initial position again (for convenience, we treat only nondegenerate cases here):

Rotate g in counterclockwise direction around its anchor p until it hits a point q of R or of L. Assume without loss of generality that $q \in R$. Note that g is now tangent to the convex hull of R. Call q the new current anchor, delete q from R, and insert p into R.

Each time g encounters a point of R or L, it is mapped to a vertex of $\mathcal{S}_k^*(H, h)$. Between two consecutive vertices, the loci of g are mapped to the points of the edge of $\mathcal{S}_k^*(H, h)$ that connects the two vertices. This "edge" is a projective edge, that is, it is either the Euclidean line segment that connects the two vertices or it is the line through the two vertices minus this line segment. The first case occurs when line g does not rotate through a vertical position and the second case occurs if it does rotate through a vertical position.

In a degenerate case, g can encounter several points of R and L at the same time. Say, g hits points $r_1, r_2, \cdots, r_i$ and $l_1, l_2, \cdots, l_j$, where $i \geq 0$, $j \geq 0$, and $i + j \geq 1$, such that

$$r_1, r_2, \cdots, r_i, p, l_1, l_2, \cdots, l_j = q_1, q_2, \cdots, q_{i+j+1}$$

appear in sorted order on g. Then q_{j+1} is the new anchor, and the other points are deleted and inserted accordingly. This case can be handled in $O(\log^2 n)$ time per point and is thus no more expensive than if the points are encountered in sequence. In fact, the additional effort needed in a degenerate case is paid off by the loss in the number of edges caused. Since an edge costs $O(\log^2 n)$ time in the nondegenerate case, this yields the following result.

Theorem 3.3: Let S be a set of n sites in E^2. There is an algorithm that constructs $\mathcal{V}_k(S)$ in $O(n^2 \log n + k(n-k)\log^2 n)$ time and $O(k(n-k))$ storage.

At this point, it is worthwhile to mention that [5] describes a technique that allows us to maintain the two-dimensional convex hull in $O(\log^2 k)$ time instead of in $O(\log^2 n)$ time per operation. This leads to a slight improvement of Theorem 3.3 if k is very small. In these cases, however, our result is inferior to the result of [13] anyway. This technique of [5] will not be applicable in Section IV since it entails a preprocessing cost of $O(n^2 \log n)$ time.

IV. Trading Storage for Speed

The $O(n^2 \log n)$ time needed for setting up $2n$ convex hulls can be improved to $O(n^2)$ if all necessary sorting is done in respective linear time. For a set S of n points in E^2, we achieve the desired speedup in two steps.

1) We show that $O(n^2)$ time suffices to construct, for each site $s \in S$, the order of the remaining sites sorted around s.

2) The central duality $\mathcal{C}$ of Section II is replaced by another dual transform that yields point sets with the ordering induced by the corresponding order in 1).

In both steps, we make use of the dual transform $\mathcal{D}$ that maps a point $p = (\pi_1, \pi_2)$ in E^2 to the nonvertical line $\mathcal{D}(p): x_2 = 2\pi_1 x_1 - \pi_2$, and vice versa, that is, it maps a nonvertical line $g: x_2 = \gamma_1 x_1 + \gamma_2$ to the point $\mathcal{D}(g) = (\gamma_1/2, -\gamma_2)$. Since g is assumed to be nonvertical, we can define the half-planes

$$g^+ : x_2 > \gamma_1 x_1 + \gamma_2 \text{ and } g^- : x_2 < \gamma_1 x_1 + \gamma_2.$$

The fundamental property of transform $\mathcal{D}$ is expressed in the following result.

Observation 4.1: Let p be a point and g be a nonvertical line in E^2. Then p belongs to g^+, g, or g^- if and only if point $\mathcal{D}(g)$ belongs to $\mathcal{D}(p)^+$, $\mathcal{D}(p)$, or $\mathcal{D}(p)^-$, respectively.

Compare Observation 4.1 to Observation 2.4. Step 1) is now achieved by constructing the two-dimensional arrangement defined by the lines in $\mathcal{D}(S)$. This subdivision of E^2 can be computed in $O(n^2)$ time and storage (see [4], [8]). The order of the sites in $S - \{s\}$ around s corresponds to the order of intersections of $\mathcal{D}(s)$ with the other lines. These sequences can be derived in $O(n^2)$ time from the representation of the arrangement described in [4], [8].

Let S be a set of n sites in E^2, let s be a site in S, and let G be the set of perpendicular bisectors defined by s and all sites in $S - \{s\}$. The order of sites around s corresponds to the left to right order of the points in $P = \mathcal{D}(G)$. More specifically, the left to right order of P can be obtained by merging the counterclockwise order of the sites above the horizontal line through s with the clockwise order of the sites below this line. For g a nonvertical line, we let g^{pos} be the open double wedge defined by g and line $\mathcal{D}(s)$ that does not contain the vertical line through $g \cap \mathcal{D}(s)$, and we let g^{neg} denote the other open double wedge. If g and $\mathcal{D}(s)$ are parallel, then g^{pos} is the open stripe between the two lines and g^{neg} is the interior of the complement of this stripe. We have the following result which is similar to Lemma 2.5.

Observation 4.2: Let S be a set of n sites in E^2, let s be a site in S, let G be the set of perpendicular bisectors of s and all sites in $S - \{s\}$, and define $P = \mathcal{D}(G)$. For any point $p \in E^2$, we have $|P \cap \mathcal{D}(p)^{\text{neg}}| = \delta(p)$.

Statements analogous to Lemma 2.5 a)–e) follow from Observation 4.2. Furthermore, the algorithmic techniques used in Section III carry over to compute all lines g such that $|P \cap \mathcal{D}(p)^{\text{neg}}| \leq k-1$ and $|P \cap \mathcal{D}(p)| + |P \cap \mathcal{D}(p)^{\text{neg}}| \geq k$. This implies the main result of this section.

Theorem 4.3: Let S be a set of n sites in E^2. There is an algorithm that constructs $\mathcal{V}_k(S)$ in $O(n^2 + k(n-k)\log^2 n)$ time and $O(n^2)$ storage.

V. Discussion and Extensions

Using the technique of geometric transformation and two-dimensional convex hull maintenance, this paper presents two

new algorithms for constructing kth-order Voronoi diagrams in the plane. For n sites, the first algorithm takes $O(n^2 \log n + k(n-k)\log^2 n)$ time and $O(k(n-k))$ storage; the second algorithm runs in $O(n^2 + k(n-k)\log^2 n)$ time and $O(n^2)$ storage. This improves upon the time complexity of the methods in [13] and [7] whenever $k = \Omega(\sqrt{n})$ (see Fig. 2). The details of the developments of this paper are related to techniques presented in [5] and in [9].

The methods of this paper are applicable to a slightly broader class of problems than stated. Below, we briefly describe two additional geometric structures that can be built using our algorithmic methods.

1) The *kth-degree Voronoi diagram of* a set S of n sites in E^2 associates each site s with the region of points p such that s is the k-closest site of p (see [8]). The complexity results stated as Theorem 3.3 and 4.3 apply since the diagram consists of exactly all edges in $\mathcal{V}_{k-1}(S)$ and $\mathcal{V}_k(S)$.

2) Let S be a set of n sites in E^3. The transformation used in this paper for two-dimensional point sets can be generalized to three dimensions and can thus be applied to S. The convex hull maintenance in E^3 can be done by a data structure that takes $O(n)$ storage, $O(n \log n)$ time for construction, and $O(\sqrt{n} \log n)$ time per update and query (see [7]). If $c(n)$ denotes the maximum number of edges of the kth-order Voronoi diagram of n sites in E^3, then the skeleton of edges of $\mathcal{V}_k(S)$ can be constructed in $O(n^2 \log n + c(n)\sqrt{n} \log n)$ time and $O(c(n))$ storage.

Acknowledgment

We would like to thank two anonymous referees for their constructive criticism.

References

[1] A. Bowyer, "Computing Dirichlet tessellations," *Comput. J.*, vol. 24, pp. 162–166, 1981.

[2] W. Brostow, J.-P. Dussault, and B. L. Fox, "Construction of Voronoi polyhedra," *J. Comput. Phys.*, vol. 29, pp. 81–92, 1978.

[3] K. Q. Brown, "Voronoi diagrams from convex hulls," *Inform. Process. Lett.*, vol. 9, pp. 223–228, 1979.

[4] B. M. Chazelle, L. J. Guibas, and D. T. Lee, "The power of geometric duality," *BIT*, vol. 25, pp. 76–90, 1985.

[5] R. Cole, M. Sharir, and C. K. Yap, "On k-hulls and related problems," *SIAM J. Comput.*, vol. 16, pp. 61–77, 1987.

[6] P. G. L. Dirichlet, Über die Reduktion der positiven quadratischen Formen mit drei unbestimmten ganzen Zhalen," *J. Reine Angew. Math.*, vol. 40, pp. 209–227, 1850.

[7] H. Edelsbrunner, "Edge-skeletons in arrangements with applications," *Algorithmica*, vol. 1, pp. 93–109, 1986.

[8] H. Edelsbrunner, J. O'Rourke, and R. Seidel, "Constructing arrangements of lines and hyperplanes with applications," *SIAM J. Comput.*, vol. 15, pp. 341–363, 1986.

[9] H. Edelsbrunner and E. Welzl, "Constructing belts in two-dimensional arrangements with applications," *SIAM J. Comput.*, vol. 15, pp. 271–284, 1986.

[10] J. Fairfield, "Contoured shape generation: Forms that people see in dot patterns," in *Proc. IEEE Conf. Syst., Man, Cybern.* 1979, pp. 60–64.

[11] S. Fortune, "A sweep line algorithm for Voronoi diagrams," in *Proc. Annu. 2nd ACM Symp. Comput. Geom.*, 1986, pp. 313–322.

[12] I. Hodder and C. Orton, *Spatial Analysis in Archeology.* Cambridge, 1976.

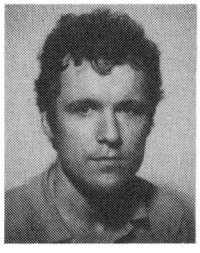
[13] D. T. Lee, "On k-nearest neighbor Voronoi diagrams in the plane," *IEEE Trans. Comput.* vol. C-31, pp. 478–487, 1982.

[14] M. H. Overmars and J. van Leeuwen, "Maintenance of configurations in the plane," *J. Comput. Syst. Sci.*, vol. 23, pp. 166–204, 1981.

[15] F. P. Preparata and M. I. Shamos, *Computational Geometry—An Introduction.* New York: Springer-Verlag, 1985.

[16] M. I. Shamos and D. Hoey, "Closest-point problems," in *Proc. 16th Annu. IEEE Symp. Found. Comput. Sci.*, 1975, pp. 151–162.

[17] G. Voronoi, "Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Premier Mémoire: Sur quelques propriétés des formes quadratiques positives parfaites," *J. Reine Angew. Math.* vol. 133, pp. 97–178, 1907. "Deuxième Mémoire: Recherches sur les paralléllo̊edres primitifs," *J. Reine Angew. Math.*, vol. 133, pp. 198–287, 1908.



Bernard Chazelle received the M.S. and Ph.D. degrees in computer science from Yale University, New Haven, CT, in 1978 and 1980, respectively.

From 1980 to 1982 he was a Research Associate in the Department of Computer Science at Carnegie-Mellon University, Pittsburgh, PA. From 1982 to 1985 he was an Assistant Professor and then an Associate Professor in the Department of Computer Science at Brown University, Providence, RI. In 1985 and 1986 he was a Visiting Professor at Ecole Normale Supérieure, Paris, France. Since September 1986 he has been an Associate Professor at Princeton University, Princeton, NJ. His research interests include the analysis and design of algorithms, computational geometry, computer graphics, and parallel computation.

Dr. Chazelle is a member of the Association for Computing Machinery, the Society for Industrial and Applied Mathematics, and the European Association of Theoretical Computer Science.

Herbert Edelsbrunner was born in Unterpremstätten, Austria. He received the Diplom Ingenieur (Dipl.Ing.) and Technical Doctor degrees from the Technical University of Graz, Graz, Austria, in 1980 and 1982, respectively.

From 1982 through 1985, he was with the Department of Information Processing at the Technical University of Graz. Since November 1985, he has been Assistant Professor in the Department of Computer Science, University of Illinois, Urbana. His current research interests are in the areas of data structures, algorithms, computational geometry, combinatorics, and discrete geometry. He is author of many research articles and of the book *Algorithms in Combinatorial Geometry*.

Dr. Edelsbrunner is a member of the European Association of Theoretical Computer Science.