CHAPTER XY

Voronoi Diagrams

Franz Aurenhammer

Institut für Grundlagen der Informationsverarbeitung, Technische Universität Graz
Klosterwiesgasse 32/2, A-8010 Graz, Austria

Rolf Klein

Praktische Informatik VI, FernUniversität Hagen
Elberfelder Straße 95, D-58084 Hagen, Germany

Contents

November 19, 1996 – Draft version; Typeset by LaTeX

Partially supported by the Deutsche Forschungsgemeinschaft, grant Kl 655 2-2.

Mathematics Series – Style files
Created by F.D. Mesman and W.j. Maas

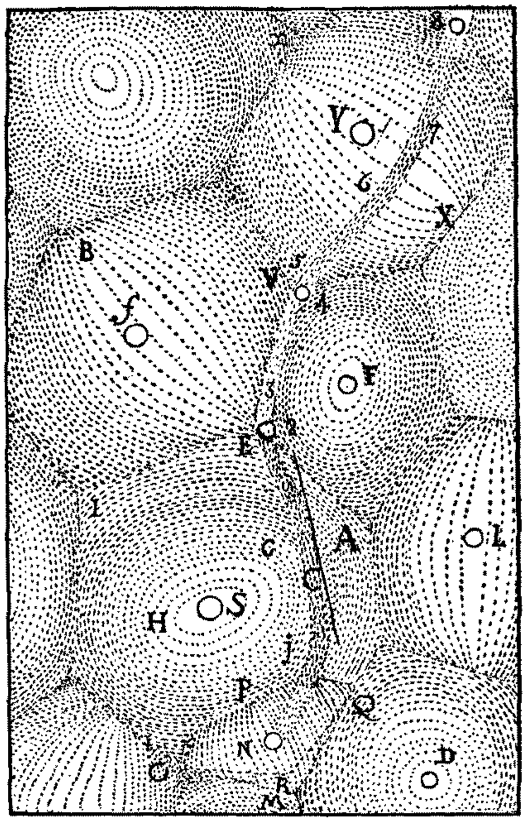


Figure 1. Descartes' decomposition of space into vortices.

1. Introduction

The topic of this chapter, Voronoi diagrams, differs from other areas of computational geometry, in that its origin dates back to the 17th century. In his book on the principles of philosophy [87], R. Descartes claims that the solar system consists of vortices. His illustrations show a decomposition of space into convex regions, each consisting of matter revolving round one of the fixed stars; see fig. 1.

Even though Descartes has not explicitly defined the extension of these regions, the underlying idea seems to be the following. Let a space M, and a set S of *sites* p in M be given, together with a notion of the *influence* a site p exerts on a point x of M. Then the *region* of p consists of all points x for which the influence of p is the strongest, over all $s \in S$.

This concept has independently emerged, and proven useful, in various fields of science. Different names particular to the respective field have been used, such as *medial axis transform* in biology and physiology, *Wigner-Seitz zones* in chemistry and physics, *domains of action* in crystallography, and *Thiessen polygons* in meteorology and geography. The mathematicians Dirichlet [95] and Voronoi [251, 250] were the first to formally introduce this concept. They used it for the study of quadratic forms; here the sites are integer lattice points, and influence is measured by the Euclidean distance. The resulting structure has been called *Dirichlet tessellation* or *Voronoi diagram*, which has become its standard name today.

Voronoi [251] was the first to consider the *dual* of this structure, where any two point sites are connected whose regions have a boundary in common. Later, Delaunay [86] obtained the same by defining that two point sites are connected iff (i. e. if and only if) they lie on a circle whose interior contains no point of S. After him, the dual of the Voronoi diagram has been denoted *Delaunay tessellation* or *Delaunay triangulation*.

Besides its applications in other fields of science, the Voronoi diagram and its dual can be used for solving numerous, and surprisingly different, geometric problems. Moreover, these structures are very appealing, and a lot of research has been devoted to their study (about one out of 16 papers in computational geometry), ever since Shamos and Hoey [230] introduced them to the field.

The reader interested in a complete overview over the existing literature should consult the book by Okabe et al. [208] who list more than 600 papers, and the surveys by Aurenhammer [27], Bernal [39], and Fortune [124]. Also, chapters 5 and 6 of Preparata and Shamos [213] and chapter 13 of Edelsbrunner [104] could be consulted. Within one chapter, we cannot review all known results and applications. Instead, we are trying to highlight the intrinsic potential of Voronoi diagrams, that lies in its structural properties, in the existence of efficient algorithms for its construction, and in its adaptability.

We start in section 2 with a simple case: the Voronoi diagram and the Delaunay triangulation of n points in the plane, under the Euclidean distance. We state elementary structural properties that follow directly from the definitions. Further properties will be revealed in section 3, where different algorithmic schemes for computing these structures are presented. In section 4 we complete our presentation of the classical two-dimensional case, and turn to generalizations. Next, in section 5, important geometric applications of the Voronoi diagram and the Delaunay triangulation are discussed. The reader who is interested mainly in these applications can proceed directly to section 5, after section 2. Finally, section 6 concludes the chapter and mentions some open problems.

2. Definitions and elementary properties

Throughout this section we denote by S a set of $n \geqslant 3$ point sites $p, q, r, \ldots$ in the plane. For points $p = (p_1, p_2)$ and $x = (x_1, x_2)$ let $d(p, x) = \sqrt{(p_1 - x_1)^2 + (p_2 - x_2)^2}$ denote their Euclidean distance. By $\overline{pq}$ we denote the line segment from p to q. The closure of a set A will be denoted by $\overline{A}$.

Definition 2.1. For $p, q \in S$ let

$$B(p,q) = \{x \mid d(p,x) = d(q,x)\}$$

be the *bisector* of p and q. $B(p,q)$ is the perpendicular line through the center of the line segment $\overline{pq}$. It separates the halfplane

$$D(p,q) = \{x \mid d(p,x) < d(q,x)\}$$

containing p from the halfplane $D(q,p)$ containing q. We call

$$\mathrm{VR}(p,S) = \bigcap_{q \in S, q \neq p} D(p,q)$$

the *Voronoi region* of p with respect to S. Finally, the *Voronoi diagram* of S is defined by

$$V(S) = \bigcup_{p,q \in S, p \neq q} \overline{\mathrm{VR}(p,S)} \cap \overline{\mathrm{VR}(q,S)}.$$

By definition, each Voronoi region $\mathrm{VR}(p,S)$ is the intersection of $n-1$ open halfplanes containing the site p. Therefore, $\mathrm{VR}(p,S)$ is open and convex. Different Voronoi regions are disjoint.

The common boundary of two Voronoi regions belongs to $V(S)$ and is called a *Voronoi edge*, if it contains more than one point. If the Voronoi edge e borders the regions of p and q then $e \subset B(p,q)$ holds. Endpoints of Voronoi edges are called *Voronoi vertices*; they belong to the common boundary of three or more Voronoi regions.

There is an intuitive way of looking at the Voronoi diagram $V(S)$. Let x be an arbitrary point in the plane. We center a circle, C, at x and let its radius grow, from 0 on. At some stage the expanding circle will, for the first time, hit one or more sites of S. Now there are three different cases.

Lemma 2.1. *If the circle C expanding from x hits exactly one site, p, then x belongs to $VR(p,S)$. If C hits exactly two sites, p and q, then x is an interior point of a Voronoi edge separating the regions of p and q. If C hits three or more sites simultaneously, then x is a Voronoi vertex adjacent to those regions whose sites have been hit.*

Proof. If only site p is hit then p is the unique element of S closest to x. Consequently, $x \in D(p,r)$ holds for each site $r \in S$ with $r \neq p$. If C hits exactly p and q, then x is contained in each halfplane $D(p,r), D(q,r)$, where $r \notin \{p,q\}$, and in $B(p,q)$, the common boundary of $D(p,q)$ and $D(q,p)$. By definition 2.1, x belongs

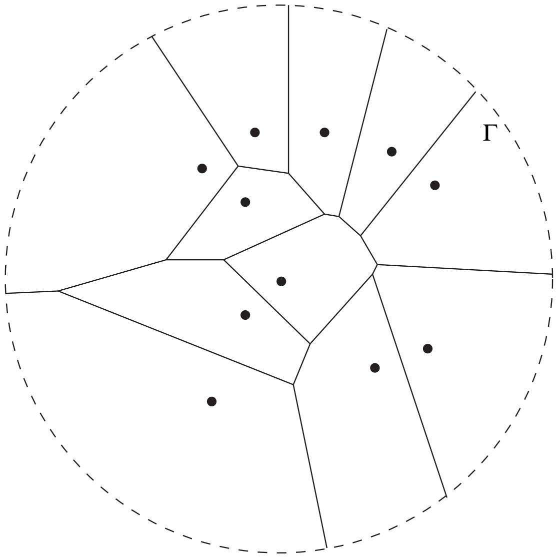


Figure 2. A Voronoi diagram of 11 points in the Euclidean plane.

to the closure of the regions of both p and q, but of no other site in S. In the third case, the argument is analogous. □

This lemma shows that the Voronoi regions form a decomposition of the plane; see fig. 2.

Conversely, if we imagine n circles expanding from the sites at the same speed, the fate of each point x of the plane is determined by those sites whose circles reach x first. This "expanding waves" view has been systematically used by Chew and Drysdale [66] and Thurston [246].

The Voronoi vertices are of degree at least three, by lemma 2.1. Vertices of degree higher than three do not occur if no four point sites are cocircular. The Voronoi diagram $V(S)$ is disconnected if all point sites are collinear; in this case it consists of parallel lines.

From the Voronoi diagram of S one can easily derive the *convex hull* of S, i. e. the boundary of the smallest convex set containing S.

Lemma 2.2. *A point p of S lies on the convex hull of S iff its Voronoi region $VR(p, S)$ is unbounded.*

Proof. The Voronoi region of p is unbounded iff there exists some point $q \in S$ such that $V(S)$ contains an unbounded piece of $B(p, q)$ as a Voronoi edge. Let $x \in B(p, q)$, and let $C(x)$ denote the circle through p and q centered at x, as shown

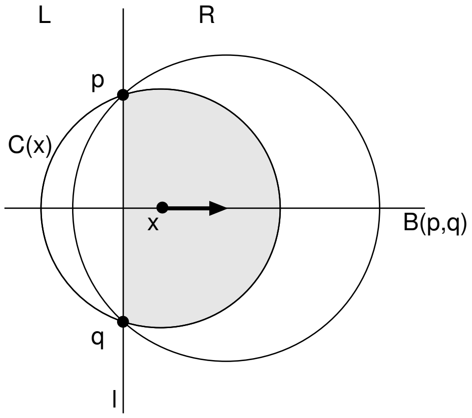


Figure 3. As x moves to the right, the intersection of circle $C(x)$ with the left halfplane shrinks, while $C(x) \cap R$ grows.

in fig. 3. Point x belongs to $V(S)$ iff $C(x)$ contains no other site. As we move x to the right along $B(p, q)$, the part of $C(x)$ contained in halfplane R keeps growing. If there is another site r in R, it will eventually be reached by $C(x)$, causing the Voronoi edge to end at x. Otherwise, all other sites of S must be contained in the closure of the left halfplane L. Then p and q both lie on the convex hull of S. □

Sometimes it is convenient to imagine a simple closed curve Γ around the "interesting" part of the Voronoi diagram, so large that it intersects only the unbounded Voronoi edges; see fig. 2. While walking along Γ, the vertices of the convex hull of S can be reported in cyclic order. After removing the halflines outside Γ, a connected embedded planar graph with $n+1$ faces results. Its faces are the n Voronoi regions and the unbounded face outside Γ. We call this graph the *finite* Voronoi diagram.

One virtue of the Voronoi diagram is its small size.

Lemma 2.3. *The Voronoi diagram $V(S)$ has $O(n)$ many edges and vertices. The average number of edges in the boundary of a Voronoi region is less than 6.*

Proof. By the Euler formula (see e. g. [129]) for planar graphs, the following relation holds for the numbers v, e, f, and c of vertices, edges, faces, and connected components.

$$v - e + f = 1 + c.$$

We apply this formula to the finite Voronoi diagram. Each vertex has at least three incident edges; by adding up we obtain $e \geqslant 3v/2$, because each edge is counted twice. Substituting this inequality together with $c = 1$ and $f = n + 1$ yields

$$v \leqslant 2n - 2 \text{ and } e \leqslant 3n - 3.$$

Adding up the numbers of edges contained in the boundaries of all $n + 1$ faces results in $2e \leqslant 6n - 6$ because each edge is again counted twice. Thus, the average number of edges in a region's boundary is bounded by $(6n - 6)/(n + 1) < 6$. The same bounds apply to $V(S)$. □

Now we turn to the Delaunay tessellation. In general, a *triangulation* of S is a planar graph with vertex set S and straight line edges, which is maximal in the sense that no further straight line edge can be added without crossing other edges.

Each triangulation of S contains the edges of the convex hull of S. Its bounded faces are triangles, due to maximality. Their number equals $2n - k - 2$, where k denotes the size of the convex hull. We call a subset of edges of a triangulation a *tessellation* of S if it contains the edges of the convex hull, and if each point of S has at least two adjacent edges.

Definition 2.2. The *Delaunay tessellation* $\mathrm{DT}(S)$ is obtained by connecting with a line segment any two points p, q of S for which a circle C exists that passes through p and q and does not contain any other site of S in its interior or boundary. The edges of $\mathrm{DT}(S)$ are called *Delaunay edges*.

The following equivalent characterization is a direct consequence of lemma 2.1.

Lemma 2.4. *Two points of S are joined by a Delaunay edge iff their Voronoi regions are edge-adjacent.*

Since each Voronoi region has at least two neighbors, at least two Delaunay edges must emanate from each point of S. By the proof of lemma 2.2, each edge of the convex hull of S is Delaunay. Finally, two Delaunay edges can only intersect at their endpoints, because they allow for circumcircles whose respective closures do not contain other sites. This shows that $\mathrm{DT}(S)$ is in fact a *tessellation* of S.

Two Voronoi regions can share at most one Voronoi edge, by convexity. Therefore, lemma 2.4 implies that $\mathrm{DT}(S)$ is the graph–theoretical *dual* of $V(S)$, realized by straight line edges.

An example is depicted in fig. 4; the Voronoi diagram $V(S)$ is drawn by solid lines, and $\mathrm{DT}(S)$ by dashed lines. Note that a Voronoi vertex (like w) need not be contained in its associated face of $\mathrm{DT}(S)$. The sites p, q, r, s are cocircular, giving rise to a Voronoi vertex v of degree 4. Consequently, its corresponding Delaunay face is bordered by four edges. This cannot happen if the points of S are in general position.

Theorem 2.1. *If no four points of S are cocircular then $DT(S)$, the dual of the Voronoi diagram $V(S)$, is a triangulation of S, called the* Delaunay triangulation. *Three points of S give rise to a Delaunay triangle iff their circumcircle does not contain a point of S in its interior.*

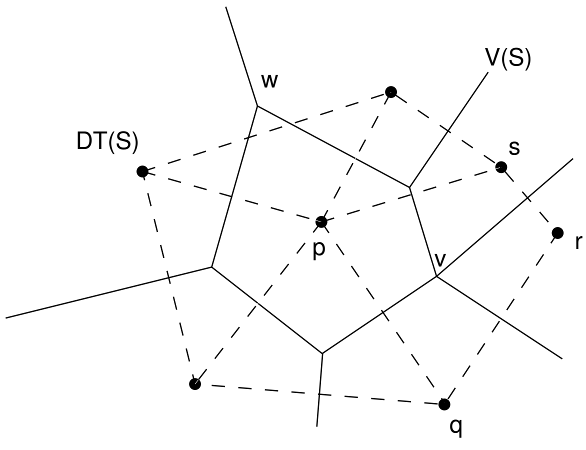


Figure 4. Voronoi diagram and Delaunay tessellation.

3. Algorithms

In this section we present several ways of computing the Voronoi diagram and its dual, the Delaunay tesselation. For simplicity, we assume of the n point sites of S that no four of them are cocircular, and that no three of them are collinear. According to theorem 2.1 we can then refer to $\mathrm{DT}(S)$ as to the Delaunay triangulation. All algorithms presented herein can be made to run without the general position assumption. Also, they can be generalized to metrics other than the Euclidean, and to sites other than points. This will be discussed in subsections 4.5 and 4.4.

Data structures well suited for working with planar graphs like the Voronoi diagram are the *doubly connected edge list*, DCEL, by Muller and Preparata [201], and the *quad edge* structure by Guibas and Stolfi [136]. In either structure, a record is associated with each edge e that stores the following information: the names of the two endpoints of e; references to the edges clockwise or counterclockwise next to e about its endpoints; finally, the names of the faces to the left and to the right of e. The space requirement of both structures is $O(n)$.

Either structure allows to efficiently traverse the edges adjacent to a given vertex, and the edges bounding a face. The quad edge structure offers the additional advantage of describing, at the same time, a planar graph and its dual, so that it can be used for constructing both the Voronoi diagram and the Delaunay triangulation. From the DCEL of $V(S)$ we can derive the set of triangles constituing the Delaunay triangulation in linear time. Conversely, from the set of all Delaunay triangles the DCEL of the Voronoi diagram can be constructed in time $O(n)$. Therefore, each algorithm for computing one of the two structures can be used for computing the other one, within $O(n)$ extra time.

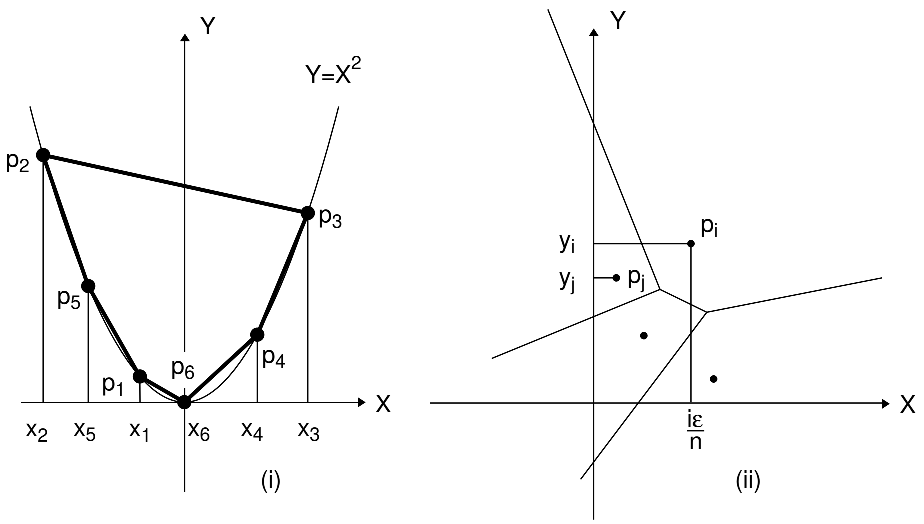


Figure 5. Proving the $\Omega(n \log n)$ lower bound for constructing the Voronoi diagram (i) by transformation from sorting, and (ii) by transformation from ε-closeness.

It is convenient to store structures describing the *finite* Voronoi diagram, as introduced before lemma 2.3, so that the convex hull of the point sites can be easily reported by traversing the bounding curve Γ; see fig. 2.

3.1. A lower bound

Before constructing the Voronoi diagram we want to establish a lower bound for its computational complexity.

Suppose that n real numbers $x_1, \ldots, x_n$ are given. From the Voronoi diagram of the point set $S = \{p_i = (x_i, {x_i}^2) \mid 1 \leqslant i \leqslant n\}$ one can derive, in linear time, the vertices of the convex hull of S, in counterclockwise order. From the leftmost point in S on, this vertex sequence contains all points p_i, sorted by increasing values of x_i; see fig. 5 (i).

This argument due to Shamos [229] shows that constructing the convex hull and, *a fortiori*, computing the Voronoi diagram, is at least as difficult as sorting n real numbers, which requires $\Theta(n \log n)$ time in the algebraic computation tree model.

However, a fine point is lost in this reduction. After sorting n points by their x-values, their convex hull can be computed in linear time [101], whereas sorting does not help in constructing the Voronoi diagram. The following result has been independently found by Djidjev and Lingas [96] and by Zhu and Mirzaian [260].

Theorem 3.1. *It takes time $\Omega(n \log n)$ to construct the Voronoi diagram of n points $p_1, \ldots, p_n$ whose x-coordinates are strictly increasing.*

Proof. By reduction from the *ε-closeness problem* which is known to be in $\Theta(n \log n)$. Let $y_1, \ldots, y_n$ be positive real numbers, and let $\varepsilon > 0$. The question is if there exist $i \neq j$ such that $|y_i - y_j| < \varepsilon$ holds. We form the sequence of points $p_i = (i\varepsilon/n, y_i), 1 \leqslant i \leqslant n$, and compute their Voronoi diagram; see fig. 5 (ii). In time $O(n)$, we can determine the Voronoi regions that are intersected by the y-axis, in bottom-up order (such techniques will be detailed in subsection 3.3.)

If, for each p_i, its projection onto the y-axis lies in the Voronoi region of p_i then the values y_i are available in sorted order, and we can easily answer the question. Otherwise, there is a point p_i whose projection lies in the region of some other point p_j. Because of

$$|y_i - y_j| \leqslant d((0, y_i), p_j) < d((0, y_i), p_i) = \frac{i\varepsilon}{n} \leqslant \varepsilon,$$

in this case the answer is positive. □

On the other hand, sorting n *arbitrary* point sites by x-coordinates is not made easier by their Voronoi diagram, as Seidel [223] has shown.

With definition 2.1 in mind one could think of computing each Voronoi region as the intersection of $n-1$ halfplanes. This would take time $\Theta(n \log n)$ per region, see [213]. In the following subsections we describe various algorithms that compute the *whole* Voronoi diagram within this time; due to theorem 3.1, these algorithms are worst-case optimal.

3.2. Incremental construction

A natural idea first studied by Green and Sibson [133] is to construct the Voronoi diagram by *incremental insertion*, i. e. to obtain $V(S)$ from $V(S \setminus \{p\})$ by inserting the site p. As the region of p can have up to $n-1$ edges, for $n = |S|$, this leads to a runtime of $O(n^2)$. Several authors fine-tuned the technique of inserting Voronoi regions, and efficient and numerically robust implementations are available nowadays; see Ohya et al. [207] and Sugihara and Iri [240]. In fact, runtimes of $O(n)$ can be expected for well distributed sets of sites.

The insertion process is, maybe, better described, and implemented in the dual environment, for the Delaunay triangulation: construct $\mathrm{DT}_i = \mathrm{DT}(\{p_1, \ldots, p_{i-1}, p_i\})$ by inserting the site p_i into DT_{i-1}. The advantage over a direct construction of V(S) is that Voronoi vertices that appear in intermediate diagrams but not in the final one need not be constructed and stored. We follow Guibas and Stolfi [136] and construct DT_i by exchanging edges, using Lawson's [175] original edge flipping procedure, until all edges invalidated by p_i have been removed.

To this end, it is useful to extend the notion of triangle to the unbounded face of the Delaunay triangulation. If $\overline{pq}$ is an edge of the convex hull of S we call

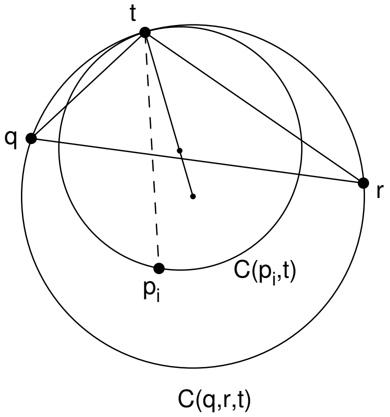


Figure 6. If triangle $T(q,r,t)$ is in conflict with p_i then former Delaunay edge $\overline{qr}$ must be replaced by $\overline{p_i t}$.

the supporting halfplane H not containing S an *infinite triangle* with edge $\overline{pq}$. Its circumcircle is H itself, the limit of all circles through p and q whose center tend to infinity within H; compare fig. 3. As a consequence, each edge of a Delaunay triangulation is now adjacent to two triangles.

Those triangles of DT_{i-1} (finite or infinite) whose circumcircles contain the new site, p_i, are said to be *in conflict* with p_i. According to theorem 2.1, they will no longer be Delaunay triangles.

Let $\overline{qr}$ be an edge of DT_{i-1}, and let $T(q,r,t)$ be the triangle adjacent to $\overline{qr}$ that lies on the other side of $\overline{qr}$ than p_i; see fig. 6. If its circumcircle $C(q,r,t)$ contains p_i then each circle through q,r contains at least one of p_i,t; see fig. 3 again. Consequently, $\overline{qr}$ cannot belong to DT_i, due to definition 2.2. Instead, $\overline{p_i t}$ will be a new Delaunay edge, because there exists a circle contained in $C(q,r,t)$ that contains only p_i and t in its interior or boundary. This process of replacing edge $\overline{qr}$ by $\overline{p_i t}$ is called an *edge flip*.

The necessary edge flips can be carried out efficiently if we know the triangle $T(q,s,r)$ of DT_{i-1} that contains p_i, see fig. 7. The line segments connecting p_i to q,r, and s will be new Delaunay edges, by the same argument from above. Next, we check if e. g. edge $\overline{qr}$ must be flipped. If so, the edges $\overline{qt}$ and $\overline{tr}$ are tested, and so on. We continue until no further edge currently forming a triangle with, but not containing p_i, needs to be flipped, and obtain DT_i.

Lemma 3.1. *If the triangle of DT_{i-1} containing p_i is known, the structural work needed for computing DT_i from DT_{i-1} is proportional to the degree d of p_i in DT_i.*

Proof. Continued edge flipping replaces $d-2$ conflicting triangles of DT_{i-1} by d new triangles in DT_i that are adjacent to p_i; compare fig. 7. □

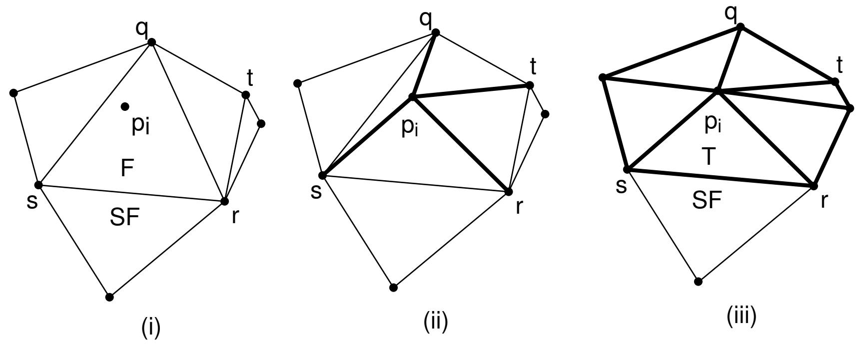


Figure 7. Updating DT_{i-1} after inserting the new site p_i. In (ii) the new Delaunay edges connecting p_i to q, r, s have been added, and edge $\overline{qr}$ has already been flipped. Two more flips are necessary before the final state shown in (iii) is reached.

Lemma 3.1 yields an obvious $O(n^2)$ time algorithm for constructing the Delaunay triangulation of n points: we can determine the triangle of DT_{i-1} containing p_i within linear time, by inspecting all candidates. Moreover, the degree of p_i is trivially bounded by n.

The last argument is quite crude. There can be single vertices in DT_i that do have a high degree, but their *average* degree is bounded by 6, as lemma 2.3 and lemma 2.4 show.

This fact calls for *randomization*. Suppose we pick p_n at random in S, then choose p_{n-1} randomly from $S - \{p_n\}$, and so on. The result is a random permutation $(p_1, p_2, \ldots, p_n)$ of the site set S.

If we insert the sites in this order, each vertex of DT_i has the same chance of being p_i. Consequently, the *expected value* of the degree of p_i is $O(1)$, and the expected total number of structural changes in the construction of DT_n is only $O(n)$, due to lemma 3.1.

In order to find the triangle that contains p_i it is sufficient to inspect all triangles that are in conflict with p_i. The following lemma shows that the expected total number of all conflicting triangles so far constructed is only logarithmic.

Lemma 3.2. *For each $h < i$, let d_h denote the expected number of triangles in $DT_h \setminus DT_{h-1}$ that are in conflict with p_i. Then,*

$$\sum_{h=1}^{i-1} d_h = O(\log i).$$

Proof. Let C denote the set of triangles of DT_h that are in conflict with p_i. A triangle $T \in C$ belongs to $\mathrm{DT}_h \setminus \mathrm{DT}_{h-1}$ iff it has p_h as a vertex. As p_h is randomly chosen in DT_h, this happens with probability $3/h$. Thus, the expected number of triangles in $C \setminus \mathrm{DT}_{h-1}$ equals $3 \cdot |C|/h$. Since the expected size of C is less than 6 we have $d_h < 18/h$, hence $\sum_{h=1}^{i-1} d_h < 18 \sum_{h=1}^{i-1} 1/h = \Theta(\log i)$. □

Suppose that T is a triangle of DT_i adjacent to p_i, see fig. 7 (iii). Its edge $\overline{sr}$ is in DT_{i-1} adjacent to two triangles: To its *father*, F, that has been in conflict with p_i; and to its *stepfather*, SF, who is still present in DT_i. Any further site in conflict with T must be in conflict with its father or with its stepfather, as illustrated by fig. 8.

This property can be exploited for quickly accessing all conflicting triangles. The *Delaunay tree* due to Boissonnat and Teillaud [46] is a directed acyclic graph that contains one node for each Delaunay triangle ever created during the incremental construction. Pointers run from fathers and stepfathers to their sons. The triangles of DT_3 are the sons of a dummy root node.

When p_{i+1} must be inserted, a Delaunay tree including all triangles up to DT_i is available. We start at its root and descend as long as the current triangle is in conflict with p_{i+1}. The above property guarantees that each conflicting triangle of DT_i will be found.

The expected number of steps this search requires is $O(\log i)$, due to lemma 3.2. Once DT_{i+1} has been computed, the Delaunay tree can easily be updated to include the new triangles.

Thus, we have the following result.

Theorem 3.2. *The Delaunay triangulation of a set of n points in the plane can be constructed in expected time $O(n \log n)$, using expected linear space. The average is taken over the different orders of inserting the n sites.*

As a nice feature, the insertion algorithm is *on-line*. That is, it is capable of constructing DT_i from DT_{i-1} without knowledge of $p_{i+1}, \ldots, p_n$.

Note also that we did not make any assumptions concerning the distribution of the sites in the plane; the incremental algorithm achieves its $O(n \log n)$ time bound for every possible input set. Only under a "poor" insertion order can a quadratic number of structural changes occur, but this is unlikely.

Randomized geometric algorithms are presented in more detail in a separate chapter of this book. Though conceptually simple, they tend to be tricky to analyze. Since Clarkson and Shor [74] introduced their technique, many researchers have been working on generalizing and simplifying the methods used. To mention but a few results, Boissonnat et. al. [43] and Guibas et. al. [135] have refined the methods of storing the past in order to locate new conflicts quickly, Clarkson et. al. [73] have generalized and simplified the analytic framework, and Seidel [228] systematically applied the technique of backward analysis first used by Chew [62]. The method in [135] for storing the past is briefly described in subsection 4.3.3 for constructing a generalized planar Voronoi diagram..

If the set S of sites can be expected to be well distributed in the plane, bucketing techniques for accessing the triangle that contains a new site p_i have been used for speed-up. Joe [152], who implemented Sloan's algorithm [236], and Su and Drys-

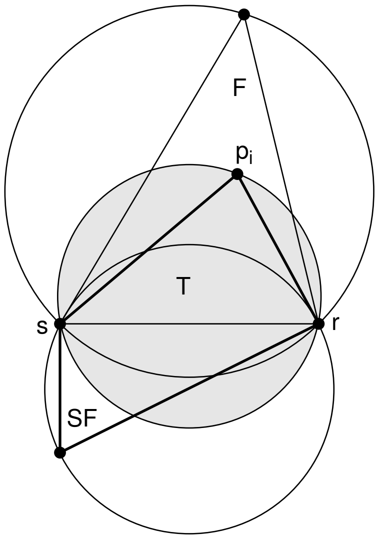


Figure 8. The circumcirle of T is contained in the union of the circumcircles of F and SF.

dale [238], who used a variant of Bentley et al.'s spiral search [36], report on fast experimental runtimes.

The arising issues of numerical stability have been addressed in Fortune [123], Sugihara [239], and Jünger et al. [153].

A technique similar to incremental insertion is *incremental search*. It starts with a single Delaunay triangle, and then incrementally discovers new ones, by growing triangles from edges of previously discovered triangles. This basic idea is used, e.g., in Maus [188] and in Dwyer [103]. It leads to efficient expected-time Delaunay algorithms in higher dimensions; see [103].

The paper [238] gives a thorough experimental comparison of available Delaunay triangulation algorithms.

3.3. Divide & conquer

The first deterministic worst-case optimal algorithm for computing the Voronoi diagram has been presented by Shamos and Hoey [230]. In their divide & conquer approach, the set of point sites, S, is split by a dividing line into subsets L and R of about the same sizes. Then, the Voronoi diagrams $V(L)$ and $V(R)$ are computed recursively. The essential part is in finding the split line, and in *merging* $V(L)$ and $V(R)$, to obtain $V(S)$. If these tasks can be carried out in time $O(n)$ then the overall running time is $O(n \log n)$.

During the recursion, vertical or horizontal split lines can be easily found if the sites in S are sorted by their x- and y-coordinates beforehand.

The merge step involves computing the set $B(L, R)$ of all Voronoi edges of $V(S)$ that separate regions of sites in L from regions of sites in R.

Suppose that the split line is vertical, and that L lies to its left.

Lemma 3.3. *The edges of $B(L, R)$ form a single y-monotone polygonal chain. In $V(S)$, the regions of all sites in L are to the left of $B(L, R)$, whereas the regions of the sites of R are to its right.*

Proof. Let b be an arbitrary edge of $B(L, R)$, and let $l \in L$ and $r \in R$ be the sites whose regions are adjacent to b. Since l has a smaller x-coordinate than r, b cannot be horizontal, and the region of l must be to its left. $\square$

Thus, $V(S)$ can be obtained by glueing together $B(L, R)$, the part of $V(L)$ to the left of $B(L, R)$, and the part of $V(R)$ to its right; see fig. 9, where $V(R)$ is depicted by dashed lines.

The polygonal chain $B(L, R)$ is constructed by finding a starting edge at infinity, and by tracing $B(L, R)$ through $V(L)$ and $V(R)$.

Due to Shamos and Hoey [230], an unbounded starting edge of $B(L, R)$ can be found in $O(n)$ time by determining a line tangent to the convex hulls of L and R, respectively. Here we describe an alternative method by Chew and Drysdale [66] since that method also works for generalized Voronoi diagrams (subsection 4.5.2). The unbounded regions of $V(L)$ and $V(R)$ are scanned simultaneously in cyclic order. For each non–empty intersection $\mathrm{VR}(l, L) \cap \mathrm{VR}(r, R)$, we test if it contains an unbounded piece of $B(l, r)$. If so, this must be an edge of $B(L, R)$, by definition 2.1. Since $B(L, R)$ has two unbounded edges, by Lemma 3.3, this search will be successful. It takes time $|V(L)| + |V(R)| = O(n)$.

Now we describe how $B(L, R)$ is traced. Suppose that the current edge b of $B(L, R)$ has just entered the region $\mathrm{VR}(l, L)$ at point v while running within $\mathrm{VR}(r, R)$, see fig. 10. We determine the points v_L and v_R where b leaves the regions of l resp. of r. The point v_L is found by scanning the boundary of $\mathrm{VR}(l, L)$ counterclockwise, starting from v. In our example, v_R is closer to v than v_L, so that it must be the endpoint of edge b.

From v_R, $B(L, R)$ continues with an edge b_2 separating l and r_2. Now we have to determine the points $v_{L,2}$ and $v_{R,2}$ where b_2 hits the boundaries of the regions of l and r_2. The crucial observation is that $v_{L,2}$ cannot be situated on the boundary segment of $\mathrm{VR}(l, L)$ from v to v_L that we have just scanned; this can be infered from the convexity of $\mathrm{VR}(l, S)$. Therefore, we need to scan the boundary of $\mathrm{VR}(l, L)$ *only from v_L on*, in counterclockwise direction.

The same reasoning applies to $V(R)$; only here, region boundaries are scanned clockwise.

Even though the same region might be visited by $B(L, R)$ several times, none of its edges is scanned more than once. The edges of $V(L)$ that are scanned all lie to the right of $B(L, R)$. This part of $V(L)$, together with $B(L, R)$, forms a planar

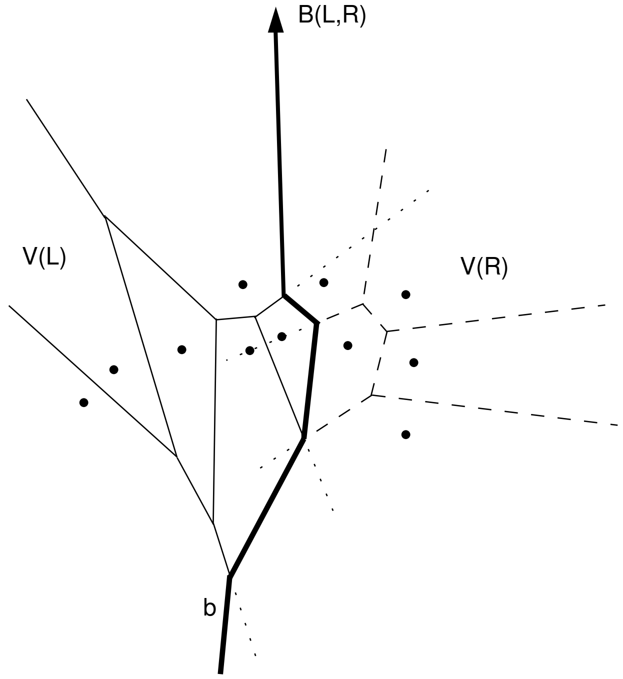


Figure 9. Merging $V(L)$ and $V(R)$ into $V(S)$.

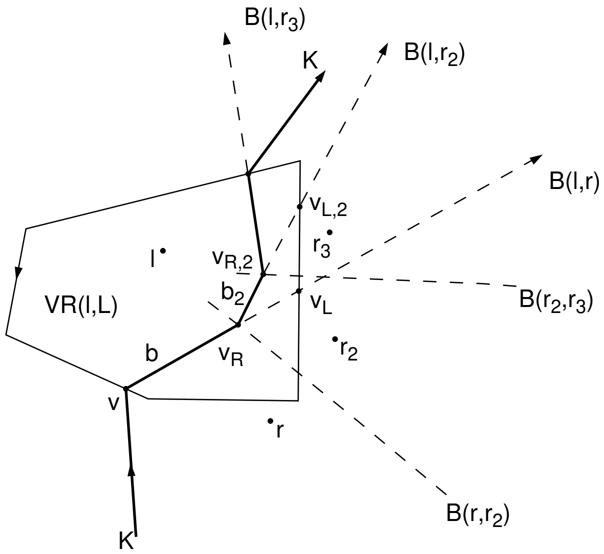


Figure 10. Computing the chain $B(L, R)$.

graph each of whose faces contains at least one edge of $B(L, R)$ in its boundary. As a consequence of lemma 2.3, the size of this graph does not exceed the size of $B(L, R)$, times a constant. The same holds for $V(R)$. Therefore, the cost of constructing $B(L, R)$ is bounded by its size, once a starting edge is given.

This leads to the following result.

Theorem 3.3. *The divide & conquer algorithm allows the Voronoi diagram of n point sites in the plane to be constructed within time $O(n \log n)$ and linear space, in the worst case. Both bounds are optimal.*

Of course, the divide & conquer paradigm can also be applied to the computation of the Delaunay triangulation DT(S). Guibas and Stolfi [136] give an implementation that uses the quad-edge data structure and only two geometric primitives, an orientation test and an in-circle test. Fortune [123] showed how to perform these tests accurately with finite precision.

Dwyer's implementation [102] uses vertical and horizontal split lines in turn, and Katajainen and Koppinen's [156] merges square buckets in a quad-tree order. Both papers report on favorable results.

Divide & conquer algorithms are candidates allowing for efficient *parallelization*. Several theoretically efficient algorithms for computing in parallel the Voronoi diagram or the Delaunay triangulation have been proposed. We refer to the recent paper by Blelloch et al. [41] for references and for a practical parallel algorithm for computing $DT(S)$. They highlight an algorithm by Edelsbrunner and Shi [116] that uses the lifting map for S (see subsection 3.5) to construct a chain of Delaunay edges that divides S. They show experimentally that their implementation is comparable in work to the best sequential algorithms.

3.4. Sweep

The well–known line sweep algorithm by Bentley and Ottmann [34] computes the intersections of n line segments in the plane by moving a vertical line, H, across the plane. The line segments currently intersected by H are stored in bottom-up order. This order must be updated whenever H reaches an endpoint of a line segment, or an intersection point. To discover the intersection points in time, it is sufficient to check, after each update of the order, those pairs of line segments that have just become neighbors on H.

It is tempting to apply the same approach to Voronoi diagrams, by keeping track of the Voronoi edges that are currently intersected by the vertical sweep line. The problem is in discovering new Voronoi regions in time. By the time the sweep line hits a new site it has been intersecting Voronoi edges of its region for a while.

Fortune [125] was the first to find a way around this difficulty. He suggested a planar transformation under which each point site becomes the leftmost point of its Voronoi region, so that it will be the first point hit during a left–to–right sweep. His transformation does not change the combinatorial structure of the Voronoi diagram.

Later, Seidel [226] and Cole [75] have shown how to avoid this transformation. They consider the Voronoi diagram of the point sites to the left of the sweep line H and of H itself, considered an additional site; see fig. 11. Because the bisector of a line and a non–incident point is a parabola, the boundary of the Voronoi region of H is a connected chain of parabola segments whose top- and bottommost edges tend to infinity. This chain is called the *wavefront*, W.

Let p be a point site to the left of H. Any point to the left of or on the parabola $B(p, H)$ is not farther to p than to H; hence, it is *a fortiori* closer to p than to any site to the right of H. Consequently, as the sweep line moves on to the right, the waves must follow because the sets $D(p_i, H)$ grow. On the other hand, each Voronoi edge to the left of W that currently separates the regions of two point sites p_i, p_j will be (part of) a Voronoi edge in $V(S)$.

During the sweep, there are two types of *events* that cause the structure of the wavefront to change, namely when a new wave appears in W, or when an old wave disappears. The first happens each time the sweep line hits a new site, e. g. p_6 in fig. 11. At that very moment $B(H, p_6)$ is a horizontal line through p_6, according to definition 2.1. A little later, its left halfline unfolds into a parabola that must be inserted into the wavefront by glueing it onto the wave of p_4 (which now contributes two segments to W.)

Let p, q be two point sites whose waves are neighbors in W. Their bisector, $B(p, q)$, gives rise to a Voronoi edge to the left of W. Its prolongation into the region of H is called a *spike*. In fig. 11 spikes are depicted as dashed lines; one can think of

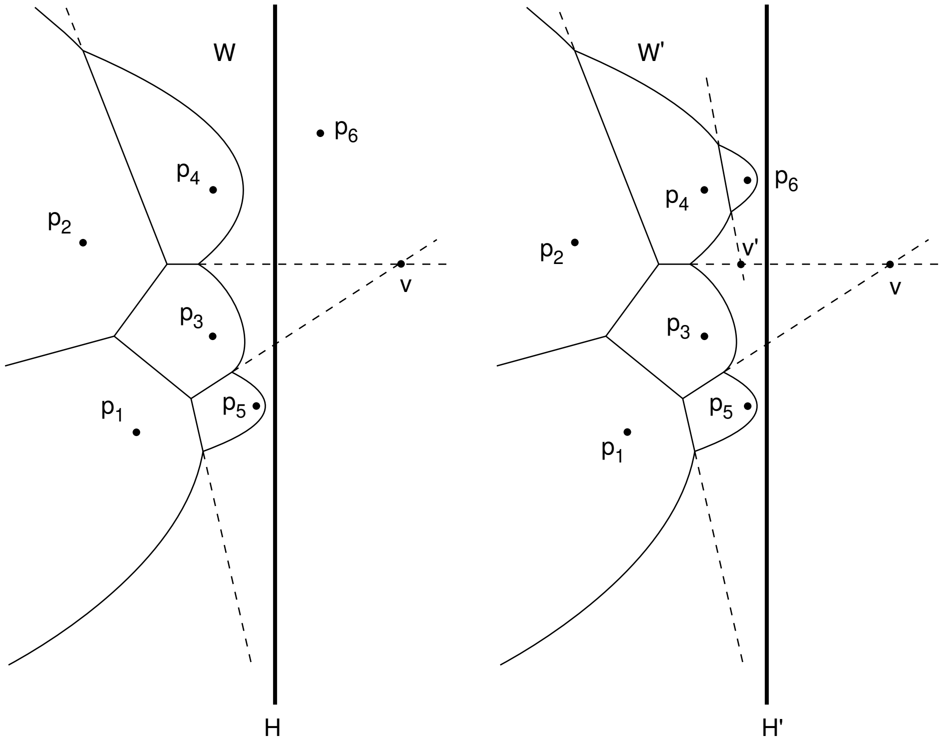


Figure 11. Voronoi diagrams of the sweep line, H, and of the points to its left.

them as tracks along which the waves are moving. A wave disappears from W when it arrives at the point where its two adjacent spikes intersect. Its former neighbors become now adjacent in the wavefront.

In fig. 11, the wave of p_3 would disappear at point v, if the new site, p_6, did not exist. But after the wave of p_6 has been inserted, there will be a previous event at v', where the lower part of the wave of p_4 disppears.

While keeping track of the wavefront one can easily maintain the Voronoi diagram of H and of the point sites to its left. As soon as all point sites have been detected and all spike intersections have been processed, $V(S)$ is obtained by removing the wavefront and extending all spikes to infinity.

Even though one wave may contribute several segments to the wavefront, the following holds.

Lemma 3.4. *The size of the wavefront is $O(n)$.*

Proof. Since any two parabolic bisectors $B(p, H), B(q, H)$ can cross at most twice, the size of the wavefront is bounded by $\lambda_2(n) = 2n - 1$, where $\lambda_s(n)$ denotes the maximum length of a Davenport–Schinzel sequence over n symbols in which no two symbols appear s times each in alternating positions; see [21]. □

The wavefront can be implemented by a balanced binary tree that stores the segments in bottom–up order. This enables us to insert a wave, or remove a wave segment, in time $O(\log n)$.

Before the sweep starts, the point sites are sorted by increasing x-coordinates and inserted into an event queue. After each update of the wavefront, newly adjacent spikes are tested for intersection. If they intersect at some point v, we insert into the event queue the time, i. e. the position x of the sweep line, when the wave segment between the two spikes arrives at v. Since the point v is a Voronoi vertex of $V(S)$, there are only $O(n)$ many events caused by spike intersections. In addition, each of the n sites causes an event. For each active spike we need to store only its first intersection event. Thus, the size of the event queue never exceeds $O(n)$. We obtain the following result.

Theorem 3.4. *Plane sweep provides an alternative way of computing the Voronoi diagram of n points in the plane within $O(n \log n)$ time and linear space.*

McAllister et al. [190] have pointed out a subtle difference between the sweep technique and the two methods mentioned before. The divide & conquer algorithm computes $\Theta(n \log n)$ many vertices, even though only a linear number of them appears in the final diagram. The randomized incremental construction method performs an expected $\Theta(n \log n)$ number of conflict tests. Both tasks, constructing a Voronoi vertex and testing a subset of sites for conflict, are usually handled by subroutines that deal directly with point coordinates, bisector equations etc. They can become quite costly if we consider sites more general than points, and distance measures more general than the Euclidean distance; see sections 4.4, 4.5, and 4.6.

The sweep algorithm, on the other hand, processes only $O(n)$ many spike events.

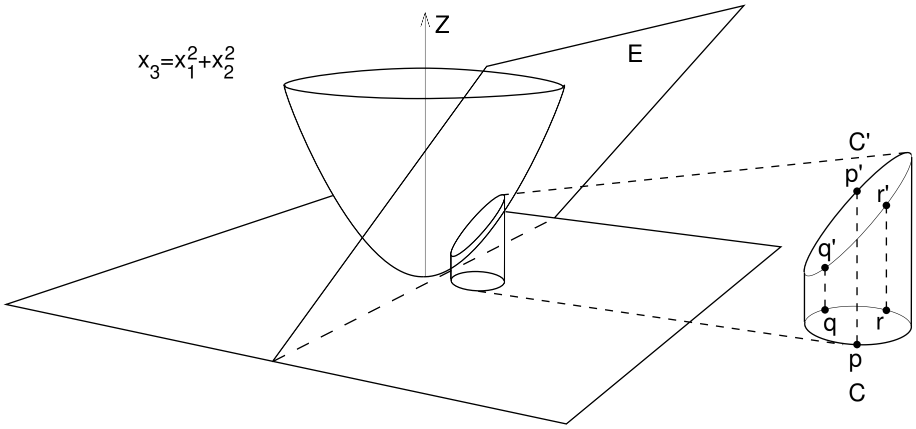


Figure 12. Lifting circles onto the paraboloid.

3.5. Lifting to 3-space

The following approach employs the powerful method of geometric transformation.

Let $P = \{(x_1, x_2, x_3) \mid x_1^2 + x_2^2 = x_3\}$ denote the paraboloid depicted in fig. 12. For each point $x = (x_1, x_2)$ in the plane, let $x' = (x_1, x_2, x_1^2 + x_2^2)$ denote its lifted image on P.

Lemma 3.5. *Let C be a circle in the plane. Then C' is a planar curve on the paraboloid P.*

Proof. Suppose that C is given by the equation

$$r^2 = (x_1 - c_1)^2 + (x_2 - c_2)^2 = x_1^2 + x_2^2 - 2x_1c_1 - 2x_2c_2 + c_1^2 + c_2^2.$$

By substituing $x_1^2 + x_2^2 = x_3$ we obtain

$$x_3 - 2x_1c_1 - 2x_2c_2 + c_1^2 + c_2^2 - r^2 = 0$$

for the points of C'. This equation defines a plane in 3-space. □

This lemma has an interesting consequence. By the lower convex hull of a set of points in 3-space we mean that part of the convex hull which is visible from the (x_1, x_2)-plane.

Theorem 3.5. *The Delaunay triangulation of S equals the projection onto the (x_1, x_2)-plane of the lower convex hull of S'.*

Proof. Let p, q, r denote three point sites of S. By lemma 3.5, the lifted image, C', of their circumcircle C lies on a plane, E, that cannot be vertical. Under the lifting mapping, the points inside C correspond to the points on the paraboloid P that lie *below* the plane E.

By theorem 2.1, p, q, r define a triangle of the Delaunay triangulation iff their circumcircle contains no further site. Equivalently, no lifted site s' is below the plane E that passes through p', q', r'. But this means that p', q', r' define a face of the lower convex hull of S'. □

Because there exist $O(n \log n)$ time algorithms for computing the convex hull of n points in 3-space, see e. g. Preparata and Shamos [213], we have obtained another optimal algorithm for the Voronoi diagram.

The connection between Voronoi diagrams and convex hulls has first been studied by Brown [49] who used the inversion transform. The simpler lifting mapping has been used, e.g., in Edelsbrunner and Seidel [113]. We shall see several applications and generalizations in subsection 4.3. In [113] also the following fact is observed.

For each point p of S, consider the paraboloid $P_p = \{(x_1, x_2, x_3) \mid (x_1 - p_1)^2 + (x_2 - p_2)^2 = x_3\}$. If these paraboloids were opaque, and of pairwise different colors, an observer looking from $x_3 = -\infty$ upwards would see the Voronoi diagram $V(S)$. In fact, the projection $x = (x_1, x_2)$ of a point $(x_1, x_2, x_3) \in P_p \cap P_q$ belongs to $B(p, q)$; and there is no site s closer to x than p and q iff (x_1, x_2, x_3) lies below all paraboloids P_s.

Instead of the paraboloids P_p one could use the surfaces $\{(x_1, x_2, f((x_1 - p_1)^2 + (x_2 - p_2)^2))\}$ generated by any function f that is strictly increasing. For example, $f(x) = \sqrt{x}$ gives rise to cones of slope $45°$ with apices at the sites. This setting illustrates the concept of circles expanding from the sites at equal speed, as mentioned after the proof of lemma 2.1. Coordinate x_3 represents time.

In order to visualize a Voronoi diagram on a graphic screen one can feed the n surfaces to a z-buffer, and eliminate by brute force those parts not visible from below.

Finally, we would like to mention a nice connection between the two ways of obtaining the Voronoi diagram by means of paraboloids explained above; it goes back to [113]. For a point $w = (w_1, w_2, w_3)$, let $\neg w$ denote its mirror image $(w_1, w_2, -w_3)$. If we apply to 3-space the mapping which sends x to $(x_1, x_2, x_3 - (x_1 - p_1)^2 - (x_2 - p_2)^2)$ then each paraboloid P_p corresponds to the tangent plane of the paraboloid $\neg P$ at the point $\neg(p')$; compare the plane equation derived in the proof of lemma 3.5, letting $c = p$ and $r = 0$.

4. Generalizations and structural properties

4.1. Characterization of Voronoi diagrams

The process of constructing the Voronoi diagram for n point sites can be seen as an assignment of a planar convex region to each of the sites, according to the nearest-neighbor rule. We now address the following, in some sense inverse, question: Given a partition of the plane into n convex regions (which are then neccessarily polygonal), do there exist sites, one for each region, such that the nearest-neighbor rule is fulfilled? In other words, when is a given convex partition the Voronoi diagram of some set of sites?

Whether a *given* set of sites induces a given convex partition as its Voronoi diagram is, of course, easy to decide by exploiting symmetry properties among the sites. For the same reason, it is easy to check whether a given triangulation is Delaunay, by exploiting the empty circumcircle property of its triangles, stated in theorem 2.1. Conditions for a given graph to be isomorphic to the Delaunay triangulation of *some* set of sites are mentioned, e.g., in the survey article by Fortune [124]. Below we concentrate on the recognition of Voronoi diagrams *without* knowing the sites.

Questions of this kind arise in facility location and in the recognition of biological growth models (as report, e.g., in Suzuki and Iri [243]) and, in particular, in the so-called gerrymander problem mentioned in Ash and Bolker [20]: When the sites are regarded as polling places and election law requires that each person votes at the respective closest polling place, the election districts form a Voronoi diagram. If the legislature draws the district lines first, how can we tell whether election law is satisfied?

Let R_i and R_j be two of the given regions. Assume that they share a common edge, and let h_{ij} be the line containing that edge. Further, let σ_{ij} denote the reflection at line h_{ij}.

Lemma 4.1. *A convex partition $R_1, \ldots, R_n$ of the plane defines a Voronoi diagram if and only if there exists a point p_i for each region R_i such that*

(1) $p_i \in R_i$ *(containment condition),*

(2) $\sigma_{ij}(p_i) = p_j$ *(reflection condition).*

Proof. If we do have a Voronoi diagram then its defining sites exist and obviously fulfil (1) and (2). To prove the converse, assume that points $p_1, \ldots, p_n$ fulfilling both conditions exist. Take any region R_i and any point x therin. We show that $d(x, p_i)$ is minimum.

To get a contradiction, suppose p_j, $j \neq i$, is closest to x. Consider an edge of R_j that is intersected by $\overline{xp_j}$, and let R_k be the region adjacent to R_j at that edge; see fig. 13. Note that $k = i$ may happen. By convexity of R_j and by (1), the line h_{jk} separates p_j from x. Hence by (2) we get $d(x, p_k) < d(x, p_j)$, a contradiction.

We conclude that p_i is closest to x among $p_1, \ldots, p_n$ which implies that R_i is the region of p_i in the Voronoi diagram $V(\{p_1, \ldots, p_n\})$. □

Based on lemma 4.1, the recognition problem can now be formulated as a linear programming problem; see Hartvigsen [138]. We first exploit the reflection condition to get a system of linear equations.

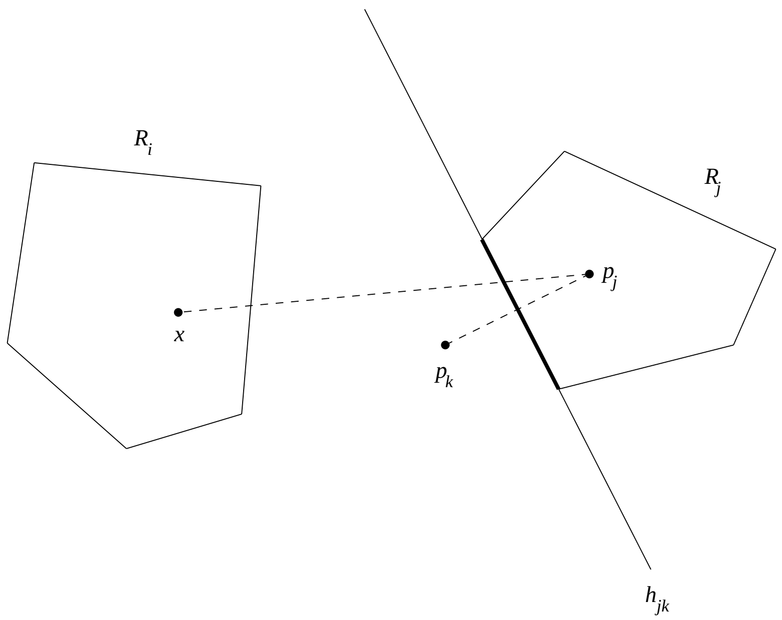


Figure 13. Region R_i must be closest to x.

Reflection at a line is an affine transformation, so we may write $\sigma_{ij}(x)$ as $A_{ij}x + b_{ij}$, for appropriate matrix A_{ij} and vector b_{ij}. Let $R_1, \ldots, R_n$ be a permutation of the regions such that R_i and R_{i+1} are adjacent, for $1 \leqslant i < n$. To get a linear system in x, set

$$p_1 = x$$

$$p_2 = A_{12}x + b_{12} := C_2 x + d_2$$

$$p_3 = A_{23}(A_{12}x + b_{12}) + b_{23} := C_3 x + d_3$$

and so on. This expresses all points p_i in terms of p_1 by using $n - 1$ adjacencies among the regions. Each of the remaining adjacencies now gives an equation of the form

$$A_{ij}(C_i x + d_i) + b_{ij} = C_j x + d_j.$$

This system has at most $3n - 3 - (n - 1) = 2n - 2$ equations by lemma 2.3. If it has no solution, or a unique solution, then we are done. In the former case, we cannot have a Voronoi diagram. In the latter, we get the coordinates of the first candidate site $p_i = x$. The corresponding other sites are obtained simply by reflection. It remains to test these sites for containment in their regions.

Setting up the system, solving it, and testing for containment can be accomplished in time $O(n)$ by standard methods. Note that only the equations of the lines bounding the regions and the adjacency information among the regions are needed. No coordinates of the region vertices are required. This is particularly interesting for the recognition problem in higher dimensions, to which the method above generalizes naturally.

The solution space of the linear system above may have dimension 1 or 2. Fig. 14 reveals that certain symmetries among the regions lead to situations of that kind. Now the containment condition is exploited to get, in addition, a set of inequalities for x.

Consider each region R_i as the intersection of all halfplanes bounded by the lines h_{ij}. Then $p_i \in R_i$ gives a set of inequalities of the form

$$p_i^T t_{ij} \leqslant a_{ij}$$

which, by plugging in $p_i = C_i x + d_i$, yields

$$(C_i^T t_{ij})x \leqslant a_{ij} - d_i^T t_{ij}.$$

Finding a feasible solution of the corresponding linear program means finding a possible site $p_1 = x$ for R_1 which, by reflection, gives all the desired sites. Since we deal with a linear program with $O(n)$ constraints and of constant dimension (actually two), also only linear time (Megiddo [193]) is spent in this more complicated case.

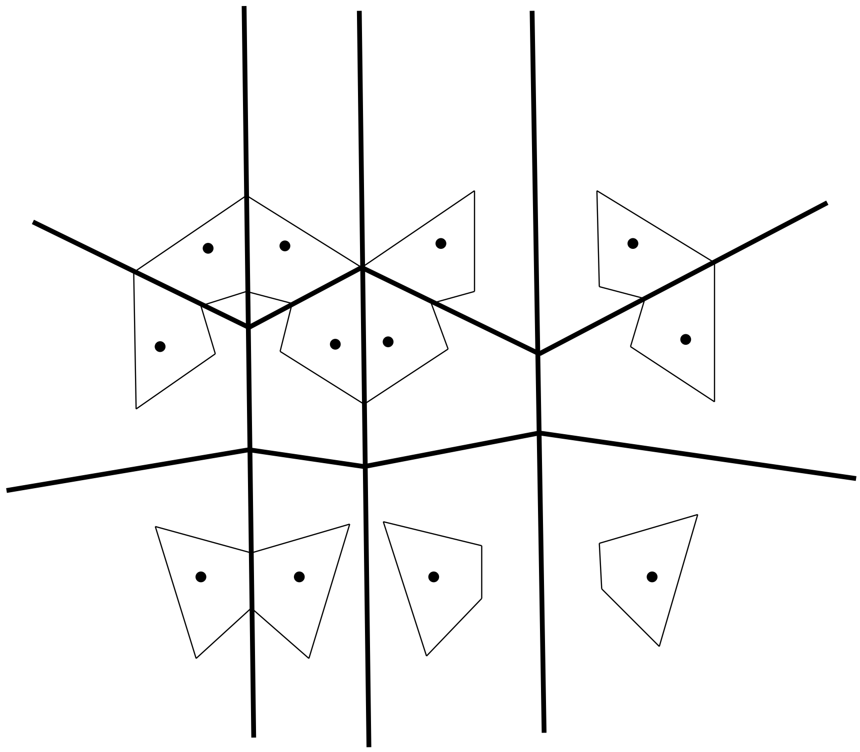


Figure 14. Sites might be taken in the polygonal areas.

Theorem 4.1. *Let $\mathcal{C}$ be a partition of 2-space into n convex regions, given by half-planes supporting the regions and by adjacencies among regions. $O(n)$ time suffices for deciding whether $\mathcal{C}$ is a Voronoi diagram, and also for restoring a suitable set of sites in case of its existence.*

This result is clearly optimal, and the underlying method easy to program. A generalization to higher dimensions is straightforward. Still, the method has to be used with care as even a slight deviation from the correct Voronoi diagram (stemming from imprecise measurement or numerical errors) will cause the method to classify $\mathcal{C}$ as non-Voronoi. Suzuki and Iri [243] give a completely different method capable of approximating $\mathcal{C}$ by a Voronoi diagram.

Lemma 4.1 extends to more general Voronoi-like partitions. The characterizing configuration of points is commonly called a *reciprocal figure*. A nice survey on this subject is Ash et al. [19]. Reciprocal figures play a role in recognizing seemingly unrelated properties of a convex partition $\mathcal{C}$, for instance checking equilibrium states of $\mathcal{C}$, see Crapo and Whiteley [77], and finding polyhedra whose boundaries project to $\mathcal{C}$, see Aurenhammer [23]. The relationship between Voronoi diagrams and polyhedra in one dimension higher will be described in subsection 4.3.2.

4.2. Optimization properties of Delaunay triangulations

The Delaunay triangulation, $\mathrm{DT}(S)$, of a set S of n sites in 2-space possesses a host of nice and useful properties many of which are well known and understood nowadays. As being the geometric dual of the Voronoi diagram $V(S)$, $\mathrm{DT}(S)$ comprises the proximity information inherent to S in a compact manner. Apart from the present subsection, various properties of $\mathrm{DT}(S)$ and their applications are described in section 5 and, in particular, in subsection 5.2. Here we look at $\mathrm{DT}(S)$ as a triangulation *per se* and concentrate on parameters which are optimized by $\mathrm{DT}(S)$ over all possible triangulations of the point set S.

Recall that a triangulation T of S is a maximal set of non-crossing line segments spanned by the sites in S. Let us call T *locally Delaunay* if, for each of its convex quadrilaterals Q, the corresponding two triangles have circumcircles empty of vertices of Q. Clearly, $\mathrm{DT}(S)$ is locally Delaunay because all circumcircles for its triangles are empty of sites; see theorem 2.1. Interestingly, the local property also implies the global one.

Theorem 4.2. *If a triangulation of S is locally Delaunay then it equals $DT(S)$.*

Proof. Let T be a triangulation of S and assume that T is locally Delaunay. We show that, for each triangle Δ of T, its circumcircle $C(\Delta)$ is empty of sites in S.

Assuming the contrary, let $s \in C(\Delta)$ for some $s \in S$ and some Δ in T. Observe $s \notin \Delta$ and let e be the edge of Δ closest to s. Suppose, w.l.o.g., that (Δ, s, e) maximizes the angle at s spanned by e, for all such triples (triangle, site, edge).

See fig. 15. Because of s, e cannot be an edge of the convex hull of S. Let triangle Δ' be adjacent to Δ at e, and let s' be the third vertex of Δ'. As T is locally

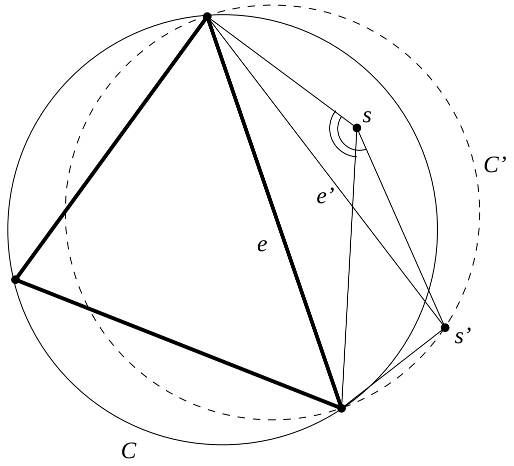


Figure 15. Angle spanned by e cannot be maximum.

Delaunay, $s' \notin C(\Delta)$, hence $s \neq s'$. Further, observe $s \in C(\Delta')$, and let e' be the edge of Δ' closest to s. The angle at s spanned by e' is larger than that spanned by e, which gives a contradiction. $\square$

An *edge flip* in a triangulation T of S is the exchange of the two diagonals in one of T's convex quadrilaterals; see subsection 3.2. Call an edge flip *good* if – after the flip – the triangulation inside the quadrilateral is locally Delaunay. Repeated exchange of diagonals of the same quadrilateral always produces an alternating sequence of good and not good flips. Theorem 4.2 now can be used to prove that DT(S) optimizes various quality measures, by showing that each good flip increases quality. Any sequence of good flips then terminates at the global optimum, the Delaunay triangulation.

One of the most prominent quality measures concerns the angles occuring in a triangulation. Recall that the number of edges (and thus of triangles) does not depend on the way of triangulating S , and let t be the number of triangles for S. The *equiangularity* of a triangulation is defined to be the sorted list of angles $(\alpha_1, \ldots, \alpha_{3t})$ of its triangles. A triangulation is called *equiangular* if it possesses lexicographically largest equiangularity among all possible triangulations for S.

As a matter of fact, every good flip increases equiangularity. Fig. 16 gives evidence for this fact. Lawson [175] called a triangulation *locally equiangular* if no flip can increase equiangularity. Locally equiangular thus is equivalent to locally Delaunay. Sibson [233] first proved theorem 4.2, showing that locally equiangular triangulations are Delaunay and hence unique. Edelsbrunner [104] observed that DT(S) is equiangular (in the global sense) as the global property implies the local

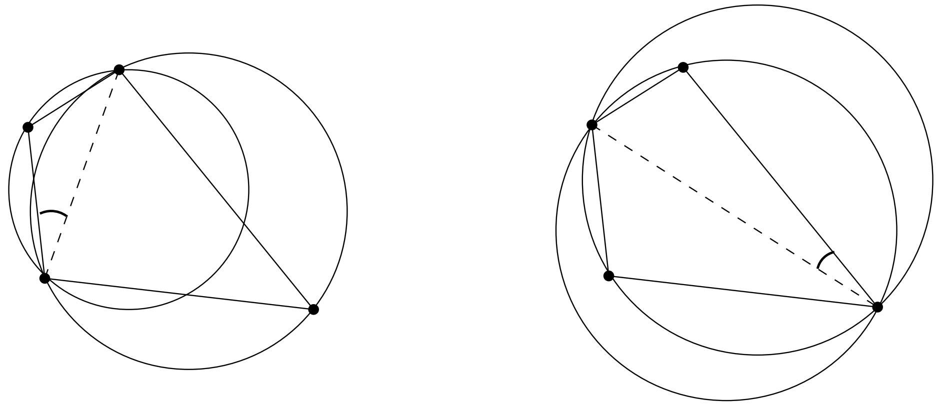


Figure 16. Equiangularity and empty circle property.

one.

In case of cocircularities among the sites, DT(S) is not a full triangulation; see section 2. Mount and Saalfeld [200] showed that DT(S) can be completed by retaining local equiangularity, in $O(n \log n)$ time.

Theorem 4.3. *Let S be a finite set of sites in 2-space. A triangulation of S is equiangular only if it is a completion of $DT(S)$.*

The equiangular triangulation obviously maximizes the minimum angle of all triangles. This property is desirable for applications to terrain modelling or to the finite element method, as was first observed in Lawson [175] and McLain [191]. By theorem 4.3, such triangulations can be computed in $O(n \log n)$ time by Delaunay triangulation algorithms, see section 3.

Only recently, it has been observed that several other parameters are optimized by DT(S). All the properties listed below can be proved by observing that every good edge flip locally optimizes the respective parameter.

Consider the smallest enclosing circle for each triangle in a triangulation, and measure *coarseness* by the largest such circle that arises. As a matter of fact, DT(S) minimizes coarseness among all possible triangulations for S. We may define coarseness also by taking smallest enclosing circles rather than circumcircles. (Note that the smallest enclosing circle differs from the circumcircle iff the triangle is obtuse.) D'Azevedo and Simpson [79] proved that DT(S) minimizes coarseness in this sense, and Rajan [215] showed that this property of Delaunay triangulations – unlike others – generalizes to higher dimensions.

Similarly, *fatness* of a triangulation may be defined as the sum of inradii of its triangles. Lambert [173] showed that DT(S) maximizes fatness, or equivalently, the mean inradius.

Given an individual function value (height) $h(p)$ for each site $p \in S$, every triangulation T of S defines a triangular surface in 3-space. The *roughness* of such a surface may be measured by

$$\sum_{\Delta \in T} |\Delta|(\alpha^2 + \beta^2)$$

with $|\Delta|$ being the area of Δ, and α, β being the slopes of the corresponding triangle in 3-space. In other words, roughness is the integral of the squared gradients. As has been shown by Rippa [218], roughness is minimum for the surface obtained from $\mathrm{DT}(S)$, for any fixed heights $h(p)$. For a simpler proof, see Powar [212].

Let us mention that, in addition, $\mathrm{DT}(S)$ provides a means for smoothing the corresponding triangular surface. As was shown in Sibson [234], each point x within the convex hull of S can be expressed as the weighted mass center of its Delaunay neighbors p in $\mathrm{DT}(S \cup \{x\})$. Weights $w_p(x)$ can be computed from area properties of the corresponding Voronoi diagram, and as functions of x, are continuously differentiable; see also Farin [121]. The corresponding interpolant to the spatial points $(p, h(p))$ is given by

$$\phi(x) = \sum_{p \in S} w_p(x) h(p).$$

This useful property of $\mathrm{DT}(S)$ is shown to generalizes to regular triangulations (duals of power diagrams for S, cf. subsection 4.3.2), and to higher-order Voronoi diagrams (subsection 4.3.3) in Aurenhammer [25].

On the negative side, $\mathrm{DT}(S)$ in general fails to fulfill optimization criteria similar to those mentionend above, such as minimizing the maximum angle, or minimizing the longest edge. Edelsbrunner et al. [119] [117] give near-quadratic time algorithms for computing triangulations optimal in that sense. $\mathrm{DT}(S)$ is not even *locally short*, in the sense that it does not always include the shorter diagonal for each of its convex quadrilaterals.

Kirkpatrick [161] proved that $\mathrm{DT}(S)$ may differ arbitrarily strongly from a *minimum-weight triangulation*, which is defined to have minimum total edge length. Computing a minimum-weight triangulation is an important and interesting problem, whose complexity status is unknown; see Garey and Johnson [127]. Subsets of edges of $\mathrm{DT}(S)$ which always have to belong to a minimum-weight triangulation are exhibited in subsection 5.2.3.

On the other hand, the widely used *greedy triangulation*, which is obtained by inserting non-crossing edges in increasing length order, can be constructed from $\mathrm{DT}(S)$ in $O(n)$ time, by a recent result in Levcopoulos and Krznaric [184].

Finally, let us mention that the Delaunay triangulation avoids an undesirable property that might be shared by other triangulations. Fix a point v in the plane, called the viewpoint. For two triangles Δ and Δ' in a given triangulation, write $\Delta < \Delta'$ if Δ fully or partially hides Δ' as seen from v. This defines a partial relation, called the *in-front/behind relation*, on the triangles. De Floriani et al. [81]

observed that this relation is acyclic if the triangulation is Delaunay. An example of a triangulation which is cyclic in spite of being minimum-weight can be found in Aichholzer et al. [10].

Edelsbrunner [105] generalized the result in [81] for regular triangulations in d dimensions, a class that includes Delaunay triangulations as a special case; see subsection 4.3.2. An application stems from a popular algorithm in computer graphics that eliminates hidden objects by first partially ordering the objects according to the in-front/behind relation and then displaying them from back to front, thereby overpainting invisible parts. In particular, this algorithm will work well for α-shapes in 3-space, discussed in subsection 5.2.2.

For a systematic treatment of planar triangulations, the reader is refered to the theses by Tan [244] and Lambert [174], respectively.

4.3. Higher dimensions, power diagrams, and order-k diagrams

In order to meet practical needs, the concept of Voronoi diagram has been modified and generalized in many ways, for example by changing the underlying space, the distance function used, or the shape of the sites. Subsections 4.3 to 4.6 give a systematic treatment of generalized Voronoi diagrams.

The most obvious generalization is to d-space, for $d \geqslant 3$. Several nice properties of the Voronoi diagram are retained (e.g., the convexity of the regions) while others are lost (e.g., the linear size). Voronoi diagrams in d-space are closely related to geometric objects in $(d+1)$-space. These relationships, and their structural and algorithmic implications, are discussed in the present subsection.

4.3.1. Voronoi diagrams and Delaunay tesselations in 3-space

Let us consider Voronoi diagrams in 3-space first. Let S be a set of n point sites in 3-space. The *bisector* of two sites $p, q \in S$ is the perpendicular plane through the midpoint of the line segment $\overline{pq}$. The region $\mathrm{VR}(p, S)$ of a site $p \in S$ is the intersection of halfspaces bounded by bisectors, and thus is a 3-dimensional convex polyhedron. The boundary of $\mathrm{VR}(p, S)$ consists of *facets* (maximal subsets within the same bisector), of *edges* (maximal line segments in the boundary of facets), and of *vertices* (endpoints of edges). The regions, facets, edges, and vertices of $V(S)$ define a *cell complex* in 3-space. This cell complex is face-to-face: if two regions have a non-empty intersection f, then f is a face (facet, edge, or vertex) of both regions. As an appropriate data structure for storing a 3-dimensional cell complex we mention the facet-edge structure in Dobkin and Laszlo [99].

The number of facets of $\mathrm{VR}(p, S)$ is at most $n-1$, at most one for each site $q \in S \setminus \{p\}$. Hence, by the Eulerian polyhedron formula, the number of edges and vertices of $\mathrm{VR}(p, S)$ is $O(n)$, too. This shows that the total number of components of the diagram $V(S)$ in 3-space is $O(n^2)$. In fact, there are configurations S that force each pair of regions of $V(S)$ to share a facet, thus achieving their maximum possible number of $\binom{n}{2}$; see, e.g., Dewdney and Vranch [89]. This fact sometimes makes Voronoi diagrams in 3-space less useful compared to 2-space. On the other hand, Dwyer [103] showed that the *expected* size of $V(S)$ in d-space is only $O(n)$, provided S is drawn uniformly at random in the unit ball. This result indicates that high-dimensional Voronoi diagrams will be small in many practical situations.

In analogy to the 2-dimensional case, the Delaunay tesselation $\mathrm{DT}(S)$ in 3-space is defined as the geometric dual of $V(S)$. It contains a tetrahedron for each vertex, a triangle for each edge, and an edge for each facet, of $V(S)$. Equivalently, $\mathrm{DT}(S)$ may be defined using the empty sphere property, by including a tetrahedron spanned by S as Delaunay iff its circumsphere is empty of sites in S. The circumcenters of these empty spheres are just the vertices of $V(S)$. $\mathrm{DT}(S)$ is a partition of the convex hull of S into tetrahedra, provided S is in general position, which will be assumed in the sequel. Note that the edges of $\mathrm{DT}(S)$ may form the complete graph on S.

Among the various proposed methods for constructing $V(S)$ in 3-space, *incremental insertion* of sites (cf. subsection 3.2) is most intuitive and easy to implement. Basically, two different techniques for integrating a new site p into $V(S)$ have been applied. The more obvious method first determines all facets of the region of p in the new diagram, $V(S \cup \{p\})$, and then deletes the parts of $V(S)$ interior to this region; see e.g. Watson [254], Field [122], and Tanemura et al. [245]. Inagaki et al. [148] describe a robust implementation of this method.

In the dual environment, this amounts to detecting and removing all tetrahedra of $\mathrm{DT}(S)$ whose circumspheres contain p, and then filling the 'hole' with empty-sphere tetrahedra with p as apex, to obtain $\mathrm{DT}(S \cup \{p\})$.

Joe [151], Rajan [215], and Edelsbrunner and Shah [115] follow a different and numerically more stable approach. Like in the planar case, after having added a site to the current Delaunay tesselation, certain flips changing the local tetrahedral structure are performed in order to achieve local Delaunayhood. The existence of such a sequence of flips is less trivial, however. Joe [150] demonstrated that no flipping sequence might exist that turns an arbitrary tetrahedral tesselation for S into $\mathrm{DT}(S)$.

Let us focus on a single flip. Recall that in 2-space, there are exactly two ways of triangulating four sites in convex position, and a flip changes one into the other. In 3-space there are also two ways of tetrahedralizing five sites in convex position, and a flip per definition exchanges them. Note, however, that the flip will replace two tetrahedra by three or vice versa; see fig. 17. This indicates an important difference between triangulations in 2-space and tetrahedral tesselations in 3-space: The number of tetrahedra *does* depend on the way of tetrahedralizing S. It may vary from $\Theta(n)$ to $\Theta(n^2)$.

After having added a site $p \in S$ to the current Delaunay tesselation, the tetrahedron containing p is split into four tetrahedra with apex p, in the obvious way. The algorithm first considers the four triangles opposite to p, that is, the bases of the tetrahedra with apex p.

Generally, each triangle Δ of the tesselation is shared by two tetrahedra T and T' which, in turn, are spanned by five sites. Three of them span Δ, and the remaining two sites q and q' belong to T and T', respectively. Δ is called *locally Delaunay* if the circumsphere of T does not enclose q' (or, equivalently, if the circumsphere of

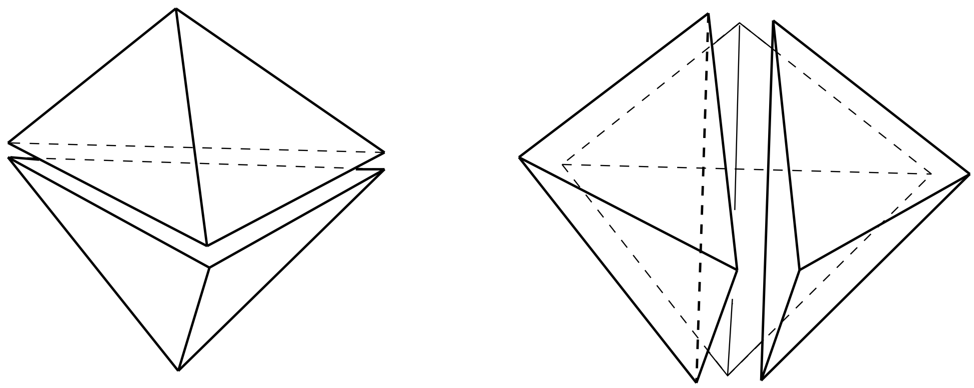


Figure 17. Two–into–three tetrahedra flip for five sites.

T' does not enclose q). A third tetrahedron might be spanned by these five sites. Δ is called *flippable* if the union of these two or three tetrahedra is convex.

For each triangle Δ opposite to p, the algorithm now performs the flip that involves the five sites corresponding to Δ, provided Δ is flippable and not locally Delaunay. Thereby, new triangles become opposite to p and possibly have to be flipped, too. This seqence of flips terminates in the Delaunay tesselation that includes p.

The algorithm works locally and thus is elegant and relatively easy to understand. The runtime is $O(n^2)$ which is optimal in the worst case. Still, the incremental approach has its drawbacks. It might construct quadratically large intermediate tesselations, in spite of the possibly linear size of the final tesselation. This unpleasant phenomenon even cannot be overcome by inserting the sites in random order. In fact, proving the existence of an – in this sense – efficient insertion order is an open problem.

Another difficulty stems from finding the starting tetrahedron that contains a newly inserted site. Search may be based on the construction history of the tesselation [115] or on bucketing techniques [152].

The algorithm can be found in much more detail in [115], including correctness proofs and a data structure for storing tetrahedral tesselations. Rajan [215] and Edelsbrunner and Shah [115] discuss the d-dimensional variant, and the latter paper generalizes the algorithm to so-called regular triangulations (subsection 4.3.2). Joe [152] provides an efficient implementation in 3-space, and Cignoni et al. [70] propose a hybrid method that works in general d-space and efficiently combines insertion, divide & conquer, and bucketing.

4.3.2. Power diagrams and convex hulls

Voronoi diagrams are intimately related to geometric objects in higher dimensions. This fact, along with one of its algorithmic applications, has already been addressed in subsection 3.5. Here, we base the discussion on a generalization of Voronoi diagrams called *power diagrams*; the geometric correspondences to be described extend to that type in a natural manner. We refer to d dimensions in order to point out the general validity of the results.

Consider a set S of n point sites in d-space. Assume that each point in S has assigned an individual weight $w(p)$. In some sense, $w(p)$ measures the capability of p to influence its neighborhood. This is expressed by the *power function*

$$pow(x,p) = (x-p)^T(x-p) - w(p)$$

of a point x in d-space with respect to a site $p \in S$.

A nice geometric interpretation is the following. For positive weights, a weighted site p can be viewed as a sphere with center p and radius $\sqrt{w(p)}$; for a point x outside this sphere, $pow(x,p) > 0$, and $\sqrt{pow(x,p)}$ expresses the distance of x to the touching point of a line tangent to the sphere and through x.

The locus of equal power with respect to two weighted sites p and q is a hyperplane called the *power hyperplane* of p and q. Let $h(p,q)$ denote the closed halfspace bounded by this hyperplane and containing the points of less power with respect to p. The *power cell* of p is given by

$$cell(p) = \bigcap_{q \in S \setminus \{p\}} h(p,q).$$

In analogy to the classical Voronoi regions, the power cells define a partition of d-space into convex polyhedra, the so-called *power diagram*, $PD(S)$, of S. See fig. 18 for a planar example. $PD(S)$ coincides with the Voronoi diagram of S if all weights are the same. In contrast to Voronoi regions, power cells might be empty if general weights are used; see $cell(p)$ in fig. 18.

$PD(S)$ is a face-to-face cell complex in d-space that consists of polyhedral faces of various dimensions j, for $0 \leqslant j \leqslant d$. In the non-degenerate case, exactly $d+1$ edges, $\binom{d+1}{2}$ facets (faces of dimension $d-1$), and $d+1$ cells meet at each vertex of $PD(S)$. For storing a d-dimensional cell complex, the cell-tuple structure in Brisson [48] seems appropriate. This data structure represents the incidence and ordering information in a cell complex in a simple uniform way.

When each weighted site $p \in S$ is interpreted as the sphere $\sigma_p = (p, \sqrt{w(p)})$, we can make the following nice observation. The part of σ_p that contributes to the union of all these spheres, $\bigcup_{p \in S} \sigma_p$, is just the part of σ_p within $cell(p)$. This means that $PD(S)$ defines a partition of this union into simply-shaped and algorithmically tractable pieces. Several algorithms concerning the union (and also the intersection) of spheres are based on this partition; see Avis et al. [32], Aurenhammer [24], and Edelsbrunner [106].

Power diagrams (and thus Voronoi diagrams) are, in a strong sense, equivalent to

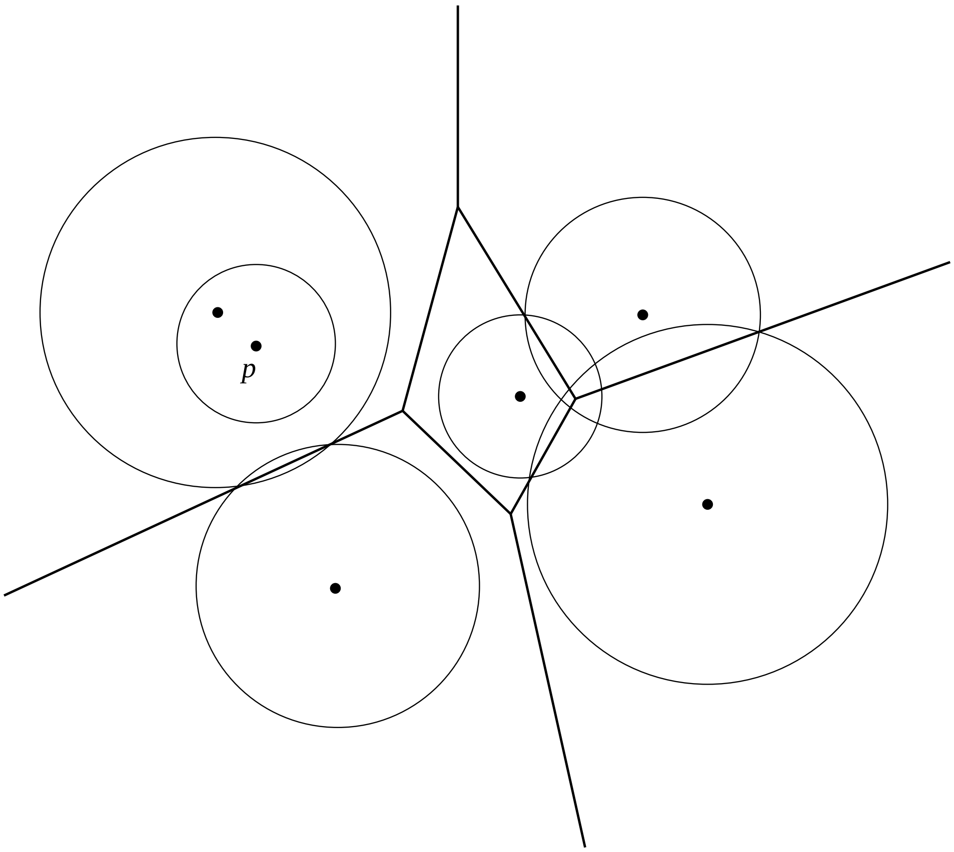


Figure 18. Power diagram for circles in the plane.

the boundaries of convex polyhedra in one dimension higher. This is a fact with far-ranging implications and has been observed in Brown [49], Klee [164], Edelsbrunner and Seidel [113], Paschinger [211], and Aurenhammer [22]. The power function $pow(x, p)$ can be expressed by the hyperplane

$$\pi(p) : x_{d+1} = 2x^T p - p^T p + w(p)$$

in $(d+1)$-space, in the sense that a point x lies in $cell(p)$ of $PD(S)$ iff, at x, $\pi(p)$ is vertically above all hyperplanes $\pi(q)$ for $q \in S \setminus \{p\}$. Hence $PD(S)$ corresponds, by vertical projection, to the upper envelope of these hyperplanes, which is the surface of a convex polyhedron, $\Pi(S)$, in $(d+1)$-space. Conversely, it is not difficult to see that every upper envelope of n non-vertical hyperplanes in $(d+1)$-space corresponds to the power diagram of n appropriately weighted sites in d-space.

The following upper bound is a direct consequence of the upper bound theorem for convex polyhedra proved in McMullen [192]. The bound is trivially sharp for power diagrams, but is achieved also for Voronoi diagrams, as was shown in Seidel [222].

Theorem 4.4. *Let S be a set of n point sites in d-space. Any power diagram for S, and in particular, the Voronoi diagram for S, realizes at most f_j faces of dimension j, for*

$$f_j = \sum_{i=0}^{a} \binom{i}{j} \binom{n-d+i-2}{i} + \sum_{i=0}^{b} \binom{d-i+1}{j} \binom{n-d+i-2}{i},$$

where $a = \lceil \frac{d}{2} \rceil$ and $b = \lfloor \frac{d}{2} \rfloor$. The numbers f_j are $O(n^{\lceil \frac{d}{2} \rceil})$, for $0 \leqslant j \leqslant d-1$.

For algorithmic issues, power diagrams can be brought in connection to convex hulls in $(d+1)$-space, by exploiting a duality (actually, polarity) between upper envelopes of hyperplanes (or intersections of upper halfspaces) and convex hulls of points. This connection is best described by generalizing the lifting map in subsection 3.5 to weighted points. A site $p \in S$ with weight $w(p)$ is transformed into the point

$$\lambda(p) = \begin{pmatrix} p \\ p^T p - w(p) \end{pmatrix}$$

in $(d+1)$-space. There is a interelation called *polarity* between the transforms λ and π. The point $\lambda(p)$ is called the pole of the hyperplane $\pi(p)$ which, in turn, is called the polar hyperplane of $\lambda(p)$. Polarity defines a one-to-one correspondence between arbitrary points and non-vertical hyperplanes in $(d+1)$-space. It is well known that polarity preserves the relative position of points and hyperplanes.

To show the connection to convex hulls, consider an arbitrary face f of the polyhedron $\Pi(S)$. Let f be the intersection of $m = d-j+1$ hyperplanes $\pi(p_1), \ldots, \pi(p_m)$, such that f is of dimension j. Each point $x \in f$ lies on these but above all other hyperplanes $\pi(q)$ defined by S. Hence the polar hyperplane of x has the points

$\lambda(p_1), \ldots, \lambda(p_m)$ on it and the remaining points $\lambda(q)$ above it. This shows that the points $\lambda(p_1), \ldots, \lambda(p_m)$ span a face of dimension $d - j$ of the convex hull of the point set $\{\lambda(p) \mid p \in S\}$.

We conclude that each j-dimensional face of $\Pi(S)$, and thus of $PD(S)$, is represented by a $(d-j)$-dimensional face of this convex hull. This implies a duality between power diagrams in d-space and convex hulls in $(d+1)$-space.

In the special case of an unweighted point set S in the plane, the parts of the convex hull that are visible from the plane project to the vertices, edges, and triangles of the Delaunay triangulation of S, and we obtain theorem 3.5 of section 3.5. A triangulation which can be obtained by projecting a convex hull is called a *regular* triangulation in Edelsbrunner and Shah [115]. Regular triangulations are just those being dual to planar power diagrams.

Once the convex hull of $\{\lambda(p) \mid p \in S\}$ has been computed, the faces of $PD(S)$, as well as their incidence and ordering relations, can be obtained in time proportional to the size of $PD(S)$.

Theorem 4.5. *Let $C_{d+1}(n)$ be the time needed to compute a convex hull of n points in $(d+1)$-space. A power diagram (and in particular, the Voronoi diagram) of a given n-point set in d-space can be computed in $C_{d+1}(n)$ time.*

Worst-case optimal convex hull algorithms working in general dimensions have been designed by Clarkson and Shor [74], Seidel [227], and Chazelle [58], yielding $C_{d+1}(n) = O(n \log n + n^{\lceil \frac{d}{2} \rceil})$. So theorem 4.5 is asymptotically optimal in the worst case. Note, however, that power diagrams in d-space may as well have a fairly small size, $O(n)$, which emphasizes the use of output-sensitive convex hull algorithms. The algorithm in Seidel [224] achieves $C_{d+1}(n) = O(n^2 + f \log f)$, where f is the total number of faces of the convex hull constructed. The latest achievements are $C_4 = O((n+f)\log^2 f)$ in Chan et al. [56] and $C_5 = O((n+f)\log^3 f)$ in Amato and Ramos [15].

Space constraints preclude our discussion of power diagrams. Still, some remarks are in order to point out their central role within the context of Voronoi diagrams. For a detailed discussion and references, see Aurenhammer and Imai [30].

The regions of a Voronoi diagram are usually defined by a set of sites and a distance function. If the regions are polyhedral, then *any* such Voronoi diagram can be shown to be the power diagram of a suitable set of weighted point sites. For instance, this is the case for the *furthest site* Voronoi diagram, whose regions consist of all points having the same furthest site in the given set.

A polyhedral cell complex in d-space is called *simple* if exactly $d+1$ cells meet at each vertex. For example, the Voronoi diagram of a set of point sites in d-space is simple if no $d+2$ sites are co-spherical. If $d \geqslant 3$, *any* simple cell complex can be shown to be a power diagram.

The class of power diagrams is closed under taking cross-sections with a hyperplane. That is, the diagram obtained from intersecting a power diagram in d-space with a hyperplane is again a power diagram, in $(d-1)$-space. Moreover, the class of power diagrams is closed under the modifications to higher order defined in subsection 4.3.3.

Several generalized Voronoi diagrams in d-space have an embedding in a power diagram in $(d+1)$-space, in the sense that they can be obtained by intersecting a power diagram with certain simple geometric objects and then projecting the intersection. For example, the *additively weighted* Voronoi diagram (i.e., the closest-point Voronoi diagram for spheres, or the Johnson-Mehl model), and the *multiplicatively weighted* Voronoi diagram (or the Apollonius model) have this property.

In all situations mentioned above, a set of weighted sites for the corresponding power diagram can be computed easily. Thus general methods of handling Voronoi diagrams and cell complexes become available. For example, the Voronoi diagram for spheres in 3-space, and the multiplicatively weighted Voronoi diagram in the plane, can both be computed in $O(n^2)$ time which is optimal. The latter diagram is investigated in detail in Aurenhammer and Edelsbrunner [28] and in Sakamoto and Takagi [221].

4.3.3. Higher-order Voronoi diagrams and arrangements

Higher-order Voronoi diagrams are natural and useful generalizations of classical Voronoi diagrams. Given a set S of n point sites in d-space, and an integer k between 1 and $n-1$, the *order-k Voronoi diagram* of S, $V_k(S)$, partitions the space into regions such that each point within a fixed region has the same k closest sites. $V_1(S)$ just is the classical Voronoi diagram of S.

The regions of $V_k(S)$ are convex polyhedra, as they arise as the intersection of halfspaces bounded by symmetry hyperplanes of the sites. A subset M of k sites in S has a non-empty region in $V_k(S)$ iff there is a sphere that encloses M but no site in $S \setminus M$. In fact, the region of M in $V_k(S)$ just is the set of centers of all such spheres.

Fig. 19 illustrates a planar order-2 Voronoi diagram. Two differences to the classical Voronoi diagram are apparent. A region need not contain its defining sites, and the bisector of two sites may contribute more than one facet.

In the extreme case of $k = n-1$, the *furthest-site* Voronoi diagram of S is obtained. It contains, for each site $p \in S$, the region of all points x for which p is the furthest site in S. Exact upper bounds on the size of furthest-site Voronoi diagrams in d-space are derived in Seidel [225].

The family of all higher-order Voronoi diagrams for a given set S of sites in d-space is closely related to an arrangement of hyperplanes in $(d+1)$-space; see Edelsbrunner and Seidel [113]. We describe this relationship in the more general setting of power diagrams, by defining an order-k power diagram, $PD_k(S)$, for a set S of weighted point sites in an analogous way. See Aurenhammer [22] [27] for more details.

Recall from subsection 4.3.2 that the power function with respect to a site $p \in S$ can be expressed by a hyperplane $\pi(p)$ in $(d+1)$-space.

The set of hyperplanes $\{\pi(p) \mid p \in S\}$ dissects $(d+1)$-space into a polyhedral cell complex called an *arrangement*. Arrangement cells are convex, and can be classified according to their relative position with respect to the hyperplanes in $\{\pi(p) \mid p \in S\}$. A cell C is said to be of *level* k if exactly k hyperplanes are

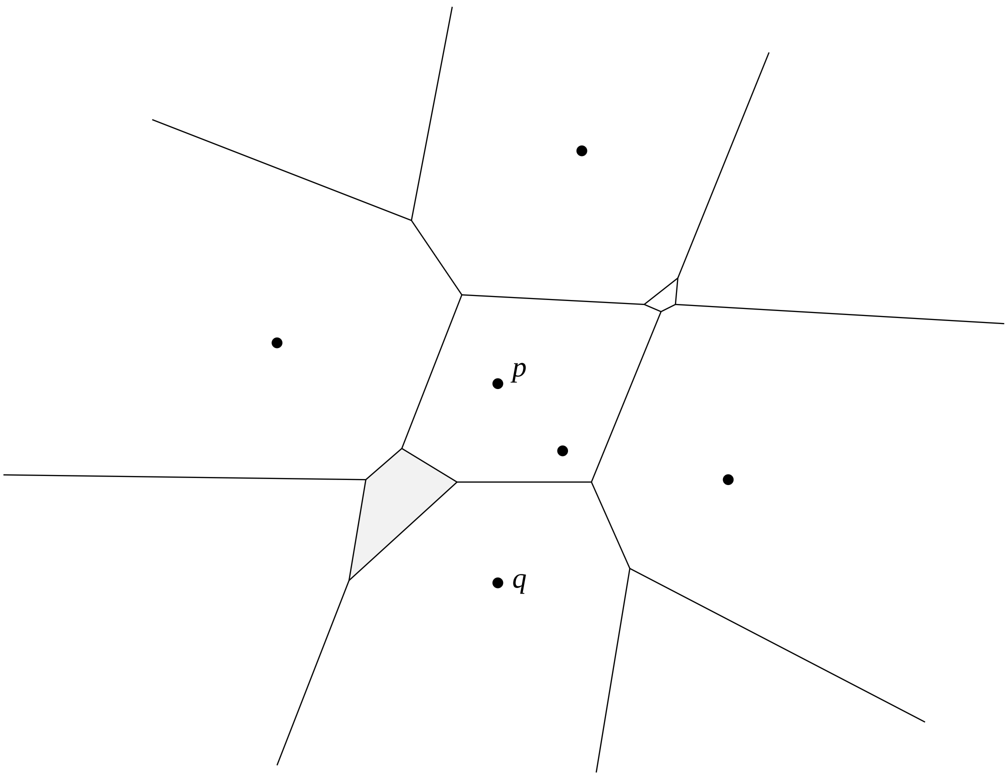


Figure 19. Region of $\{p, q\}$ in $V_2(S)$.

vertically above C. For example, the upper envelope of $\{\pi(p) \mid p \in S\}$ bounds the only cell, $\Pi(S)$, of level 0. All cells of level 1 share some facet with $\Pi(S)$, so that their vertical projection gives the (order-1) power diagram $PD(S)$.

More generally, the cells of level k project to the regions of $PD_k(S)$, for each k between 1 and $n-1$. To see this, let x be a point in some k-level cell C. Then k hyperplanes $\pi(p_1), \ldots, \pi(p_k)$ are above x, and $n-k$ hyperplanes $\pi(p_{k+1}), \ldots, \pi(p_n)$ are below x. That means that, for the vertical projection x' of x onto d-space, we have $pow(x', p_i) < pow(x', p_j)$ for $1 \leqslant i \leqslant k$ and $k+1 \leqslant j \leqslant n$. Hence x' is a point in the region of $\{p_1, \ldots, p_k\}$ in $PD_k(S)$.

Hyperplane arrangements are well-investigated objects, concerning their combinatorial as well as their algorithmic complexity; see Edelsbrunner et al. [112] [114]. We obtain:

Theorem 4.6. *Let S be a set of n (weighted or unweighted) point sites in d-space. The family of all higher-order power diagrams (or Voronoi diagrams) for S realizes a total of $\Theta(n^{d+1})$ faces, and it can be computed in optimal $\Theta(n^{d+1})$ time.*

Clarkson and Shor [74] proved that the number of arrangement cells with levels up to a given value of k is $O(n^{\lfloor d/2 \rfloor} k^{\lceil d/2 \rceil})$. The collection of these cells can be constructed within this time for $d \geqslant 4$, with the algorithm in Mulmuley [202]. A modification by Agarwal et al. [2] achieves roughly $O(nk^2)$ time also in 3-space. An output-sensitive construction algorithm is given in Mulmuley [203]. All these results apply to families of order-k diagrams in one dimension lower.

Most practical applications ask for the computation of a single order-k Voronoi diagram $V_k(S)$ in the plane, for a given value of k. (Typically, k does not depend on $|S| = n$ but is a small constant.) As for the classical Voronoi diagram $V(S)$, edges are pieces of perpendicular bisectors of sites. Vertices are centers of circles that pass through three sites. However, these circles are no longer empty; they enclose either $k-1$ or $k-2$ sites. Lee [180] showed that this diagram has $O(k(n-k))$ regions, edges, and vertices. It is easy to see that the regions of $V_2(S)$ are in one-to-one correspondence with the edges of $V(S)$. Hence $V_2(S)$ realizes at most $3n-6$ regions in the plane.

Considerable efforts have been made to compute the single planar order-k Voronoi diagram efficiently. Different approaches have been taken in Lee [180], Chazelle and Edelsbrunner [59], Aurenhammer [26], Clarkson [71], and Agarwal et al. [2]. In the last two papers randomized runtimes of $O(kn^{1+\varepsilon})$ and (roughly) $O(k(n-k)\log n)$ are achieved, respectively, which is close to optimal. Below we describe a (roughly) $O(k^2(n-k)\log n)$ time randomized incremental algorithm by Aurenhammer and Schwarzkopf [31], that can be modified to handle arbitrary on-line sequences of site insertions, site deletions, and k-nearest neighbor queries. Though not being most time efficient, the algorithm profits by its simplicity and flexibility.

The heart of the algorithm is a duality transform that relates the diagram $V_k(S)$ to a certain convex hull in 3-space. This transform allows us to insert and also delete sites in a simple fashion by computing convex hulls. Let $M \subset S$ be any subset of k sites. M is transformed into a point $q(M)$ in 3-space, by taking the centroid of M

and lifting it up vertically. More precisely,

$$q(M) = \frac{1}{k} \left(\sum_{p \in M} p, \sum_{p \in M} p^T p \right).$$

Now consider the set $Q_k(S)$ of all points that can be obtained from S in this way. That is, $Q_k(S) = \{q(M) \mid M \subset S\}$.

Lemma 4.2. *The part of the convex hull of $Q_k(S)$ that is visible from the plane is dual to $V_k(S)$.*

The lemma can be proved by first mapping each k-subset M of S into a non-vertical plane $\pi(M)$ in 3-space,

$$\pi(M) : x_3 = \frac{2}{k} \sum_{p \in M} x^T p - \frac{1}{k} \sum_{p \in M} p^T p,$$

and then considering the upper envelope $\Pi_k(S)$ of all these planes. It is not difficult to show that the facets of $\Pi_k(S)$ project vertically to the regions of $V_k(S)$. The lemma follows from observing the polarity (cf. subsection 4.3.2) between the planes $\pi(M)$ and the points $q(M)$.

To construct $V_k(S)$, we could just compute $Q_k(S)$, determine its convex hull, and then dualize its triangles, edges, and vertices that are visible from the plane. However, $Q_k(S)$ contains a point for each k-subset of S, and thus has cardinality $\binom{n}{k} = \Theta(n^k)$. Only $O(k(n-k))$ points lie on the convex hull, as $V_k(S)$ has this many regions.

We use randomized incremental insertion of sites in order to compute this convex hull efficiently. Let $S = \{p_1, \ldots, p_n\}$, and let C_i denote the visible part of the convex hull of $Q_k(\{p_1, \ldots, p_i\})$, for $k+1 \leqslant i \leqslant n$. Points of $Q_k(S)$ lying on the triangular surface C_i are called *corners* of C_i.

We start by determining C_{k+1}. $Q_k(\{p_1, \ldots, p_{k+1}\})$ contains $k+1$ points which can be calculated in time $O(k)$, so $O(k \log k)$ time suffices. The generic step of the algorithm is the insertion of site p_i into C_{i-1}, for $i \geqslant k+2$.

(1) Identify all triangles of C_{i-1} which are destroyed by p_i and cut them out. Let B be the set of corners on the boundary of the hole.

(2) Calculate the set P of all new corners created by p_i.

(3) Compute the convex hull of $P \cup B$, and fill the hole with the visible part Γ_i of this convex hull. This gives C_i.

Each triangle Δ of C_{i-1} is dual to a vertex of $V_k(S)$. This vertex is the center of a circle that passes through three sites in S. Δ will be destroyed if this circle encloses p_i. The destroyed triangles of C_{i-1} form a connected surface Σ_{i-1}. Hence, if we know one of them in advance, Σ_{i-1} can be identified in time proportional to the number n_i of its triangles. Moreover, the set P of new corners can be calculated easily from the edges of Σ_{i-1}, as each such edge gives rise to a unique corner.

Lemma 4.3. *Given C_{i-1} we can construct C_i in time $O(n_i \log n_i)$, provided we know a triangle of C_{i-1} that is destroyed by p_i.*

When looking for a starting triangle of Σ_{i-1}, we profit from another nice property of the duality transform: If the vertical projection Δ' of a triangle Δ of C_{i-1} contains p_i then Δ is destroyed by the insertion of p_i. This leaves us with the problem of locating p_i in the triangulation given by the planar projection of C_{i-1}.

In fact, we get the desired point-location structure nearly for free. Adapting a technique used in Guibas et al. [135] for constructing Delaunay triangulations, we do not remove the triangles of C_{i-1} that get destroyed by the insertion of p_i, but mark them as old. When marked old, each triangle gets a pointer to the newly constructed part Γ_i of C_i. The next site then is located by scanning through the 'construction history' of C_i. The structure for point location *within* each surface Γ_j, $j \leqslant i$, which is needed in addition, is a byproduct of the randomized incremental convex hull algorithm in [135], which we used for computing Γ_j.

In summary, the order-k Voronoi diagram for n sites in the plane can be computed in expected time $O(k^2(n-k)) \log n + nk \log^3 n)$, and optimal $O(k(n-k))$ deterministic space, by an online randomized incremental algorithm.

Full details and the following extensions are given in [31]. Deletion of a site can be done reversely to insertion, again by computing a convex hull. The history-based point location structure used by the algorithm can be adapted to support k-nearest neighbor queries (see subsection 5.1.1). A dynamic data structure is obtained, allowing for insertions and deletions of sites in expected time $O(k^2 \log n + k \log^2 n)$, and k-nearest neighbor queries in expected time $O(k \log^2 n)$. This promises a satisfactory performance for small values of k.

4.4. Generalized sites

It is commonly agreed that most geometric scenarios can be modeled with sufficient accuracy by polygonal objects. Two typical and prominent examples are the description of the workspace of a robot moving in the plane, and the geometric information contained in a geographical map. In both applications, robot motion planning and geographical information systems, the availability of proximity information for the scenario is crucial. This is among the reasons why considerable attention has been paid to the study of Voronoi diagrams for polygonal objects.

Still, in some applications the scenario can be modeled more appropriately when curved objects, for instance, circular arcs are also allowed. Many Voronoi diagram algorithms working for line segments can be modified to work for curved objects as well.

4.4.1. Line segment Voronoi diagram and medial axis
Let G be a planar straight line graph on n points in the plane, that is, a set of non-crossing line segments spanned by these points. For instance, G might be a tree, or a collection of disjoint line segments or polygons, or a complete triangulation of the points. The number of segments of G is maximum, $3n - 6$, in the last case. We will discuss several types of diagrams for planar straight line graphs in the present and following subsections.

The classical type is the *(closest point) Voronoi diagram*, $V(G)$, of G. It consists of all points in the plane which have more than one closest segment in G. $V(G)$ is known under different names in different areas, for example, as the *line Voronoi diagram* or *skeleton* of G, or as the *medial axis* when G is a simple polygon. Applications in such diverse areas as biology, geography, pattern recognition, computer graphics, and motion planning exist; see e.g. Kirkpatrick [160] and Lee [179] for references.

See fig. 20. $V(G)$ is formed by straight line edges and parabolically curved edges, both shown as dashed lines. Straight edges are part of either the perpendicular bisector of two segment endpoints, or of the angular bisector of two segments. Curved edges consist of points equidistant from a segment endpoint and a segment's interior. There are two types of vertices, namely of *type 2* having degree two, and of *type 3* having degree three (provided G is in general position). Both are equidistant from a triple of objects (segment or segment endpoint), but for type-2 vertices the triple contains a segment along with one of its endpoints.

Together with G's segments, the edges of $V(G)$ partition the plane into regions. These can be refined by introducing certain normals through segment endpoints (shown dotted in fig 20), in order to delineate *faces* each of which is closest to a particular segment or segment endpoint. Two such normals start at each segment endpoint where G forms a reflex angle, and also at each *terminal* of G which is an endpoint belonging to only one segment in G. A normal ends either at a type-2 vertex of $V(G)$ or extends to infinity.

It is well known that the number of faces, edges and vertices of $V(G)$ is linear in n, the number of segment endpoints for G. The number of vertices is shown to be at most $4n - 3$ in Lee and Drysdale [181]. An exact bound, that also counts the 'infinite' vertices at unbounded edges and segment normals, is given below.

Lemma 4.4. *Let G be a planar straight line graph on n points in the plane, and let G realize t terminals and r reflex angles. The number of (finite and infinite) vertices of $V(G)$ is exactly $2n + t + r - 2$.*

Proof. Suppose first that G consists of e disjoint segments (that do not touch at their endpoints). Then there are e regions, and each type-3 vertex belongs to three of them. By the Euler formula for planar graphs, there are exactly $2e - 2$ such vertices, if we also count those at infinity. To count the number of type-2 vertices, observe that each segment endpoint is a terminal and gives rise to two segment normals each of which, in turn, yields one (finite or infinite) vertex of type 2. Hence there are $4e$ such vertices, and $6e - 2$ vertices in total.

Now let G be a general planar straight line graph with e segments. We simulate

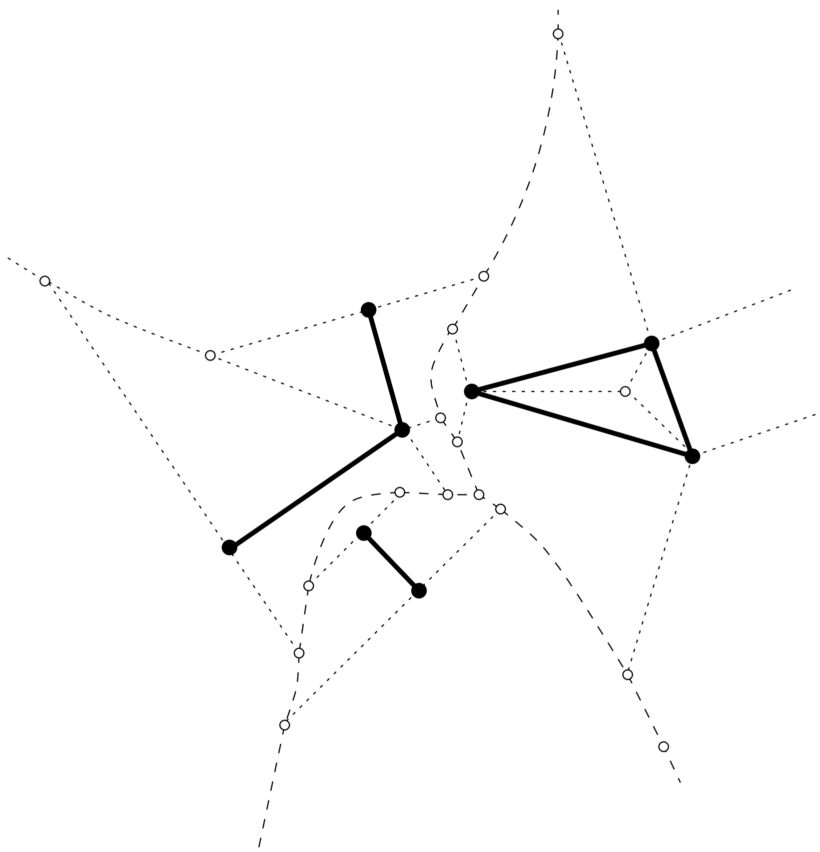


Figure 20. Line segment Voronoi diagram

G by disjoint segments, by shortening each segment slightly such that the segment endpoints are in general position. Then we subtract from $6e-2$ the number of vertices which have been generated by this simulation.

Consider an endpoint p that is incident to $d \geqslant 2$ segments of G. Obviously, p gives rise to d copies in the simulation.

The Voronoi diagram of these copies has $d-2$ finite vertices, which are new vertices of type 3. As the sum of the degrees $d \geqslant 2$ in G is $2e-t$, we get $2e-t-2(n-t)$ new vertices in this way.

Each convex angle at p gives rise to two new normals emanating at the respective copies of p, and thus to two (finite) type-2 vertices. A possible reflex angle at p gives rise to one (finite or infinite) type-3 vertex, on the perpendicular bisector of the corresponding copies of p. There are r reflex angles in G, and thus $2e-t-r$ convex angles. This gives $r+2(2e-t-r)$ new vertices in addition.

The lemma follows by simple arithmetic. □

Surprisingly, the number of edges of G does not influence the bound in lemma 4.4. The maximum number of vertices, $3n-2$, is achieved, for example, if G is a set of disjoint segments ($t=n$ and $r=0$), or if G is a simple polygon P ($t=0$ and $r=n$).

In the latter case, the majority of applications concerns the part of $V(P)$ interior to P. This part is commonly called the *medial axis* of P. The medial axis of an n-gon with r reflex interior angles has a tree-like structure and realizes exactly $n+r-2$ vertices and at most $2(n+r)-3$ edges. Lee [179] first mentioned this bound, and also listed some applications of the medial axis. An interesting application to NC pocket machining is described in Held [139].

Several algorithms for computing $V(G)$, for general or restricted planar straight line graphs G, have been proposed and tested for practical efficiency. $V(G)$ can be computed in $O(n \log n)$ time and $O(n)$ space by divide & conquer (Kirkpatrick [160], Lee [179], and Yap [259]), plane sweep (Fortune [125]), and randomized incremental insertion (Boissonnat et al. [43] and Klein et al. [169]).

Burnikel et al. [51] give an overview of existing methods, and discuss implementation details of an algorithm in Sugihara et al. [241] that first inserts all segment endpoints, and then all the segments, of G in random order. An algorithm of comparable simplicity and practical efficiency (though with a worst-case running time of $O(n^2)$) is given in Gold et al. [130]. They first construct a Voronoi diagram for point sites by selecting one endpoint for each segment, and then maintain the diagram while expanding the endpoints, one by one, to their corresponding segments. During an expansion, the resulting topological updates in the diagram can be carried out efficiently. In fact, Voronoi diagrams for moving point sites are well-studied concepts; see e.g. Guibas et al. [134] and Roos [219].

An efficient $O(n \log^2 n)$ work *parallel* algorithm for computing $V(G)$ is given in Goodrich et al. [132]. This is improved to $O(\log n)$ parallel (randomized) time using $O(n)$ processors in Rajesekaran and Ramaswami [216]. (The latter result also implies an optimal parallel construction method for the classical Voronoi diagram.)

If G is a connected graph then $V(G)$ can be computed in randomized time

$O(n \log^* n)$; see Devillers [88]. Recently, $O(n)$ time randomized, and deterministic, algorithms for the medial axis of a simple polygon have been designed by Klein and Lingas [168] and Chin et al. [68], settling open questions of long standing. The case of a convex polygon is considerably easier; see subsection 4.4.3.

Some of the algorithms above also work for curved objects. The plane-sweep algorithm in Fortune [125] elegantly handles arbitrary sets of circles (i.e., the additively weighted Voronoi diagram, or Johnson-Mehl model) without modification from the point site case. Yap [259] allows sets of disjoint segments of arbitrary degree-two curves. A randomized incremental algorithm for general curved objects is given in Alt and Schwarzkopf [12]. They show that complicated curved objects can be partitioned into 'harmless' ones by introducing new points. All these algorithms achieve an optimal running time, $O(n \log n)$.

In dimensions more than two, the known results are sparse. The complexity of the Voronoi diagram for n line segments in d-space may be as large as $\Omega(n^{d-1})$, as was observed by Aronov [17]. By the relationship of Voronoi diagrams to lower envelopes of hypersurfaces (see subsection 4.6), the results in Sharir [232] imply an upper bound of roughly $O(n^d)$. No better upper bounds are known even for line segments in 3-space.

The Voronoi diagram for n spheres in d-space has a size of only $O(n^{\lfloor d/2 \rfloor + 1})$, by its relationship to power diagrams proved in Aurenhammer and Imai [30].

A case of particular interest in several applications is the medial axis $M(P)$ of a (generally non-convex) polyhedron P in 3-space. $M(P)$ contains pieces of parabolic and hyperbolic surfaces and thus has a fairly complicated structure. A practical and numerically stable algorithm for computing $M(P)$ is proposed in Milenkovic [197].

4.4.2. Straight skeletons

In comparison to the Voronoi diagram for point sites, which is composed of straight edges, the occurrence of curved edges in the line segment Voronoi diagram $V(G)$ is a disadvantage in the computer representation and construction, and sometimes also in the application, of $V(G)$.

There have been several attempts to linearize and simplify $V(G)$, mainly for the sake of efficient point location and motion planning; see Canny and Donald [54], Kao and Mount [154], de Berg et al. [80], and McAllister et al. [190]. The compact Voronoi diagram in [190] is particularly suited to these applications. It is defined for the case where G is a set of k disjoint convex polygons. Its size is only $O(k)$, rather than $O(n)$, and it can be computed in time $O(k \log n)$; see subsection 5.1.1 for more details.

As a different alternative to $V(G)$, we now describe the *straight skeleton*, $S(G)$, of a planar straight line graph G. This structure is introduced, and discussed in much more detail, in Aichholzer and Aurenhammer [9]. $S(G)$ is composed of angular bisectors and thus does not contain curved edges. In general, its size is even less than that of $V(G)$. Beside its use as a type of skeleton for G, $S(G)$ applies, for example, to the reconstruction of terrains from a given geographical map as will be sketched later.

$S(G)$ is defined as the interference pattern of certain wavefronts propagated from the segments and segment endpoints of G. Let F be a connected component (called a *figure*) of G. Imagine F as being surrounded by a belt of (infinitesimally small) width ε. For example, a single segment s gives rise to a rectangle of length $|s| + 2\varepsilon$ and width 2ε, and a simple polygon P gives rise to two homothetic copies of P with minimum distance 2ε. In general, if F partitions the plane into c connected faces then F gives rise to c simple polygons called *wavefronts* for F.

The wavefronts arising from all the figures of G are now propagated simultanously, at the same speed, and in a self-parallel manner. Wavefront vertices move on angular bisectors of wavefront edges which, in turn, may increase or decrease in length during the propagation. This situation continues as long as wavefronts do not change combinatorially. Basically, there are two types of changes.

(1) *Edge event*: A wavefront edge collapses to length zero. (The wavefront may vanish due to three simultanous edge events.)

(2) *Split event*: A wavefront edge splits due to interference or self-interference. In the former case, two wavefronts merge into one, whereas a wavefront splits into two in the latter case.

After either type of event, we are left with a new set of wavefronts which are propagated recursively.

The edges of $S(G)$ are just the pieces of angular bisectors traced out by wavefront vertices. Each vertex of $S(G)$ corresponds to an edge event or to a split event. $S(G)$ is a unique structure defining a polygonal partition of the plane; see fig. 21.

During the propagation, each wavefront edge e sweeps across a certain area which we call the *face* of e. Each segment of G gives rise to two wavefront edges and thus to two faces, one on each side of the segment. Each terminal of G (endpoint of degree one) gives rise to one face. Faces can be shown to be monotone polygons and thus are simply connected. This gives a total of $2m + t = O(n)$ faces, if G realizes m edges and t terminals. There is also an exact bound on the number of vertices of $S(G)$.

Lemma 4.5. *Let G be a planar straight line graph on n points, t of which are terminals. The number of (finite and infinite) vertices of $S(G)$ is exactly $2n+t-2$.*

From lemma 4.4 in the previous subsection it is apparent that $S(G)$ has r vertices less than $V(G)$ if G has r reflex angles. In particular, if G is a simple polygon with r reflex interior angles, then the part of $S(G)$ interior to G is a tree with only $n-2$ vertices, whereas the medial axis of G has $n + r - 2$ vertices.

A wavefront model similar to that yielding $S(G)$ is sometimes used to define the Voronoi diagram $V(G)$ of G (cf. the expanding waves view in section 2). There, the propagation speed of all points on the wavefront is the same, whereas, in the model for $S(G)$, speed is controlled by the angle between wavefront edges. This behaviour may make $S(G)$ completely different from the Voronoi diagram of G. It can be shown that, without prior knowledge of its structure, $S(G)$ cannot be defined by means of distances from G. Moreover, $S(G)$ does not fit into the framework of abstract Voronoi diagrams described in subsection 4.6: The bisecting curve for two segments of G would be the interference pattern of the rectangular wavefronts they

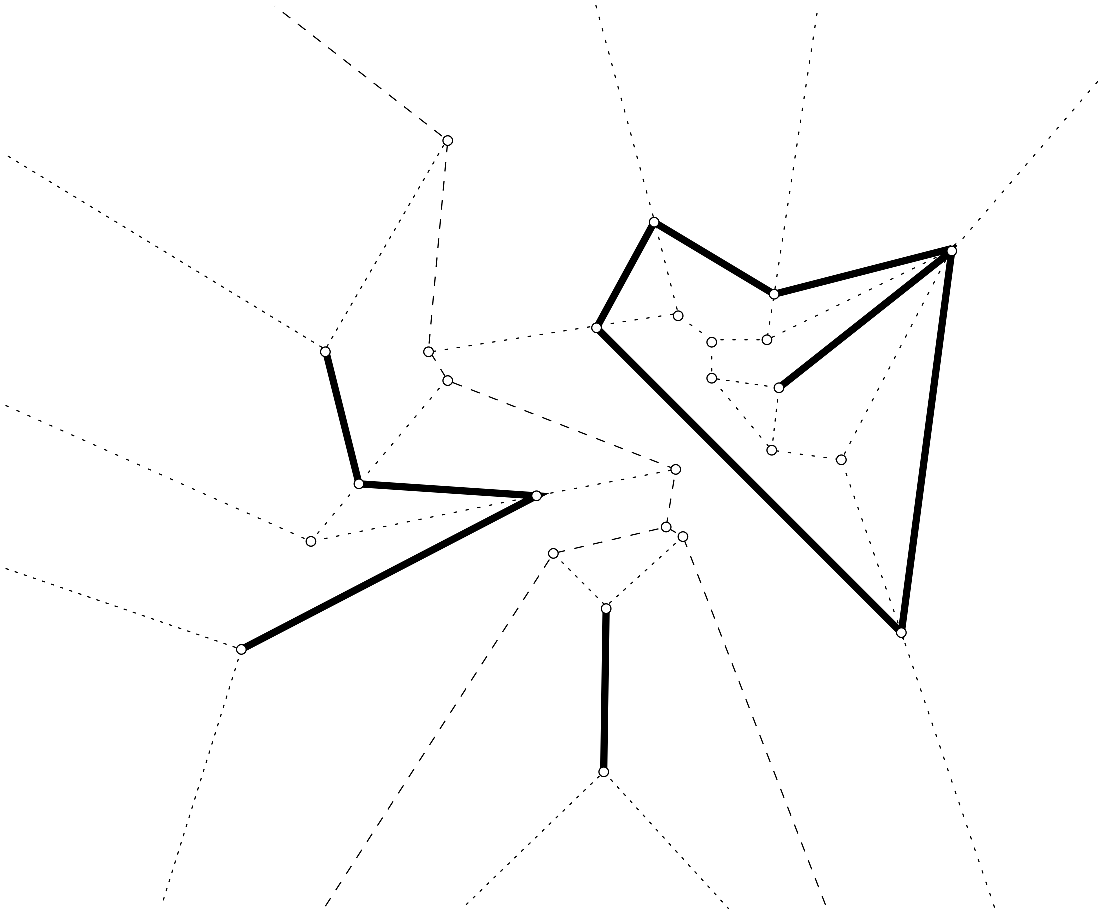


Figure 21. Straight skeleton

send out, but these curves do not fulfill condition (ii) in definition 4.2.

As a consequence, the well-developed machinery for constructing Voronoi diagrams (see section 3) does not apply to $S(G)$. An algorithm that simulates the wavefront propagation by maintaining a triangulation of the wavefront vertices is given in [9]. The method is simple and practically efficient but has a worst-case running time of $O(n^2 \log n)$.

$S(G)$ has a three-dimensional interpretation, obtained by defining the height of a point x in the plane as the unique time when x is reached by a wavefront. In this way, $S(G)$ lifts up to a polygonal surface Σ_G, where points on G have height zero. In a geographical application, G may delineate rivers, lakes, and coasts, and Σ_G represents a corresponding terrain with fixed slope. Σ_G has the nice property that every raindrop that hits a terrain facet f runs off to the segment or terminal of G defining f; see Aichholzer et al. [8]. This may have applications in the study of rain water fall and its impact on the floodings caused by rivers in a given geographic area.

The concept of $S(G)$ can be generalized by tuning the propagation speed or angle of the individual wavefront edges, in order to achieve prescribed facet slopes for Σ_G, or individual elevations for the terrain points on G. The size of $S(G)$, and its construction algorithm, remain unaffected.

When restricted to the interior of a simple polygon P, Σ_P is used in [8] as a canonical construction of a roof of given slope above P. For rectilinear (and axis-aligned) polygons P, the medial axis of P in the L_∞-metric will do the job. $S(P)$ coincides with this structure for such polygons, and thus generalizes this roof construction technique to general shapes of P.

Straight skeletons can be generalized to higher dimensions without much difficulties. They retain their piecewise-linear shape and thus, for example, offer a simpler alternative to the medial axis of a non-convex polyhedron in 3-space.

4.4.3. Convex polygons

Voronoi diagrams for a single convex polygon have a particularly simple structure. Taylor-made algorithms for their construction have been designed.

Let C be a convex n-gon in the plane. The medial axis $M(C)$ of C is a tree whose edges are pieces of angular bisectors of C's sides. In fact, $M(C)$ coincides with the part of the straight skeleton $S(C)$ interior to C. $M(C)$ realizes exactly n faces, $n-2$ vertices, and $2n-3$ edges.

There is a simple randomized incremental algorithm by Chew [62] that computes $M(C)$ in $O(n)$ time. The algorithm first removes, in random order, the halfplanes whose intersection is C. Removing a halfplane means removing the corresponding side e, and extending the two sides adjacent to e so that they become adjacent in the new convex polygon. This can be done in constant time per side. The adjacency history of C is stored. That is, for each removed side e, one of its formerly adjacent sides is recorded. In a second stage, the sides are put back in reversed (still randomized) order, and the medial axis is maintained during these insertions.

Let us focus on the insertion of the i-th side e_i. We have to integrate the face,

$f(e_i)$, of e_i into the medial axis of the $i-1$ sides that have been inserted before e_i. From the adjacency history, we already know a sides e' of the current polygon that will be adjacent to e_i after its insertion. Hence we know that the angular bisector of e_i and e' will contribute an edge to $f(e_i)$.

Having a starting edge available in $O(1)$ time, the face $f(e_i)$ now can be constructed in time proportional to its size. We construct $f(e_i)$ edge by edge, by simply tracing and deleting parts of the old medial axis interior to $f(e_i)$. As the medial axis of an i-gon has $2i-3$ edges, and each edge belongs to two faces, the expected number of edges of a randomly chosen face is less than 4. Thus $f(e_i)$ can be constructed in constant expected time, which gives an $O(n)$ randomized algorithm for computing $M(C)$.

The same technique also applies to the Voronoi diagram for the *vertices* of a convex n-gon C, that is, to the Voronoi diagram of a set S of n point sites in *convex position*. By lemma 2.2, all regions in $V(S)$ are unbounded, and the edges of $V(S)$ form a tree. Hence $V(S)$ has the same numbers of edges and (finite) vertices as the medial axis of C.

For each $p \in S$, its region $\mathrm{VR}(p, S)$ shares an unbounded edge with the regions $\mathrm{VR}(p', S)$ and $\mathrm{VR}(p'', S)$, where p' and p'' are adjacent to p on the convex hull of S (which is the polygon C). An adjacency history can be computed in $O(n)$ time, by removing the sites in random order and maintaining their convex hull. For each site that is re-inserted, the expected number of edges is less than 4. So an $O(n)$ randomized construction algorithm is obtained.

The diagrams $V(S)$ and $M(C)$ can also be computed in deterministic linear time; see Aggarwal et al. [4]. The details of this algorithm are much more involved, however.

4.4.4. Constrained Voronoi diagrams and Delaunay triangulations

In certain situations, unconstrained proximity among a set of sites is not enough information to meet practical needs. There might be reasons for not considering two sites as neighbors although they are geometrically close to each other. For example, two cities that are geographically close but separated by high mountains might be far from each other from the point of view of a truck driver. The concepts described below have been designed to model constrained proximity among a set of sites.

Let S be a set of n point sites in the plane, and let L be a set of non-crossing line segments spanned by S. Note that $|L| \leqslant 3n-6$. The segments in L are viewed as obstacles: we define the *bounded distance* between two points x and y in the plane as

$$b(x,y) = \begin{cases} d(x,y) & \text{if } \overline{xy} \cap L = \emptyset \\ \infty & \text{otherwise} \end{cases}$$

where d stands for the Euclidean distance. In the resulting *bounded Voronoi diagram* $V(S, L)$, regions of sites that are close but not visible from each other are clipped by segments in L. Regions of sites being segment endpoints are nonconvex near the

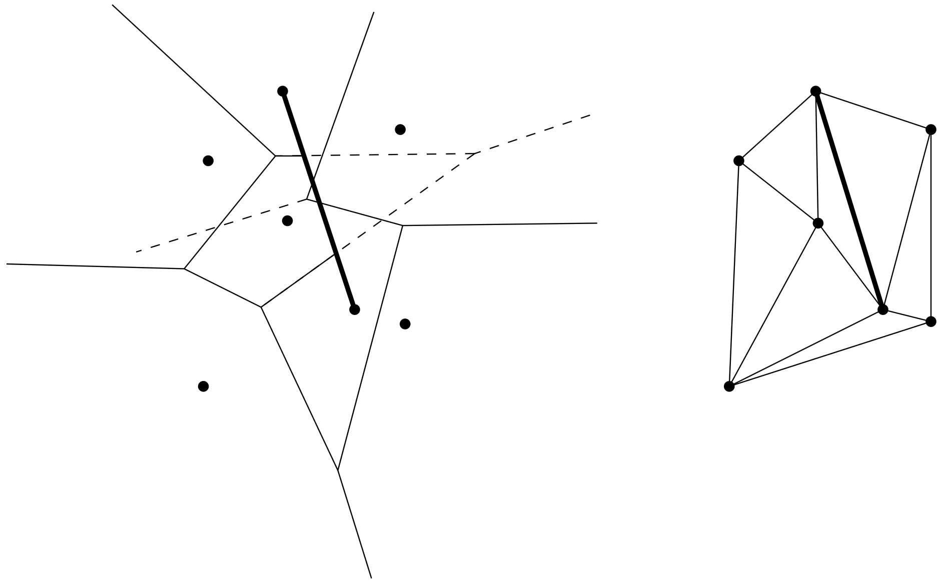


Figure 22. Bounded Voronoi diagram extended, and its dual

corresponding segment; see fig. 22.

The dual of $V(S, L)$ is not a full triangulation of S, even if the segments in L are included. However, $V(S, L)$ can be modified to dualize into a triangulation which includes L and, under this restriction, is as much 'Delaunay' as possible.

The modification is simple but nice. For each segment $\ell \in L$, the regions clipped by ℓ from the right are extended to the left of ℓ, as if only the sites of these regions were present. The regions clipped by ℓ from the left are extended similarly; see fig. 22. Of course, extended regions may overlap now, so they fail to define a partition of the plane. If we dualize now by connecting sites of regions that share an edge, a full triangulation that includes L is obtained: the *constrained Delaunay triangulation* $\mathrm{DT}(S, L)$. It is clear that the number of edges of $\mathrm{DT}(S, L)$ is at most $3n - 6$, and that, in general, the number of edges of $V(S, L)$ is even less. Hence both structures have a linear size.

The original definition of $\mathrm{DT}(S, L)$ in Lee and Lin [182] is based on a modification of the empty circle property: $\mathrm{DT}(S, L)$ contains L and, in addition, all edges between sites $p, q \in S$ that have $b(p, q) < \infty$ and that lie on a circle enclosing only sites $r \in S$ with at least one of $b(r, p), b(r, q) = \infty$.

Algorithms for computing $V(S, L)$ and $\mathrm{DT}(S, L)$ have been proposed in Lee and Lin [182], Chew [63], Wang and Schubert [253], Wang [252], Seidel [226], and Kao and Mount [155]. The last two methods seem best suited to implementation. For an application of $\mathrm{DT}(S, L)$ to quality mesh generation see Chew [65].

We sketch the $O(n \log n)$ time plane sweep approach in [226]. If only $V(S, L)$ is required then the plane sweep algorithm described in subsection 3.4 can be applied without much modification. If $\mathrm{DT}(S, L)$ is desired then the extensions of $V(S, L)$ as described above are computed in addition. To this end, an additional sweep is carried out for each segment $\ell \in L$. The sweep starts from the line through ℓ in both directions. It constructs, on the left side of this line, the (usual) Voronoi diagram of the sites whose regions in $V(S, L)$ are clipped by ℓ from the right, and vice versa.

The special case where S and L are the sets of vertices, and sides, of a simple polygon has received special attention, mainly because of its applications to visibility problems in polygons. The bounded Voronoi diagram $V(S, L)$ is constructable in $O(n)$ randomized time in this case; see Klein and Lingas [166]. If the L_1-metric instead of the Euclidean metric is used to measure distances, the same authors [168] give a deterministic linear time algorithm. Both algorithms, as well as the linear time medial axis algorithms in [167] and in [68], first decompose the polygon into smaller parts called histograms. These are polygons whose vertices, when considered in cyclic order, appear in sorted order in some direction.

An alternative concept that forces a set L of line segments spanned by S into $\mathrm{DT}(S)$ is the *conforming Delaunay triangulation*. For each segment $\ell \in L$ that does not appear in $\mathrm{DT}(S)$, new sites on ℓ are added such that ℓ becomes expressible as the union of Delaunay edges in $\mathrm{DT}(S \cup C)$, where C is the total set of added sites. For several site adding algorithms, $|C|$ depends on the size as well as on the geometry of L. See, e.g., the survey article by Bern and Eppstein [37] and references therein. Edelsbrunner and Tan [118] show that $|C| = O(k^2 n)$ is always sufficient, and construct a set of sites with this size in time $O(k^2 n + n^2)$, for $k = |L|$.

A different, and more complicated, type of constrained Voronoi diagram is the *geodesic Voronoi diagram*. Here, the distance between a point site p and a point x in the plane is equal to the length of the shortest obstacle-avoiding path between p and x. The obstacles are usually modeled by a set of non-crossing line segments. If all segment endpoints are sites then the bounded Voronoi diagram is obtained. However, this is typically not the case. The fact that computing geodesic distances is not a constant-time operation complicates the construction of geodesic Voronoi diagrams. The only known subquadratic algorithm is by Mitchell [198]. An $O((n + k)\log(n + k))$ time algorithm for the geodesic Voronoi diagram of k sites inside a simple n-gon is given in Papadopoulou and Lee [210], improving over an earlier approach in Aronov [16].

4.5. Generalized spaces and distances

So far we have maily discussed Voronoi diagrams of sites in d-space, that are defined with respect to the Euclidean distance function. Now we want to generalize both the space in which the sites are situated and the distance measure used; but we shall only discuss the case of point sites.

The main questions are which of the structural properties the standard Voronoi diagram enjoys will be preserved, and will the remaining properties be strong enough to apply one of the algorithmic approaches for computing the Voronoi diagram introduced in section 3.

4.5.1. Generalized spaces

Since the surface of earth is not flat, it seems very natural to ask about Voronoi diagrams of point sites on curved surfaces in 3-space. The distance between two points on the surface is the minimum Euclidean length of a curve that connects the points and runs entirely inside the surface. Such a curve will be called a *shortest path*.

Brown [50] has addressed the Voronoi diagram of points on the surface of the two-dimensional *sphere*. Here great circles play the role of lines in the Euclidean plane. In fact, the bisector of two points is a great circle, and the shortest paths are segments of great circles, too. (One can show that the only other metric space in which all bisector segments are shortest paths is the hyperbolic space; see Busemann [52]).

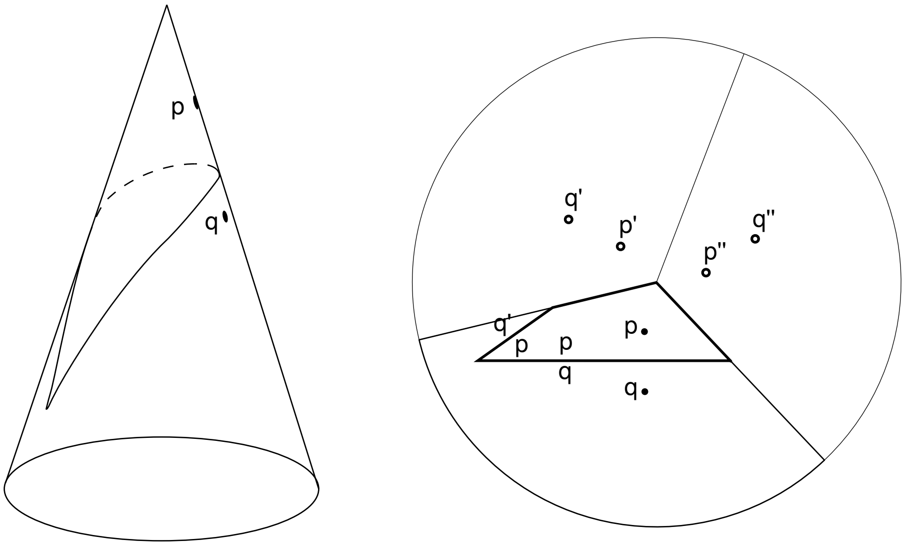
For each pair of antipodal points on the sphere there is a continuum of shortest paths connecting them. But this does not affect the Voronoi diagram of n points; it can be computed in optimal $O(n \log n)$ time and linear space, by adaption of the algorithms mentioned in section 3.

Quite different is the situation on the surface of a *cone*. In order to determine the bisector of two points, p and q, we can cut the cone along a halfline emanating from the apex, and unfold it; in fig. 23 the halfline diametrically opposed to p has been chosen. Since curve length does not change in this process, each shortest path on the cone that does not cross the cut is transformed into a shortest path in the plane, i. e. into a line segment. In order to represent those shortest paths that cross the cut, we add to the unfolded cone two more copies, as shown in fig. 23 (ii). Now the shortest path on the cone from some point x to site q corresponds to the shortest one of the line segments $\overline{qx}$, $\overline{q'x}$, and $\overline{q''x}$. This explains why the unfolded bisector $B(p,q)$ consists of segments of the planar bisectors of p,q and p,q'.

In spite of this strong connection to the plane, the Voronoi diagram of points on a cone has structural properties surprisingly different from the planar Voronoi diagram. If the unfolded cone forms a wedge of angle less than 180° then the bisector of two points can be a closed curve. If three points p,q,r are placed, in this order, on a halfline emanating from the apex of such a cone, the bisector $B(q,r)$ fully encircles $B(p,q)$ which in turn encircles the apex. This causes the Voronoi region of q in $V(\{p,q,r\})$ to be not simply connected.

Also, the closures of two Voronoi regions can have more than one Voronoi edge in common. Such a situation is shown in fig. 24, on the unfolded cone. The bisectors of the three points cross twice, at the Voronoi vertices v and w; the latter happens to lie on the cut. (It is interesting to observe that none of these phenomena occurs on the sphere, although there, too, bisectors are closed curves and cross twice.)

Despite these fundamental differences to the plane, the Voronoi diagram of n points on the surface of a cone can be constructed in optimal time and space, using a sweep circle that expands from the apex; see Dehne and Klein [83]. This approach



(i) (ii)

Figure 23. A cone sliced and unfolded, showing the bisector of p and q.

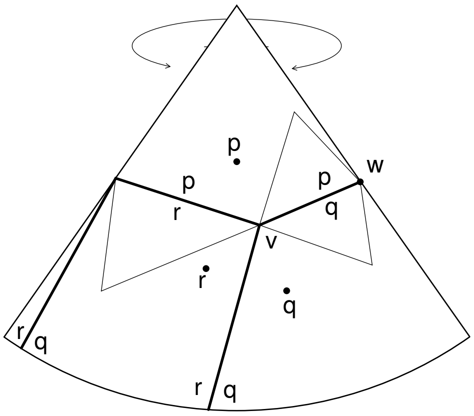


Figure 24. The common border of the Voronoi regions of q and r consists of two Voronoi edges.

works without unfolding the cone.

Mazón and Recio [189] have independently pointed out the algebraic background of the unfolding procedure illustrated by fig. 23 (i), and obtained the following generalization. Let P denote the Euclidean plane or two-dimensional sphere, and let G be a *discrete group* of motions on P: a group of bijections that leave the distance between any pair of points of P invariant, such that for each point $p \in P$ there exists a constant c satisfying

$$p \neq g(p) \Longrightarrow |p - g(p)| \geqslant c$$

for all motions $g \in G$.

Examples in the plane are the group generated by a rotation of rational angle about some given point, or the group generated by two translations that move each point a fixed distance to the right and a fixed distance upwards, respectively. In the 19th century, mathematicians have completely classified all discrete groups.

Two points $p, p' \in P$ are equivalent if there exists a motion in G that takes p to p'; the equivalence class, $[p]$, of p is called the orbit of p. The *quotient space*, P/G, consists of all orbits. In order to geometrically represent P/G one starts with a connected subset of P that contains a representative out of every orbit; equivalent points must be on the boundary. Such a set is called a fundamental domain, if it is convex. The following lemma provides a nice way of obtaining a fundamental domain; a proof can be found in Ehrlich and Im Hof [120].

Lemma 4.6. *Let p be a point of P that is left fixed only by the unit element of G. Then its Voronoi region $VR(p, [p])$ is a fundamental domain.*

In fig. 23 (ii), for example, the point set $\{p, p', p''\}$ is the orbit of p under a clockwise rotation by 120°. The Voronoi region $\mathrm{VR}(p, \{p, p', p''\})$ equals the master copy of the unfolded cone, as drawn by solid lines. Each interior point x is the only point of $[x]$ contained in this region, only the points on the boundary (i. e. the cut) of the unfolded cone are mapped into each other by rotation. If we identify these two halflines we obtain the cone depicted in fig. 23 (i), a model of the quotient space of the Euclidean plane over the cyclic group of order 3.

In a similar way we would obtain a rectangle as fundamental domain of the group of two translations mentioned above, and identifying opposite edges would result in a *torus*.

If we want to compute the Voronoi diagram of a set S of point sites on a surface associated with such a quotient space P/G, we can proceed as follows. Let S_0 denote a set of representatives of S in a fundamental domain $D \subset P$. First, we compute the Voronoi diagram $V([S_0])$, where $[S_0]$ denotes the union of the orbits of the elements of S_0, an infinite but periodic set. Due to [189], $V([S_0])$ can be obtained by applying the motions in G to the Voronoi diagram of a finite set of points of S_0 and translated copies of S_0.

Theorem 4.7. *There exists a finite subset U of $[S_0]$ such that $V([S_0]) = [V(U) \cap D]$ holds.*

If one removes from $V([S_0])$ all Voronoi edges that separate points of the same orbit and intersects the resulting structure with the fundamental domain D, the desired diagram $V(S)$ results, after identifying equivalent points. Although the set S_0 can be constructed effectively, it seems hard to establish an upper bound for the efficiency of this step.

To mention a few other spaces considered, Ehrlich and Im Hof [120] have studied, from a differential geometrist's point of view, structural properties of the Voronoi diagram in such Riemannian manifolds where any two points are connected by a unique shortest path.

In order to compute the Voronoi diagram of n points on a *polyhedral surface* in 3-space containing m vertices, one can make use of its discrete structure and apply the *continuous Dijkstra* technique usually employed for computing shortest paths. It allows the Voronoi diagram to be computed in $O(N^2 \log N)$ time and $O(N^2)$ space, where $N = \max(m, n)$; see Mitchell et al. [199].

4.5.2. Convex distance functions

In numerous applications does the Euclidean metric not provide an appropriate way of measuring distance. In the following subsections we consider the Voronoi diagram of point sites under distance measures different from the Euclidean metric. We start with convex distance functions, a concept that generalizes the Euclidean distance but slightly. Whereas this generalization does not cause serious difficulties in the plane, surprising changes will occur as we move to 3-space.

Let C denote a compact, convex set in the plane that contains the origin in its interior. Then a *convex distance function* can be defined in the following way. In order to measure the distance from a point p to some point q, the set C is translated by the vector p. The half line from p through q intersects the boundary of C at a unique point q'; see fig. 25. Now one puts

$$d_C(p,q) = \frac{d(p,q)}{d(p,q')}.$$

By definition, C equals the *unit circle* of d, that is, the set of all points q satisfying $d_C(0,q) \leqslant 1$. The value of $d_C(p,q)$ does not change if both p and q are translated by the same vector. One can show that the *triangle inequality* $d_C(p,r) \leqslant d_C(p,q) + d_C(q,r)$ is fulfilled, with equality holding for collinear points p, q, r. In general, we have $d_C(p,q) = d_{C'}(q,p)$, where C' denotes the reflected image of C about the origin. We can define the Voronoi diagram based on an arbitrary convex distance function by associating with each site p all points x of the plane such that $d_C(p,x) < d_C(q,x)$ holds for all other sites q.

If the set C is symmetric about the origin then d_C is called a *symmetric convex distance function*. This is equivalent to saying that the function $q \mapsto d_C(0,q)$ is a *norm* in the plane.

Well-known is the family of L_p (or: Minkowski) norms, $1 \leqslant p < \infty$, defined by

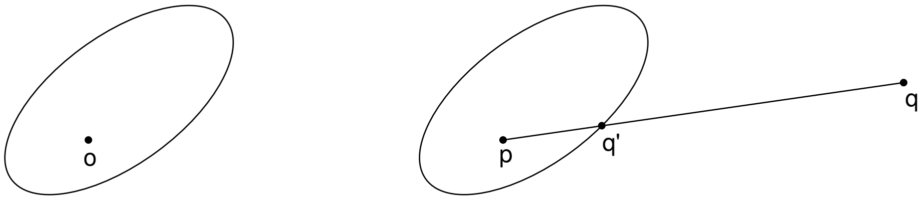


Figure 25. By $d_C(p,q) = \frac{d(p,q)}{d(p,p')}$ a convex distance function d_C with unit circle C is defined.

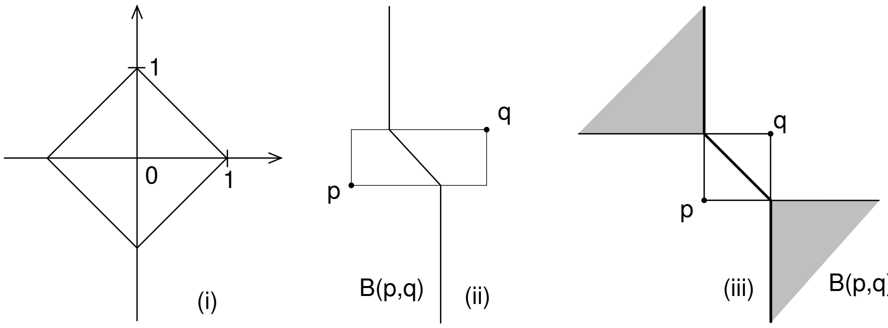


Figure 26. Unit circle and bisectors of the Manhattan distance L_1.

the equation

$$L_p(q,r) = \sqrt[p]{|q_1 - r_1|^p + |q_2 - r_2|^p} \text{ for } q = (q_1, q_2), r = (r_1, r_2).$$

Whereas L_2 is the Euclidean distance, $L_1(q,r) = |q_1 - r_1| + |q_2 - r_2|$ is called the *Manhattan distance* of q and r, because it equals the minimum length of a path from q to r that follows a rectangular grid. Its unit circle is shown in fig. 26 (i). If index p tends to ∞, the value of $L_p(q,r)$ converges to $L_\infty(q,r) = \max(|q_1 - r_1|, |q_2 - r_2|)$. The unit circle of L_∞ equals the aligned square $[-1,1]^2$; therefore, L_1 and L_∞ are related by a 45° rotation of the plane.

In fig. 26 (ii) the bisector of two points under the Manhattan distance L_1 is shown. Another possible situation can be obtained by rotating picture (ii) by 90°. If p and q are the diagonal vertices of an aligned square then their bisector $B(p,q)$ is no longer a curve: it consists of two quarterplanes connected by a line segment; see fig. 26 (iii).

In general, if the unit circle C is *strictly convex*, that is, if its boundary contains no line segment, this phenomenon cannot occur. For a strictly convex distance function d_C, each bisector $B(p,q)$ is a curve homeomorphic to a line. Also, the strict triangle inequality holds: We have $d_C(p,r) < d_C(p,q) + d_C(q,r)$ unless p,q,r are collinear. Moreover, two circles with respect to d_C intersect in at most two points, and two bisectors $B(p,q), B(p,r)$ intersect in at most one point; proofs can be found in Icking et al. [146].

If C is not strictly convex then there are points p,q whose bisector contains two-dimensional pieces. In this case one can choose from the set $B(p,q)$ a curve that separates p and q, and use this curve as the bisector. In fig. 26 (iii), for example, one could proceed as if p and q were slightly moved apart in y-direction and use the curve drawn by solid lines as their bisector. In general, bisecting curves can be chosen as follows. We assume that the point sites in S are linearly ordered. If $p < q$ then the common boundary of $D(q,p)$ and $B(p,q)$ is chosen, i. e. the set $B(p,q)$ is added to the region of p with respect to q. This choice leads to a consistent definition of the Voronoi diagram.

Voronoi regions based on convex distance functions are in general not convex, as the example of the Manhattan distance shows. But they are still *star-shaped*, as seen from their sites: For each point $x \in D(p,q)$, the line segment $\overline{px}$ is also contained in $D(p,q)$. This follows from a more general fact shown in lemma 4.8 below.

The star-shapedness of the Voronoi regions, together with the convexity of the circles, is a property strong enough for applying the divide & conquer algorithm; cf. subsection 3.3. Hwang [145], Lee [178], and Lee and Wong [183] have studied the Voronoi diagram based on L_p norms; they provided algorithms that run within $O(n \log n)$ many steps. Here and in the sequel, a *step* not only denotes a single operation of a Real Random Access Machine (cf. [213]) but also an elementary operation like computing the intersection of two bisector curves.

Widmayer et al. [256] have described an optimal algorithm for computing the Voronoi diagram of a distance function based on a convex m-gon, C. This generalizes the Manhattan distance to an environment where motions are restricted to a *finite set of orientations* given by the rays from the origin through the vertices of C.

Eventually, Chew and Drysdale [66] have shown how to construct, in $O(n \log n)$ many steps, the Voronoi diagram based on an arbitrary convex distance function in the plane, using divide & conquer. One crucial point is in the merge step. If we use a split line to subdivide the site set S into subsets L and R, the set $B(L,R)$ of all Voronoi edges of $V(S)$ that separate L-regions from R-regions need not be connected, but it must be *acyclic*. Each of its connected components is an unbounded chain, not a loop. Therefore, a starting edge of each component can be found at infinity, as in the Euclidean case.

Skyum [235] has shown how to construct the dual of the Voronoi diagram based on a convex distance function using the sweep line approach, in $O(n \log n)$ many steps.

It is well known that each symmetric convex distance function d_C is equivalent to the Euclidean distance d, in the sense that for suitable constants a and A, the

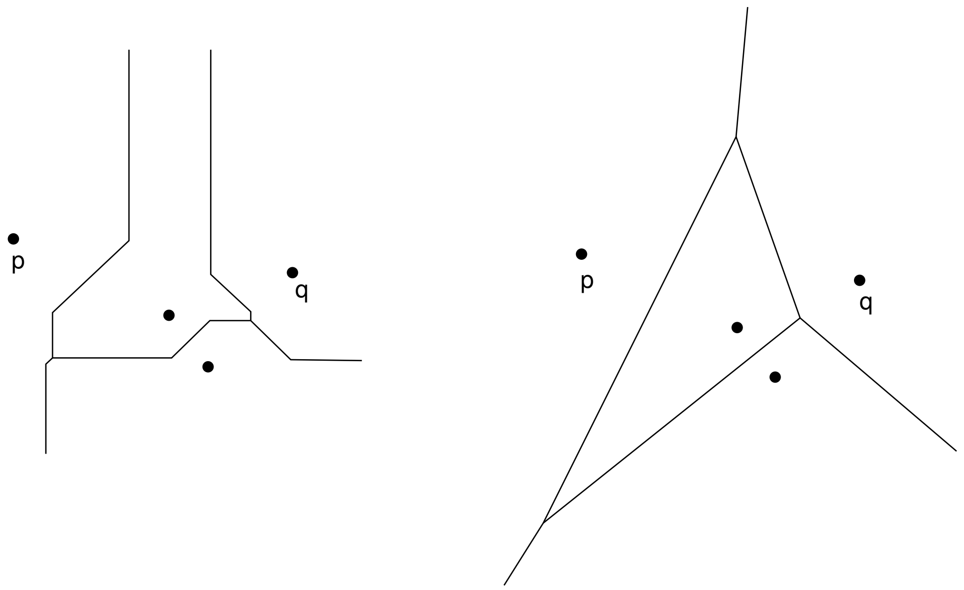


Figure 27. Four point sites whose Voronoi diagrams based on the Manhattan and the Euclidean distance have a different combinatorial structure.

inequalities

$$a \cdot d(p,q) \leqslant d_C(p,q) \leqslant A \cdot d(p,q)$$

hold for all points p, q. In particular, a sequence of points p_i converges to some limit point p under d_C iff it does so under the Euclidean distance. One might wonder if these similarities cause the Voronoi diagrams of d and d_C to have similar combinatorial structures. A counterexample is shown in fig. 27; the regions of p and q are adjacent under L_2 but not under L_1.

Corbalán et al. [76] have provided systematic answers to the above question. Let d_C, d_D denote two symmetric, strictly convex distance functions whose unit circles are smooth. If for each set S of at most 4 points in the plane, $V_{d_C}(S)$ and $V_{d_D}(S)$ have the same structure as embedded planar graphs, then D must be a scaled version of C. More generally, if for each set S of at most 4 points the Voronoi diagram $V_{d_C}(S)$ has the same combinatorial structure as $V_{d_D}(f(S))$, for some bijection f of the plane, then f is linear and $f(C) = D$ holds, up to scaling. Conversely, if f is a linear bijection of the plane then the Voronoi diagram of $f(S)$ with respect to the convex set $f(C)$ can be obtained by applying f to $V_{d_C}(S)$, for each site set S.

The image of the Euclidean circle under a linear bijection is an ellipse. The above result implies that convex distance functions based on ellipses are the only

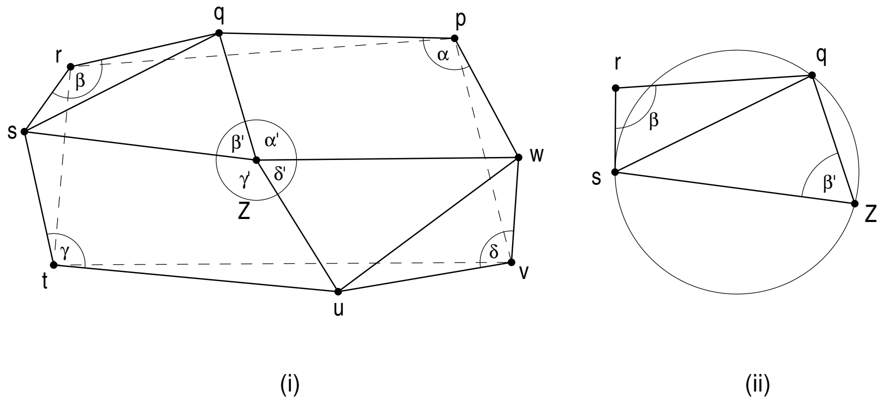


Figure 28. The graph shown in (i) is not a Euclidean Delaunay tessellation. In (ii) we have $\beta + \beta' < \pi$ iff r lies outside the circle. Equality holds iff r lies on the circle.

ones whose Voronoi diagrams can be obtained from the Euclidean Voronoi diagram of a transformed set of sites. The following theorem gives an even stronger reason why such a reduction is not possible for unit circles other than ellipses.

Theorem 4.8. *Let C be a strictly convex, compact set, symmetric about the origin, which is not an ellipse. Then there exists a set of 9 points in the plane whose Voronoi diagram with respect to d_C has a structure no Euclidean Voronoi diagram can achieve.*

In the proof given in [76] a Voronoi diagram based on d_C is constructed whose dual is either of the topological shape shown in fig. 28 (i), or of similar type. Let us assume that it can be realized by a Euclidean Delaunay tessellation; cf. section 2. Since r lies outside the circumcircle of q, s, z we have $\beta + \beta' < \pi$; see fig. 28 (ii). Similarly, $\delta + \delta' < \pi$ holds. Since p lies on the circumcircle of w, q, z we have $\alpha + \alpha' = \pi$, and $\gamma + \gamma' = \pi$ holds for the same reason. The primed angles at z add up to 2π. Therefore, we obtain

$$\alpha + \beta + \gamma + \delta < 2\pi.$$

But this is impossible because the points different from z must be in convex position (the convex hull of a point set equals the boundary of the unbounded face of its Delaunay triangulation) so that each of the angles at p, r, t, v includes an angle of the rectangle drawn by dashed lines. These angles add up to 2π.

The definitions of a convex distance function and of the L_p-norms stated above can easily be extended to dimensions higher than 2. In the remainder of this subsection we study *convex distance functions in 3-space*. Such a distance function is based on a convex, compact set C in 3-space which contains the origin in its interior. We call such a unit sphere C *good* if it is, in addition, strictly convex, symmetric, and smooth.

Some of the pleasant properties of 2-dimensional convex distance functions have their counterparts in 3-space. For example, the bisector $B(p,q)$ of two points is a surface homeomorphic to the plane, and two such bisector surfaces $B(p,q), B(p,r)$ have an intersection homeomorphic to the line, or empty. In these aspects, convex distance functions in 3-space do not differ from the 3-dimensional Euclidean distance. But whereas a Euclidean sphere is uniquely determined by four points in space, this is no longer true for convex distance functions. In Icking et al. [146] the following result has been shown.

Theorem 4.9. *For each $n > 0$ there exist a good convex set C and 4 points in 3-space such that there are $2n+1$ homothetic copies of C containing these points in their boundaries. This number does not decrease as the 4 points are independently moved within small neighborhoods.*

The center v of a scaled and translated copy of C containing p, q, r, s in its boundary is of the same d_C distance to each of these points. Hence, v is a Voronoi vertex in the diagram $V_{d_C}(\{p,q,r,s\})$. Therefore, theorem 4.9 implies that there exists no upper bound to the number of vertices of the Voronoi diagram of 4 points in 3-space, that holds for arbitrary convex distance functions.

If it is known that for a particular convex distance function d_C no more than k homothetic copies of C can pass through any four points in general position, then the complexity of the Voronoi diagram of n points based on d_C is in $O(kn^4)$, because at least 4 Voronoi regions meet at a Voronoi vertex. As an explicit function of k and n, nothing better than this trivial bound seems to be known, which is conjectured to be far off. For the Euclidean distance, for example, $k = 1$ holds, and the true complexity is $\Theta(n^2)$; see subsection 4.3.1.

Since the L_p norms are defined by algebraic equations of degree p, it is possible to say more about their Voronoi diagrams. From a general result by Sharir [232] on lower envelopes it follows that the complexity of the Voronoi diagram of n points in d-space under L_p is in $O(n^{d+\varepsilon})$. However, the constant in O tends to ∞, as p grows.

Lê [176] has obtained the following result on the number of L_p spheres that can pass through a set of given points. The proof uses results from the theory of additive complexity, see Benedetti and Risler [33]. This and the subsequent results require that the sites be in general position.

Theorem 4.10. *There exists an upper bound to the number of L_p spheres in d-space that can pass through $d+1$ given points, which depends only on d but not on p.*

More is known only for special cases. Recently, Boissonnat et. al. [45] have shown the following.

Theorem 4.11. *The Voronoi diagram of n points in 3-space based on the L_1 norm is of complexity $\Theta(n^2)$. The Voronoi diagrams of n points in d-space based on L_∞ or on a simplex as the unit sphere are both of complexity $\Theta(n^{\lceil d/2 \rceil})$.*

The unit spheres for the L_∞ and L_1 norm are hypercubes and crosspolytopes (i. e. duals of hypercubes), respectively. In contrast to hypercubes, crosspolytopes have a large number of facets in high dimensions, which partially explains why tight bounds for L_1-diagrams are only available for dimensions up to three.

Whereas for the complexity of the Voronoi diagram of a set of lines in 3-space under the Euclidean distance only an $O(n^{3+\varepsilon})$ upper bound is known (see [232]), Chew et. al. [67] were able to provide a smaller bound for lines under a polyhedral convex distance function.

Theorem 4.12. *Let C denote a convex polyhedron of constant complexity in 3-space. Then the Voronoi diagram of n lines based on d_C is of complexity $O(n^2\alpha(n)\log n)$. A lower bound is given by $\Omega(n^2\alpha(n))$.*

Here $\alpha(n)$ denotes the extremely slowly growing inverse of the Ackermann function.

4.5.3. Nice metrics

Convex distance functions do not apply to environments where the distance between two points can change under translations. This happens in the presence of obstacles, or in cities whose streets do not form a regular grid. In order to model such environments in a realistic way we can use the concept of a *metric*.

In this subsection we consider point sites in the plane, and Voronoi diagrams that are based on a metric, m. It associates with any two points, p and q, a non-negative real number $m(p,q)$ which equals 0 iff $p = q$. Moreover, $m(p,q) = m(q,p)$, and the triangle inequality $m(p,r) \leqslant m(p,q) + m(q,r)$ holds.

General metrics are quite a powerful modeling tool. Suppose there is an air-lift between two points a, b in the plane. Then

$$m(p,q) = \min \begin{cases} d(p,q) \\ d(p,a) + f + d(b,q) \\ d(p,b) + f + d(a,q) \end{cases}$$

describes how fast one can travel from p to q; we assume that going by car takes time equal to the Euclidean distance, d, whereas the flight takes time $f < d(a,b)$. It is easy to verify that the function m is a metric in the plane (we could warp the plane so that a and b have Euclidean distance f in 3-space, and glue on a straight handle connecting them; the function m describes the lengths of shortest paths in the resulting space).

Voronoi regions with respect to this metric m need not be connected. Suppose site q lies on the line segment $\overline{ab}$, as shown in fig. 29. In order to construct the Voronoi diagram of the two sites a and q we use the expanding waves approach

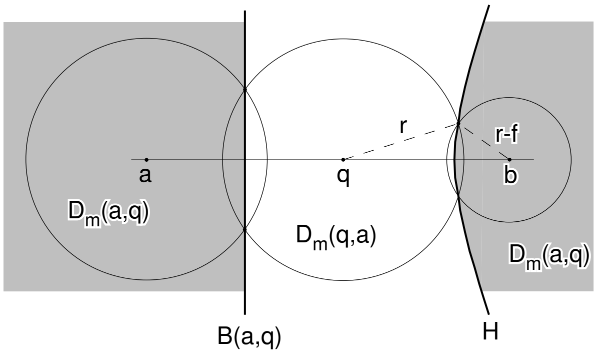


Figure 29. If there is an air line connecting a and b, the Voronoi region of a with respect to q is disconnected.

mentioned in section 2, and let two Euclidean circles expand from a and q at the same speed. Their intersections form the Euclidean bisector $B(a,q)$. As soon as the radius of these circles has reached f, the duration of the flight from a to b, a new circle starts growing from b. Subsequently, its radius is always by f smaller than the radius r of the other two circles. The intersections of the circles expanding from q and from b form a hyperbola, H; it is the locus of all points x satisfying $d(q,x) - f = d(b,x)$.

Such Voronoi diagrams of *points with additive weights* are also known as the *Johnson-Mehl model* or *Voronoi diagrams of circles*; compare the end of subsection 4.4.1. They have been studied e. g. by Lee and Drysdale [181] and Sharir [231]. Fortune [125] has shown how to construct them in time $O(n \log n)$ using the sweep line approach.

Clearly, the region of a with respect to metric m, $D_m(a,q)$, consists of the two parts shaded in fig. 29; they are not connected. In order to exclude such phenomena we restrict ourselves to a subclass of metrics introduced in Klein and Wood [170].

Definition 4.1. A metric m in the plane is called *nice* if it enjoys the following properties.

(i) A sequence p_i converges to p under m iff this holds under the Euclidean distance; the case $p = \infty$ is included.

(ii) For any two points p, r there exists a point q different from p and r such that $m(p,r) = m(p,q) + m(q,r)$ holds.

(iii) For any two points p, q is the common boundary of $D_m(q,p)$ and $B_m(p,q)$ a curve homeomorphic to the line. It is called a *bisector curve*, $J_m(p,q)$.

(iv) The intersection of two bisector curves $J_m(p,q), J_m(p,r)$ consists of only finitely many connected components.

Each symmetric, strictly convex distance function is a nice metric. Conversely, if a nice metric m is invariant under translations, and if $m(p,r) = m(p,q) + m(q,r)$ holds for any three consecutive points p, q, r on a line, then m is a convex distance function.

Whereas in definition 4.1 properties (iii) and (iv) are of technical nature, (i) and (ii) have an important structural consequence, due to results by Menger [196].

Lemma 4.7. *Let m be a nice metric in the plane. Then for any two points p, r there exists a path π connecting them, such that for each point q on π the equality $m(p,r) = m(p,q) + m(q,r)$ holds.*

Such paths are called *m-straight*; they are true generalizations of straight lines in the Euclidean distance: If one defines the m-length of a path π in a way analogous to the Euclidean path length (i. e., by adding up the m-distances between consecutive points on π, making the resolution arbitrarily fine) then the m-straight paths from p to r are precisely the paths of minimum m-length, which equals $m(p,r)$.

Voronoi diagrams based on nice metrics have a nice structural property.

Lemma 4.8. *Let m be a nice metric. Then each Voronoi region $VR_m(p,S)$ is connected: Each m-straight path π from p to some point $x \in VR_m(p,S)$ is fully contained in $VR_m(p,S)$. That is, the Voronoi regions under a nice metric m are* m-star-shaped *as seen from their sites.*

Proof. Suppose that some point y on π does not belong to $\mathrm{VR}_m(p,S)$. Then there must be a site $q \in S$ different from p such that $m(q,y) \leqslant m(p,y)$ holds, and by the straightness of π we obtain

$$\begin{aligned} m(p,x) &= m(p,y) + m(y,x) \\ &\geqslant m(q,y) + m(y,x) \geqslant m(q,x), \end{aligned}$$

a contradiction to the fact that x lies in the Voronoi region of p. This shows that π is in fact contained in $\mathrm{VR}_m(p,S)$. Lemma 4.7 ensures that for each x in the region of p at least one such m-straight path from p to x exists. Consequently, $\mathrm{VR}_m(p,S)$ is a connected set. $\square$

Clearly, in the example of the air-lift metric, property (ii) of definition 4.1 is violated: All points q satisfying $m(a,b) = m(a,q) + m(q,b)$ that are not equal to one of the two airports, a and b, are in mid air!

Examples of nice metrics are *composite metrics* that result from assigning different convex distance functions to the regions of a planar subdivision, or the *Karlsruhe* (or: *Moscow*) *metric*; see Klein [165]. Here a center point, 0, is fixed and only such paths are allowed that consist of pieces of circles around 0 and of pieces of radii from 0. The Karlsruhe metric $m(p,q)$ denotes the minimum Euclidean length of an allowed path from p to q.

It depends on the angle between p and q if the shortest connecting path runs through the center, or around it; see fig. 30 (i). The bisector of two points consists of up to eight segments that are pieces of straight lines or hyperbolae if written in

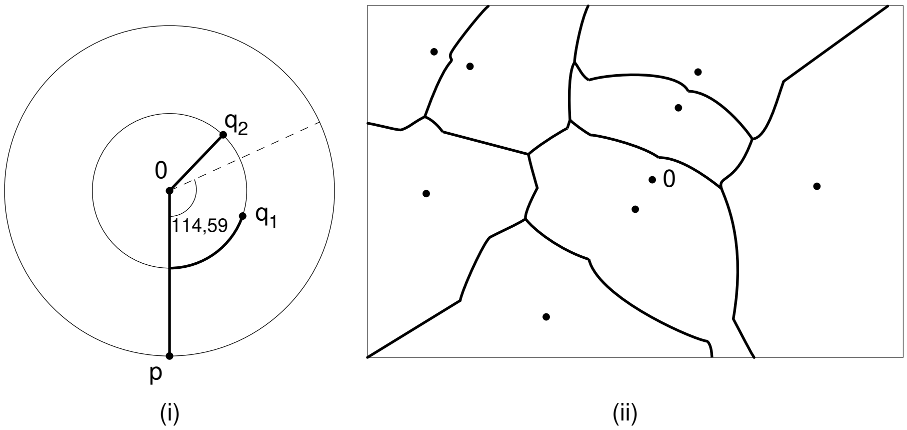


Figure 30. The shortest path in the Karlsruhe metric runs through the center iff the angle between p and q exceeds $114,59\ldots°$. In (ii) a Voronoi diagram in the Karlsruhe metric is shown.

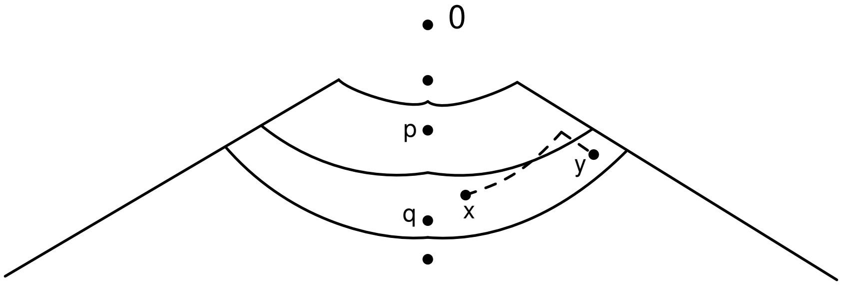


Figure 31. The unique shortest path between x and y does not stay within the region of q.

polar coordinates. In (ii), the Voronoi diagram of 8 point sites under the Karlsruhe metric is shown.

Whereas the star-shapedness of the Voronoi regions is preserved, the example of the Karlsruhe metric m shows that the convexity of the Euclidean Voronoi regions has no counterpart in nice metrics. In fig. 31 the Voronoi diagram of 4 point sites on the same radius is shown. There exists exactly one m-straight path connecting the points x, y of the Voronoi region of q, but this path is not contained in the region of q.

That the Voronoi regions under a nice metric m are m-star-shaped is a property strong enough for computing the Voronoi diagram efficiently. The following has been shown by Dehne and Klein [84].

Theorem 4.13. *The Voronoi diagram of n point sites under a nice metric in the plane can be constructed within $O(n \log n)$ many steps, using the sweep line approach.*

It is also possible to apply the randomized incremental construction method introduced in subsection 3.2, or the divide & conquer technique. With the latter a new difficulty arises: If we use a line to subdivide S into subsets L and R then the set $B_m(L, R)$ of Voronoi edges separating L-regions from R-regions can contain cycles that would not be detected during the merge phase.

In [165] two criteria have been introduced for $B_m(L, R)$ to be acyclic. For example, if the m-circles are simply-connected we can use, as a divider of S, any curve l homeomorphic to the line whose intersection with all m-circles is connected (possibly empty). This generalizes a result by Chew and Drysdale [66] on convex distance functions, where each straight line possesses this property.

4.6. General Voronoi diagrams

One might wonder if there exist some intrinsic properties most Voronoi diagrams have in common, and if these properties are a foundation strong enough for a unifying approach to the definition and to the construction of Voronoi diagrams.

Edelsbrunner and Seidel [113] have introduced a very general approach. It does not explicitly use the concepts of sites and distance functions. Rather, the site set, S, is a mere set of indices $\{p, q, \ldots\}$. A fixed domain, X, is given, and for each $p \in S$ a real-valued function $f_p : X \to \mathbf{R}$. The graph

$$H_p = \{(x, f_p(x)) \mid x \in X\}$$

of f_p can be considered a hypersurface in the space $X \times \mathbf{R}$. Now the Voronoi diagram of $(S, \{f_p \mid p \in S\})$ is defined to be the projection onto X of the lower envelope

$$L = \{(x, \min_{p \in S} f_p(x)) \mid x \in X\}$$

of the arrangement $\{H_p \mid p \in S\}$ of hypersurfaces.

At the end of subsection 3.5 we have already studied an example of this approach: If S denotes a set of points in $D = \mathbf{R}^2$, and if the functions f_p are defined by $f_p(x) = d(p, x)$, we obtain an arrangement of cones whose lower envelope projects onto the Euclidean Voronoi diagram of S. More importantly, subsection 4.3.2 shows that *linear* functions f_p exist whose envelope projects onto the Voronoi (or power) diagram of S. Moreover, the order-k Voronoi (or power) regions of S are the projections of certain cells of the corresponding arrangement, see subsection 4.3.3.

To establish this connection between Voronoi diagrams and arrangements of hypersurfaces in a space one dimension higher is one of the great advantages of this approach. As one implication, an upper bound of $O(n^{d+\varepsilon})$ on the combinatorial complexity of general classes of Voronoi diagrams follows from a result in Sharir [232] on envelopes of hypersurfaces; see subsections 4.4.1 and 4.5.2.

A different approach to planar Voronoi diagrams has been suggested in Klein [165]. Again, there are no physical sites but an index set, S. Instead of distance functions or their graphs in 3-space, *bisecting curves* are used as primary objects.

Let us assume that for any two different indices p, q in S, a bisecting curve $J(p,q) = J(q,p)$ homeomorphic to the line is given, which cuts the plane into two unbounded, open domains, $D(p,q)$ and $D(q,p)$. In the subsequent pictures, the index p denotes the "halfplane" $D(p,q)$.

As in definition 2.1, the Voronoi region $\mathrm{VR}(p,S)$ is defined as the intersection of the open domains $D(p,q)$, where $q \in S \setminus \{p\}$. The Voronoi diagram $V(S)$ of $\{J(p,q) \mid p,q \in S \text{ and } p \neq q\}$ is defined as the union of all boundaries at least two Voronoi regions have in common. $V(S)$ is called the *abstract Voronoi diagram* with respect to the bisecting curves $J(p,q)$.

Definition 4.2. The system $\mathcal{J} = \{J(p,q) \mid p,q \in S \text{ and } p \neq q\})$ is called *admissible* iff for each subset S' of S of size at least 3 the following conditions are fulfiled.

(i) The Voronoi regions $\mathrm{VR}(p,S')$ are pathwise connected.

(ii) Each point of the plane lies in a Voronoi region $\mathrm{VR}(p,S')$, or on the Voronoi diagram $V(S')$.

(iii) The intersection of two curves $J(p,q)$ and $J(p,r)$ consists of only finitely many components.

From these conditions important structural properties of "concrete" Voronoi diagrams can be derived, as the following example shows. For simplicity, we assume that two bisecting curves cross transversally wherever they intersect.

Lemma 4.9. *Let $\mathcal{J}$ be an admissible system, and suppose that $J(p,q)$ and $J(p,r)$ cross at the point x. Then $J(q,r)$ also passes through x, in the way shown in fig. 32.*

Proof. In a suitable neighborhood of x the curves $J(p,q)$ and $J(p,r)$ run as depicted in fig. 32 (i). Let $S' = \{p,q,r\}$. Precisely the shaded area belongs to $\mathrm{VR}(p,S')$, by definition of the abstract Voronoi diagram. Point y is not contained in $\mathrm{VR}(r,S')$ because it lies in $D(p,r)$. By moving y a little we can make sure that it is not on the curve $J(q,r)$. Now condition (ii) of definition 4.2 implies $y \in \mathrm{VR}(q,S')$. Similarly, z must belong to the Voronoi region of r. But if, in each neighborhood of x, points of $\mathrm{VR}(q,S') \subset D(q,r)$ and of $\mathrm{VR}(r,S') \subset D(r,q)$ can be found in the wedges containing y and z, respectively, then $J(q,r)$ must run through x in the way shown in picture (ii) of fig. 32. □

The *finite* abstract Voronoi diagram (i. e. the result of encircling it with a closed curve Γ, as proposed in section 2) is a 2-connected planar graph with at most $n+1$

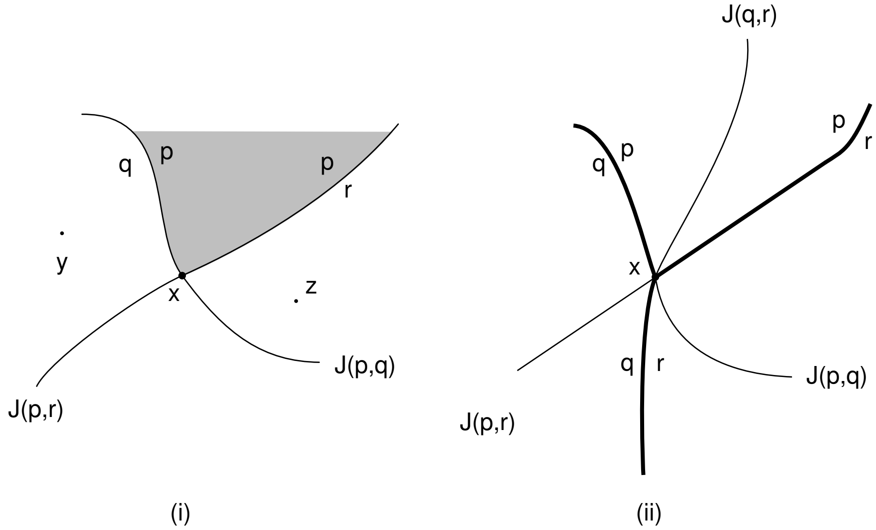


Figure 32. Where two related bisecting curves $J(p,q), J(p,r)$ meet, the third one, $J(q,r)$, also passes through, like in a Euclidean Voronoi diagram.

faces, all of whose vertices have degree at least 3 (exactly 3 for the vertices on Γ). Each such graph can be obtained as an abstract Voronoi diagram.

One can show that definition 4.2 holds for all subsets S' of S if it is satisfied for each subset S' of size 3. The conditions are met, for example, by the bisectors of power diagrams (subsection 4.3.2), of sites with respect to a nice metric in the plane (subsection 4.5.3), of sites with additive weights, and of non-intersecting line segments or convex polygons (subsection 4.4.1); see Meiser [195]. Not covered by this approach are e. g. non-convex polygonal sites, or point sites with multiplicative weights (the Apollonius model); in either case the bisector curves can be closed, rather than homeomorphic to the line.

Another example has been provided by Abellanas et al. [1]. They studied disjoint compact, convex sites in the plane and their *Hausdorff Voronoi diagram* based on a convex distance function d_C, which puts each point x in the Voronoi region of the first site that is fully contained in a circle C expanding from x; cf. subsection 4.5.2. It can be shown that the resulting regions are pathwise connected. If, in addition, all convex sets have semi-algebraic boundaries then also property (iii) in definition 4.2 is fulfiled.

Of the four algorithmic methods mentioned in section 3 for computing the Euclidean Voronoi diagram, two have been adapted to abstract Voronoi diagrams. How to generalize the divide & conquer approach has been shown in [165]. As in the case of nice metrics (subsection 4.5.3) one has to assume that the index set, S, can recur-

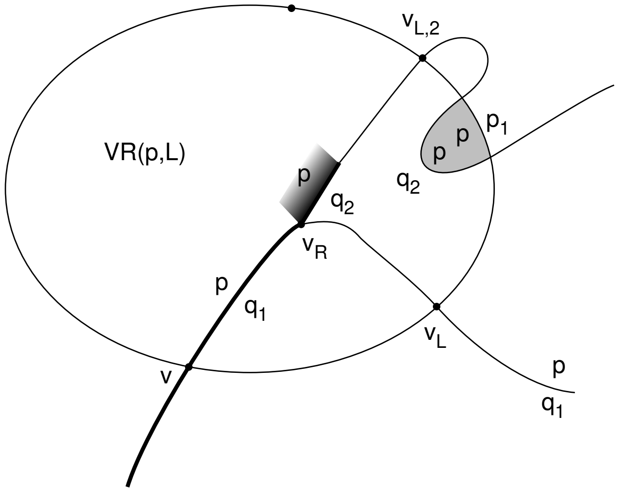


Figure 33. If $J(p, q_2)$ ran like this, the disconnected shaded areas would both belong to $\mathrm{VR}(p, \{p, p_1, q_2\})$, a contradiction! Therefore, the first point, $v_{L,2}$, on the boundary of $\mathrm{VR}(p, L)$ can still be found by scanning the boundary from v_L on counterclockwise.

sively be split into subsets L and R such that $B(L, R)$, the set of edges separating L- from R-regions, contains no cycle, in order to find the starting edges of $B(L, R)$ at infinity. Once a starting edge has been found, $B(L, R)$ can be traced efficiently, because abstract Voronoi regions are pathwise connected; no stronger properties, like star-shapedness or convexity, are needed. This is illustrated in fig. 33, which should be compared to fig. 10.

It has been shown in Mehlhorn et al. [194] and in Klein et al. [169] that one can, without further assumptions, apply the randomized incremental construction technique to abstract Voronoi diagrams.

Theorem 4.14. *The abstract Voronoi diagram of an admissible system* $\mathcal{J} = \{J(p,q) \mid p, q \in S$ *and* $p \neq q\})$, *where* $|S| = n$, *can be constructed within expected* $O(n \log n)$ *many steps and expected space* $O(n)$, *by randomized incremental construction.*

This algorithm provides a universal tool that can be used for computing all types of concrete Voronoi diagrams whose bisector curves have the properties required in definition 4.2. To adapt the algorithm to a special type one has to implement only certain elementary operations on bisector curves. For example, it is sufficient to have a subroutine that accepts 5 elements of S as input, and returns the graph structure of their abstract Voronoi diagram. All numerical computations can be kept to this subroutine.

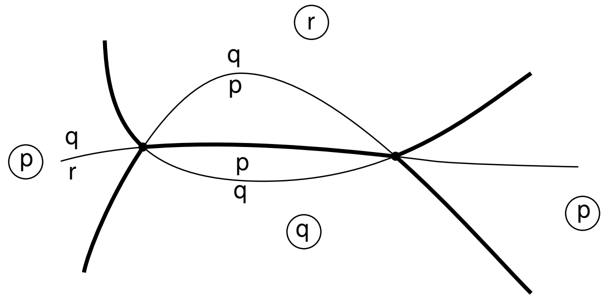


Figure 34. An inverse abstract Voronoi diagram. The region of p is not connected.

It seems difficult to apply the sweep approach to abstract Voronoi diagrams. One reason is the absence of sites whose detection by the sweep line could indicate that a new region must be started.

Abstract Voronoi diagrams can also be thought of as lower envelopes, in the sense mentioned at the beginning of this subsection. Namely, for each point x not situated on a bisecting curve, the relation

$$p <_x q \Leftrightarrow x \in D(p, q)$$

defines a total ordering on S. If we construct a set of surfaces H_p, $p \in S$, in 3-space such that H_p is below H_q iff $p <_x q$ holds, then the projection of their lower envelope equals the abstract Voronoi diagram.

The concept of abstract Voronoi diagrams has been generalized to 3-space in Lê [177], under structural restrictions; for example, any two bisecting surfaces $J(p, q)$, $J(p, r)$ must have an intersection homeomorphic to a line, or empty. These assumptions apply e. g. to convex distance functions whose unit sphere is an ellipsoid; compare subsection 4.5.2.

Rasch [217] has introduced *abstract inverse* (or: *furthest-site, order* $n-1$) *Voronoi diagrams* in the plane; cf. subsection 4.3.3. Here the region of p is defined as the intersection of all open domains $D(q, p)$, where $q \in S \setminus \{p\}$. In contradistinction to the Euclidean case, the resulting regions are in general not pathwise connected; a situation that could also arise in the concrete furthest site diagram of line segments is shown in fig. 34.

If the admissible system $\mathcal{J}$ is such that none of the regular abstract Voronoi regions is empty then the inverse diagram with respect to $\mathcal{J}$ is a tree of complexity $O(n)$. It can be constructed within expected $O(n \log n)$ many steps and expected linear storage.

5. Geometric applications

This section is devoted to some of the numerous applications of the Voronoi diagram $V(S)$ and its dual, the Delaunay triangulation $\mathrm{DT}(S)$, in solving geometric problems. Some applications have been mentioned previously in this chapter, mostly along with generalizations of the classical type. Applications in relation to optimization properties of the Delaunay triangulation can be found in subsection 4.2.

Whereas distance is implicitly involved in almost all applications of the Voronoi diagram, we start with some examples of "pure" distance problems. These distance problems are, apart from their direct use in practical applications that will be mentioned occasionally, frequently arising subroutines in more involved geometric algorithms.

5.1. Distance problems

Unless otherwise stated, the following problems address point sites in the plane under the standard Euclidean distance. However, most solutions can be generalized, at least to convex distance functions; see subsection 4.5.2.

5.1.1. Post office

Our first example is the *post office* (or: *nearest neighbor*) problem mentioned in the seminal paper [230] by Shamos and Hoey. Given a set of n sites (post offices) in the plane, determine, for an arbitrary query point x, the post office closest to x.

The Voronoi diagram of the post offices represents the *locus approach* to the post office problem: it partitions the plane into regions whose points x have identical nearest post offices.

It remains to quickly determine the region that contains the query point, x. This *point location* problem has received a lot of interest in computational geometry; it is the topic of a separate chapter of this book the reader is refered to. Here we sketch an elementary solution first mentioned in Dobkin and Lipton [100], called the *slab method*.

Through each Voronoi vertex a horizontal line is drawn. These extra lines partition the Voronoi regions into triangles and trapezoids. By construction, no horizontal "slab" between two consecutive lines contains a Voronoi vertex in its interior, so that all crossing Voronoi edges are ordered within the slab. To locate a query point, $x = (x_1, x_2)$, we use one binary search for x_2 among the slabs, and another one for x_1 among the edge segments of the slab found. This gives us $O(\log n)$ query time, at quadratic storage cost.

There are more efficient techniques like e. g. Kirkpatrick [162] or Edelsbrunner et al. [109] that need only linear storage and can be derived in linear time from the Voronoi diagram. Together with any of the optimal Voronoi diagram algorithms presented in section 3 this yields the following solution to the post office problem.

Theorem 5.1. *Given a set S of n point sites in the plane, one can, within $O(n \log n)$ time and linear storage, construct a data structure that supports nearest neighbor queries: for an arbitrary query point x, its nearest neighbor in S can be found in time $O(\log n)$.*

Storing the history of the Voronoi diagram during incremetal insertion (cf. subsection 3.2) even obviates the need of processing the diagram for point location. Guibas et al. [135] showed that nearest neighbor queries are supported in (expected) time $O(\log^2 n)$ by the resulting structure.

Analogously, the order-k Voronoi diagram introduced in subsection 4.3.3 can be used for finding the k nearest neighbors in S of a query point x. Aurenhammer and Schwarzkopf [31] generalized the practical approach in [135], obtaining a dynamic structure with expected query time $O(k \log^2 n)$, that allows both insertion and deletion of sites, at a space requirement of $O(k(n-k))$; see also subsection 4.3.3. A sophisticated technique of compacting order-k Voronoi diagrams in Aggarwal et al. [5] achieves optimal query time $O(k + \log n)$ and space $O(n)$, but with high constants.

The locus approach also works for more general sites and distance measures. Let us assume that the site set, S, consists of k non-intersecting convex polygons with n edges in total. The bisector of two sites is composed of $O(n)$ many straight or parabolic segments; if we also count the endpoints of such segments as vertices (of degree 2), then the Voronoi diagram of the k convex polygons is of complexity $\Theta(n)$, as lemma 4.4 shows. It could be constructed in time $O(n \log n)$ by first computing the line segment Voronoi diagram of the polygon edges (see subsection 4.4.1), and then joining those regions that belong to parts of the same convex polygon.

A more efficient structure for solving the post office problem of k disjoint convex polygons has been introduced in McAllister et al. [190]. Using only $O(k)$ many line segments, it partitions the plane into regions bounded by at most 6 edges each. With each region one or two sites are associated: for any point x in the region, one of them is the nearest neighbor of x in S.

This planar straight line graph can be defined quite easily. Its vertices are the $O(k)$ original Voronoi vertices defined by the k polygonal sites, plus, for each Voronoi vertex v of 3 sites, the closest points to v on their respective boundaries. Straight line edges run from each vertex v to its closest points and, around the boundary of each site, between the points closest to its Voronoi vertices; see fig. 35. For each site C in extreme position, a halfline to infinity is added all of whose points have the same closest point on the boundary of C.

Theorem 5.2. *Given a set S of k disjoint convex polygonal sites in the plane with a total of n edges, one can, within $O(k \log n)$ time and $O(k)$ storage, construct a data structure that allows nearest neighbor queries to be answered within time $O(\log n)$.*

The essential task is in constructing the $O(k)$ many original Voronoi vertices of the k polygons spending only $O(\log n)$ time on each vertex. This can be achieved by a clever adaption of the sweep line approach described in subsection 3.4, and by

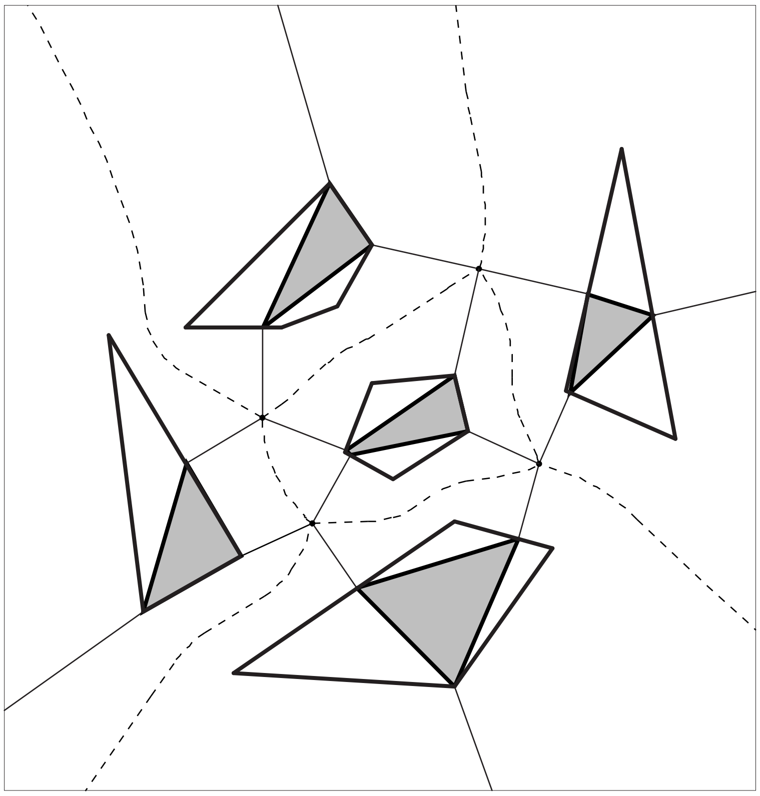


Figure 35. A linear subdivision of size $O(k)$ for solving the post office problem of convex polygons.

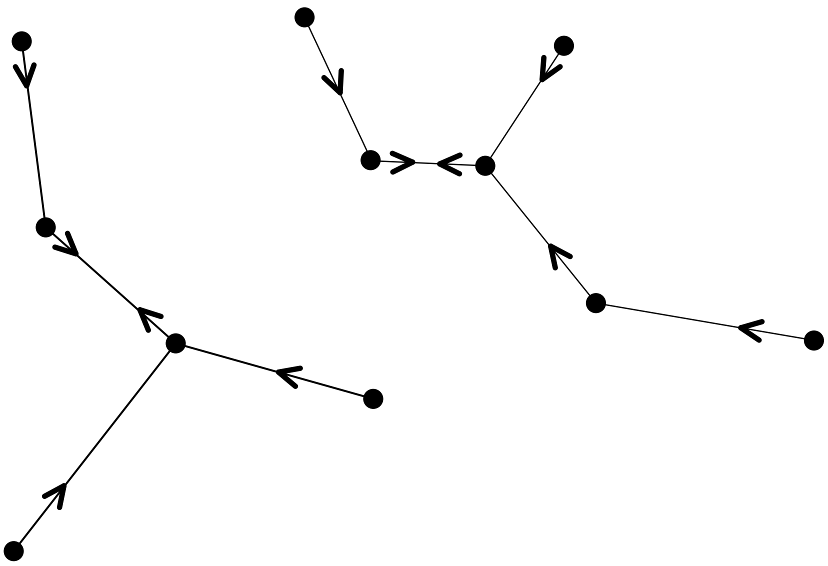


Figure 36. The nearest neighbor graph of 11 points.

means of a subroutine that computes, in $O(\log n)$ time, the Voronoi vertex of three convex polygons. The subroutine uses the *tentative prune and search* technique described in Kirkpatrick and Snoeyink [159].

The same approach works for convex distance functions; see subsection 4.5.2.

5.1.2. Nearest neighbors and the closest pair

Another distance problem the Voronoi diagram directly solves is the *all nearest neighbors* problem: For each point site in the set S, a nearest neighbor in S is required. An example is shown in fig. 36; arrows are pointing from each site towards its nearest neighbor. For sites in general position the resulting nearest neighbor graph is a forest of trees each of whose vertices is of outdegree 1 and indegree at most 6.

The solution offered by the Voronoi diagram is based on the following fact.

Lemma 5.1. *Let $S = P \cup Q$ be a disjoint decomposition of the point set S, and let $p_0 \in P$ and $q_0 \in Q$ be such that*

$$d(p_0, q_0) = \min_{p \in P, q \in Q} d(p, q).$$

Then the Voronoi regions of p_0 and q_0 are edge-adjacent in $V(S)$.

Proof. Otherwise, the line segment $\overline{p_0 q_0}$ contains a point z that belongs to the closure of some Voronoi region $\mathrm{VR}(r, S)$, where $r \neq p_0, q_0$. Let us assume that r belongs to Q; the case $r \in P$ is symmetric. From $z \in \overline{\mathrm{VR}(r, S)}$ follows $d(z, r) \leqslant d(z, q_0)$, hence

$$\begin{aligned} d(p_0, r) &\leqslant d(p_0, z) + d(z, r) \\ &\leqslant d(p_0, z) + d(z, q_0) = d(p_0, q_0) \leqslant d(p_0, r) \end{aligned}$$

by minimality of $d(p_0, q_0)$. Hence, equality must hold, and we obtain $d(p_0, r) = d(p_0, q_0)$ and $d(z, r) = d(z, q_0)$.

But since p_0 lies on the bisector $B(q_0, r)$, each point z in the interior of $\overline{q_0 p_0}$ is strictly closer to q_0 than to r, a contradiction. □

If we apply lemma 5.1 to the subsets $\{p\}$ and $S \setminus \{p\}$ it follows that the nearest neighbor of p is sitting in a neighboring Voronoi region (i. e. it is a Delaunay neighbor). Therefore, it is sufficient to inspect, for each $p \in S$, all neighbors of p in the Voronoi diagram, and select the closest of them. This way, each edge of $V(S)$ will be accessed twice. Since their total number is linear, due to lemma 2.3, we can solve the all nearest neighbor problem by constructing the Voronoi diagram.

Theorem 5.3. *Given a set S of n points in the plane, $O(n \log n)$ time and linear space are sufficient for determining, for each $p \in S$, a nearest neigbor in S.*

Once for each p its nearest neighbor in S is known, we can easily determine a pair of points whose distance is a minimum.

Corollary 5.1. *The* closest pair *among n points in the plane can be determined within $O(n \log n)$ time and linear space.*

Hinrichs et al. [142] have shown that it suffices to maintain certain parts of the Voronoi diagram by a sweep algorithm (cf. subsection 3.4), in order to detect the closest pair.

Whereas finding the closest pair seems an easier task than constructing the Voronoi diagram, it still has an $\Omega(n \log n)$ lower bound, by reduction from the ε-closeness problem; compare theorem 3.1.

In dimension d, constructing the Voronoi diagram is no longer the method of choice for finding the closest pair, due to its exponentially increasing size; see subsection 4.3.2. Already in 1976, Bentley and Shamos [35] provided an $O(n \log n)$ algorithm for arbitrary fixed dimensions. Golin et al. [131] showed that randomization plus use of the floor fuction (which is forbidden in the algebraic decision tree model of computation) allows the closest pair of n points in d-space to be found within expected $O(n)$ time, for any fixed d.

The problem of finding the closest pair of n points can be generalized to *reporting the k closest pairs in S*, for some number $k \leqslant \binom{n}{2}$. A brute-force solution would sort all interpoint distances in time $O(n^2 \log n)$ and then report the k smallest. Time $O(n^2)$ is sufficient if one uses the selection method for finding, in time $O(m)$, the

k-th smallest of m objects. Smid [237] has shown how to enumerate the $O(n^{2/3})$ smallest distances in time $O(n \log n)$ and linear space.

A simple yet elegant solution using the Delaunay triangulation DT(S) of S has been provided by Dickerson et al. [90]. Their approach is as follows.

The closest pair in S corresponds to the shortest edge of the Delaunay triangulation DT(S), as lemma 5.1 shows. But the k-th closest pair (p, q) need not define an edge of DT(S), not even for small values of k. However, there exists a path of Delaunay edges connecting p and q that are of length smaller than $d(p, q)$ each.

This can be used for proving the following elegant algorithm correct. It maintains a queue Q of pairs of points by distance. Initially, Q contains all Delaunay neighbors. In the i-th step, the closest pair (p, q) in Q is removed and reported i-th closest. Then, for all Delaunay edges $\overline{op}$ satisfying $d(o, p) \leqslant d(p, q)$, the pair (o, q) is inserted into Q; similarly, each pair (p, r) is added to Q where $\overline{qr}$ is a Delaunay edge of length at most $d(p, q)$.

The performance of this algorithm is stated in the next theorem.

Theorem 5.4. *The k closest pairs of a set S of n points can be computed in time $O((n + k) \log n)$ and space $O(n + k)$.*

A similar approach even works efficiently in higher dimensions. Dickerson and Eppstein [91] showed that several interdistance enumeration problems for a point set S in d-space can be solved by means of DT(S). The prohibitive size of DT(S) in d-space is reduced by augmenting S to have a linear-size, bounded degree Delaunay triangulation, with the method of Bern et al. [38].

5.1.3. Largest empty and smallest enclosing circle

Suppose someone wants to build a new residence within a given area, as far away as possible from each of n sources of disturbance.

If the area is modeled by a convex polygon A over m vertices, and the disturbing sites by a point set S, we are looking for the largest circle with center in A that does not contain a point of S. The task of determining this circle has been named the *largest empty circle* problem.

Shamos and Hoey [230] observed that this circle must have its center at a Voronoi vertex of $V(S)$, or at the intersection of a Voronoi edge with the boundary of A, or at a vertex of P. For, if a circle C with center $x \in A$ contains no $p \in S$, not even on its boundary, we blow it up until it does. If its boundary now contains three sites then x is a Voronoi vertex; see lemma 2.1. If there are two sites on its boundary, x lies on a Voronoi edge. In this case we move x along their bisector away from the two sites until the expanding circle hits a third site, or x hits the boundary of A. If, in the beginning, only one point site $p \in S$ lies on the boundary of C then we expand C, while keeping its boundary in contact with p, until x reaches a vertex of A, or one of the before-mentioned events occurs.

This observation leads to the following result.

Theorem 5.5. *The largest circle not containing a point of S, whose center lies* *inside a convex polygon A, can be found within $O((m+n)\log n)$ time and linear space. Here, m denotes the number of vertices of A, and n is the size of S.*

Proof. We spend $O(n \log n)$ time on constructing the Voronoi diagram $V(S)$ and a point location structure, as mentioned in subsection 5.1.1, so that $O(m \log n)$ total time is sufficient for finding, for each vertex of A, its nearest neighbor in S. The largest empty circles centered at the Voronoi vertices can be determined in constant time each. Finally, we can trace the convex boundary of A through the Voronoi diagram, in order to detect its intersections with the Voronoi edges, in time $O(n+m)$. $\square$

The *smallest enclosing circle* problem asks for the circle of minimum radius that encloses a given set S of n points in the plane. Applications are the placement of a least powerful transmitter station that can reach a given set of locations, or the placement of a plant that minimizes the maximum distance to the customers.

As was observed in Bhattacharya and Toussaint [40], there are two cases. Either the smallest enclosing circle contains three points of S on its boundary, or it runs through two antipodal points whose distance equals the diameter of S.

From the convex hull of S its diameter, together with a pair of points realizing it, can be derived in linear time; then it can be checked if the smallest circle through these points contains S.

Otherwise, the center v of the smallest enclosing circle C must be a vertex of the *furthest site* (or: *order* $n-1$) Voronoi diagram introduced in subsection 4.3.3. Namely, the three sites on the boundary of C are furthest from v.

This yields an $O(n \log n)$ algorithm for computing the smallest enclosing circle. The linear programming technique by Megiddo [193] is more efficient: For fixed dimension d it allows the smallest sphere enclosing n points in d-space to be constructed in time $O(n)$. A more practical way of achieving linear (randomized) time is the elegant *minidisk* algorithm by Welzl [255]; here the constant in O has been shown to be a subexponential function of d by Matoušek et al. [186].

5.2. Subgraphs of Delaunay triangulations

The Delaunay triangulation $\mathrm{DT}(S)$ of a set S of n point sites contains, as subgraphs, various structures with diverse applications. One example, the nearest-neighbor graph of S, has already been discussed in subsection 5.1.2. Efficient algorithms for computing these structures follow from the fact that $\mathrm{DT}(S)$ has size $O(n)$, and can be constructed in $O(n \log n)$ time, in the plane. This positive effect is partially lost in higher dimensions as $\mathrm{DT}(S)$ may be the complete graph on S.

5.2.1. Minimum spanning trees and cycles

A minimum spanning tree, $\mathrm{MST}(S)$, of S is a planar straight line graph on S which is connected and has minimum total edge length. This structure plays an important role, for instance, in transportation problems, pattern recognition, and clustering. Shamos and Hoey [230] first observed the connection to Delaunay triangulations.

Lemma 5.2. *MST(S) is a subgraph of DT(S).*

Proof. Let e be an edge of MST(S). Removal of e splits MST(S) into two subtrees, and S into two subsets S_1 and S_2. Clearly, e is the shortest edge connecting S_1 and S_2. A shorter edge, if existent, would lead to a spanning tree shorter than MST(S). It follows that the circle C with diameter e is empty of sites in S. A site enclosed by C would have to belong to either S_1 or S_2, leading to a connection between S_1 and S_2 shorter than e. By definition 2.2 in section 2, C proves e Delaunay. □

We thus can select the $n-1$ edges of MST(S) from the $O(n)$ edges of DT(S), rather than from all the $\binom{n}{2}$ edges spanned by S, by standard application of Kruskal's [171] or Prim's [214] greedy algorithms. Edges are considered in increasing length order, and an edge is classified as a tree edge if it does not violate acyclicity. An $O(n \log n)$ time algorithm for computing MST(S) is obtained.

Lemma 5.2 generalizes to metrics different from the Euclidean (for example L_1, see Hwang [145]), and to higher dimensions. When staying in the plane, equally efficient algorithms result. In higher dimensions, however, DT(S) may not be of much use for the construction of MST(S), because of the possibly quadratic size. Subquadratic worst-case time algorithms for computing MST(S) in d-space exist (Yao [257] and Agarwal et al. [3]), but it still remains open whether an $O(n \log n)$ time algorithm can be developed, at least for 3-space.

A *traveling salesman tour*, TST(S), for a set S of point sites in the plane is a minimum length cycle passing through all the sites. It is known that TST(S) is no subgraph of DT(S), in general. Dillencourt [92] showed that DT(S) need not even contain any cycle through all sites in S (a Hamiltonian cycle), and that finding Hamiltonian cycles in Delaunay triangulations is NP-complete [94]. He also gives a partial explanation for the fact that DT(S) is Hamiltonian in most cases [93].

Another optimization graph which is known to be not part of DT(S) is a *minimum length matching* for S; see Akl [11]. However, DT(S) is guaranteed to contain some perfect matching, see Dillencourt [93], a property not shared by all triangulations of S.

Finding a traveling salesman tour has been shown to be NP-complete in Papadimitriou [209], but a factor-2 length approximation of TST(S) can be found easily using MST(S). Let $A(S)$ be a cycle through S that results from traversing MST(S) in preorder. Rosenkrantz et al. [220] observed the following. Let $|X|$ be the total length of a set X of edges.

Lemma 5.3. $|A(S)| < 2 \cdot |TST(S)|$.

Proof. When traversing each edge of MST(S) twice, a tour longer than $A(S)$ is obtained. Hence $|A(S)| < 2 \cdot |\text{MST}(S)|$. To see $|\text{MST}(S)| < |\text{TST}(S)|$, note that removing an edge from TST(S) leaves a path which is some, but not neccessarily the minimum, spanning tree of S. □

$A(S)$ can be constructed in linear time from MST(S). A more sophisticated construction of a spanning cycle by means of MST(S) is given in Christofides [69]. An approximation factor of 1.5 is achieved, at an expense of roughly $O(n^2\sqrt{n})$ in construction time.

Another NP-complete problem, which has a factor-2 approximation by means of MST(S), is the construction of optimal radii graphs; see Huang [143]. Let R be a real-valued vector that associates each site $p \in S$ with an individual radius $r_p > 0$. The corresponding *radii graph*, $G_R(S)$, contains an edge between two sites $p, q \in S$ iff $d(p,q) \leqslant \min\{r_p, r_q\}$. The optimization problem now asks for a radii vector R^* such that $G_{R^*}(S)$ is connected and $|R^*| = \sum_{p\in S} r_p^*$ is minimum. Radii graphs have applications in the design of radio networks.

Consider the vector R that takes, as a radius for each site $p \in S$, the length of the longest edge of MST(S) incident to p. Then $G_R(S)$ is connected; it contains MST(S) as a subgraph.

Lemma 5.4. $|R| < 2 \cdot |R^*|$

Proof. Let $e = (p,q)$ be some edge of MST(S). Then e appears at most twice as a radius in R, namely if e is the longest edge incident to p and the longest edge incident to q. This gives $|R| < 2 \cdot |\mathrm{MST}(S)|$.

To verify $|\mathrm{MST}(S)| \leqslant |R^*|$, let us consider $G_{R^*}(S)$. As being connected, this graph contains some spanning tree, T. We may orient the edges of T such that each site $p \in S$ (except one) has a unique edge e_p of T pointing at it. As e_p is also an edge of $G_{R^*}(S)$, we have $r_p^* \geqslant |e_p|$. Hence $|R^*| > |T| \geqslant |\mathrm{MST}(S)|$. □

Lemma 5.4 remains true for more general measures $|R| = \sum_{p\in S} f(r_p)$, for any increasing function f. This is because MST(S) also minimizes $\sum_{e\in T} f(|e|)$, for all spanning trees T of S. The case $f(r) = r^2$ is relevant to the application mentioned above, as the received power of a radio terminal decreases with the square of the distance.

5.2.2. α-shapes

Extracting the shape of a given set S of point sites in the plane is a problem that arises, for example, in data visualization and pattern recognition. To some extent, the shape of S is reflected by the convex hull of S. The edges of the convex hull are part of the Delaunay triangulation DT(S). The concept of *α-shape*, introduced in Edelsbrunner et al. [110] (see also [104]), generalizes the convex hull of S for the sake of better shape approximation, while still remaining a subgraph of DT(S).

For $\alpha > 0$, the *α-shape* $\alpha(S)$ of S is defined to contain an edge between sites $p, q \in S$ iff there is some disc of radius α that avoids S and has p and q on its boundary. By definition 2.2 in section 2, $\alpha(S)$ is always part of DT(S) and thus has only $O(n)$ edges.

For α being sufficiently large, the convex hull of S is obtained. The smaller is the value of α, the finer is the level of resolution of the shape of S; see fig. 37. $\alpha(S)$

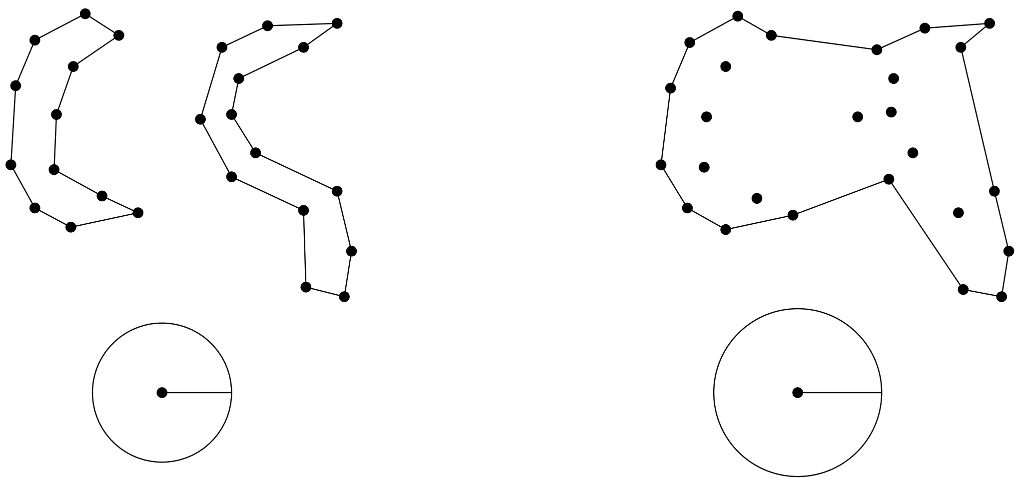


Figure 37. α-shapes for different values of α.

contains no edges if α is smaller than the distance of the closest pair in S.

The definition of $\alpha(S)$ can be extended to negative values of α, by requiring that the discs of radius $-\alpha$ *fully contain* S. In this case, approximations of the shape of S more crude than the convex hull are obtained, and the edges of $\alpha(S)$ appear in the dual triangulation of the *furthest-site Voronoi diagram* of S; see subsection 4.3.3.

It follows that all the edges of the whole family of α-shapes for S, for $-\infty < \alpha < \infty$, are contained in two triangulations of size $O(n)$. For each triangulation edge e, an interval of activity can be specified, containing all values of α such that $\alpha(S)$ contains e. These activity intervals can be computed in $O(n \log n)$ time, and allow us to extract $\alpha(S)$ for an input value α in $O(n)$ time.

Another nice feature of α-shapes is their flexibility. The notion of α-shape, along with its relationship to Delaunay triangulations, nicely generalizes to higher dimensions. α-shapes in 3-space are of particular interest, and implementations have been used in several areas of science and engineering; see Edelsbrunner and Mücke [111].

Also, the concept of α-shape can be generalized to represent different levels of detail at different parts of space; see Edelsbrunner [107]. This is achieved by weighting the sites in the given set S individually. The corresponding *weighted α-shapes* are subgraphs of the *regular triangulation* for the weighted set S which, in turn, is the dual of the *power diagram* for this set; see subsection 4.3.2. The combinatorial and algorithmic properties of weighted α-shapes are similar to that of their unweighted counterparts.

A different approach for reconstructing shapes in 3-space by means of Delaunay triangulations is persued in Boissonnat and Geiger [44]. They exploit additional information on the sites available in certain applications, namely that S is contained in

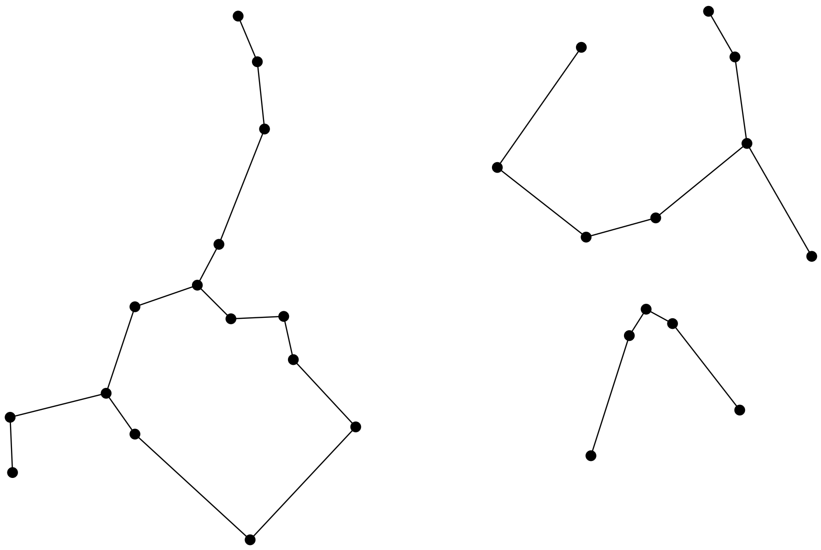


Figure 38. β-skeleton for $\beta = 1.2$

k parallel planes (corresponding to cross sections of the object to be reconstructed). The Delaunay edges connecting sites in neighbored planes can be computed in an output-sensitive manner; see Boissonnat et al. [42]. The edges describing the final shape are selected according to several criteria. Geiger [128] reports that satisfactory shapes are produced even for complicated medical images.

5.2.3. β-skeletons

Another family of graphs, defined by empty discs and thus consisting of subgraphs of DT(S), is the family of *β-skeletons* of a set S of point sites. β-skeletons have been introduced in Kirkpatrick and Radke [163] as a class of empty neighborhood graphs, and have recently received increased attention because of their relation to minimum-weight triangulations. Actually, in [163] both a circle-based and a lune-based version of β-skeletons are proposed. Here we will consider only the former, as defined below.

In contrast to α-shapes, the radii of the empty discs defining a β-skeleton are not fixed but depend on the interpoint distances in S. Let $p, q \in S$. For $\beta \geqslant 1$, the edge (p, q) is included in the β-skeleton $\beta(S)$ iff the two discs of diameter $\beta \cdot d(p, q)$ that pass through both p and q are empty of sites in S. See fig. 38.

Note that $\beta(S)$ is a subgraph of $\beta'(S)$ if $\beta > \beta'$. For $\beta = 1$, the *Gabriel graph* of S is obtained. Whereas $\beta(S)$ may be disconnected for any $\beta > 1$, the Gabriel graph is always connected as it contains the minimum spanning tree of S as a subgraph;

cf. the proof of lemma 5.2. It is easy to see that the Gabriel graph just consists of those edges of DT(S) that cross their dual Voronoi edges. This observation yields an $O(n \log n)$ algorithm for its construction. In fact, general β-skeletons can be constructed from DT(S) in linear time; see Jaromczyk et al. [149] and Lingas [185]. Gabriel graphs have proved useful in the processing of geographical data; see Gabriel and Sokal [126] and Matula and Sokal [187].

Recently it has been realized that, for β large enough, $\beta(S)$ is contained in the *minimum-weight triangulation*, MWT(S), of S (a triangulation having minimum total edge length). The original bound $\beta \geqslant \sqrt{2}$ in Keil [157] has been improved in Cheng and Xu [61] to $\beta > 1.1768$, which is close to the value $2/\sqrt{3}$ for which a counterexample is known. Apart from the edges of the convex hull of S, only the shortest edge in S has been known to be part of MWT(S) before.

One of the heuristics for the intriguing problem of constructing MWT(S) first computes $\beta(S)$, for the smallest admissable value of β. The obtained k connected components of $\beta(S)$ then can be completed length-optimally to a triangulation, using dynamic programming, in $O(n^{k+2})$ time; see Cheng et al. [60]. Thus, if $\beta(S)$ is connected or has only a constant number of components, MWT(S) can be constructed in polynomial time. Unfortunately, k tends to be linear in n.

A slight modification of the empty neighborhood of an edge in S gives rise to the class of *relative neighborhood graphs* for S. Applications in pattern recognition are reported in Toussaint [247]. These graphs are also constructable in $O(n \log n)$ time by exploiting their containment in DT(S); see Supowit [242] and Kirkpatrick and Radke [163].

5.2.4. Shortest paths

Let S be a set of n point sites in the plane. A (connected) planar straight line graph G on S is said to have *dilation* t if, for any $p, q \in S$, the length of the shortest path in G between p and q is at most $t \cdot d(p, q)$. In this case, G is also called a t*-spanner* (of the complete Euclidean graph) for S. Sparse t-spanners that have only $O(n)$ edges are of special interest in the field of robotics.

The minimum spanning tree MST(S), though being optimally sparse, may have dilation $\Theta(n)$. In contrast, DT(S) is sparse but 'sufficiently connected' to exhibit constant dilation, independently from the size n of S. Consequently, good spanners for S can be constructed in $O(n \log n)$ time. A dilation of $t \approx 5$ for DT(S) is proved in Dobkin et al. [98], a result which has been strengthened to $t \approx 2.5$ in Keil and Gutwin [158]. An easy lower bound is $t \geqslant \pi/2$.

Surprisingly, better upper bounds can be achieved by taking the Delaunay dual of generalized Voronoi diagrams. The convex distance function (subsection 4.5.2) whose unit circle is an equilateral triangle leads to a triangulation with $t = 2$. This was shown in Chew [64], along with the following result concerning the *constrained* Delaunay triangulation (subsection 4.4) for the L_1-metric: There is always a path between two given sites, whose length is at most $\sqrt{10}$ times the *geodesic* distance with respect to the constraining line segments. This yields an $O(n \log n)$ time algorithm for computing constant approximations of optimal paths in the presence of polygonal obstacles.

Alternative triangulations of S are known to exhibit good spanner properties. Examples are the greedy triangulation and the minimum-weight triangulation; see Das and Joseph [78]. However, only for $\mathrm{DT}(S)$ are there algorithms available that are worst-case efficient and easy to implement.

In higher dimensions, $\mathrm{DT}(S)$ may loose its sparseness and thus is of minor interest for computing spanners. Different techniques have been used with success; see e.g. Vaidya [249] and Chandra et al. [57].

5.3. Geometric clustering

Clustering a set of data means finding subsets (called *clusters*) whose in-class members are similar, and whose cross-class members are dissimilar, according to a predefined similarity measure. Data clustering is important in diverse areas of science, and various techniques have been developed; see e.g. Hartigan [137]. In many situations, similarity of objects has a geometric interpretation in terms of distances between points in d-space.

The rôle of Voronoi diagrams in the context of clustering is manyfold. For certain applications, the relevant cluster structure among the objects is well reflected, in a direct manner, by the structure of the Voronoi diagram of the corresponding point sites; see e.g. Ahuja [7]. For instance, dense subsets of sites give rise to Voronoi regions of small area (or volume). Regions of sites in a homogeous cluster will have similar shape. For clusters having a direction-sensitive density, the regions will exhibit an extreme width in the corresponding direction.

Perhaps more important is the fact that numerous types of *optimal* clusterings are induced by Voronoi diagrams, and/or can be computed by first computing a Voronoi diagram, for a certain set of sites. This set need not coincide with the set of points to be clustered.

A general distiction is between *off-line* and *on-line* clustering methods. The former methods assume the availability of the whole data set before starting, whereas the latter cluster the data upon arrival, according to an on-line classification rule. Though Voronoi diagrams play a role in on-line clustering, see e.g. Hertz et al. [140], we restrict attention to off-line methods below.

5.3.1. Partitional clusterings

Let P be a set of n points in d-space. A k-clustering of P is a partition of P into k subsets (clusters) $C_1, \ldots, C_k$. The dissimilarity of a single cluster C is measured by an appropriate *intra-cluster* criterion $\mu(C)$, which may be variance, diameter, radius, etc. The dissimilarity of the whole clustering, in turn, is expressed as a (usually monotone) function f of $\mu(C_1), \ldots, \mu(C_k)$, called the *inter-cluster* criterion. Common examples for f are the maximum or the sum. $C_1, \ldots, C_k$ is called *optimal* if $f(\mu(C_1), \ldots, \mu(C_k))$ is minimal for all possible k-clusterings of P.

If k is part of the input, the problem of finding an optimal k-clustering is NP-complete in general, even in the plane; see Capoyleas et al. [55] for references. For fixed k, polynomial-time algorithms are known for various criteria. This is due to the fact that optimal clusters are separable in a certain sense, and in several cases are induced by the regions of (generalized) Voronoi diagrams. This approach was first systematically used in [55].

A common intra-cluster criterion is *variance*,

$$\mu(C) = \frac{1}{|C|} \sum_{p,q \in C} d(p,q)^2.$$

Variance can be rewritten as $\sum_{p \in C} d(p, s(C))^2$, with $s(C)$ being the centroid (mass center) of C. Note that $s(C)$ is the point x in d-space that minimizes the *squared distance* $\sum_{p \in C} d(p,x)^2$ to C. Hence, if the inter-cluster criterion f is sum, the clustering problem is equivalent to the *k-centroid problem*: Given a set P of points to be clustered, find a set S of k sites (cluster centroids) that minimizes $\sum_{p \in P} d(p,S)^2$, where $d(p,S)$ is the distance from p to the closest site in S. This implies that the optimal k-clustering $C_1^*, \ldots, C_k^*$ has the following nice property; see Boros and Hammer [47]. For $1 \leqslant i \leqslant k$, C_i^* is contained in the region of $s(C_i^*)$ in the Voronoi diagram of $S = \{s(C_1^*), \ldots, s(C_k^*)\}$.

The candidates for an optimal k-clustering of P thus are the possible partitions of P induced by a Voronoi diagram of k point sites. The number of such partitions is $n^{O(dk)}$ in d-space; see Inaba et al. [147]. This implies a polynomial-time algorithm for computing an optimal k-clustering for fixed k, which proceeds by enumerating all candidate solutions and comparing their dissimilarities.

The problem becomes considerably easier when a set S of k 'cluster centers' is fixed in advance. A clustering minimizing the sum of the squared distances of the clusters to their centers is easily found by constructing the Voronoi diagram of S. Of more interest is the case where the *sizes* of the k clusters are prescribed. Now a corresponding optimal clustering is induced by a power diagram of S (subsection 4.3.2), a fact which leads to an algorithm with roughly $O(k^2 n)$ runtime if P is a set of n points in the plane; see Aurenhammer et al. [29].

In Inaba et al. [147], optimal k-clusterings for the intra-cluster measure $\mu(C) = \sum_{p,q \in C} d(p,q)^2$ (like variance, but without division by $|C|$) are considered. Again, these clusterings are induced by a kind of power diagram of the cluster centroids, weighted additively by $|C|$ and multiplicatively by $\mu(C)$.

Another popular measure $\mu(C)$ is the radius of the smallest sphere enclosing C. If maximum is taken as the inter-cluster criterion, the *k-center problem* is obtained: Find a set S of k centers such that $r_S = \max_{p \in P} d(p,S)$ is minimized. In other words, choose a set S of centers such that P can be covered by k spheres of minimum radius r_S. (This simplifies to the smallest enclosing sphere problem for $k = 1$; see subsection 5.1.3). It is easy to see that the Voronoi diagram of S gives rise to optimal clusters. Capoyleas et al. [55] observed that an optimal k-clustering for μ, for any monotone increasing inter-cluster criterion f, is induced by the power diagram of the enclosing spheres.

For both intra-cluster measures μ above, polynomial-time algorithms for fixed k

result from considering the $n^{O(dk)}$ candidate clusterings.

For $k = 2$, the 'Voronoi property' of the 2-centroid or 2-center problem just means linear separability of the two optimal clusters. However, linear separability for all pairs of clusters in a given clustering does not always imply that this clustering has a realization by means of Voronoi or power diagrams. An example is the intra-cluster criterion diameter in 2-space [55].

The problem of constructing linearly (or circularly) separable clusterings in the plane, where each cluster C of P is separable by a straight line (or circle) from $P \backslash C$, is addressed in Dehne and Noltemeier [85] and Heusinger and Noltemeier [141]. They give a polynomial-time algorithm for deciding the existence, and finding an optimal clustering, with prescribed cluster sizes. A correspondence between separable clusters and regions of higher-order Voronoi diagrams for P (subsection 4.3.3) is exploited.

The order-k Voronoi diagram, $V_k(P)$, of P is also useful for selecting from P a k-sized cluster C^* of minimal dissimilarity $\mu(C^*)$. This approach is persued in Dobkin et al. [97] and Aggarwal et al. [6]. For instance, if μ is variance, C^* has a non-empty region in $V_k(P)$. If μ is diameter, C^* is contained in a subset of P having a non-empty region in $V_{3k-3}(P)$. These properties hold in arbitrary d-space and, in the plane, lead to efficient cluster selection algorithms. Moreover, if $\mu(C)$ measures the perimeter of the axis-aligned enclosing square, or rectangle, of a cluster C, higher-order Voronoi diagrams in the L_∞-metric (subsection 4.5.2) yield improved solutions. Except for the diameter case, the running time of all these algorithms is dominated by the cost of computing an order-k diagram in the plane.

5.3.2. Hierarchical clusterings

Hierarchical methods are based solely on a given *inter-cluster distance* δ. They cluster a set S of n points as follows. Initially, each point is considered to be a cluster itself. As long as there are two or more clusters, a pair C, C' of clusters is joined into one cluster if $\delta(C, C')$ is minimum for all cluster pairs.

A frequently used inter-cluster distance is the *single-linkage distance*

$$\delta(C, C') = \min\{d(p, q) \mid p \in C, q \in C'\}.$$

As was pointed out in Shamos and Hoey [230], constructing the single-linkage hierarchy for S just means simulating the greedy algorithm of Kruskal [171] for computing the minimum spanning tree $\mathrm{MST}(S)$ of S . We thus obtain a time bound of $O(n \log n)$ in the plane, and subquadratic bounds in d-space; see subsection 5.2.1.

After having performed $n - k$ steps in constructing the hierarchy (that is, after having added the $n - k$ shortest edges to $\mathrm{MST}(S)$), the intermediate k-clustering $C_1, \ldots, C_k$ is optimal in the following sense; see Asano et al. [18]: It maximizes the minimum single-linkage distance between the clusters, for all possible k-clusterings of S.

The practical value of single-linkage clusterings is, however, restricted by the fact that the produced clusters tend to exhibit large dissimilarity in terms of variance, radius, and diameter. To remedy this deficiency, other inter-cluster distances have been used. Among them is the *complete-linkage distance*

$$\delta(C, C') = \max\{d(p, q) \mid p \in C, q \in C'\}.$$

Constructing the complete-linkage hierarchy for S efficiently is much more elusive. A direct approch leads to an $O(n^3)$ time and $O(n)$ space algorithm in general d-space. (The frequently cited $O(n^2)$ time insertion algorithm in Defays [82] only approximates the hierarchy. Its output depends on the insertion order.)

In the plane, it seems of advantage to use, as an auxiliary structure, the Voronoi diagram of the intermediate clusters $C_1, \ldots, C_k$, defined by the *Hausdorff distance* $h(x, C) = \max\{d(x, p) \mid p \in C\}$ of a point x to a cluster C; see e.g. Edelsbrunner et al. [108] and Huttenlocher et al. [144]. Unfortunately, the closest pair of clusters does not necessarily yield adjacent regions in this diagram.

Recently, an $O(n \log^2 n)$ time algorithm has been given in Krznaric and Levcopoulos [172]. Among other structures, they use the Delaunay triangulation $\mathrm{DT}(S)$ and the furthest-site Voronoi diagram of the points in the intermediate clusters C_i. They also show that approximations of the complete-linkage hierarchy for S can be obtained from $\mathrm{DT}(S)$ in $O(n)$ time.

5.4. Motion planning

Suppose that for a disc-shaped robot centered at some start point, s, a motion to some target point, t, must be planned in the presence of n line segments as obstacles. We assume that the line segments are pairwise disjoint, and that there are four line segments enclosing the scene, as shown in fig. 39. While the robot is navigating through a gap between two line segments, l_1 and l_2, at each position x its "clearence", i. e. its distance

$$d(x, l_i) = \min\{d(x, y); y \in l_i\}$$

to the obstacles, should be a maximum. This goal is achieved if the robot maintains the same distance to either segment! In other words, the robot should follow the bisector $B(l_1, l_2)$ of the line segments l_1 and l_2 until its distance to another obstacle gets smaller than $d(x, l_i)$.

Roughly, this observation implies that the robot should walk along the edges of the Voronoi diagram $V(S)$ of the line segments in $S = \{l_1, \ldots, l_n\}$. This diagram is connected, due to the four surrounding line segments.

If start and target point are both lying on $V(S)$, the motion planning task immediately reduces to a discrete graph problem: After labeling each edge of $V(S)$ with its minimum distance to its two sites, and adding s and t as new vertices to $V(S)$, a breadth first search from s will find, within $O(n)$ time, a path to t in $V(S)$ whose minimum label is a maximum. If this value exceeds the robot's radius, a collision-free motion has been found.

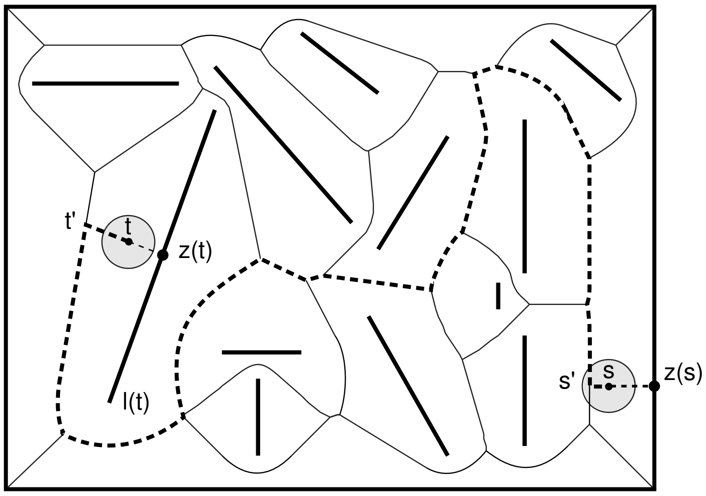


Figure 39. Moving a disk from s to t in the presence of barriers is faciliated by their Voronoi diagram.

If the target point, t, does not lie on $V(S)$, we first determine the line segment $l(t)$ whose Voronoi region contains t; to this end, point location techniques as mentioned in subsection 5.1.1 can be applied. Next, we find the point $z(t)$ on $l(t)$ that is closest to t; see fig. 39. If its distance to t is less than the robot's radius then the robot cannot be placed at t and no motion from s to t exists. Otherwise, we consider the ray from $z(t)$ through t. It hits a point t' on $V(S)$ which serves as an intermediate target point.

Similarly a point s' can be defined if the original start point, s, does not lie on $V(S)$.

Theorem 5.6. *The robot can move from s to t without collision iff its radius does not exceed one of the distances $d(z(t), t)$ and $d(z(s), s)$, and if there exists a collision-free motion from s' to t' along edges of the Voronoi diagram $V(S)$.*

Proof. Suppose the conditions are fulfiled. Then the straight motion from s to s' does not cause a collision since the robot's distance to its closest obstacle is ever increasing. The same holds for t and t'. Combining these pieces with a safe motion from s' to t' yields the desired result. Conversely, let us assume that a path π from s to r exists along which the robot can move without hitting one of the line segments.

The mapping $f : t \mapsto t'$ can be extended not only to s but to all points x of the scene (leaving fixed exactly the points of $V(S)$). One can show that f is *continuous.* Consequently, path π is mapped by f onto a path π' in $V(S)$ that runs from s' to t'. For each point x on π its corresponding point x' on π' is even farther away from its closest obstacle. □

This "retraction" approach has first been taken by Ó'Dúnlaing and Yap [206]. The Voronoi diagram of n line segments plus a structure suitable for point location in $V(S)$ can be preprocessed in time $O(n \log n)$ time. Afterwards, for each pair (s, t) of start and target point, the above method can find a safe motion from s to t within time $O(n)$, if it exists, or report otherwise.

If approximating the robot's shape by a disc is not accurate enough, a compact convex set C can be used, with some fixed reference point in its interior. As long as only *translational* motions are considered, the retraction approach still works; only the Voronoi diagram of the obstacles must now be based on the convex distance function defined by the convex set C^* that results from reflecting C about the reference point; compare subsection 4.5.2.

If the scene consists of polygonal obstacles the same situation occurs as in the post office problem; see subsection 5.1.1: Even though the number k of obstacles may be small, the complexity of their Voronoi diagram increases in proportion with their total number n of edges. Again, the piecewise-linear approximation of the Voronoi diagram by McAllister et al. [190] comes to the rescue; it is of complexity $O(k)$ and can be constructed within time $O(k \log n)$ for the Euclidean distance, and in time $O(k \log n \log m)$ for a distance function based on a convex m-gon (that is, for solving the translational motion planning problem of a convex robot with m edges).

If the robot is also allowed rotational motions, Voronoi diagrams can still be applied. Ó'Dúnlaing et al. [204, 205] have studied the problem of moving a line segment amidst polygonal obstacles.

Further results on the use of Voronoi diagrams in motion planning can be found in the surveys [13, 14] by Alt and Yap, in Yap [258], and in Canny [53, 54].

6. Concluding remarks and open problems

Although this chapter is quite long and, as we believe, reasonably comprehensive, we are aware that we did not cover all aspects of the Voronoi diagram and related topics. Below are some topics that we feel are important and interesting but that have not been treated in detail here.

A significant stream of research concerns the *stochastic properties* of Voronoi diagrams for randomly distributed sites. Motivation stems from their relevance in the natural and social sciences. The interested reader is refered to chapters 5 and 8 of Okabe et al. [208] for a comprehensive treatment. Also, the survey paper [27] could be consulted for a brief overview. In the present chapter, randomization enters only in the design of efficient construction algorithms. We decided not to include more stochastic aspects because their relevance to computational geometry is less strong.

In view of the large scope of applications of Voronoi diagrams, fast construction is an important issue. *Parallelization* is a tool to reach this goal. We did not discuss parallel methods for constructing Voronoi diagrams in detail; a separate chapter of this book, devoted to parallel computational geometry, will partially cover this gap.

Though a host of applications of the Voronoi diagram has been covered, *applications in the natural and social sciences* have been mentioned only marginally in the chapter. The book by Okabe et al. [208] puts strong emphasis on these applications, which led us, as being no experts in these areas, to refer to this book instead.

Finally let us mention, out of many interesting open problems concerning Voronoi diagrams, a few that seem most important to us in view of their practical applications.

Among the few algorithmic problems for *planar* Voronoi diagrams that have not been solved in a satisfactory manner is the computation of the *geodesic Voronoi diagram* (subsection 4.4.4). Apart from the instance of simple polygons [210], the only known subquadratic solution is [198]. Is there a simple and practically efficient construction method, with a good theoretical bound, $O(n \log n)$?

The insert & flip algorithms for computing *Delaunay triangulations in 3-space* (subsection 4.3.1) in [151], [215], and [115] are simple and elegant, but suffer from the fact that intermediate triangulations might be quite large compared to the final result. This cannot happen if the set of sites to be inserted is well distributed, but this requirement need not be met in typical applications like contour modeling. Does there always exist an 'output-sensitive' insertion order not running into the above problem?

Despite its importance in practice, the *minimum spanning tree in 3-space* (subsection 5.2.1) has eluded efficient construction so far. The existing subquadratic solutions [257], [3], are far from easy implementation. A partial answer are the efficient expected-time and approximation algorithms in [72], [248]. Is there a practical *and* worst-case efficient algorithm for computing exact minimum spanning trees in 3-space?

References

[1] M. Abellanas, G. Hernandez, R. Klein, V. Neumann-Lara, and J. Urrutia. Voronoi diagrams and containment of families of convex sets on the plane. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 71–78, 1995.

[2] P. K. Agarwal, M. de Berg, J. Matoušek, and O. Schwarzkopf. Constructing levels in arrangements and higher order Voronoi diagrams. In *Proc. 10th Annu. ACM Sympos. Comput. Geom.*, pages 67–75, 1994.

[3] P. K. Agarwal, H. Edelsbrunner, O. Schwarzkopf, and E. Welzl. Euclidean minimum spanning trees and bichromatic closest pairs. *Discrete Comput. Geom.*, 6(5):407–422, 1991.

[4] A. Aggarwal, L. J. Guibas, J. Saxe, and P. W. Shor. A linear-time algorithm for computing the Voronoi diagram of a convex polygon. *Discrete Comput. Geom.*, 4(6):591–604, 1989.

[5] A. Aggarwal, M. Hansen, and T. Leighton. Solving query-retrieval problems by compacting Voronoi diagrams. In *Proc. 22nd Annu. ACM Sympos. Theory Comput.*, pages 331–340, 1990.

[6] A. Aggarwal, H. Imai, N. Katoh, and S. Suri. Finding k points with minimum diameter and related problems. *J. Algorithms*, 12:38–56, 1991.

[7] N. Ahuja. Dot pattern processing using Voronoi polygons as neighborhoods. *IEEE Trans. Pattern Anal. Mach. Intell.*, 4:336–343, 1982.

[8] O. Aichholzer, D. Alberts, F. Aurenhammer, and B. Gärtner. A novel type of skeleton for polygons. *J. Universal Comput. Sci.*, 1:752–761, 1995.

[9] O. Aichholzer and F. Aurenhammer. Straight skeletons for general polygonal figures. Technical Report 432, Inst. for Theor. Comput. Sci., Graz Univ. of Technology, Graz, Austria, 1995.

[10] O. Aichholzer, F. Aurenhammer, S.-W. Cheng, N. Katoh, G. Rote, M. Taschwer, and Y.-F. Xu. Triangulations intersect nicely. *Discrete Comput. Geom.*, to appear.

[11] S. G. Akl. A note on Euclidean matchings, triangulations, and spanning trees. *Journal of Combinatorics, Information and System Sciences*, 8(3):169–174, 1983.

[12] H. Alt and O. Schwarzkopf. The Voronoi diagram of curved objects. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 89–97, 1995.

[13] H. Alt and C. K. Yap. Algorithmic aspect of motion planning: a tutorial, part 1. *Algorithms Rev.*, 1(1):43–60, 1990.

[14] H. Alt and C. K. Yap. Algorithmic aspect of motion planning: a tutorial, part 2. *Algorithms Rev.*, 1(2):61–77, 1990.

[15] N. M. Amato and E. A. Ramos. On computing Voronoi diagrams by divide-prune-and-search. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, to appear.

[16] B. Aronov. On the geodesic Voronoi diagram of point sites in a simple polygon. *Algorithmica*, 4:109–140, 1989.

[17] B. Aronov. Personal communication, 1996.

[18] T. Asano, B. Bhattacharya, J. M. Keil, and F. Yao. Clustering algorithms based on minimum and maximum spanning trees. In *Proc. 4th Annu. ACM Sympos. Comput. Geom.*, pages 252–257, 1988.

[19] P. Ash, E. Bolker, H. Crapo, and W. Whiteley. Convex polyhedra, Dirichlet tessellations, and spider webs. In M. Senechal and G. Fleck, editors, *Shaping Space: A Polyhedral Approach*, chapter 17, pages 231–250. Birkhäuser, Boston, MA, 1988.

[20] P. F. Ash and E. D. Bolker. Recognizing Dirichlet tesselations. *Geom. Dedicata*, 19:175–206, 1985.

[21] M. J. Atallah. Some dynamic computational geometry problems. *Comput. Math. Appl.*, 11:1171–1181, 1985.

[22] F. Aurenhammer. Power diagrams: properties, algorithms and applications. *SIAM J. Comput.*, 16:78–96, 1987.

[23] F. Aurenhammer. Recognising polytopical cell complexes and constructing projection polyhedra. *J. Symbolic Comput.*, 3:249–255, 1987.

[24] F. Aurenhammer. Improved algorithms for discs and balls using power diagrams. *J. Algorithms*, 9:151–161, 1988.

[25] F. Aurenhammer. Linear combinations from power domains. *Geom. Dedicata*, 28:45–52, 1988.

[26] F. Aurenhammer. A new duality result concerning Voronoi diagrams. *Discrete Comput. Geom.*, 5:243–254, 1990.

[27] F. Aurenhammer. Voronoi diagrams: A survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23:345–405, 1991.

[28] F. Aurenhammer and H. Edelsbrunner. An optimal algorithm for constructing the weighted Voronoi diagram in the plane. *Pattern Recogn.*, 17:251–257, 1984.

[29] F. Aurenhammer, F. Hoffmann, and B. Aronov. Minkowski-type theorems and least-squares partitioning. In *Proc. 8th Annu. ACM Sympos. Comput. Geom.*, pages 350–357, 1992.

[30] F. Aurenhammer and H. Imai. Geometric relations among Voronoi diagrams. *Geom. Dedicata*, 27:65–75, 1988.

[31] F. Aurenhammer and O. Schwarzkopf. A simple on-line randomized incremental algorithm for computing higher order Voronoi diagrams. *Internat. J. Comput. Geom. Appl.*, 2:363–381, 1992.

[32] D. Avis, B. K. Bhattacharya, and H. Imai. Computing the volume of the union of spheres. *Visual Comput.*, 3:323–328, 1988.

[33] R. Benedetti and J.-J. Risler. *Real algebraic and semi-algebraic sets.* Actualités mathématiques. Hermann, Paris, 1990.

[34] J. L. Bentley and T. A. Ottmann. Algorithms for reporting and counting geometric intersections. *IEEE Trans. Comput.*, C-28:643–647, 1979.

[35] J. L. Bentley and M. I. Shamos. Divide-and-conquer in multidimensional space. In *Proc. 8th Annu. ACM Sympos. Theory Comput.*, pages 220–230, 1976.

[36] J. L. Bentley, B. W. Weide, and A. C. Yao. Optimal expected-time algorithms for closest-point problems. *ACM Trans. Math. Softw.*, 6:563–580, 1980.

[37] M. Bern and D. Eppstein. Mesh generation and optimal triangulation. In D.-Z. Du and F. K. Hwang, editors, *Computing in Euclidean Geometry*, volume 1 of *Lecture Notes Series on Computing*, pages 23–90. World Scientific, Singapore, 1992.

[38] M. Bern, D. Eppstein, and J. Gilbert. Provably good mesh generation. *J. Comput. Systems Sci.*, 48:384–409, 1994.

[39] J. Bernal. Bibliographic notes on Voronoi diagrams. Technical report, National Institute of Standards and Technology, Gaithersburg, MD 20899, 1992.

[40] B. K. Bhattacharya and G. T. Toussaint. On geometric algorithms that use the furthest-point Voronoi diagram. In G. T. Toussaint, editor, *Computational Geometry*, pages 43–61. North-Holland, Amsterdam, Netherlands, 1985.

[41] G. Blelloch, G. L. Miller, and D. Talmor. Developing a practical projection-based parallel Delaunay algorithm. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, to appear.

[42] J.-D. Boissonnat, A. Cérézo, O. Devillers, and M. Teillaud. Output-sensitive construction of the 3-d Delaunay triangulation of constrained sets of points. In *Proc. 3rd Canad. Conf. Comput. Geom.*, pages 110–113, 1991.

[43] J.-D. Boissonnat, O. Devillers, R. Schott, M. Teillaud, and M. Yvinec. Applications of random sampling to on-line algorithms in computational geometry. *Discrete Comput. Geom.*, 8:51–71, 1992.

[44] J.-D. Boissonnat and B. Geiger. Three dimensional reconstruction of complex shapes based on the Delaunay triangulation. In R. S. Acharya and D. B. Goldgof, editors, *Biomedical Image Processing and Biomedical Visualization*, volume 1905, pages 964–975. SPIE, 1993.

[45] J.-D. Boissonnat, M. Sharir, B. Tagansky, and M. Yvinec. Voronoi diagrams in higher dimensions under certain polyhedral distance functions. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 79–88, 1995.

[46] J.-D. Boissonnat and M. Teillaud. On the randomized construction of the Delaunay tree. *Theoret. Comput. Sci.*, 112:339–354, 1993.

[47] E. Boros and P. L. Hammer. On clustering problems with connected optima in Euclidean spaces. Technical Report, RUTCOR, Ruters University, New Brunswick, New Jersey, Mar. 1988.

[48] E. Brisson. Representing geometric structures in d dimensions: Topology and order. In *Proc. 5th Annu. ACM Sympos. Comput. Geom.*, pages 218–227, 1989.

[49] K. Q. Brown. Voronoi diagrams from convex hulls. *Inform. Process. Lett.*, 9:223–228, 1979.

[50] K. Q. Brown. *Geometric transforms for fast geometric algorithms.* Ph.D. thesis, Dept. Comput. Sci., Carnegie-Mellon Univ., Pittsburgh, PA, 1980. Report CMU-CS-80-101.

[51] C. Burnikel, K. Mehlhorn, and S. Schirra. How to compute the Voronoi diagram of line segments: Theoretical and experimental results. In *Proc. 2nd Annu. European Sympos. Algorithms (ESA '94)*, volume 855 of *Lecture Notes in Computer Science*, pages 227–239. Springer-Verlag, 1994.

[52] H. Busemann. *The Geometry of Geodesics.* Academic Press Inc., New York, 1955.

[53] J. Canny. A Voronoi method for the piano-movers problem. In *Proc. IEEE Internat. Conf. Robot. Autom.*, pages 530–535, St. Louis, MO, 1985.

[54] J. Canny and B. R. Donald. Simplified Voronoi diagrams. *Discrete Comput. Geom.*, 3:219–236, 1988.

[55] V. Capoyleas, G. Rote, and G. Woeginger. Geometric clusterings. *J. Algorithms*, 12:341–356, 1991.

[56] T. M. Chan, J. Snoeyink, and C.-K. Yap. Output sensitive construction of polytopes in four dimensions and clipped Voronoi diagrams in three. In *Proc. 6th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 282–291, 1995.

[57] B. Chandra, G. Das, G. Narasimhan, and J. Soares. New sparseness results on graph spanners. *Internat. J. Comput. Geom. Appl.*, 5:125–144, 1995.

[58] B. Chazelle. An optimal convex hull algorithm in any fixed dimension. *Discrete Comput. Geom.*, 10:377–409, 1993.

[59] B. Chazelle and H. Edelsbrunner. An improved algorithm for constructing kth-order Voronoi diagrams. *IEEE Trans. Comput.*, C-36:1349–1354, 1987.

[60] S. W. Cheng, M. J. Golin, and J. C. F. Tsang. Expected case analysis of β-skeletons with applications to the construction of minimum-weight triangulations. In *Proc. 7th Canad. Conf. Comput. Geom.*, pages 279–284, 1995.

[61] S.-W. Cheng and Y.-F. Xu. Approximating the largest β-skeleton within a minimum-weight-triangulation. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, to appear.

[62] L. P. Chew. Building Voronoi diagrams for convex polygons in linear expected time. Technical Report PCS-TR90-147, Dept. Math. Comput. Sci., Dartmouth College, Hanover, NH, 1986.

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

[64] L. P. Chew. There are planar graphs almost as good as the complete graph. *J. Comput. Syst. Sci.*, 39:205–219, 1989.

[65] L. P. Chew. Guaranteed-quality mesh generation for curved surfaces. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 274–280, 1993.

[66] L. P. Chew and R. L. Drysdale, III. Voronoi diagrams based on convex distance functions. In *Proc. 1st Annu. ACM Sympos. Comput. Geom.*, pages 235–244, 1985.

[67] L. P. Chew, K. Kedem, M. Sharir, B. Tagansky, and E. Welzl. Voronoi diagrams of lines in 3-space under polyhedral convex distance functions. In *Proc. 6th ACM-SIAM Sympos. Discrete Algorithms*, pages 197–204, 1995.

[68] F. Chin, J. Snoeyink, and C.-A. Wang. Finding the medial axis of a simple polygon in linear time. In *Proc. 6th Annu. Internat. Sympos. Algorithms Comput. (ISAAC 95)*, volume 1004 of *Lecture Notes in Computer Science*, pages 382–391. Springer-Verlag, 1995.

[69] N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. In J. F. Traub, editor, *Sympos. on New Directions and Recent Results in Algorithms and Complexity*, page 441, New York, NY, 1976. Academic Press.

[70] P. Cignoni, C. Montani, and R. Scopigno. A merge-first divide & conquer algorithm for E^d Delaunay triangulations. Technical report, Consiglio Nazionale delle Ricerche, Pisa, Italy, 1994.

[71] K. L. Clarkson. New applications of random sampling in computational geometry. *Discrete Comput. Geom.*, 2:195–222, 1987.

[72] K. L. Clarkson. An algorithm for geometric minimum spanning trees requiring nearly linear expected time. *Algorithmica*, 4:461–469, 1989.

[73] K. L. Clarkson, K. Mehlhorn, and R. Seidel. Four results on randomized incremental constructions. *Comput. Geom. Theory Appl.*, 3(4):185–212, 1993.

[74] K. L. Clarkson and P. W. Shor. Applications of random sampling in computational geometry, II. *Discrete Comput. Geom.*, 4:387–421, 1989.

[75] R. Cole. Reported by C. Ó'Dúnlaing, 1989.

[76] A. G. Corbalan, M. Mazon, T. Recio, and F. Santos. On the topological shape of planar Voronoi diagrams. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 109–115, 1993.

[77] H. Crapo and W. Whiteley. Plane stresses and projected polyhedra. Technical Report ??, Univ. Montreal, Montreal, PQ, 1986.

[78] G. Das and D. Joseph. Which triangulations approximate the complete graph. In *Proc. International Symposium on Optimal Algorithms*, volume 401 of *Lecture Notes in Computer Science*, pages 168–192. Springer-Verlag, 1989.

[79] E. F. D'Azevedo and R. B. Simpson. On optimal interpolation triangle incidences. *SIAM*

J. Sci. Statist. Comput., 10(6):1063–1075, 1989.
[80] M. de Berg, J. Matoušek, and O. Schwarzkopf. Piecewise linear paths among convex obstacles. In *Proc. 25th Annu. ACM Sympos. Theory Comput.*, pages 505–514, 1993.
[81] L. De Floriani, B. Falcidieno, G. Nagy, and C. Pienovi. On sorting triangles in a Delaunay tessellation. *Algorithmica*, 6:522–532, 1991.
[82] D. Defays. An efficient algorithm for a complete link method. *Comput. J.*, 20:364–366, 1977.
[83] F. Dehne and R. Klein. A sweepcircle algorithm for Voronoi diagrams. In *Proc. Internat. Workshop Graph-Theoret. Concepts Comput. Sci. (WG'87)*, pages 59–70, 1987.
[84] F. Dehne and R. Klein. "the big sweep": On the power of the wavefront approach to Voronoi diagrams. In *Proc. Math. Found. Comput. Sci.*, volume 841 of *Lecture Notes in Computer Science*, pages 296–305, 1994.
[85] F. Dehne and H. Noltemeier. A computational geometry approach to clustering problems. In *Proc. 1st Annu. ACM Sympos. Comput. Geom.*, pages 245–250, 1985.
[86] B. Delaunay. Sur la sphère vide. A la memoire de Georges Voronoi. *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskih i Estestvennyh Nauk*, 7:793–800, 1934.
[87] R. Descartes. *Principia Philosophiae.* Ludovicus Elzevirius, Amsterdam, 1644.
[88] O. Devillers. Randomization yields simple $O(n \log^* n)$ algorithms for difficult $\Omega(n)$ problems. *Internat. J. Comput. Geom. Appl.*, 2(1):97–111, 1992.
[89] A. K. Dewdney and J. K. Vranch. A convex partition of R^3 with applications to Crum's problem and Knuth's post-office problem. *Utilitas Math.*, 12:193–199, 1977.
[90] M. T. Dickerson, R. L. Drysdale, and J. R. Sack. Simple algorithms for enumerating interpoint distances and finding k nearest neighbors. *Internat. J. Comput. Geom. Appl.*, 2(3):221–239, 1992.
[91] M. T. Dickerson and D. Eppstein. Algorithms for proximity problems in higher dimensions. *Comput. Geom. Theory Appl*, 5:277–291, 1996.
[92] M. B. Dillencourt. A non-Hamiltonian, nondegenerate Delaunay triangulation. *Inform. Process. Lett.*, 25:149–151, 1987.
[93] M. B. Dillencourt. Toughness and Delaunay triangulations. *Discrete Comput. Geom.*, 5:575–601, 1990.
[94] M. B. Dillencourt. Finding Hamiltonian cycles in Delaunay triangulations is NP-complete. In *Proc. 4th Canad. Conf. Comput. Geom.*, pages 223–228, 1992.
[95] P. G. L. Dirichlet. über die reduction der positiven quadratischen formen mit drei unbestimmten ganzen zahlen. *Reine Angew. Mathematik*, 40:209–227, 1850.
[96] H. Djidjev and A. Lingas. On computing the Voronoi diagram for restricted planar figures. In *Proc. 2nd Workshop Algorithms Data Struct.*, volume 519 of *Lecture Notes in Computer Science*, pages 54–64. Springer-Verlag, 1991.
[97] D. P. Dobkin, R. L. Drysdale, III, and L. J. Guibas. Finding smallest polygons. In F. P. Preparata, editor, *Computational Geometry*, volume 1 of *Advances in Computing Research*, pages 181–214. JAI Press, London, England, 1983.
[98] D. P. Dobkin, S. J. Friedman, and K. J. Supowit. Delaunay graphs are almost as good as complete graphs. *Discrete Comput. Geom.*, 5:399–407, 1990.
[99] D. P. Dobkin and M. J. Laszlo. Primitives for the manipulation of three-dimensional subdivisions. *Algorithmica*, 4:3–32, 1989.
[100] D. P. Dobkin and R. J. Lipton. Multidimensional searching problems. *SIAM J. Comput.*, 5:181–186, 1976.
[101] J.-M. Drappier. Envellopes convexes. Technical report, Rapport Centre CPAO, ENSTA Palaiseau, 1983.
[102] R. A. Dwyer. A faster divide-and-conquer algorithm for constructing Delaunay triangulations. *Algorithmica*, 2:137–151, 1987.
[103] R. A. Dwyer. Higher-dimensional Voronoi diagrams in linear expected time. *Discrete Comput. Geom.*, 6:343–367, 1991.
[104] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*, volume 10 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, Heidelberg, West Germany, 1987.

[105] H. Edelsbrunner. An acyclicity theorem for cell complexes in d dimensions. *Combinatorica*, 10:251–260, 1990.

[106] H. Edelsbrunner. The union of balls and its dual shape. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 218–231, 1993.

[107] H. Edelsbrunner. Weighted α-shapes. *Discrete Comput. Geom.*, to appear.

[108] H. Edelsbrunner, L. Guibas, and M. Sharir. The upper envelope of piecewise linear functions: algorithms and applications. *Discrete Comput. Geom.*, 4:311–336, 1989.

[109] H. Edelsbrunner, L. J. Guibas, and J. Stolfi. Optimal point location in a monotone subdivision. *SIAM J. Comput.*, 15:317–340, 1986.

[110] H. Edelsbrunner, D. G. Kirkpatrick, and R. Seidel. On the shape of a set of points in the plane. *IEEE Trans. Inform. Theory*, IT-29:551–559, 1983.

[111] H. Edelsbrunner and E. P. Mücke. Three-dimensional alpha shapes. *ACM Trans. Graph.*, 13(1):43–72, Jan. 1994.

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

[113] H. Edelsbrunner and R. Seidel. Voronoi diagrams and arrangements. *Discrete Comput. Geom.*, 1:25–44, 1986.

[114] H. Edelsbrunner, R. Seidel, and M. Sharir. On the zone theorem for hyperplane arrangements. *SIAM J. Comput.*, 22(2):418–429, 1993.

[115] H. Edelsbrunner and N. R. Shah. Incremental topological flipping works for regular triangulations. *Algorithmica*, 15:223–241, 1996.

[116] H. Edelsbrunner and W. Shi. An $O(n \log^2 h)$ time algorithm for the three-dimensional convex hull problem. *SIAM J. Comput.*, 20:259–277, 1991.

[117] H. Edelsbrunner and T. S. Tan. A quadratic time algorithm for the minmax length triangulation. *SIAM J. Comput.*, 22:527–551, 1993.

[118] H. Edelsbrunner and T. S. Tan. An upper bound for conforming Delaunay triangulations. *Discrete Comput. Geom.*, 10(2):197–213, 1993.

[119] H. Edelsbrunner, T. S. Tan, and R. Waupotitsch. $O(N^2 \log N)$ time algorithm for the minmax angle triangulation. *SIAM J. Sci. Statist. Comput.*, 13(4):994–1008, July 1992.

[120] P. E. Ehrlich and H.-C. Im Hof. Dirichlet regions in manifolds without conjugate points. *Comment. Math. Helvetici*, 54:642–658, 1979.

[121] G. Farin. Surfaces over dirichlet tesselations. *Computer Aided Geometric Design*, 7:281–292, 1990.

[122] D. A. Field. Implementing Watson's algorithm in three dimensions. In *Proc. 2nd Annu. ACM Sympos. Comput. Geom.*, pages 246–259, 1986.

[123] S. Fortune. Numerical stability of algorithms for 2-d Delaunay triangulations and Voronoi diagrams. In *Proc. 8th Annu. ACM Sympos. Comput. Geom.*, pages 83–92, 1992.

[124] S. Fortune. Voronoi diagrams and Delaunay triangulations. In D.-Z. Du and F. K. Hwang, editors, *Computing in Euclidean Geometry*, volume 1 of *Lecture Notes Series on Computing*, pages 193–233. World Scientific, Singapore, 1992.

[125] S. J. Fortune. A sweepline algorithm for Voronoi diagrams. *Algorithmica*, 2:153–174, 1987.

[126] K. R. Gabriel and R. R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18:259–278, 1969.

[127] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, New York, NY, 1979.

[128] B. Geiger. 3D Modeling using the Delaunay triangulation. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages V11–V12, 1995.

[129] A. Gibbons. *Algorithmic Graph Theory.* Cambridge University Press, Cambridge, 1985.

[130] C. M. Gold, P. M. Remmele, and T. Roos. Voronoi diagrams of line segments made easy. In *Proc. 7th Canad. Conf. Comput. Geom.*, pages 223–228, 1995.

[131] M. Golin, R. Raman, C. Schwarz, and M. Smid. Simple randomized algorithms for closest pair problems. *Nordic J. Comput.*, 2:3–27, 1995.

[132] M. T. Goodrich, C. Ó'Dúnlaing, and C. Yap. Computing the Voronoi diagram of a set of

line segments in parallel. *Algorithmica*, 9:128–141, 1993.

[133] P. J. Green and R. R. Sibson. Computing Dirichlet tessellations in the plane. *Comput. J.*, 21:168–173, 1978.

[134] L. Guibas, J. S. B. Mitchell, and T. Roos. Voronoi diagrams of moving points in the plane. In *Proc. 17th Internat. Workshop Graph-Theoret. Concepts Comput. Sci.*, volume 570 of *Lecture Notes in Computer Science*, pages 113–125. Springer-Verlag, 1991.

[135] L. J. Guibas, D. E. Knuth, and M. Sharir. Randomized incremental construction of Delaunay and Voronoi diagrams. *Algorithmica*, 7:381–413, 1992.

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

[137] J. A. Hartigan. *Clustering Algorithms.* John Wiley, New York, 1975.

[138] D. Hartvigsen. Recognizing Voronoi diagrams with linear programming. *ORSA J. Comput.*, 4:369–374, 1992.

[139] M. Held. A geometry-based investigation of the tool path generation for zigzag pocket machining. *Visual Comput.*, 7(5–6):296–308, Sept. 1991.

[140] J. Hertz, A. Krogh, and R. G. Palmer. *Introduction to the Theory of Neural Computation.* Addison-Wesley, New York, 1991.

[141] H. Heusinger and H. Noltemeier. On separable clusterings. *J. Algorithms*, 10:212–227, 1989.

[142] K. Hinrichs, J. Nievergelt, and P. Schorn. Plane-sweep solves the closest pair problem elegantly. *Inform. Process. Lett.*, 26:255–261, 1988.

[143] N.-F. Huang. The power connecting problem on a radio network. Technical report, Dept. of Comput. Sci., National Tsing Hua Univ., Hsinchu, Taiwan 300, Republic of China, 1991.

[144] D. P. Huttenlocher, K. Kedem, and M. Sharir. The upper envelope of Voronoi surfaces and its applications. *Discrete Comput. Geom.*, 9:267–291, 1993.

[145] F. K. Hwang. An $O(n \log n)$ algorithm for rectilinear minimal spanning tree. *J. ACM*, 26:177–182, 1979.

[146] C. Icking, R. Klein, N.-M. Lê, and L. Ma. Convex distance functions in 3-space are different. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 116–123, 1993.

[147] M. Inaba, N. Katoh, and H. Imai. Applications of weighted Voronoi diagrams and randomization to variance-based k-clustering. In *Proc. 10th Annu. ACM Sympos. Comput. Geom.*, pages 332–339, 1994.

[148] H. Inagaki, K. Sugihara, and N. Sugie. Numerically robust incremental algorithm for constructing three-dimensional Voronoi diagrams. In *Proc. 4th Canad. Conf. Comput. Geom.*, pages 334–339, 1992.

[149] J. W. Jaromczyk, M. Kowaluk, and F. Yao. An optimal algorithm for constructing β-skeletons in the L_p-metrik. *SIAM J. Comput.*, to appear.

[150] B. Joe. 3-dimensional triangulations from local transformations. *SIAM J. Sci. Statist. Comput.*, 10(4):718–741, 1989.

[151] B. Joe. Construction of three-dimensional Delaunay triangulations using local transformations. *Comput. Aided Geom. Design*, 8(2):123–142, May 1991.

[152] B. Joe. Geompack. A software package for the generation of meshes using geometric algorithms. *Advances in Engineering Software and Workstations*, 13(5–6):325–331, Sept. 1991.

[153] M. Jünger, G. Reinelt, and D. Zepf. Computing correct Delaunay triangulations. *Computing*, 47(1):43–49, 1991.

[154] T. C. Kao and D. M. Mount. An algorithm for computing compacted Voronoi diagrams defined by convex distance functions. In *Proc. 3rd Canad. Conf. Comput. Geom.*, pages 104–109, 1991.

[155] T. C. Kao and D. M. Mount. Incremental construction and dynamic maintenance of constrained Delaunay triangulations. In *Proc. 4th Canad. Conf. Comput. Geom.*, pages 170–175, 1992.

[156] J. Katajainen and M. Koppinen. Constructing Delaunay triangulations by merging buckets in quadtree order. *Annales Societatis Mathematicae Polonae, Series IV, Fundamenta Informaticae*, 11(3):275–288, 1988.

[157] J. M. Keil. Computing a subgraph of the minimum weight triangulation. *Comput. Geom. Theory Appl*, 4:13–26, 1994.

[158] J. M. Keil and C. A. Gutwin. Classes of graphs which approximate the complete Euclidean graph. *Discrete Comput. Geom.*, 7:13–28, 1992.

[159] D. Kirkpatrick and J. Snoeyink. Tentative prune-and-search for computing Voronoi vertices. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 133–142, 1993.

[160] D. G. Kirkpatrick. Efficient computation of continuous skeletons. In *Proc. 20th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 18–27, 1979.

[161] D. G. Kirkpatrick. A note on Delaunay and optimal triangulations. *Inform. Process. Lett.*, 10:127–128, 1980.

[162] D. G. Kirkpatrick. Optimal search in planar subdivisions. *SIAM J. Comput.*, 12:28–35, 1983.

[163] D. G. Kirkpatrick and J. D. Radke. A framework for computational morphology. In G. T. Toussaint, editor, *Computational Geometry*, pages 217–248. North-Holland, Amsterdam, Netherlands, 1985.

[164] V. Klee. On the complexity of d-dimensional Voronoi diagrams. *Archiv der Mathematik*, 34:75–80, 1980.

[165] R. Klein. *Concrete and Abstract Voronoi Diagrams*, volume 400 of *Lecture Notes in Computer Science*. Springer-Verlag, 1989.

[166] R. Klein and A. Lingas. A linear-time randomized algorithm for the bounded Voronoi diagram of a simple polygon. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 124–132, 1993.

[167] R. Klein and A. Lingas. Fast skeleton construction. In *3rd Annual European Symposium on Algorithms (ESA '95)*, volume 979 of *Lecture Notes in Computer Science*, pages 582–596, 1995.

[168] R. Klein and A. Lingas. Manhattonian proximity in a simple polygon. *Internat. J. Comput. Geom. Appl.*, 5:53–74, 1995.

[169] R. Klein, K. Mehlhorn, and S. Meiser. Randomized incremental construction of abstract Voronoi diagrams. *Comput. Geom. Theory Appl.*, 3(3):157–184, 1993.

[170] R. Klein and D. Wood. Voronoi diagrams based on general metrics in the plane. In *Proc. 5th Sympos. Theoret. Aspects Comput. Sci.*, volume 294 of *Lecture Notes in Computer Science*, pages 281–291, 1988.

[171] J. B. Kruskal, Jr. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proc. Amer. Math. Soc.*, 7:48–50, 1956.

[172] D. Krznaric and C. Levcopoulos. The first subquadratic algorithm for complete linkage clustering. In *Proc. 6th Annu. Internat. Sympos. Algorithms Comput. (ISAAC 95)*, volume 1004 of *Lecture Notes in Computer Science*, pages 392–401. Springer-Verlag, 1995.

[173] T. Lambert. The Delaunay triangulation maximizes the mean inradius. In *Proc. 6th Canad. Conf. Comput. Geom.*, pages 201–206, 1994.

[174] T. Lambert. *Empty-Shape Triangulation Algorithms*. Ph.D. thesis, Dept. Comput. Sci., Univ. of Manitoba, Winnipeg, MB, Aug. 1994.

[175] C. L. Lawson. Software for C^1 surface interpolation. In J. R. Rice, editor, *Math. Software III*, pages 161–194, New York, NY, 1977. Academic Press.

[176] N.-M. Lê. On Voronoi diagrams in the l_p-metric in higher dimensions. In *Proc. 11th Sympos. Theoret. Aspects Comput. Sci.*, volume 775 of *Lecture Notes in Computer Science*, pages 711–722, 1994.

[177] N.-M. Lê. Randomized incremental construction of simple abstract Voronoi diagrams in 3-space. In *Proc. 10th International Conference on Fundamentals of Computation Theory (FCT'95)*, volume 965 of *Lecture Notes in Computer Science*, pages 333–342, 1995.

[178] D. T. Lee. Two-dimensional Voronoi diagrams in the L_p-metric. *J. ACM*, 27:604–618, 1980.

[179] D. T. Lee. Medial axis transformation of a planar shape. *IEEE Trans. Pattern Anal. Mach. Intell.*, PAMI-4:363–369, 1982.

[180] D. T. Lee. On k-nearest neighbor Voronoi diagrams in the plane. *IEEE Trans. Comput.*,

C-31:478–487, 1982.
[181] D. T. Lee and R. L. Drysdale, III. Generalization of Voronoi diagrams in the plane. *SIAM J. Comput.*, 10:73–87, 1981.
[182] D. T. Lee and A. K. Lin. Generalized Delaunay triangulation for planar graphs. *Discrete Comput. Geom.*, 1:201–217, 1986.
[183] D. T. Lee and C. K. Wong. Voronoi diagrams in L_1 (L_∞) metrics with 2-dimensional storage applications. *SIAM J. Comput.*, 9:200–211, 1980.
[184] C. Levcopoulos and D. Krznaric. Quasi-greedy triangulations approximating the minimum weight triangulation. In *Proc. 7th ACM-SIAM Sympos. Discrete Algorithms*, pages 392–401, 1996.
[185] A. Lingas. A linear-time construction of the relative neighborhood graph from the Delaunay triangulation. *Comput. Geom. Theory Appl.*, 4:199–208, 1994.
[186] J. Matoušek, M. Sharir, and E. Welzl. A subexponential bound for linear programming. In *Proc. 8th Annu. ACM Sympos. Comput. Geom.*, pages 1–8, 1992.
[187] D. W. Matula and R. R. Sokal. Properties of Gabriel graphs relevant to geographic variation research and clustering of points in the plane. *Geogr. Anal.*, 12:205–222, 1980.
[188] A. Maus. Delaunay triangulation and the convex hull of n points in expected linear time. *BIT*, 24:151–163, 1984.
[189] M. L. Mazón and T. Recio. Voronoi diagrams coming from discrete groups on the plane. In *Proc. 2nd Canad. Conf. Comput. Geom.*, pages 223–226, 1990.
[190] M. McAllister, D. Kirkpatrick, and J. Snoeyink. A compact piecewise-linear Voronoi diagram for convex sites in the plane. *Discrete Comput. Geom.*, 15:73–105, 1996.
[191] D. H. McLain. Two dimensional interpolation from random data. *Comput. J.*, 19:178–181, 1976.
[192] P. McMullen. The maximal number of faces of a convex polytope. *Mathematica*, 17:179–184, 1970.
[193] N. Megiddo. Linear programming in linear time when the dimension is fixed. *J. ACM*, 31:114–127, 1984.
[194] K. Mehlhorn, S. Meiser, and C. Ó'Dúnlaing. On the construction of abstract Voronoi diagrams. *Discrete Comput. Geom.*, 6:211–224, 1991.
[195] S. Meiser. *Zur Konstruktion abstrakter Voronoidiagramme.* Ph.D. thesis, Dept. Comput. Sci., Univ. Saarlandes, Saarbrücken, Germany, 1993.
[196] K. Menger. Untersuchungen über allgemeine metrik. *Math. Ann.*, 100:75–163, 1928.
[197] V. Milenkovic. Robust construction of the Voronoi diagram of a polyhedron. In *Proc. 5th Canad. Conf. Comput. Geom.*, pages 473–478, Waterloo, Canada, 1993.
[198] J. S. B. Mitchell. Shortest paths among obstacles in the plane. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 308–317, 1993.
[199] J. S. B. Mitchell, D. M. Mount, and C. H. Papadimitriou. The discrete geodesic problem. *SIAM J. Comput.*, 16:647–668, 1987.
[200] D. M. Mount and A. Saalfeld. Globally-equiangular triangulations of co-circular points in $O(n \log n)$ time. In *Proc. 4th Annu. ACM Sympos. Comput. Geom.*, pages 143–152, 1988.
[201] D. E. Muller and F. P. Preparata. Finding the intersection of two convex polyhedra. *Theoret. Comput. Sci.*, 7:217–236, 1978.
[202] K. Mulmuley. Output sensitive construction of levels and Voronoi diagrams in Re^d of order 1 to k. In *Proc. 22nd Annu. ACM Sympos. Theory Comput.*, pages 322–330, 1990.
[203] K. Mulmuley. On levels in arrangements and Voronoi diagrams. *Discrete Comput. Geom.*, 6:307–338, 1991.
[204] C. Ó'Dúnlaing, M. Sharir, and C. K. Yap. Generalized Voronoi diagrams for a ladder: I. Topological analysis. *Commun. Pure Appl. Math.*, 39:423–483, 1986.
[205] C. Ó'Dúnlaing, M. Sharir, and C. K. Yap. Generalized Voronoi diagrams for a ladder: II. Efficient construction of the diagram. *Algorithmica*, 2:27–59, 1987.
[206] C. Ó'Dúnlaing and C. K. Yap. A "retraction" method for planning the motion of a disk. *J. Algorithms*, 6:104–111, 1985.

[207] T. Ohya, M. Iri, and K. Murota. Improvements of the incremental method for the Voronoi diagram with computational comparison of various algorithms. *J. Oper. Res. Soc. Japan*, 27(4):306–336, 1984.

[208] A. Okabe, B. Boots, and K. Sugihara. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams.* John Wiley & Sons, Chichester, England, 1992.

[209] C. H. Papadimitriou. The Euclidean traveling salesman problem is NP-complete. *Theoret. Comput. Sci.*, 4:237–244, 1977.

[210] E. Papadopoulou and D. T. Lee. Efficient computation of the geodesic voronoi diagram of points in a simple polygon. In *3rd Annual European Symposium on Algorithms (ESA '95)*, volume 979 of *Lecture Notes in Computer Science*, pages 238–251, 1995.

[211] I. Paschinger. *Konvexe Polytope und Dirichletsche Zellenkomplexe.* Ph.D. thesis, Institut für Mathematik, Universität Salzburg, Salzburg, Austria, 1982.

[212] P. L. Powar. Minimal roughness property of the Delaunay triangulation: a shorter approach. *Computer Aided Geometric Design*, 9:491–494, 1992.

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

[214] R. C. Prim. Shortest connection networks and some generalizations. *Bell Syst. Tech. J.*, 36:1389–1401, 1957.

[215] V. T. Rajan. Optimality of the Delaunay triangulation in R^d. In *Proc. 7th Annu. ACM Sympos. Comput. Geom.*, pages 357–363, 1991.

[216] S. Rajasekaran and S. Ramaswami. Optimal parallel randomized algorithms for the Voronoi diagram of line segments in the plane and related problems. In *Proc. 10th Annu. ACM Sympos. Comput. Geom.*, pages 57–66, 1994.

[217] R. Rasch. *Abstrakte inverse Voronoidiagramme.* Ph.D. thesis, Dept. Comput. Sci., Univ. Saarlandes, Saarbrücken, Germany, 1994.

[218] S. Rippa. Minimal roughness property of the Delaunay triangulation. *Comput. Aided Geom. Design*, 7:489–497, 1990.

[219] T. Roos. Tighter bounds on Voronoi diagrams of moving points. In *Proc. 5th Canad. Conf. Comput. Geom.*, pages 358–363, Waterloo, Canada, 1993.

[220] D. J. Rosenkrantz, R. E. Stearns, and P. M. Lewis. An analysis of several heuristics for the traveling salesman problem. *SIAM J. Comput.*, 6:563–581, 1977.

[221] M. Sakamoto and M. Takagi. Patterns of weighted Voronoi tesselations. *Science and Form*, 3:103–111, 1988.

[222] R. Seidel. The complexity of Voronoi diagrams in higher dimensions. In *Proc. 20th Allerton Conf. Commun. Control Comput.*, pages 94–95, 1982.

[223] R. Seidel. A method for proving lower bounds for certain geometric problems. In G. T. Toussaint, editor, *Computational Geometry*, pages 319–334. North-Holland, Amsterdam, Netherlands, 1985.

[224] R. Seidel. Constructing higher-dimensional convex hulls at logarithmic cost per face. In *Proc. 18th Annu. ACM Sympos. Theory Comput.*, pages 404–413, 1986.

[225] R. Seidel. On the number of faces in higher-dimensional Voronoi diagrams. In *Proc. 3rd Annu. ACM Sympos. Comput. Geom.*, pages 181–185, 1987.

[226] R. Seidel. Constrained Delaunay triangulations and Voronoi diagrams. In *Report 260*, pages 178–191. IIG-TU, Graz, Austria, 1988.

[227] R. Seidel. Small-dimensional linear programming and convex hulls made easy. *Discrete Comput. Geom.*, 6:423–434, 1991.

[228] R. Seidel. Backwards analysis of randomized geometric algorithms. In J. Pach, editor, *New Trends in Discrete and Computational Geometry*, volume 10 of *Algorithms and Combinatorics*, pages 37–68. Springer-Verlag, 1993.

[229] M. I. Shamos. *Computational Geometry.* Ph.D. thesis, Dept. Comput. Sci., Yale Univ., New Haven, CT, 1978.

[230] M. I. Shamos and D. Hoey. Closest-point problems. In *Proc. 16th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 151–162, 1975.

[231] M. Sharir. Intersection and closest-pair problems for a set of planar discs. *SIAM J. Comput.*, 14:448–468, 1985.

[232] M. Sharir. Almost tight upper bounds for lower envelopes in higher dimensions. *Discrete Comput. Geom.*, 12:327–345, 1994.

[233] R. Sibson. Locally equiangular triangulations. *Comput. J.*, 21:243–245, 1978.

[234] R. Sibson. A vector identity for the Dirichlet tesselation. *Math. Proc. Camb. Phil. Soc.*, 87:151–155, 1980.

[235] S. Skyum. A sweepline algorithm for generalized Delaunay triangulations. Technical Report DAIMI PB-373, CS Dept., Aarhus University, 1991.

[236] S. W. Sloan. A fast algorithm for constructing Delaunay triangulations in the plane. *Adv. Eng. Softw.*, 9(1):34–55, Jan. 1987.

[237] M. Smid. Maintaining the minimal distance of a point set in less than linear time. *Algorithms Rev.*, 2:33–44, 1991.

[238] P. Su and R. L. S. Drysdale. A comparison of sequential Delaunay triangulation algorithms. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 61–70, 1995.

[239] K. Sugihara. A simple method for avoiding numerical errors and degeneracy in Voronoi diagram construction. *IEICE Trans. Fundamentals*, E75-A(4):468–477, Apr. 1992.

[240] K. Sugihara and M. Iri. Construction of the Voronoi diagram for 'one million' generators in single-precision arithmetic. *Proc. IEEE*, 80(9):1471–1484, Sept. 1992.

[241] K. Sugihara, Y. Ooishi, and T. Imai. Topology-oriented approach to robustness and its applications to several Voronoi-diagram algorithms. In *Proc. 2nd Canad. Conf. Comput. Geom.*, pages 36–39, 1990.

[242] K. J. Supowit. The relative neighborhood graph with an application to minimum spanning trees. *J. ACM*, 30:428–448, 1983.

[243] A. Suzuki and M. Iri. Approximation of a tessellation of the plane by a Voronoi diagram. *J. Oper. Res. Soc. Japan*, 29:69–96, 1986.

[244] T.-S. Tan. *Optimal two-dimensional triangulations.* Ph.D. thesis, Univ. of Illinois at Urbana-Champaign, 1993.

[245] M. Tanemura, T. Ogawa, and W. Ogita. A new algorithm for three-dimensional Voronoi tesselation. *J. Comput. Phys.*, 51:191–207, 1983.

[246] W. P. Thurston. The geometry of cirles: Voronoi diagrams, moebius transformations, convex hulls, fortune's algorithm, the cut locus, and parametrization of shapes. Technical report, Princeton University, 1986.

[247] G. T. Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern Recogn.*, 12:261–268, 1980.

[248] P. M. Vaidya. Minimum spanning trees in k-dimensional space. *SIAM J. Comput.*, 17:572–582, 1988.

[249] P. M. Vaidya. A sparse graph almost as good as the complete graph on points in K dimensions. *Discrete Comput. Geom.*, 6:369–381, 1991.

[250] G. F. Voronoï. Deuxième mémoire: recherches sur les paralléloedres primitifs. *J. Reine Angew. Math.*, 136:67–181, 1909.

[251] G. M. Voronoi. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. deuxième Mémoire: Recherches sur les parallélloèdres primitifs. *J. Reine Angew. Math.*, 134:198–287, 1908.

[252] C. A. Wang. Efficiently updating the constrained Delaunay triangulations. *BIT*, 33:238–252, 1993.

[253] C. A. Wang and L. Schubert. An optimal algorithm for constructing the Delaunay triangulation of a set of line segments. In *Proc. 3rd Annu. ACM Sympos. Comput. Geom.*, pages 223–232, 1987.

[254] D. F. Watson. Computing the n-dimensional Delaunay tesselation with applications to Voronoi polytopes. *Comput. J.*, 24(2):167–172, 1981.

[255] E. Welzl. Smallest enclosing disks (balls and ellipsoids). In H. Maurer, editor, *New Results and New Trends in Computer Science*, volume 555 of *Lecture Notes in Computer Science*,

pages 359–370. Springer-Verlag, 1991.

[256] P. Widmayer, Y. F. Wu, and C. K. Wong. On some distance problems in fixed orientations. *SIAM J. Comput.*, 16:728–746, 1987.

[257] A. C. Yao. On constructing minimum spanning trees in k-dimensional spaces and related problems. *SIAM J. Comput.*, 11:721–736, 1982.

[258] C.-K. Yap. Algorithmic motion planning. In J. T. Schwartz and C.-K. Yap, editors, *Advances in Robotics 1: Algorithmic and Geometric Aspects of Robotics*, pages 95–143. Lawrence Erlbaum Associates, Hillsdale, NJ, 1987.

[259] C. K. Yap. An $O(n \log n)$ algorithm for the Voronoi diagram of a set of simple curve segments. *Discrete Comput. Geom.*, 2:365–393, 1987.

[260] B. Zhu and A. Mirzaian. Sorting does not always help in computational geometry. In *Proc. 3rd Canad. Conf. Comput. Geom.*, pages 239–242, 1991.