Farthest-Polygon Voronoi Diagrams*

Otfried Cheong[1] Hazel Everett[2] Marc Glisse[2] Joachim Gudmundsson[3]
Samuel Hornus[1] Sylvain Lazard[2] Mira Lee[1] Hyeon-Suk Na[4]

April 13, 2007

Abstract

Given a family of k disjoint connected polygonal sites of total complexity n, we consider the farthest-site Voronoi diagram of these sites, where the distance to a site is the distance to a closest point on it. We show that the complexity of this diagram is $O(n)$, and give an $O(n \log^3 n)$ time algorithm to compute it. We also prove a number of structural properties of this diagram. In particular, a Voronoi region may consist of $k-1$ connected components, but if one component is bounded, then it is equal to the entire region.

1 Introduction

Consider a family $\mathcal{S}$ of geometric objects (called "sites") in the plane. The farthest-site Voronoi diagram of $\mathcal{S}$ subdivides the plane into regions, each region associated with one site $P \in \mathcal{S}$, and containing those points $x \in \mathbb{R}^2$ for which P is the farthest among the sites of $\mathcal{S}$.

While closest-site Voronoi diagrams have been studied extensively [3], their farthest-site cousins have received somewhat less attention. The case of (possibly intersecting) line segment sites was only solved recently by Aurenhammer et al. [4]; they gave an $O(n \log n)$ time algorithm to compute the diagram for n line segments.

Farthest-site Voronoi diagrams have a number of important applications. Perhaps the most well-known one is the problem of finding a smallest disk that intersects all the sites. This disk can be computed in linear time once the diagram is known, since its center is a vertex or lies on an edge of the diagram. Other applications are finding the largest gap to be bridged between sites, or building a data structure to quickly report the site farthest from a given query point.

We are here interested in the case of complex sites with non-constant description complexity. This setting was perhaps first considered by Abellanas et al. [1]: their sites are finite point sets, and so the distance to a site is the distance to the nearest point of that site. Put differently, they consider n points colored with k different colors, and their *farthest color Voronoi diagram* subdivides the plane depending on which color is farthest away. The motivation for this problem is the one mentioned above, namely to find a smallest disk that contains a point of each color—this is a facility location problem where the goal is to find a position that is as close as possible to each of k different types of facilities (such as schools, post offices, supermarkets, etc.). In a companion paper [2] the authors study other color-spanning objects.

The farthest color Voronoi diagram is easily seen to be the projection of the upper envelope of the k Voronoi surfaces corresponding to the k color classes. Huttenlocher et al. [8] show that this upper envelope has complexity $\Theta(nk)$ for n points, and can be computed in time $O(nk \log n)$ (see also the book by Sharir and Agarwal [14], Section 8.7).

Van Kreveld and Schlechter [10] consider the farthest-site Voronoi diagram for a family of disjoint simple polygons. Again, they are interested in finding the center of the smallest disk intersecting or

*This research was supported by the French-Korean Science and Technology Amicable Relationships program (STAR), and the Brain Korea 21 Project, the School of Information Technology, KAIST, 2007.

[1]Dept. of Computer Science, KAIST, Daejeon, Korea. {otfried,hornus,mira}@tclab.kaist.ac.kr.

[2]LORIA – INRIA Lorraine, Université Nancy 2, Nancy, France. Firstname.Name@loria.fr.

[3]National ICT Australia Ltd., Sydney, Australia. joachim.gudmundsson@nicta.com.au. National ICT Australia is funded through the Australian Government's Backing Australia's Ability initiative, in part through the Australian Research Council.

[4]School of Computing, Soongsil University, Seoul, Korea. hsnaa@ssu.ac.kr.

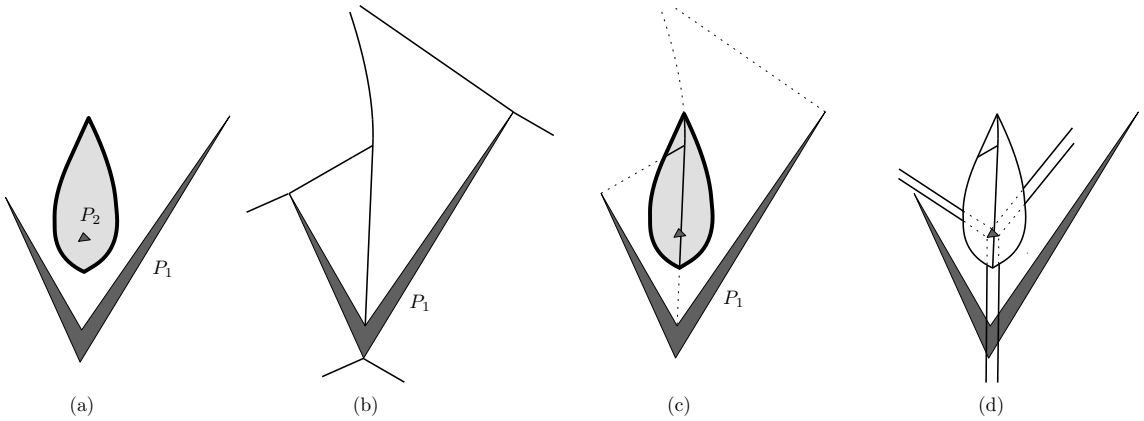

Figure 1: (a) The bisector of two polygons can be a closed curve. (b) The medial axis $\mathfrak{M}(P_1)$. (c) The Voronoi region $\mathcal{R}(P_1)$. (d) The farthest polygon Voronoi diagram $\mathfrak{F}(\{P_1, P_2\})$.

touching all polygons, which they then apply to the cartographic problem of labeling groups of islands. Their algorithm is based on the claim that this *farthest-polygon Voronoi diagram* is an instance of the *abstract farthest-site Voronoi diagram* defined by Mehlhorn et al. [12]—but this claim is false, since the bisector of two disjoint simple polygons can be a closed curve, see Fig. 1(a). In particular, Voronoi regions can be bounded (which is impossible for regions in abstract farthest-site Voronoi diagrams).

Note that the farthest-polygon Voronoi diagram can again be expressed as the upper envelope of k Voronoi surfaces—but this does not seem to lead to anything stronger than near-quadratic complexity and time bounds.

We show in this paper that in fact the complexity of the farthest-polygon Voronoi diagram of k disjoint simple polygons of total complexity n is $O(n)$, independent of the number of polygons. (And in fact we discuss *polygonal sites*, which are slightly more general than simple polygons, see Section 2.)

When computing closest-site Voronoi diagrams, one makes use of the fact that Voronoi regions surround and are close to their sites. Similarly, in the computation of farthest-site Voronoi diagrams, one makes use of the unboundedness of the Voronoi regions, and often builds up regions from infinity [4]. The difficulty in computing farthest-polygon Voronoi diagrams is that neither of these properties holds: Voronoi regions can be bounded, and finding the location of these bounded regions is the bottleneck in the computation.

We give a divide-and-conquer algorithm with running time $O(n \log^3 n)$ to compute the farthest-polygon Voronoi diagram. Our key idea is to build point location data structures for the partial diagrams already computed, and to use parametric search on these data structures to find suitable starting vertices for the merging step. This idea may find applications to the computation of other complicated Voronoi diagrams. Our algorithm implies an $O(n \log^3 n)$ algorithm to compute the smallest disk touching or intersecting all the input polygons.

We note that for a family of disjoint *convex* polygons, finding the smallest disk touching all of them is much easier, and can be solved in time $O(n)$ (where n is the total complexity of the polygons) [9].

Further than the combinatorial bound and the algorithm for constructing the farthest-polygon Voronoi diagram, we also give prove interesting properties of the latter: Voronoi regions can be disconnected, and in fact, the region of a polygon can consist of up to $k-1$ connected components, but then all components must be unbounded. Also, if the non-empty Voronoi region of polygon P is bounded, then the convex hull of P contains another polygon in its interior and the Voronoi region of P is simply connected.

Farthest-polygon Voronoi diagrams are beautiful, so we invite the reader to examine a few specimens at http://tclab.kaist.ac.kr/∼hornus/fpvd/.

2 Definition of farthest polygon Voronoi diagrams

We consider a family $\mathcal{S}$ of k pairwise-disjoint polygonal sites of total complexity n. Here, a *polygonal site* of complexity m is the union of m line segments, whose interiors are pairwise disjoint, but whose union is

connected. (In other words, the corners[1] and edges of a polygonal site form a one-dimensional connected simplicial complex in the plane.) In particular, the boundary of a simple polygon is a polygonal site and we can model a polygon with h holes using its $h+1$ boundaries and connecting them with at most h additional edges. For a point $x \in \mathbb{R}^2$, the distance $d(x, P)$ between x and a site $P \in \mathcal{S}$ is the distance from x to the closest point on P.

We assume that the family $\mathcal{S}$ is in general position, that is, no disk touches four edges, and no two edges are parallel.

The *features* of a site P are its corners and edges. For a site $P \in \mathcal{S}$, we define the function $\Psi_P : \mathbb{R}^2 \mapsto \mathbb{R}$ as $\Psi_P(x) = d(x, P)$. The graph of Ψ_P is a *Voronoi surface*, it is the lower envelope of circular cones for each corner of P and of rectangular wedges for each edge of P. The orthogonal projection of this surface on the plane induces a subdivision of the plane, the *medial axis* $\mathfrak{M}(P)$ of P. Each cell C of this subdivision corresponds to a feature w of P, it is the set of all points $x \in \mathbb{R}^2$ such that w is or contains the unique closest point on P to x. Here, edges of P are considered relatively open, so the cell of a corner is disjoint from the cells of its incident edges. The boundaries between cells are the *arcs* and *vertices* of the medial axis. Since the medial axis of P is the closest-site Voronoi diagram of P's features, it can be computed in time $O(m \log m)$, where m is the complexity of P [7].

A pocket of P is a connected component of $\mathrm{CH}(P) \setminus P$, where $\mathrm{CH}(P)$ is the convex hull of P. The medial axis $\mathfrak{M}(P)$ contains exactly one tree for each pocket of P. If a pocket shares an edge with $\mathrm{CH}(P)$ that is not an edge of P, then its medial axis tree contains exactly one infinite arc. (There can be additional infinite arcs for each corner of an edge of P that appear on the convex hull.)

We now consider the function $\Phi : \mathbb{R}^2 \mapsto \mathbb{R}$ defined as $\Phi(x) = \max_{P \in \mathcal{S}} \Psi_P(x)$. The graph of Φ is the upper envelope of the surfaces Ψ_P, for $P \in \mathcal{S}$. The surface Φ consists of conical and planar patches from the Voronoi surfaces Ψ_P, and the arcs separating such patches are either arcs of a Voronoi surface Ψ_P (we call these *medial axis arcs*), or intersection curves of two Voronoi surfaces Ψ_P and Ψ_Q (we call these *pure arcs*). The vertices of Φ are of one of the following three types:

- Vertices of one Voronoi surface Ψ_P. We call these *medial axis vertices*.
- Intersections of an arc of Ψ_P with a patch of another surface Ψ_Q. We call these *mixed vertices*.
- Intersections of patches of three Voronoi surfaces Ψ_P, Ψ_Q, Ψ_R. We call these *pure vertices*.

The projection of the surface Φ onto the plane is the *farthest-polygon Voroni diagram* $\mathfrak{F}(\mathcal{S})$ of $\mathcal{S}$. It is a subdivision of the plane into cells, arcs, and vertices. For a point $x \in \mathbb{R}^2$, let us define $D(x) = D_{\mathcal{S}}(x)$ as the smallest disk centered at x that intersects all sites $P \in \mathcal{S}$. By definition, there is always at least one site that touches $D(x)$ without intersecting its interior, and the radius of $D(x)$ is equal to $\Phi(x)$. By our general position assumption, only the following five cases can occur:

- If $D(x)$ touches one site P in only one feature w, and all other sites intersect the interior of $D(x)$, then x lies in a cell of $\mathfrak{F}(\mathcal{S})$, and the cell belongs to the feature w of P.
- If $D(x)$ touches one site P in two or three features, and all other sites intersect the interior of $D(x)$, then x lies on a medial axis arc or medial axis vertex of $\mathfrak{F}(\mathcal{S})$, and is incident to cells belonging to different features of P.
- If $D(x)$ touches one feature w of site P, one feature u of site Q, and all other sites intersect the interior of $D(x)$, then x lies on a pure arc separating cells belonging to features w and u.
- If $D(x)$ touches two features of site P and one feature of site Q, and all other sites intersect the interior of $D(x)$, then x is a mixed vertex incident to a medial axis arc of P.
- If $D(x)$ touches one feature each of three sites P, Q, and R, and all other sites intersect the interior of $D(x)$, then x is a pure vertex.

Put differently, vertices of $\mathfrak{F}(\mathcal{S})$ are points $x \in \mathbb{R}^2$ where $D(x)$ touches three distinct features of sites. If all three features are on the same site, the vertex is a medial axis vertex. If the three features are on three distinct sites, then the vertex is a pure vertex. In the remaining case, if two features are on a site P, and the third feature is on a different site Q, the vertex is a mixed vertex.

Consider now an arc α of $\mathfrak{F}(\mathcal{S})$. If we let x move along α, then $\Phi(x)$—which is the radius of $D(x)$—changes continuously. Since the arc is defined by two features (corners or edges), $\Phi(x)$ cannot assume a local maximum in the interior of α, but it can assume a local minimum. We denote the location of such a local minimum a *pseudo-vertex* of $\mathfrak{F}(\mathcal{S})$. After introducing pseudo-vertices, $\Phi(x)$ is a monotone function on each arc, and we can thus orient all arcs in the direction of increasing $\Phi(x)$.

[1] We reserve the word "vertex" for vertices of the Voronoi diagram.

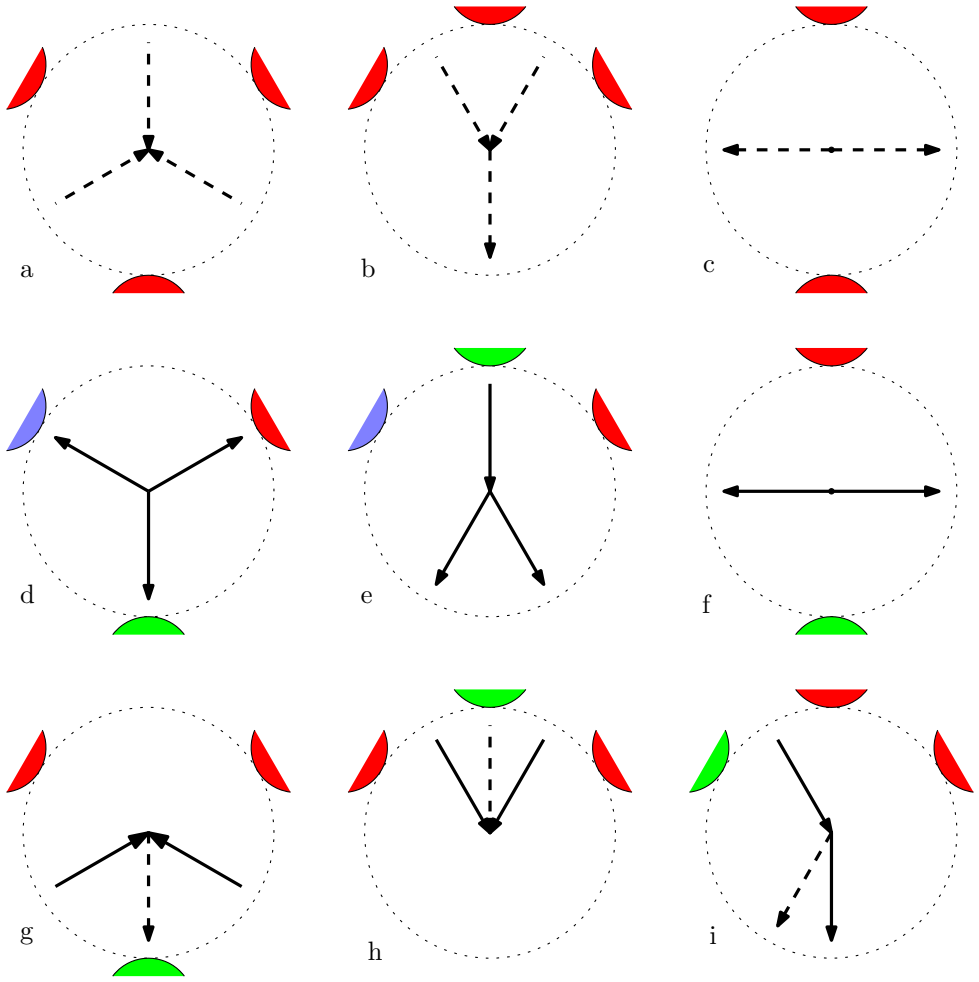

Figure 2: The different types of vertices in the farthest polygon Voronoi diagram. Pure arcs are shown solid, medial axis arcs dashed. The arrows indicate the direction of increasing $\Phi(x)$.

Fig. 2 illustrates all different vertex types, including the two types of pseudo-vertices of degree two (types (c) and (f)). The top row shows the possible medial axis vertices: types (a) and (b) differ in whether the triangle formed by the three features contains the vertex or not. Similarly, the middle row shows the pure vertex types, with the same distinction between (d) and (e). The bottom row shows the three possible types of mixed vertices. Again, we have to distinguish whether the three features enclose the vertex or not, and in the latter case we need to distinguish which two features are on the same site.

Since the local shape of $\mathfrak{F}(\mathcal{S})$ around a vertex v is determined solely by the features defining the vertex, in each configuration we can uniquely determine the orientation of the incident arcs. Fig. 2 shows the orientation of these arcs in each case. This orientation plays a crucial role in the next section, where we show that the complexity of the diagram, that is, the total number of vertices, is only $O(n)$.

In addition to the nine types of vertices discussed above, we need to consider *vertices at infinity*, that is, we consider the semi-infinite arcs of $\mathfrak{F}(\mathcal{S})$ to have a degree-one vertex at their end. For a vertex v at infinity, the "disk" $D(v)$ is a halfplane, and we have two cases:

- If $D(v)$ touches one site P in two features, and all other sites intersect the interior of $D(v)$, then $D(v)$ is the "infinite" endpoint of a medial axis arc, and we consider it a medial axis vertex at infinity.
- If $D(v)$ touches two distinct sites, and all other sites intersect its interior, then $D(v)$ is the "infinite" endpoint of a pure edge, and we consider it a pure vertex at infinity.

Finally, let us define the *Voronoi region* of a site $P \in \mathcal{S}$. The Voronoi region $\mathcal{R}(P)$ of P is simply the union of all cells, medial axis arcs, and medial axis vertices of $\mathfrak{F}(\mathcal{S})$ belonging to features of P. Voronoi

regions are not necessarily connected, in fact, a single Voronoi region can have up to $k-1$ connected components—we give an example in Section 5.

3 Complexity bound

Consider again Fig. 2. Let us call a vertex a *source* if it has more outgoing pure arcs than incoming pure arcs, and a *sink* if it has more incoming pure arcs than outgoing pure arcs. As shown in the figure, all finite pure vertices are sources. The pure vertices at infinity are sinks. The only other sinks are the mixed vertices of type (g) and (h). Note that mixed vertices of type (i) are neither sources nor sinks.

We can now partition the pure arcs of $\mathfrak{F}(\mathcal{S})$ into disjoint directed paths. Each such path starts at a source, and ends at a sink. This implies that we can bound the number of sources by twice the number of sinks, as at most two paths can end in a sink.

We show in the following that the number of pure vertices at infinity is at most $2k-2$, and that the number of mixed vertices is $O(n)$. Since the total number of vertices of all medial axes $\mathfrak{M}(P)$, for $P \in \mathcal{S}$, is $O(n)$, this implies the main theorem of this section:

Theorem 1 *The complexity of the farthest-polygon Voronoi diagram of a family of disjoint polygonal sites of total complexity n is $O(n)$.*

We first discuss the number of vertices at infinity.

Lemma 2 *The number of pure vertices at infinity of $\mathfrak{F}(\mathcal{S})$ is at most $2k-2$. The total number of vertices at infinity of $\mathfrak{F}(\mathcal{S})$ is $O(n)$.*

Proof. For two sites $P, Q \in \mathcal{S}$, consider the diagram $\mathfrak{F}(\{P,Q\})$. A pure vertex at infinity corresponds to an edge of $\mathrm{CH}(P \cup Q)$ supported by a corner of P and a corner of Q. But $\mathrm{CH}(P \cup Q)$ can have at most two such edges, since P and Q are disjoint and both are connected, and so $\mathfrak{F}(\{P,Q\})$ has at most two pure vertices at infinity.

Consider now again $\mathfrak{F}(\mathcal{S})$, and let $\sigma(\mathcal{S})$ denote the sequence of sites whose Voronoi regions appear at infinity in circular order, starting and ending at the same region. We claim that $\sigma(\mathcal{S})$ is a Davenport-Schinzel sequence of order 2, and has therefore length at most $2k-1$ [14]. Indeed, $\sigma(\mathcal{S})$ has by definition no two consecutive identical symbols. Assume now that there are two sites P and Q such that the subsequence $PQPQ$ appears in $\sigma(\mathcal{S})$. If we delete all other sites, then
$sigma(\{P,Q\})$ would still need to contain the subsequence $PQPQ$, and therefore $\mathfrak{F}(\{P,Q\})$ would contain at least three pure vertices at infinity, a contradiction to the observation above.

It now suffices to observe that the pure vertices at infinity are exactly the transitions between consecutive Voronoi regions, and their number is at most $2k-2$.

All remaining vertices at infinity are medial axis vertices. Since the total complexity of all $\mathfrak{M}(P)$, for $P \in \mathcal{S}$, is $O(n)$, the bound follows. □

It remains to show that the total number of mixed vertices is $O(n)$. Consider a site $P \in \mathcal{S}$ of complexity m. Its medial axis $\mathfrak{M}(P)$ has complexity $O(m)$, and we can consider $\mathfrak{M}(P)$ as a graph embedded in $\mathbb{R}^2 \setminus P$. It consists of a collection of trees. In Lemma 4 below we show that for each tree $\mathcal{T}$ of $\mathfrak{M}(P)$ the intersection $\mathcal{T} \cap \mathcal{R}(P)$ is a connected subtree. Since the mixed vertices on $\mathcal{T}$ are exactly the finite leaves of this subtree, this implies that the number of mixed vertices on $\mathfrak{M}(P)$ is $O(m)$. Summing over all $P \in \mathcal{S}$ then proves that the number of mixed vertices of $\mathfrak{F}(\mathcal{S})$ is $O(n)$.

It remains to prove Lemma 4. We need another notation: for a point $x \in \mathbb{R}^2$ and a site P, let $D_P(x)$ denote the largest disk centered at x whose interior does not intersect P (and which is therefore touching P). Note that a point $x \in \mathfrak{M}(P)$ is in $\mathcal{R}(P)$ if and only if $D_{\mathcal{S}}(x) = D_P(x)$, which is true if and only if all other sites intersect the interior of $D_P(x)$.

Lemma 3 *Let γ be a path in $\mathfrak{M}(P)$, let $Q \in \mathcal{S} \setminus \{P\}$ be another site, and let γ_Q be the set of points $x \in \gamma$ where $D_P(x)$ intersects Q. Then γ_Q is a connected subset of γ, that is, a subpath.*

Proof. We can assume γ to be a maximal path in $\mathcal{T}$, connecting a corner w of P with another corner u (or possibly going to infinity). Assume for a contradiction that there are points x, y, z on γ in this order such that $x, z \in \gamma_Q$, but $y \notin \gamma_Q$.

The disk $D_P(y)$ separates one connected component of $\mathbb{R}^2 \setminus P$ into two components A and B. The two endpoints of γ must lie in different components, say $w \in A$ and $u \in B$.

We first argue that any disk D that does not intersect P cannot contain points in both A and B. Indeed, the boundary of D would have to intersect the boundary of $D_P(y)$ in four points, a contradiction.

The disk $D_P(z)$ touches P on the boundary of B, and so it contains a point in B—which implies that it cannot contain a point in A. Since $D_P(y)$ does not intersect Q and Q is connected, this implies that Q is entirely contained in B. On the other hand, the disk $D_P(x)$ touches P on the boundary of A, and by our claim that means that it cannot intersect B. That implies that $D_P(x) \cap Q$ is empty, and the claim follows. □

Lemma 4 *Let $\mathcal{T}$ be a tree of $\mathfrak{M}(P)$. Then $\mathcal{T} \cap \mathcal{R}(P)$ is a connected subtree of $\mathcal{T}$.*

Proof. Pick two points $p, q \in \mathcal{T} \cap \mathcal{R}(P)$, let γ be the path on $\mathcal{T}$ from p to q, and let x be any point between p and q on γ. We need to show that $x \in \mathcal{R}(P)$, which is equivalent to showing that every site $Q \neq P$ intersects the interior of $D_P(x)$. Let Q be such a site. Since $p, q \in \mathcal{R}(P)$, Q intersects the interior of $D_P(p)$ and $D_P(q)$. By Lemma 3 this implies that Q intersects the interior of $D_P(x)$. □

4 Computing the Voronoi diagram

The proof of Theorem 1 suggests an algorithm for computing the diagram by tracing the paths considered there. This is roughly equivalent to computing the surface Φ by sweeping a horizontal plane downwards, and maintaining the part of Φ above this plane. This is essentially the approach used by Aurenhammer et al. [4] for the computation of farthest-segment Voronoi diagrams. It does not seem to work for our diagram because of the mixed vertices of type (h), where Φ has a local maximum.

We instead offer a divide-and-conquer algorithm.

Theorem 5 *The farthest polygon Voronoi diagram $\mathfrak{F}(\mathcal{S})$ of a family $\mathcal{S}$ of disjoint polygonal sites of total complexity n can be computed in expected time $O(n \log^3 n)$.*

Proof. Let $\mathcal{S} = \{P_1, \ldots, P_k\}$, and let n_i be the complexity of P_i. If $k = 1$, then $\mathfrak{F}(\mathcal{S})$ is simply the medial axis $\mathfrak{M}(P_1)$, which can be computed in time $O(n \log n)$ [7]. Otherwise, we split $\mathcal{S}$ into two disjoint families $\mathcal{S}_1, \mathcal{S}_2$ as follows:

- If there is a site P_i with complexity $n_i \geqslant n/2$, then $\mathcal{S}_1 = \{P_i\}$ and $\mathcal{S}_2 = \mathcal{S} \setminus \{P_i\}$.
- Otherwise there must be an index j such that $n/4 \leqslant \sum_{i=1}^{j} n_i \leqslant 3n/4$. We let $\mathcal{S}_1 = \{P_1, \ldots, P_j\}$ and $\mathcal{S}_2 = \{P_{j+1}, \ldots, P_k\}$.

We recursively compute $\mathfrak{F}(\mathcal{S}_1)$ and $\mathfrak{F}(\mathcal{S}_2)$. We show below that we can then merge these two diagrams to obtain $\mathfrak{F}(\mathcal{S})$ in time $O(n \log^2 n)$, proving the theorem. □

It remains to discuss the merging step. We are given a family $\mathcal{S}$ of disjoint polygonal sites of total complexity n, and we are given $\mathfrak{F}(\mathcal{S}_1)$ and $\mathfrak{F}(\mathcal{S}_2)$, where $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2$ is a disjoint partition of $\mathcal{S}$.

Consider the diagram $\mathfrak{F}(\mathcal{S})$ to be computed. We color the Voronoi regions of $\mathfrak{F}(\mathcal{S})$ defined by sites in $\mathcal{S}_1$ red, and the Voronoi regions defined by sites in $\mathcal{S}_2$ blue. A pure arc of $\mathfrak{F}(\mathcal{S})$ is red if it separates two red regions, and blue if it separates two blue regions. The remaining pure arcs, which separate a red and a blue region, are called *purple*. A vertex of $\mathfrak{F}(\mathcal{S})$ is purple if it is incident to a purple arc. We observe that by our general position assumption, every finite purple vertex is incident to exactly two purple arcs, and so the purple arcs form a collection of open and closed chains, see Fig. 2.

Merging $\mathfrak{F}(\mathcal{S}_1)$ and $\mathfrak{F}(\mathcal{S}_2)$ can be done in linear time once all purple arcs are known. In fact, the diagram $\mathfrak{F}(\mathcal{S})$ consists of those portions of $\mathfrak{F}(\mathcal{S}_1)$ lying in the red regions of $\mathfrak{F}(\mathcal{S})$, and those portions of $\mathfrak{F}(\mathcal{S}_2)$ lying in the blue regions of $\mathfrak{F}(\mathcal{S})$, see Fig. 3.

We show below that if we have a starting vertex on every purple chain, then we can trace the purple chains in total time $O(n)$. For the open chains, we can use the purple vertices at infinity as starting vertices, as these are easy to compute. For the closed purple chains, we make use of the following lemma:

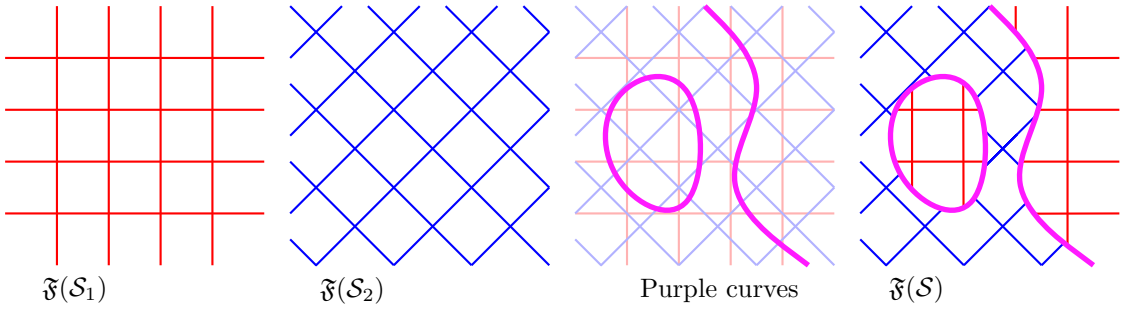

Figure 3: Merging $\mathfrak{F}(\mathcal{S}_1)$ and $\mathfrak{F}(\mathcal{S}_2)$ to obtain $\mathfrak{F}(\mathcal{S})$.

Lemma 6 *Any closed purple chain contains a mixed vertex.*

Proof. A closed purple chain is a compact set in the plane, and so $\Phi(x)$ assumes a maximum in some point x on this curve. But Φ can assume a local maximum only in mixed vertices of type (g) and (h), see Fig. 2. □

It remains to discuss the following three steps:

- Computing the pure vertices at infinity.
- Computing the mixed vertices.
- Tracing the purple chains.

4.1 Tracing the purple chains

We first need to discuss a monotonicity property of cells of $\mathfrak{F}(\mathcal{S})$. Let C be a cell of $\mathfrak{F}(\mathcal{S})$ belonging to feature w of site P. For a point $x \in C$, let x^* be the point on w closest to x. Let f_x be a directed line segment starting at x and extending in direction $\overrightarrow{x^*x}$ until we reach $\mathfrak{M}(P)$ (a semi-infinite segment if this does not happen). We call f_x the *fiber* of x. We note that if w is an edge, then all fibers of C are parallel and normal to w; if w is a corner then all fibers are supported by lines through w.

Lemma 7 *For any $x \in C$, the fiber f_x lies entirely in C (and therefore in $\mathcal{R}(P)$).*

Proof. The disk $D(x)$ touches P in x^* only, and its interior intersects all other sites. When we move a point y from x along f_x, the disk D centered at y through x^* keeps containing $D(x)$, and it therefore still intersects all other sites. This implies that $y \in C$ as long as D does not intersect P in another point. This does not happen until we reach $\mathfrak{M}(P)$. □

An immediate consequence is that cells are "monotone":

Lemma 8 *Let C be a cell belonging to feature w. If w is a corner, then any line through w intersects C in a segment. If w is an edge, then any line normal to w intersects C in a segment.*

Proof. Consider such a line ℓ, and let x be the point closest to w in $\ell \cap C$. Then the entire fiber f_x lies in C, and no point on ℓ beyond the medial axis can be in C. □

Lemma 8 implies that the boundary of a cell C belonging to a feature w consists of two chains monotone with respect to w (that is, monotone in the direction of an edge, and rotationally monotone around a corner). The *lower chain* is closer to the feature and consists of pure arcs only, the *upper chain* consists of medial axis arcs only.

An important consequence of Lemma 7 for tracing the purple chains is the following:

Lemma 9 *Let f_x be a fiber of a cell C in $\mathfrak{F}(\mathcal{S}_1)$ or $\mathfrak{F}(\mathcal{S}_2)$. Then f_x is intersected by at most one purple arc of $\mathfrak{F}(\mathcal{S})$.*

Proof. Assume for the contrary that two purple arcs intersect f_x in points p and q, where q lies on f_p. Then there is a point p' on f_x very close to p such that $p' \in \mathcal{R}(P)$ in $\mathfrak{F}(\mathcal{S})$, where C belongs to site P.

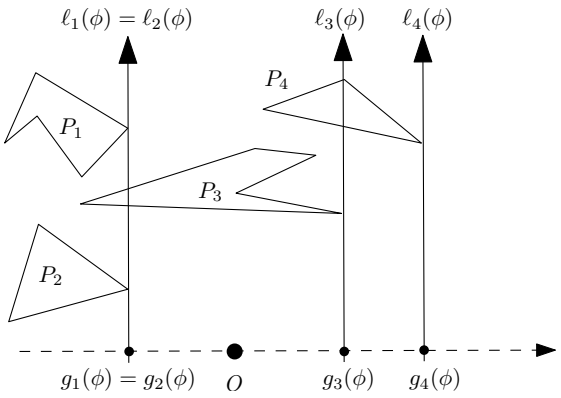

Figure 4: Constructing the vertices at infinity.

But then the fiber $f_{p'}$ lies in $\mathcal{R}(P)$ in $\mathfrak{F}(\mathcal{S})$, and cannot intersect a purple arc at q. □

Lemma 9 allows us to trace purple chains very easily. Once we have located a starting vertex in both $\mathfrak{F}(\mathcal{S}_1)$ and $\mathfrak{F}(\mathcal{S}_2)$, we trace the chain through both diagrams at the same time. We observe that every time it intersects a cell boundary in $\mathfrak{F}(\mathcal{S}_1)$ or $\mathfrak{F}(\mathcal{S}_2)$, we have indeed found a vertex of the purple chain, so the total number of such intersections is only $O(n)$.

When we enter a cell C along a purple arc, we simply follow the arc through the cell as it intersects the fibers of C. We consider the upper and lower chain of C at the same time, and trace the cells of $\mathfrak{F}(\mathcal{S}_1)$ and $\mathfrak{F}(\mathcal{S}_2)$ in parallel. This allows us to charge the cost of tracing to those features of the two cells incident to the fibers the purple chain actually intersects. Since no fiber is intersected by more than one purple arc, the total tracing time is $O(n)$.

4.2 Computing the vertices at infinity

This is relatively easy, making use of the same Davenport-Schinzel arguments used in Lemma 2. For site $P_i \in \mathcal{S}$ and angle $\phi \in [0, 2\pi)$, let $\ell_i(\phi)$ be the oriented line with direction ϕ tangent to P_i and leaving P_i entirely on its left, see Fig. 4. Let $g_i(\phi)$ be the signed distance from the origin to $\ell_i(\phi)$ (positive if the origin lies left of $\ell_i(\phi)$, negative otherwise). If P_i is a polygonal site of complexity m, we first compute the convex hull $\mathrm{CH}(P_i)$ in time $O(m \log m)$, and can then read off a description of the function g_i in time $O(m)$.

We then compute the lower envelope g of the functions g_i. The pure vertices at infinity correspond exactly to the breakpoints of this lower envelope. Since two functions g_i, g_j can intersect at most twice (see the proof of Lemma 2), the lower envelope can be computed in time $O(n \log n)$ [14].

4.3 Computing the mixed vertices

This is the hardest part of the algorithm. We start by computing the randomized[2] point-location data structure of Mulmuley [13] (see also [5, Chapter 6]) for the two given Voronoi diagrams $\mathfrak{F}(\mathcal{S}_1)$ and $\mathfrak{F}(\mathcal{S}_2)$. This data structure only needs two primitive operations:

- For a given point p in the plane, determine whether the query point x lies left or right of p; and
- For an x-monotone line segment or parabola arc γ, determine whether the query point lies above or below γ.

Both cases can be summarized as follows: Given a *comparator* γ, determine on which side of γ the query point x lies. The comparator can be either a line or a parabola arc.

We compute the mixed vertices lying on each medial axis $\mathfrak{M}(P)$ separately. For each tree $\mathcal{T}$ of $\mathfrak{M}(P)$, the intersection $\mathcal{T} \cap \mathcal{R}(P)$ is a connected subtree by Lemma 4. We can determine the *inner* vertices of this subtree easily, by performing a point location operation for each vertex v of $\mathcal{T}$ in $\mathfrak{F}(\mathcal{S}_1)$ and $\mathfrak{F}(\mathcal{S}_2)$.

[2]This is the only use of randomization in our algorithm. It can probably be avoided by using the point location data structure of Edelsbrunner et al. [6] instead.

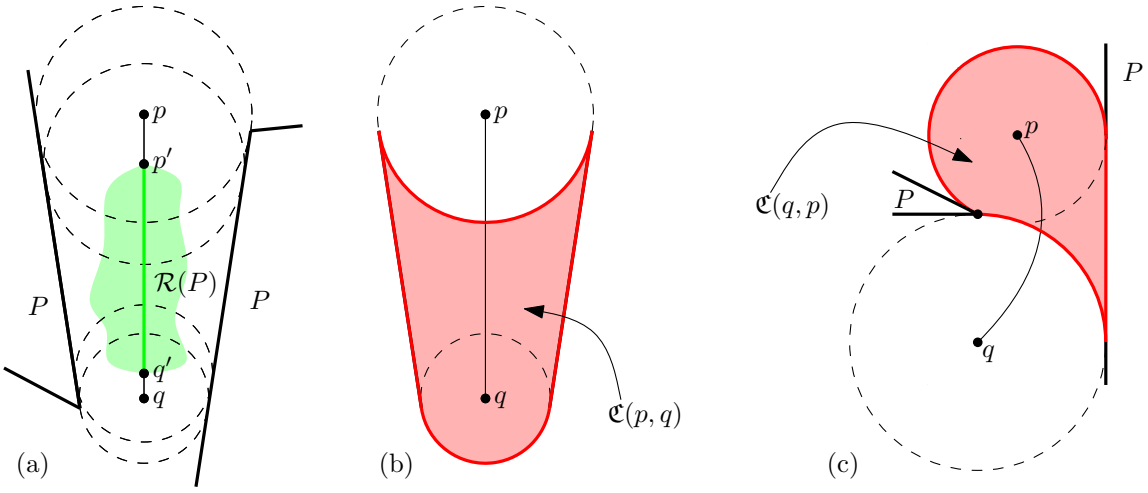

Figure 5: (a) pq is a sub-arc of $\mathfrak{M}(P)$. $p'q'$ is the intersection of pq with $\mathcal{R}(P)$. We also have $p'q' = \mathfrak{M}(P) \cap \mathcal{R}(P)$. (b) The cylinder $\mathfrak{C}(p,q)$ of the pair (p,q) in (a). (c) The cylinder $\mathfrak{C}(q,p)$ of the pair (q,p) for a parabola arc.

This tells us which site is farthest from v—and v lies in $\mathcal{R}(P)$ if and only if this is the site P. Let I be the set of vertices of $\mathcal{T}$ that lie in $\mathcal{R}(P)$. We now need to consider two cases.

If I is non-empty, then every arc α of $\mathcal{T}$ incident to one vertex in I and one vertex not in I must contain exactly one mixed vertex x^* by Lemma 4. We locate this vertex x^* by using *parametric search* along the arc α [11]. The idea is to execute two point location queries in $\mathfrak{F}(\mathcal{S}_1)$ and $\mathfrak{F}(\mathcal{S}_2)$ using x^* as the query point. Each query executes a sequence of primitive operations, where we compare the (unknown) location of x^* with a comparator γ (a line or a parabola arc). This primitive operation can be implemented by intersecting α with γ, resulting in a set of at most four points. In $O(\log n)$ time, we can test for each of these points whether it lies in $\mathcal{R}(P)$. This tells us between which of these points the unknown mixed vertex x^* lies, and we can answer the primitive operation.

It follows that we can execute the two point location queries in time $O(\log^2 n)$, and we obtain the cells C_1 and C_2 of $\mathfrak{F}(\mathcal{S}_1)$ and $\mathfrak{F}(\mathcal{S}_2)$ containing x^*. The mixed vertex x^* lies on the bisector of the features w_1 and w_2 to which these two cells belong (one of them is necessarily a feature of P). We compute the intersection of this bisector with α to obtain x^*.

It remains to consider the case where I is empty, that is, no vertex of $\mathcal{T}$ lies in $\mathcal{R}(P)$. Nevertheless, the region $\mathcal{R}(P)$ may intersect a single arc α of $\mathcal{T}$, and there are then two mixed vertices on α that we need to find. We first need to identify the arcs of $\mathcal{T}$ where this could happen.

Let p and q be two points on the same arc α of $\mathfrak{M}(P)$. We define the *cylinder* $\mathfrak{C}(p,q)$ of the pair (p,q) as

$$\mathfrak{C}(p,q) = \bigcup_{x \in pq} D_P(x) \setminus D_P(p),$$

where the union is taken over all points x on the arc α between p and q, see Fig. 5. We define a condition $G(p,q)$ as follows: Let Q be a site farthest from p, and let w be a feature of Q closest to p. Then $G(p,q)$ is true if $w \in \mathfrak{C}(p,q)$ or if Q intersects $D_P(q)$.

We can now prove the following lemma:

Lemma 10 *Let p, q be points on the same arc α of $\mathfrak{M}(P)$, such that neither p nor q lie in $\mathcal{R}(P)$. If α intersects $\mathcal{R}(P)$ between p and q, then $G(p,q)$ and $G(q,p)$ both hold.*

Proof. Since the statement is symmetric, we only need to show $G(p,q)$. Let Q be a site farthest from p, and let x be a point between p and q on α that lies in $\mathcal{R}(P)$. This implies that Q intersects $D_{\mathcal{S}}(x) = D_P(x)$. Since Q does not intersect $D_P(p)$, we deduce that Q intersects $\mathfrak{C}(p,q)$. If Q does not lie entirely in $\mathfrak{C}(p,q)$, then it must intersect $D_P(q)$, and indeed $G(p,q)$ holds. □

Let us call an arc α connecting vertices p and q of $\mathcal{T}$ a *candidate arc* if $G(p,q)$ and $G(q,p)$ both hold.

Lemma 11 *If α is a candidate arc of a tree $\mathcal{T}$ of $\mathfrak{M}(P)$, then all points in $\mathcal{T} \cap \mathcal{R}(P)$ lie on α.*

Proof. Let α connecting p and q be a candidate arc. Since $G(p,q)$ holds, there is a site $Q \neq P$ such that Q does not intersect $D_{\mathcal{S}}(p)$, but does intersect $\mathfrak{C}(p,q)$. In particular, there must be a point $x \in \alpha$ such that Q intersects $D_{\mathcal{S}}(x) \supset D_P(x)$. Let now y be a point on $\mathcal{T}$ such that the path from q to y goes through α. By Lemma 3, Q cannot intersect $D_P(y)$, and so $y \notin \mathcal{R}(P)$.

Symmetrically, since $G(q,p)$ holds, we find that any point $z \in \mathcal{T}$ such that the path from p to z goes through α cannot be in $\mathcal{R}(P)$, and so any point of $\mathcal{T} \cap \mathcal{R}(P)$ must lie on α. □

Lemma 11 implies immediately that if there are two candidate arcs, then $\mathcal{T} \cap \mathcal{R}(P)$ is empty, and there are no mixed vertices on $\mathcal{T}$.

Since we have point-location data structures for $\mathfrak{F}(\mathcal{S}_1)$ and $\mathfrak{F}(\mathcal{S}_2)$, we can test the condition $G(p,q)$ in time $O(\log n)$ for a given arc α in $\mathfrak{M}(P)$ and two points $p, q \in \alpha$. This allows to identify all candidate arcs in $O(m \log n)$ time, where m is the complexity of $\mathcal{T}$. If there is zero or more than one candidate arcs, we can stop immediately, as there are no mixed vertices on $\mathcal{T}$.

It remains to consider that there is a single candidate arc α in $\mathcal{T}$. We again apply parametric search, using an unknown point x^* in $\alpha \cap \mathcal{R}(P)$ as the query point. During the point-location query, we maintain an interval pq on α that must contain x^* (if x^* exists at all). To implement a primitive query, we must determine the location of x^* with respect to a comparator γ. If the current interval pq on α does not intersect γ, we can proceed immediately, otherwise we get a sequence of points $p = x_0, x_1, x_2, \ldots, x_s = q$ on α, where $2 \leqslant s \leqslant 5$. We first test if some $x_i \in \mathcal{R}(P)$ in $O(\log n)$ time. If so, we abort the process, and use the method discussed above to find the mixed vertices on the arc between p and x_i and between x_i and q. If no x_i lies in $\mathcal{R}(P)$, we test the conditions $G(x_i, x_{i+1})$ and $G(x_{i+1}, x_i)$ for each consecutive pair. If the conditions hold for no pair or more than one pair, we can stop immediately, as there cannot be a point of $\mathcal{R}(P)$ on α. If there is exactly one pair, we have found the location of x^* with respect to the comparator γ, and we continue the point location query.

If both point location queries terminate without encountering a point in $\mathcal{R}(P)$, we have found the cells C_1 and C_2 of $\mathfrak{F}(\mathcal{S}_1)$ and $\mathfrak{F}(\mathcal{S}_2)$ containing x^*. We can then compute a point $x \in \alpha \cap \mathcal{R}(P)$ in constant time, and use again the method above to find the two mixed vertices on α.

5 Connectedness of Voronoi regions

So far, we have developed the theory of farthest-polygon Voronoi diagrams only as far as we needed it for the complexity bound and the algorithm. In this section, we give two interesting properties on the Voronoi regions. First, we prove that the Voronoi region of a single polygonal site can have $k-1$ connected components. Second, we prove that a bounded Voronoi region is necessarily (simply) connected.

Lemma 12 *A single Voronoi region can have $k-1$ connected components.*

Proof. The construction is shown in Fig. 6. It consists in one $(k-1)$-regular polygon R and $k-1$ polygonal chains $C_1, \ldots, C_{k-1}$. Let $e_1, e_2, \ldots, e_{k-1}$ denote the edges of R in circular order. We inductively construct the polygonal sites $C_i, i = 1, 2, \ldots, k-1$ as follows. For a supporting line l of e_i, let l^+ be the closed halfplane containing R and l^- be the other. Then, consider the intersection C_i^* between l^+ and the four edges of a square that contains $R, C_1, \ldots, C_{i-1}$ inside. We define C_i as the set of points of C_i^* whose distance to l is larger than some fixed small $\varepsilon > 0$. C_i has complexity at most 4. Note that l^+ contains C_i completely. Consider a ray from e_i to infinity in l^- which is orthogonal to l. Since l^- intersects all sites but C_i, the infinite endpoint of this ray lies in the region $\mathcal{C}_i = \mathcal{R}(C_i)$ at infinity. On the other hand, for a sufficiently small ε, there is a line l passing through v_i (the vertex incident to e_{i-1} and e_i) such that we can define l^+ as an open halfplane containing $R \backslash \{v_i\}$ and l^- as the other open halfplane intersecting all the other sites but R. The infinite endpoint of the ray from v_i to infinity in l^- which is orthogonal to l lies in a connected component of $\mathcal{R}(R)$, which we call $\mathcal{R}_i$.

Therefore at infinity $\mathcal{C}_1, \mathcal{R}_1, \mathcal{C}_2, \cdots, \mathcal{C}_{k-1}, \mathcal{R}_{k-1}$ appear in turn. Note that for a point x in the region $\mathcal{R}(R)$ of R, its fiber f_x is an infinite ray because R is convex. For $i = 1, 2, \ldots, k = 1$, consider the half-line $x_i\omega_i$ from an infinite point $\omega_i \in \mathcal{C}_i$ to a point $x_i \in R$ closest to ω_i. If $x \in x_i\omega_i$ lies in $\mathcal{R}(R)$, then $f_x \subset \mathcal{R}(R)$, which is impossible because $\omega_i \in \mathcal{C}_i$. Define $W = R \cup \bigcup_i x_i\omega_i$ (W looks like a hand-drawn sun). We then have $W \cap \mathcal{R}(R) = \emptyset$, and $\mathbb{R}^2 \setminus W$ consists in $k-1$ unbounded connected subsets of the plane. The connected subset bounded by $x_i\omega_i$, $x_{i+1}\omega_{i+1}$ and R contains $\mathcal{R}_i$ completely. It follows that

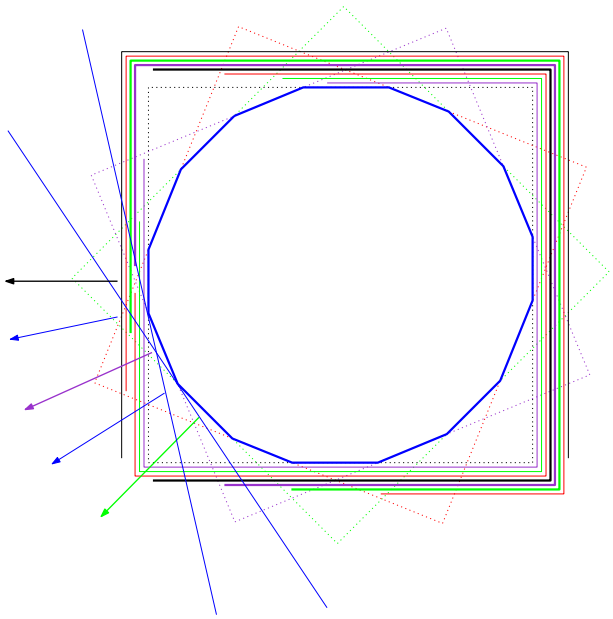

Figure 6: The Voronoi region of the middle polygon has $k-1$ connected components.

$\mathcal{R}_i \neq \mathcal{R}_j$ when $i \neq j$. □

We now turn our attention to bounded regions in $\mathfrak{F}(\mathcal{S})$. Let P be a site of $\mathcal{S}$, whose Voronoi region, $\mathcal{R}(P)$, is bounded and non-empty. We prove that $\mathcal{R}(P)$ is simply connected. Due to space limitation in this extended abstract, we only sketch the following proofs.

Lemma 13 *P properly contains a site $Q \neq P$ inside one of its pockets.*

Proof. We first observe that $\mathcal{R}(P)$ contains some points of the medial axis $\mathfrak{M}(P)$ of P. Indeed, let $x \in \mathcal{R}(P)$ and consider its fiber f_x. Since $\mathcal{R}(P)$ is bounded, f_x does not extend to infinity and therefore touches $\mathfrak{M}(P)$.

Let p be a point in $\mathcal{R}(P) \cap \mathfrak{M}(P)$. If p lies in a bounded pocket of P, then clearly this pocket does contain all the other sites in $\mathcal{S} \setminus \{P\}$ and the lemma is proved. Otherwise, we let p move along the medial axis $\mathfrak{M}(P)$ up to infinity. At some point p^*, p gets out of $\mathcal{R}(P)$. This means that $D_{\mathcal{S}}(p^*) = D_P(p^*)$ is tangent to another site Q without intersecting it properly. We can then argue that the site Q lies in a pocket of P. □

Lemma 14 *$\mathcal{R}(P)$ is simply connected.*

Proof. We first observe that P cannot properly contain two sites in two different pockets. Indeed, in such a case, the region of P is empty, since no disc avoiding P can intersect two of its pockets. But this contradicts our hypothesis that $\mathcal{R}(P)$ is non-empty.

Let $\mathbb{P}$ be the pocket of P containing another site. As argued in the proof of lemma 13, it is clear that the tree $\mathcal{T}$ of $\mathfrak{M}(P)$ is the only tree of $\mathfrak{M}(P)$ whose intersection with $\mathcal{R}(P)$ is not empty. By lemma 4, $\mathcal{T} \cap \mathcal{R}(P)$ is a connected sub-tree of $\mathcal{T}$. By letting the points of $\mathcal{R}(P)$ move along their fiber, we can design a (continuous) deformation retraction of $\mathcal{R}(P)$ onto $\mathcal{T} \cap \mathcal{R}(P)$ (we omit the details in this extended abstract). Since $\mathcal{T} \cap \mathcal{R}(P)$ is simply connected, so is $\mathcal{R}(P)$. □

Acknowledgments

We thank the participants of the 8th Korean Workshop on Computational Geometry, organized by Tetsuo Asano at JAIST, Kanazawa, Japan, Aug. 1–6, 2005.

References

[1] M. Abellanas, F. Hurtado, C. Icking, R. Klein, E. Langetepe, L. Ma, B. Palop, and V. Sacristán. The farthest color Voronoi diagram and related problems. In *Abstracts 17th European Workshop Comput. Geom.*, pages 113–116. Freie Universität Berlin, 2001.

[2] M. Abellanas, F. Hurtado, C. Icking, R. Klein, E. Langetepe, L. Ma, B. Palop, and V. Sacristán. Smallest color-spanning objects. In *Proc. 9th Annu. European Sympos. Algorithms. Lecture Notes Comput. Sci.*, vol. 2161, pages 278–289. Springer-Verlag, 2001.

[3] F. Aurenhammer and R. Klein. Voronoi diagrams. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 201–290. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 2000.

[4] F. Aurenhammer, R.L.S.Drysdale, and H. Krasser. Farthest line segment Voronoi diagrams. *Information Processing Letters*, 100:220–225, 2006.

[5] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, 2nd edition, 2000.

[6] H. Edelsbrunner, L. J. Guibas, and J. Stolfi. Optimal point location in a monotone subdivision. *SIAM Journal on Computing*, 15, 1986.

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

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

[9] S. Jadhav, A. Mukhopadhyay, and B. K. Bhattacharya. An optimal algorithm for the intersection radius of a set of convex polygons. *J. Algorithms*, 20:244–267, 1996.

[10] M. van Kreveld and T. Schlechter. Automated label placement for groups of islands. In *Proc. of the 22nd International Cartographic Conference*, 2005.

[11] N. Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *J. ACM*, 30:852–865, 1983.

[12] K. Mehlhorn, S. Meiser, and R. Rasch. Furthest site abstract Voronoi diagrams. *Int. J. Comput. Geom. & Appl.*, 11:583–616, 2001.

[13] K. Mulmuley. A fast planar partition algorithm, I. *J. Symbolic Comput.*, 10:253–280, 1990.

[14] M. Sharir and P. K. Agarwal. *Davenport-Schinzel Sequences and Their Geometric Applications*. Cambridge University Press, New York, 1995.