The Voronoi Diagram of Planar Convex Objects*

Menelaos I. Karavelas[1] and Mariette Yvinec[2]

[1] University of Notre Dame, Computer Science and Engineering Department,
Notre Dame, IN 46556, U.S.A.
mkaravel@cse.nd.edu
[2] INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93,
06902 Sophia-Antipolis Cedex, France
Mariette.Yvinec@sophia.inria.fr

Abstract. *This paper presents a dynamic algorithm for the construction of the Euclidean Voronoi diagram of a set of convex objects in the plane. We consider first the Voronoi diagram of smooth convex objects forming pseudo-circles set. A pseudo-circles set is a set of bounded objects such that the boundaries of any two objects intersect at most twice. Our algorithm is a randomized dynamic algorithm. It does not use a conflict graph or any sophisticated data structure to perform conflict detection. This feature allows us to handle deletions in a relatively easy way. In the case where objects do not intersect, the randomized complexity of an insertion or deletion can be shown to be respectively* $O(\log^2 n)$ *and* $O(\log^3 n)$*. Our algorithm can easily be adapted to the case of pseudo-circles sets formed by piecewise smooth convex objects. Finally, given any set of convex objects in the plane, we show how to compute the restriction of the Voronoi diagram in the complement of the objects' union.*

1 Introduction

Given a set of sites and a distance function from a point to a site, a Voronoi diagram can be roughly described as the partition of the space into cells that are the locus of points closer to a given site than to any other site. Voronoi diagrams have proven to be useful structures in various fields such as astronomy, crystallography, biology etc. Voronoi diagrams have been extensively studied. See for example the survey by Aurenhammer and Klein [1] or the book by Okabe, Boots, Sugihara and Chiu [2]. The early studies were mainly concerned with point sites and the Euclidean distance. Subsequent studies considered extended sites such has segments, lines, convex polytopes and various distances such as L_1 or L_∞ or any distance defined by a convex polytope as unit ball. While the complexity and the related algorithmic issues of Voronoi diagrams for extended sites in higher dimensions is still not completely understood, as witnessed in the recent works by Koltun and Sharir [3, 4], the planar cases are now rather well masterized, at least for linear objects.

* Work partially supported by the IST Programme of the EU as a Shared-cost RTD (FET Open) Project IST-2000-26473 (ECG - Effective Computational Geometry for Curves and Surfaces).

The rising need for handling curved objects triggered further works for the planar cases. Klein et al. [5, 6] set up a general framework of *abstract Voronoi diagrams* which covers a large class of planar Voronoi diagrams. They provided a randomized incremental algorithm to construct diagrams of this class. Alt and Schwarzkopf [7] handled the case of generic planar curves and described a randomized algorithm for this case. Since they handle curves, they cannot handle objects with non-empty interior, which is our focus. Their algorithm is incremental but does not work in-line (it requires the construction of a Delaunay triangulation with one point on each curve before the curve segments are really treated). Another closely related work is that by McAllister, Krikpatrick and Snoeyink [8], which deals with the Voronoi diagrams of disjoint convex polygons. The algorithm presented treats the convex polygons as objects, rather than as collections of segments; it follows the sweep-line paradigm, thus it is not dynamic. Moreover, the case of intersecting convex polygons is not considered. The present paper deals with the Euclidean Voronoi diagram of planar smooth or piecewise smooth convex objects, and generalizes a previous work of the same authors on the Voronoi diagram of circles [9].

Let p be a point and A be a bounded convex object in the Euclidean plane $\mathbb{E}^2$. We define the distance $\delta(p, A)$ from p to A to be:

$$\delta(p, A) = \begin{cases} \min_{x \in \partial A} \|p - x\|, & p \notin A \\ -\min_{x \in \partial A} \|p - x\|, & p \in A \end{cases}$$

where ∂A denotes the boundary of A and $\|\cdot\|$ denotes the Euclidean norm. Given the distance $\delta(\cdot, \cdot)$ and a set of convex objects $\mathcal{A} = \{A_1, \ldots, A_n\}$, the *Voronoi diagram* $\mathcal{V}(\mathcal{A})$ is the planar partition into cells, edges and vertices defined as follows. The Voronoi cell of an object A_i is the set of points which are closer to A_i than to any other object in $\mathcal{A}$. Voronoi edges are maximal connected sets of points equidistant to two objects in $\mathcal{A}$ and closer to these objects than to any other in $\mathcal{A}$. Voronoi vertices are points equidistant to at least three objects of $\mathcal{A}$ and closer to these objects than to any other object in $\mathcal{A}$.

We first consider Voronoi diagrams for special collections of smooth convex objects called *pseudo-circles sets*. A pseudo-circles set is a set of bounded objects such that the boundaries of any two objects in the set have at most two intersection points. In the sequel, unless specified otherwise, we consider pseudo-circles sets formed by smooth convex objects, and we call them *smooth convex pseudo-circles sets*, or *sc-pseudo-circles sets* for short.

Let A be a convex object. A line L is a *supporting line* of A iff A is included in one of the closed half-planes bounded by L, and $\partial A \cap L$ is not empty. Given two convex objects A_i and A_j, a line L is a *(common) supporting line* of A_i and A_j iff L is a supporting line of A_i and A_j, such that A_i and A_j are both included in the same half-plane bounded by L. In this paper, we first deal with smooth bounded convex objects forming pseudo-circles sets. Any two objects in such a set have at most two common supporting lines. Two convex objects have no common supporting line if one is included in the other. They have two common supporting lines if they are either disjoint or properly intersecting at two points (a proper intersection point is a point where the boundaries are not

only touching but also crossing each other) or externally tangent (which means that their interiors are disjoint and their boundaries share a common tangent point). Two objects forming a pseudo-circles set may also be internally tangent, meaning that one is included in the other and their boundaries share one or two common points. Then they have, respectively, one or two common supporting lines. A pseudo-circles set is said to be in *general position* if there is no pair of tangent objects. In fact, tangent objects which are properly intersecting at their common tangent point or externally tangent objects do not harm our algorithm and we shall say that a pseudo-circles set is in general position when there is no pair of internally tangent objects.

The algorithm that we propose for the construction of the Voronoi diagram of sc-pseudo-circles sets in general position is a dynamic one. It is a variant of the incremental randomized algorithm proposed by Klein et al. [6]. The data structures used are simple, which allows us to perform not only insertions but also deletions of sites in a relatively easy way. When input sites are allowed to intersect each other, it is possible for a site to have an empty Voronoi cell. Such a site is called *hidden*, otherwise *visible*. Our algorithm handles hidden sites. The detection of the first conflict or the detection of a hidden site is performed through closest site queries. Such a query can be done by either a simple walk in the Voronoi diagram or using a hierarchy of Voronoi diagrams, i.e., a data structure inspired from the Delaunay hierarchy of Devillers [10].

To analyze the complexity of the algorithm, we assume that each object has constant complexity, which implies that each operation involving a constant number of objects is performed in constant time. We show that if sites do not intersect, the randomized complexity of updating a Voronoi diagram with n sites is $O(\log^2 n)$ for an insertion and $O(\log^3 n)$ for a deletion. The complexities of insertions and deletions are more involved when sites intersect.

We then extend our results by firstly dropping the hypothesis of general position and secondly by dealing with pseudo-circles sets formed by convex objects whose boundaries are only piecewise smooth. Using this extension, we can then build the Voronoi diagram of any set $\mathcal{A}$ of convex objects in the complement of the objects' union (i.e., in free space). This is done by constructing a new set of objects $\mathcal{A}'$, which is a pseudo-circles set of piecewise smooth convex objects and such that the Voronoi diagrams $\mathcal{V}(\mathcal{A})$ and $\mathcal{V}(\mathcal{A}')$ coincide in free space.

The rest of the paper is structured as follows. In Section 2 we study the properties of the Voronoi diagram of sc-pseudo-circles sets in general position, and show that such a diagram belongs to the class of abstract Voronoi diagrams. In Section 3 we present our dynamic algorithm. Section 4 describes closest site queries, whereas Section 5 deals with the complexity analysis of insertions and deletions. Finally, in Section 6 we discuss the extensions of our approach.

2 The Voronoi diagram of sc-pseudo-circles sets

In this section we present the main properties of the Voronoi diagram of sc-pseudo-circles sets in general position. We first provide a few definitions and

notations. Henceforth, we consider any bounded convex object A_i as closed and we note ∂A_i and A_i°, respectively, the boundary and the interior of A_i.

Let $\mathcal{A} = \{A_1, \ldots, A_n\}$ be an sc-pseudo-circles set. The Voronoi cell of an object A is denoted as $V(A)$ and is considered a closed set. The interior and boundary of $V(A)$ are denoted by $V^\circ(A)$ and $\partial V(A)$, respectively. We are going to consider maximal disks either included in a given object A_i or disjoint from A_i°, where the term maximal refers to the inclusion relation. For any point x, we denote by $C_i(x)$ the closed disk centered at x with radius $|\delta(x, A_i)|$. If $x \notin A_i$, $C_i(x)$ is the maximal disk centered at x and disjoint from A_i°. If $x \in A_i$, $C_i(x)$ is the maximal disk centered at x and included in A_i. In the latter case there is a unique maximal disk inside A_i containing $C_i(x)$, which we denote by $M_i(x)$. Finally, the *medial axis* $S(A_i)$ of a bounded convex object A_i is defined as the locus of points that are centers of maximal disks included in A_i.

Let A_i and A_j be two smooth bounded convex objects. The set of points $p \in \mathbb{E}^2$ that are at equal distance from A_i and A_j is called the bisector π_{ij} of A_i and A_j. Theorem 1 ensures that π_{ij} is an one-dimensional set if the two objects A_i and A_j form an sc-pseudo-circles set in general position and justifies the definition of Voronoi edges given above. Theorem 2 ensures that each cell in the Euclidean Voronoi diagram of an sc-pseudo-circles set in general position is simply connected. The proofs of Theorems 1 and 2 below are omitted for lack of space.

Theorem 1 *Let $\{A_i, A_j\}$ be an sc-pseudo-circles set in general position and let π_{ij} be the bisector of A_i and A_j with respect to the Euclidean distance $\delta(\cdot, \cdot)$. Then: (1) if A_i and A_j have no common supporting line, $\pi_{ij} = \emptyset$; (2) if A_i and A_j have two common supporting lines, π_{ij} is a single curve homeomorphic to the open interval $(0, 1)$.*

Theorem 2 *Let $\mathcal{A} = \{A_1, \ldots, A_n\}$ be an sc-pseudo-circles set in general position. For each object A_i, we denote by $N(A_i)$ the locus of the centers of maximal disks included in A_i that are not included in the interior of any object in $\mathcal{A} \setminus \{A_i\}$, and by $N^\circ(A_i)$ the locus of the centers of maximal disks included in A_i that are not included in any object in $\mathcal{A} \setminus \{A_i\}$. Then: (1) $N(A_i) = S(A_i) \cap V(A_i)$ and $N^\circ(A_i) = S(A_i) \cap V^\circ(A_i)$; (2) $N(A_i)$ and $N^\circ(A_i)$ are simply connected sets; (3) the Voronoi cell $V(A_i)$ is weakly star-shaped with respect to $N(A_i)$, which means that any point of $V(A_i)$ can be connected to a point in $N(A_i)$ by a segment included in $V(A_i)$. Analogously, $V^\circ(A_i)$ is weakly star-shaped with respect to $N^\circ(A_i)$; (4) $V(A_i) = \emptyset$ iff $N(A_i) = \emptyset$ and $V^\circ(A_i) = \emptyset$ iff $N^\circ(A_i) = \emptyset$.*

In the sequel we say that an object A is *hidden* if $N^\circ(A) = \emptyset$.

In the framework of abstract Voronoi diagrams introduced by Klein [5], the diagram is defined by a set of bisecting curves $B_{i,j}$. In this framework, a set of bisectors is said to be *admissible* if: (1) each bisector is homeomorphic to a line; (2) the closures of the Voronoi regions covers the entire plane; (3) regions are path connected. (4) two bisectors intersect in at most a finite number of connected components. Let us show that Euclidean Voronoi diagrams of sc-pseudo-circles, such that any pair of objects has exactly two supporting lines,

fit into the framework of abstract Voronoi diagrams. Theorems 1 and 2 ensure, respectively, that Conditions 1 and 3 are fulfilled. Condition 2 is granted for any diagram induced by a distance. Condition 4 is a technical condition that we have not explicitly proved. In our case this results indeed from the assumption that the objects have constant complexity. The converse is also true: if we have a set of convex objects in general position, then their bisectors form an admissible system only if every pair of objects has exactly two supporting lines. Indeed, if this is not the case, one of the following holds : (1) the bisector is empty; (2) the bisector is homeomorphic to a ray; (3) there exist Voronoi cells that consist of more than one connected components.

Theorem 3 *Let $A = \{A_1, \ldots, A_n\}$ be a set of smooth convex objects of constant complexity and in general position. The set of bisectors π_{ij} is an admissible system of bisectors iff every pair of objects has exactly two supporting lines.*

3 The dynamic algorithm

The algorithm that we propose is a variant of the randomized incremental algorithm for abstract Voronoi diagrams proposed by Klein and al. [6]. Our algorithm is fully dynamic and maintains the Voronoi diagram when a site is either added to the current set or deleted from it. To facilitate the presentation of the algorithm we first define the compactified version of the diagram and introduce the notion of conflict region.

The compactified diagram. We call 1-skeleton of the Voronoi diagram, the union of the Voronoi vertices and Voronoi edges. The 1-skeleton of the Voronoi diagram of an sc-pseudo-circles set $\mathcal{A}$ may consist of more than one connected components. However, we can define a compactified version of the diagram by adding to $\mathcal{A}$ a spurious site, A_∞ called the infinite site. The bisector of A_∞ and $A_i \in \mathcal{A}$ is a closed curve at infinity, intersecting any unbounded edge of the original diagram (see for example [5]). In the sequel we consider such a compactified version of the diagram, in which case the 1-skeleton is connected.

The conflict region. Each point x on a Voronoi edge incident to $V(A_i)$ and $V(A_j)$ is the center of a disk $C_{ij}(x)$ tangent to the boundaries ∂A_i and ∂A_j. This disk is called a *Voronoi bitangent disk*, and more precisely an interior Voronoi bitangent disk if it is included in $A_i \cap A_j$, or an exterior Voronoi bitangent disk if it is lies in the complement of $A_i^\circ \cup A_j^\circ$. Similarly, a Voronoi vertex that belongs to the cells $V(A_i)$, $V(A_j)$ and $V(A_k)$ is the center of a disk $C_{ijk}(x)$ tangent to the boundaries of A_i, A_j and A_k. Such a disk is called a *Voronoi tritangent disk*, and more precisely an interior Voronoi tritangent disk if it is included in $A_i \cap A_j \cap A_k$, or an external Voronoi tritangent disk if it is lies in the complement of $A_i^\circ \cup A_j^\circ \cup A_k^\circ$.

Suppose we want to add a new object $A \notin \mathcal{A}$ and update the Voronoi diagram from $\mathcal{V}(\mathcal{A})$ to $\mathcal{V}(\mathcal{A}^+)$ where $\mathcal{A}^+ = \mathcal{A} \cup \{A\}$. We assume that $\mathcal{A}^+$ is also an sc-pseudo-circles set. The object A is said to be in conflict with a point x on the 1-skeleton of the current diagram if the Voronoi disk centered at x is either an

internal Voronoi disk included in A° or an exterior Voronoi disk intersecting A°. We call *conflict region* the subset of the 1-skeleton of $\mathcal{V}(\mathcal{A})$ that is in conflict with the new object A. A Voronoi edge of $\mathcal{V}(\mathcal{A})$ is said to be in conflict with A if some part of this edge is in conflict with A.

Our dynamic algorithm relies on the two following theorems, which can be proved as in [6].

Theorem 4 *Let $\mathcal{A}^+ = \mathcal{A} \cup \{A\}$ be an sc-pseudo-circles set such that $A \notin \mathcal{A}$. The conflict region of A with respect to $\mathcal{V}(\mathcal{A})$ is a connected subset of the 1-skeleton of $\mathcal{V}(\mathcal{A})$.*

Theorem 5 *Let $\{A_i, A_j, A_k\}$ be an sc-pseudo-circles set in general position. Then the Voronoi diagram of A_i, A_j and A_k has at most two Voronoi vertices.*

Theorem 5 is equivalent to saying that two bisecting curves π_{ij} and π_{ik} relative to the same object A_i have at most two points of intersection. In particular, it implies that the conflict region of a new object A contains at most two connected subsets of each edge of $\mathcal{V}(\mathcal{A})$.

The data structures. The Voronoi diagram $\mathcal{V}(\mathcal{A})$ of the current set of objects is maintained through its dual graph $\mathcal{D}(\mathcal{A})$.

When a deletion is performed, a hidden site can reappear as visible. Therefore, we have to keep track of hidden sites. This is done through an additional data structure that we call the *covering graph* $\mathcal{K}(\mathcal{A})$. For each hidden object A_i, we call *covering set* of A_i a set $K(A_i)$ of objects such that any maximal disk included in A_i is included in the interior of at least one object of $K(A_i)$. In other words, in the Voronoi diagram $\mathcal{V}(K(A_i) \cup \{A_i\})$ the Voronoi cell $V(A_i)$ of A_i is empty. The covering graph is a directed acyclic graph with a node for each object. A node associated to a visible object is a root. The parents of a hidden object A_i are objects that form a covering set of A_i. The parents of a hidden object may be hidden or visible objects.

Note that if we perform only insertions or if it is known in advance that all sites will have non-empty Voronoi cells (e.g., this is the case for disjoint objects), it is not necessary to maintain a covering graph.

The algorithm needs to perform nearest neighbor queries. Optionally, the algorithm maintains a location data structure to perform efficiently those queries. The location data structure that we prone here is called a Voronoi hierarchy and described below in subsection 4.

3.1 The insertion procedure

The insertion of a new object A in the current Voronoi diagram $\mathcal{V}(\mathcal{A})$ involves the following steps: (1) find a first conflict between an edge of $\mathcal{V}(\mathcal{A})$ and A or detect that A is hidden in $\mathcal{A}^+$; (2) find the whole conflict region of A; (3) repair the dual graph; (4) update the covering graph; (5) update the location data structure if any. Steps 1 and 4 are discussed below. Steps 2 and 3 are performed exactly as in [9] for the case of disks. Briefly, Step 2 corresponds to finding the

boundary of the star of A in $\mathcal{D}(\mathcal{A}^+)$. This boundary represents a hole in $\mathcal{D}(\mathcal{A})$, i.e., a sequence of edges of $\mathcal{D}(\mathcal{A})$ forming a topological circle. Step 3 simply amounts to "staring" this hole from A_i (which means to connect A_i to every vertex on the hole boundary).

Finding the first conflict or detecting a hidden object. The first crucial operation to perform when inserting a new object is to determine if the inserted object is hidden or not. If the object is hidden we need to find a covering set of this object. If the object is not hidden we need to find an edge of the current diagram in conflict with the inserted object.

The detection of the first conflict is based on closest site queries. Such a query takes a point x as input and asks for the object in the current set $\mathcal{A}$ that is closest to x. If we didn't have any location data structure, then we perform the following *simple walk* on the Voronoi diagram to find the object in $\mathcal{A}$ closest to x. The walk starts from any object $A_i \in \mathcal{A}$ and compares the distance $\delta(x, A_i)$ with the distances $\delta(x, A)$ to the neighbors A of A_i in the Voronoi diagram $\mathcal{V}(\mathcal{A})$. If some neighbor A_j of A_i is found closer to x than A_i, the walk proceeds to A_j. If there is no neighbor of A_i that is closer to x than A_i, then A_i is the object closest to x among all objects in $\mathcal{A}$. It is easy to see that this walk can take linear time. We postpone until the next section the description of the location data structure and the way these queries can be answered more efficiently.

Let us consider first the case of disjoint objects. In this case there are no hidden objects and each object is included in its own cell. We perform a closest site query for any point p of the object A to be inserted. Let A_i be the object of $\mathcal{A}$ closest to p. The cell of A_i will shrink in the Voronoi diagram $\mathcal{V}(\mathcal{A}^+)$ and at least one edge of $\partial V(A_i)$ is in conflict with A. Hence, we only have to look at the edges of $\partial V(A_i)$ until we find one in conflict with A.

When objects do intersect, we perform an operation called *location of the medial axis*, which either provides an edge of $\mathcal{V}(\mathcal{A})$ that is in conflict with A, or returns a covering set of A. There is a simple way to perform this operation. Indeed, the medial axis $S(A)$ of A is a tree embedded in the plane, and for each object A_i, the part of $S(A)$ that is not covered by A_i (that is the part of $S(A)$ made up by the centers of maximal disks in A, not included in A_i) is connected. We start by choosing a leaf vertex p of the medial axis $S(A)$ and locate the object A_i that is closest to p. Then we prune the part of the medial axis covered by A_i and continue with the remainder of the medial axis in exactly the same way. If, at some point, there is no part of $S(A)$ left, we know that A is hidden, and the set of objects A_i, which pruned a part of $S(A)$, forms a covering of A. Otherwise we perform a nearest neighbor query for any point of $S(A)$ which has not been pruned. A first conflict can be found from the answer to this query in exactly the same way as in the case of disjoint objects, discussed above.

It remains to explain how we choose the objects A_i that are candidates for covering parts of $S(A)$. As described above, we determine the first object A_i by performing a nearest neighbor query for a leaf vertex p of $S(A)$. Once we have pruned the medial axis, we consider one of the leaf vertices p' created after the pruning. This corresponds to a maximal circle $M(p')$ of A centered at p', which is

also internally tangent to A_i. To find a new candidate object for covering $S(A)$, we simply need to find a neighbor of A_i in the Voronoi diagram that contains $M(p')$; if $M(p')$ is actually covered by some object in $\mathcal{A}$, then it is guaranteed that we will find one among the neighbors of A_i. We then continue, as above, with the new leaf node of the pruned medial axis and the new candidate covering object, as above.

Updating the covering graph. We now describe how Step 4 of the insertion procedure is performed. We start by creating a node for A in the covering graph.

If A is hidden, the location of its medial axis yields a covering set $K(A)$ of A. In the covering graph we simply assign the objects in $K(A)$ as parents of A.

If the inserted object A is visible, some objects in $\mathcal{A}$ can become hidden due to the insertion of A. The set of objects that become hidden because of A are provided by Step 2 of the insertion procedure. They correspond to cycles in the conflict region of A. The main idea for updating the covering graph is to look at the neighbors of A in the new Voronoi diagram.

Lemma 6 *Let $\mathcal{A}$ be an sc-pseudo-circles set. Let $A \notin \mathcal{A}$ be an object such that $\mathcal{A}^+ = \mathcal{A} \cup \{A\}$ is also an sc-pseudo-circles set and A is visible in $\mathcal{V}(\mathcal{A}^+)$. If an object $A_i \in \mathcal{A}$ becomes hidden upon the insertion of A, then the neighbors of A in $\mathcal{V}(\mathcal{A}^+)$ along with A is a covering set of A_i.*

Let A_i be an object that becomes hidden upon the insertion of A. By Lemma 6 the set of neighbors of A in $\mathcal{V}(\mathcal{A}^+)$ along with A is a covering set $K(A_i)$ of A_i. The only modification we have to do in the covering graph is to assign all objects in $K(A_i)$ as parents of A_i.

Updating the location data structure. The update of the location data structure is really simple. Let A be the object inserted. If A is hidden we do nothing. If A is not hidden, we insert A in the location data structure, and delete from it all objects than become hidden because of the insertion of A.

3.2 The deletion procedure

Let A_i be the object to be deleted and let $K_p(A_i)$ be the set of all objects in the covering graph $\mathcal{K}(\mathcal{A})$ that have A_i as parent. The deletion of A_i involves the following steps: (1) remove A_i from the dual graph; (2) remove A_i from the covering graph; (3) remove A_i from location data structure; (4) reinsert the objects in $K_p(A_i)$.

Step 1 requires no action if A_i is hidden. If A_i is visible, we first build an annex Voronoi diagram for the neighbors of A_i in $\mathcal{V}(\mathcal{A})$ and use this annex Voronoi diagram to fill in the cell of A_i (see [9]). In Step 2, we simply delete all edges of $\mathcal{K}(\mathcal{A})$ to and from A_i, as well as the node corresponding to A_i. In Step 3, we simply delete A_i from the location data structure. Finally, in Step 4 we apply the insertion procedure to all objects in $K_p(A_i)$. Note, that if A_i is hidden, this last step simply amounts to finding a new covering set for all objects in $K_p(A_i)$.

4 Closest site queries

The location data structure is used to answer closest site queries. A closest site query takes as input a point x and asks for the object in the current set $\mathcal{A}$ that is closest to x. Such queries can be answered through a simple walk in the Voronoi diagram (as described in the previous section) or using a hierarchical data structure called the Voronoi hierarchy.

The Voronoi hierarchy. The hierarchical data structure used here, denoted by $\mathcal{H}(\mathcal{A})$, is inspired from the Delaunay hierarchy proposed by Devillers [10]. The method consists of building the Voronoi diagrams $\mathcal{V}(\mathcal{A}_\ell)$, $\ell = 0, \ldots, L$, of a hierarchy $\mathcal{A} = \mathcal{A}_0 \supseteq \mathcal{A}_1 \supseteq \ldots \supseteq \mathcal{A}_L$ of subsets of $\mathcal{A}$. Our location data structure conceptually consists of all subsets A_ℓ, $1 \leq \ell \leq L$.

The hierarchy $\mathcal{H}(\mathcal{A})$ is built together with the Voronoi diagram $\mathcal{V}(\mathcal{A})$ according to the following rules. Any object of $\mathcal{A}$ is inserted in $\mathcal{V}(\mathcal{A}_0) = \mathcal{V}(\mathcal{A})$. If A has been inserted in $\mathcal{V}(\mathcal{A}_\ell)$ and is visible, it is inserted in $\mathcal{V}(\mathcal{A}_{\ell+1})$ with probability β. If, upon the insertion of A in $\mathcal{V}(\mathcal{A})$, an object becomes hidden it is deleted from all diagrams $\mathcal{V}(\mathcal{A}_{\ell'})$, $\ell' > 0$, in which it has been inserted. Finally, when an object A_i is deleted from the Voronoi diagram $\mathcal{V}(\mathcal{A})$, we delete A_i from all diagrams $\mathcal{V}(\mathcal{A}_\ell)$, $\ell \geq 0$, in which it has been inserted. Note that all diagrams $\mathcal{V}(\mathcal{A}_\ell)$, $\ell > 0$, do not contain any hidden objects.

The closest site query for a point x is performed as follows. The query is first performed in the top-most diagram $\mathcal{V}(\mathcal{A}_L)$ using the simple walk. Then, for $\ell = L-1, \ldots, 0$ a simple walk is performed in $\mathcal{V}(A_\ell)$ from $A_{\ell+1}$ to A_ℓ where $A_{\ell+1}$ (resp. A_ℓ) is the object of $\mathcal{A}_{\ell+1}$ (resp. of $\mathcal{A}_\ell$) closest to x.

It easy to show that the expected size of $\mathcal{H}(\mathcal{A})$ is $O(\frac{1}{1-\beta}\, n)$, and that the expected number of levels in $\mathcal{H}(\mathcal{A})$ is $O(\log_{1/\beta} n)$. Moreover, it can be proved that the expected number of steps performed by the walk at each level is constant ($O(1/\beta)$).

We still have to bound the time spend in each visited cells. Let A_i be the site of a visited cell in $\mathcal{V}(\mathcal{A}_\ell)$. Because the complexity of any cell in a Voronoi diagram is only bounded by $O(n_\ell)$ if n_ℓ is the number of sites, it is not efficient to compare the distances $\delta(x, A_i)$ and $\delta(x, A)$ for each neighbor A of A_i in $\mathcal{V}(\mathcal{A}_\ell)$. Therefore we attach an additional balanced binary tree to each cell of each Voronoi diagram in the hierarchy. The tree attached to the cell $V_\ell(A_i)$ of A_i in the diagram $\mathcal{V}(\mathcal{A}_\ell)$ includes, for each Voronoi vertex v of $V_\ell(A_i)$, the ray $\rho_i(p_v)$ where p_v is the point on ∂A_i closest to v, and $\rho_i(p_v)$ is defined as the ray starting from the center of the maximal disk $M_i(p_v)$ and passing through p_v. The rays are sorted according to the (counter-clockwise) order of the points p_v on ∂A_i. When $V_\ell(A_i)$ is visited, the ray $\rho_i(p_x)$ corresponding to the query point x is localized using the tree. Suppose that it is found to be between the rays of two vertices v_1 and v_2. Then it suffice to compare $\delta(x, A_i)$ and $\delta(x, A_j)$ where A_j is the neighbor of A_i in $\mathcal{V}(\mathcal{A}_\ell)$ sharing the vertices v_1 and v_2. Thus the time spend in each visited cell of $\mathcal{V}(\mathcal{A}_\ell)$ is $O(\log n_\ell) = O(\log n)$, which (together with with the expected number of visited nodes) yields the following lemma

Lemma 7 *Using a hierarchy of Voronoi diagrams with additional binary trees for each cell, a closest site query can be answered in time* $O(\frac{1}{\beta \log(1/\beta)} \log^2 n)$.

5 Complexity analysis

In this section we deal with the cost of the basic operations of our dynamic algorithm. We consider three scenarios. The first one assumes objects do not intersect. In the second scenario objects intersect but there are no hidden objects. The third scenario differs from the second one in that we allow the existence of hidden objects.

In each of the above three cases, we consider the expected cost of the basic operations, namely insertion and deletion. The expectation refers to the insertion order, that is, all possible insertion orders are considered to be equally likely and each deletion is considered to deal equally likely with any object in the current set. In all cases we assume that the Voronoi diagram hierarchy is used as the location data structure. Note that the hierarchy introduces another source of randomization. In the first two scenarios, i.e., when no hidden object exist, there is no covering graph to be maintained. Note the the randomized analysis obviously does not apply to the reinsertion of objects covered by a deleted object A_i, which explains why the randomization fails to improve the complexity of deletion in the presence of hidden objects.

Our results are summarized in the table below. The corresponding proofs are omitted due to lack of space; in any case they follow directly from a careful step by step analysis of the insertion and deletion procedures described above.

	Disjoint	No hidden	Hidden
Insertion	$O(\log^2 n)$	$O(n)$	$O(n)$
Deletion	$O(\log^3 n)$	$O(n)$	$O(n^2)$

6 Extensions

In this section we consider several extensions of the problem discussed in the preceding sections.

Degenerate configurations. Degenerate configurations occur when the set contains pairs of internally tangent objects. Let $\{A_i, A_j\}$ be an sc-pseudo-circles set with A_i and A_j internally tangent and $A_i \subseteq A_j$. The bisector π_{ij} is homeomorphic to a ray, if A_i and A_j have a single tangent point, or to two disconnected rays, if A_i and A_j have two tangent points. In any case, the interior $V^\circ(A_i)$ of the Voronoi region of A_i in $\mathcal{V}(\{A_i, A_j\})$ is empty and we consider the object A_i as hidden. This point of view is consistent with the definition we gave for hidden sites, which is that an object A is hidden if $N^\circ(A) = \emptyset$.

Let us discuss the algorithmic consequences of allowing degenerate configurations. When the object A is inserted in the diagram, the case where A is internally tangent to a visible object $A_i \in \mathcal{A}$ is detected at Step 1, during the

location the medial axis of A. The case of an object $A_j \in \mathcal{A}$ is internally tangent to A is detected during Step 2, when the entire conflict region is searched. In the first case A is hidden and its covering set is $\{A_i\}$. In the second case A_i becomes hidden and its covering set is $\{A\}$.

Pseudo-circles sets of piecewise smooth convex objects. In the sections above we assumed that all convex objects have smooth boundaries, i.e., their boundaries are at least C^1-continuous. In fact we can handle quite easily the case of objects whose boundaries are only piecewise C^1-continuous. Let us call *vertices* the points on the boundary of an object where there is no C^1-continuity. The main problem of piecewise C^1-continuous objects is that they can yield two-dimensional bisectors when two objects share the same vertex. The remedy is similar to the commonly used approach for the Voronoi diagram of segments (e.g., cf. [11]): we consider the vertices on the boundary of the objects as objects by themselves and slightly change the distance so that a point whose closest point on object A_i is a vertex of A_i is considered to be closer to that vertex. All two-dimensional bisectors then become the Voronoi cells of these vertices.

As far as our basic operations are concerned, we proceed as follows. Let A be the object to be inserted or deleted. We note A_v the set of vertices of A and $\hat{A}$ the object A minus the points in A_v. When we want to insert A in the current Voronoi diagram we at first insert all points in A_v and then $\hat{A}$. When we want to delete A we at first delete $\hat{A}$ and then all points in A_v. During the latter step we have to make sure that points in A_v are not vertices of other objects as well. This can be done easily by looking at the neighbors in the Voronoi diagram of each point in A_v.

Generic convex objects. In the case of smooth convex objects which do not form pseudo-circles sets we can compute the Voronoi diagram in the complement of their union (free space). The basic idea is that the Voronoi diagram in free space depends only on the arcs appearing on the boundary of the union of the objects.

More precisely, let $\mathcal{A}$ be a set of convex objects and let C be a connected component of the union of the objects in $\mathcal{A}$. Along the boundary ∂C of C, there exists a sequence of points $\{p_1, \ldots, p_m\}$, which are points of intersection of objects in $\mathcal{A}$. An arc α_i on ∂C joining p_i to p_{i+1} belongs to a single object $A \in \mathcal{A}$. We form the piecewise smooth convex object A_{α_i}, whose boundary is $\alpha_i \cup p_i p_{i+1}$, where $p_i p_{i+1}$ is the segment joining the points p_i and p_{i+1}. Consider the set $\mathcal{A}'$ consisting of all such objects A_{α_i}. $\mathcal{A}'$ is a pseudo-circles set (consisting of disjoint piecewise smooth convex objects) and the Voronoi diagrams $\mathcal{V}(\mathcal{A})$ and $\mathcal{V}(\mathcal{A}')$ coincide in free space.

The set $\mathcal{A}'$ can be computed by performing a line-sweep on the set $\mathcal{A}$ and keeping track of the boundary of the connected components of the union of the objects in $\mathcal{A}$. This can be done in time $O(n \log n + k)$, where $k = O(n^2)$ is the complexity of the boundary of the afore-mentioned union. Since the objects in $\mathcal{A}'$ are disjoint, we can then compute the Voronoi diagram in free space in total expected time $O(k \log^2 n)$.

7 Conclusion

We presented a dynamic algorithm for the construction of the euclidean Voronoi diagram in the plane for various classes of convex objects. In particular, we considered pseudo-circles sets of piecewise smooth convex objects, as well as generic smooth convex objects, in which case we can compute the Voronoi diagram in free space. Our algorithm uses fairly simple data structures and enables us to perform deletions easily.

We are currently working on extending the above results to non-convex objects, as well as understanding the relationship between the euclidean Voronoi diagram of such objects and abstract Voronoi diagrams. We conjecture that, given a pseudo-circles set in general position, such that any pair of objects has exactly two supporting lines, the corresponding set of bisectors is an admissible system of bisectors.

References

1. Aurenhammer, F., Klein, R.: Voronoi diagrams. In Sack, J.R., Urrutia, J., eds.: Handbook of Computational Geometry. Elsevier Science Publishers B.V. North-Holland, Amsterdam (2000) 201–290
2. Okabe, A., Boots, B., Sugihara, K., Chiu, S.N.: Spatial tessellations: concepts and applications of Voronoĭ diagrams. 2nd edn. John Wiley & Sons Ltd., Chichester (2000)
3. Koltun, V., Sharir, M.: Polyhedral Voronoi diagrams of polyhedra in three dimensions. In: Proc. 18th Annu. ACM Sympos. Comput. Geom. (2002) 227–236
4. Koltun, V., Sharir, M.: Three dimensional euclidean Voronoi diagrams of lines with a fixed number of orientations. In: Proc. 18th Annu. ACM Sympos. Comput. Geom. (2002) 217–226
5. Klein, R.: Concrete and Abstract Voronoi Diagrams. Volume 400 of Lecture Notes Comput. Sci. Springer-Verlag (1989)
6. Klein, R., Mehlhorn, K., Meiser, S.: Randomized incremental construction of abstract Voronoi diagrams. Comput. Geom.: Theory & Appl. **3** (1993) 157–184
7. Alt, H., Schwarzkopf, O.: The Voronoi diagram of curved objects. In: Proc. 11th Annu. ACM Sympos. Comput. Geom. (1995) 89–97
8. McAllister, M., Kirkpatrick, D., Snoeyink, J.: A compact piecewise-linear Voronoi diagram for convex sites i n the plane. Discrete Comput. Geom. **15** (1996) 73–105
9. Karavelas, M.I., Yvinec, M.: Dynamic additively weighted Voronoi diagrams in 2D. In: Proc. 10th Europ. Sympos. Alg. (2002) 586–598
10. Devillers, O.: The Delaunay hierarchy. Internat. J. Found. Comput. Sci. **13** (2002) 163–180
11. Burnikel, C.: Exact Computation of Voronoi Diagrams and Line Segment Intersections. Ph.D thesis, Universität des Saarlandes (1996)