Divide-and-Conquer for Voronoi Diagrams Revisited*

Oswin Aichholzer
IST, University of Technology
Graz, Austria
oaich@ist.tugraz.at

Wolfgang Aigner
IST, University of Technology
Graz, Austria
waigner@ist.tugraz.at

Franz Aurenhammer
IGI, University of Technology
Graz, Austria
auren@igi.tugraz.at

Thomas Hackl
IST, University of Technology
Graz, Austria
thackl@ist.tugraz.at

Bert Jüttler
AG, Johannes Kepler
University Linz, Austria
Bert.Juettler@jku.at

Elisabeth Pilgerstorfer
AG, Johannes Kepler
University Linz, Austria
Elisabeth.Pilgerstorfer@jku.at

Margot Rabl
AG, Johannes Kepler
University Linz, Austria
Margot.Rabl@jku.at

ABSTRACT

We show how to divide the edge graph of a Voronoi diagram into a tree that corresponds to the medial axis of an (augmented) planar domain. Division into base cases is then possible, which, in the bottom-up phase, can be merged by trivial concatenation. The resulting construction algorithm—similar to Delaunay triangulation methods—is not bisector-based and merely computes dual links between the sites, its atomic steps being inclusion tests for sites in circles. This guarantees computational simplicity and numerical stability. Moreover, no part of the Voronoi diagram, once constructed, has to be discarded again. The algorithm works for polygonal and curved objects as sites and, in particular, for circular arcs which allows its extension to general free-form objects by Voronoi diagram preserving and data saving biarc approximations. The algorithm is randomized, with expected runtime $O(n \log n)$ under certain assumptions on the input data. Experiments substantiate an efficient behavior even when these assumptions are not met. Applications to offset computations and motion planning for general objects are described.

Categories and Subject Descriptors

F.2.2 [**Theory of Computation**]: Nonnumerical Algorithms and Problems—*geometrical problems and computations*

General Terms

Algorithms, Theory

*Supported by the Austrian FWF Joint Research Project 'Industrial Geometry', S9205-N12.

SCG'09, June 8–10, 2009, Aarhus, Denmark.

Keywords

Voronoi diagram, medial axis, divide-and-conquer, biarc approximation, trimmed offset, motion planning

1. INTRODUCTION

The divide-and-conquer paradigm gave the first optimal solution for constructing the closest-site Voronoi diagram in the plane [27]. Though being a classical example for applying a powerful algorithmic method in computational geometry, the resulting algorithm became no favorite for implementation, not even in the case of point sites.

For Voronoi diagrams of general objects the situation is more intricate, as such diagrams may have all kinds of artifacts. Their edge graph may be disconnected, and their bisectors may be closed curves, which complicates the construction. In particular, the abstract Voronoi diagram machinery in [18, 19] is ruled out. Literature tells us that divide-and-conquer is involved if emphasis is on the bottom-up phase, even if the sites are of relatively simple shape. See the papers [21] and [28], respectively, for early algorithms on line segments and circles, and the optimal $O(n \log n)$ variants in [17] for line segments, in [30] for line segments and circular arcs, and in [7] for convex distance functions. The crux is the missing separability condition for the sites, which would prevent the merge curve from breaking into several components. Even this issue being solved, we still have to intersect complicated bisectors and discard old parts of the diagram, which makes the algorithms complex and hard to implement.

Many alternative strategies for computing generalized Voronoi diagrams have been tried. Incremental insertion cannot be applied directly to general sites without loss of efficiency. In particular, the framework in [19] for abstract Voronoi diagrams may not apply. Still, randomized insertion can be made efficient [3], but needs pre-requisites like splitting sites into 'harmless' pieces, each piece then acting as several sites. The plane-sweep technique, on the other hand, generalizes nicely for line segments and circles [12] but, unfortunately, not for circular arcs or more general sites. Line segments have to be split into 3 sites to 'domesticate' their bisector. Many implementation details occur.

In fact, in all these algorithms the bisector curves take part in the computation. Already in the case of line segments, bisectors are

composed of up to 7 pieces, and may even be two-dimensional if not defined carefully in the case of shared endpoints. Such situations cannot be considered degenerate; they occur naturally when decomposing complex sites into simpler ones. Consequently, the algorithms are involved and also suffer from numerical imprecison. Difficulties may be partially eluded when working in the dual environment: Instead of intersecting two bisectors, the center of a circle tangent to the three defining sites is calculated. This bears the advantage of working on the sites directly, linking them accordingly rather than computing new geometric objects that themselves take part in later calculations. The classical example is, of course, the Delaunay triangulation for point sites. For general sites, tangent circles may not be unique. Up to 8 solutions do exist, which are usually difficult to calculate; see e.g. [11].

The algorithm we propose in the present paper works directly on the sites, too, but its atomic operation is much simpler, namely, an inclusion test of a site in a *fixed* circle. We first extract the combinatorial structure of the Voronoi diagram, and fill in the bisector curves later on. In contrast to existing Voronoi/Delaunay algorithms, no constructed object is ever discarded. Our setting is very general: Sites are pairwise disjoint topological disks of dimensions 2, 1, or 0. This includes polygonal sites, circular disks, spline curves, but also single points and straight-line segments. Boundaries of curved planar objects with holes can be modeled. We do not split complex sites into pieces beforehand, because we need not care about the bisectors.

Our idea is to calibrate the top-down phase of divide-and-conquer by dividing the edge graph of the Voronoi diagram without prior knowledge. A simple plane sweep is used to generate a set of points whose removal from the edge graph leaves a geometric tree. This tree is then computable as the medial axis of a generalized domain that, combinatorially, behaves like a simply connected domain. While classical medial axis algorithms [20, 6, 8] cannot be applied, not even in the presence of simple sites, we show that the methods in [2, 1] are flexible enough to be extended to work for such domains. In particular, the edge graph is split further in a recursive manner, until directly solvable base cases remain. The bottom-up phase is trivial and consists of reassembling the respective pieces of the edge graph.

The paper includes a theoretical and an applied part. We take particular interest in sites represented by circular arc splines, for several reasons. The modeling power of such splines beats that of polylines, which results in a significantly smaller input data volume. Our algorithm naturally, and with almost no increase of numeric complexity, works for this case. Also, a stable approximation of the Voronoi diagram for algebraically complex original sites can be guaranteed. If the number of sites is small compared to the number, n, of their describing arcs, the graph diameter of the medial axis mentioned above tends to be linear, and our algorithm runs in $O(n \log n)$ randomized time. Experiments substantiate this behavior with small constants, but also show that, in the case of point sites, the runtime is slightly larger. Thus, the simplicity and generality of our algorithm come at a price. Still, this is maybe the first practical algorithm that works reasonably efficient for general planar sites. Existing practical methods, e.g. in [14, 10], are confined to polygonal inputs; curved objects, if accepted, are converted to polygonal ones, blowing up the data volume in a non-linear manner. [1]

[1]We recently learned that the VRONI Voronoi code [14] for points and line segments has been extended to include circular arcs as sites [15]. The underlying algorithm is incremental insertion. A circular arc's endpoints have to be inserted prior to their defining object.

Applications are manifold. The two ones we sketch here use sites in piecewise circular (PC) representation. This enables motion planning in PC-environments [31] which, compared to piecewise linear (PL)-environments, is shown to lead to shorter and 'smoother' robot paths. Moreover, shape offset computations are eased by the fact that PC representations are closed under offset operations. Compared to other offsetting algorithms that are based on Voronoi diagrams [16, 13, 4], our method is simpler because we compute only a combinatorial representation of the diagram for this application.

2. DIVIDING THE VORONOI DIAGRAM

Let us define the Voronoi diagram of general objects. Our sites are pairwise disjoint and closed[2] topological disks of dimension two, one, or zero in the Euclidean plane $\mathbb{R}^2$. That is, a site is either homeomorphic to a disk or to a line segment, or is simply a point. This includes polygons, circular disks, and open spline curves as sites. Here and throughout this paper, let S denote the given set of sites. The distance of a point x to a site $s \in S$ is $d(x, s) = \min_{y \in s} \delta(x, y)$, where δ denotes the Euclidean distance function. As done e.g. in [3, 30], we define the Voronoi diagram, $V(S)$, of S via its *edge graph*, $\mathcal{G}_S$, which is the set of all points having more than one closest point on the union of all sites. Under the assumption that sites are represented in a reasonable way (say, by real analytic curve pieces), this geometric graph is well defined by results in [9]. An edge of $\mathcal{G}_S$ containing points equidistant from two or more different points on the same site s is called a *self-edge* for s. The *regions* of $V(S)$ are the maximal connected subsets of the complement of $\mathcal{G}_S$ in $\mathbb{R}^2$. They are topologically open sets.

OBSERVATION 1. *The regions of $V(S)$ bijectively correspond to the sites in S. Each site is contained in its region, and regions are simply connected.*

PROOF. Let x be a point in the region R of $V(S)$. To x there exists a unique closest point, y, on the union of the sites in S. (Otherwise, x would be a point on the edge graph $\mathcal{G}_S$.) The sites are pairwise disjoint, so there is a unique site $s \in S$ with $y \in s$. Site s is the same for all $x \in R$, because $d(x, s)$ is a continuous function of x. This maps regions to sites.

Now, obviously, with x also the closed line segment $\overline{xy}$ is part of R. This implies that R is simply connected. In particular, we have $y \in R$, which implies $s \subset R$ and maps sites to regions. □

We thus can talk of the region *of a site*, s, which we will denote with $R(s)$ in the sequel.

The differences to a bisector-based definition of the Voronoi diagram should be noticed. Self-edges are ignored in such a definition unless the sites are split into suitable pieces. Such pieces, however, share boundaries—a fact that, if not treated with care, may give rise to unpleasant phenomena like two-dimensional bisectors.

To get control over the unbounded components of the diagram, we include a surrounding circle, Γ, (or any other desired curve) into the set S of sites. We can always choose Γ in a way such that each vertex of $V(S \setminus \{\Gamma\})$ is also a vertex of $V(S)$. All regions of $V(S)$ are bounded now, except, of course, the region $R(\Gamma)$.

For later purposes, we intend to show that removal of certain points on the edge graph $\mathcal{G}_S$ breaks all its cycles. Finding such

[2]Topological properties are meant to be relative to the dimension of the considered object.

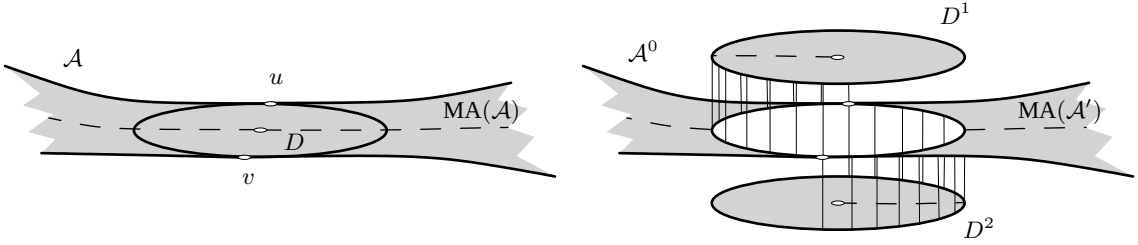


Figure 1: Domain $\mathcal{A}'$ (right) obtained as the augmentation of a given domain (left) with a splitting disk D. The medial axis (dashed) is split at the center of D.

points is nontrivial, in view of the possible presence of self-edges. For a site $s \neq \Gamma$, let $p(s)$ be a point on s with smallest ordinate, and denote with $q(s)$ the closest point on $\mathcal{G}_S$ vertically below $p(s)$. By the boundedness of $R(s)$, the point $q(s)$ always exists. Without loss of generality, let us assume that $q(s)$ is not an endpoint of any edge of $\mathcal{G}_S$; this can always be achieved by rotating the coordinate system slightly. We define a new geometric graph as

$$\mathcal{T}_S = \mathcal{G}_S \setminus \{q(s) \mid s \in S \setminus \{\Gamma\}\}. \tag{1}$$

LEMMA 1. *The graph $\mathcal{T}_S$ is a tree.*

PROOF. For each bounded region of $V(S)$, the edge graph $\mathcal{G}_S$ contains a unique elementary cycle, because of the simple connectivity of regions (Observation 1). For the same reason, the set of cycles does not change if self-edges are ignored. Interrupting each elementary cycle at a point vertically below its site leaves a geometric forest, because no path can continue below any site. Moreover, when as many points are removed as there are elementary cycles, and removal takes place at the interiors of edges of $\mathcal{G}_S$, a geometric tree is obtained.

It remains to show that, for each site $s \neq \Gamma$, the point $q(s) \in \mathcal{G}_S$ does not lie on a self-edge for s. Recall that $q(s)$ is equidistant from $p(s)$ and from at least one other point, say y, on the union of all the sites. The ordinate of y is smaller than the ordinate of $p(s)$, because $p(s)$ lies vertically above $q(s)$. Thus, assuming that such a point y stems from s, which has to be the case if $q(s)$ lies on a self-edge for s, contradicts the definition of $p(s)$. □

3. AUGMENTED DOMAINS

Our next aim is to interpret the tree $\mathcal{T}_S$ in Lemma 1 as the medial axis of a generalized planar domain. In this way, we will be able to construct the Voronoi diagram $V(S)$ by means of a medial axis algorithm, as if a *simply connected* domain was the input. Usually, the similarity between these two structures is exploited the other way round: Medial axes are constructed as special cases of Voronoi diagrams.

Consider a bounded and connected two-manifold $\mathcal{B}$, here just called a *shape*, in $\mathbb{R}^2$. An inscribed disk for $\mathcal{B}$ is defined as a disk which lies entirely in $\mathcal{B}$. The set of inscribed disks is partially ordered with respect to inclusion. The *medial axis transform* of $\mathcal{B}$, for short MAT($\mathcal{B}$), is the set of all maximal inscribed disks. Similarly, the *medial axis*, MA($\mathcal{B}$), of $\mathcal{B}$ is the set of all centers of the disks in MAT($\mathcal{B}$). It is easy to interpret $V(S)$ as the medial axis of a planar shape. Simply take the surrounding circle Γ as part of the shape boundary, and consider each remaining site $s \in S$ as a (possibly degenerate) hole. That is, we define

$$\mathcal{B} = B_0 \setminus \{s \in S \mid s \neq \Gamma\}, \tag{2}$$

where B_0 denotes the disk bounded by Γ. The medial axis MA($\mathcal{B}$) is just the closure[3] of the edge graph $\mathcal{G}_S$ of $V(S)$.

Our goal is, however, a different one. We want to combinatorially disconnect the shape $\mathcal{B}$ at appropriate positions, such that the medial axis of the resulting domain corresponds to the tree decomposition $\mathcal{T}_S$ of $V(S)$. As observed in [9], a maximal inscribed disk can be used to split the medial axis of a simply connected shape into two components which share a point at the disk's center. In order to extend this result to general shapes, we introduce the notion of an *augmented domain*. Its definition is recursive, as follows.

An augmented domain is a set $\mathcal{A}$ together with a projection $\pi_{\mathcal{A}} : \mathcal{A} \to \mathbb{R}^2$. Initially, $\mathcal{A}$ is the original shape $\mathcal{B}$, and the associated projection $\pi_{\mathcal{B}}$ is the identity.

Now, consider a maximal inscribed disk D of an augmented domain $\mathcal{A}$, which touches the boundary $\partial\mathcal{A}$ in exactly two points u and v. Denote with $\overset{\frown}{uv}$ and $\overset{\frown}{vu}$ the two circular arcs which the boundary of D is split into. The new augmented shape, $\mathcal{A}'$, which is obtained from $\mathcal{A}$ by splitting it with D, is defined as

$$\mathcal{A}' = \mathcal{A}^0 \cup D^1 \cup D^2$$

where $\mathcal{A}^0 = \{(x,0) \mid x \in A \setminus D\}$, $D^1 = \{(x,1) \mid x \in D\}$, and $D^2 = \{(x,2) \mid x \in D\}$. See Figure 1 for an illustration. The associated projection is

$$\pi_{\mathcal{A}'} : \mathcal{A}' \to \mathbb{R}^2,\ (x,i) \mapsto \pi_A(x).$$

We say that the line segment in $\mathcal{A}$ between points (x,i) and (y,j) is *contained* in $\mathcal{A}'$ if one of the following conditions is satisfied:

1. $i = j$ and the line segment $\overline{xy}$ avoids ∂D,
2. $\{i,j\} = \{0,1\}$ and $\overline{xy}$ intersects the arc $\overset{\frown}{uv}$, or
3. $\{i,j\} = \{0,2\}$ and $\overline{xy}$ intersects the arc $\overset{\frown}{vu}$.

For any two points (x,i) and (y,j) in $\mathcal{A}'$, their distance now can be defined. It equals the distance of $\pi_{\mathcal{A}}(x)$ and $\pi_{\mathcal{A}}(y)$ in $\mathbb{R}^2$, provided the connecting line segment is contained in $\mathcal{A}'$, and is ∞, otherwise. An (open) disk in $\mathcal{A}'$ with center (m,i) and radius ϱ is the set of all points in $\mathcal{A}'$ whose distance to (m,i) is less than ϱ. Such a disk is said to be *inscribed* in $\mathcal{A}'$ if its projection into $\mathbb{R}^2$ is again an open disk.

Having specified inscribed disks for $\mathcal{A}'$, the boundary of $\mathcal{A}'$ and the medial axis (transform) of $\mathcal{A}'$ can be defined as in the case of

[3]The reason why these two structures are not identical lies in the possible existence of osculating maximal inscribed disks for $\mathcal{B}$. The centers of such disks, while belonging to MA($\mathcal{B}$), are not part of $\mathcal{G}_S$. This subtle difference may be ignored for the purposes of the present paper.

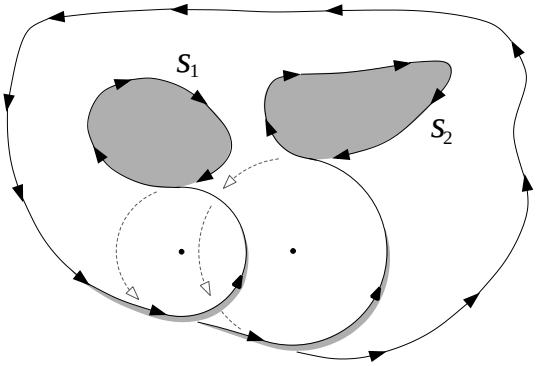


Figure 2: Oriented boundary of an augmented domain.

planar shapes. In particular, $\partial\mathcal{A}'$ derives from $\partial\mathcal{A}$ by disconnecting the latter boundary at the contact points u and v of the splitting disk D, and reconnecting it with the circular arcs $\overset{\frown}{uv}$ and $\overset{\frown}{vu}$. This process is depicted schematically in Figure 2. Note that when $\partial\mathcal{A}'$ is traversed in a fixed orientation, the interior of $\mathcal{A}'$ stays on a fixed side.

Concerning the medial axis, every maximal inscribed disk in $\mathcal{A}$ different from D corresponds to exactly one maximal inscribed disk in $\mathcal{A}'$, hence there is a bijection between $\mathrm{MAT}(\mathcal{A}) \setminus \{D\}$ and $\mathrm{MAT}(\mathcal{A}') \setminus \{D^1, D^2\}$. The medial axis of $\mathcal{A}'$ therefore is the same geometric graph as $\mathrm{MA}(\mathcal{A})$, except that the edge of $\mathrm{MA}(\mathcal{A})$ containing the center of D is split into two disconnected edges which both have the center of D as one of their endpoints. These two points are two leaves of $\mathrm{MA}(\mathcal{A}')$; consult Figure 1 again.

To draw the connection to the edge graph $\mathcal{G}_S$ of $V(S)$, the initial shape $\mathcal{B}$ in (2) is augmented with $|S| - 1$ maximal inscribed disks, namely, the ones centered at the points $q(s) \in \mathcal{G}_S$, where $q(s)$ was the vertical projection onto $\mathcal{G}_S$ of a point with smallest ordinate on the site s. Denote with $\mathcal{A}_S$ the resulting domain after these $|S| - 1$ augmentation steps. We may conclude the main finding of this section as follows.

LEMMA 2. *The tree $\mathcal{T}_S$ in (1) is the medial axis of the augmented domain $\mathcal{A}_S$.*

4. THE ALGORITHM

Using Lemma 2, the Voronoi diagram $V(S)$ can be obtained by computing the medial axis of the augmented domain $\mathcal{A}_S$. We show how to compute $\mathcal{A}_S$ efficiently, and how to construct its medial axis without the need of computing distances between points in $\mathcal{A}_S$ directly. The resulting algorithm is very simple and lends itself to robust implementation. It runs in optimal (randomized) time $O(n \log n)$ if certain quite realistic assumptions on the input are met, and in $O(n\sqrt{n})$ time in the unrestricted case. Its observed runtime, however, is close to the former with rather small factors.

4.1 Computing the boundary of $\mathcal{A}_S$

Consider the planar shape $\mathcal{B}$ in (2) whose augmentation has led to the domain $\mathcal{A}_S$. From the algorithmic point of view, augmenting $\mathcal{B}$ amounts to connecting its boundary $\partial\mathcal{B}$ to a *single* cyclic sequence, $\partial\mathcal{A}_S$, that consists of pieces from $\partial\mathcal{B}$ and from circles bounding the splitting disks. (One-dimensional sites contribute to $\partial\mathcal{B}$ with two curves, one for either orientation, and the special

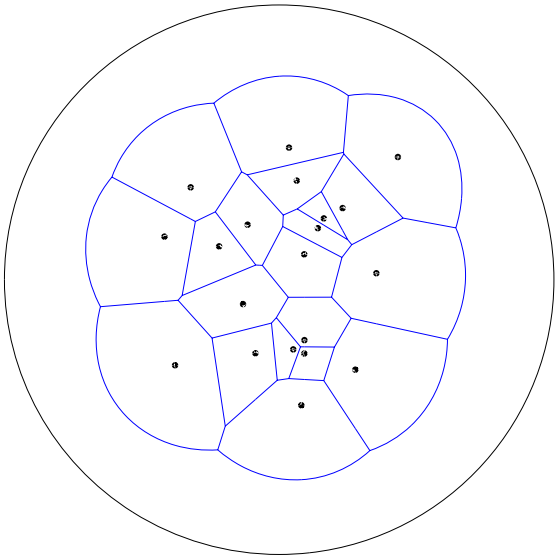


Figure 3: Voronoi diagram for point sites.

case of point sites can be handled consistently.) Each such boundary piece is used exactly once on $\partial\mathcal{A}_S$, and traversing $\partial\mathcal{A}_S$ corresponds to tracing the medial axis tree $\mathrm{MA}(\mathcal{A}_S)$ in preorder. See Figure 2, where a shape having two planar sites s_1 and s_2 as its holes is augmented with two disks, and the boundary of the resulting augmented domain is oriented for better visualization.

The construction of $\partial\mathcal{A}_S$ is trivial once the splitting disks are available. [4] The main task is, therefore, to find these disks D_i, one for each site $s_i \in S \setminus \{\Gamma\}$. Recall from Section 2 that D_i is horizontally tangent to s_i at a lowest point $p(s_i)$ of s_i. The center $q(s_i)$ of D_i lies on the edge graph $\mathcal{G}_S$ of $V(S)$ but, of course, D_i need to be found without knowledge of $\mathcal{G}_S$.

Indeed, a simple and efficient plane-sweep can be applied as follows. Sweep across S from above to below with a horizontal line L. For a site $s_i \neq \Gamma$, let x_i be the abscissa of $p(s_i)$, and define $E_L(i) = s_i \cap L$. Note that $E_L(i)$ may consist of more than one component. We maintain, for each site s_i whose point $p(s_i)$ has been swept over, the site s_j where $E_L(j)$ is closest to x_i on L. The unique disk with north pole $p(s_i)$ and touching s_j is computed, and the minimal such disk for s_i so far, $D_L(i)$, is updated if necessary. The abscissa x_i is deactivated again when $D_L(i)$ has been fully swept over by L.

LEMMA 3. *After completion of the sweep, $D_L(i) = D_i$ holds for each index i.*

PROOF. For a fixed index i, let s_k be the site that defines the disk D_i. We have to show that $E_L(k)$ and x_i become neighbors on L while x_i is active. Consider a point t where D_i is tangent to s_k. Then, because D_i avoids all the sites, the line segment $\overline{x_i t} \subset D_i$ does the same. Thus $E_L(k)$ and x_i are adjacent when L passes through t. Also, x_i is active at this moment, because $D_i \subset D_L(i)$ holds. □

[4] As a possible degenerate case, a splitting disk may have more than two points of contact with the boundary $\partial\mathcal{B}$. In that case, we may choose any two contact points on different components of $\partial\mathcal{B}$. The algorithm we are going to describe automatically yields such a pair of points for each disk.

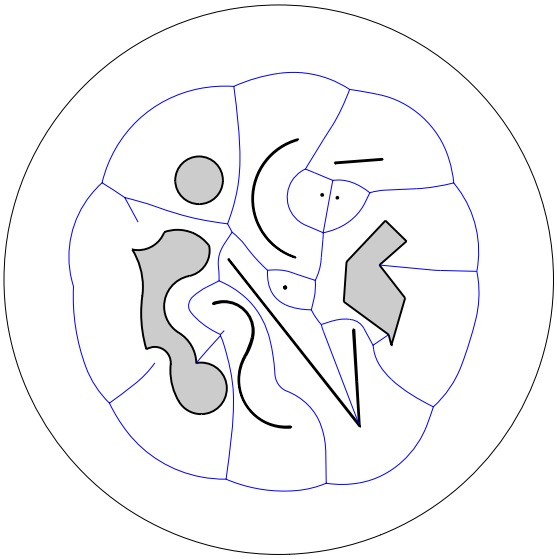


Figure 4: A mixed set of sites

To keep small the number of neighbor pairs (x_i, s_j) on L processed during the plane sweep, we only consider pairs where no other active abscissa x_m lies between x_i and $E_L(j)$; the disk $D_L(i)$ cannot have a contact beyond the one of $D_L(m)$, otherwise. The number of such pairs is linear. Thus the construction can be implemented in $O(n \log n)$ time if the sites in S are described by a total of n objects, each being managable in constant time. Note that $\partial\mathcal{A}_S$ then consist of $\Theta(n)$ pieces.

4.2 Computing the medial axis of $\mathcal{A}_S$

Given the description of an augmented domain by its boundary, it may, at first glance, seem complicated to compute its medial axis. In our case, however, the domain $\mathcal{A}_S$ has a connected boundary. Therefore it can be split into subdomains with the same property using maximal inscribed disks. This suggests a divide-and-conquer algorithm for computing $\mathrm{MA}(\mathcal{A}_S)$. The domain and its medial axis tree are split recursively, until directly solvable base cases remain. For simply connected shapes, a similar approach has been applied in [2, 1].

In fact, it is easy to obtain splitting disks for $\mathcal{A}_S$. Recall that $\partial\mathcal{A}_S$ consists of pieces that bound inscribed disks (called *artificial arcs*) and pieces that stem from site boundaries (called *site segments*). Now, to calculate a splitting disk, the algorithm fixes some point p on a site segment and computes a maximal inscribed disk D for $\mathcal{A}_S$ that touches $\partial\mathcal{A}_S$ at p. Starting with an (appropriately oriented) disk of large radius, $\partial\mathcal{A}_S$ is scanned and the disk is shrunk accordingly whenever an intersection with a site segment occurs. Intersections with artificial arcs are, however, ignored.

LEMMA 4. *The algorithm above correctly computes the required disk D for $\mathcal{A}_S$ at point p.*

PROOF. From Section 3 we know that the set of maximal inscribed disks is the same for $\mathcal{A}_S$ and for $\mathcal{B}$, except for the (finitely many) disks taking part in the augmentation. The assertion follows. □

In other words, the distances to the sites which are needed in the medial axis computation are the same in $\mathcal{A}_S$ and in $\mathcal{B}$. (This is, of course, not true for *all* possible distances.) Note that the artificial arcs are used only to link the site segments in the correct cyclic order; they do not play any geometric role. Computing a splitting disk takes $O(n)$ time, if each object describing the sites can be handled in $O(1)$ time.

n	atomic steps	ratio $n \log_2 n$	ratio $n(\log_2 n)^2$
507	6620	1.45	0.16
2070	32892	1.44	0.13
5196	91649	1.43	0.12
10474	199001	1.42	0.11
20488	417839	1.42	0.10
172198	4223178	1.41	0.09

Table 1: Five complex sites bounded by n arcs

4.3 Practical aspects

In view of keeping the algorithm efficient, disks that split the domain $\mathcal{A}_S$ in a balanced way are desired. Unfortunately, computing such a disk with simple means turns out to be hard. We can, however, choose a disk D randomly, by taking a random site segment on $\partial\mathcal{A}_S$ as its basis. Objects on $\partial\mathcal{A}_S$ and edges of $\mathrm{MA}(\mathcal{A}_S)$ correspond to each other in an (almost) bijective way, which suffices to convey randomness from boundary objects to medial axis edges. For the analysis, we thus may suppose that the center c of D lies on every edge of $\mathrm{MA}(\mathcal{A}_S)$ with the same probability. Under the assumption that the graph diameter of $\mathrm{MA}(\mathcal{A}_S)$ is linear in n, the point c lies on the diameter with constant probability, and $\mathrm{MA}(\mathcal{A}_S)$ is split at c into two parts of expected size $\Theta(n)$. A randomized runtime of $O(n \log n)$ results.

The assumption above is realistic in scenarios where a small number of sites is represented by a large number of individual objects. The required accuracy for approximating the sites then typically leads to an input size that is independent from the branching of $\mathrm{MA}(\mathcal{A}_S)$. In particular, if biarcs are used for approximation (see Section 5) then the number of leaves (hence also the number of vertices) of $\mathrm{MA}(\mathcal{A}_S)$ is determined by the original sites and not by the number of biarcs used. Our tests report small constants in the $O(n \log n)$ term in this case. See Table 1, where step counts are averaged (and rounded) over 40 different equal-sized inputs.

n	atomic steps	ratio $n \log_2 n$	ratio $n(\log_2 n)^2$
400	7591	2.20	0.25
2000	54662	2.49	0.23
4000	143391	3.00	0.25
20000	1015149	3.55	0.25
40000	2659149	4.35	0.28
200000	19820012	5.63	0.32

Table 2: Uniform distribution of n point sites

The other extreme is the case of n point sites. Here, by the way how $\mathcal{A}_S$ is constructed, the diameter of $\mathrm{MA}(\mathcal{A}_S)$ will be typically much smaller, because many long 'vertical' branches will emanate from the surrounding circle Γ. As a simple heuristic, we may choose a small number of splitting disks tangent to Γ first, and continue with randomly splitting the resulting augmented subdomains. This (almost) yields an observed $O(n \log^2 n)$ behavior, with very

small factors; see Table 2. We took uniformly distributed point sites — an input likely to avoid long paths in $\mathrm{MA}(\mathcal{A}_S)$ and thus slowing down the algorithm. Note that, for point sites, $\mathrm{MA}(\mathcal{A}_S)$ is basically the (piecewise-linear) medial axis of a union of disks, the augmenting disks plus the splitting disks.

Domain splitting could be combined with local tracing, as done in [2], to guarantee an $O(n\sqrt{n})$ expected time. However, the simple randomized version performed best in all our tests. We implemented the algorithm to accept circular arc input in its current version, including (though not optimizing) the handling of line segments and points. The Voronoi diagrams in Figures 3 and 4, and also the structures in Figures 9 and 8 in Subsection 6.2 have been produced by this code.

An excerpt of our experiments for point sites and circular arcs is given in Table 1 and Table 2. For input size n, the number of atomic steps is listed along with its ratio to the functions $n \log_2 n$ and $n(\log_2 n)^2$.

The atomic step needed in Subsections 4.1 and 4.2 is an intersection test of a site-describing object and a given disk. This is among the simplest imaginable tests when a closest-site Voronoi diagram is to be computed by means of distance calculations. Neither circles touching three given sites, nor intersection points of two bisectors, have to be calculated, apart from (but only if desired in) the base cases delivered by divide-and-conquer. This reduces the numerical effort and liability to errors caused by such operations, which themselves get rapidly complicated with the algebraic complexity of the sites; see e.g. [11]. We used CGAL [5] to implement the atomic operation for sites described by circular arcs (the intersection of two given circles).

Table 3 displays the CPU time in seconds we measured for line segment sites as input to our algorithm (column NEW), in comparison to the relevant CGAL demo packages for polygons (column POLY) and line segments (column SEG). Our algorithm's runtimes have to be interpreted with care, however. In its unoptimized implementation, our algorithm treats each line segment as six individual pieces (which describe a topological disk after possible splitting due to initial domain augmentation), whereas only two pieces would be actually needed (the split line segment). That is, we can expect a speedup by a factor of three from a more tailored implementation.

n	NEW	POLY	SEG
100	0.14	0.26	0.29
500	0.85	1.50	1.62
1000	2.22	3.11	3.44
5000	13.75	18.54	19.85
10000	39.26	37.63	42.50
50000	395.26	201.6	221.85

Table 3: Comparison to CGAL for n line segments

The structure and variety of the base cases depend on the sites. For point sites, there are only two of them, if the surrounding circle Γ is handled symbolically. They are of the simple form shown in Figure 5. (Artificial arcs are drawn dashed.) For circular arc splines, we get four generic base cases for C^1 continuity and nine for C^0 continuity; see [1]. These numbers do not increase for polynomial splines of higher order. Solving a base case includes calculating the equations describing the bisectors curves.

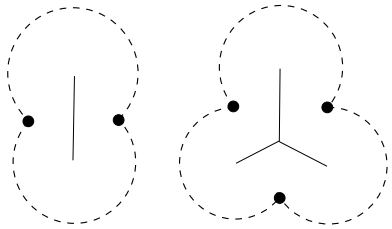


Figure 5: The two base cases for point sites.

Note that the algorithm allows us to separate geometric from combinatorial issues. If one is interested only in the topological structure of $V(S)$, then the base cases need not be resolved at all, because the type of a base case already determines the degree of the involved Voronoi vertices.

5. SITE APPROXIMATION

We put particular emphasis on circular arcs as sites, because no practical algorithm for constructing their Voronoi diagram is available, and our algorithm naturally offers the ability to handle them. Moreover, so-called biarcs enable a data-inexpensive and Voronoi diagram preserving approximation of general polynomial spline curves, as is described briefly below.

A *biarc* is the concatenation of two arcs which meet with a common tangent at a joint J. It connects two given endpoints p_0, p_1 with associated tangent vectors t_0, t_1, possibly sampled from a given curve. There exists a one-parameter family of biarcs matching these data, and the locus of all possible joints J is a circle.

Several different choices for the joint of a biarc are meaningful; see e.g. [23, 29]. The equal chord (EC) biarc generates arcs of equal length, whereas the parallel tangent (PT) biarc makes the tangent at the joint parallel to the line p_0p_1. The intersection (IS) biarc determines J by intersecting the joint circle with the given curve. The spiral (SP) biarc chooses one of the arcs as a segment of an osculating circle of the given curve.

For data sampled from a smooth boundary segment of a site, the Hausdorff distance between the biarc and the segment decreases with the biarc length h. Table 4 provides the Taylor expansions of the errors, where κ_i is the i-th derivative of the curve's curvature with respect to the arc length parameter at the point of interest; see [25] for more details.

Comparing the different methods for biarc interpolation, a first observation is that the error in all four methods is $\Theta(h^3)$ (for non-circular input). Consequently, when the tolerated maximum error ε is decreased, the number n of arcs grows moderately, $\Theta(\sqrt[3]{1/\varepsilon})$. This is much less than the number of line segments needed to get

Type	Maximal distance error (up to $O(h^5)$)
EC	$\max\left(\left\lvert\frac{\kappa_1}{324}h^3 - \frac{\kappa_2}{1944}h^4\right\rvert, \left\lvert -\frac{\kappa_1}{324}h^3 - \frac{7\kappa_2}{1944}h^4\right\rvert\right)$
PT	$\max\left(\left\lvert \pm \frac{\kappa_1}{324}h^3 + \frac{6\kappa_1^2 - \kappa_0\kappa_2}{1944\kappa_0}h^4\right\rvert\right)$
IS	$\max\left(\left\lvert\frac{\kappa_1}{324}h^3 + \frac{7\kappa_2}{3888}h^4\right\rvert, \left\lvert -\frac{\kappa_1}{324}h^3 - \frac{5\kappa_2}{3888}h^4\right\rvert\right)$
SP	$\left\lvert -\frac{\kappa_1}{96}h^3 - \frac{\kappa_2}{192}h^4\right\rvert$

Table 4: Approximation quality of biarcs

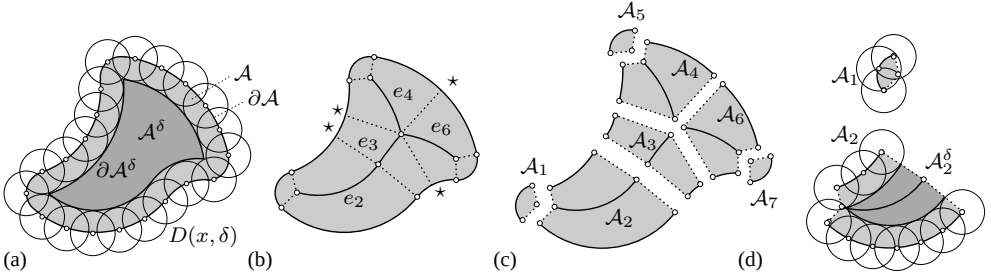


Figure 6: (a) Definition of the trimmed offset, (b) segmentation of the edge graph (additional arc endpoints are marked with $\star$), (c) segmentation of the shape, (d) offsets of subshapes.

the same accuracy, which is $\Theta(\sqrt{1/\varepsilon})$. The number of needed sample points even is $\Theta(1/\varepsilon)$. For high accuracies, the use of circular arcs for site approximation thus leads to a significantly smaller data volume.

Note that the constants at h^3 are identical for EC, PT, and IS. An analysis of the h^4 terms (see Figure 7) reveals that the IS method performs better than EC or PT in most situations, except for the case $\frac{4\kappa_1^2}{3\kappa_0} \le \kappa_2 \le \frac{12\kappa_1^2}{7\kappa_0}$ where PT is better. In the case of SP, the constant of the h^3 term is $(\frac{3}{2})^3$ times larger. Consequently, when approximating a site with spiral biarcs, the number of segments needed to achieve the same accuracy is roughly $\frac{3}{2}$ times larger. The experimental data listed in Tables 5 and 6 in Subsection 6.2 reflect this fact.

On the other hand, the approximation of sites by spiral biarcs guarantees convergence of the Voronoi diagram. More precisely, the error of the Voronoi diagram is $\Theta(n^{-3})$, where n is the number of biarcs. This can be proved by extending the arguments in [2] for the convergence of the medial axis and using the observation that the leaves of the edge graph correspond to self-edges of the sites.

According to our experience, in most cases the first three types of biarcs preserve the curvature distribution too. This is also supported by theoretical results [24]. So all biarc schemes are well suited for fast approximate Voronoi diagram computation. Biarc approximations of polynomial spline curves can be found in $O(n)$ time, by simple bisection or iteration algorithms.

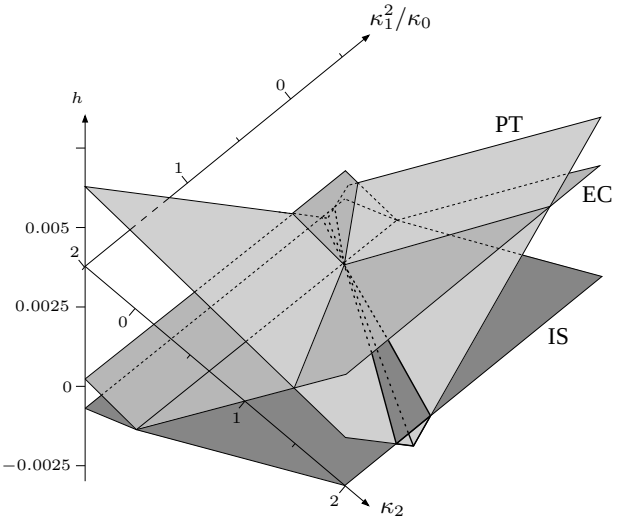


Figure 7: Coefficients of h^4 in the maximal errors for the biarc types EC, PT and IS. The lower envelope gives the smallest error.

6. APPLICATIONS

To document the practicality of the Voronoi diagram algorithm, let us briefly describe two of its appications.

6.1 Robot motion planning

Motion planning is among the classical applications of generalized Voronoi diagrams [30, 3, 22]. It is based on the observation that moving on the edge graph keeps the robot locally away from the sites (obstacles) as far as possible.

We may use the Voronoi diagram $V(S)$ of a set S of circularly approximated sites as a tool for planning a robot motion in a piecewise-circular environment [31]. Compared to PL-environments, this offers several advantages. The edges of $\mathcal{G}_S$ are still of degree only two—all types of conics can occur now—but a more data-saving approximation of the real scene is guaranteed by the results in Section 5. Not only can $V(S)$ be computed more quickly now, but $\mathcal{G}_S$ also will consist of significantly fewer edges, namely, $\Theta(n^{\frac{2}{3}})$ instead of n. This leads to a more compact description of the paths the robot is supposed to move on. Another feature not shared by PL-environments is that the paths are locally C^1 between any two sites with C^1 boundaries, except for junctions with self-edges.

Note that, in order to keep maximal distance to the sites in S, the robot will not move on self-edges of $V(S)$. Such edges thus can be pruned before planning a motion (with possible exceptions close to start and target of the robot). As self-edges are the only place where leaves of $V(S)$ are present, the convergence speed of $\Theta(n^{-3})$ of the relevant parts of the Voronoi diagram is ensured.

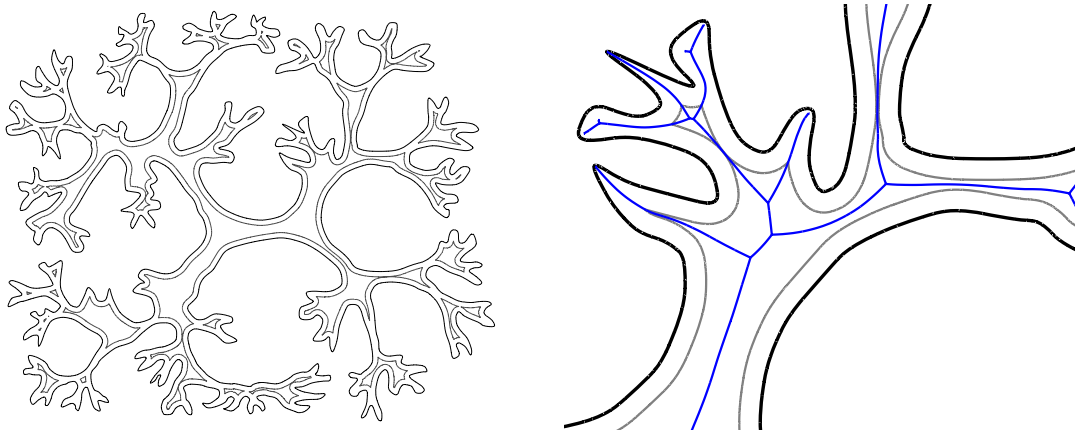
6.2 Trimmed offsets

Offsetting is a fundamental operation for planar shapes, and it is needed frequently, e.g., in computer-aided manufacturing [16, 26]. Several authors base their offsetting algorithms on the Voronoi diagram or the medial axis [16, 13, 4]. Once more, a PC-representation of the input shape is advantageous, because the class of such shapes is closed under offsetting operations.

Our Voronoi diagram algorithm is particularly well suited to this task, because it delivers the necessary combinatorial structure without computing the edge graph explicitly. Depending on the application, we can compute the inner or the outer offset of a given planar shape $\mathcal{A}$. For inner offsets, we take the outer boundary of $\mathcal{A}$ as the surrounding curve (replacing the circle Γ) and the holes of $\mathcal{A}$ as the sites. For outer offsets, we compute the inner offsets of the complement of $\mathcal{A}$ within a suitable disk covering $\mathcal{A}$.

Let $\mathcal{A}$ be a shape given in PC-representation. The *(trimmed inner) offset* of $\mathcal{A}$ at distance δ is defined as

$$\mathcal{A}^\delta = \mathcal{A} \setminus \bigcup_{x \in \partial A} D(x,\delta)$$



(a) Complete shape (b) Detail with axis

Figure 8: Shape, offset, and edge graph details.

where $D(x, \delta)$ is the disk with center x and radius δ; see Figure 6a. Its boundary $\partial\mathcal{A}^\delta$ consists of circular arcs again, which are offsets of the circular arcs in ∂A. However, simply offsetting $\partial\mathcal{A}$ does not give $\partial\mathcal{A}^\delta$, since self-intersections may be present. We use the corresponding Voronoi diagram, $V(S)$, to trim away these self-intersecting parts.

We first define certain subshapes of $\mathcal{A}$; consult Figure 6bc. The edge graph $\mathcal{G}_S$ consists of conic segments e_i, each being the bisector of two arcs a_i^1 and a_i^2. For a point x on either arc, consider the segment of the normal which is contained in $\mathcal{A}$ and connects x with e_i. The union of these line segments forms the subshape $\mathcal{A}_i \subseteq \mathcal{A}$ associated with e_i. In addition, each leaf v_j of $\mathcal{G}_S$ defines a subshape $\mathcal{A}_j$ as the circular region consisting of all line segments which connect the points of the arc with its center, v_j.

A subshape $\mathcal{A}_i$ is said to be *monotonic* if the radii of the maximal disks of $\mathcal{A}$ with centers on e_i have no inner extrema. The extremal radii $r_{\min}, r_{\max}$ are then realized at the boundaries. Depending on the position (respect to $\mathcal{A}_i$) of the line L spanned by the centers of the arcs a_i^1, a_i^2, the radii have no, one, or two extrema. The subshapes associated with leaves are already monotonic. Note that for splitting into monotonic subshapes we simply intersect a_i^1, a_i^2 with the line L, rather than computing the bisector of these arcs.

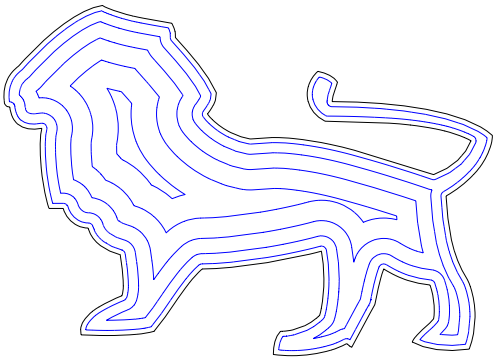


Figure 9: Inner offsets for different values of δ.

The offsetting is done separately for each monotonic subshape. If $\delta < r_{\min}$, then the offsets of the arcs at distance δ are fully contained in ∂A^δ. For $r_{\min} \le \delta \le r_{\max}$, the offset arcs are trimmed at their intersection; see Figure 6d, bottom. Finally, if $r_{\max} < \delta$, then the subshape does not contribute to ∂A^δ.

Error	SP	PT	Diagram	Offset
$k \cdot 10^{-1}$	732	468	0.07	0.02
$k \cdot 10^{-2}$	1230	916	0.16	0.04
$k \cdot 10^{-3}$	2656	1860	0.30	0.07
$k \cdot 10^{-4}$	5678	3872	0.64	0.15
$k \cdot 10^{-5}$	12044	8156	1.39	0.31

Table 5: Numbers of arcs (left) and runtimes (right) for the shape in Figure 9. The biarc types SP and PT have been used. Times are given in seconds for the type PT on a Pentium IV 2.8Ghz. The parameter k is a constant related to the bounding box of the input.

Error	SP	PT	Diagram	Offset
$k \cdot 10^{-1}$	9440	8768	2.24	0.29
$k \cdot 10^{-2}$	20132	17080	4.08	0.56
$k \cdot 10^{-3}$	43332	34008	7.14	1.03
$k \cdot 10^{-4}$	93224	69312	17.10	2.06
$k \cdot 10^{-5}$	201688	143348	29.53	4.25

Table 6: Arcs and runtimes for the shape in Figure 8.

An implementation shows that offset computations require only little additional time after the Voronoi diagram construction; Tables 5 and 6 give two examples. The total time thus will not increase much in applications where many different offset layers are needed. Note the difference in the numbers of biarcs needed to reach a given accuracy for both shapes.

7. REFERENCES

[1] O. Aichholzer, W. Aigner, F. Aurenhammer, T. Hackl, B. Jüttler, and M. Rabl. Medial axis computation for planar free-form shapes. *Computer-Aided Design*. To appear.

[2] O. Aichholzer, F. Aurenhammer, T. Hackl, B. Jüttler, M.Oberneder, and Z. Sír. Computational and structural advantages of circular boundary representation. In *Springer Lecture Notes in Computer Science*, volume 4619, pages 374–385, 2007.

[3] H. Alt, O. Cheong, and A. Vigneron. The Voronoi diagram of curved objects. *Discrete & Computational Geometry*, 34:439–453, 2005.

[4] L. Cao and J. Liu. Computation of medial axis and offset curves of curved boundaries in planar domain. *Computer-Aided Design*, 40(4):465–475, 2008.

[5] CGAL. Computational Geometry Algorithms Library. http://www.cgal.org/.

[6] B. Chazelle. A theorem on polygon cutting with applications. In *Proc. 23^{rd} Ann. IEEE Symp. Foundations of Computer Science*, pages 339–349, 1982.

[7] L.P. Chew and R.L. Drysdale. Voronoi diagrams based on convex distance functions. In *Proc. 1st Ann. ACM Symposium on Computational Geometry*, pages 235–244, 1985.

[8] F.Y.L. Chin, J. Snoeyink, and C.A. Wang. Finding the medial axis of a simple polygon in linear time. *Discrete & Computational Geometry*, 21:405–420, 1999.

[9] H.I. Choi, S.W. Choi, and H.P. Moon. Mathematical theory of medial axis transform. *Pacific Journal of Mathematics*, 181:57–88, 1997.

[10] G. Elber, E. Cohen, and S. Drake. MATHSM: Medial axis transform toward high speed machining of pockets. *Computer-Aided Design*, 37:241–250, 2005.

[11] I.Z. Emiris, E.P. Tsigaridas, and G.M. Tzoumas. The predicates for the Voronoi diagram of ellipses. In *Proc. 22nd Ann. ACM Symposium on Computational Geometry*, pages 227–236, 2006.

[12] S. Fortune. A sweep line algorithm for Voronoi diagrams. *Algorithmica*, 2:153–174, 1987.

[13] M. Held. Voronoi diagrams and offset curves of curvilinear polygons. *Computer-Aided Design*, 30(4):287–300, 1998.

[14] M. Held. VRONI: An engineering approach to the reliable and efficient computation of Voronoi diagrams of points and line segments. *Computational Geometry: Theory and Applications*, pages 95–123, 2001.

[15] M. Held. Topology-oriented incremental computation of Voronoi diagrams of circular arcs and straight line segments. *Computer-Aided Design*, to appear.

[16] M. Held, G. Lukacs, and L. Andor. Pocket machining based on contour-parallel tool paths generated by means of proximity maps. *Computer-Aided Design*, pages 189–203, 1994.

[17] D.G. Kirkpatrick. Efficient computation of continuous skeletons. In *Proc. 20th Ann. IEEE Symp. Foundations of Computer Science*, pages 18–27, 1979.

[18] R. Klein. *Concrete and Abstract Voronoi diagrams*. Springer Lecture Notes in Computer Science 400, 1990.

[19] R. Klein, K. Mehlhorn, and S. Meiser. Randomized incremental construction of abstract Voronoi diagrams. *Computational Geometry: Theory and Applications*, 3:157–184, 1993.

[20] D.T. Lee. Medial axis transformation of a planar shape. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 4:363–369, 1982.

[21] D.T. Lee and R.L.S. Drysdale. Generalization of Voronoi diagrams in the plane. *SIAM Journal on Computing*, 10:73–87, 1981.

[22] M. McAllister, D. Kirkpatrick, and J. Snoeyink. A compact piecewise-linear Voronoi diagram for convex sites in the plane. *Discrete & Computational Geometry*, 15:73–105, 1996.

[23] D. S. Meek and D. J. Walton. Spiral arc spline approximation to a planar spiral. *J. Comput. Appl. Math.*, 107:21–30, 1999.

[24] D.S. Meek and D.J. Walton. Approximating smooth planar curves by arc splines. *J. Comput. Appl. Math.*, 59:221–231, 1995.

[25] E. Pilgerstorfer. Asymptotischer Vergleich verschiedener Verfahren der Biarc-Approximation. *Master Thesis, Johannes Kepler University Linz, Austria*, 2008.

[26] J-K. Seong, G. Elber, and M-S. Kim. Trimming local and global self-intersections in offset curves/surfaces using distance maps. *Computer-Aided Design*, 38:183–193, 2006.

[27] M.I. Shamos and D. Hoey. Closest-point problems. In *Proc. 16th Ann. IEEE Symp. Foundations of Computer Science*, pages 151–162, 1975.

[28] M. Sharir. Intersection and closest-pair problems for a set of circular discs. *SIAM Journal on Computing*, 14:448–468, 1985.

[29] Z. Šír, R. Feichtinger, and B. Jüttler. Approximating curves and their offsets using biarcs and Pythagorean hodograph quintics. *Computer-Aided Design*, 38:608–618, 2006.

[30] C.-K. Yap. An $O(n \log n)$ algorithm for the Voronoi diagram of a set of simple curve segments. *Discrete & Computational Geometry*, 2:365–393, 1987.

[31] C.-K. Yap and H. Alt. Motion planning in the CL-environment. In *Lecture Notes In Computer Science 382*, pages 373–380, 1989.