International Journal of Computational Geometry & Applications

COMPUTATIONAL AND STRUCTURAL ADVANTAGES OF CIRCULAR BOUNDARY REPRESENTATION

OSWIN AICHHOLZER[1], FRANZ AURENHAMMER[2], THOMAS HACKL[1]
Institutes [1]for Software Technology and [2]for Theoretical Computer Science, University of Technology
Inffeldgasse 16B/II, 8010 Graz, Austria
oaich@ist.tugraz.at, auren@igi.tugraz.at, thackl@ist.tugraz.at

BERT JÜTTLER, MARGOT RABL
Institute of Applied Geometry, Johannes Kepler University
Altenberger Str. 69, 4040 Linz, Austria
bert.juettler@jku.at, margot.rabl@jku.at

ZBYNEK ŠÍR
Faculty of Mathematics and Physics, Charles University in Prague
Sokolovská 83, 186 75 Praha 8, Czech Republic
zbynek.sir@mff.cuni.cz

Received (received date)
Revised (revised date)
Communicated by (Name)

Boundary approximation of planar shapes by circular arcs has quantitative and qualitative advantages compared to using straight-line segments. We demonstrate this by way of three basic and frequent computations on shapes – convex hull, decomposition, and medial axis. In particular, we propose a novel medial axis algorithm that beats existing methods in simplicity and practicality, and at the same time guarantees convergence to the medial axis of the original shape.

Keywords: convex hull, decomposition, medial axis, circular arc, approximation by arc splines

1. Introduction

The plain majority of algorithms in computational geometry have been designed for processing *linear* objects, like lines, planes, or polygons. On the one hand, this is certainly due to the fact that many interesting and deep computational and combinatorial questions do arise already for inputs of this simple form. Again, the pragmatic reason is that algorithms for linear objects are usually both easier to develop and simpler to implement. To make things work for nonlinear objects, which arise frequently in practical settings, such objects are usually approximated in a piecewise-linear manner and up to a tolerable error. Existing approaches [15] to directly extending polygonal algorithms to curved objects are rare and, due to their generality, of limited practical use.

In its simplest form, the input object is a single planar shape, $\mathcal{A}$, with curved and connected boundary $\partial\mathcal{A}$. Frequent tasks to be performed on $\mathcal{A}$ – each being prior to a variety of more involved computations – include constructing the convex hull of $\mathcal{A}$, decomposing $\mathcal{A}$ into primitives, and calculating the medial axis of $\mathcal{A}$. These tasks are well investigated in the case of polygonal shapes. In certain situations, however, the number of line segments required for approximating $\partial\mathcal{A}$ with high accuracy may be prohibitively large. Even more seriously, making a piecewise-linear approximation of $\partial\mathcal{A}$ and invoking a polygonal-shape algorithm may generate results that are not homeomorphic to the correct result; the medial axis is a well-known example.

The intention of the present paper is to highlight the use of circular arcs for boundary representation. It is well known that for nonlinear curve segments the approximation order increases in comparison to using straight-line segments. For instance, it has been conjectured by Höllig and Koch [28], and proved for $d = 2$, that polynomial curves of degree d in the plane have approximation order $2d$. Many related results exist in the rich literature on geometric interpolation, including results on circular arcs [34].

In particular, if a given accuracy ε is achieved by using N line segments, then as few as $n = \Theta(N^{2/3})$ circular arcs can accomplish the same. This has been an issue in approximation theory, but in computational geometry this gain seems to have been less valued than eliminating small factors in the complexity of the subsequently applied algorithm. Boundary approximation by circular arcs may be of advantage also in a qualitative respect. For instance, it avoids the mentioned structural inconsistencies in medial axis computations, and it supports the computation of shape offsets, as the class of shapes bounded by circular arcs is closed under offset operations.

We will show that for the three basic problems mentioned above – convex hull, decomposition, and medial axis – simple and practical, though still efficient, algorithms exist that work for circular arc inputs. The first two problems are less demanding; we treat them mainly to point out the respective favorable (in our opinion) approach, whose practicality shall encourage the use of circular arc boundary representation. Nevertheless, substantial differences to the polygonal case occur; see below. For computing the medial axis, we propose a novel and extremely simple algorithm that is based on a known (though less recognized) decomposition lemma. After having computed a purely combinatorial description of the medial axis using tailored shape splitting, its individual parts (conics and line segments, like in the polygonal case) are re-assembled in trivial merge steps. The algorithm and its analysis are not specific to circular arc inputs.

Suitable circular arc approximations of shapes can be found in linear time. In summary, the obtained shape processing algorithms are superior in runtime to their line segment based counterparts, retain much (if not all) of their simplicity, and are even more natural in some cases.

2. Outline and Background

We briefly describe the contributions of this paper and relate them to existing literature.

Section 3 deals with approximating general curves by suitable primitives. This is a topic of importance in geometric modeling and in CAD and NC applications, and many quite recent results are available [16,26,35,36,38,41,43]. Our aim is to approximate a parametric curve $\boldsymbol{c}(t)$ by circular arcs. We assume that $\boldsymbol{c}(t)$ is piecewise-polynomial of constant degree, and we use biarcs (pairs of smoothly joined circular arcs) [36,40,41] as primitives. A straight-forward bisection algorithm for biarc generation already fits our purposes. It uniquely assigns biarcs to parameter intervals, which facilitates the error evaluation. An approximating spline curve $\boldsymbol{b}$ of size n is computed in $O(n)$ time. It fits the input curve $\boldsymbol{c}(t)$ in slope at biarc endpoints, and can be tuned to match $\boldsymbol{c}(t)$ in curvature at certain points (a fact being important in subsequent medial axis computations). Though not being optimal in the number of arcs, the approximation order of $\boldsymbol{b}$ is still three [35,41]. In contrast, with line segments one cannot exceed order two, and a polyline of size $N = \Theta(n^{3/2})$ is needed to arrive at the same precision.

The remaining sections propose algorithms for *circular arc shapes* $\mathcal{A}$, where the boundary $\partial\mathcal{A}$ of $\mathcal{A}$ is given as a connected curve composed of n circular arcs. Choice is guided by efficiency as well as by reducibility to basic operations that have robust implementations [18].

Section 4 outlines an algorithm for computing the convex hull of $\mathcal{A}$. This task is one of the most basic to be performed for a given shape, and has a variety of applications including shape fitting, motion planning, shape separation, and many others. At least four linear-time algorithms have been developed for polygonal shapes [6,25,33,37]. The incremental method by Melkman [37] stands out by its simplicity, and it is this candidate we generalize for circular arc shapes. Compared to the original setting, two difficulties arise. Deciding inclusion for a currently inserted arc in the convex hull constructed so far is no trivial test, and the convex hull cannot be described by a sequence of input vertices of the shape. We show that a runtime of $O(n)$ is still possible. The basic subroutine of the algorithm computes the convex hull of only two circular arcs.

Section 5 deals with shape triangulation, a fundamental building block in algorithms for decomposition, shortest path finding, and visibility – to name a few. Most existing algorithms are meant for polygonal shapes. They partition a given (simple) N-vertex polygon into triangles without introducing Steiner points. Efficient candidates are [5,9,23,27,32] which all show an $O(N \log N)$ runtime. Theoretically more efficient methods do exist, but when aiming at simplicity, choice should be made from the list above.

When trying to generalize to shapes $\mathcal{A}$ bounded by circular arcs, we face two problems. First of all, if the use of Steiner points is disallowed, then a partition of $\mathcal{A}$ into primitives bounded by a constant number of circular arcs need not exist. (In certain cases, however, such a partition of $\mathcal{A}$ will exist, but a partition with straight line segments will not.) Also, not all triangulation methods are suited to generalization. This applies, for instance, to the extremely simple ear cutting method in [30] which runs in time $O(r \cdot N)$, where r is the number of reflex vertices of $\mathcal{A}$. The triangulation algorithm we propose is closest to Chazelle's [9]. It manages with an (almost) worst-case minimal number of Steiner points on $\partial\mathcal{A}$, runs in $O(n \log n)$ time, and uses a dictionary as its only nontrivial data structure.

The produced primitives are arc triangles with at least one straight edge. The most complex geometric operation is intersecting a circle with a line.

Section 6 is devoted to the medial axis, a frequently used structure associated with a given input shape. Its main applications include shape recognition, solid modeling, pocket machining, and others. Interest in mathematical properties of the medial axis and the Voronoi diagram for general shapes found renewal in recent years [3,7,8,11,19,20,39]. In our case, where the shape $\mathcal{A}$ is simply connected and $\partial\mathcal{A}$ consists of n circular arcs, its medial axis $M(\mathcal{A})$ is known to be a tree composed of $O(n)$ conic edges. Algorithmic work on the (exact) medial axis either concentrated on the case where $\mathcal{A}$ is a polygon [9,10,31], or on general sets of curved arcs [2,12,29,39] (and their Voronoi diagram) without, however, exploiting the fact that the input arcs define a simple curve. Though theoretically efficient as $O(n \log n)$ or better, these algorithms suffer from involved merge or insertion steps which, even for straight arcs as input, are difficult to implement. In addition, numerical stability issues arise heavily; intersections of conics have to be determined repeatedly which, when not calculated exactly, are bound to accumulate the error. If the vertices of the medial axis are assumed to be known, the in-between edges can be traced numerically [12,17]. This approach, however, requires an expensive a-priori analysis of the global connectivity structure of the medial axis.

We present a simple randomized divide-and-conquer algorithm for computing $M(\mathcal{A})$ that overcomes these drawbacks. In contrast to comparable algorithms, the costly part is delegated to the divide step. The geometrically most complex operations in this step are computing the intersection of two circles. The merge step is trivial: it concatenates two medial axes. The expected runtime is bounded by $O(n^{3/2}) = O(N)$, but is provably better for most types of shape. For example, $O(n \log n)$ expected time suffices if the diameter of $M(\mathcal{A})$ is $\Theta(n)$. No nontrivial data structures are used.

To guarantee applicability of our methods to approximating the medial axes of general shapes $\mathcal{A}$, a convergence result is needed. We prove in Section 7 that, for a suitable approximation of $\partial\mathcal{A}$ by circular arcs, $M(\mathcal{A})$ is the limit of $M(\mathcal{B})$ when the approximating arc shape $\mathcal{B}$ converges to $\mathcal{A}$. Related results exist, but either presuppose C^2 conditions on $\partial\mathcal{B}$ not attainable by circular arcs [8], or concern only subsets of the medial axis [7] that survive after pruning the Voronoi diagram of point samples from $\partial\mathcal{A}$. It is well known [3] that medial axis convergence is *not* given for polygonal approximations of $\mathcal{A}$. While certain conditions on the approximation with circular arcs guarantee convergence, there is no way to guarantee convergence for point samples or line segments without pruning. If we do restrict ourselves to the pruned part in the case of a point sample (the λ-medial axis [7]), a data volume of $\Theta(n^3) = \Theta(N^2)$ arises, compared to n circular arcs or N line segments, for the same approximation quality. In conclusion, circular arcs are the simplest possible tool for boundary conversion that guarantees a stable medial axis approximation.

We mention that there exist several point-based methods to approximate the medial axis in 3D. Using a proper subset of the Voronoi facets, chosen via the edges of the dual Delaunay triangulation, a convergence guarantee when taking an ε-sampling can be shown, see [13,14]. However, these approaches are not needed for 2D, as the Voronoi diagram vertices

of a sample already converge to the medial axis in this case.

3. Approximation by Circular Arcs

In order to represent a general shape $\mathcal{A}$ in a form suitable for geometric computations, we discuss methods for approximating $\partial\mathcal{A}$ by circular arcs. We assume that $\partial\mathcal{A}$ is given as a polynomial spline curve. While particular attention is paid to the cubic case, being the most popular one in applications [21], the methods can be applied to spline curves of any degree.

Several approaches to generating circular arc splines exist; see e.g. [35] for a review. We consider a simple bisection algorithm consisting of two steps, approximation and error measurement. A geometric primitive b (an arc or a biarc) is fitted to a segment s of the given curve $\boldsymbol{c}(t)$, and the distance from b to s is numerically computed. The algorithm is relatively easy to implement and still adapts the degrees of freedom to the input data. As a slight disadvantage, the number of primitives (the resulting data volume) is optimal only in the asymptotic sense.

Define the one-sided Hausdorff distance from a primitive b to a segment $s \subseteq \boldsymbol{c}(t)$ as

$$\delta(b, s) = \max_{p \in b} \min_{q \in s} ||p - q||.$$

(We consider b and s as closed sets.) Let ε denote the error tolerance to be met by the algorithm.

Algorithm BISECT(t_0, t_1)

Construct b
Compute $\delta = \delta(b, \boldsymbol{c}[t_0, t_1])$
If $\delta \leq \varepsilon$ then return $\{b\}$
Else return BISECT$(t_0, \frac{t_0+t_1}{2}) \cup$ BISECT$(\frac{t_0+t_1}{2}, t_1)$

Depending on the primitive b used, Algorithm BISECT produces splines of different quality: merely continuous (C^0) circular arc splines, or tangent continuous (C^1) arc splines. When being content with the former type, we simply can choose for b the unique circular arc passing through the three points $\boldsymbol{c}(t_0)$, $\boldsymbol{c}(\frac{(t_0+t_1)}{2})$, and $\boldsymbol{c}(t_1)$. To obtain C^1 arc splines, so-called biarcs [40] are utilized.

A *biarc* b consists of two circular arcs with common unit tangent vector at their joint. Usually, b is described by its source x with associated unit tangent vector v_x, and its target y with unit tangent vector v_y. Given these data, there exists a one-parameter family of interpolating biarcs. All possible joints are located on the circle σ passing through x and y and having the same oriented angles with v_x and v_y. Several ways for choosing the joint m have been proposed; see e.g. [36,41]. For many applications, taking $m = \sigma \cap \boldsymbol{c}[t_0, t_1]$ is appropriate. To calculate m in the cubic case, a polynomial of degree 4 has to be solved (where a closed-form solution is still available). The output is a C^1 arc spline with all arc endpoints sitting on $\boldsymbol{c}(t)$.

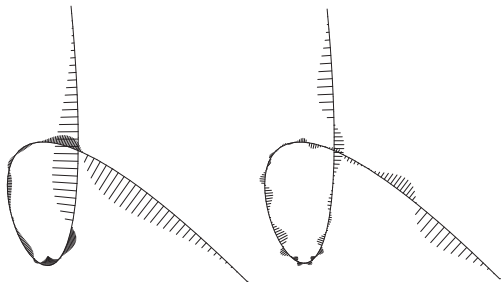

Fig. 1: z-magnified error for n biarcs

In view of subsequent stable medial axis computations, the choice of m has to be made more carefully. Define an *apex* of $\boldsymbol{c}(t)$ as a local curvature maximum. The apices split the curve $\boldsymbol{c}(t)$ into pieces of monotonic signed curvature, so-called *spirals*. Following [36], we aim at approximating spirals of $\boldsymbol{c}(t)$ by circular arc spirals. To this end, we split $\boldsymbol{c}(t)$ at its apices. In the cubic case, these points can be found by solving polynomials of degree 5. Now, we exploit that spiral biarcs can be constructed that connect two given points x and y, match unit tangents there, and assume a predefined curvature in one of them. Let k_x and k_y be the curvature of $\boldsymbol{c}(t)$ at x and y, respectively, and suppose $k_x < k_y$. To match curvature at x, we choose the radius of the first arc, b_1, equal to $r_x = 1/k_x$. The joint m is obtained by intersecting the circle supporting b_1 with the joint circle σ. According to [36], the radii and curvatures satisfy $r_x > r_y > 1/k_y$. When starting the next biarc from y with $r_y = 1/k_y$ (unless y is an apex), monotonicity of signed curvature will be preserved.

Each arc is found in $O(1)$ time, where the constant depends on the degree of the polynomial to be solved. Fig. 1 shows an example of a biarc conversion. The scaled curve normals visualize the magnified error distribution.

Concerning the error measurement, each produced circular arc b_i has to be matched to its corresponding segment $s = \boldsymbol{c}[t'_0, t'_1]$. This is, of course, trivial when the biarc joint m has been chosen to lie on $\boldsymbol{c}(t)$. In the case of biarc spirals, we intersect $\boldsymbol{c}(t)$ with the normal of b_i at m. In the case of degree 3 input curves, this leads to a cubic equation. If multiple solutions within the total biarc interval $[t_0, t_1]$ exist, then the error is set to ∞. Otherwise, we compute the one-sided Hausdorff distance $\delta(b_i, s)$ by substituting the parametric representation of s into the implicit equation K (with leading coefficients 1) of the circle supporting b_i. If r is the radius of K, and d and D are the minimum and maximum values of $(K \circ \boldsymbol{c})(t)$ for $t \in [t'_0, t'_1]$, we get

$$\delta(b_i, s) \leq \max\{|\sqrt{r^2 - d} - r|, |\sqrt{r^2 + D} - r|\}$$

and this bound is sharp. Consequently, in the cubic case, $\delta(b_i, s)$ can be evaluated by solving a quintic polynomial equation on the interval $[t'_0, t'_1]$. Alternatively, a simpler upper bound can be calculated (without polynomial solving) by replacing d and D with the minimum and maximum coefficient of the Bernstein-Bézier representation [22] of $K \circ \boldsymbol{c}$ with

respect to $[t_0', t_1']$. As the length of s decreases, this bound converges to $\delta(b_i, s)$. As another simple but important observation, the *two-sided* Hausdorff distance between b_i and s, $\max\{\delta(b_i, s), \delta(s, b_i)\}$, vanishes with $\delta(b_i, s)$ because b_i and s are of constant degree. Thus controlling the latter distance already ensures that b_i and s are ε-close with respect to the former one.

In summary, when algorithm BISECT spans a binary recursion tree with n leaves (the returned n primitives), any of the described types of arc splines can be constructed in $O(n)$ time.

Let us discuss the asymptotic behaviour of the number n for decreasing tolerance ε. For a given curve $\boldsymbol{c}(t)$ with domain $[t_0, t_1]$, which is assumed to contain neither inflections nor apices, we consider primitives having approximation order k. Adapting the analysis in [35,41] (as done in the Appendix), we get $\delta = \Theta(h^k)$ for the one-sided Hausdorff distance δ, provided that $\boldsymbol{c}(t)$ is approximated with (small) parameter step size h, and that k is considered a constant.

This relation implies a general lower bound. For *any* approximation of $\boldsymbol{c}(t)$ obtained by BISECT(t_0, t_1) using n primitives with approximation order k, the largest step size satisfies $\Delta t \geq \frac{t_1 - t_0}{n}$. Moreover, we have $\delta \leq \varepsilon$ by the terminating condition of the approximation algorithm. From $\delta = \Theta((\Delta t)^k)$, we get $n = \Omega(1/\varepsilon^{1/k})$. On the other hand, the minimum step size $\Delta' t$ taken by any algorithm for an interval I satisfies $\Delta' t \leq \frac{t_1 - t_0}{n}$. Assume we stop BISECT$(t_0, t_1)$ with doubled step size $2\Delta' t$. Then there exists at least one interval, for example the one containing I, for which $\delta > \varepsilon$. As we have $\delta = \Theta((2\Delta' t)^k)$ it follows that $n = O(1/\varepsilon^{1/k})$. We obtain:

Lemma 1. *For sufficiently small tolerance ε, the number n of primitives constructed by algorithm BISECT is asymptotically optimal.*

Lemma 1 also holds in the general case where $\boldsymbol{c}(t)$ contains inflections and apices, because the resulting number of spirals of $\boldsymbol{c}(t)$ is independent of n. In conclusion, to arrive at tolerance ε, Algorithm BISECT needs $n = \Theta(1/\sqrt[3]{\varepsilon})$ circular arcs (order 3), whereas $N = \Theta(1/\sqrt{\varepsilon})$ line segments (order 2) have to be invested by any polygonal approximation method.

Corollary 1. *Compared to approximating the curve $\boldsymbol{c}(t)$ with a polyline, the data volume drops from N to $n = \Theta(N^{2/3})$ when circular arc splines are used.*

It should be observed that, the other way round, when approximating $\boldsymbol{c}(t)$ with a point sample (as commonly done for medial axis computations [3]), the data volume increases to $\Theta(n^3)$ compared to n circular arcs.

4. Convex Hull

Let $\mathcal{A}$ be some shape given in arc boundary representation. More specifically, $\partial\mathcal{A}$ is approximated by a simple (i.e., not self-crossing) and connected curve $\boldsymbol{b}$ composed of n

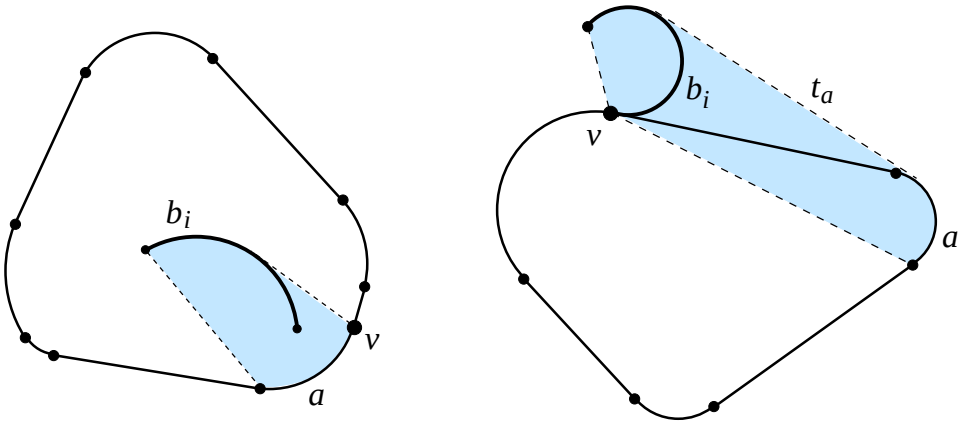

Fig. 2: Cases 2.1 (left) and 2.4 (right)

circular arcs. Clearly, if $\boldsymbol{b}$ converges to $\partial\mathcal{A}$ then the convex hull of $\boldsymbol{b}$ converges to the convex hull of $\mathcal{A}$. Moreover, the Hausdorff distance of the two convex hulls is bounded by the Hausdorff distance of $\boldsymbol{b}$ and $\partial\mathcal{A}$. We show that the convex hull algorithm for polylines in Melkman [37] can be generalized to simple circular arc curves $\boldsymbol{b}$ while retaining its $O(n)$ runtime.

In a nutshell, this algorithm processes each of the vertices of the given polyline in order and maintains their convex hull. If the currently processed vertex v_i falls into the convex hull, CH_{i-1}, constructed so far then v_i is deleted and we put $CH_i = CH_{i-1}$. Otherwise, tangents are placed from v_i to CH_{i-1}, and the sequence of vertices (if any) between the corresponding two vertices of tangency is deleted from CH_{i-1} in order to construct CH_i.

The linear runtime of this strategy hinges on two propositions: (1) A constant-time inclusion test $v_i \in CH_{i-1}$, and (2) deletion of vertices of CH_{i-1} which are non-extreme in CH_i in time proportional to their number. While (2) is achieved by a standard Graham scan [24], proposition (1) is met by exploiting simplicity of the given polyline: $v_i \in CH_{i-1}$ is equivalent to the fact that v_i lies in the wedge spanned by the interior angle at v, where v was the last vertex added to CH_{i-1}.

Staying with vertices works correctly with polygonal curves because the convex hull of two points equals the convex hull of their connecting line segment. This is, of course, not true for a connecting circular arc. As a consequence, the set of vertices of the convex hull to be constructed is, in general, no subset of the input vertices. Also, the inclusion test for a circular arc to be inserted is a more complicated operation, The following variant of Melkman's algorithm is able to cope with circular (and more general) arcs and still runs in $O(n)$ time. Its main subroutine computes the convex hull of only two arcs.

Let $b_1 \dots b_n$ be the given simple circular arc curve. The second endpoint of each arc b_i (in this order) is called the *target* of b_i. Some of the arcs may be line segments, and the

curve may be cyclic. Assume first that the curve is C^1. Let CH denote the convex hull operator, and abbreviate $CH(b_1 \dots b_i)$ as CH_i. Consult Figure 2.

Algorithm HULL

Construct $CH_2 = CH(b_1 b_2)$. Let v be the last point along the chain $b_1 b_2$ that lies on CH_2.

For $i = 3, \dots, n$, process the arc b_i as follows:

Search for the first arc, a, of CH_{i-1} clockwise from v that contributes with non-zero length to $CH(a, b_i)$ and such that this hull and CH_{i-1} are on the same side of a. Similarly, search for the first arc, c, counter-clockwise from v with analogous properties. ($a = c$ is possible.) Arcs a and c already provide the information needed to construct CH_i correctly.

Case 1 Arc a (and equivalently, arc c) does not exist. This means $CH_{i-1} \subset CH(b_i)$. Put $CH_i = CH(b_i)$, and assign to v the target of b_i.

Case 2 Arcs a and c do exist. Check for some tangent, t_a, which appears on $CH(a, b_i)$ and is clockwise tangent to CH_{i-1}, c.f. Figure 2 (right). Also, check for some tangent, t_c, which appears on $CH(c, b_i)$ and is counter-clockwise tangent to CH_{i-1}.

Case 2.1 Tangents t_a and t_c both do not exist. This means $b_i \in CH_{i-1}$. Put $CH_i = CH_{i-1}$.

Case 2.2 t_a exists (uniquely) but t_c does not. Let $t_a = x_a y_a$, where x_a is its point of tangency on CH_{i-1}. To obtain CH_i, delete from CH_{i-1} the clockwise part between v and x_a, and add t_a and the piece of the arc b_i between y_a and v. Update v as the last point along b_i on CH_i (either y_a or b_i's target).

Case 2.3 t_c exists (uniquely) but t_a does not. Let $t_c = x_c y_c$, with x_c being its point of tangency on CH_{i-1}. To get CH_i, delete from CH_{i-1} the counter-clockwise part between v and x_c, and add t_c and the piece of the arc b_i between y_c and v. Update v as in Case 2.2 (either y_c or b_i's target).

Case 2.4 t_a and t_c both do exist. Here we get CH_i by deleting arcs from CH_{i-1} as in Cases 2.2 and 2.3, and then adding t_a, t_c, and the piece of b_i between y_a and y_c. We update v as the point among y_a and y_c that is closer to the target of b_i.

Correctness of algorithm HULL is verified by observing that t_a and t_c are indeed tangents from the currently inserted arc b_i to the convex hull CH_{i-1} constructed so far. Thereby, as the algorithm stands now, it is of importance that the input curve is C^1. This guarantees that the boundary of CH_{i-1} is C^1 as well (except possibly at the target of b_{i-1}), such that the arcs a and c are found correctly. Minor modifications in the selection criteria for these arcs will make the algorithm work without this restriction.

The runtime is dominated by the search for a and c, where the necessary number of calls of the two-arc hull subroutine is proportional to the total number of arcs constructed or deleted. This number is $O(n)$ because only $O(1)$ arcs are constructed per i-loop. The rest can be accomplished in $O(1)$ time per arc b_i if CH_i is stored as a doubly linked list, or in $O(n)$ total time if CH_i is represented in a (more space-saving) dequeue.

5. Triangulation

We next propose a triangulation algorithm for circular arc shapes. Define an *arc triangle* as a (simply connected) face bounded by at most three circular arcs or line segments.

A partition of a circular arc shape $\mathcal{A}$ into arc triangles need not always exist when the use of Steiner points is disallowed. (Observe, however, that such a partition may exist, although a partition of $\mathcal{A}$ with straight line segments may not.) The situation does not change if the n arcs describing ∂A are x-monotone pieces (and hence span semi-circles at most), which we will assume below. In fact, there are examples where at least $2n-7$ Steiner points are necessary. See Figure 3. For no pair of vertices of the depicted shape $\mathcal{A}$ does there exist a connecting circular arc inside $\mathcal{A}$. Thus no part of $\mathcal{A}$ can be split off using a circular arc between two vertices. The interested reader may convince her/himself that placing $n-4$ Steiner points as shown is no waste. The asserted lower bound then follows, because each of the resulting faces needs additional Steiner points. Note that a single point per face suffices only if circular arcs rather than line segments are used to split the face.

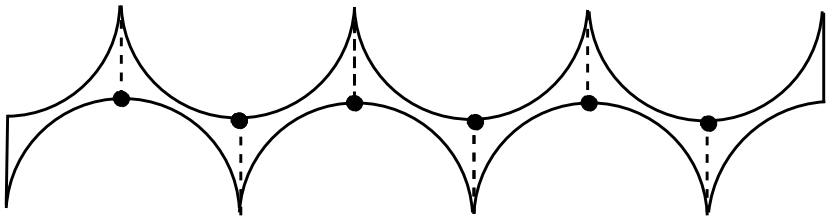

Fig. 3: Many Steiner points

The triangulation algorithm we are going to describe introduces at most $2n-5$ Steiner points (on the boundary of $\mathcal{A}$, rather than in its interior), runs in $O(n \log n)$ time, and uses a dictionary as its most involved data structure. The produced primitives are arc triangles where at least one edge is a line segment. Standard plane sweep is used to compute the vertical visibilities inside $\mathcal{A}$ for each pair (vertex, arc) of ∂A. Each such pair defines a vertical line segment that splits $\mathcal{A}$ and ends at a Steiner point on ∂A. A decomposition of $\mathcal{A}$ into arc triangles and arc trapezoids results. No priority queue is needed, as all events guiding the plane sweep (namely, the vertices of ∂A) are known in advance and thus can be x-sorted beforehand. For simplicity, suppose that their x-coordinates are pairwise different.

Lemma 2. *The decomposition above contains exactly $n-2$ Steiner points.*

Proof. Let us call a vertex *type* k if it vertically sees exactly k arcs, i.e., defines k Steiner points. We have vertices of types $0, 1$, and 2. At each type-2 vertex v, the shape $\mathcal{A}$ is vertically split into three parts, each part having a type-0 vertex as an x-extremum. Two such parts lie on the same side of the splitting segment, and among their extreme type-0 vertices, we map v to the one which is x-closer to v. This mapping is injective, and does not address the two x-extrema of ∂A. The lemma follows. □

The obtained faces are exactly $n-1$ in number, at least two being arc triangles. Each face F that is an arc trapezoid can be easily split into arc triangles. If F is convex then a line segment will do. Also, if at least one of the two arcs on ∂F is avoided by the central line g of their supporting circles, then a single splitting arc or line segment for F exists (because there is a normal to g that touches that arc at an endpoint). Otherwise, we use an intersection of g with a reflex arc on ∂F as a Steiner point and split F with two arcs. Figure 4 illustrates two typical cases. In total, at most $2n-5$ Steiner points are used for an arc triangulation.

We stress the fact that generalizing the classical plane sweep for polygon triangulation [27] – though well possible in $O(n \log n)$ time – results in a more complicated algorithm for arc triangulation. Large parts already swept across have to be remembered for later processing, and the produced primitives are more complex than arc trapezoids. Also, line segments being simultaneously tangent to two given circles have to be calculated, whereas in our algorithm the most complex operation is intersecting a circle with a straight line. As an open question we pose finding an algorithm that *always* manages with a (nearly) optimal number of Steiner points, not only in the worst case.

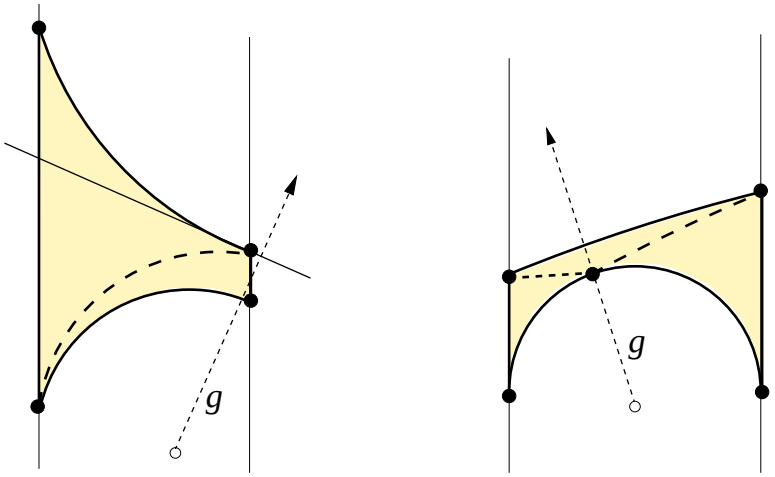

Fig. 4: Splitting arc trapezoids

6. Medial Axis

Let $\mathcal{A}$ be the circular arc shape under consideration. (In the sequel, all objects are considered to be topologically closed sets.) Call a disk $D \subseteq \mathcal{A}$ *maximal* if there exists no disk D' different from D such that $D' \supset D$ and $D' \subseteq \mathcal{A}$ holds. The medial axis, $M(\mathcal{A})$, of $\mathcal{A}$ is defined as the set of all centers of maximal disks.

As the boundary of $\mathcal{A}$ is a connected and simple curve with n circular arcs, $M(\mathcal{A})$

consists of finitely many conic arcs, is connected, and cycle-free [11] and thus forms a tree, cf. [11]. $M(\mathcal{A})$ can be decomposed into $O(n)$ *edges*, which are maximal pieces of straight lines and (possibly all four types of) conics. Endpoints of edges will be called *vertices* of $M(\mathcal{A})$. Compared to polygonal shapes, the medial axis for circular arc shapes is not more complicated, as both structures contain edges of degree 2 in general.

The contribution of this section is a simple and practical randomized algorithm for computing $M(\mathcal{A})$. It works by divide-and-conquer and accepts as input any description of $\partial\mathcal{A}$ by circular arcs and/or line segments. The costly part is delegated to the divide step, which basically consists of inclusion tests for arcs in circles. In particular, no conics take part in these calculations. The merge step is trivial; it just concatenates two partial medial axes. The expected runtime is bounded by $O(n^{3/2})$, and will be proved to be $O(n \text{ polylog } n)$ for several types of shape. A qualitative difference to existing medial axis algorithms is that a *combinatorial* description of $M(\mathcal{A})$ is extracted first, which can then be directly (and robustly) converted into a geometric representation. We base our algorithm on the following simple though elegant decomposition lemma [11].

Lemma 3. *Consider any maximal disk D for $\mathcal{A}$. Let $A_1, \ldots, A_t$ be the connected components of $\mathcal{A} \setminus D$, and denote with p the center of D.*

$$\textit{(1)} \quad M(\mathcal{A}) = \bigcup_{i=1}^{t} M(A_i \cup D)$$

$$\textit{(2)} \quad \{p\} = \bigcap_{i=1}^{t} M(A_i \cup D)$$

In plain words, having at hands some maximal disk one can compute the medial axes for the resulting components recursively, and then glue them together at a single point. However, the desired efficiency of this strategy calls for a balanced decomposition. Its existence is given below.

Lemma 4. *There exists a maximal disk D for $\mathcal{A}$ such that at most $\frac{n}{2}$ arcs from $\partial\mathcal{A}$ are (completely) contained in each component of $\mathcal{A} \setminus D$.*

Proof. Each point $p \in M(\mathcal{A})$ corresponds to a unique maximal disk D_p for $\mathcal{A}$. Let $f(D_p)$ be the number of arcs from $\partial\mathcal{A}$ in the largest component induced by D_p. As long as $f(D_p) > \frac{n}{2}$, the component that realizes $f(D_p)$ is unique, and we can decrease $f(D_p)$ by continuously moving p on $M(\mathcal{A})$ such that D_p enters into this component. This process terminates at some point p^* where $f(D_{p^*}) \leq \frac{n}{2}$. We never move back the way we came, as the component we move out never exceeds a size of $\frac{n}{2}$. □

We are left with the algorithmic problem of finding some maximal disk that yields a well-balanced partition. Observe that the optimal point p^* above may be not unique, because the number $f(D_p)$ is invariant under motion of p within the relative interior of any fixed edge $e \subset M(\mathcal{A})$. Let us define *Walk*(e) as the path length in $M(\mathcal{A})$ from e to p^*.

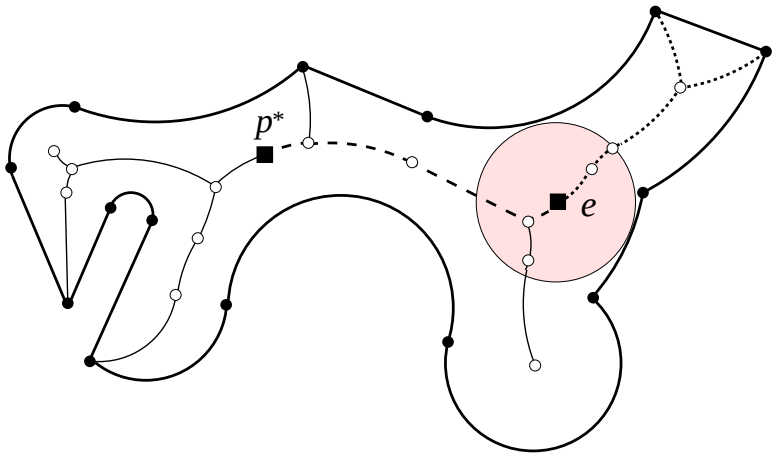

Fig. 5: Walk (dashed) and cut (dotted)

Further, define *Cut*(e) as the size of the smaller one among the two subtrees which constitute $M(\mathcal{A}) \setminus \{e\}$. See Figure 5. Any tree with small 'cuts' tends to have short 'walks', in the following respect.

Lemma 5. *Let e be an edge of $M(\mathcal{A})$, chosen uniformly at random. Then $E[$Walk$(e)] = \Theta(E[$Cut$(e)])$.*

Proof. Orient all the paths in $M(\mathcal{A})$ away from the point p^*. This defines a partial order $\prec$ on the edges of $M(\mathcal{A})$. That is, for any two edges e and e' on the same path to p^*, we write $e' \prec e$ if e' is at closer distance to p^*. We have the set equality

$$\bigcup_{e \in M(\mathcal{A})} \{(a, e) \mid a \prec e\} = \bigcup_{e \in M(\mathcal{A})} \{(e, b) \mid b \succ e\}$$

because either set contains each pair of the relation exactly once. The (disjoint) subsets united in the left set, L, represent all the paths in $M(\mathcal{A})$ between its edges e and p^*. Thus we have $E[$Walk$(e)] = \frac{1}{m} \cdot |L|$, where m is the number of edges of $M(\mathcal{A})$. Each subset united in the right set, R, represents that one among the two subtrees in $M(\mathcal{A}) \setminus \{e\}$ which avoids p^*. So we get $\frac{1}{m} \cdot |R| > E[Cut(e)]$, because for Cut$(e)$ we always consider the smaller subtree. Moreover, if we neglect in R all the subtrees of sizes larger than $\frac{m}{2}$, then the cardinality of the set drops by a constant factor (of at most 4, if $\prec$ would be a total order, hence less). This implies $E[$Cut$(e)] > \frac{1}{m} \cdot \frac{|R|}{4}$. The lemma now follows from $|R| = |L|$. □

Lemma 5 motivates the following disk finding algorithm which combines random cutting with local walking. Its main subroutine, MAX(b), selects for an arc $b \subset \partial\mathcal{A}$ its midpoint x and returns the unique maximal disk for $\mathcal{A}$ with x on its boundary. For the ease of description, we assume that this disk splits $\mathcal{A}$ into exactly two components. The algorithm

can be easily adapted, otherwise, and its runtime even decreases. Let $c \geq 3$ be a (small) integer constant.

Procedure CUT$(\mathcal{A})$

Put $A' = \mathcal{A}$
Repeat
  Choose a random arc b of $\partial A'$
  Compute D=MAX(b) and let $\mathcal{A}_0$ be the larger
    component of $\mathcal{A}$ induced by D
  Assign $A' = A' \cap \mathcal{A}_0$
Until $\mathcal{A}_0$ contains less than $n - \frac{n}{c}$ arcs
Report D

Procedure WALK$(\mathcal{A})$

Choose a random arc b of $\partial\mathcal{A}$
Compute D=MAX(b)
Let $\mathcal{A}_0$ be the larger component induced by D
While $\mathcal{A}_0$ contains more than $n - \frac{n}{c}$ arcs do
  Let b_1 (b_2) be the first (last) complete arc of $\partial\mathcal{A}$ in $\mathcal{A}_0$
  Compute D_1=MAX(b_1) and D_2=MAX(b_2)
  Assign to $\mathcal{A}_0$ the smaller one of the respective larger
    components of $\mathcal{A}$ for D_1 and D_2
  Memorize the corresponding disk $D \in \{D_1, D_2\}$
Report D

The disk finding algorithm now combines the CUT procedure and the WALK procedure as follows. The repeat loop of CUT and the while loop of WALK are executed by turns. Whenever CUT is closer to the goal (i.e., yields a smaller largest component than does WALK), we readjust the current disk for WALK to be that of CUT. Termination takes place in either WALK or CUT.

To analyze the resulting runtime, let us first consider the assignment of arcs on $\partial\mathcal{A}$ to edges of $M(\mathcal{A})$, as done in subroutine MAX. Namely, if MAX(b)=D then arc b is mapped to the edge e that contains the center of D. Observe that either $0, 1$, or 2 arcs are mapped to a fixed edge. Moreover, no two unaddressed edges and no two doubly addressed edges are neighbored. This assignment is sufficiently uniform to convey randomness from arcs to edges.

It remains to apply the result of Lemma 5, which asserts that no tree structure can have both small cuts and long walks. Observe that this is, thus, true for the medial axes of all the subshapes of $\mathcal{A}$ considered by the algorithm. In the worst case of walk length being balanced with cut number, a bound of $O(\sqrt{n})$ on the expected number of total loop

executions in CUT and WALK holds.[a]

The costly part in both procedures is their subroutine MAX, whose expected number of calls obeys the same bound, $O(\sqrt{n})$. Computing D=MAX(b) has a trivial implementation which runs in $O(n)$ time: We initialize the disk D as the (appropriately oriented) halfplane that supports b at its midpoint x and, for all remaining arcs $b_i \subset \partial\mathcal{A}$ that intersect D, we shrink D so as to touch b_i while still being tangent to b at x. The most complex operation for shrinking D is computing the intersection of two circles. In particular, and unlike previous medial axis algorithms, no conics take part in geometric operations.

In summary, the randomized complexity for computing the medial axis is given by $T(n) = T(\frac{1}{c}n) + T((1-\frac{1}{c})n) + O(n^{3/2})$ for $c \geq 3$, which evaluates to $T(n) = O(n^{3/2})$. In many cases, however, will the algorithm perform substantially better. Let d be the graph diameter of $M(\mathcal{A})$. Then the loop in WALK$(\mathcal{A})$ is executed less than d times. So, for example, if $d = \Theta(\log n)$ then an overall runtime of $O(n \log^2 n)$ is met. For the other extreme case, $d = \Theta(n)$, our strategy is even faster. With constant probability, an edge on the diameter is chosen, and $\Theta(n)$ such edges e have *Cut*$(e) = \Theta(n)$. The expected number of loop executions in CUT$(\mathcal{A})$ now is only $O(1)$, and an $O(n \log n)$ algorithm results. We conjecture that the latter situation is quite relevant in practice. In many applications, for typical shapes their medial axes will not branch extensively. Even if so, the branching will be independent of n, because each branch will be approximated by a large number of circular arcs in order to achieve the predefined precision.

The output of the algorithm is a list of $O(n)$ points on $M(\mathcal{A})$, namely, the centers of the splitting disks, plus a list of $O(n)$ edges connecting them. Each edge is given implicitly by its defining two arcs on $\partial\mathcal{A}$. To make sure that the reported point list includes all the vertices of $M(\mathcal{A})$, base cases that involve constantly many (pieces of) original arcs from $\partial\mathcal{A}$ have to be solved directly. (The constant is at most 3 if $\partial\mathcal{A}$ is C^1.) Note that the algorithm works exclusively on $\partial\mathcal{A}$ except for a final step, where the conic edges of $M(\mathcal{A})$ are explicitly calculated and reassembled. This gives rise to increased numeric stability in comparison to existing approaches.

Opposed to approximating $\partial\mathcal{A}$ with the same accuracy by a polyline of size N, our circular arc algorithm takes $O(n^{3/2}) = O(N)$ time; see Corollary 1 in Section 3. Thus, even for (probably rare) worst-case inputs, our simple algorithm competes asymptotically well with previous methods. Other advantages over polygonal (and also point sample) approximations are described in the next section.

7. Convergence of Medial Axis

A well-known unpleasant phenomenon of the medial axis is its instability under perturbations of the shape boundary. Several papers discussing this issue have been published

[a]There is a subtlety to be noted in the (degenerate) case where $M(\mathcal{A})$ is a tree of non-constant maximal degree. As we always cut and walk on edges instead of vertices of $M(\mathcal{A})$, termination might not be reached. This problem can be dealt with easily with a modification described in [1].

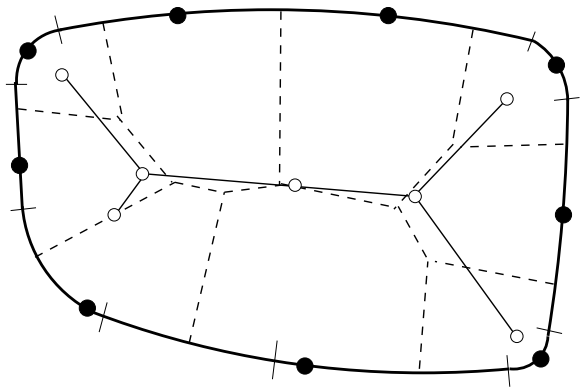

Fig. 6: Small point sample (dashed Voronoi diagram) versus few arcs (solid medial axis)

recently. A result in [8] shows that stability is, in general, not given unless perturbations are C^2. In particular, medial axis convergence is not guaranteed for polygonal approximations. To deal with general shapes, the so-called λ-medial axis has been introduced as a tool in [7]. After drawing a point sample from the shape boundary, the Voronoi diagram of these points is constructed and pruned appropriately. The λ-medial axis converges to the original for vanishing sample distance. Drawbacks are the large sample size for a close (and homotopy-equivalent) approximation, the lack of its C^1 behavior, and the need of computing a general planar Voronoi diagram. Figure 6 gives an illustrative example.

We prove in this section that convergence of the medial axis under the Hausdorff distance comes as a byproduct of the careful (though, of course, still C^1) biarc boundary conversion described in Section 3.

Given some shape $\mathcal{A}$ and a point p on its medial axis $M(\mathcal{A})$, denote with D_p the unique maximal disk with center p. Recall that $M(\mathcal{A})$ is a geometric graph [11], defined as the set of centers of all maximal disks for $\mathcal{A}$. Define a *leaf* of $M(\mathcal{A})$ as a vertex with a single incident edge. There are two ways how a vertex of $M(\mathcal{A})$ can be a leaf; it is either the center of the osculating circle at an apex (a point of maximal curvature) of $\partial\mathcal{A}$, or it is a 'sharp' vertex of $\partial\mathcal{A}$. The first type will be called a *proper leaf*. Our convergence proof is based on an analysis of the behavior of the medial axis in the vicinity of proper leaves, as well as at points being sufficiently distant from proper leaves.

For a point $p \in M(\mathcal{A})$ define $\xi_p \leq \pi$ as the largest angle at p spanned by two points in the set $D_p \cap \partial\mathcal{A}$. Further, put

$$k_p = \frac{4}{1-\cos\frac{\xi_p}{2}} \quad \text{which is equivalent to} \quad \cos\tfrac{\xi_p}{2} = 1 - \frac{4}{k_p}. \tag{1}$$

When p is not a proper leaf then $\xi_p > 0$. The lemma below, which addresses the parts of the medial axis sufficiently remote from the proper leaves, does not assume any regularity

condition for the shape boundaries. A convergence result in a similar spirit is presented in [4].

Lemma 6. *Let $\mathcal{A}$ and $\mathcal{B}$ be two shapes whose (two-sided) Hausdorff distance satisfies $H(\partial\mathcal{A}, \partial\mathcal{B}) = \varepsilon$. Let D_p denote any maximal disk for $\mathcal{A}$ whose radius r_p fulfills $r_p > k_p \cdot \varepsilon > 0$. Then there exists a maximal disk D_q for $\mathcal{B}$ such that $\|p - q\| < k_p \cdot \varepsilon$.*

Proof. Let $x, y \in D_p \cap \partial\mathcal{A}$ be two points realizing the angle ξ_p, that is, $\angle xpy = \xi_p$. Let D'_p be the largest disk centered at p and contained in $\mathcal{B}$, and let $z \in D'_p \cap \partial\mathcal{B}$. Define D_q as the maximal disk with respect to $\mathcal{B}$ and containing D'_p; then clearly $z \in \partial D_q$. We consider the set of all disks D_x containing D'_p and satisfying $z \in \partial D_x$. Within this set we define D_s as the disk satisfying $\|p - s\| = k_p \cdot \varepsilon$. See Figure 7 for an illustration.

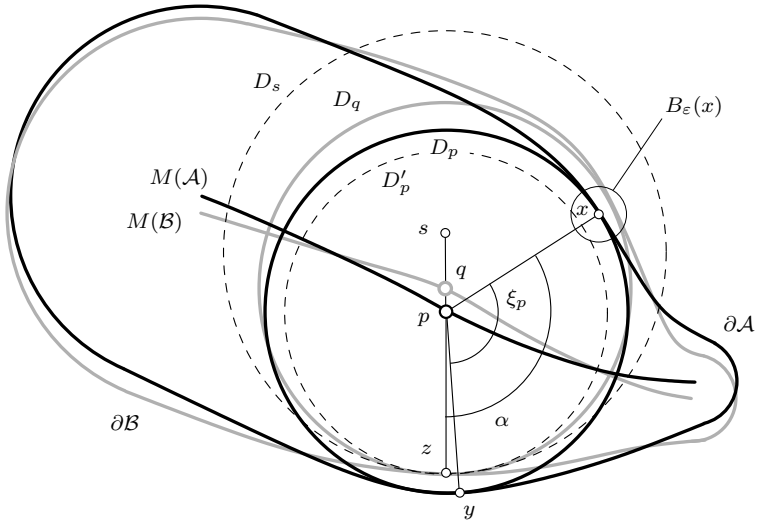

Fig. 7: Notations from the proof of Lemma 6.

Without loss of generality we assume that the angle $\alpha = \angle xpz \leq \pi$ satisfies $\alpha \geq \xi_p/2$; otherwise one may swap x and y. Due to this, and taking (1) into account, we get

$$k_p \cdot r_p \cdot \cos\alpha \leq k_p \cdot r_p \cdot \cos\tfrac{\xi_p}{2} = k_p \cdot r_p \cdot (1 - \frac{4}{k_p}) = k_p \cdot r_p - 4r_p \tag{2}$$

As we assumed that $r_p > k_p \cdot \varepsilon$, we obtain

$$k_p \cdot r_p \cdot \cos\alpha < k_p \cdot r_p - 2r_p - 2k_p \cdot \varepsilon + 2\varepsilon. \tag{3}$$

This inequality implies

$$r_p^2 + (k_p \cdot \varepsilon)^2 - 2 \cdot r_p \cdot k_p \cdot \varepsilon \cdot \cos(\pi - \alpha) < (r_p - 2\varepsilon + k_p \cdot \varepsilon)^2. \tag{4}$$

Due to $\|p-x\| = r_p$ and $\|s-p\| = k_p \cdot \varepsilon$, we thus have

$$\|p-x\|^2 + \|s-p\|^2 - 2 \cdot \|p-x\| \cdot \|s-p\| \cdot \cos(\pi-\alpha) < (r_p - 2\varepsilon + k_p \cdot \varepsilon)^2. \quad (5)$$

Finally we apply the law of cosines to get

$$\|s-x\| < r_p - 2\varepsilon + k_p \cdot \varepsilon \quad \text{or, equivalently,} \quad \|s-x\| + \varepsilon < r_p - \varepsilon + k_p \cdot \varepsilon. \quad (6)$$

On the other hand, the radius $\|s-z\|$ of D_s is at least $r_p - \varepsilon + k_p \cdot \varepsilon$, by construction. Consequently, D_s contains the ε-neighborhood of x, hence at least one point of $\partial\mathcal{B}$. This implies $D_q \subset D_s$, hence

$$\|p-q\| < \|p-s\| = k_p \cdot \varepsilon. \quad (7)$$

This completes the proof. □

Let $p \in M(\mathcal{A})$ be a proper leaf, let $x \in \partial\mathcal{A}$ be the corresponding apex, and consider some point q lying on the unique edge incident to p. Then the maximal disk D_q touches $\partial\mathcal{A}$ at two points, which are the endpoints of a segment of $\partial\mathcal{A}$ through x of length, say, ℓ. When $\ell \to 0$, then clearly $q \to p$. The following lemma describes the speed of this convergence.

Lemma 7. *If $\partial\mathcal{A}$ is piecewise analytic and C^2 in the neighborhood of x then there exists a constant $w > 0$ such that*

$$\|q-p\| \le w \cdot \ell^2.$$

Proof. For a point $y \in \partial\mathcal{A}$, let n_y denote the normal line to $\partial\mathcal{A}$ through y. Given an apex x of $\partial\mathcal{A}$, define the set

$$I_x = \{n_y \cap n_z \mid y \neq z, \;\; d(y,x) \le \ell, d(z,x) \le \ell\},$$

of intersection points of the curve normals, with distance d being measured along $\partial\mathcal{A}$. Clearly, $q \in I_x$. Using the formal Taylor expansion of $\partial\mathcal{A}$ at x it can be proved directly that $\delta(p, I_x) \le w \cdot \ell^2$ for suitable $w > 0$. □

We are now prepared to prove the claimed convergence result.

Theorem 1. *Let some shape $\mathcal{A}$ with piecewise analytic boundary $\partial\mathcal{A}$ be approximated by a sequence of shapes $\mathcal{B}_n$, where $\partial\mathcal{B}_n$ is a spline of n circular arcs produced by the spiral preserving variant of Algorithm BISECT. For the one-sided Hausdorff distance δ we have*

$$\delta(M(\mathcal{B}_n), M(\mathcal{A})) = O(n^{-1}) \quad \textit{and}$$
$$\delta(M(\mathcal{A}), M(\mathcal{B}_n)) = O(n^{-3/2}).$$

Proof. We will give the full proof only for the case of globally C^2 boundary $\partial\mathcal{A}$ which does not contain circular arcs. The proof generalizes easily to the case where $\partial\mathcal{A}$ is an arbitrary concatenation of analytic pieces, and thus, in particular, may contain sharp vertices.

As $\partial\mathcal{A}$ is assumed to be globally C^2, all leaves of $M(\mathcal{A})$ are proper leaves. For sufficiently large n, each leaf of $M(\mathcal{A})$ is also a leaf of $M(\mathcal{B}_n)$, and all leaves of $M(\mathcal{B}_n)$ are

contained in $M(\mathcal{A})$. This is because the circular arc spline $\partial\mathcal{B}_n$ preserves not only spirals, but also position, normal vector, and curvature at each apex x of $\partial\mathcal{A}$.

Let us remove from $\partial\mathcal{B}_n$ the containing circular arc b_x for each apex x whose osculating disk is contained in $\mathcal{B}_n$ (and hence is maximal for $\mathcal{B}_n$). This decomposes $\partial\mathcal{B}_n$ into components. We define d_n as the minimum of the lengths of all the removed arcs b_x. This minimum length shrinks to zero, as ∂A does not contain circular arcs, and it behaves as $\Omega(n^{-1})$ by construction of $\mathcal{B}_n$. Now define

$$\xi_n = 2\arcsin(d_n/2L), \tag{8}$$

where L denotes an upper bound on the geometric diameters of all the shapes $\mathcal{B}_n$. Apart from disks for leaves, each maximal disk D_p for $\mathcal{B}_n$ has contact to at least two different components. (Otherwise, there would be a supplementary leaf of $M(\mathcal{B}_n)$.) For such a disk D_p, we have the angle inequality $\xi_p \geq \xi_n$, provided that n is sufficiently large (due to the fact that $n \to \infty$ implies $d_n \to 0$).

Because $d_n = \Omega(n^{-1})$ and since L is a constant, we have $1 - \cos(\xi_n/2) = \Omega(n^{-2})$. Moreover, $\delta(\partial\mathcal{A}, \partial\mathcal{B}_n) = O(n^{-3})$ by construction. That is, the condition in Lemma 6 holds for all maximal disks D_p for $\mathcal{B}_n$ (with exception of finitely many proper leaves p), when n is sufficiently large. Indeed, in Lemma 6 we have $k_p \cdot \varepsilon = O(n^{-1})$. By the same lemma, this is also a bound on $\|p - q\|$ and therefore we get $\delta(M(\mathcal{B}_n), M(\mathcal{A})) = O(n^{-1})$.

The other direction can be proved similarly. For each leaf p of $M(\mathcal{A})$, with corresponding apex x of $\partial\mathcal{A}$, we define a neighborhood c_x on $\partial\mathcal{A}$ of length $n^{-3/4}$. Removal of all the segments c_x leads us to two types of maximal disks D_q for $\mathcal{A}$, depending on whether D_q touches a single segment c_x (q is then close to p), or not. For the latter type, the analysis is the same as above, and shows that q approaches the center of some maximal disk for $\mathcal{B}_n$ at speed $O(n^{-3/2})$. For the former type, due to Lemma 7, the distance between q and leaf p (which is also a leaf of $M(\mathcal{B}_n)$) behaves as $O((n^{-3/4})^2)$, i.e., the same. The one-sided Hausdorff distance $\delta(M(\mathcal{A}), M(\mathcal{B}_n))$ thus converges at that speed. □

Note that the global convergence speed of the medial axis with respect to the Hausdorff distance is $O(n^{-1})$, whereas the error of the boundary approximation improves as $O(n^{-3})$. This is due to the behavior of the medial axis close to its leaves. When we restrict ourselves to the λ-medial axis [7] for *any* $\lambda > 0$, then d_n in formula (8) becomes a constant, and the approximation speed is $\Theta(n^{-3})$ by Lemma 6. This compares favorably to using a size-m point sample on $\partial\mathcal{A}$ and pruning its Voronoi diagram, as the approximation speed is then only $\Theta(m^{-1})$.

8. Conclusions

We have given several examples for the efficient handling of shapes with nonlinear boundaries. In particular, the use of circular arcs for boundary conversion has been shown to be highly useful. Our results profit from the confluence of geometric approximation theory and computational geometry. To our knowledge, this is the first systematic approach

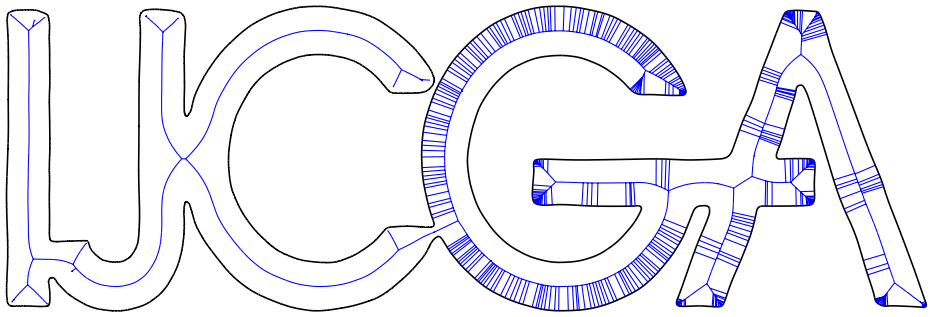

Fig. 8: Exact medial axis of circular arc boundary representation (IJC) versus polygonal boundary representation (GA).

in this direction. Compared to conversion into polylines, the gain in efficiency increases with the complexity of the subsequent algorithm. This makes affordable suboptimal (hence sometimes less complicated) algorithms.

Based on the theoretical investigations in the present paper we have developed an efficient and reliable implementation of the 2D medial axis construction; see [1] for details and empirical evaluations. An example of the obtained results is given in Figure 8. It shows the exact medial axis of a circular arc approximation as opposed to the exact medial axis of a piecewise-linear approximation.

Other approximating primitives could be considered (e.g., cubics). In our opinion though, circular arc splines yield the best trade-off. The presented algorithms, in principle, work for arbitrary primitives. In particular, in our medial axis algorithm, the added numerical complexity is not raised further by the algorithm itself. This is a nontrivial property of this algorithm, which is the first to combine practicality, efficiency, and stability [1]. Its generalization to shapes with holes is possible, as Lemma 3 has a counterpart for this case.

Finally, we raise the question of whether results of this paper can be extended to three-space.

Acknowledgements

Thanks go to Raimund Seidel for discussions on Section 6 and to Wolfgang Aigner for implementing our algorithms and producing Figure 8. We also thank two anonymous referees for their helpful comments, which improved the presentation of the paper. Research supported by the FWF Joint Research Program Industrial Geometry, subprojects S9202-N12 and S9205-N12, and by the project MSM 0021620839 of the Czech Ministry of Education.

References

1. O. Aichholzer, W. Aigner, F. Aurenhammer, T. Hackl, B. Juettler, M. Rabl. Medial axis computation for planar free-form shapes. *Computer Aided Design* **41** (2009), 339–349.
2. H. Alt, O. Cheong, A. Vigneron. The Voronoi diagram of curved objects. *Discrete & Computational Geometry* **34** (2005), 439–453.
3. D. Attali, J.-D. Boissonnat, H. Edelsbrunner. Stability and computation of medial axes – a state-of-the-art report. *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, T. Müller, B. Hamann, B. Russell (eds.), Springer Series on Mathematics and Visualization, (2008).
4. D. Attali, J.-D. Boissonnat, S. Oudot. Prototype implementation of medial axis and Voronoi diagram for sampled objects in space. ECG-TR-304109-01. `http://www-sop.inria.fr/prisme/ECG/Results/Reports.html`
5. D. Avis, G.T. Toussaint. An efficient algorithm for decomposing a polygon into star-shaped polygons. *Pattern Recognition* **13** (1981), 395–398.
6. B.K. Bhattacharya, H. El Gindy. A new linear convex hull algorithm for simple polygons. *IEEE Trans. Information Theory* **IT-30** (1984), 85–88.
7. F. Chazal, A. Lieutier. Stability and homotopy of a subset of the medial axis. *Proc.* 9^{th} *ACM Symp. Solid Modeling and Applications*, (2004), 243–248.
8. F. Chazal, R. Soufflet. Stability and finiteness properties of medial axis and skeleton. *J. Dynamical and Control Systems* **10** (2004), 149–170.
9. B. Chazelle. A theorem on polygon cutting with applications. *Proc.* 23^{rd} *IEEE Symp. FOCS*, (1982), 339–349.
10. F. Chin, J. Snoeyink, C.A. Wang. Finding the medial axis of a simple polygon in linear time. *Discrete & Computational Geometry* **21** (1999), 405–420.
11. H.I. Choi, S.W. Choi, H.P. Moon. Mathematical theory of medial axis transform. *Pacific J. Mathematics* **181** (1997), 57–88.
12. W.L.F. Degen. Exploiting curvatures to compute the medial axis for domains with smooth boundary. *Computer Aided Geometric Design* **21** (2004), 641–60.
13. T.K. Dey, W. Zhao. Approximating the medial axis from the Voronoi diagram with a convergence guarantee. *Algorithmica* **38** (2004) 179–200.
14. T.K. Dey, W. Zhao. Approximate medial axis as a Voronoi subcomplex. *Computer-Aided Design* **36** (2004), 195–202.
15. D.P. Dobkin, D.L. Souvaine. Computational geometry in a curved world. *Algorithmica* **5** (1990), 421–457.
16. S. Drysdale, G. Rote, A. Sturm. Approximation of an open polygonal curve with a minimum number of circular arcs and biarcs. *Computational Geometry: Theory and Applications* **41** (2008), 31–47.
17. G. Elber, M.S. Kim. Bisector curves of planar rational curves. *Computer-Aided Design* **30** (1998), 1089–1096.
18. I.Z. Emiris, A. Kakargias, S. Pion, M. Teillaud, E.P. Tsigaridas. Towards an open curved kernel. *Proc.* 20^{th} *Ann. ACM Symp. Computational Geometry*, (2004), 438–446.
19. I.Z. Emiris and G.M. Tzoumas. A real-time and exact implementation of the predicates for the Voronoi diagram of parametric ellipses. *Proc. 2007 ACM Symp. Solid and Physical Modeling*, 133–142.
20. I.Z. Emiris, E.P. Tsigaridas, and G.M. Tzoumas. The predicates for the exact Voronoi diagram of ellipses under the euclidean metric. *Int. J. Comp. Geometry & Appl.* **18** (2008), 567–597.
21. G. Farin. Curves and Surfaces for Computer Aided Geometric Design. *Academic Press*, San Diego, (1997).
22. G. Farin, J. Hoschek, M.-S. Kim. *Handbook of Computer Aided Geometric Design*, Elsevier, (2002).

23. M.R. Garey, D.S. Johnson, F.P. Preparata, R.E. Tarjan. Triangulating a simple polygon. *Information Processing Letters* **7** (1978), 175–179.
24. R.L. Graham. An efficient algorithm for determining the convex hull of a finite planar set. *Information Processing Letters* **1** (1972), 132–133.
25. R.L. Graham, F.F. Yao. Finding the convex hull of a simple polygon. *J. Algorithms* **4** (1984), 324–331.
26. M. Held, J. Eibl. Biarc approximation of polygons with asymmetric tolerance bands. *Computer-Aided Design* **37** (2005), 357–371.
27. S. Hertel, K. Mehlhorn. Fast triangulation of the plane with respect to simple polygons. *Information & Control* **64** (1985), 52–76.
28. K. Höllig, J. Koch. Geometric Hermite interpolation with maximal order and smoothness. *Computer Aided Geometric Design* **13** (1996), 681–695.
29. R. Klein, K. Mehlhorn, S. Meiser. Randomized incremental construction of abstract Voronoi diagrams. *Computational Geometry: Theory and Applications* **3** (1993), 157–184.
30. X. Kong, H. Everett, G.T. Toussaint. The Graham scan triangulates simple polygons. *Pattern Recognition Letters* **11** (1990), 713–716.
31. D.T. Lee. Medial axis transformation of a planar shape. *IEEE Trans. Pattern Analysis and Machine Intelligence* **PAMI-4** (1982), 363–369.
32. D.T. Lee, F.P. Preparata. Location of a point in a planar subdivision and its applications. *SIAM J. Computing* **6** (1977), 594–606.
33. D. McCallum, D. Avis. A linear algorithm for finding the convex hull of a simple polygon. *Information Processing Letters* **9** (1979), 201–206.
34. D.S. Meek, D.J. Walton. Approximating smooth planar curves by arc splines. *J. Computational and Applied Mathematics* **59** (1995), 221–231.
35. D.S. Meek, D.J. Walton. Approximation of a planar cubic Bézier spiral by circular arcs. *J. Computational and Applied Mathematics* **75** (1996), 47–56.
36. D.S. Meek, D.J. Walton. Spiral arc spline approximation to a planar spiral. *J. Computational and Applied Mathematics* **107** (1999), 21–30.
37. A. Melkman. On-line construction of the convex hull of a simple polygon. *Information Processing Letters* **25** (1987), 11–12.
38. C.J. Ong, Y.S. Wong, H.T. Loh, X.G. Hong. An optimization approach for biarc curve fitting of B-spline curves. *Computer-Aided Design* **28** (1996), 951–959.
39. R. Ramamurthy, R.T. Farouki. Voronoi diagram and medial axis algorithm for planar domains with curved boundaries I. Theoretical foundations. *J. Computational and Applied Mathematics* **102** (1999), 119–141.
40. M.A. Sabin. The use of circular arcs to form curves interpolated through empirical data points. *Rep. VTO/MS/164, British Aircraft Corporation*, (1976).
41. Z. Šír, R. Feichtinger, B. Jüttler. Approximating curves and their offsets using biarcs and Pythagorean hodograph quintics. *Computer-Aided Design* **38** (2006), 608–618.
42. X. Yang. Efficient circular arc interpolation based on active tolerance control. *Computer-Aided Design* **34** (2002), 1037–1046.
43. C.K. Yap. An $O(n \log n)$ algorithm for the Voronoi diagram of a set of simple curve segments. *Discrete & Computational Geometry* **2** (1987), 365–393.

Appendix

Let $\boldsymbol{c}(t)$ be a given analytic curve on the domain $[t_0, t_1]$ and suppose that $\boldsymbol{c}(t)$ contains neither inflections nor apices in $[t_0, t_1]$. For given step size h, consider geometric primitives $b(t, h)$ that approximate the curve segments $\boldsymbol{c}[t, t + h]$. The geometric primitives are either line segments or arcs or biarcs. Assume that the domain of $\boldsymbol{c}(t)$ can slightly be enlarged to $[t_0, t_1 + h_{\max}]$, where $h_{\max}$ is a suitable constant which specifies the largest stepsize.

In order to evaluate the one-sided Hausdorff distance from $b(t, h)$ to $\boldsymbol{c}[t, t + h]$, we analyze the stationary points $\tau = \tau_i$ of the function

$$d(\tau, t, h) = \min_{q \in b(t,h)} \|q - \boldsymbol{c}(\tau)\|, \quad \tau \in [t, t + h],$$

which are characterized by

$$\frac{\partial}{\partial \tau} d(\tau, t, h)\Big|_{\tau=\tau_i} = 0, \quad i = 1, \ldots, s(t) \; .$$

Provided that h is sufficiently small, the number $s(t)$ of stationary points is independent of t. For instance, this number is 1 for line segments and 2 for arcs interpolating three points. For each stationary point τ_i we consider the associated distance

$$d_i(t, h) = d(\tau_i(t, h), t, h)$$

where we define $d_i(t, 0) = 0$. The one-sided Hausdorff distance $\delta(b(t, h), \boldsymbol{c}[t, t + h])$ is the maximum of all these distances.

For each value of t, consider the Taylor expansion at $t = 0$. The first non-vanishing derivative is used to define the remainder term,

$$d_i(t, h) = \frac{1}{k!} \, d_i^{[k]}(t, h_i^*(h)) \cdot h^k,$$

where $[k]$ indicates the k^{th} derivative with respect to the step size h and $h_i^*(h) \in [0, h]$. The order k of this term is called the *approximation order* of the geometric primitive; it equals 2 for line segments and 3 for circular arcs.

Since the curve $\boldsymbol{c}(t)$ contains neither inflections nor apices, and due to the compactness of its domain $[t_0, t_1]$, there exist positive constants C, D such that the functions d_i satisfy

$$0 < C < d_i^{[k]}(t, 0) < D.$$

Moreover, since the k^{th} derivative is continuous, there exists a step size $g > 0$ such that

$$\forall (t, h^*) \in [t_0, t_1] \times [0, g] : \quad \frac{C}{2} \leq d_i^{[k]}(t, h^*) \leq 2D.$$

Consequently, if h is sufficiently small, then the one-sided Hausdorff distance δ satisfies

$$\frac{1}{k!} \frac{C}{2} h^k \leq \delta \leq \frac{1}{k!} 2D h^k \; .$$

This proves $\delta = \Theta(h^k)$ for the case where k is a constant.