Mitered offsets and straight skeletons for circular arc polygons

Bastian Weiß[1], Bert Jüttler[1], and Franz Aurenhammer[2]

1 **Johannes Kepler University Linz**
`bastian.weiss@jku.at`
`bert.juettler@jku.at`
2 **Graz University of Technology**
`franz.aurenhammer@igi.tugraz.at`

Abstract

We generalize the offsetting process that defines straight skeletons of polygons to circular arc polygons. The offsets and the associated skeleton are obtained by applying an evolution process to the boundary and tracing the paths of vertices. These paths define the associated patch decomposition. While the skeleton is a forest, the patches of the decomposition possess a radial monotonicity property. Analyzing the events that occur during the evolution process is non-trivial. This leads us to an event-driven algorithm for offset and skeleton computation. Several examples (both manually created ones and approximations of planar free-form shapes by arc polygons) are presented and used to analyze the performance of our algorithm.

1 Introduction

We define mitered offsets and straight skeletons for planar free-form shapes represented as circular arc polygons, see Fig. 1, which also provides a comparison with classical offsets and polygons.

Offsets and skeletons of planar shapes are widely used in shape analysis, shape design, motion planning, image processing and tool path generation. Besides classical offsets, various generalizations have been considered.

The singularities and self-intersections of offset curves are closely related to the medial axis [6], which is a particular skeleton, i.e., a structural shape descriptor. Algorithms for computing the medial axes of planar shapes are studied in several publications, see [1, 7].

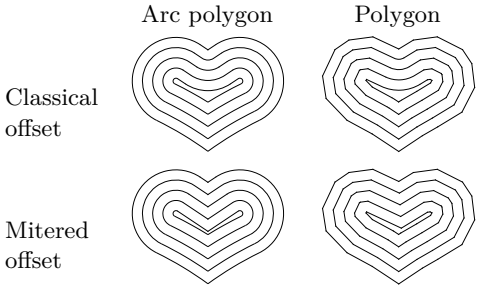


Figure 1 Comparison of classical and mitered offsets.

In the case of piecewise linear shapes, mitered offsets provide an alternative to classical offsets with enhanced shape-preserving properties around reflex vertices [9]. They are defined procedurally, by specifying the evolution of the boundary of a shape as the offset distance increases. Special attention has to be paid to topological changes of the offsets, which can be classified into events.

An evolution is used in [3, 4, 8] to define the straight skeleton of simple polygons and planar straight line graphs.

Circular arc polygons are potentially piecewise G^1 smooth and possess a better approximation order than piecewise linear boundaries [5]. Algorithms for the approximate conversion of general shapes into arc polygons are also well understood. The mitered offsets

34th European Workshop on Computational Geometry, Berlin, Germany, March 21–23, 2018.
This is an extended abstract of a presentation given at EuroCG'18. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

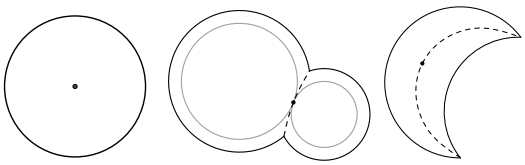
are again defined by specifying the evolution of the boundary, where the use of arcs leads to a wider variety of events.

2 Definitions

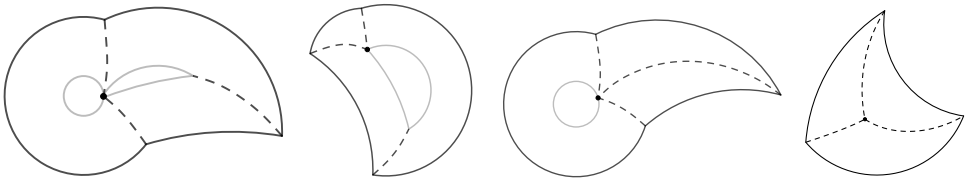
We define offsets, and consequently the skeleton, for arc polygons P as the result of an evolution of its boundary. Offsetting P means that the edges shrink or expand in radial direction (parallel for straight line segments) with constant speed towards the interior of the polygon. Simultaneously, the edges' endpoints travel on certain paths, which are determined by the evolution of the adjacent edges. This process is well-defined until we encounter (self-) intersections or the boundary becomes disconnected, see *splice event* below. We introduce events in order to obtain a globally consistent definition of trimmed offsets. More precisely, we distinguish between four classes of events.

Vanish event An edge e of P vanishes. This happens if the two endpoint vertices of e become coincident and the length of e shrinks to zero, see Fig. 4.

Terminal event These events occur if a connected component of P has three or fewer edges. Depending on the situation, these events can be grouped into seven possible types, see Fig. 2. The label $k\mathrm{Tv}\ell$ of each type depends on the number k of involved edges and ℓ of vanishing edges.



(a) Events 1Tv1, 2Tv0 and 2Tv2 (from left to right).



(b) Events 3Tv0, 3Tv1, 3Tv2 and 3Tv3 (from left to right).

Figure 2 Terminal events.

Contact event The arc polygon touches itself in its interior. There are three events of this type (Fig. 3): A *split event* occurs when a reflex vertex hits an edge of P. Two touching edges create a *squeeze event*. Finally, a *bubble event* happens when the endpoints of a shrinking edge e meet while the length of that edge is not zero.

Circular edges can become disconnected during the offsetting process. Since this situation was not encountered for straight skeletons, we need to introduce a new event:

Splice event A reflex vertex splices at the moment when the adjacent edges become tangential. The continuation of the evolution is not unique. We propose to close the gap in P by inserting a semicircle that starts to expand, thereby creating two smooth vertices, see Fig. 4. This solution ensures the property of local invertibility, which will be discussed in the next section.

Terminal events play a role in the later stages of the evolution. In these situations, it is common that three endpoints meet in a single point. While this might happen also in

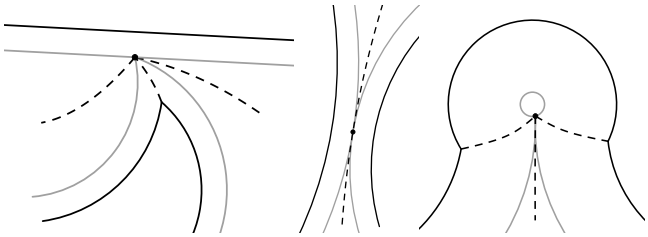


Figure 3 Split, squeeze and bubble event.

earlier stages of the evolution, we do not consider it as it has zero probability, i.e., it occurs in non-generic cases only.

Most events are of local nature, since they involve only a small number of adjacent edges. The only exceptions are split and squeeze events, which entail global modifications of the polygon's topology. Consequently we will distinguish between *local* and *global events*.

3 Properties and computation

The arc polygon P consists of N *vertices* v_i (with indices modulo N) connected by *edges* $e_i = \overline{v_i v_{i+1}}$ in counterclockwise order. Each edge has a center m_i (possibly at infinity) and a radius r_i (non-zero, possibly infinite). Positive and negative values of the radius correspond to arcs with counterclockwise and clockwise orientation, respectively. Vertices with an inner angle of π are called *smooth*. The path of a vertex v_i is a conic section and is determined by its neighbor edges. Note that all edges can be straight line segments, hence we include straight skeletons [3] as a special case.

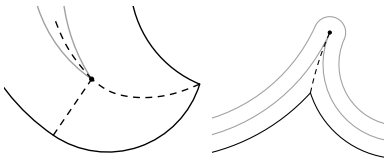


Figure 4 Vanish event (left) and splice event (right). The gray curve is the offset. The dashed lines are part of the skeleton.

Due to the fact that our offsets are defined by an evolution process there is a local invertibility: It is possible to reverse the evolution between two consecutive events. In addition, it is possible to reverse the process when contact or splice events occur. However, this does not hold for vanished edges. The local invertibility is not valid for classic offsets where we would loose convex corners as shown in Fig. 5.

▶ **Definition 3.1.** The skeleton $\mathcal{S}$ of a circular arc polygon P is defined as the path of all non-smooth vertices.

Figure 6 shows an arc polygon P (black) and its skeleton (blue). The extended skeleton is the union of $\mathcal{S}$ with the paths of all smooth vertices (green). This extended skeleton together with P separates the domain into disjoint patches, the so called *patch decomposition* $\mathcal{P}$. In Fig. 6 there is a hyperbolic part of $\mathcal{S}$ emerging from the reflex vertex. The upper part of $\mathcal{S}$ from left to right, consists of a hyperbolic arc, an elliptic arc and another hyperbolic arc, joined smoothly. The skeleton $\mathcal{S}$ is a *forest* and has, in general, several connected components. It can be shown that $\mathcal{S}$ has *linear complexity*. While this is rather obvious for straight skeletons of polygons [3], the analysis is slightly more involved for circular arc polygons, due to the presence of splice events.

Figure 5 Local invertibility.

The straight skeleton possesses a monotonicity perpendicular to the defining edge, while the patch decomposition $\mathcal{P}$ has *radial monotonicity*. That means, every radial line intersected with its patch is connected (Fig. 6, red segments).

The skeleton $\mathcal{S}$ is a generalization of straight skeletons. It is not obvious how to apply well known principles like sweep-line and divide & conquer since straight skeletons do not have a Voronoi structure [4]. The intuitive reason for this is that the velocity of a vertex is determined by its inner angle. In fact, the vertex velocity tends to infinity as the angle approaches zero. Instead, our approach is based on a simulation of the event driven evolution that defines mitered offsets.

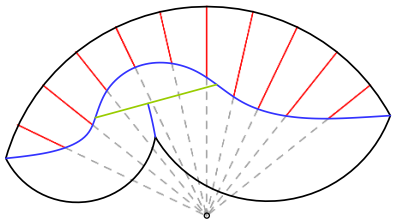


Figure 6 shows the skeleton $\mathcal{S}$ (blue) and the paths of smooth vertices in green. The radial red lines show the radial monotonicity property.

Our main data structures consists of a list E containing edges and vertices which represents P and an *event queue* Q. Events are interlinked to edges and vertices and stored in Q, ordered consecutively. The arc polygon P and its offsets are represented by patches on right circular cones in space-time, where the time has been added as third (vertical) coordinate. The patch associated with the edge e_i is denoted by c_i. A patch is represented by its boundary curves. Splice events create new edges and associated patches. Vanish and terminal events trigger the deletion of edges from the arc polygon. At this stage, the associated patch is complete.

Our algorithm takes P given as an edge list E and computes the skeleton and patch structure. Initially, the event queue Q is populated by all local events that can be computed from P. Now we loop over E as long as it is not empty. Each loop performs three steps: First, computation of the next possible global event and comparison with the next local event give us the next event e. Second, grow $\mathcal{S}$ and $\mathcal{P}$ until we reach the event. Third, the event is handled (see details below). This may trigger the insertion of new local events into Q.

The computation of events is done differently for local and global ones. All local events are computed by intersecting cones in space-time, which represent the edges, and certain planes. For instance, a splice event of v_i is computed by analyzing the intersection $c_{i-1} \cap c_i \cap p_i$, where p_i is the vertical plane defined by the points m_{i-1}, m_i.

While computing local events of edges or vertices requires knowledge about their local neighborhood only, the computation of global events is more expensive since the entire arc polygon P has to be considered. To speed up this computation, we use a sweep-line algorithm over axis–aligned bounding boxes of edges and vertices of P.

The procedure for handling an event operates on E which represents P, the event queue Q, the event e itself and the data structures holding the skeleton $\mathcal{S}$ and patches $\mathcal{P}$.

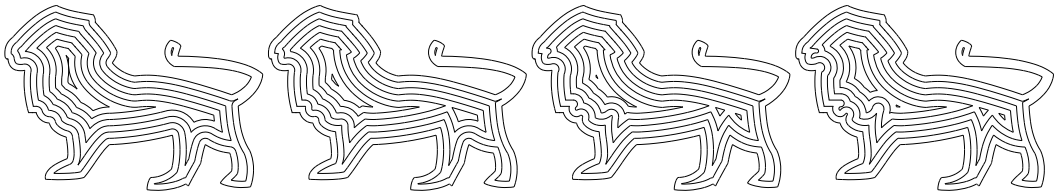
1. Add/delete/split edges.
2. Reconnect edges properly such that there is no gap in P.
3. Update S and $\mathcal{P}$.
4. Delete e from Q.
5. $Q \leftarrow Q \cup \{\text{local events for edges having a new neighbour}\}$

The splice event is necessary to close the gap of the polygon boundary if it breaks. We suggested in Section 2 to do this at the latest possible point. However, an earlier splice is achievable: The introduction of an angle σ, which is the inner angle of a reflex vertex v,

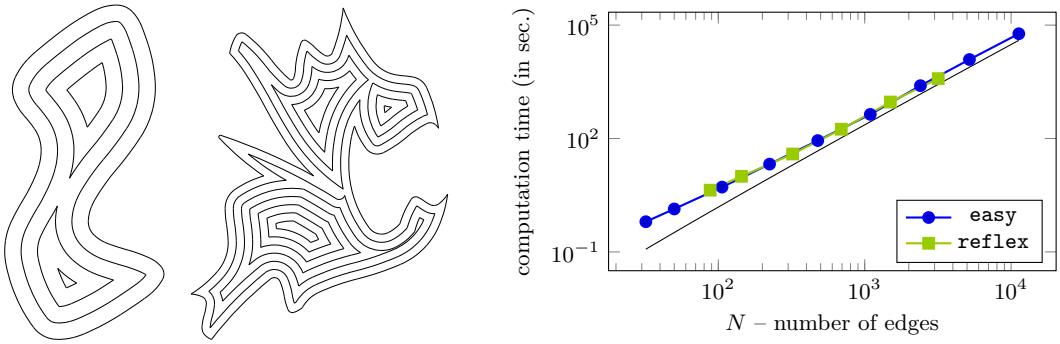
allows us to control the splice behavior of P. Choosing $\sigma = 2\pi$ results in the known splice event. An angle of $\pi < \sigma < 2\pi$ forces v to splice earlier and $\sigma = \pi$ cause each reflex vertex to splice immediately which results in classical offsets and the obtained skeleton is the medial axis. The analysis of the algorithms' complexity is still ongoing work. Experimental results are provided in the next section.

4 Examples

We perform experiments with manually designed arc polygons as well as approximations of planar free form shapes by circular arc polygons. These arc splines are created by spiral biarcs following the approach described in [2]. The results are depicted in Fig. 7.



(a) The `lion` shape for $\sigma = 180°$, $280°$, $320°$ and $360°$.



(b) The examples `easy`, `reflex` and computation times.

Figure 7 Offsets of examples `easy` and `reflex` (left and center). Computation times (right) for arc splines of different size representing these examples. As a reference, the black line indicates $\mathcal{O}(N^2 \log N)$.

Figure 7a shows offsets of the lion shape, which is represented by spiral biarcs, for various values of the splicing parameter σ. The leftmost picture visualizes the classical offset. Increasing σ delays the occurring splice events, thereby preserving the reflex vertices.

The examples `easy` and `reflex` (Fig. 7b) are used to analyze the complexity of the algorithm. Finally, we also include Fig. 8, containing straight line segments (arcs with infinite radii) on the left. The right of Fig. 8 shows the skeleton (blue) and the paths of smooth vertices (green).

The proposed algorithm has been implemented in Python 3.6 on an Intel i7-6700 CPU machine with 8 GB RAM. We demonstrate experimentally that our algorithms' complexity does not exceed $\mathcal{O}(N^2 \log N)$ in practice: Figure 7b (right) shows the computation time in seconds over the number of edges N for the `easy` and `reflex` examples. Both axes use logarithmic scales. Since the graphs depicting the computation times for the two examples seem to become tangential to the reference line (black) Fig. 7b supports the claimed complexity.

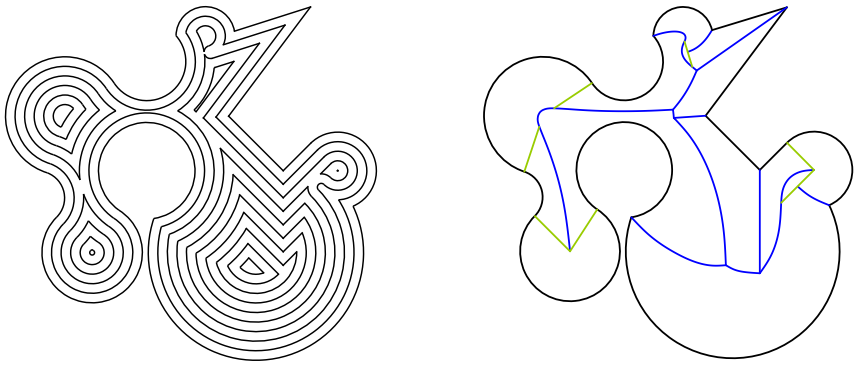


Figure 8 Mitered offsets (left) and skeleton (right, blue).

5 Conclusion

We presented an extension of straight skeletons to shapes bounded by circular arc polygons. Experimental results indicate that the proposed algorithm computes offsets and its skeleton for N edges in $\mathcal{O}(N^2 \log N)$ time in practice. The size of the resulting skeleton is linear, the patch decomposition possesses a radial monotonicity property and the offsets are locally invertible. The introduction of the parameter σ allows us to control the inner angle at which reflex vertices splice, thereby influencing the structure of the skeleton. Consequently, our algorithm can be adapted to compute either the medial axis or the straight skeleton (for polygons with only straight edges) as special cases. Finally we note that the algorithm can be extended to general circular arc figures in the plane.

References

1 O. Aichholzer, W. Aigner, F. Aurenhammer, T. Hackl, B. Jüttler, E. Pilgerstorfer, and M. Rabl. Divide-and-conquer for voronoi diagrams revisited. *Computational Geometry: Theory and Applications*, 8(43):688–699, 2010.

2 O. Aichholzer, W. Aigner, F. Aurenhammer, T. Hackl, B. Jüttler, and M. Rabl. Medial axis computation for planar free–form shapes. *Computer-Aided Design*, 41(5):339–349, 2009.

3 O. Aichholzer and F. Aurenhammer. Straight skeletons for general polygonal figures in the plane. In *International Computing and Combinatorics Conference*, pages 117–126. Springer, 1996.

4 O. Aichholzer, F. Aurenhammer, D. Alberts, and B. Gärtner. A novel type of skeleton for polygons. In *The Journal of Universal Computer Science*, pages 752–761. Springer, 1996.

5 O. Aichholzer, F. Aurenhammer, T. Hackl, B. Jüttler, M. Rabl, and Z. Šír. Computational and structural advantages of circular boundary representation. *Int. J. Comput. Geom. Appl.*, 21(1):47–69, 2011.

6 H. Blum. A transformation for extracting new descriptors of shape. In Weiant Wathen-Dunn, editor, *Models for the Perception of Speech and Visual Form*, pages 362–380. MIT Press, Cambridge, 1967.

7 D.T. Lee. Medial axis transformation of a planar shape. *IEEE Transactions on pattern analysis and machine intelligence*, pages 363–369, 1982.

8 Peter Palfrader and Martin Held. Computing mitered offset curves based on straight skeletons. *Computer-Aided Design and Applications*, 12(4):414–424, 2015.

9 S.C. Park and Y.C. Chung. Mitered offset for profile machining. *Computer-Aided Design*, 35(5):501–505, 2003.