Approximation of an Open Polygonal Curve with a Minimum Number of Circular Arcs and Biarcs

R. L. Scot Drysdale[1], Günter Rote[2], and Astrid Sturm[3]

[1] Department of Computer Science, Dartmouth College
scot@cs.dartmouth.edu
[2] Institut für Informatik, Freie Universität Berlin
rote@inf.fu-berlin.de
[3] Department of Computer Science, Free University Berlin
sturm@inf.fu-berlin.de

Abstract. An algorithm for approximating a given open polygonal curve with a minimum number of circular arcs is introduced. In computer-aided manufacturing environments, the paths of cutting tools are usually described with circular arcs and straight line segments. Greedy algorithms for approximating a polygonal curve with curves of higher order can be found in the literature. Without theoretical bounds it is difficult to prove anything about the quality of these algorithms. We present an algorithm which allows us to build a directed graph of all possible arcs and look for the shortest path from the start point to the end point of the polygonal curve. We can prove a runtime of $O(n^2 \log n)$ for an original polygonal chain with n vertices. Using the same approach, we can prove a runtime of $O(n^2 \log^2 n)$, for computing a tangent-continuous approximation with the minimum number of biarcs, for a sequence of points with given tangent directions.

1 Introduction

In computer-aided manufacturing environments, tool paths are usually made of line segments and circular arcs [3–5]. Approximating the data by curves of higher order [1–8] has been investigated extensively in the past. In contrast to approximation by *polygonal* curves, the theoretical bounds of these problem are not so well studied. In the first part of this paper we will introduce the basic ideas and the algorithm for approximation of a polygonal curve with the minimum number of circular arcs. Building on this we later present an algorithm for tangent-continuous approximation of an open polygonal curve with minimum number of biarcs.

Results and Techniques. We assume that the problem is given in the form of a tolerance region around the given curve, which is split into subregions by *gates* through the given points, see Figure 1. The precise formulation is given below.

The main idea for the optimal approximation by circular arcs (Sections 2–5) is the use of a Voronoi diagram of the tolerance boundary. We have to incrementally maintain one cell in this Voronoi diagram of line segments. Geometric

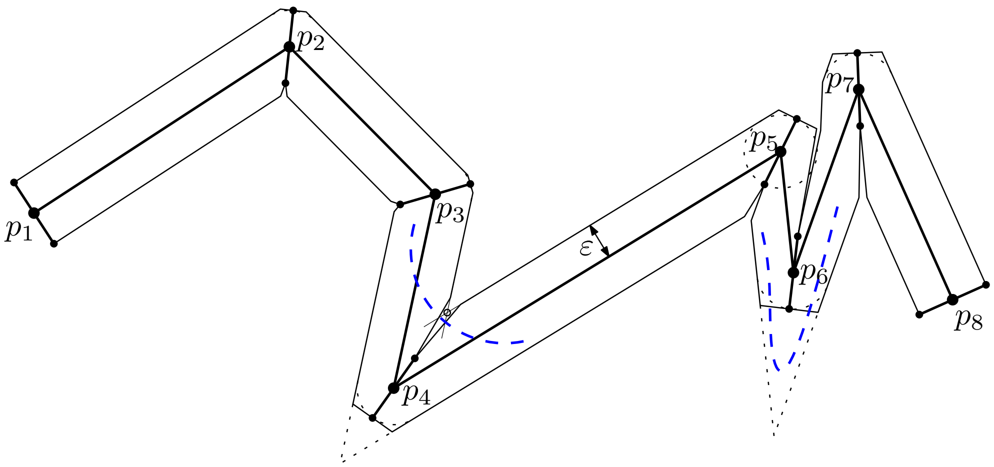


Fig. 1. Polygonal tolerance region R with gates

considerations (Lemma 7) make the location step in the update easy, leading to constant amortized time per insertion. In total, the algorithm takes $O(n^2 \log n)$ time.

We also obtain an optimal tangent-continuous approximation with *biarcs*, pieces consisting of pairs of circular arcs, with given tangent directions in $O(n^2 \log^2 n)$ time (Section 6). Here, the core of the algorithm is a reduction to circular ray shooting in simple polygons.

Problem Setting. We wish to approximate a polygonal chain $P = (p_1, \ldots, p_n)$ by a series of circular arcs (which could include straight line segments, as circles of infinite radius). The endpoints of the arcs are vertices of P. We want our approximating curve to have distance at most ε from P. As a first approximation to this problem, one can look at a region formed from strips of width ε centered at the polygon edges. However, in the vicinity of sharp corners, this does not guarantee that the curve remains close to the given points. Figure 1 shows a circular piece of a hypothetic curve that can shortcut the bend at p_4 if it is only required to remain in the strips. (Also, it might overshoot the bend, as indicated in the vicinity of p_6, although this looks like a theoretical possibility only.) To avoid this, we introduce a *gate* at every vertex. The approximating curve is required to pass through all gates in succession, and the curves are not allowed to pass through a gate twice. This will guarantee that any curve into a point p_i can be joined with any curve out of p_i without danger of an intersection other than at p_i.

For our problem, we assume that we are given a polygonal "tolerance region" R and a sequence of gates $g_1, g_2, \ldots, g_n$, which are segments through the points p_i. We will refer to endpoints of gates lying to the left of P as we walk from p_1 to p_n as left endpoints and the other endpoints as right endpoints. We require that the gates do not cross. We require that the input satisfies the following assumptions:

(A) R is a simple polygon passing through all gate endpoints;

(B) R does not intersect the interior of gates or cross the segments connecting corresponding endpoints of successive gates.
(C) No line through two points on successive gates g_i and g_{i+1} crosses the portion of R connecting g_i with g_{i+1}.

(Assumption (B) is actually a consequence of (C).) Ideally, the gate g_i at vertex p_i is a line segment of length 2ε centered at p_i that bisects the angle $p_{i-1}p_ip_{i+1}$. For a convolved curve with sharp bends close together, we might have to shorten the gates and to reduce the ε-strip around the edges, as shown in the right part of Figure 1.

Modeling the curve approximation problem by an appropriate tolerance region with gates is a problem of its own, which we do not discuss here. Eibl and Held [1, 2] have methods that can be adapted to produce such gates and tolerance regions. In Figure 1, we have chosen to approximate the "ideal" circular boundary at the outer angle of each vertex by a single edge of P. One can use more edges to get a finer approximation, or one could also choose to approximate the circular arc from inside, to get a guaranteed upper distance bound of ε. Our time bounds assume that R has constant complexity between successive gates and thus the total size of R is proportional to n.

Definition 1 (proper gate stabbing). *A circular arc stabs gates* $g_i, \ldots, g_j$ *properly, if:*

1. *the curve passes through gate* $g_m \in \{g_i, \ldots, g_j\}$ *from the side of* $\overline{p_{m-1}p_m}$ *to the side of* $\overline{p_m p_{m+1}}$
2. *the circle on which the arc lies intersects each gate only once.*

Condition 2 of this definition is stronger than what would be strictly necessary: an arc from p_i to p_j might intersect each intermediate gate only once, but the continuation of the arc beyond p_j might bend back and intersect, say, g_j and g_{j-1} a second time. This would be a sensible arc, but it is excluded by our definition. But such a situation can only happen if the gates are very close together (relative to their length).

Definition 2 (valid circular arc). *A circular arc* a_{ij} *with starting point* p_i *and endpoint* p_j *is a valid arc if:*

- *the arc stabs the gates* $g_{i+1}, \ldots, g_{j-1}$ *properly,*
- *the arc does not cross the boundary of the tolerance region* R*.*
- *the arc reaches* p_i *from the correct side of* g_i *and reaches* p_j *from the correct side of* g_j*.*

Note that because R passes through the gate endpoints, any arc that goes through a series of gates without crossing the tolerance boundary must go through them in the correct order, so we do not need to test this separately.

We can split the problem of determining if a valid circular arc connects p_i with p_j into three parts. First, we compute the set of all arcs between p_i and p_j that stab all intermediate gates properly (Sect. 2). Second, we compute all arcs that start at p_i and end at p_j, reaching both from the correct side (Sect. 3). Third, we compute all arcs between p_i and p_j that do not intersect with the tolerance boundary (Sect. 4). A valid circular arc has to be member of all three result sets.

2 Stabbing through the Gates

Definition 3 (point/gate bisectors). *Given a point p and a gate g, let b_l be the bisector of p and g's left endpoint and b_r be the bisector of p and g's right endpoint.*

Lemma 1. *The centers of all circles passing through a vertex p and intersecting a gate g exactly once lie in a double-wedge whose boundary is b_l and b_r. The sections we want are the ones where one half plane includes p and the other excludes it. (Figure 2 illustrates this.) In the degenerate case where b_l is parallel to b_r one wedge is the strip between the bisectors and the other wedge is empty.*

Proof. Consider the intersection of the half plane bounded by b_l that includes p and the half plane bounded by b_r that excludes p. Points in the interior of this region are closer to p than the right endpoint of the gate and are also closer to the left endpoint than to p. Disks centered in this region which have p on their boundary include the left endpoint and exclude the right endpoint of the gate. Therefore all circles centered in the wedge intersect the gate exactly once. The case for the second wedge is symmetric. This argument works for the degenerate case, also, but in this case all circles will include the nearer gate endpoint and exclude the further one. Thus one wedge must be empty.

Centers of circles that are located in the same region as p outside of the double-wedge are always closer to p then to the endpoints of the gate. Therefore these circles exclude the endpoints if they pass through p. These circles can not intersect the gate only once, unless the circle is tangent to the gate. Looking at the other side of the double-wedge boundary, all centers of circles located here are closer to the endpoints of the gate than to p. Each disk which includes p has to include the endpoints and its boundary does not intersect the gate at all. □

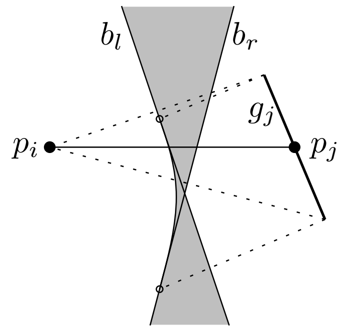


Fig. 2. The region of all centers of circles passing through p_i and gate g_j. The circles with centers close to the intersection of b_l and b_r, in the region with the curved boundary, intersect g_j twice and are not considered as centers of valid arcs.

By Definition 1, an arc stabs the gates properly only if every gate is intersected only once. Therefore the centers of circular arcs stabbing an intermediate gate are located in the double wedge of the gate. For the first and last gates of the arc we insist that the arc goes through the original point located at the gate. Thus the first and last gates are treated differently from the intermediate gates (see Section 3).

According to Lemma 1, one wedge is the region of the centers of disks including the left endpoint of the gate and excluding the right endpoint. Circular arcs centered in this region pass the gate from the correct side, according to the stabbing condition, if they are in CCW (counter-clockwise) orientation. In CW (clockwise) orientation, the arc would walk around the left endpoint before intersecting the gate. The unbounded part of this wedge lies to the left of P. Symmetrically the circular arcs in the other wedge need CW orientation to pass the gate in the correct direction, and the unbounded part of this wedge lies to the right of P.

So from now on we talk about the left wedge and the right wedge. A circular arc stabbing through the gates cannot change its orientation.

Lemma 2. *A circular arc a starting at a point p stabs gates $g_i, \ldots, g_j$ properly if and only if its center lies in the intersection of the left wedges defined by p and the gates, or the intersection of the right wedges.* □

Lemma 3. *Incrementally computing the two regions of centers of all valid circular arcs passing through a point p and stabbing all $g_i, \ldots, g_j$ gates properly can be done in $O(n \log n)$ time.*

Proof. It is the incremental intersection of $O(n)$ half-planes. □

3 Arc Endpoints

A valid circular arc from p_i to p_j must reach each endpoint from the correct side of its gate. All arcs that start at p_i and end at p_j have their centers on the bisector of the segment connecting p_i and p_j. We can go around each circle whose center is on the bisector in a CW or CCW direction. When the circle is tangent to the gate, both directions are possible. For other circles the arc in one direction will approach the gate from the correct side and the arc from the other direction will approach the gate from the wrong side. This leads to the following characterization of the desired arcs.

Lemma 4. *Let b be the perpendicular bisector of the segment between p_i and p_j. Let s_i be the point of b which is the center of a circle tangent to g_i at p_i, and let s_j be defined symmetrically. The centers of all CW arcs that reach both p_i and p_j from the correct side are all points on b to the right of both s_i and s_j. CCW arcs are symmetric.* □

4 Tolerance Boundary

The tolerance boundary R consists of two polygonal chains, one on each side of the original polygonal chain P. For a CW arc we will only check that it does not cross the boundary on the *left* side of P. It cannot cross the boundary on the right side of P if it passes through all gates, by assumption (B), and therefore we need not check for such an intersection explicitly. (For a CCW arc, the situation is symmetric.)

A circle passing though point p does not intersect or contain any edge on a polygonal chain C if its center lies closer to p than to any point on C. That is, if we compute the Voronoi diagram of $C \cup p$, the center of the circle must lie in point p's region, $V(p)$.

This is not quite the condition that we want, namely that a circular arc does not cross chain C. The Voronoi region guarantees that an entire circle does not cross C. However, in our case these are equivalent.

Lemma 5. *If an arc from g_i to g_j does not intersect a tolerance boundary between g_i and g_j then neither does the circle on which that arc lies.*

Proof. Look at the arc between consecutive gates g_k and g_{k+1}. Let q and q' be the intersection points with these gates. By assumption (C), the line ℓ through q and q' does not intersect the tolerance boundary between g_k and g_{k+1}, i.e., the tolerance boundary lies entirely on one side of ℓ. For a CW arc, the tolerance boundary in question lies on the left side of ℓ. On the other hand, ℓ is the line that splits the circle into the arc from q to q' (on the left side) and into the opposite part which is not used. Thus the part of the circle which is not used can never intersect the relevant part of the tolerance boundary. □

While we could compute the entire Voronoi diagram of $C \cup p$ to determine $V(p)$, this would be too expensive. Fortunately, we can iteratively add n consecutive segments of C and update p's Voronoi region $V(p)$ in $O(n)$ total time.

Voronoi regions are "generalized star shaped". This means that a shortest segment from a boundary point to a nearest point in the shape defining the region lies entirely within the region.

Lemma 6. *Each segment added will either cause no change to $V(p)$ or will replace a section of $V(p)$ by at most three new segments* (*two straight lines and a parabola*). (*If $V(p)$ is unbounded we think of an edge "at infinity" connecting the two infinite rays, so that these three "segments" are considered consecutive.*)

Proof. Suppose that the boundary between $V(p)$ and $V(S)$ for some open segment S consisted of two disconnected pieces. Draw a shortest segment (or ray for the piece "at infinity") from each disconnected piece to S. Because Voronoi regions are generalized star shaped, both of these segments lie within $V(S)$ and cannot be crossed by another Voronoi region. S itself cannot be crossed by another Voronoi region. This means that some segment S' must lie within the region bounded by S, the two segments, and the boundary of $V(p)$ between the two segments. But such an S' would then be disconnected from the rest of the chain, a contradiction. □

There are two parts to updating p's Voronoi region $V(p)$ when adding a segment S to the diagram. First, we find a place on the boundary of $V(p)$ that is equidistant from p and S, if such a place exists. If so, we walk around the boundary of $V(p)$, eliminating boundary sections until we reach the other place on the boundary where p is equidistant from S. (Note that either of these places could be "at infinity".)

The second part is easy — walk around the boundary of $V(p)$ from the starting point, eliminating obsolete bisector segments until you get to the finish point.

Because C is a polygonal chain, the first part is also easy. $V(p)$ is bounded by bisector pieces between p and a subset of the segments in C. Of the segments in this subset, there is a first segment F and a last segment L, according to the order along the chain.

Lemma 7. *If $V(p)$ changes, then its boundary with either $V(F)$ or $V(L)$ must change.*

Proof. The intuition is, if you can't go through the chain C, then the only way to get to $V(p)$ is through $V(F)$ or $V(L)$.

If the chain from F to L consists of only F and L (which could be the same segment), the lemma is trivially true. Otherwise consider the union of the chain C between F and L exclusive, the boundary of $V(F)$ from the endpoint it shares with the next segment on C to the end of its boundary with $V(p)$, and the boundary of $V(L)$ from the endpoint it shares with the segment before it on C until the end of its boundary with $V(p)$. If $V(p)$ is bounded these two boundaries end at the same point - the point where $V(p)$, $V(F)$, and $V(L)$ meet. If $V(p)$ is unbounded then its boundaries with $V(F)$ and $V(L)$ end in infinite rays. In either case, this union separates the plane into two parts, one including p (the inside) and the other not including p (the outside). We will call this union the separator. Note that F and L are defined to lie outside of this separator (except for the endpoint that is part of the separator).

Suppose that a segment S is added that changes $V(p)$. The previous segment on C is either L or some segment that did not modify $V(p)$. In either case, the endpoint shared with that previous segment is outside of the separator, so we know that at least part of S lies outside of separator.

If S crosses the separator, then it cannot cross C, because the chain is simple. If it crosses the Voronoi boundary of $V(F)$ then the part of the boundary between the crossing point and the end of the boundary between $V(p)$ and $V(F)$ will be eliminated. A similar argument holds for L. Thus if S crosses the separator then the boundary of $V(p)$ with either $V(F)$ or $V(L)$ must change.

If S does not cross the separator, pick some point q that is on the boundary of the new $V(p)$ that was not on the boundary of the old $V(p)$ and let E be the shortest segment from q to a point on S. E must lie entirely in $V(S)$ and must cross the separator. It cannot cross C. The rest of the analysis is exactly as in the paragraph above, with E replacing S. □

Lemma 8. *For a fixed gate g_i, we can compute the centers of all circular arcs that pass between g_i and each gate g_j without crossing the tolerance boundary in $O(n)$ time.*

Proof. Incrementally add segments from C and amortize the update time. We have shown that the centers of CW [CCW] arcs are the region of $V(p_i)$ in the Voronoi diagram of p along with the CW [CCW] boundary between g_i and g_j. We can compute these regions incrementally. It takes constant time to test if segment S changes the boundary between p and either F or L, so the total time for finding starting points is $O(n)$.

Walking along the boundary of $V(p)$ will take time proportional to the number of pieces eliminated. Because an eliminated piece is removed and never reappears, the total time for this step in all n insertions is bounded by the number of boundary pieces added. This is at most $3n$, because a bisector curve between p and a segment consists of at most three pieces. Thus this requires time $O(n)$. □

5 Computing the Shortest Path

To determine the shortest path from the start point to the end point of the polygonal curve we can build a directed graph of all possible valid arcs and do a BFS to find the shortest path from p_1 to p_n. The following theorem summarizes how to find the valid arcs from p_i to p_j.

Theorem 1. *A point c is the center of a valid CW circular arc from p_i to p_j if and only if it is in the intersection of:*

- *the intersection of the right wedges between p_i and each of the gates g_{i+1} through g_{j-1};*
- *the region of $V(p_i)$ in the Voronoi diagram of p_i and all of the segments on the left boundary between g_i and g_j; and*
- *all points on b to the right of both s_i and s_j, where b, s_i, and s_j are as defined in Lemma 4.*

The conditions for valid CCW arcs are symmetric. □

We find the possible arcs from a point p_i to all points further along P incrementally. We maintain the intersection of the right wedges, the intersection of the left wedges, the Voronoi region of p_i with the left boundary, and the Voronoi region of p_i with the right boundary. At each step we update each of the four items. We intersect each bisector ray with an intersection of wedges and with a Voronoi region, and then test if the two intersections overlap. Because wedge intersections and $V(p_i)$ are convex these intersections require $O(\log n)$ time.

Note that we can quit early as soon as both wedge intersection regions become empty. This may lead to a better behavior of the algorithm in practice than the worst-case time bound proved in the theorem below.

Theorem 2. *Given an open polygonal curve $P = (p_1, \ldots, p_n)$, a polygonal tolerance boundary of size $O(n)$, and a gate for each p_i, we can approximate P by a minimum number of valid circular arcs in $O(n^2 \log n)$ time.*

Proof. Each starting point takes $O(n \log n)$ time. □

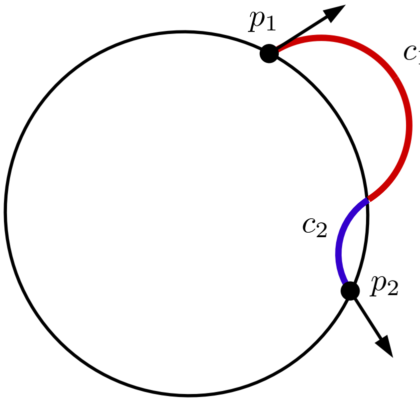


Fig. 3. The joint circle, and an S-shaped biarc with both tangents pointing outside the joint circle

6 Biarcs

The sequence of arcs produced in the previous algorithm may have arbitrary corners at the vertices. In many situations, a smooth curve is desired. We now assume that an oriented tangent direction t_i is specified for each vertex p_i of the open polygonal curve. (If such tangent information is not available, it can also be computed from the point data alone, using various tangent estimation methods.)

Our algorithm will select a subsequence of the input points and interpolate between them smoothly by *biarcs*, pieces consisting of pairs of circular arcs, respecting the tangent directions t_i at the points which are used. Our algorithm will find such an approximation with the minimum number of biarcs given a set of gates and a tolerance region. In this setting the gates would ideally be perpendicular to the tangent directions t_i, but we do not require this. We do require that the gates are chosen so that each tangent t_i passes through gate g_i in the same direction as the original polygonal chain P. Otherwise our requirements on gates and the tolerance region R are the same as for arcs.

Again we first find all valid biarcs and then build the directed graph of these biarcs from the start point to the end point of the polygonal curve. The last step is the computation of the shortest path as described in the previous section. The difference between the two algorithms is the computation of the valid arcs/biarcs. Therefore we will now focus on the computation of valid biarcs.

Biarcs. Biarc curves were introduced by Bolton [12] and are used for curve approximation in a tangent-continuous manner. A biarc consisting of two circular arcs that share an endpoint with a common tangent. This common endpoint is called the *joint* of the biarc. Given two points p_i and p_j with two tangent vectors t_i, t_j at these points, a biarc B_{ij} between p_i and p_j is characterized in the following way [2, 12]:

– B_{ij} consists of two consecutive circular arcs, a_1, a_2

– a_1 is an oriented arc from p_i to point p_{joint} and a_2 is an oriented arc from p_{joint} to p_j;
– a_1 matches the tangent vector t_i at the point p_i and a_2 matches the tangent vector t_j at p_j;
– both arcs have a common tangent at p_{joint}, the joint of the biarc.

These conditions leave one degree of freedom. The locus of possible joints forms a circle that passes through p_i and p_j [9–11], see Figure 3. For each point on this *joint circle*, there is a unique biarc which uses this point as the joint. (There are some degenerate cases, which we ignore in the sequel: as a limiting case, the joint could be one of the points p_i or p_j: the joint circle might be a line; if there is a circle through p_i and p_j with the given tangents, this is the joint circle, but all joints on this circle lead essentially to the same biarc.)

Proposition 1. *One circular arc of the biarc lies outside the joint circle, and the other lies inside the joint circle, except for their endpoints, which lie on the joint circle. Both tangents t_i and t_j point to the same side* (*either inside or outside*) *of the joint circle, and they form equal angles with the joint circle.* □

(In fact, the last property characterizes the joint circle.)

Definition 4 (Valid biarc). *A* valid biarc *B_{ij} from p_i to p_j consists of two circular arcs a_1 and a_2 and satisfies the following conditions*:

– *a_1 matches t_i at the point p_i, a_2 matches t_j at p_j, and they meet at a point on the joint circle.*
– *a_1 and a_2 stay inside the tolerance boundary.*
– *a_1 and a_2 intersect the gates g_i and g_j only once.*

The joint, which is the ending point of a_1 and the starting point of a_2, is not required to be an original point. The joint must of course lie inside the tolerance region. Note that we relaxed the gate stabbing condition. The arcs a_1 and a_2 are allowed to intersect the gates of the starting and ending points only once, but all intermediate gates can be intersected twice. The restrictions on intersecting g_i and g_j guarantee that successive biarcs will not intersect except at endpoints.

For each possible starting point p_i of a biarc, the tangent direction t_i is fixed. The pencil of circular arcs starting in this direction forms a “circular visibility region” W_i inside the feasible region R, see Fig. 4. The tangent ray splits the visibility region into CW and CCW visibility regions ($W_{i_{CW}}, W_{i_{CCW}}$).

To find a valid biarc that starts at p_i and ends at p_j we need to reach a point on the joint circle via a valid arc from p_i and then continue via a valid arc to p_j. The possible joints from the perspective of p_i are the intersection of the joint circle and the circular visibility region of p_i, consisting of $W_{i_{CW}}$ and $W_{i_{CCW}}$. By reversing the direction of the second arc and tangent we can make computing the second arc symmetric to computing the first. We will use arc $\tilde{a}_2$, which has opposite orientation and whose tangent at p_j is $\tilde{t}_j$, the reverse of t_j. We will call this circular visibility region $\tilde{W}_j$ ($\tilde{W}_{j_{CW}}, \tilde{W}_{j_{CCW}}$). Our goal is to find all points on the joint circle which are in both W_i and $\tilde{W}_j$ of the biarc class.

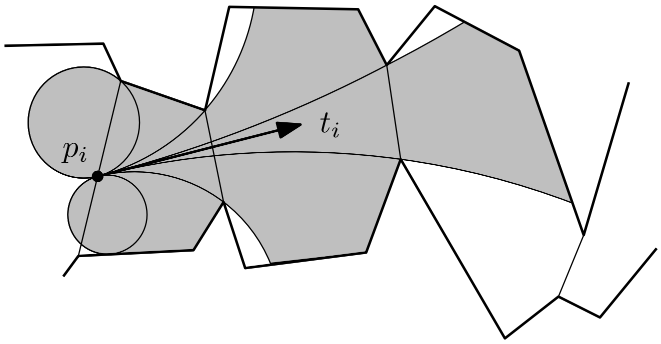


Fig. 4. A circular visibility region W_i

As a first step in this process we determine the portion of each gate that is within W_i for each point p_i and the portion of each gate that is in $\tilde{W}_j$ for each point p_j. Our second step is to determine a *joint interval* region for each pair of points p_i, p_j and biarc class. This is the part of their joint circle which might be visible from both endpoints; it depends on the orientation of the arcs and their position to the joint circle. Finally, we compute for each joint interval its intersection with the corresponding W_i and $\tilde{W}_j$. This step uses the precomputed information about the intersection of circular visibility regions with gates.

We could compute the intersection of W_i with all later gates by using intersections of wedges and Voronoi regions, as we did before. However, because we know the tangent at p_i we can do this more efficiently by computing W_i directly. The pencil of circular arcs consists of an interval of possible curvatures. As we walk along the left and right tolerance boundaries the interval of curvatures either remains unchanged or shrinks, but it always remains a single interval.

Lemma 9. *For a given point p_i, the oriented circular visibility regions of W_i and their intersection with and all gates can be computed in $O(n)$ time. The part of a gate that is visible in $W_{i_{CW}}$ or $W_{i_{CCW}}$ is at most two segments.*

Proof. We cut each oriented visibility region into two pieces, forward and backward visibility. The forward visibility region is the part of the visibility region which is reached by portions of arcs that do not intersect any gate twice. The backward visibility is the part reached by portions of arcs after they have intersected a gate twice, so they are moving backwards through the gates. We walk along the left and right boundaries of the tolerance region, determining the intersection between each boundary and a part of the pencil of arcs and computing the forward visible region. When the last reachable gate is known, we can compute for each gate moving backwards the arcs that build the backward visibility region. The backward part of the visibility region for a gate g_i consists of the arcs that intersect gate g_{i+1} twice (possibly after passing through even higher-numbered gates twice) and reach back to g_i, plus the arcs that don't cross g_{i+1} but intersect the gate g_i a second time. These arcs correspond to a connected piece of the pencil of arcs and we need to determine the intersection of this pencil part with the corresponding boundary of the tolerance region moving from g_{i+1} to g_i. Because the complexity of the tolerance boundary between two gates is constant we can do the forward and backward visibility computations between two gates in constant time, so the total time required is $O(n)$. □

Note that the interval on a gate reachable by forward portions of arcs is disjoint from the interval reachable by backward portions of arcs, because a given point is reachable by exactly one arc leaving p_i with tangent t_i. These regions (if non-empty) may join at the point where an arc is tangent to the gate, but if this arc is invalid (because it intersects the boundary) the regions will be separated. See Fig. 5.

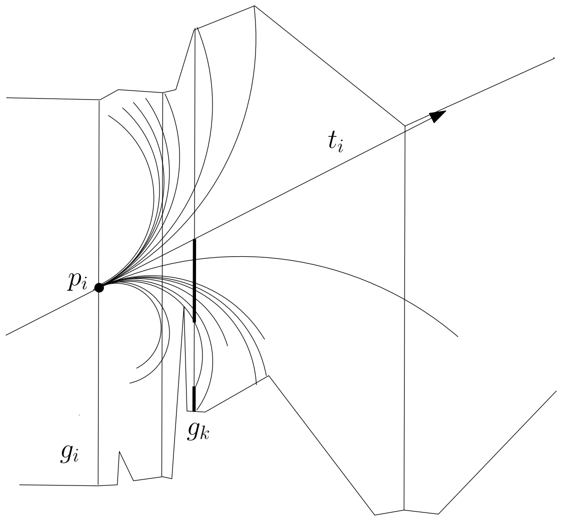


Fig. 5. Forward and backward visibility segments of region $W_{i_{CW}}$ on gate g_k

For the rest of the paper we will deal with one particular situation: The joint is on the CW arc of the joint circle from p_i to p_j. The first arc a_1 of the biarc is CW and second arc a_2 is CCW, and the two tangents point outside the joint circle. This is the case illustrated in Figure 3. The other cases can be treated analogously.

All valid arcs a_1 lie outside the joint circle and all valid arcs $\tilde{a}_2$ lie inside the joint circle. Thus, we define the potentially reachable region of the joint circle from p_i to be the part of the joint circle between p_i and the first intersection with the left tolerance boundary when going CW. This is the part of the joint circle that can be reached from p_i while staying outside the joint circle and intersecting g_i once. (Note that points on the joint circle before p_i may be in the circular visibility region of p_i, but are only reachable by intersecting g_i twice.)

Analogously, we define the potentially reachable region of the joint circle from p_2 to be the part of the joint circle between p_2 and the first intersection with the right tolerance boundary when going CCW. We define the *joint interval* to be the intersection of these potentially reachable regions of the joint circle, see Figure 6. Note that by the way it is defined the joint interval cannot cross the tolerance boundary.

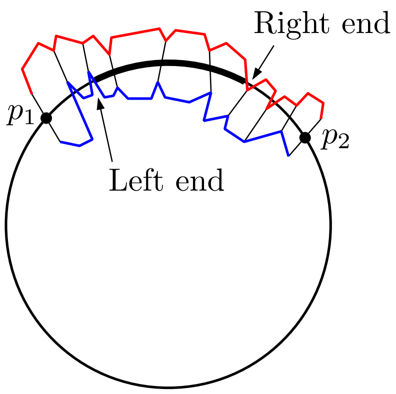
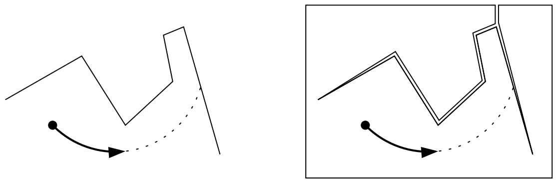


Fig. 6. Joint interval for an S-shaped (CW,CCW) biarc

Lemma 10. *The joint interval is either empty or is a single arc of the joint-circle which lies inside the tolerance region, and is the only place where valid joints can occur.* □

The intersection of the joint circle with the tolerance boundary can be found by circular ray shooting with the joint arc. For the circular ray shooting we use the algorithm introduced by Cheong et al. [13], that performs ray shooting in a simple polygon. Their algorithm requires $O(n \log n)$ preprocessing time and space and $O(\log^2 n)$ time to shoot a ray. We need to shoot a circular ray against one boundary, i.e., against a polygonal chain, as in Fig. 7a. Conceptually, this can be reduced to ray shooting inside a simple polygon by doubling the path and enclosing it in a bounding rectangle as illustrated in Fig. 7b. Of course, one can also adapt the circular ray shooting algorithm to handle open chains directly.



(a) (b)

Fig. 7. circular ray shooting

All circular ray shootings needed will take place in these two polygons. Each computation of a joint interval takes the time for two ray shootings, which is $O(\log^2 n)$, where n is the number of points of the polygon.

Once we know the joint interval, we know between which gates the joint of valid biarcs will be, if it exists.

Lemma 11. *The intersection of $W_{i_{CW}}$ with the joint interval is one connected arc.*

Proof. If two arcs a_1 and a'_1 starting from p_i reach two points on an arc of the joint circle, every point between these points can be reached by an arc "between" a_1 and a'_1, and all these arcs are within the feasible region R. □

We need to establish exactly where the joint interval enters and leaves parts of the circular visibility region from p. As we have only arcs with one orientation in the circular visibility region the circular visibility region intersects the gates in at most two segments. At any gate we can test whether the joint interval of the joint circle intersects the gate to the right of, within, or to the left of the segment (or segments in the case of two) of the gate reachable by valid arcs from p. The fact that the joint circle passes through the circular visibility region in a monotone order allows us use binary search to find pairs of gates between which a transition occurs. A transition occurs if the joint circle moves from outside to inside or inside to outside of the circular visibility region. Once such a pair of gates is found it is a constant-time operation to determine where the joint circle intersects the boundary of the circular visibility region. If the circular visibility region is disconnected into two regions we have two segments on the gates, but we can still only have a maximum of two transitions: entering and leaving the circular visibility region. The joint circle can not intersect both the forward and backward regions, because then it would intersect arcs of the circular visibility region three times, which is impossible. If the joint circle intersects with the gate in between the segments this implies that the joint circle has not yet entered the circular visibility region.

Because the arcs of the biarc are not allowed to intersect the gates of the start and endpoints twice we have to adjust this process slightly. Starting the potentially reachable region of the joint circle at p_i avoids arcs that intersect g_i more than once. However, it is possible that arcs start at p_i, pass through g_j, and then come back and intersect g_{j-1} and earlier gates. Such arcs are not allowed, because we cannot consider backward regions whose arcs intersect g_j twice. But because the backward visibility region is computed once during pre-processing, such regions are included. (They may be valid for biarcs ending at points beyond p_j.)

This means that we need to update the backward visibility regions of the gates between g_i and g_j to rule out arcs that pass through g_j twice. Updating all gates directly would take O(n) time, but we instead update regions "on the fly" as we consider them during the binary search. Because the invalid part of the backward visibility region corresponds to a constant interval of arcs, we can update the backward intersection segment on a gate in constant time. Therefore we update only O(log n) gates, taking only O(log n) time.

Once we have found the part of the joint interval reachable from each endpoint, we intersect them. All points in the intersection of these arcs on the joint circle are valid joints for biarcs.

Theorem 3. *Given an open polygonal curve $P = (p_1, \ldots, p_n)$, a polygonal tolerance boundary of size $O(n)$, and a gate for each p_i, we can approximate P by a minimum number of valid biarcs in $O(n^2 \log^2 n)$ time.*

Proof. For each point p_i, determine which part of on every other gate is reachable by computing W_i and $\tilde{W}_i$ in $O(n^2)$ time. For each pair of points use circular ray shooting to find where the joint circle first hits each boundary, and thus the joint interval (if any) in $O(n^2 \log^2 n)$ time. For each joint interval we determine the part of the joint interval reachable from the left endpoint and the part reachable from the right endpoint, doing binary search on the gates crossed by the joint interval. All points in the intersection of these arcs on the joint circle are valid joints for biarcs. This requires $O(n^2 \log n)$ time. This gives us a total run time of $O(n^2 \log^2 n)$. □

7 Future Work

It would be good to find algorithms that have fewer restrictions. For example, we could allow arcs or biarcs that do not start and end at original points.

References

1. J. Eibl. Approximation of Planar Curves Within an Asymmetric Tolerance Band. *M. Sc. thesis, Universität Salzburg, Institut für Computerwissenschaften*, 2002.
2. M. Held and J. Eibl. Biarc approximation of polygons within asymmetric tolerance bands. *Computer-Aided Design*, 37:357–371, 2005.
3. D. S. Meek and D. J. Walton. Approximation of discrete data by G^1 arc splines. *Computer-Aided Design*, 24:301–306, 1992.
4. D. S. Meek and D. J. Walton. Approximating quadratic NURBS curves by arc splines. *Computer-Aided Design*, 25:371–376, 1993.
5. D. S. Meek and D. J. Walton. Approximating smooth planar curves by arc splines. *Journal of Computational and Applied Mathematics*, 59:221–231, 1995.
6. L. Piegl. Curve fitting for rough cutting. *Computer-Aided Design*, 18:79–82, 1986.
7. J. Schoenherr. Smooth biarc curves. *Computer-Aided Design*, 25:365–370, 1993.
8. M. Yeung and DJ. Walton. Curve fitting with arc splines for NC toolpath generation. *Computer-Aided Design*, 26:845–849, 1994.
9. B.-Q. Su and D.-Y. Liu. Computational Geometry — Curve and Surface Modeling. *Academic Press*, 1989.
10. J.-H. Yong and S.-M. Hu and J.-G. Sun. A note on approximation of discrete data by G^1 arc splines. *Computer-Aided Design*, 31:911–915, 1999.
11. F. Chazal and G. Vegter. Computation of the medial axis of smooth curves with topological guarantees. *ACS Technical Report*, ACS-TR-122102-01, 2006.
12. K. Bolton. Biarc curves. *Computer-Aided Design*, 7:89–92, 1975.
13. S.-W. Cheng and O. Cheong and H. Everett and R. van Oostrum. Hierarchical decomposition and circular ray shooting in simple polygons. *Discrete and Comput Geom*, 32:401-415, 2004.