NISTIR 4586

CONTOUR OUTLINES

October 5, 1989

By:
Thomas R. Kramer

CONTENTS

LIST OF FIGURES

CONTOUR OUTLINES

Dr. Thomas R. Kramer
Guest Researcher, National Institute of Standards and Technology
& Research Associate, Catholic University

October 5, 1989

Funding for the research reported in this paper was provided to Catholic University under Grant No. 70NANB9H0923 and Grant No. 70NANB7H0716 from the National Institute of Standards and Technology

CONTOUR OUTLINES

I. <u>BASICS</u>

1. INTRODUCTION

This paper discusses geometric methods pertaining to planar curves consisting of sequences of straight line segments and arcs of circles joined end to end to make continuous composite curves. A composite curve of this sort will be called a “contour outline”. The following topics will be addressed:

1. Methods for rounding the corners of a frame for a contour outline.
2. Methods for creating contour outlines as arc-splines to:
   a. mimic the shape of an existing curve or,
   b. generate a “nice” curve which passes through or near a set of control points.

Contour outlines are interesting in connection with machining metal parts because instructions for numerically controlled machining centers typically may specify tool paths which are either straight line segments or arcs of circles (or helixes), but they typically may not specify any other type of curve. The driving force behind the research reported here was the need to create machinable curves, and the desire to introduce them at the design stage. Contour outlines may also be used effectively in computer graphics.

Other authors [MORT] [BEZI] have discussed various types of composite curves, but few research results concerning contour outlines, as defined here, have been reported. Contour outlines were discussed by the author in [KR&J] and [KRAM].

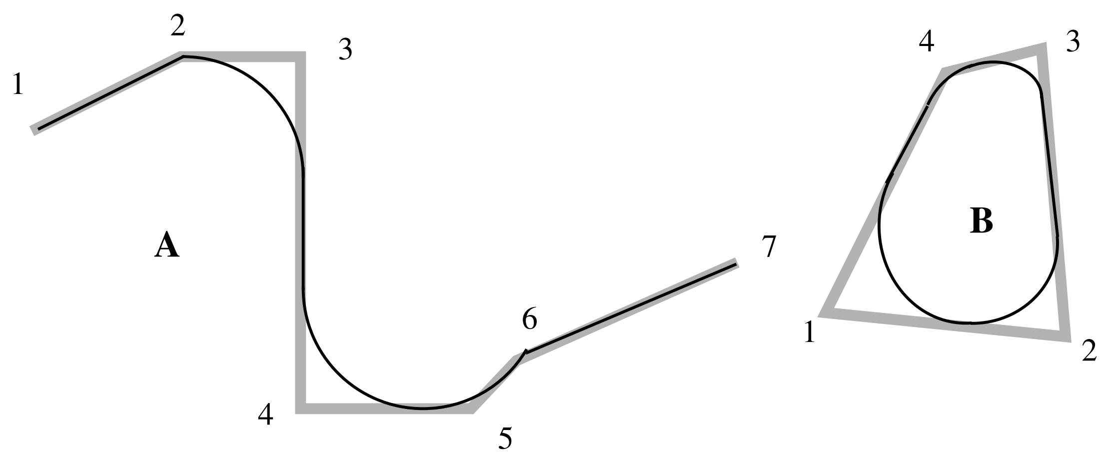
2. OPEN AND CLOSED

“Contour outline” has already been defined. If the starting point of a contour outline is identical to its end point, the outline will be “closed”. Otherwise, it will be “open”. Two examples of contour outlines are shown in Figure 1 with solid black lines. Figure 1A is open. Figure 1B is closed.

3. FRAME

The shape of a contour outline may be expressed efficiently by constructing a “frame” of straight line segments, or “arms”, around the outline and specifying how to round the corners of the frame. Frames of the contour outlines in Figure 1 are shown in heavy grey lines. A frame may be constructed for any contour outline by using the straight line segments already in the outline, plus two or more straight line segments for each arc. Two of the segments for an arc are constructed tangent to the arc at the ends of the arc. If the arc is less than a semicircle, the point at which the two segments intersect is taken as the far endpoint of each of them. If the arc is not less than a semicircle, a third segment tangent to the arc at the midpoint of the arc is constructed, and the points where it intersects the first two segments are used as the endpoints of the segments. Actually, any number of segments may be placed tangent to an arc and be part of a frame, so that a contour outline does not have a unique frame. In Figure 1A, for example, the arm between points 4 and 5 is not essential; the arms from 3 to 4 and from 6 to 5 could be extended to meet at a point below the bottom of the figure, making a different frame for the same contour outline.

Figure 1. Two Contour Outlines



Corners
1 (x 1.3 y 2)
2 (x 2.0 y 2.5 radius 0)
3 (x 2.6 y 2.5 radius 0.6)
4 (x 2.6 y 0.0 radius 0.6)
5 (x 3.5 y 0.0 radius 0.6)
6 (x 3.68 y 0.24 radius 0.0)
7 (x 5.0 y 0.8)

Corners
1 (x 5.5 y 0.5 radius auto)
2 (x 6.7 y 0.4 radius auto)
3 (x 6.5 y 2.5 radius auto)
4 (x 6.1 y 2.3 radius auto)
Rounding 100

In the application of the methods described in this paper which has been built by the author, there is an additional limitations on frames: the three endpoints of two successive arms of a frame may not be colinear. This is not a very severe restriction because if the second of two colinear segments is an extension of the first, the two may be replaced by a single segment extending from the beginning of the first to the end of the second. Cases of one arm doubling back along the preceding one are excluded, however.

In computer graphics terminology, what we have called a frame is often called a “polyline”.

4. CORNERS AND ROUNDING

The data to express the shape of a contour outline consists of a numbered list of “corners”, and an optional “rounding” parameter. A corners list is numbered sequentially, starting with 1. It will be convenient to use “n” as the index for corners, and “m” for the total number of corners.

The data for each corner include the x- and y-coordinates of the corner (expressed in terms of an ordinary Cartesian coordinate system) and, possibly, a value for the radius at the corner. An open contour outline has no radius at the first and last corners. Otherwise, every corner must have a value for the radius. The radius may have one of four types of values:

1. The value “auto”.

If there is a “rounding” parameter for the contour outline, “auto” means this corner should be subject to automatic rounding (as shown for all corners in Figure 1 B). If there is no rounding parameter, “auto” should not be used.

2. A non-negative real number.

A numerical value for the radius means the radius should have that value.

3. The value “join_back”, explained below.

4. The value “join_ahead”, also explained below.

The frame of a contour outline is formed by joining points given by the x- and y-coordinates of the corners with straight lines. The line segment of the frame which joins corner n to corner n-1 will be called the downstream arm of corner n. The segment joining corner n to corner n+1 is the upstream arm. In general, the direction of decreasing n will be called downstream and the direction of increasing n will be called upstream, so that downstream corresponds to the value of n moving down.

If a corner is to be rounded, it is rounded with a circular arc which is tangent to the arms of the corner. Each arm must be at least as long as the sum of the lengths of the lines from its adjoining corners to the tangent points of the respective arcs.

5. OTHER DETAILS

For a closed contour outline, corner 1 servers as corner n+1 for corner m (the last corner), and corner m serves as corner n-1 for corner 1. Closed contour outlines may be clockwise or counterclockwise in the upstream direction.

An arm of a frame may include a portion which becomes part of the contour outline, as shown between corners 3 and 4 in Figure1A, or all of the arm may be taken up with arcs, as shown between corners 4 and 5 in the same figure.

6. JOIN_BACK AND JOIN_AHEAD

If the corner radius at corner n is join_back, that means that corner n-1 must be rounded first, and the arc at corner n is large enough to be tangent to the downstream arm at the tangent point of the arc in corner n-1. Thus, the arc at corner n joins back to the arc at corner n-1. The idea of join_ahead is similar with respect to corner n+1. Examples are shown in Figure 2. In Figure 2 the frame is again shown in grey and the outline in black.

If the radius at corner n-1 is zero, or if there is no radius at corner n-1 (the only case of which is corner 1 of an open contour outline), then join_back means the arc at corner n is tangent to the downstream arm at corner n-1. The meaning of join_ahead is analogous if corner n+1 has zero or no radius.

Figure 2. Join-back and Join-ahead

Corners
1 (x 0.0 y 0.0)
2 (x 0.0 y 0.6 radius join_back)
3 (x 1.0 y 0.6 radius join_back)
4 (x 1.0 y 0.0 radius join_back)
5 (x 1.8 y 0.0 radius 0)
6 (x 1.8 y 1.8 radius join_ahead)
7 (x 0.4 y 1.8 radius 0.4)
8 (x 0.4 y 1.2)

Corners
1 (x 5.0 y 0.0 radius join_ahead)
2 (x 7.2 y 1.6 radius 0)
3 (x 6.0 y 2.0 radius join_ahead)

An entire open contour outline may be rounded by specifying join_back or join_ahead, since the end points provide a place to start. A closed contour outline requires at least one corner to have a numerical radius to provide a starting point.

A radius of join_back may not follow a radius of join_ahead. Otherwise, any combination of numbers, join_back, and join_ahead may be used, as long as the lengths of the arms of the frame will accommodate the arcs.

When building a contour outline which uses join_ahead and/or join_back, a subsequence of corners, each of which is to be rounded by the same type of join, must be rounded in order (backwards for a chain of join_ahead's, forwards for join_back's). Thus, the corners of the frame cannot normally be rounded in sequential order. The final result, however, is independent of the order of rounding corners, where there is a choice. The corners of the frame for Figure 2A, for example, could be rounded in the order 2,3,4,7,6 or the order 7,6,2,3,4.

Join_back and join_ahead are not used if automatic rounding is to be done.

II. AUTOMATIC FRAME ROUNDING

1. INTRODUCTION

A novel method of automatic frame rounding called the “fair share” algorithm has been devised. For simplicity, the method will be described as it applies to closed contour outlines. Then the relatively minor modification necessary to handle open contour outlines will be presented.

2. CONCEPT

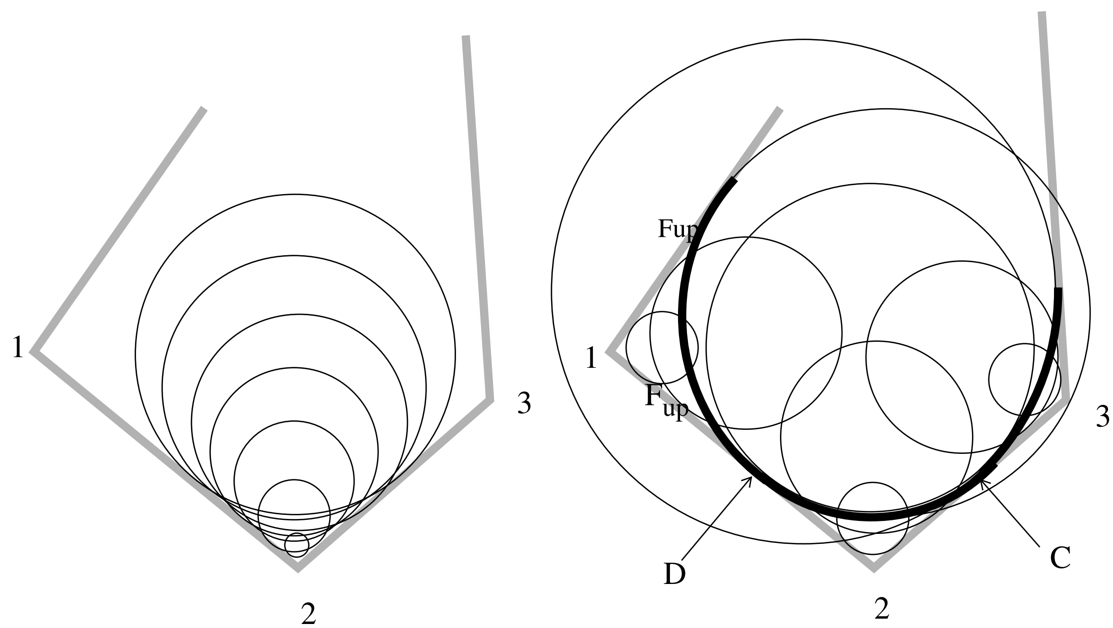
The approach to automatic frame rounding which has been implemented is illustrated in Figure 3, which shows three corners of a frame. Imagine that a tiny circle is placed in each corner of the frame. Think of each circle as a two-dimensional bubble. The bubbles are inflated, keeping them nestled in their corners, tangent to the arms of the corner. As the bubbles grow larger, the tangent points of each corner move away from the corner along the arms of the corner. A sequence of seven growing bubbles is shown for corner 2 on the left side of Figure 3. On the right side, three bubbles are shown in each corner.

As the bubbles get larger, eventually the two tangent points on some arm reach the same spot. When this happens, the size of the two bubbles which join at that spot is frozen. The arcs of the two bubbles which are in the respective corners are kept as portions of the contour outline. The bubbles in all unfrozen corners continue to be inflated until they join up with either frozen or unfrozen arcs. The process stops when all bubbles are frozen.

The rates at which the bubbles are inflated may be determined in a variety of ways. The method which has been implemented requires that the tangent points of all bubbles move away from their respective corners at the same rate. This has the happy result that arcs at two adjacent corners will meet at the midpoint of the arm that joins them, if the arcs are not first constrained at the other end. Thus, on the right side of Figure 3, since the arm between corners 2 and 3 is shorter than the arm between corners 1 and 2, the arcs in corners 2 and 3 freeze simultaneously when they join at C, the midpoint of the arm, before the arc in corner 1 freezes. When the arc in corner 2 is frozen, the point D becomes fixed. The arc in corner 1 continues to grow until the tangent point between corners 1 and 2 reaches D, at which time it is frozen. Observe that the radius of the arc in corner 3 is larger than the radius of the arc in corner 1, even though its bubble was frozen first. This is because the radii of the arcs are not growing at fixed rates.

An alternative to moving tangent points at equal rates would be increasing radii at equal rates. Another alternative would be to allow the areas between the bubbles and their corners to grow at the same rate. A third alternative would be to allow the centers of the bubbles to move away from their respective corners at the same rate. None of these alternatives has been implemented.

Figure 3. Principle of Automatic Frame Rounding



Seven views of the bubble growing in corner 2 are shown on the left. The picture on the right shows three views of the bubble in each corner. The final contour outline for corners 1, 2, and 3 is shown on the right with a heavy black line. In both views, the frame is shown with grey lines and the growing bubbles with light black lines.

3. FAIR SHARE ALGORITHM

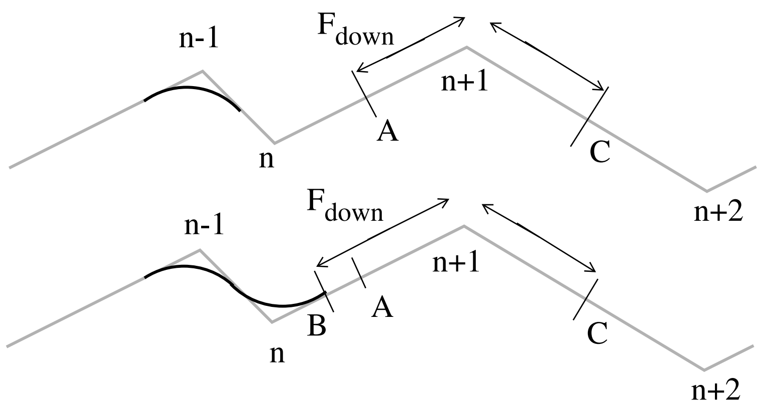
Conceptually, the rounding process is continuous. The bubbles continue to grow smoothly until they freeze. To implement the process on a computer, it is necessary to make the process discrete and desirable to make it iterative. This is done by the fair share algorithm. The process is made discrete by focusing on the instants at which the bubbles freeze and ignoring the growth process.

3.1. Calculating the Fair Share

We may assign to each corner a "fair share" of arm length. The fair share of a corner is the minimum of its upstream fair share and its downstream fair share. If the radius of corner n+1 has not yet been fixed, the upstream fair share at corner n is half the distance from corner n to corner n+1. If the radius at corner n+1 has been fixed, the upstream fair share at corner n is the distance from corner n to the point of tangency of the arc in corner n+1 to the arm between corner n and corner n+1. The downstream fair share is defined analogously. These definitions are derived from the growth process described earlier, since the growth of a bubble will not terminate until the bubble has taken at least its current fair share.

When an arc freezes (at corner n, we will say), the fair share of the upstream and downstream corners is recalculated according the method just given, if the radius at those corners has not already been fixed. Figure 4 shows an example of this. Note that the recalculated fair share is never smaller than the old value, since when a corner is rounded, it may give up a section of the frame to an adjacent corner (AB in Figure 4, for example).

Figure 4. Recalculating Fair Share



Suppose corner n-1 has been rounded but corners n and n+1 have not, as shown in the upper view. The fair share at corner n+1 is the minimum of F_{down} (half the distance from corner n+1 to corner n, as marked by point A) and F_{up} (half the distance from corner n+1 to corner n+2, as marked by point C).

In the lower view, corner n is rounded and is tangent at B to the arm between corner n and corner n+1. The fair share of corner n+1 is recalculated as the minimum of a new F_{down} (the distance from B to corner n+1) and the unrevised F_{up}.

The fair share of corner n-1 is not recalculated because the radius is already fixed.

3.2. Fixing Radii

The algorithm fixes radii in the same order as the growth process, according to the following steps:

1. Make a list of all corners and calculate the original fair share of each corner.
2. Sort the list by fair share size, smallest first.
3. Assign the radius of the first corner on the list so that distance from the corner to the tangent point of the arc is the corner's current fair share and remove the corner from list.
4. If the list is empty, stop.

5. Recalculate fair shares for adjacent corners of the corner to which a radius was just given, if they are still on the list.
6. Resort the list (note that at most two corners on the list have to move).
7. Go to step 3.

In the sorting process, if the fair shares of several sides are equal, the order in which those sides are listed does not matter; the result will be the same. If none of the several is adjacent to any of the others, this is obvious, since the fair shares of the others will not have to be recalculated, and the fair shares that are recalculated can only get larger. If two corners (say n and n+1) with currently equal fair shares whose radii are not fixed are adjacent, then the upstream fair share at n (call it F) is equal to the downstream fair share at n+1. If the fair share is equal to F, then the downstream fair share at n and the upstream fair share at n+1 cannot be smaller than F. Hence when the radius at either corner is fixed, the distance from the corner to the tangent point will be equal to F and the point of tangency will be at the midpoint of the arm between n and n+1. If the fair share is less than F (call it G), then the downstream fair share at n and the upstream fair share at n+1 must both be equal to this smaller number. Regardless of which corner is rounded first, the fair share of the other corner will not change, so the distance from that corner to the tangent point will also be equal to G when it is fixed.

3.3. Modification for Open Frames

If a frame is open, the downstream fair share of corner 2 is always the distance between the first and second corners, and the upstream fair share of the next-to-last corner is always the distance from that corner to the last corner.

3.4. Observations

The fair share algorithm produces several interesting effects.

1. The radius assigned to each corner is at least its original fair share radius (half the smaller of adjacent arms).

This is because a recalculation of fair share can never decrease the fair share of a corner.

2. Any frame which is a regular polygon is rounded into a circle.

The fair shares at all corners start out equal (and do not change) and the included angle at all corners is the same, so the radii will be the same. Since all the arcs join at the points of tangency to the arms of the frame, they form a circle.

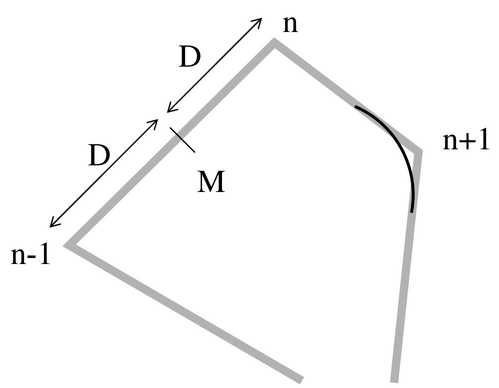
3. Every arc is attached to at least one other arc (except that the arcs in the second and next-to-last corners of an open contour outline may possibly not be attached to any other arc).

This may be seen as follows. At the time the radius for the arc at corner n is to be assigned, there are three possibilities for the two adjacent corners: both have already had radii assigned, only one has had a radius assigned, or neither has had a radius assigned.

Case 1 - Radii of both adjacent corners already assigned:
In this case the fair share at corner n will be equal to the smaller of the two distances between corner n and the points of tangency of the arcs in the adjacent corners to the arms of corner n. Hence, the arc in corner n will join with the closest adjacent arc.

Case 2 - Radius of one adjacent corner is already assigned.
This situation is shown in Figure 5. Suppose a radius has been assigned to the upstream corner, n+1. Note that the fair share point on the arm between corner n and corner n-1 is at the midpoint M of that arm, a distance D from corners n and n-1. If the arc in corner n does not link to the arc in corner n+1, then the arc in corner n reaches to M. But the fair share at corner n-1 cannot be smaller than the fair share at n (or n-1 would get a radius before n), so the downstream fair share at corner n-1 must be at least as large as D. Hence, when corner n-1 is rounded, its arc will join at M to the arc at n. A similar argument applies if the downstream corner rather than the upstream one is already rounded.

Figure 5. Radius of Adjacent Corner Assigned



Case 3 - Neither adjacent arc has been assigned a radius.
The arc in corner n will reach to the midpoint of the smaller adjoining side. Since the fair share of the adjoining corner on that side is not smaller than that at n, it must be the same size as the fair share at n. Hence, the arc in that corner will join with the arc at corner n when it is made.

4. Any closed frame with an odd number of corners will result in a contour outline with at least one sequence of at least three arcs joined smoothly end to end.

This is a direct result of observation 3, since if it were not true, there would be an unattached arc somewhere.

4. VARIATIONS OF THE FAIR SHARE ALGORITHM

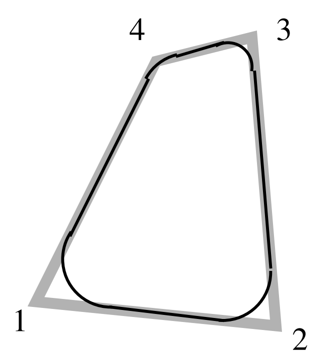
Several variations of the fair share algorithm have been implemented.

4.1. Reduced Radii

The simplest variation is to reduce corner rounding after radii have been assigned to all corners by multiplying each radius by the same factor, which must be between zero and 1. The “rounding” parameter used in describing contour outlines is used this way, except that the value of “rounding” is a percentage, and is thus in the range of zero to 100.

This variation is useful for drawing. It is common for computer graphics drawing systems to have the capability to round the corners of a frame slightly. The fair share algorithm with the “rounding” parameter provides a method of accomplishing this. Results are pleasing to the eye. Figure 6, for example, shows the same frame as Figure 1B with rounding reduced to 40.

Figure 6. Reduced Rounding




```
Corners
  1 (x 5.5    y 0.5   radius auto)
  2 (x 6.7    y 0.4   radius auto)
  3 (x 6.5    y 2.5   radius auto)
  4 (x 6.1    y 2.3   radius auto)
Rounding 40
```

This contour outline has the same frame as in Figure 1B, but rounding is reduced.

4.2. Alternate Methods of Computing Fair Shares

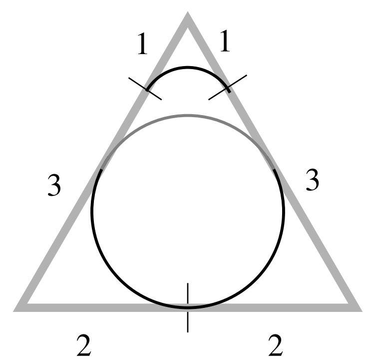
The fair share algorithm consists of two separable parts: (i) a method of computing fair shares at each corner, and (ii) a method of fixing radii. The seven-step procedure for fixing radii described earlier may be used unchanged with different methods of computing fair shares. One of the spline methods described later in this paper uses an alternate fair share computation.

If an alternate method of computing fair shares is used, the observations numbered 1 through 4 above do not necessarily hold. Figure 7 is an example of a frame rounded by determining fair shares differently, but using the same method of fixing radii, in which there is an unattached arc. The figure is an equilateral triangle, 4 units on a side. The numbers on the outside give the upstream and downstream fair share at each corner. The top corner has a fair share of 1 and will be rounded first. The bottom corners have a fair share of 2, and will be rounded next. The arc in the top corner is not attached to any other arc.

If the arcs in Figure 7 had been created using the original method of computing fair shares, they all would be the same as shown, except the top arc, which would join at both ends with the arcs below it, as shown by the dotted arc.

If an alternate method of computing fair shares is used, the sum of the two fair shares on an arm of the frame does not have to equal the length of the arm, but neither share can be more than the length. If the sum is more than the length, the fair share of one of the corners will decrease during the radius fixing process.

Figure 7. Unattached Arc



4.3. Pre-assignment of Radii

Some of the corners of a frame may have radii assigned to them by some other means before the fair share algorithm is applied. The bubble growth concept and its implementation may be applied to the remaining unrounded corners with little change in the implementation. It is not difficult to use the “rounding” parameter for reducing radii together with pre-assignment of radii, and this has been implemented.

III. ARC-SPLINES

1. INTRODUCTION

A "spline" is a continuous curve that passes through or near a set of specified points, called control points. A great deal has been written about splines ([BROD] [CA&R] [FA&P] [MORT], are a few references), and it is not the purpose of this paper to present a review of spline methods.

A contour outline constructed as a spline will be called an arc-spline. Two methods of constructing arc-splines are presented here: the "easy" method and the "expert" method. Both arc-spline methods use control points to create arcs, use the arcs to define a frame, and then use rounding methods (such as fair-share) to fix the radii of the corners of the frame. Both methods have been implemented. In the implementation, control points are located by moving a pointer on a computer screen with a mouse and clicking a mouse button when the pointer is at the desired location. The user may add, delete, or move control points with the mouse and keyboard. Once the user is finished locating control points, the system automatically performs the rest of the calculations. More details of the implementation are given in [KRAM].

A third type of spline may by constructed by using the control points as the corners of a frame and then assigning radii to the corners. In the implementation, the user has a choice of five methods of assigning radii:

1. The user assigns a radius to each corner desired. The radius of any corner not dealt with by the user defaults to zero.
2. The user specifies a number which is used as the radius for all corners.
3. The user tells the system to assign radii by the fair share method and then take a percentage, as given by the value of the "rounding" parameter.
4. The user assigns a radius to each corner desired. The radius of any corner not dealt with by the user defaults to a value specified by the user.
5. The user assigns a radius to each corner desired. The radius of any corner not dealt with by the user is assigned by the fair share method.

This method is easy to use, and is excellent if a rounded frame is what is desired. It is not recommended for copying or drawing, however, because the contour outline which is created is apt to be far from the control points.

Work on curve fitting using circular arcs was reported previously by [BOLT] and [MOSE]. In [MOSE] a spline is fitted to a set of control points by putting an arc through the first three points and fitting every other point by putting an arc through it tangent to the previous arc.

2. EXPERT ARC-SPLINE METHOD

The “expert” arc-spline method was developed for copying existing curves and gets its name from the requirement that the user have some expertise in visually decomposing an existing curve into straight line segments and roughly circular arcs. The method is illustrated in Figure 8.

The user examines the curve to be copied and divides it into parts that appear to be arcs of circles or straight line segments, as shown in Figure 8A. If an arc is larger than a third of a circle, it should be subdivided into two or more parts.

The user marks control point 1. If the curve is open, control point 1 may be at either end. If the curve is closed, control point 1 may be at the end of any arc or line segment. The user marks control point 2 anywhere in the interior of the first arc or line segment, control point 3 at the end of the first arc or line segment (which is also the beginning of the second), control point 4 anywhere in the interior of the second arc or line segment, and so on. Control points used for sharp bends (which includes the ends of open curves) are represented differently from control points used for smooth joints and interior points. In Figure 8B a heavy **Y** indicates an end or sharp bend, while a light X indicates a smooth region. An open curve requires an odd number of control points and a closed curve an even number. Every even numbered control point is an interior point marked with a smooth marker. The distance between an interior control point and the ends of the arc or line segment on which it lies is not critical.

This completes the role of human judgement in the process. The rest is deterministic and is best handled by a computer. In the implementation, the computer takes over as soon as the user stops marking control points.

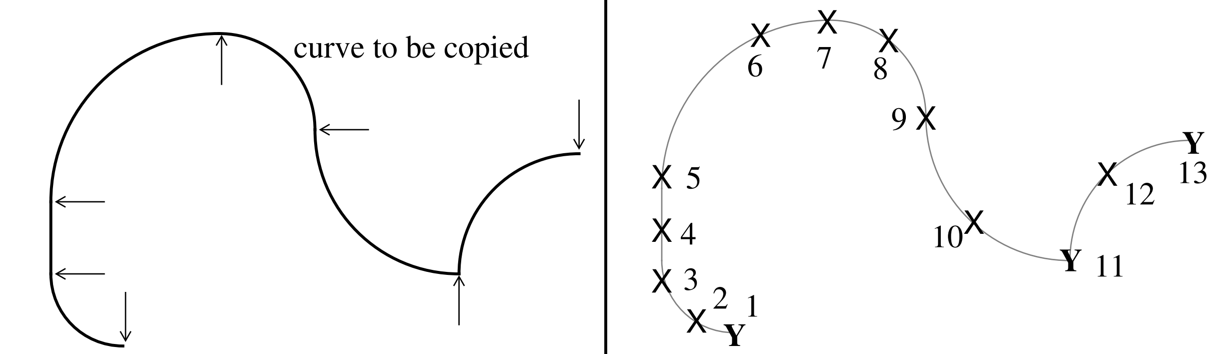
Each triple of control points starting with an odd-numbered point (1 2 3, 3 4 5, 5 6 7, etc.) is used to determine the circular arc which passes through them. If the points are colinear, an arc of large radius (1000000 is large in the implementation) is used to approximate the line. This eliminates the need to handle straight line segments separately after this step in the procedure. Figure 8C shows an exploded view of the arcs.

A two-arm frame is placed around each arc. The point at which the two arms intersect is calculated for each two-arm frame. A frame for the entire curve is generated by taking all of these intersection points plus the control points at all sharp corners (including the ends of the curve, if any). These frame corners are shown with heavy dots in Figure 8C and 8D. They are numbered in 8D.

A radius of zero is assigned to any sharp corner of the frame. The fair share of any other corner is the length of an arm of the two-arm frame for the corner (the arms are the same length).

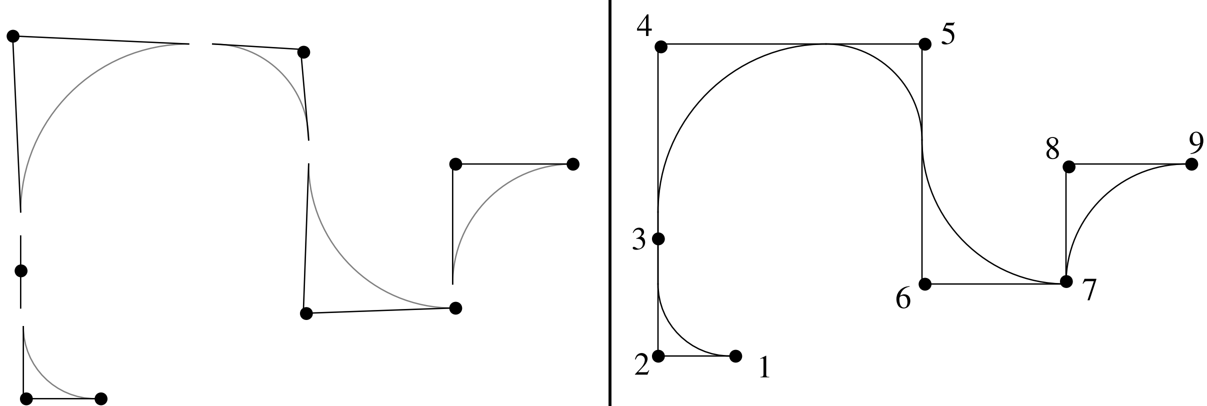
Finally, use the radius fixing algorithm described earlier to assign radii to each corner of the frame.

Figure 8. Expert Method for Arc-Splines



A. Perceive arcs and straight line segments on the curve to be copied. Arrows on the figure show where this particular curve might be divided.

B. Place the curve over the computer screen. Use the mouse to mark ends and interior points of each arc, with different marks for smooth (X) and sharp (**Y**) corners. Remove the curve from the screen.



C. For overlapping successive triples of control points, (1 2 3, 3 4 5, etc.) construct a circular arc passing through each triple (dotted lines), and put a two-arm frame around each arc. Use the midpoint of each frame, plus all control points at sharp corners as the corners of a frame for the entire curve (shown with heavy dots above).

This diagram is shown in exploded view for ease of perception only.

D. Assign fair shares to each corner of the frame. The fair share of a sharp corner is zero. Fair shares for the rest of the corners are the length of either arm of the two-arm frame at the corner (they are both the same). Use the radius fixing algorithm to assign radii, completing the process.

Neighboring arms of the two-arm frames of adjacent non-sharp corners of will be colinear if the arcs of the curve being copied that gave rise to those corners are exactly circular and the user has located control points exactly (the near-vertical arms in the middle of Figure 8C, for example). Since all that exactness is very unlikely, such arms will be slightly non-colinear. Thus, the sum of the fair shares of adjacent corners in this algorithm may be slightly greater than the distance between those corners, but this has little practical effect on the operation of the radius fixing algorithm.

The expert method produces a frame with relatively few corners, compared with the number of control points. If the number of control points is C, of which S are sharp, then the number of frame corners, F, will be $F = C/2 + S$ for a closed frame and $F = (C-1)/2 + S$ for an open frame. S is at least 2 for an open frame, since ends are counted as sharp corners. Thus, the minimum number of frame corners is $C/2$ for a closed frame and $(C+3)/2$ for an open frame. The maximum number of sharp control points, S, for a closed frame (since only odd-numbered control points can be sharp) is $C/2$ for a closed frame and $(C+1)/2$ for an open frame. Thus, the maximum number of frame corners is $C/2 + C/2 = C$ for a closed frame and $(C-1)/2 + (C+1)/2 = C$ for an open frame.

3. EASY ARC-SPLINE METHOD

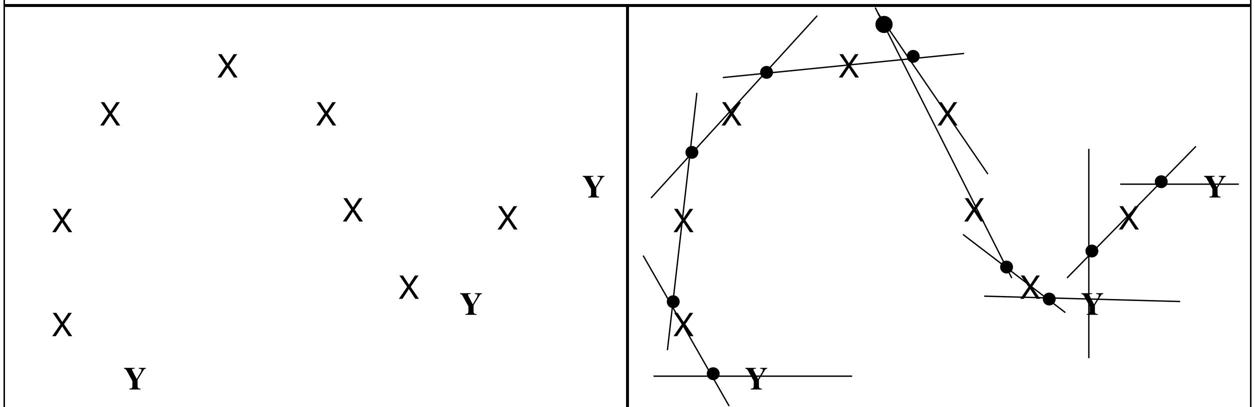
The easy Arc-Spline method may be used either for copying an existing curve or drawing a new curve. The method is depicted in Figure 9. The figure and the description below show how the algorithm works when drawing a new curve. The only difference for copying an existing curve is that the points are selected to lie on the existing curve.

The user marks a sequence of control points, as shown in Figure 9A. As with the expert method, marks used for sharp bends (which includes the ends of open curves) differ from marks used for smooth joints. Locating control points so that the change in direction of lines joining successive pairs of control points is less than 60 degrees on the smooth parts of the curve produces the best results. The change of direction should normally not be more than 90 degrees. A few trick constructions, such as approximating a circle by locating three control points, violate this rule.

This completes the role of human judgement in the process. In the implementation, the computer takes over as soon as the user stops marking control points. Unlike the expert method, no distinction is made between even- and odd-numbered points.

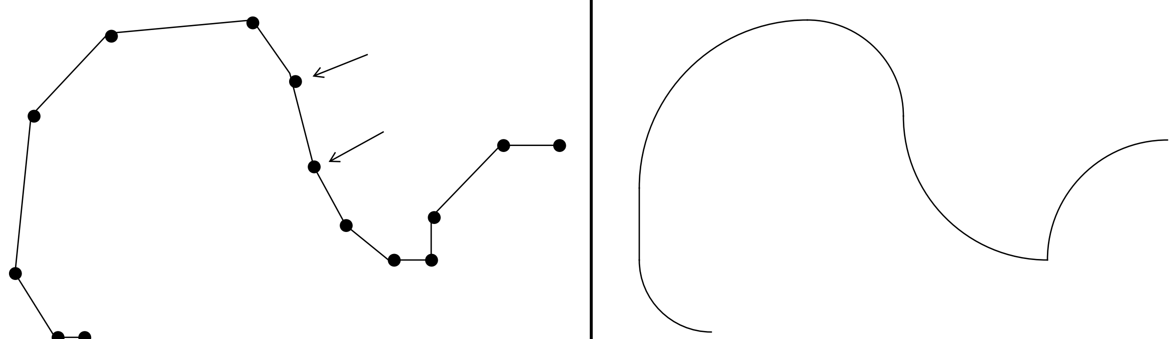
The system looks at each control point marked by the user and, if possible, constructs a circular arc through the point and two of its neighbors. Three cases are shown in Figure 10. If the control point is not at a sharp bend or end, the neighboring points are the one upstream and the one downstream (Figure 10A). If the control point is an end, and the point next to it is not at a sharp bend, the two neighboring points are used (Figure 10B). If the control point is at a sharp bend, two arcs through the point are constructed, one using the downstream two points, and one the upstream two points (Figure 10C) - unless the first point downstream or upstream is at a sharp bend, in which case no arc is constructed on that side.

Figure 9. Easy Method for Arc-Splines



A. Use the mouse to mark control points, with different marks for smooth (X) and sharp (**Y**) points.

B. At each control point, construct an arc through the point and two of its neighbors (arcs not shown). Draw a line through each control point tangent to its arc. Sharp corners get two tangents.



C. Create a frame by using the intersection points of successive tangent lines, plus all control points marked as sharp bends.

Exception: at a place of inflection, the middle two control points of the place of inflection are both used as corners, and the intersection of the tangent lines at those points is not used (see arrows above).

D. Round the frame by assigning a zero radius at sharp corners and using the fair share method without variation to assign radii to the rest of the corners.

A line is drawn tangent to the arc through each control point which has an arc, as shown in Figure 9B and Figure 10. A sharp corner with two arcs gets two tangent lines.

In general, these tangent lines will be used to form a frame, but there is a problem in the case of a place of inflection. A place of inflection is a sequence of control points (A B C D) such that the change of direction between line AB and line BC has a different sign from the change of direction between line BC and line CD.

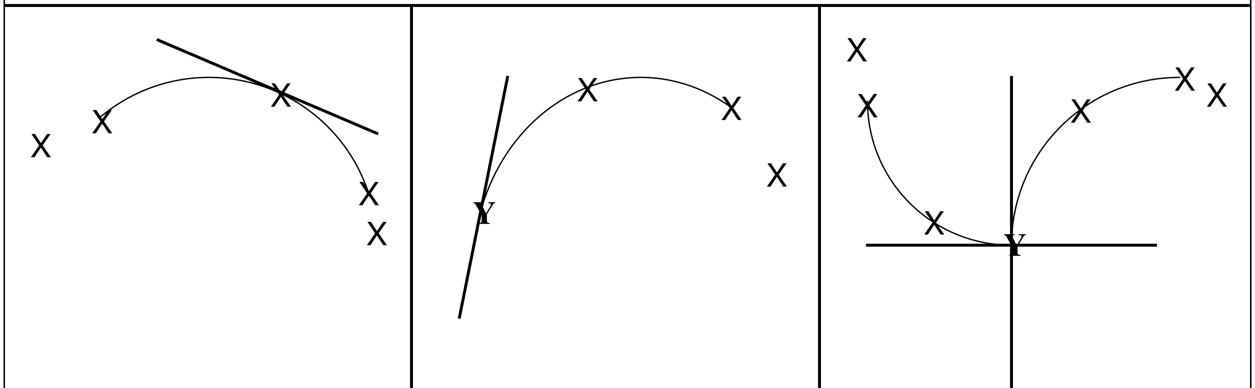
Intersection points of successive tangent lines are found. If two successive tangent lines are not lines passing through the middle two points of a place of inflection, the intersection point of the pair is used as a corner of the frame for the final curve. The intersection points are shown with heavy dots in Figure 9B.

If there is a place of inflection, the control points in the middle of the place of inflection are used as corners, and the intersection of the tangents at those points is not used. As may be seen in Figure 9B, the intersection point of such tangents (the heavy dot nearest the top of the figure) may be in a location that has no relevance to constructing a frame.

If a control point is at a sharp bend, it is used as a corner of the frame.

As shown in Figure 9C, the frame consists of all corners selected as just described.

Figure 10. Constructing Arcs and Tangents from Control Points



A. Control point and its neighbors are not ends or sharp bends. One arc and tangent are constructed.

B. Control point is an end and nearest neighbor is not an end or sharp bend. One arc and tangent are constructed.

C. Control point is a sharp bend and two nearest neighbors are not ends or sharp bends. Two arcs and tangents are constructed.

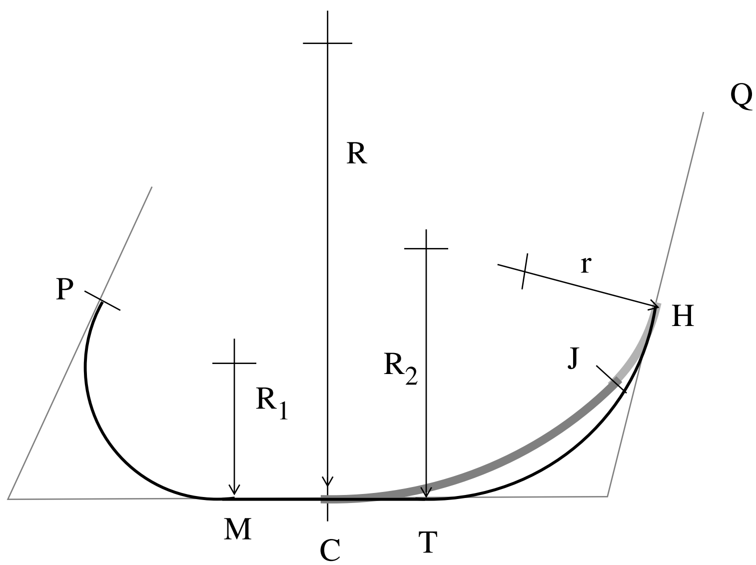
Round the frame, as shown in Figure 9D, by assigning a zero radius at sharp corners and using the fair share method without variation to assign radii to the rest of the corners.

The easy method produces a frame with somewhat more corners than control points. If the number of control points is C, then the minimum number of frame corners, F, will be F = C for any closed frame with no sharp corners and no inflection places in the control points, and F = C+1 for any open frame with the same limitations. The effect of sharp control points is variable, and places of inflection are a confounding factor, so there is no general formula relating the number of control points, sharp control points, and places of inflection to the number of frame corners. A nearly maximum number of frame corners, however, occurs when every other control point is sharp, giving F = 3C/2 for a closed frame and F = (3C-1)/2 for an open frame.

IV. REMOVING STRAIGHT LINE SEGMENTS FROM CONTOUR OUTLINES

Contour outlines produced directly from frames or by either arc-spline method will generally include some straight line segments. Other types of splines normally contain no straight line segments. If it is desired to remove straight line segments from a contour outline, that may be done as follows.

Figure 11. Replacing Straight Lines with Arcs



1. APPROACH TO REMOVING STRAIGHT LINE SEGMENTS

In Figure 11, two corners of a frame (dotted line) have been rounded with circular arcs (solid black). The radius of the arc on the left is R_1 and on the right R_2. The left arc is tangent to the frame at points M and P, the right arc at T and H. A straight line segment is part of the contour outline from M to T. The line may be eliminated as follows. Locate point C between M and T. Any place will do, but the spot where $MC/CT=R_1/R_2$ seems best from an esthetic viewpoint. We may assume another arc joins arc TH from the direction of Q, since if it did not, R_2 could be increased until arc TH joined with one of the neighboring arcs. Line segment CT and arc TH are replaced by two arcs, CJ (dark grey) and JH (lighter grey). Arc CJ is constructed tangent to the frame at C and tangent to arc JH at J (a point to be determined). Arc JH is constructed tangent to the frame at H and tangent to arc CJ at J. The radius, R, of arc CJ will be larger than R_2, and the radius, r, of arc JH will be smaller than R_2.

A similar construction, not shown in the figure, can replace line CM and arc MP with two arcs.

The two new arcs are under-determined by the requirements imposed on them. Any positive radius r smaller than R_2 may be used for arc JH, and that will determine a value for R. This means an esthetic criterion may be applied in the determination of R and r. It would be nice to require that R equal r, which is equivalent to saying R/r=1, but this is impossible, since $R>R_2>r$. It is possible, however, to impose the condition that R/r be as close to 1 as possible, i.e. it is at a minimum. That R/r has a minimum somewhere in the interval $0<r<R_2$ is clear from Figure 11, because (i) the ratio approaches infinity as r approaches zero, and (ii) the ratio approaches infinity as r approaches R_2, since arc CJ approaches straight line segment CT, implying R approaches infinity.

The determination of R and r is illustrated by Figure 12.

2. CONSTRUCTION OF TANGENT ARCS

To construct Figure 12, suppose that lines CF and FH meet at angle θ. It is desired to construct two arcs, HJ and JC so that arc JC is tangent to CF at C, arc HJ is tangent to FH at H, and the two arcs are mutually tangent at J. Let G be the point at which a perpendicular from H meets CF. Let the length of HG be h and the length of CG be d. The parameters d, h and θ specify the situation completely.

In the figure, we have let θ be obtuse. If θ is acute, Figure 12 is slightly different, the two angles labeled θ-90 are 90-θ instead, and some equations which follow are different. The differences, however, are slight, and equations [5] and [7] through [22] are unchanged. It is also possible for point D to lie between F and G, leading to the same result.

1. Construct a perpendicular to FH at H on the inside of corner CFH and a perpendicular to CG at C, also on the inside.

2. Locate the point I anywhere on the perpendicular from H, as long as the distance from I to H is less than the perpendicular distance from I to CF. The length of IH will be the radius r of the small arc.

3. Drop a perpendicular from I to CF, meeting CF at D, and passing beyond D.

4. Draw an arc with its center at I of radius r downward from H to intersect ID at K.

5. Draw line CK and extend it to intersect the arc at J.

6. Draw line JI and extend it to intersect the perpendicular from C at A. AC is the radius R of the larger arc, and A is the center of the arc.

Observe that ΔIJK is isosceles, and AC is parallel to IK. Hence ΔACJ is also isosceles. Thus, J lies on the arc of radius R whose center is at A. Since I is the center of an arc passing through J and lying on the radius AJ of a larger arc, the two arcs must be tangent at J.

7. Draw the arc JC. Call angle KCD α. Observe that angle CAJ is 2α.

The construction of the arcs is now complete, but to calculate R and r, it is useful to add a few construction lines.

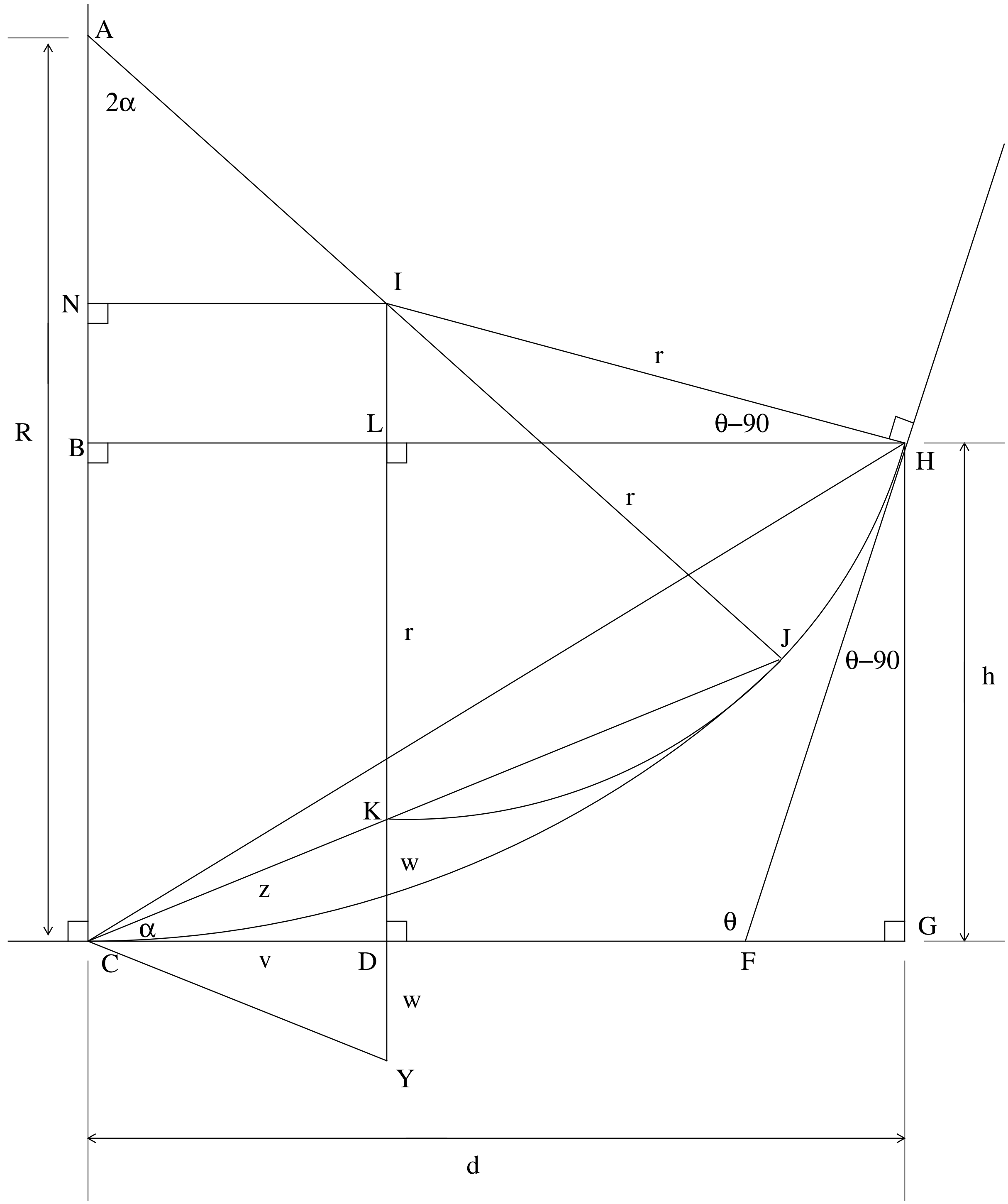
8. Drop perpendiculars from H and I to AC, meeting at B and N, respectively. Let L be the intersection point of ID and HB. Observe that angle LHI is θ–90.

9. Mark point Y on ID extended, so that YD = DK. Draw line CY.

Observe that ΔCYK is isosceles and similar to ΔACJ, since they both have an angle of 2α between the two equal sides.

10. Draw line CH.

Figure 12. Construction of Tangent Arcs



3. MINIMIZING R/r

From ΔILH,
[1] $IL = r\sin(\theta-90) = -r\cos\theta$
[2] $HL = r\cos(\theta-90) = r\sin\theta$

Since DGHL is a rectangle,
[3] $DG = HL = r\sin\theta$
[4] $LD = HG = h$

Let v = CD. By examining line CG and using equation [3]

[5] $v = CD = CG - DG = d - r\sin\theta$

By examining line ID and using equations [1] and [4]

[6] $ID = LD + IL = h - r\cos\theta$

Let w = KD. By examining line ID, using equation [6], and noting how IK was constructed,

[7] $w = KD = ID - IK = h - r\cos\theta - r$

Then from ΔCDK and equations [5] and [7],

[8] $\tan\alpha = w/v = (h - r\cos\theta - r)/(d - r\sin\theta)$

Let z = KC. Since ΔCYK is similar to ΔACJ,

[9] $z/2w = R/CJ$

But, since IY is parallel to AC, the base of ΔACJ,

[10] $R/CJ = AI/z$

By examining line AJ,

[11] $AI = AJ - IJ = R - r$

Substituting [11] in [10] and then [10] in [9],

[12] $z/2w = (R-r)/z$

Solving for R,

[13] $R = r + z^2/2w$

Since ΔCDK is a right triangle, $z^2 = w^2 + v^2$, so [13] becomes:

[14] $R = r + w/2 + v^2/2w$

Substituting [5] and [7] in [14] yields:

[15] $R = h/2 - (r\mathit{cos}\theta)/2 + r/2 + (d - r\mathit{sin}\theta)^2 / 2(h - r\mathit{cos}\theta - r)$

As discussed earlier, we would like to minimize R/r. We will use equation [15] to calculate f(r) = R/r, then take the derivative of f, set it equal to zero, and solve for r.

[16] $f(r) = R/r = h/2r + (1 - \mathit{cos}\theta)/2 + (d - r\mathit{sin}\theta)^2 / 2(hr - r^2\mathit{cos}\theta - r^2)$

[17] $$f'(r) = -h/2r^2 + \frac{\{(hr - r^2\mathit{cos}\theta - r^2)2(d - r\mathit{sin}\theta)(-\mathit{sin}\theta) - (d - r\mathit{sin}\theta)^2(h - 2r\mathit{cos}\theta - 2r)\}}{2r^2(h - r\mathit{cos}\theta - r)^2}$$

[18] $0 = -h(h - r\mathit{cos}\theta - r)^2 + 2(hr - r^2\mathit{cos}\theta - r^2)(r\mathit{sin}^2\theta - d\mathit{sin}\theta) - (d - r\mathit{sin}\theta)^2(h - 2r\mathit{cos}\theta - 2r)$

This simplifies to:

[19] $0 = r^2\,\{(\mathit{cos}\theta + 1)(h\mathit{cos}\theta + d\mathit{sin}\theta)\} - r\,\{(\mathit{cos}\theta + 1)(h^2 + d^2)\} + \{h(h^2 + d^2)/2\}$

Equation [19] may be solved by the quadratic formula, using the coefficients:

$a = (\mathit{cos}\theta + 1)(h\mathit{cos}\theta + d\mathit{sin}\theta)$
$b = -(\mathit{cos}\theta + 1)(h^2 + d^2)$
$c = h(h^2 + d^2)/2$

The discriminant, b^2-$4ac$, simplifies to a perfect square, $(h^2 + d^2)\{d(\mathit{cos}\theta + 1) - h\mathit{sin}\theta\}^2$, so, after substituting for a, b, and c in the quadratic formula and simplifying, we get:

[20]
$$r = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} = \frac{(h^2 + d^2) \pm \frac{d - h\mathit{sin}\theta}{\mathit{cos}\theta + 1}\sqrt{h^2 + d^2}}{2\,(h\mathit{cos}\theta + d\mathit{sin}\theta)}$$

The value of r found by using the + option of the ± sign is larger than R_2 and cannot be used.
After solving for r, R and α may be found by substituting in equations [15] and [8].
Equation [20] is more attractive if written in terms of the sides a=CF, b=FH, and c=CH of ΔCFH. Noting that $d=(a-b\mathit{cos}\theta)$, $h=b\mathit{sin}\theta$, and $c = \sqrt{h^2+d^2}$ equation [20] becomes:

[21] $r = c(c+a-b)/2a\mathit{sin}\theta$

By writing equation [21] as follows, and noting that it is easy to construct $a\sin\theta$ given a and θ, we observe that by using any of several common constructions for determining one term of an equality of two ratios, given the other three terms, the length of r at which R/r is a minimum may be constructed with straight-edge and compass.

[22] $r/c = (c+a-b)/2a\sin\theta$

4. APPLICATIONS

This technique has not been implemented, but it could be used several ways.

The most straightforward use would be to eliminate straight line segments in any contour outline which had already been completely defined by any of the methods above.

Another application would be in a variation of the easy arc-spline method. The frame would be constructed exactly as described for the easy method, but the fair share method of rounding the frame would be replaced as follows:

1. Use the control points from the easy method (except the middle two points of a place of inflection) as points at which the arc-spline should be tangent to the frame (like points C and H from Figure 12).

2. At a place of inflection, use the midpoints of the three line segments between successive points of the place of inflection as points of tangency.

3. Construct pairs of arcs between points of tangency as shown earlier in this section.

This method would yield an arc-spline which consists entirely of arcs of circles, and passes through all control points except those which are in the middle of places of inflection. Moreover, because of the method of using control points for constructing a frame in the easy method, the effect of control point n on the shape of the arc-spline would not extend below control point n-2 or above n+2, except at places of inflection, where the effect would stop at n-3 and n+3. In other words, control point n could be moved without affecting the shape beyond those bounds.

A technique for making pairs of tangent arcs similar to the one just described was implemented by the British Ship Research Association [BOLT] and has been used in shipbuilding. None of the calculations given here were reported, however.

V. CRITERIA AND COMPARISON

This section presents criteria for judging spline curves and compares the expert and easy arc-splines just described with each other, with other spline methods, and with the use of join_ahead and join_back.

1. PASS THROUGH CONTROL POINTS

Several authors, [CA&R] for example, distinguish between "interpolating splines", which pass through their control points and "approximating splines", which do not necessarily pass through their control points, but pass near them.

The expert and easy method for making arc-splines both produce approximating splines. If the guidelines given previously for constructing control points are followed, however, many or all of the control points may line on the final curve, and the maximum distance of any control point from the final curve will be small compared to the distance between control points. No theory of the maximum possible deviation has been developed.

2. NEAR TO LINES BETWEEN CONTROL POINTS

It is desirable that a spline curve stay near the path defined by straight lines between control points, bending away from one part of the path only to get closer to the next part.

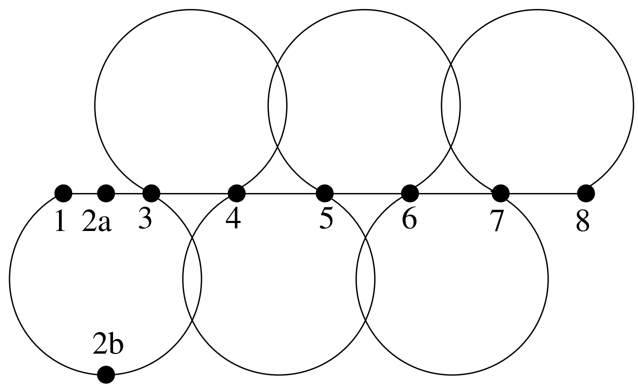
Some types of splines, such as cubic splines, tend to produce spurious loops away from the path [BROD, p. 7], and "spline tension" has been devised to eliminate this problem. Arc-splines are very well behaved in this regard as long as the guidelines for control points are followed. If the guidelines are violated, the results may be extremely bad. In the expert method, for example, if control points are located so poorly that adjacent arms of neighboring two-arm frames are far from colinear, the final curve may incorporate spurious 360 degree loops and stray far from the path.

The method used in [MOSE] is an excellent example of an interpolating spline (it passes through all control points) which is of little practical value because its normal behavior is to stray far from the path. Figure 13 shows a simple example of this.

3. INVARIANCES

It is desirable that the shape of a spline curve be invariant under (i) rotation and translation of axes (ii) reversal of the numbering of control points, and (iii) if the curve is closed, circular permutations of the numbering of control points. The methods described here have all of these kinds of invariance.

Figure 13. Badly Behaved Spline Method



One method of using arcs to make a spline, as described in [MOSE], is to pass an arc through the first three control points, and then construct each successive arc by making it tangent to the previous arc and passing it through the next control point. This method results in a spline which passes through all control points but, typically, does not stay close to the path formed by straight line segments between control points.

In the picture above, points 1 and 3 through 8 are colinear and equally spaced. If point 2 is placed midway between point 1 and point 3, as shown by point 2a, this method produces a straight line through all eight points. If point 2 is moved to the place shown by point 2b, the method makes the loopy curve shown above, not an attractive result.

4. INFLECTION

The easy arc-spline method, as already described, handles places of inflection in the control points awkwardly. Places of inflection are handled transparently by the expert arc-spline method and most other methods.

5. QUALITY OF COPY

If a spline is used to mimic an existing curve, how many control points are needed to achieve a given level of approximation?

No formula has been derived to answer this question, but these methods have been applied in enough cases that the following empirical observations may be made.

1. The expert arc-spline method gives good results with relatively few control points when the guidelines for control points are followed.

2. The easy arc-spline method gives good results if the guidelines are followed but requires about twice as many control points as the expert method for the same quality. The expert

method is more efficient because it benefits from human judgment in detecting transitions in curvature, while the easy method does not.

3. Using too few control points results in the produced curve straying to one side of the curve being copied over a large interval. Using too many control points makes the produced curve wiggle perceptibly back and forth over the curve being copied. These behaviors are typical of other spline methods.

No head-to-head tests of arc-spline methods against other methods have been conducted.

6. VISUAL APPEAL AND MECHANICAL USEFULNESS

A point at which a curve is continuous may be called a C0 point. A point at which the first derivative of a curve is continuous may be called a C1 point. A point at which the second derivative of a curve is continuous may be called a C2 point, and so on for C3, C4, etc. A sharp corner of a curve is a point which is C0 but not C1.

The human eye detects discontinuities and sharp corners (points which are not C0 or C1) automatically and without conscious effort. The human eye does not automatically detect points which are not C2. Humans can detect and locate a non-C2 point only by studying a curve closely. Thus, points which are not C2 do not generally detract from the visual appeal of a curve.

Having a curve be C2 at all points is mechanically important, however. If a car were driving around a curve with a non-C2 point, the driver would have to turn the steering wheel instantly from one position to another to stay on track at that point. An air molecule passing through such a point would have to have the force on it jump from one value to another.

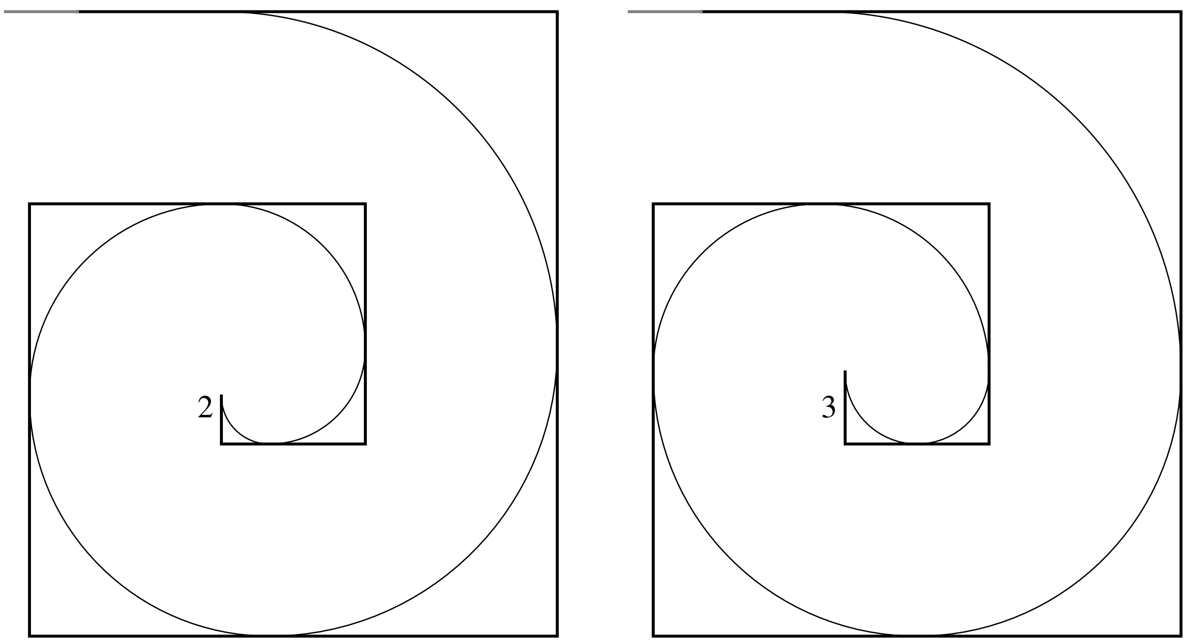
All points of arc-splines and all other splines are C0 points. All points of an arc-spline are C1, except those where a user has specified otherwise.

Arc-splines have points which are not C2 at every junction of an arc or line segment with the following arc or line segment, except if two successive arcs have the same radius. The use of arc-splines for defining shapes with mechanical purposes is likely to be limited for that reason. Their use for curves meant to please the eye should not be limited for that reason.

7. LOCALIZATION OF CONTROL POINT IMPACT

It is very desirable to localize the impact of each control point. This is done rather elegantly for some types of splines. When the arc-spline methods described here are applied to typical curves, such as might be found in a person's signature, the effect of a control point is usually localized fairly well; only a few adjacent parts of the contour outline will change when a control point is moved, removed, or added. This is because a break in the monotonicity of the lengths of frame arms stops propagation of the effect of the location of a point. The effect of moving one point may propagate the entire length of an outline, however, if the lengths of arms of the frame are monotonically increasing or decreasing. An example of this is shown in Figure 14.

Figure 14. Propagation of Effect of One Point



The two frames above are identical except for the length of the innermost arm. If the arms are numbered from the inside out, starting at 1, the length of each arm is 4n-2, which gives the sequence (2, 6, 10, 14, 18, ...), except that the length of the innermost arm on the right is 3. If radii are calculated either by the fair share method, or by making every radius be join_back, the radii on the left are in the sequence (2, 4, 6, 8, ..., 2n) while the radii on the right are in the sequence (3, 3, 7, 7, 11, 11, ...). These spirals can be extended indefinitely and never return into alignment.

The spline method of [MOSE] is an example of a method in which moving a single control point can make a radical change in the entire spline, as shown in Figure 13.

8. HANDLING OF SHARP CORNERS

All of the contour outline methods given in this paper handle sharp corners without undue effort. In the expert and easy arc-spline methods, the user is required to assign sharpness to a control point. A sharpness parameter is part of the data for each control point, while the value of the radius at a corner indicates sharpness in a frame.

9. CALCULATION TIME

The implementation of the methods described here is written in Franz LISP and runs uncompiled on a Sun 3/160 workstation with 8 Mbytes of RAM. The software has not been optimized for speed. Time trials of frame-rounding and arc-spline methods were run in a window environment with other processes. The timing results shown in Table 1 are from a wall clock and are not CPU times. These are typical times that a user would experience.

Table 1. Timing Results

Type of Operation	Number of Corners or Control Points	Number of Cases	Average Time (seconds)	Maximum Time (seconds)	Minimum Time (seconds)
Fix Radii by Fair Share	10 corners	8	8	11	4
Fix Radii by Fair Share	40 corners	4	34	35	32
Easy Method Construct Frame Fix Radii	10 control points	6	3 9	8 12	2 3
Easy Method Construct Frame Fix Radii	40 control points	3	20 33	25 35	13 30
Expert Method Construct Frame Fix Radii	10 control points	6	1 4	1 10	1 3
Expert Method Construct Frame Fix Radii	40 control points	3	5 15	10 15	3 15

10. MACHINABILITY

As stated at the outset, the techniques presented in this paper were developed with the requirement that machinable shapes be made. This has been achieved. Once the data structures for a contour outline have been created by these techniques, and enhanced as described in [KR&J], the automatic generation of numerical control code (NC code) which uses the outline as the path of the center point of a tool is easy, and has been implemented. Cutting along the left or right of an outline or varying the depth of cut while following an outline are feasible though not easy, and have also been implemented.

Spline methods which do not create circular arcs result in curves for which the generation of NC code is more of a problem. Linear interpolation to approximate the curve is not difficult, but requires a great deal of calculation and results in a very large amount of code, with the amount varying according to the tolerance required.

VI. APPLICATIONS

The contour outline methods described here have been applied in devising a design protocol [KR&J] and a system for automatic generation of NC-code [KR&W] for a 3-axis machining center. Three types of design features have been based on contour outlines. In addition, the text system described in [KRAM] allows placement of text along a contour outline.

The spline methods described here have been used for copying signatures and line drawings on a 3-axis machining center. The fidelity of the copies has been excellent. Figure 15 shows a picture of the signature of the author superimposed on a photograph of the machined copy of it made by using the expert arc-spline method.

Figure 15. Machined Signature

The figure shows the superimposed images of the author's signature and a copy of it engraved on an aluminum block. A viewgraph of the signature was made and pasted on the screen of a Sun 3 computer. Control points were entered with the mouse. The signature was copied with eight vee-bottomed contour grooves, the contour outlines for which were made by the expert arc-spline method. A total of 134 control points were used, resulting in contour outlines with a total of 81 corners. Process planning and NC-coding were done automatically by the VWS2 system. Machining was done in the VWS by the Monarch VMC-75 vertical machining center.

REFERENCES

[BEZI]
Bezier, P.; *Numerical Control: Mathematics and Applications*; John Wiley & Sons; New York; 1972.

[BOLT]
Bolton, K. M.; “Biarc Curves”; *Computer Aided Design*; 7; 1975; pp. 89 - 92.

[BROD]
Brodlie, K. W.; “A Review of Methods for Curve and Function Drawing”; *Mathematical Methods in Computer Graphics and Design*; edited by K. W. Brodlie; Academic Press; New York; 1980; pp. 1 - 37.

[CA&R]
Catmull, Edwin and Rom, Raphael; “A Class of Local Interpolating Splines”; *Computer Aided Geometric Design*; edited by Barnhill, Robert E. and Riesenfeld, Richard F.; Academic Press; New York; 1974; pp. 317 - 326.

[FA&P]
Faux, I. D. and Pratt, M. J.; *Computational Geometry for Design and Manufacture*; John Wiley & Sons; New York; 1979; pp. 153 -197.

[GO&R]
Gordon, William J. and Riesenfeld, Richard F.; “B-Spline Curves and Surfaces”; *Computer Aided Geometric Design*; edited by Barnhill, Robert E. and Riesenfeld, Richard F.; Academic Press; New York; 1974; pp. 95 - 126.

[KRAM]
Kramer, Thomas R.; *Enhancements to the VWS2 Data Preparation Software*; NISTIR 89-4201; 1989; National Institute of Standards and Technology.

[KR&J]
Kramer, Thomas R.; and Jun, Jau-Shi; *The Design Protocol, Part Design Editor, and Geometry Library of the Vertical Workstation of the Automated Manufacturing Research Facility at the National Bureau of Standards*; NBSIR 88-3717; 1988; National Bureau of Standards; pp. 63 - 66.

[MOSE]
Mosel, Eric G.; *Computer Assisted Engraving with N. C. Machine Tools*; Masters Thesis submitted to George Washington University; April, 1987.

[MORT]
Mortenson, Michael E.; *Geometric Modeling*; John Wiley & Sons; New York; 1985; particularly pp. 91 -150.