EXAMINATION OF THE CIRCLE SPLINE ROUTINE

Ronald M. Dolin and Dwight L. Jaeger

Los Alamos National Laboratory
Albuquerque, New Mexico

ABSTRACT

The Circle Spline routine was developed by D. L. Jaeger and is currently being used at the Los Alamos National Laboratory for generating both two- and three-dimensional spline curves. It was developed for use in ESCHER, a mesh generating routine written by W. R. Oakes, to provide a computationally simple and efficient method for building meshes along curved surfaces. Circle Spline is a parametric linear blending spline. Because many computerized machining operations involve circular shapes, the Circle Spline is well suited for both the design and manufacturing processes and shows promise as an alternative to the spline methods currently supported by the Initial Graphics Exchange Specification (IGES).

Circle Spline constructs a spline by generating a series of circular arcs, each of which passes through three successive data points (Pi-1, Pi, Pi+1), where i = 2,...,n and n = the number of data points. Thus, for n data points, n-2 arcs are constructed. These segmented arcs are then blended together, using a linear blending function, to represent the desired curve. Circle Splines need only have the data point coordinates specified. Because it is both a two- and three-dimensional routine, the three coordinates must always be specified, even for two-dimensional applications, in which case the third coordinate is set to zero. The spline passes through all the data points and these points do not need to be evenly spaced. Circle Spline does not require first or second derivative continuity, but instead has a different approach for determining aesthetic qualities such as smoothness and fairness.

The routine generates the spline by taking three successive data points (Pi-1, Pi, Pi+1), calculating the parametric location of a circle center $\underline{R}$, which passes through the points, and constructing an arc, of radius $\underline{R}$, through these points. With this information, the curved arc length between data points can be calculated. The value of the curved arc length for each interval is then stored. The routine then increments along the curve one data point and repeats the process for data points (Pi, Pi+1, Pi+2). With the exception of the two end point pairs (Pi, Pi+1) and (Pn-1, Pn), each pair of points (Pi, Pi+1) i = 2,...,n-2, will have two arcs passing through it. This means that each interval (i, i+1), i = 2,...n-2, has two values for the curved arc length (one for each arc that passes through it). The routine determines which of these two curved arc lengths is larger and retains that value.

Once all the arc lengths are calculated, the total arc length and parametric position of each data point are calculated. The routine uses a rational parameterization. If the sum of the curved arc lengths is S, the parametric value of each data point becomes the ratio of the spline length from the start of the curve to the data point divided by S. The parameter "u" will then range from $0 \leq u \leq 1$.

The use of the larger curved arc length within each interval was based on Circle Splines implementation in ESCHER. Better mesh point spacing is achieved using this scheme. A better parameterization may be achieved by using a function of the two arc lengths. In any event, using the curved arc lengths to calculate the total spline length no doubt leads to a better parameterization than the chordal arc length method used by most spline routines.

The two arc segments passing through each pair of data points are blended into a single curve. The user specifies the number of subpoints that will be used for the blending operation. These subpoints are evenly spaced parametrically along the curve. The code can also be used so that the subpoints are weighted, e.g., left, right, or center weighted. An automatic scheme whereby the subpoints are clustered in areas of varying curvature may produce a smoother blend.

The data points and the subpoints may be coalescent at points along the curve without affecting the spline. Because the data points are allowed to be unevenly spaced, the number of subpoints that fall within a data point interval may vary. The number of subpoints within an interval represents the number available for a particular blending operation.

Let arc-i represent the arc passing through points (Pi-1, Pi, Pi+1), and arc-i+1 represent the arc passing through points (Pi, Pi+1, Pi+2). Both arcs will pass through points Pi and Pi+1. For the purpose of example, suppose 10 subpoints lie within this interval. These 10 points will be evenly spaced along the interval. The routine will determine how far each subpoint is from points Pi and Pi+1 and based on this information will calculate the subpoints position relative to the two arcs. The closer a subpoint is to Pi, the closer it will be positioned to arc-i. As the subpoint placement approaches Pi+1, their positions will move closer to arc-i+1. At the start of an interval the blended curve is tangent to arc-i, at the end of the interval it is tangent to arc-i+1, in the center of the interval each arc will have an equal influence on the subpoint positioning, and the position of the blended spline will be in the geometric center of the plane defined by the two arcs.

If Pb equals the parametric value of a subpoint within the interval $Pi \leq Pb \leq Pi+1$, its position will be determined using the following equations

$$Ratio = (Pb - Pi) / (Pi+1 - Pi)$$

$$Ratm1 = 1 - Ratio$$

The value of the X-coordinate is thus,

$$XX(i) = Ratm1 * X(i) + Ratio * X(i+1)$$

where X(i) is the X-coordinate of Pi and X(i+1) is the X-coordinate of Pi+1. YY(i) and ZZ(i) arc calculated similarly. Using this scheme intuitively suggests that the blended curve can have, at most, one inflection point per interval.

Spline smoothness can be measured by examining how well the circle centers for each arc relate. Because curvature is the inverse of the radius, when the curvature is zero, R will approach infinity. Thus, the Rs provide an excellent way of monitoring the curvature. Many manufactured items contain circular shapes, for

example, holes, rounded edges, etc.; thus, the Circle Spline is well suited for Computer Aided Manufacturing (CAM).

The Circle Spline is computationally efficient and easy to understand. It was recently compared to the Wilson-Fowler spline in two dimensions; it compared favorably for most applications, but as might be expected, pathological cases existed where each routine produced better results. Circle Spline can be used in both two and three dimensions and its applicability to the design and manufacturing process makes its future use promising.