APPROXIMATION OF A CUBIC BEZIER CURVE BY CIRCULAR ARCS AND VICE VERSA

Aleksas Riškus
Department of Multimedia Engineering, Kaunas University of Technology
Studentu St. 50, LT–51368 Kaunas, Lithuania

Abstract. In this paper problem for converting a circular arc into cubic Bezier arc and approximation of cubic Bezier curve by a set of circular arcs are discussed. These questions occur in CAD/CAM systems during data exchange from data formats which support Bezier curves or during data exchange into data formats, which do not support Bezier curves. Some simple and practical solutions are proposed. An algorithm for approximation of a cubic Bezier curve and the results of its testing are presented.

Keywords: Bezier curve; control point; circular arc; approximation.

1. Introduction

Despite the fact that different CAD/CAM systems perform different layout editing operations and have some specific tasks (calculation of insulate channels, shape manipulation operations and so on), all they have some common features – they can import data in various data formats and later export the results in various data formats as well. These data formats (Gerber, GerberX, PDF, DXF, HPGL, ODB++, ISO 10303-210 and others) are standardized, but here is one big problem in the data interchange between them – not all curve types are supported in these formats.

There are no problems with the standard shape primitives like circle and rectangle. But shapes with arcs (open curve, closed curve, polygon) in some data formats have a few representations. A standard arc representation usually has coordinates of an arc start and end points and information about its center point (either as incremental distance from the arc start point as in Gerber format or center point coordinates as in PDF). The DXF, ODB++ and ISO 10303-210 formats additionally support the arc representation as Bezier curve. So, while performing the import/export operations in various formats we have to convert curves with standard circular arcs into curves with Bezier arcs and vice versa.

A cubic Bezier curve can approximate a circle but not perfectly fit a circle. A standard approach is to split a circle into four separate arcs. Errors of the approximation of a quarter of the circle (90 degree circular arc) have been analyzed in [3].

Approximation of cubic Bezier curve by a curve with circular arcs is a much more complicated task.

An algorithm for a cubic Bezier spiral (a curve whose curvature varies monotonically with arc-length) approximation is given in [7]. The algorithm is based on the recursive subdivision of the cubic Bezier spiral. But, in general, a Bezier curve is not naturally curvature continuous. It also can have cusps, loops and inflection points. In [7], the subdivision is performed at the point of maximum deviation of the spiral from the approximating biarc. In this paper, a few other subdivision techniques are proposed and their experimental characteristics are presented. The goal is to achieve the minimum number of approximating arcs.

The paper is organized as follows. Section 2 gives an introduction into Bezier curves – lists their main properties and subdivision techniques. Section 3 analyzes problems for converting a circular arc into cubic Bezier arc (arcs). Universal control points equations for an arbitrary circular arc up to 90 degree are presented. Section 4 investigates an inverse problem – approximation of cubic Bezier arc by a set of circular arcs. An algorithm for approximating a cubic Bezier arc with nondecreasing curvature is described and experimental comparison of six different subdivision strategies, developed for this approximation algorithm, are presented in Section 5. Finally, the paper is ended by some concluding remarks.

2. Bezier curves

Bezier curves were originally introduced by Paul de Casteljau in 1959. But they became a famous shape only when Pierre Bezier, French engineer at Renault, used them to design automobiles in the 1970’s. Bezier curves are now widely used in many fields such as
industrial and computer-aided design, vector-based drawing, font design (especially in PostScript font) and 3D modeling.

The most commonly used Bezier curves of third order are fully defined by four points: two endpoints \((P_1, P_4)\) and two control points \((P_2, P_3)\). The control points do not lie on the curve itself but define its shape [2, 8]. The curve, shown in Figure 1, starts at \(P_1\), goes toward \(P_2\) and arrives at \(P_4\) coming from the direction of \(P_3\).

![Figure 1. Cubic Bezier curve](image)

In general, it will not pass through \(P_2\) or \(P_3\). Such a curve is called a cubic Bezier curve. Its equation is [8]:

\[
B(t) = (1-t)^3P_1 + 3t(1-t)^2P_2 + 3t^2(1-t)P_3 + t^3P_4 \quad (1)
\]

Bezier equations are parametric equations in variable \(t\), and are symmetrical with respect to \(x\) and \(y\).

The parameter \(t\), varying in interval \([0, 1]\), cuts the segment \(P_1P_2\) into intervals, according to the wanted accuracy. When \(t = 0\), the result is \(B(0) = P_1\). For \(t = 1\), the result is \(B(1) = P_4\).

The Bezier curve is tangent to the segment of line \(P_1 - P_2\) at the start and \(P_3 - P_4\) at the end. The curve remains within the convex hull of the control points.

A possible approach, useful when dealing with the problem of converting a Bezier arc into 1 circular arc or set of circular arcs, is to sub-divide the Bezier curve into two sections and in each section approximate the curve by its control polygon. This process can be repeated on each sub-section until the control polygon for a sub-section is within some tolerance. The subdivision algorithm was devised in 1959 by Paul de Casteljau and is referred to as the de Casteljau algorithm [2, 8] (it is sometimes known as the geometric construction algorithm).

Let us consider the De Casteljau algorithm. Suppose that a cubic Bezier curve, defined over the parameter interval \([0, 1]\), is divided into two new cubic Bezier curves with corresponding parameter intervals \([0, \frac{1}{2}]\) and \([\frac{1}{2}, 1]\). This means that, from the original control points \(P_1\) to \(P_4\), we obtain new control points \(R_1\) to \(R_4\) and \(S_1\) to \(S_4\) of two Bezier curves that together make up the original curve. This process is illustrated in Figure 2.

![Figure 2. Illustration of the De Casteljau algorithm](image)

The new control points are obtained as follows:

\[
\begin{align*}
R_1 &= P_1 \\
S_1 &= R_2 \\
R_2 &= (P_1 + P_2) / 2 \\
R_3 &= R_2 / 2 + (P_2 + P_4) / 4 \\
S_2 &= (P_4 + P_2) / 4 + S_3 / 2 \\
R_4 &= (R_3 + S_4) / 2 \\
S_4 &= P_4
\end{align*}
\]

It is clear that the piecewise linear approximation \(P_1, R_2, R_4, S_2, S_4\) obtained after one such sub-division is a better approximation to the curve shape than the original control polygon \(P_1, P_2, P_3, P_4\). If this sub-division process is continued, then the piecewise linear polygon eventually collapses onto the curve. So, the cubic Bezier curve will be drawn.

The sub-division can be successfully applied for splitting an original cubic Bezier curve at any point (the split point is on the curve) and calculating control points for both new cubic Bezier curves.

Let us set the parameter \(t\) to any value \(k\) from the interval \([0, ..., k, ..., 1]\). Suppose that the \(C\) is corresponding sub-division point of the cubic Bezier curve. According to (1), we have \(P_1 = B(0), P_4 = B(1)\) and \(C = B(k)\). So, the resulting Bezier curves are \(P_1, R_2, R_4, S_2, S_4\), where their control points are:

\[
\begin{align*}
R_2 &= P_1 + k(P_2 - P_1) \\
S_2 &= P_2 + k^*(P_3 - P_1) \\
R_4 &= R_2 + k^*((P_4 - P_2) - P_3) \\
S_4 &= T + k^*(S_3 - (P_2 + k^*(P_3 - P_2)))
\end{align*}
\]

3. Converting a circular arc into Bezier arc

It is impossible to draw an absolutely exact circle with one Bezier curve. But we can approximate a unit quarter of a circle (90° arc) by a cubic Bezier curve with an error \(1.96 \times 10^{-4}\) in the radius [3], which is acceptable for most practical cases.

The approximation of a circle with four cubic Bezier curves is widely described in books for curves and surfaces (e.g., [8]) and in popular articles in the internet (e.g., [5]). To approximate, one should divide the circle into four arcs as it is shown in Figure 3 and convert each of them separately.

Let us consider only the upper right segment (the arc from point \(A\) to point \(B\)) shown in Figure 3, because we can convert other segments in the similar way (only some values will be negative). Since the angle \(AOB\) is of 90 degrees, the Bezier control line \(AA'\) is horizontal, and the Bezier control line \(BB'\) is vertical. The radius \(r\) of the circle is equal to the
length of the lines \( OA, OB \), as well as \( OC \). The point \( C \) is on the middle of the arc \( AB \), so the angles \( AOC \) and \( COB \) equal 45 degrees. The length \( d \) of \( AA' \) and \( BB' \) is unknown, however, it can be expressed as \( d = r \cdot k \), where \( k \) is a constant (in the literature this constant very often is called as “magic number”).

Let us assume that \( r = 1 \) and the coordinates of the center point \( O = [0, 0] \). In this case, \( d = k \), so the coordinates of the four points, defining the Bezier curve, are:

\[
A = [0, \frac{1}{2}]
\]

\[
A' = [k, \frac{1}{2}]
\]

\[
B' = [1, k]
\]

\[
B = [1, 0]
\]

From the definition of the cubic Bezier curve (1), we have:

\[
C(t) = (1-t)^3A + 3t(1-t)^2A' + 3t^2(1-t)B' + t^3B
\]

Since the point \( C \) lies at \( t = 0.5 \), \( (1-t) = 0.5 \) and the \( x \) coordinate of \( C \) equals the \( y \) coordinate of \( C \), we can write the following two equations for \( C \):

\[
C = \frac{1}{8}A + \frac{3}{8}A' + \frac{3}{8}B' + \frac{1}{8}B,
\]

\[
C = \sqrt{1/2} = \sqrt{2}/2.
\]

Solving the equations (3), (4) and (2) for \( C \) on the \( x \) axis (the same result would be for \( y \) axis as well) we obtain:

\[
0 + \frac{3}{8}k + \frac{3}{8} + \frac{1}{8} = \sqrt{2}/2
\]

\[
k = \frac{4}{3} (\sqrt{2} - 1) = 0.5522847498.
\]

So, the control points of the cubic Bezier curve for the upper right arc of a circle with radius \( r \) are:

\[
A = [0, r]
\]

\[
A' = [r, \frac{1}{2}]
\]

\[
B' = [r, k]
\]

\[
B = [r, 0].
\]

Consider an arc of less than 90 degree and radius \( r \). Assume that we have to approximate it by one segment of a cubic Bezier curve. As shown in Figure 4, the CW (clockwise) arc is centered along the positive \( x \) axis. The resulting Bezier curve connects \( P_1 \) and \( P_4 \) and its boundary tangents are collinear with the vectors \((P_1 - P_3)\) at the start point and \((P_4 - P_3)\) at the end point. The variation of tangent magnitude \( L \) is within the domain \([0, R \cdot \tan(q/2)]\), where \( q \) is half the angular width of the arc segment [4], i.e. \( q = \beta/2 \). We need to calculate the coordinates of the control points \( P_2 \) and \( P_3 \).

Let the coordinates of the arc start point \( P_1 \) and end point \( P_4 \) be \((x_1, y_1)\) and \((x_4, y_4)\), respectively. Then, from the elementary geometry, the coordinates of the cubic Bezier control points are:

\[
x_2 = x_1 + k \cdot R \cdot \sin(q)
\]

\[
y_2 = y_1 - k \cdot R \cdot \cos(q)
\]

\[
x_3 = x_4 + k \cdot R \cdot \sin(q)
\]

\[
y_3 = y_4 + k \cdot R \cdot \cos(q).
\]

For an arbitrarily positioned circle, operations of rotation, scaling and transformations are used.

As it was shown in [3], an approximation error in the radius varies from \(1.96 \times 10^{-4}\) to \(2.73 \times 10^{-4}\). In [3], it is mentioned that the magic number \(0.55191496\) provides the minimum error, the \(0.55228475\) value provides the maximum error.

By using a combined numerical and analytical approach, [4] made a conclusion that the least error occurs at three locations of a Bezier approximation curve – one is at \( t = 0.5 \) and the other two are symmetric to the mid-point of the curve \( (t = 0.18 \) and \( t = 0.82) \).

Another approach to finding Bezier control points, when the angle \( q \) directly does not participate in the calculation of the magic number, could be as follows. Let the coordinates of the arc start point \( P_1 \) are end point \( P_4 \) and arc center point \( C \) be \((x_1, y_1)\), \((x_4, y_4)\) and \((x_c, y_c)\), respectively. Then:

\[
a x = x_1 - x_c
\]

\[
ay = y_1 - y_c
\]

\[
hx = x_4 - x_c
\]

\[
by = y_4 - y_c
\]

\[
q_1 = ax*ax + ay*ay
\]

\[
q_2 = q_1 + ax*bx + ay*by
\]

\[
k_2 = \frac{4}{3} (\sqrt{2} \cdot q_1 \cdot q_2 - q_2) / (ax*by - ay*bx).
\]

The resulting coordinates of the Bezier control points \( P_2 \) and \( P_3 \) are:

\[
x_2 = xc + x_1 - k_2 * y_1,
\]

\[
y_2 = yc - y_1 + k_2 * x_1.
\]
4. Approximation of Bezier curve by circular arcs

An algorithm for a cubic Bezier spiral approximation by circular arcs is given in [7]. The spiral is a planar cubic Bezier curve segment whose curvature varies monotonically with arc-length (it does not have cusps, loops and inflection points).

The algorithm is based on the recursive subdivision of the cubic Bezier spiral at point of maximum deviation of the spiral from the approximation biarc within a given tolerance. The biarc (a pair of circular arcs) is constructed as follows [7]:

1. extend the tangent vector from Bezier start point \( P_1 \), extend the tangent vector from Bezier end point \( P_2 \) and find their intersection point \( V \);
2. calculate an incentre point \( G \) of the triangle \( (P_1, V, P_2) \), which defines the biarc (the incentre is the centre of the inscribed circle which touches the three sides of the triangle);
3. the biarc joining point \( G \) lies on the circle that passes through \( P_1, G \) and \( P_2 \).

Figure 5 illustrates the triangle \( (P_1, V, P_2) \), its incentre point \( G \) and biarc.

\[
\begin{align*}
y_2 &= y_1 + k_2 x_1, \\
x_3 &= x_4 - k_2 y_4, \\
y_3 &= y_4 + k_2 x_4.
\end{align*}
\] (9)

The advantage of (9) is that control points \( P_1 \) and \( P_2 \) are in absolute coordinates and any rotations and transformations are already not needed. For a counterclockwise arc, the value of \( k_2 \) is positive, for clockwise arc, it is negative. The correctness of the proposed approach for calculation of \( k_2 \) and control points \( P_3 \) was tested using the LPKF CircuitCAM software [10]. An absolute value of \( k_2 \) for 30, 45, 60 and 90 degree arcs was 0.175534, 0.265216, 0.357259 and 0.552284, respectively. An analysis of the approximation errors using (8) is out of the scope of this article.

The deviation is measured along a radial direction of the biarc. For 0.0001 mm tolerance, the maximum deviation is \( 0.97 \times 10^{-5} \).

Unfortunately, Bezier curve is not naturally curvature continuous. The curves, which are used in CAD/CAM systems for manufacturing of modern printed circuit boards (PCB), usually are composed of arcs and straight-line segments. Not only a curvature of each arc can be equal to a different constant, but the curvature does not vary monotonically within one arc-length. Approximation of cubic Bezier curves, which are circulars are by nature, does not create problems – the Bezier arcs are convex. But data, imported from the formats, which support Bezier curves or Bezier curves, which were modified in CAD/CAM systems during layout editing operations, cause some approximation problems. First, the angle of biarc of Bezier arc can be more than 90 degrees. Second, a Bezier curve can have cusps, loops and inflection points. Third, it can be quadratic, cubic or even higher degree Bezier curve. As regards the quadratic Bezier curves, their approximation is quite thoroughly investigated in [1, 6, 9]. According to [9], an arbitrary quadratic Bezier curve either has monotone curvature, or can be divided into two quadratic Bezier curve segments with monotone curvature, respectively. An algorithm for approximation of an arbitrary quadratic Bezier curve by arc splines is presented in [1] as well.

A full algorithm for approximating of an arbitrary cubic Bezier curve into curve with a set of circular arcs can be described as follows.

**Step 1:**
1) set the flag \( fCurvatureChanged = false \);
2) set the flag \( fFirstSegment = true \);
3) follow a current Bezier curve from its start point and find a point \( C \) where its curvature changes (convex segment changes into concave or vice versa);
4) if the point \( C \) was found, than go to Step 2.
5) current Bezier = entire Bezier curve; go to Step 3.

**Step 2:**
1) set \( fCurvatureChanged = true \);
2) subdivide the Bezier curve at the point \( C \);
3) current Bezier = first Bezier curve.

**Step 3:** Get the angle of the biarc of current Bezier arc.
If the angle is less than or equal to \( 90^\circ \), than go to Step 6.

**Step 4:** If the angle of the biarc is less than or equal to \( 180^\circ \) then:
1) subdivide the Bezier curve in two equal parts;
2) current Bezier = first Bezier;
3) set the flag \( fFirstSegment = true \);
4) go to Step 6.

**Step 5:** The angle of the biarc is more than \( 180^\circ \).
1)Subdivide the Bezier curve in such a way that the first (current) Bezier curve would be the first \( 90^\circ \) degree biarc segment.
2) set the flag \( fFirstSegment = true \).
Approximation of a Cubic Bezier Curve by Circular Arcs and Vice Versa

Step 6: Approximate the current Bezier curve by a set of circular arcs with given tolerance.

Step 7: If Step 5 was executed then:
1) current Bezier = second Bezier curve;
2) go to Step 3.

Step 8: If Step 4 was executed but Step 5 was skipped then:
1) current Bezier = second Bezier curve;
2) set fFirstSegment = false;
3) go to Step 6.

Step 9: If fCurvatureChanged = true then:
1) current Bezier = second Bezier curve;
2) set the flag fCurvatureChanged = false;
3) set the flag fFirstSegment = true;
4) go to Step 3.

Step 10: Stop.

Obviously, the core of the algorithm is Step 6 - “Approximate the current Bezier curve by a set of circular arcs with given tolerance”. As it was mentioned before, in the [7] a curve is split at the point of maximum deviation of the spiral from the approximating biarc. The deviation is measured along a radial direction of the biarc. We propose five approximation strategies (S1-S5) arbitrary (not only spiral) cubic Bezier curves.

All these strategies have the same basis: when the maximum deviation of the cubic Bezier curve from the approximating circular arc exceeds a given tolerance, than the Bezier curve is subdivided into two Bezier curves and the approximation algorithm is recursively used for both new Bezier curves. The differences are only in calculation of approximation arc (arc center point and radius) and in selection of the subdivision point (value of parameter t from interval [0, 1]).

The following two different approaches in calculation of the approximation arc were used:

A1-middle point: The approximation arc starts at the Bezier start point P1, goes through Bezier “middle point” M and ends at Bezier end point P2. According to equation (1), P1 = B(0), M = B(0.5), P2 = B(1).

Note: Our algorithm for approximation Steps 3 to 5 uses the biarc angle. Instead of this the approximation arc angle can be used as well.

A2-biarc: The approximation arc starts at the Bezier start point P1, goes through biarc joining point G and ends at Bezier end point P2.

To calculate the radius and center point of circular arc, use the well known mathematical fact that three points, which are not collinear (all on the same line), uniquely define a circle.

The five strategies for a subdivision point selection are as follows:

S1-middle point: The subdivision point is the middle point within the interval [0, 1], i.e., t = 0.5.

S2-cutting point: The subdivision point is a point where the Bezier curve intersects the approximation arc. If intersection did not occur, then t = 0.5.

S3-max error: The subdivision point is a point where the Bezier curve’s deviation from the approximation arc, measured along a radial direction of the arc, is maximum.

Note: This strategy within the approach A2 corresponds to the strategy used in [7].

S4-min error: The subdivision point is a point where the Bezier curve’s deviation from the approximation arc, measured along a radial direction of the arc, already exceeds the given tolerance.

S5-incentre point: The subdivision point is a point where the straight line through the approximation arc center point and biarc joining point G intersects the Bezier curve.

5. Computational experiments

We programmed and tested all five presented approximation strategies (S1-S5) in both A1 and A2 approximation arc calculation approaches. The experiment was done using the LPKF CircuitCAM software [10].

Tables 1 and 2 show the main characteristics of the Bezier curve approximation – the number of circular arcs and the maximum deviation from given tolerance – for Bezier curve, depicted in Figure 6 (unit of measurement for tolerance and deviation is mm). This initial cubic Bezier curve was defined by points P1=(16.9753, 0.7421), P2=(23.1643, 1.6148).

Figure 6. Bezier curve and its circular arcs

The initial angle of the arc approximating Bezier curve is 65 degrees (less than 90) and its curvature does not change. This means that any subdivisions in Steps 2, 4 and 5 of the algorithm will not be performed. The approximation with tolerance 0.001 mm is visually identical to the cubic Bezier curve. The dots in Figure 6 show the start and end points of the six approximating arcs.

The best result according to the main approximation criterion – minimum number of arcs – was reached by using strategy S2 in approach A1. The minimal deviation from tolerance for this combination is acceptable as well. The second result was shown by S1 strategy in both approaches A1 and A2. The worst strategy is S4.
Table 1. Approach A1 - approximating circle through Bezier middle point (t=0.5)

<table>
<thead>
<tr>
<th>Tolerance (mm)</th>
<th>S1 # of arcs and deviation (mm)</th>
<th>S2 # of arcs and deviation</th>
<th>S3 # of arcs and deviation</th>
<th>S4 # of arcs and deviation</th>
<th>S5 # of arcs and deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0,1</td>
<td>1 0.09122</td>
<td>1 0.09122</td>
<td>1 0.09122</td>
<td>1 0.09122</td>
<td>1 0.09122</td>
</tr>
<tr>
<td>0,01</td>
<td>3 0.002688</td>
<td>3 0.002688</td>
<td>3 0.008665</td>
<td>19 0.004908</td>
<td>3 0.002828</td>
</tr>
<tr>
<td>0,001</td>
<td>6 0.0008042</td>
<td>7 0.0003279</td>
<td>7 0.006311</td>
<td>158 0.0009435</td>
<td>8 0.0002378</td>
</tr>
<tr>
<td>0,0001</td>
<td>14 0.00005609</td>
<td>13 0.00004939</td>
<td>18 0.00007249</td>
<td>600 0.00008308</td>
<td>14 0.00008147</td>
</tr>
<tr>
<td>0,00001</td>
<td>28 0.000009069</td>
<td>27 0.000008428</td>
<td>34 0.00009485</td>
<td>1244 0.00008982</td>
<td>30 0.000007759</td>
</tr>
</tbody>
</table>

Table 2. Approach A2 - approximating circle through joining point G

<table>
<thead>
<tr>
<th>Tolerance (mm)</th>
<th>S1 # of arcs and deviation</th>
<th>S2 # of arcs and deviation</th>
<th>S3 # of arcs and deviation</th>
<th>S4 # of arcs and deviation</th>
<th>S5 # of arcs and deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0,1</td>
<td>1 0.09122</td>
<td>1 0.09122</td>
<td>1 0.09122</td>
<td>1 0.09122</td>
<td>1 0.09122</td>
</tr>
<tr>
<td>0,01</td>
<td>3 0.002688</td>
<td>3 0.002688</td>
<td>3 0.009194</td>
<td>10 0.002913</td>
<td>3 0.002828</td>
</tr>
<tr>
<td>0,001</td>
<td>6 0.0008042</td>
<td>6 0.0008042</td>
<td>9 0.0006025</td>
<td>107 0.0008818</td>
<td>8 0.0002378</td>
</tr>
<tr>
<td>0,0001</td>
<td>14 0.00005609</td>
<td>15 0.00009945</td>
<td>16 0.00007666</td>
<td>562 0.00009854</td>
<td>14 0.00008147</td>
</tr>
<tr>
<td>0,00001</td>
<td>28 0.000009069</td>
<td>29 0.000008888</td>
<td>28 0.00009796</td>
<td>2012 0.0009950</td>
<td>30 0.00007759</td>
</tr>
</tbody>
</table>

The second more complicated cubic Bezier curve is shown in Figure 7. It is defined by points $P_1 = (17.5415, 0.9003), P_2 = (18.4778, 3.8448), P_3 = (22.4037, -0.9109), P_4 = (22.563, 0.7782)$.

![Figure 7. Bezier curve and its circular arcs](image)

Because of changing curvature the first subdivision occurred in Step 2 at point $C = (20.9014, 0.9942)$. Let us take the first (left) segment. Because the biarc size of the current Bezier arc in Step 3 was 105 degrees, in Step 5 it was subdivided at point $L = (18.9614, 1.8617)$. The biarc size of the second segment was 95 degrees and according to Step 5 it was subdivided at point $R = (22.0377, 0.4328)$.

Incentre points of these four new Bezier curves (incentre point is used in approximation arc calculation approach A2) were: $G_1 = (18.1237, 1.686), G_2 = (19.5478, 1.686), G_3 = (21.7507, 0.5412), G_4 = (22.3907, 0.4912)$.

Tables 3 and 4, respectively, show the main results of approximation of the cubic Bezier curve displayed in Figure 7. The strategy S4 is rejected because of the worst results.

The approximation with tolerance 0.001 mm is visually identical to the cubic Bezier curve as well. The dots in Figure 7 show the start and end points of the 16 approximating arcs. The black square shows the point where curvature changes (point C).

Here we have the same result as that we obtained in previous experiment – strategy S1 in approximation arc calculation approach A1 and strategy S2 in both approaches are better than other variants.
Approximation of a Cubic Bezier Curve by Circular Arcs and Vice Versa

Table 3. Approach A1 - approximating circle through Bezier middle point (t=0.5)

<table>
<thead>
<tr>
<th>Tolerance (mm)</th>
<th>S1 # of arcs and deviation</th>
<th>S2 # of arcs and deviation</th>
<th>S3 # of arcs and deviation</th>
<th>S5 # of arcs and deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>4</td>
<td>0.02788</td>
<td>4</td>
<td>0.02788</td>
</tr>
<tr>
<td>0.01</td>
<td>7</td>
<td>0.008319</td>
<td>7</td>
<td>0.008319</td>
</tr>
<tr>
<td>0.001</td>
<td>16</td>
<td>0.0006906</td>
<td>16</td>
<td>0.0006906</td>
</tr>
<tr>
<td>0.0001</td>
<td>32</td>
<td>0.00009902</td>
<td>32</td>
<td>0.00009902</td>
</tr>
<tr>
<td>0.00001</td>
<td>71</td>
<td>0.000009450</td>
<td>73</td>
<td>0.000009799</td>
</tr>
</tbody>
</table>

Table 4. Approach A2 - approximating circle through joining point G

<table>
<thead>
<tr>
<th>Tolerance (mm)</th>
<th>S1 # of arcs and deviation</th>
<th>S2 # of arcs and deviation</th>
<th>S3 # of arcs and deviation</th>
<th>S5 # of arcs and deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>4</td>
<td>0.02788</td>
<td>4</td>
<td>0.02788</td>
</tr>
<tr>
<td>0.01</td>
<td>7</td>
<td>0.008319</td>
<td>7</td>
<td>0.008319</td>
</tr>
<tr>
<td>0.001</td>
<td>16</td>
<td>0.0006906</td>
<td>16</td>
<td>0.0006906</td>
</tr>
<tr>
<td>0.0001</td>
<td>32</td>
<td>0.00009902</td>
<td>32</td>
<td>0.00009902</td>
</tr>
<tr>
<td>0.00001</td>
<td>71</td>
<td>0.000009450</td>
<td>73</td>
<td>0.000009799</td>
</tr>
</tbody>
</table>

6. Concluding remarks

A problem of communication between CAD/CAM applications working with curve representations in different data formats was discussed. Some old but up to now very popular data formats, like Gerber, PDF, HPGL do not support splines and Bezier curves.

Possible solutions for converting of circular arc of up to 90 degrees into a cubic Bezier arc were analyzed and universal equations for calculation of control points were proposed and tested.

The problems of approximation of a cubic Bezier curve by circular arcs were discussed as well. As a result, an algorithm for approximation of an arbitrary cubic Bezier curve by circular arcs was described. Six different strategies for the basic algorithm step, Step 6 – “Approximate the current Bezier curve by a set of circular arcs with given tolerance”, were proposed and tested experimentally.

The proposed variations of the algorithms were tested using LPKF CircuitCAM software. One of the requirements for CAD/CAM software for the production of SMT (Surface mount technology) solder paste stencils is to minimize the number of segments while producing a shape (pad or track). Each additional line or arc segment (one additional stop and one additional move) is not acceptable for prototyping and laser machines, because it decreases a production quality.

The experiments showed that the best results were obtained by using S2 (cutting point) and S1 (middle point) strategies. We can see this from the empirical results presented in Figures 6 and 7. For strategy S2 in approach A1 (the approximation arc goes through points \( P_1 = B(0), M = B(0.5), P_4 = B(1) \)), applied for 0.0001 mm tolerance, the maximum deviations were 0.4939×10^{-4} and 0.9733×10^{-4}. The number of arcs is 13 and 32, respectively.

As it was mentioned in the definition of strategies S1 to S6, the strategy S3 within the approach A2 (approximation arc goes through the biarc joining point) corresponds to the strategy proposed in [7]. These results for the same samples are as follows: the maximum deviation is 0.7666×10^{-4} and 0.8949×10^{-4} and the number of arcs is 16 and 40, respectively.

In this paper we restricted our attention to cubic Bezier curves only. However, Bezier curves quadratic, quartic and higher degree can be used in CAD/CAM software as well.
References


Received October 2006.