Approximation of Smooth Planar Curves by Circular Arc Splines

Kazimierz Jakubczyk

May 30, 2010[*]

Abstract

The paper is concerned with the problem of approximating smooth curves in the plane by circular arc splines. The algorithm based on fitting a biarc to a segment of a given function or parametric curve of monotonic curvature is presented. The biarc formed of two tangentially joined circular arcs is constructed by assuming that it matches two end points and two tangents of the segment. However, such an interpolation problem imposes four conditions whereas the biarc has five degrees of freedom, and there exists a one parameter family of interpolating biarcs, which is searched iteratively by bisection method for finding a suitable one. If a given accuracy has not been achieved the segment is subdivided and this process is continued on smaller pieces. Finally, the fitting circular arc spline is obtained as the result of sequential applying the local biarc interpolation procedure.

Key words: biarc, arc spline, circular spline, circular arc spline, approximation by circular splines, interpolation by biarcs

Introduction

One of the basic problems in designing curved shapes in computer aided geometric design and geometric modeling is approximating of planar curves by circular arc splines with an accurate precision. Circular arc spline, in short circular spline or arc spline, is a curve composed of some number of tangentially joined circular arcs and straight line segments. We assume that straight line segment is a circular arc having an infinite radius. Of course, the curvature of each arc equals to a different constant. Thus, circular spline is only G^1 continuous (tangent continuity) and is not as excellent as cubic polynomial spline, which is G^2 continuous (curvature continuity) and is mathematical idealization of a mechanical spline, a thin elastic strip of flexible material, traditionally wood, used for many years by designers to smoothly interpolate a sequence of points.

However, circular spline curves have several nice properties. They are invariant to rotation, their tangent vector is defined everywhere and is continuous, their curvature is well defined as piecewise constant (step function), it is simple to compute arc length, a standard arc representation is composed of coordinates

[*]Rev. January 28, 2012, e-mail: `kazimierz@jakubczyk.net`

of the two arc end points and its center. Moreover, toolpaths for numerically controlled machining are almost always defined by piecewise linear and circular curves. Straight line segments and circular arcs are geometric primitives for generating so called G-code [8, 10] and NC program [14].

The main technique for converting of planar curves into G-code relies on approximating them by circular splines via biarc interpolation. A biarc is a pair of circular arcs having G^1 continuity (the same tangent vector) at their joint point. Biarc curves have been studied by many authors, see e.g. [1, 3, 4, 7, 8, 10, 12, 13, 14, 15] and the references cited therein. It has been generally assumed that biarcs were introduced in 1975 by Bolton [1], but it is not quite true, because a biarc interpolation algorithm was published in 1970 in an internal paper of polish design-research office of shipbuilding industry [3] and was applied in ship designing and production process [5].

Simple geometry of circular arc splines, and biarcs in particular, makes them attractive for solving of several significant issues, among others approximating discrete data [7, 15] and a given planar curve [4, 6, 8, 10, 12, 14], fairing and fitting of planar point set [13], generating a sweep surface in three dimensional space [11], representing boundaries of vector graphics texture in the space [9], and even solving of ordinary differential equations [2, 4]. Circular splines are still a very active area of research on computational geometry, numerical control and computer graphics.

In this paper we propose an efficient algorithm for computing the biarc that interpolates a segment of a given function or parametric curve of monotonic and of one sign curvature. If the segment cannot be approximated within a given accuracy it is subdivided into two smaller pieces, and next the first of them is interpolated. When the suitable biarc for the first piece has been found, this procedure is continued on the rest of the segment. Because two end points and tangents at the end points of each interpolating biarc are exactly the same as those in original segment, the whole circular spline has G^1 continuity.

In more general case, when the shape is described by a function or parametric curve of non-monotonic curvature, it usually can be subdivided into some number of pieces of monotonic curvature by determining its points of inflection and points of extreme curvature. In order to guarantee that each piece represents exactly some mathematical function, rotations and further partitions may be required. If the curvature of the shape has a finite number of points of discontinuity, such points determine extra partition. Converting each piece of this partition into circular spline yields to the curve made of circular arcs that is piecewise circular spline with cusps and sharp turns.

Computing of interpolating biarcs

As mentioned above, circular arc spline approximating a given smooth planar curve with an accurate precision can be found by applying biarc interpolation procedure. Assume that the curve is defined in the traditional x-y coordinate system as a function $y = y(x)$, and that its curvature is monotonic and does not change sign. Other cases can be usually recovered by subdivision the original curve into shorter curve segments and rotations some of them if necessary (see next section). Finding the subdivision points, particularly the points of extreme curvature, may turn out quite difficult and computationally expensive for the formula for the curvature is rather complicated:

$$\kappa(x) = \frac{y''(x)}{\sqrt{1+y'(x)^2}^3}. \tag{1}$$

The curvature is commonly used with the radius of curvature that is defined as

$$\rho(x) = 1/\left|\kappa(x)\right|.$$

Figure 1 illustrates a simple geometric way that leads to the biarc, which passes from the start point $P_0(x_0, y_0)$ to the end point $P_2(x_2, y_2)$ and is composed of two circular arcs of radii r_0 and r_1 sharing tangentially one common point $P_1(x_1, y_1)$. The biarc, to say that $y = s(x)$, interpolates the curve in the sense that it takes exactly the same two y-axis values and two tangent vectors at x_0 and x_2 as the function:

$$\begin{aligned} s(x_0) &= y_0 = y(x_0), \quad s'(x_0) = y'_0 = y'(x_0), \\ s(x_2) &= y_2 = y(x_2), \quad s'(x_2) = y'_2 = y'(x_2). \end{aligned} \tag{2}$$

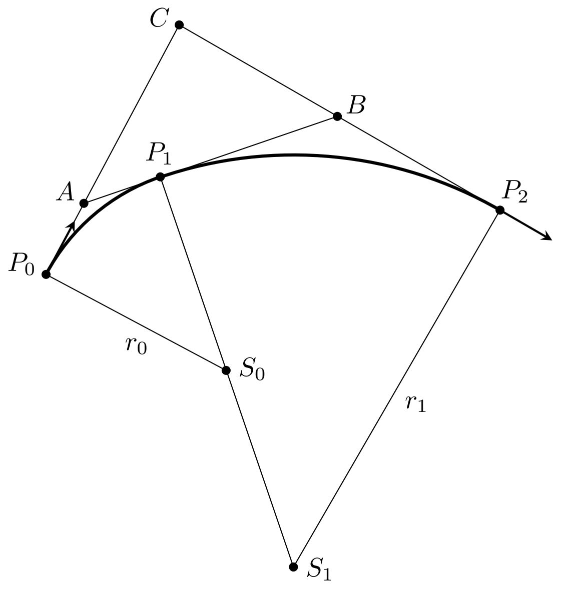

Figure 1: Biarc spline construction

Without loss of generality we assume now that the radius of curvature is monotonically ascending, i.e. the absolute value of the curvature is monotonically decreasing (another case is y-axially symmetric). Then $|P_0C| \leq |P_2C|$ and the conditions (2) determine two tangent lines to the curve at P_0 and P_2:

$$y - y_0 = y'_0(x - x_0), \quad y - y_2 = y'_2(x - x_2), \tag{3}$$

that have one common point $C(x_C, y_C)$ whose coordinates are:

$$\begin{aligned} x_C &= x_0 + \frac{y'_2(x_2 - x_0) - y_2 + y_0}{y'_2 - y'_0} = x_2 + \frac{y'_0(x_2 - x_0) + y_2 - y_0}{y'_2 - y'_0}, \\ y_C &= y_0 + y'_0(x_C - x_0) = y_2 - y'_2(x_2 - x_C). \end{aligned} \tag{4}$$

Let x_A be any real number between x_0 and x_C, i.e. $x_A \in (x_0, x_C)$, and let

$$y_A = y_0 + y_0'(x_A - x_0). \tag{5}$$

According to the first of (3), the point $A(x_A, y_A)$ lies on the segment P_0C. Now, let us draw a straight line through A defined by

$$y - y_A = m(x - x_A), \tag{6}$$

that intersects P_2C in some point $B(x_B, y_B)$. In view of (3), (5) and (6), we get

$$\begin{aligned} x_B &= x_A + \frac{y_2 - y_A - y_2'(x_2 - x_A)}{m - y_2'} = x_2 + \frac{y_2 - y_A - m(x_2 - x_A)}{m - y_2'}, \\ y_B &= y_A + m(x_B - x_A) = y_2 - y_2'(x_2 - x_B). \end{aligned} \tag{7}$$

In order to find joint point P_1 of the two circular arcs we first determine m in (6) such that $|P_0A| + |P_2B| = |AB|$, or more precisely

$$\begin{aligned} &\sqrt{(x_A - x_0)^2 + (y_A - y_0)^2} + \sqrt{(x_2 - x_B)^2 + (y_2 - y_B)^2} = \\ &\qquad = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2}. \end{aligned}$$

Combining this, (5) and the second of (7) yields

$$(x_A - x_0)\sqrt{1 + {y_0'}^2} + (x_2 - x_B)\sqrt{1 + {y_2'}^2} = (x_B - x_A)\sqrt{1 + m^2}.$$

Hence, the first of (7) and (5), as a result of strenuous algebraic transformations we obtain the following quadratic equation with respect to m:

$$m^2(D^2 - E^2) - 2mDF + F^2 - E^2 = 0,$$

where

$$\begin{aligned} D &= (x_A - x_0)\sqrt{1 + {y_0'}^2} + (x_2 - x_A)\sqrt{1 + {y_2'}^2}, \\ E &= y_2 - y_0 - y_0'(x_A - x_0) - y_2'(x_2 - x_A), \\ F &= y_2'D + \sqrt{1 + {y_2'}^2}. \end{aligned}$$

This equation is satisfied by unessential $m = y_2'$ that is excluded, and by

$$m = y_2' + 2E\frac{y_2'E + \sqrt{1 + {y_2'}^2}D}{D^2 - E^2}.$$

Now, the coordinates x_1 and y_1 of joint point P_1 of the biarc can be determined by the simple equation $|P_0A| = |P_1A|$ that is equivalent to

$$(x_A - x_0)\sqrt{1 + {y_0'}^2} = (x_1 - x_A)\sqrt{1 + m^2}.$$

Thus, by virtue of (6) and (5), we get

$$\begin{aligned} x_1 &= x_A + (x_A - x_0)\sqrt{\frac{1 + {y_0'}^2}{1 + m^2}}, \\ y_1 &= y_A + m(x_1 - x_A) = y_0 + (x_A - x_0)\left(y_0' + m\sqrt{\frac{1 + {y_0'}^2}{1 + m^2}}\right). \end{aligned} \tag{8}$$

Finally, we determine the centers $S_0(p_0, q_0)$ and $S_1(p_1, q_1)$, and the radii r_0 and r_1 of the two adjacent circular arcs forming the biarc. It is clear (see Fig. 1) that S_0 is a common point of a line perpendicular to P_0C passing through P_0, and a line perpendicular to AB passing through P_1. Similarly, S_1 is a common point of a line perpendicular to P_2C passing through P_2, and the same second line as in the first case. Of course, radii r_0 and r_1 are equal to $|S_0P_0|$ and $|S_1P_1|$, respectively. Converting (3) and (6) into equations, which represent these three lines, we compute

$$\begin{aligned} q_0 &= y_0 + \frac{x_1 - x_0 + m(y_1 - y_0)}{m - y_0'}, & q_1 &= y_2 - \frac{x_2 - x_1 + m(y_2 - y_1)}{m - y_2'}, \\ p_0 &= x_0 - y_0'(q_0 - y_0), & p_1 &= x_2 - y_2'(q_1 - y_2), \\ z_0 r_0 &= (q_0 - y_0)\sqrt{1 + {y_0'}^2}, & z_1 r_1 &= (q_1 - y_2)\sqrt{1 + {y_2'}^2}, \end{aligned} \tag{9}$$

where z_0 and z_1 denote the direction of the biarc, which is defined as 1 if it passes counterclockwise, and -1 if clockwise. More precisely, z_0 is the direction of the first arc passing from P_0 to P_1, and z_1 is the direction of the second arc passing from P_1 to P_2. Thus, according to the assumption that the curvature remains the same sign, both z_0 and z_1 are equal to 1 if the function is convex, and -1 if it is concave.

Formulae for parametric curve

The described above computational method for finding the interpolating biarc can be applied in the case when the function is defined as a parametric curve

$$x = x(t), \quad y = y(t), \quad t \in [t_0, t_2].$$

Then the first derivative of the function is given by

$$y'(x) = y'(t)/x'(t)$$

and all desired quantities are determined as

$$\begin{aligned} x_0 &= x(t_0), & y_0 &= y(t_0), & y_0' &= y'(t_0)/x'(t_0), \\ x_2 &= x(t_2), & y_2 &= y(t_2), & y_2' &= y'(t_2)/x'(t_2). \end{aligned}$$

Computing the interpolating biarc becomes a bit complicated if a curve segment does not represent any mathematical function, i.e. if two or more different y-axis values correspond to exactly one x-axis value. Then rotations and possibly some partitions are required. It is obvious that circular arcs and angles between any two vectors are invariant to rotations and translations, so the following affine transformation is quite accurate:

$$\begin{aligned} x^* &= (x - x_0)\cos\alpha + (y - y_0)\sin\alpha, \\ y^* &= -(x - x_0)\sin\alpha + (y - y_0)\cos\alpha, \end{aligned} \tag{10}$$

where

$$\sin\alpha = (y_2 - y_0)/d, \quad \cos\alpha = (x_2 - x_0)/d, \quad d = \sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2}.$$

In this new coordinate system, marked to star, suitable formulae reduce a little for the new coordinates of P_0 and P_2 satisfy

$$x_0^* = y_0^* = y_2^* = 0, \quad x_2^* = d.$$

Now that the biarc interpolating the rotated curve segment has been determined, it can be transformed to the original coordinate system by the following affine transformation, which is invariant to (10):

$$\begin{aligned} x &= x^* \cos\alpha - y^* \sin\alpha + x_0, \\ y &= x^* \sin\alpha + y^* \cos\alpha + y_0. \end{aligned}$$

Error estimation

In the sequel we assume that curve segment is given by $y = y(x)$ and that its curvature is monotonically decreasing and negative, just as shown in Figure 1 (other cases of monotonic and of one sign curvature can be recovered through the symmetry with respect to the x-axis, or y-axis, or both). The interpolation error is the maximal absolute value of error function defined as the difference between the curve interpolated and the biarc:

$$e(x) = y(x) - s(x), \quad x \in [x_0, x_2]. \tag{11}$$

A good estimate of the error can be obtained by sampling a set of points on (11), but computational cost of such a method is too expensive. A quite different way results from a simple analysis of the behaviour of the error function, which leads us to an efficient and practical method that computes the error by interpolating the error function by two cubic polynomial segments joined tangentially at x_1, and by finding their extreme values.

This is the fact that the interpolating biarc has five degrees of freedom, but it satisfies only four conditions (2). Which is why there exists a one parameter family of interpolating biarcs characterized by a single scalar parameter $x_A \in (x_0, x_C)$, and various choices of this parameter generate different biarc curves. If A moves left towards P_0, joint point P_1 also moves left towards P_0 and both radii of the biarc decrease, so the entire biarc converges to the degenerated form with $r_0 = 0$ (Fig. 2a). Similarly, if A moves right towards C, P_1 proceeds right to catch up with B and both radii of the biarc increase, so the entire biarc converges to the boundary biarc with $r_1 = \infty$ (Fig. 2b).

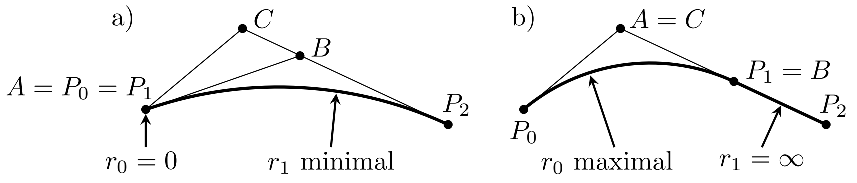

Figure 2: Degenerated and boundary biarc

We can view these situations with regard to the behaviour of the error function that depends on x_1 (see Fig. 3), and indirectly on x_A. If x_1 starts nearby x_0 and moves away to the right, both radii r_0 and r_1 of the two arcs forming the biarc are at the beginning lesser than the radii of curvature at x_0 and x_2, respectively, and increase, so that after a while r_0 becomes bigger than the radius of curvature at x_0, and in the end r_1 becomes bigger than the radius of curvature at x_2, or equal to if the curvature at x_2 is zero. Also, the minimal value of the error function starts out zero and, next, increases continuously with a negative sign, whereas the maximal value of the error function starts as positive and decreases to zero. This allows us to conclude that there exists the one and only x_1, for which the error function attains twice its maximal value with alternating sign, and that it is the smallest its maximal absolute value [4]. It just means that the interpolating biarc, which corresponds to such x_1, is optimal in the sense of it minimizes the y-axis distance from the curve.

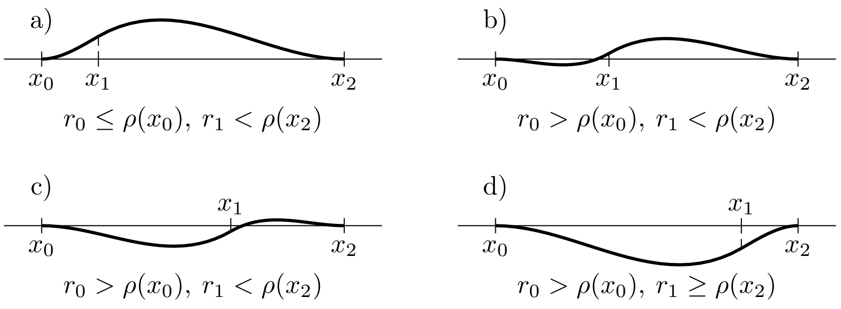

Figure 3: Four types of error function

The behaviour of the error function suggests that it can be well represented by two cubic polynomials defined on $[x_0, x_1]$ and $[x_1, x_2]$, to say that $y = w_0(x)$ and $y = w_1(x)$, such that they vanish with the first derivative at x_0 and x_2, respectively, and both take exactly the same value and tangent at x_1 as the error function. In practice such a representation of the error function is entirely acceptable [3, 4], and in theory it is satisfying for both polynomials are de facto Hermite interplants, of which the interpolation error is relatively quite small. It can be easy deduced that there exist $\alpha_0, \beta_0, \alpha_1$ and β_1 such that

$$w_0(x) = \alpha_0(x - x_0)^2(x - \beta_0), \quad w_1(x) = \alpha_1(x - x_2)^2(x - \beta_1). \tag{12}$$

Thus by interpolation conditions

$$w_0(x_1) = w_1(x_1) = e_1 = e(x_1), \quad w_0'(x_1) = w_1'(x_1) = e_1' = e'(x_1),$$

we obtain

$$\begin{aligned} \alpha_0 &= \frac{e_1'(x_1 - x_0) - 2e_1}{(x_1 - x_0)^3}, & \alpha_1 &= \frac{e_1'(x_2 - x_1) + 2e_1}{(x_2 - x_1)^3}, \\ \beta_0 &= x_1 - \frac{e_1(x_1 - x_0)}{e_1'(x_1 - x_0) - 2e_1}, & \beta_1 &= x_1 - \frac{e_1(x_2 - x_1)}{e_1'(x_2 - x_1) + 2e_1}. \end{aligned} \tag{13}$$

Of course, x_0 and x_2 are the extreme points of (12). We find another two, ξ_0 and ξ_1, by solving $w_0'(\xi_0) = 0$ and $w_1'(\xi_1) = 0$. Then

$$\xi_0 = (2\beta_0 + x_0)/3, \quad \xi_1 = (2\beta_1 + x_2)/3.$$

Finally, the interpolation error between the function, which has a monotonic and of one sign curvature, and the interpolating biarc curve can be estimated as the maximum of the two values computed as

$$d_0 = \begin{cases} |w_0(\xi_0)|, & \xi_0 \in (x_0, x_1), \\ 0, & \xi_0 \notin (x_0, x_1), \end{cases} \qquad d_1 = \begin{cases} |w_1(\xi_1)|, & \xi_1 \in (x_1, x_2), \\ 0, & \xi_1 \notin (x_1, x_2). \end{cases} \tag{14}$$

Note that if any of denominators in (13) for β_0 or β_1 equals to zero, then well-known trapezoidal rule, satisfied by polynomials of at most second degree, occurs. This enables us to set $\xi_0 = x_0$ or $\xi_1 = x_2$, respectively, and consistently to $d_0 = 0$ or $d_1 = 0$.

Biarc interpolation algorithm

As mentioned above, monotonic and of one sign curvature is a desirable property for curve segments used to approximate them into circular arc splines by biarc interpolating procedure. To satisfy this requirement each curve segment, of which the curvature does not vary monotonically, should be first subdivided into some smaller segments by determining its points of inflection and points of extreme curvature. Also, all computational formulae described above hold, provided that the curve segment is defined as a function, so rotations and other partitions can be required. Obviously, the interpolation error between the rotated curve segment and interpolating biarc is measured perpendicularly to the segment chord, not as y-axis distance in normal case.

We assume now that the curve segment is defined as a function $y = y(x)$ for $x \in [a, b]$, and that the absolute value of its curvature is monotonically decreasing on $[a, b]$ (other cases can be recovered by y-axis symmetry). Let $\epsilon > 0$ denote the error tolerance to be met. The biarc interpolation algorithm that converts such a curve segment into a circular arc spline with the maximum error $\le \epsilon$ can be described as follows:

```
input $a,\ b,\ \epsilon$
$x_2 \leftarrow a,\ y_2 \leftarrow y(a)$
while $x_2 < b$ do
    $x_0 \leftarrow x_2,\ y_0 \leftarrow y_2,\ x_2 \leftarrow b,\ y_2 \leftarrow y(b)$
    $bigError \leftarrow true$
    while $bigError$ do
        compute $x_C$ using the first of (4)
        $d \leftarrow (x_C - x_0)/2,\ x_A \leftarrow x_0 + d$
        $nextChoice \leftarrow true$
        while $nextChoice$ do
            compute $x_1$ and $y_1$ using (8), $d_0$ and $d_1$ using (14)
            if $d_0 \le \epsilon$ and $d_1 \le \epsilon$ then                { error $\le \epsilon$ }
                compute $p_0,\ q_0,\ r_0,\ z_0,\ p_1,\ q_1,\ r_1,\ z_1$ using (9)
                output $x_0,\ y_0,\ p_0,\ q_0,\ r_0,\ z_0,\ x_1,\ y_1,\ p_1,\ q_1,\ r_1,\ z_1$
                $nextChoice \leftarrow false,\ bigError \leftarrow false$
            else
                if $d_0 \ge \epsilon$ and $d_1 \ge \epsilon$ then            { error $> \epsilon$ }
                    $x_2 \leftarrow (x_0 + x_2)/2,\ y_2 \leftarrow y(x_2)$
                    $nextChoice \leftarrow false$
        else
            $d \leftarrow d/2$
            if $d_0 > \epsilon$ then                    $\{ \ d_0 > \epsilon, \ d_1 < \epsilon \ \}$
                $x_A \leftarrow x_A - d$
            else                                  $\{ \ d_0 < \epsilon, \ d_1 > \epsilon \ \}$
                $x_A \leftarrow x_A + d$
            end if
        end if
      end if
    end while
  end while
end while
output $x_2, \ y_2$
```

As can be observed, first the whole interval $[a, b]$ is regarded as $[x_0, x_2]$, and on each interval $[x_0, x_2]$ the interpolating biarc is searched iteratively by bisection method in order to reduce the interpolation error. If it becomes evident that the given tolerance cannot be achieved, the current interval $[x_0, x_2]$ is subdivided into two equal subintervals and the local biarc interpolation procedure is repeated for the first of them, regarded as new $[x_0, x_2]$, until the interpolation error can be acceptable. Then the suitable biarc is determined and the interpolation procedure is continued on the rest of $[a, b]$, i.e. on $[x_2, b]$ that is regarded as new $[x_0, x_2]$.

Example

We consider now an example of cubic polynomial spline that describes a part of a real water-line of a ship. The curve is shown in Figure 4. It is represented in the form

$$y(x) = c_0x^3 + c_1x^2 + c_2x + c_3 + \sum_{i=1}^{7} c_{i+3}(x - x_i)_+^3, \tag{15}$$

where

$$(x - x_i)_+ = \begin{cases} 0 & (x - x_i \le 0) \\ x - x_i & (x - x_i > 0) \end{cases} \qquad (i = 1, 2, \ldots, 7)$$

and all joint points and coefficients are given in Table 1.

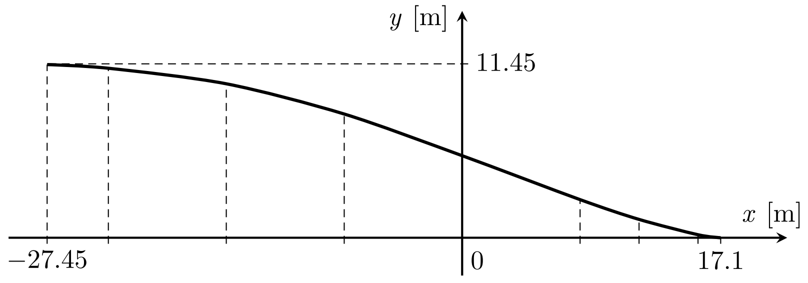

Figure 4: Prow part of a water-line (based on [3])

i	x_i	c_i
0	-27.45	0.001529536
1	-23.40	0.105035831
2	-15.60	2.308940510
3	-7.80	27.322024800
4	0.00	-0.001846248
5	7.80	0.000450526
6	11.70	0.000103903
7	15.60	-0.000045882
8	17.10	0.000374533
9		-0.000195034
10		0.024243101

Table 1: Joint points and coefficients of a water-line (based on [3])

It can be easily noticable that each curve segment between two neighboring joint points is a cubic polynomial, and any two neighboring segments are joined at common joint point in G^2 manner, meaning that the curve is continuous everywhere with its first and second derivatives. Though the curve is composed only of eight cubic segments and its total length exceeds 44 meters, it has been entirely useful. For some specific tasks in ship designing and production process it has been converted into circular arc spline by applying biarc interpolating algorithm presented in previous section. A question arose which was, how to decompose the curve into some number of pieces of monotonic curvature, i.e. how to determine its points of inflection and points of extreme curvature?

In general, such a problem may turn out quite difficult, but in cubic case, just as the considered, it can be solved separately for each curve segment. Thus, without loss of generality we assume now that the curve is given by

$$y(x) = a_0x^3 + a_1x^2 + a_2x + a_3 \quad (a_0 \neq 0) \tag{16}$$

($a_0 = 0$ has been omitted for simplicity). Then

$$y'(x) = 3a_0x^2 + 2a_1x + a_2, \quad y''(x) = 6a_0x + 2a_1, \quad y'''(x) = 6a_0. \tag{17}$$

Equating the second of (17) to zero, we get the point of inflection

$$\eta_p = -a_1/3a_0. \tag{18}$$

We determine now the desired points of extreme curvature. Differentiating (1) and equating the obtained expression for the first derivative of the curvature to zero, we get

$$y'(3y''^2 - y'y''') - y''' = 0. \tag{19}$$

Combining (17) and (19) yields algebraic equation of the fourth degree

$$\begin{aligned}45a_0^3x^4 + 60a_0^2a_1x^3 + 2a_0(13a_1^2 + 6a_0a_2)x^2 + \\ +4a_1(a_1^2 + 2a_0a_2)x + a_2(2a_1^2 - a_0a_2) - a_0 = 0,\end{aligned}$$

which has only two real solutions

$$\eta_1 = \eta_p - \delta, \qquad \eta_2 = \eta_p + \delta, \tag{20}$$

where

$$\delta = \sqrt{\frac{3\sqrt{a_1^2(a_1^2 - 6a_0a_2) + a_0^2(9a_2^2 + 5)} + 2(a_1^2 - 3a_0a_2)}{45a_0^2}}.$$

Of course, both η_1 and η_2 are the extreme points of the curvature of cubic polynomial given by (16), and they are symmetrical with respect to the inflection point η_p.

Returning to our water-line (15), it can be verified by numerical computation using (18) and (20) for each of the eight cubic polynomial segments that only one segment, defined on $[0, 7.8]$, should be subdivided by its inflection point $\eta_p = 1.8329$. Thus, in order to convert the whole curve into circular spline we subdivide it into nine pieces instead of eight. The obtained results for 1 mm tolerance are illustrated in Table 2.

i	x_i	y_i	p_i	q_i	r_i	z_i
0	−27.4500	11.4500	−27.4500	−13.9772	25.4272	−1
1	−26.5341	11.4335	−27.8768	−25.8199	37.2775	−1
2	−25.4250	11.3769	−28.4509	−34.5293	46.0058	−1
3	−24.6375	11.3182	−34.0537	−101.8901	113.5992	−1
4	−23.4000	11.2084	−38.7318	−151.5528	163.4817	−1
5	−21.1644	10.9823	−30.8764	−78.8735	90.3790	−1
6	−19.5000	10.7867	−29.0966	−64.8466	76.2397	−1
7	−17.4025	10.4909	−26.0901	−45.4778	56.6389	−1
8	−15.6000	10.1812	−25.9299	−44.6277	55.7739	−1
9	−13.7135	9.7918	−27.8705	−53.2725	64.6338	−1
10	−11.7000	9.3058	−29.1648	−58.2811	69.8070	−1
11	−9.8259	8.7936	−32.8370	−71.0180	83.0627	−1
12	−7.8000	8.1815	−35.8885	−80.6707	93.1862	−1
13	−6.0287	7.6020	−49.4811	−120.8535	135.6058	−1
14	−3.9000	6.8621	−59.7401	−149.5987	166.1268	−1
15	−2.3225	6.2901	−129.8578	−339.9683	368.9989	−1
16	0.0000	5.4259	−223.8841	−590.0584	636.1805	−1
17	0.6108	5.1959	−690.5747	−1827.5012	1958.7026	−1
18	1.8329	4.7344	377.7545	999.4908	1063.4178	1
19	4.8245	3.6090	78.6765	200.8253	210.5906	1
20	7.8000	2.5202	51.7727	125.5513	130.6532	1
21	9.7732	1.8327	30.4774	62.8212	64.4071	1
22	11.7000	1.2121	26.8353	50.8715	51.9146	1
23	13.7757	0.6263	23.6160	38.4857	39.1174	1
24	15.6000	0.1985	19.5963	19.2864	19.5017	1
25	16.0581	0.1083	17.4992	7.9196	7.9431	1
26	16.3500	0.0600	17.2694	6.3476	6.3545	1
27	16.7611	0.0135	17.1000	4.2369	4.2369	1
28	17.1000	0.0000				

Table 2: Circular spline representation of a water-line with 0.001 tolerance (based on [3])

As can be seen, the curve has been approximated by circular spline composed of only 28 arcs. Some of them have large radii, which is why encoding them into G-code has involved linear interpolation for drawing and cutting machines could not move its tool, pen or cutter, along a circular arc of a radius larger than 104 m [3].

Conclusions

The algorithm described in the paper enables circular arc spline approximation of smooth planar curve within a desired tolerance. It accepts any curve segment of monotonic and of one sign curvature, and is based on biarc interpolating procedure. The explicit formulae for biarc construction and tight estimate of interpolation error are derived. The biarc interpolating procedure searches iteratively a one parameter family of interpolating biarcs to reduce the interpolation error and to find the suitable biarc. It is convexity preserving and can be turned to preserve monotonicity of curvature of the original shape.

The proposed method is simple in concept and acceptable in computation. The algorithm runs in practice, it is used to fit circular splines to classical cubic polynomial splines. The paper provides an example of such a practical application in shipbuilding.

References

[1] Bolton K. M., Biarc curves, *Computer Aided-Design* 7 (1975), 98-92.

[2] Jakubczyk K., Approximation by Circular Splines for Solution of Ordinary Differential Equations, *Applicationes Mathematicae* XVI, 2 (1978), 284-292.

[3] Jakubczyk K., *Circular Approximation of Ship Curves in ASTER System* [in Polish], Information Processing Center of Shipbuilding Industry, XPS-026, COKB Gdańsk, March 1970.

[4] Jakubczyk K., *The Applications of Circular Spline Functions in Computer Computations* [in Polish], Doctoral thesis, Silesian Technical University, Gliwice 1978.

[5] Juranek J., ASTER – Preparatory and Control System in Processing of Hull Construction Elements [in Polish], *Informatyka* 2 (1971), 13-17.

[6] Marciniak K., Putz B., Approximation of spirals by piecewise curves of fewest circular arc segments, *Computer Aided-Design* 16 (1984), 87-90.

[7] Meek D. S., Walton D. J., Approximating discrete data by G^1 arc splines, *Computer Aided-Design* 24 (1992), 301-306.

[8] Meek D. S., Walton D. J., Approximating smooth planar curves by arc splines, *Journal of Computational and Applied Mathematics* 59 (1995), 221-231.

[9] Quin Z., Kaplan C., Mc Cool M., *Circular Arcs as Primitives for Vector Textures*, Technical report CS-2007-41, School of Computer Science, University of Waterloo, http://www.cgl.uwaterloo.ca/~zqin/jgt2007/submitted_cr.pdf.

[10] Šir Z., Feichtinger R., Jüttler B., Approximating Curves and Their Offsets using Biarcs and Pythagorean Hodograph Quintics, *Computer Aided-Design* 38 (2006), 608-617.

[11] Wang W., Joe B., Robust computation of the rotation minimizing frame for sweep surface modeling, *Computer Aided-Design* 29 (1997), 379-391.

[12] Yang X., Efficient circular arc interpolation based on active tolerance control, *Computer Aided-Design* 34 (2002), 1037-1046.

[13] Yang X., Wang G., Planar point set fairing and fitting by arc splines, *Computer Aided-Design* 33 (2002), 35-43.

[14] Yeung M. K., Walton D. J., Curve fitting with arc splines for NC toolpath generation, *Computer Aided-Design* 26 (1994), 845-849.

[15] Yong J. H., Hu S. M., Sun J. G., A note on approximation of discrete data by G^1 arc splines, *Computer Aided-Design* 31 (1999), 911-915.

* * *