ELSEVIER

PII: S0010-4485(96)00028-0

*Computer-Aided Design*, Vol. 28, No. 12, pp. 951–959, 1996

0010-4485/96/$15.00 + 0.00

# An optimization approach for biarc curve-fitting of B-spline curves

C J Ong, Y S Wong, H T Loh and X G Hong

We present an approach to the optimal fitting of a biarc-spline to a given B-spline curve. The objective is to minimize the area between the original B-spline curve and the fitted curve. Such an objective has obvious practical implications. This approach differs from conventional biarc curve-fitting techniques in two main aspects and has some desirable features. Firstly, it exploits the inherent freedom in the choice of the biarc that can be fitted to a given pair of end-points and their tangents. The conventional approach to biarc curve-fitting introduces additional constraints, such as the minimal difference in curvature or others to uniquely determine successive biarcs. In this approach, such constraints are not imposed. Instead, the freedom is exploited in the problem formulation to achieve a better fit. Secondly, the end-points do not lie on the curve so that appropriate tolerance control can be imposed through the use of additional constraints. Almost all previous biarc-fitting methods consider end-points that are on the original curve. As a result of these two aspects, the resulting biarc curve fits closely to the original curve with relatively fewer segments. This has a desirable effect on the surface finish, verification of CNC codes and memory requirement. Numerical results of the application of this approach to several examples are presented. Copyright © 1996 Elsevier Science Ltd

**Keywords: optimal biarc-spline, biarc curve-fitting, CNC curve machining**

## INTRODUCTION

The control modules in numerically controlled machines usually only provide straight line and/or circular interpolations. General curves for machining are therefore approximated by collections of linear and/or circular arc segments to some given tolerance. Research in this area has been active[1–9]. Linear segments are often used because of their simplicity in implementation and computational efficiency. The resultant surface finish is, however, poor because of the lack of smoothness in piecewise linear curves. Simple circular arc fitting algorithms that do not consider tangential ($GC^1$) continuity at junction points suffer from a similar lack of smoothness, though to a lesser extent. One popular choice that ensures $GC^1$ continuity between adjoining segments is the biarc.

Several works on curve fitting using biarcs have been reported[1–4,6,7,10]. Given two end-points and their tangential directions, it is well known that the biarc which passes through these end-points and directions is not unique. Most of the biarc fitting algorithms focused on the choice and characterization of an additional constraint that will uniquely determine the biarc. For example, Bolton[1] and Su and Liu[4] look for the biarc which minimizes the difference of the radii of the two component circular arcs. Others, like Moreton and Parkinson[6] and Meek and Walton[7], consider a constraint which minimizes the integral of the square of the curvature along the biarc. It is argued that this minimal solution also minimizes the strain energy of an imaginary elastic spline which satisfies the end-point and tangent constraints. Other variations of this idea exist[2].

In biarc curve-fitting of general curves, there are many biarc segments. The common approach is to discretize (by some method) the continuous curve into a grid of points (with the points lying on the curve), and fit biarcs to successive pairs of points. Additional constraints like those mentioned earlier are imposed to uniquely determine each of these biarcs. Such as approach looks at the curve-fitting problem from a 'local' point of view. As a result, many segments are needed for a good fit. A more general approach is to look at the curve-fitting problem as a whole and consider a criterion which takes into account the freedoms in all of the biarc segments. In this paper, we introduce a formulation of the curve-fitting problem which minimizes the absolute area between the original and the fitted curves. It has several advantages: the objective function allows for the post-machining or finished operations needed; the formulation exploits the freedom in the choice of biarc determination to achieve a better fit; the resultant biarc-spline has fewer segments (for the same tolerance) as the grid points are not confined to lie on the original curve.

Tolerance control is an essential part of the machining process. For example, operations such as the machining of aerofoils, turbine blades and marine propellers do not permit undercutting into the workpiece due to aerodynamical requirements. Additional examples of overcut

Department of Mechanical and Production Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore 0511
*Paper received: 21 August 1995. Revised: 13 March 1996*

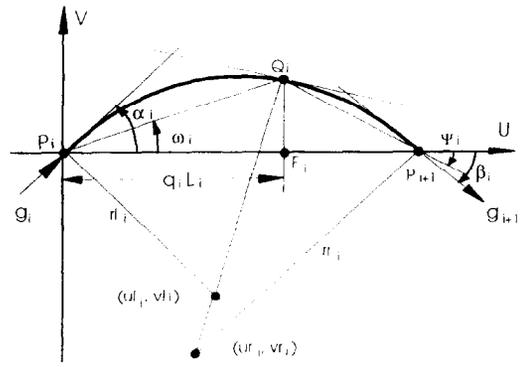


**Figure 1** The C-shaped biarc

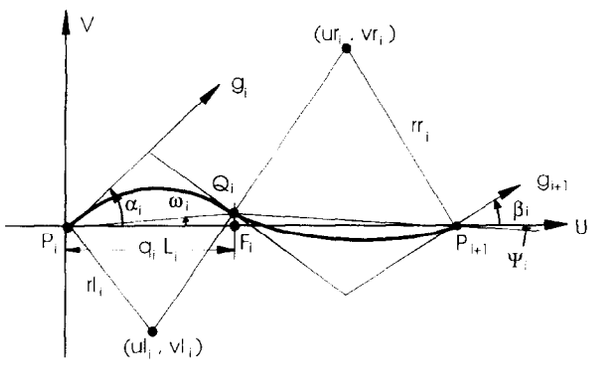


**Figure 2** The S-shaped biarc

and undercut requirements of workpieces are commonly found in the precision machining of parts to be assembled or fitted together. It is advantageous to take into consideration this tolerance factor of the machining process during the curve fitting phase. Our formulation allows easy specification of such overcut and/or undercut requirements. These requirements are likely to be satisfied since the grid points are not confined to be on the original curve.

We will adopt the following notations throughout this paper. The $n$-dimensional Euclidean space is $R^n$. $\mathbf{x} \in R^n$ is a $n$-dimensional vector with $x_i$ being the $i$ th element of the vector. The derivative of a function $C(\bullet)$ with respect to its argument is denoted by $C'(\bullet)$. $\|\bullet\|$ denotes the Euclidean distance of a vector.

## THE BIARC MODEL

We review the biarc model in $R^2$. It is well known that there are two types of biarc models: the C-shaped (see *Figure 1*) and the S-shaped biarcs (see *Figure 2*). The derivation given here is applicable to both types of biarc. For ease of presentation, we describe the biarc model using a local coordinate system. In the formulation of the curve-fitting problem, a global coordinate system is needed. The conversion from one coordinate system to another is easily achieved by direct transformation.

Let the two end-points of the biarc be $\mathbf{P}_i$ and $\mathbf{P}_{i+1}$ (see *Figures 1* and *2*). The tangents at these two points are denoted by $g_i$ and $g_{i+1}$. In $R^2$, $g_i$ and $g_{i+1}$ can also be represented by the angles that they make with the axes of the local or the global coordinate frames. Coordinates of the centres of the left and right circular arcs are $(ul_i, vl_i)$ and $(ur_i, vr_i)$ with radii $rl_i$ and $rr_i$, respectively. The common chord between $\mathbf{P}_i$ and $\mathbf{P}_{i-1}$ has a length of $L_i$. The common tangent point $\mathbf{Q}_i$ can be represented by a parameter $q_i$. Several choices of this parameter exist. In our case, we make use of the projected distance of the point $\mathbf{Q}_i$ onto the common chord. When measured from $\mathbf{P}_i$, this projected distance is $l_i$. Our choice of the parameter is $q_i = l_i/L_i$. Hence, $0 \leq q_i \leq 1$. (The range of $q_i$ may be smaller for specific ranges of $g_i$ and $g_{i+1}$[4].) This choice of parameter provides explicit relations for the other parameters of the biarc. The derivation of these relations are provided in Appendix A. These relations are:

$$\alpha_i = \tan(g_i)$$

$$\beta_i = \tan(g_{i+1})$$

$$\omega_i = \arctan\left(\frac{\sqrt{((2\sin 0.5(\alpha_i - \beta_i))^{-1})^2 - (0.5 - q_i)^2} - 0.5[\tan(0.5(\alpha_i - \beta_i))]^{-1}}{q_i}\right)$$

$$\Psi_i = \arctan\left(\frac{\sqrt{((2\sin 0.5(\alpha_i - \beta_i))^{-1})^2 - (0.5 - q_i)^2} - 0.5[\tan(0.5(\alpha_i - \beta_i))]^{-1}}{1 - q_i}\right)$$

$$\begin{aligned}
L_i &= \|\mathbf{P}_{i-1} - \mathbf{P}_i\| \\
rl_i &= \frac{q_i \cdot L_i}{2 \cdot \sin(\alpha_i - \omega_i) \cdot \cos(\omega_i)} \\
ul_i &= \frac{q_i \cdot L_i \cdot \sin(\alpha_i)}{2 \cdot \sin(\alpha_i - \omega_i) \cdot \cos(\omega_i)} \\
vl_i &= \frac{-q_i \cdot L_i \cdot \cos(\alpha_i)}{2 \cdot \sin(\alpha_i - \omega_i) \cdot \cos(\omega_i)} \\
rr_i &= \frac{(1 - q_i) \cdot L_i}{2 \cdot \sin(\beta_i - \Psi_i) \cdot \cos(\Psi_i)} \\
ur_i &= \frac{(1 - q_i) \cdot L_i \cdot \sin(\beta_i)}{2 \cdot \sin(\beta_i - \Psi_i) \cdot \cos(\Psi_i)} \\
vr_i &= -\frac{(1 - q_i) \cdot L_i \cdot \cos(\beta_i)}{2 \cdot \sin(\beta_i - \Psi_i) \cdot \cos(\Psi_i)}
\end{aligned} \tag{1}$$

It is easy to see, from Equation 1, that the biarc is uniquely defined only when $L_i$, $q_i$, $g_i$ and $g_{i+1}$ are known. For a given set of $\mathbf{P}_i$, $\mathbf{P}_{i+1}$, $g_i$ and $g_{i+1}$, the choice of biarc is not unique as there is additional freedom in $q_i$. Indeed, there are six parameters needed to define two circles (coordinates of the centres and the radii). Specifying $\mathbf{P}_i$, $\mathbf{P}_{i-1}$, $g_i$ and $g_{i-1}$ imposes only five constraints: each circle passes through one data point $\mathbf{P}_i$; each circle has a prescribed tangent $g_i$ at $\mathbf{P}_i$; the two circles must have a

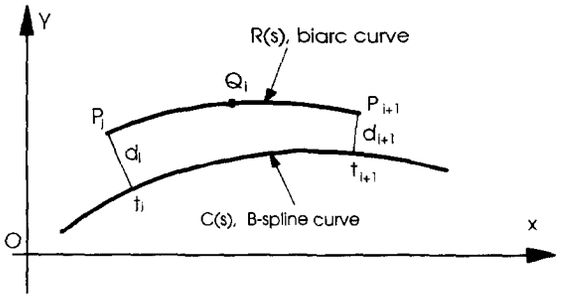


**Figure 3** The location of the biarc

common tangent at the point of intersection. This fact and other properties of the biarc are given in References 1 and 4.

## FORMULATION OF THE BIARC FITTING PROBLEM

In this work, we assume that the curve to be fitted with biarc segments is given by an explicit or implicit expression. With the increasing use of CAD packages, such an assumption is reasonable. For example, curves and surfaces in CAD systems are typically represented by B-spline, Bézier curves or primitive objects. All of these have explicit or implicit mathematical expressions. In the case where only discrete data are available, it is also easy to approximate these data points by, say, a B-spline curve. The fitting is usually good for most surfaces with errors kept to within a tight tolerance. The formulation given in this paper can be easily extended to all of these curves although we have used B-spline for the ease of discussion.

Let the given curve $\mathbf{C} : [0, 1] \rightarrow R^2$ be parameterized by a scalar $s$ and suppose that we wish to fit $(n-1)$ biarc segments to $\mathbf{C}(s)$. In loose terms, the curve-fitting problem is to find the best 'locations' of the $(n-1)$ biarc segments which will minimize the absolute area between $\mathbf{C}(s)$ and the biarc curve subject to some maximum and minimum deviations.

For ease of presentation, we denote the biarc curve (with $(n-1)$ biarc segments) by $\mathbf{R}(s)$ having the same parameter $s \in [0, 1]$. The unit tangent vector of $\mathbf{C}(s)$ at $s = t_i$ is given by $\left.\frac{\mathbf{C}'(s)}{\|\mathbf{C}'(s)\|}\right|_{s=t_i}$ and the unit normal vector is $\mathbf{u}(s)|_{s=t_i} = \left.\frac{\mathbf{C}''(s)}{\|\mathbf{C}''(s)\|}\right|_{s=t_i}$. Suppose the end-points of the $(n-1)$ segments are $\mathbf{P}_i$, $i = 1, 2, \ldots, n$. The coordinates of $\mathbf{P}_i$ are parameterized in terms of $t_i$ and $d_i$ where $d_i$ is the distance taken along the normal vector $\mathbf{u}(s)$ at $s = t_i$. Specifically, $\mathbf{P}_i(t_i, d_i)$ means the coordinates of $\mathbf{P}_i$ are $\mathbf{C}(s)|_{s=t_i} + d_i \cdot \mathbf{u}(s)|_{s=t_i}$. *Figure 3* shows the locations of $\mathbf{P}_i(t_i, d_i)$ and $\mathbf{P}_{i+1}(t_{i+1}, d_{i+1})$ for a typical curve. Since $s$ is between 0 and 1, so must $t_i$. Ideally, $d_i$ can take any real value. In practice, the choice of $d_i$ is often restricted by the allowable maximum and minimum deviations. These constraints are typically imposed by tolerance requirements of the machining operations and other design considerations. For example, one can impose $d_{\min} \leq d_i \leq d_{\max}$ for $i = 1, \ldots, n$ as constraints with $d_{\max} \geq d_{\min} \geq 0$ to enforce all $\mathbf{P}_i$ to stay 'above' $\mathbf{C}(s)$.

The remaining parameters for the biarc segments are the points of common tangency, as determined by the value of $q_i$s, and the tangents $g_i$s at the end-points. Let the collection of these $t_i$s, $d_i$s, $q_i$s and $g_i$s be $\mathbf{t} = [t_1, t_2, \ldots, t_n]$, $\mathbf{d} = [d_1, d_2, \ldots, d_n]$, $\mathbf{q} = [q_1, q_2, \ldots, q_n]$ and $\mathbf{g} = [g_1, g_2, \ldots, g_n]$. *Figure 4* illustrates the connection between the biarc-spline curve and the variables $(\mathbf{t}, \mathbf{d}, \mathbf{q}, \mathbf{g})$. It is also clear from *Figure 4* that $t_1 = 0$ and $t_n = 1$ for any $\mathbf{C}(s)$. So, let $\mathbf{t} = [t_2, t_3, \ldots, t_{n-1}]$ and $(\mathbf{t}, \mathbf{d}, \mathbf{q}, \mathbf{g})$ define the $(n-1)$ biarcs.

The curve-fitting algorithm is formulated as an optimization problem:

$$\text{min} \qquad A(\mathbf{t}, \mathbf{d}, \mathbf{q}, \mathbf{g}) \tag{2}$$

$$\text{subject to} \qquad h(\mathbf{t}, \mathbf{d}, \mathbf{q}, \mathbf{g}; s) \leq 0 \tag{3}$$

The function $A(\mathbf{t}, \mathbf{d}, \mathbf{q}, \mathbf{g})$ is a function (hereafter called the area function) related to the absolute area between $\mathbf{R}(s)$ and $\mathbf{C}(s)$. The optimization variables or the design variables are $(\mathbf{t}, \mathbf{d}, \mathbf{q}, \mathbf{g}) \in R^{4n-3}$ with $\mathbf{t} \in R^{n-2}$, $\mathbf{q} \in R^{n-1}$ and $\mathbf{d}, \mathbf{g} \in R^n$. The inequality $h(\mathbf{t}, \mathbf{d}, \mathbf{q}, \mathbf{g}; s)$ in Equation 3 is the collection of all possible constraints of the system like upper and lower bounds on $\mathbf{t}, \mathbf{d}, \mathbf{q}$ and $\boldsymbol{g}$.

## IMPLEMENTATION ISSUES

The area function computes the absolute area between $\mathbf{R}(s)$ and $\mathbf{C}(s)$ segment by segment. There are several ways to compute this absolute area numerically. One way is to convert the integration of the area in the $x$–$y$ plane into a line integration through the use of Green's

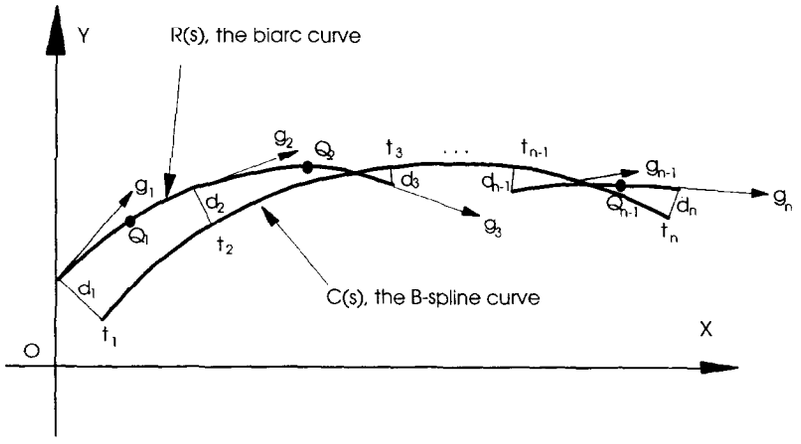


**Figure 4** Optimal biarc-spline curve-fitting model

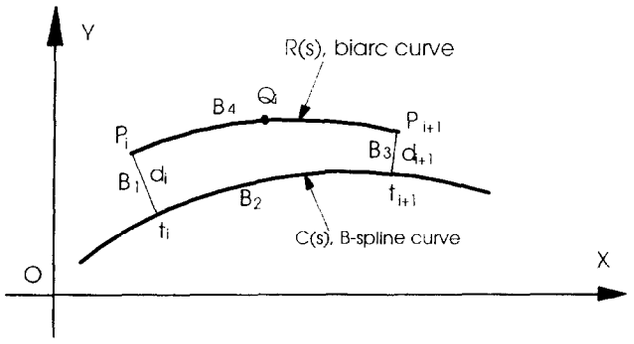


**Figure 5** The area Green's line integration

theorem[11]. It is well known that the area of a closed and bounded region $D$ can be computed as a line integral over its closed boundary $B$. For example, *Figure 5* depicts one typical biarc segment and the corresponding portion of $\mathbf{C}(s)$ between $s = t_i$ and $s = t_{i+1}$. It can be shown that the absolute area is:

$$A = \int\int_D \mathrm{d}x\mathrm{d}y = \oint_B x\mathrm{d}y = \oint_B x(\sigma)\frac{\mathrm{d}y(\sigma)}{\mathrm{d}\sigma}\mathrm{d}\sigma \tag{4}$$

In this example, the line integral over $B$ can be further decomposed into four distinct regions $B_i, i = 1, \ldots, 4$ as shown in *Figure 5*. In each region, the expressions of $x(\sigma)$ and $\mathrm{d}y(\sigma)/\mathrm{d}\sigma$ are known and the integrand can be evaluated exactly. For example, $x(\sigma)$ and $\mathrm{d}y(\sigma)/\mathrm{d}\sigma$ in region $B_2$ can be obtained from the expression of $\mathbf{C}(s)$ over an appropriate range of $s$. The same is true for region $B_4$. Expressions for $x(\sigma)$ and $y(\sigma)$ for regions $B_1$ and $B_3$ are also available since they have lengths of $d_i$ and $d_{i+1}$, respectively, with known starting and ending coordinates.

In general, it is possible that $\mathbf{R}(s)$ intersects $\mathbf{C}(s)$ at several places within each segment. (If $\mathbf{C}(s)$ is a B-spline, a maximum of six intersections in each arc segment is possible[4]. In practice, the number of intersection points is much fewer.) These points of intersection have to be known since Equation 4 is applicable in a region where there is no intersection. We omit the details but mention that these points of intersection can be obtained by standard root-finding algorithms. The computation of these roots are fast because $\mathbf{C}(s)$ and $\mathbf{R}(s)$ are low-order polynomials. The root-finding algorithm, together with Equation 4 constitute the area function.

It is also possible to use other numerical integrations like higher-order quadrature for the evaluation of Equation 4. In this case, the absolute area $A$ is

$$A = \int\int_D \mathrm{d}x\mathrm{d}y = \int(\mathbf{R}(s) - \mathbf{C}(s))\mathrm{d}s \tag{5}$$

We approximate Equation 5 using the Simpson's rule with $m$ grid points. Of course, the accuracy of the integration depends on the fineness of the grid, or the value of $m$. This value of $m$, in turns, depends on the accuracy of the approximation needed. We used a standard procedure of obtaining $m$ via an iterative process. More exactly, suppose $A^k$ is the area evaluated using Simpson's rule for Equation 5 based on $k$ grid points. We successively double the grid points and evaluate the area function $A^{2^j k}$ for $j = 0, 1, 2, \ldots$ until $|A^{2^j k} - A^{2^{j-1} k}| \leq \epsilon$ for any given small number $\epsilon > 0$. Let the smallest $j$ such that $|A^{2^j k} - A^{2^{j-1} k}| \leq \epsilon$ is satisfied be $\kappa$ then the desired grid is $m = 2^\kappa \cdot k$.

We have implemented both the exact and the Simpson's rule integration (for $\epsilon = 10^{-5}$). In all of the examples, the difference is less than 0.015%. While the numerical integration is easier to implement and does not require the determination of intersection points within each segment, the computation effort is higher than the exact integration method. Our experience with the examples described in later sections shows that the increase in computational effort varies from a factor of 2.5 to 6 (for $\epsilon = 10^{-5}$).

Inequality **h** in Equation 3 represents the collection of all the constraints on the curve-fitting problem. One example of these constraints mentioned earlier is $d_{\min} \leq d_i \leq d_{\max}$ for $i = 1, \ldots, n$. These constraints enforce the end-points $\mathbf{P}_i$ to stay within a desirable range. Ideally, the overcut or undercut requirements are applicable to the entire biarc curve, i.e. we want $d_{\min} \leq ||\mathbf{R}(s) - \mathbf{C}(s)|| \leq d_{\max}$ for $s \in [0, 1]$. These constraints can be included in Equation 3. The resultant problem becomes a semi-infinite optimization problem since the objective function depends on finite dimensional variables but the constraints are applicable for all $s \in [0, 1]$. It is known that the analytical approach to these problems can have particular difficulties[12]. We avoid these problems by requiring the constraints to be met on fixed grid points along $\mathbf{R}(s)$. Specifically, we impose the constraints on $p$ intermediate points within each biarc. When these constraints are added, the curve-fitting problem becomes

$$\min \quad A(\mathbf{t}, \mathbf{d}, \mathbf{q}, \mathbf{g}) \tag{6}$$

subject to

$$q_{\min} \leq q_i \leq q_{\max} \quad \text{for } i = 1, 2, \ldots, n-1 \tag{7}$$

$$g\min_i \leq g_i \leq g\max_i \quad \text{for } i = 1, 2, \ldots, n \tag{8}$$

$$d_{\min} \leq d_i \leq d_{\max} \quad \text{for } i = 1, 2, \ldots, n \tag{9}$$

$$d_{\min} \leq \left|\left|\mathbf{R}\left(t_i + j\frac{t_{i+1} - t_i}{p+1}\right) - \left(\mathbf{C}(t_i + j\frac{t_{i+1} - t_i}{p+1}\right)\right|\right| \leq d_{\max} \quad \text{for } j = 1, 2, \ldots, p, i = 1, 2, \ldots, n-1 \tag{10}$$

$$t_i \leq t_{i+1} - \delta \quad \text{for } i = 1, 2, \ldots, n-1 \tag{11}$$

Inequalities 7 and 8 are needed to ensure the validity of Equation 1; constraints 9 and 10 refer to the upper and lower deviation constraints; inequality 11 ensures that $t_{i+1}$ is strictly greater than $t_i$. All in, there are $(2 \cdot p + 7)n - 3 - 2 \cdot p$ inequality constraints. There are several choices for the numerical solution of Equations 6–11. We use an optimization package called NLPQL[13] which is based on the sequential quadratic programming method for solving non-linear optimization problems. It is implemented in FORTRAN77 and is developed by Schittkowski.

The performance of the numerical optimization procedure is closely related to the initial guess of the optimal solution. It is well known that a good initial guess can have a significant speedup in the convergence of the algorithm. While it is easy to come up with simple initial guesses, they often have a high value of $n$. We describe a systematic way of choosing an initial guess with the intention of getting as few segments as possible. The attempt here is to get as long a biarc as the

constraints allowed. We start with $t_1 = 0$, $d_1 = 0.5$ $(d_{\min} + d_{\max})$ and increase $t_2$ from 0 while keeping $d_2 = 0.5(d_{\min} + d_{\max})$ until some part of this first biarc hits the upper or lower constraints. More exactly, we let $t_2 = \rho$ where $\rho$ is a small positive number and increase $t_2$ by doubling $\rho$ successively. This process is repeated as long as the upper and lower constraints are not violated. Suppose one upper or lower limit constraint is violated after $Z$ iterations ($t_2 = 2^Z\rho$). Golden section search is used in the region between $2^Z\rho$ and $2^{Z-1}\rho$ to determine a more accurate $t_2$. This golden section search is terminated when a sufficiently small bracket is found and the value of $t_2$ is chosen to be the lower end of this bracket. The size of the bracket is relatively large since only an initial guess is needed. This process is then repeated for the next choice of $t_i$ and so on until the entire curve is covered. Note that checking of the violation of constraints is accomplished through the use of a fine discretization of $\mathbf{C}(s)$. The $q_i$s are all initialized to a value of 0.5. The initial guess to $g_i$ is equal to the tangent of the B-spline at the corresponding end-points, i.e. $g_i = \left.\frac{\partial \mathbf{C}(s)}{\partial s}\right|_{s=t_i}$. This initial guess to $g_i$ is also used to define the limits $\text{gmin}_i$ and $\text{gmax}_i$. Specifically, $\text{gmin}_i = \left.\frac{\partial \mathbf{C}(s)}{\partial s}\right|_{s=t_i} - \Delta g$ and $\text{gmax}_i = \left.\frac{\partial \mathbf{C}(s)}{\partial s}\right|_{s=t_i} + \Delta g$ for some $\Delta g > 0$. The above procedure provides values for $n$, $\text{gmin}_i$ and $\text{gmax}_i$ as well as initial guess to $\mathbf{t}, \mathbf{d}, \mathbf{q}$, and $\mathbf{g}$.

## RESULTS AND DISCUSSIONS

The need to compare performances of the optimization algorithm arises in many places in this section. To this end, we define some measures of performance that, in some sense, indicate how well the algorithm is performing.

We compare results before and after the optimization process on many occasions. Let $\mathbf{R}^b(s)$ and $\mathbf{R}^a(s)$ be the biarc curves before and after the optimization process, respectively. We also need to distinguish the cases where $\mathbf{R}(s)$ is 'above' or 'below' $\mathbf{C}(s)$. For this purpose, we define

$$\Delta^b(s) = ||\mathbf{R}^b(s) - \mathbf{C}(s)|| \text{ if } (\mathbf{R}(s) - \mathbf{C}(s)) \bullet \mathbf{u}(s) > 0$$
$$= -||\mathbf{R}^b(s) - \mathbf{C}(s)|| \text{ if } (\mathbf{R}(s) - \mathbf{C}(s)) \bullet \mathbf{u}(s) < 0$$

**Table 1** Values of parameters for the optimal biarc-spline fitting

| $d_{\min}$ | $d_{\max}$ | $q_{\min}$ | $q_{\max}$ | $\Delta g$ | $\delta$ | $p$ |
|---|---|---|---|---|---|---|
| −0.05 | 0.05 | 0.1 | 0.9 | 10 | 0.00005 | 7 |

$\Delta^b(s)$ shows the deviation between $\mathbf{R}^b(s)$ and $\mathbf{C}(s)$ as a function of $s$. The function $\Delta^a(s)$ is similarly defined using $\mathbf{R}^a(s)$. The maximum and minimum of $\Delta(s)$ over $s \in [0, 1]$ are denoted by $\Delta_{\max}(s)$ and $\Delta_{\min}(s)$. Hence, $\Delta^a_{\max}(s)$ is the maximum deviation of $\Delta^a(s)$ or $\Delta^a_{\max}(s) = \max_{s\in[0,1]} \Delta^a(s)$ and $\Delta^a_{\min}(s)$ is the minimum deviation of $\Delta^a(s)$ or $\Delta^a_{\min}(s) = \min_{s\in[0,1]} \Delta^a(s)$.

Another measure is the ratio of the absolute areas. Specifically, we define $c_{\text{area}} = A^a/A^b$ where $A^b$ and $A^a$ are the absolute areas between $\mathbf{R}(s)$ and $\mathbf{C}(s)$ curves before and after optimization, respectively. In addition, we also define $c_i = (\Delta^a_{\max} - \Delta^a_{\min})/(\Delta^b_{\max} - \Delta^b_{\min})$ to indicate, on a percentage basis, the amount of maximum deviation due to the optimization process.

The first numerical example is that of a cam. The exact equations of the profile are given in polar coordinates in Appendix B.

We discretize the profile of the cam uniformly using 800 points. A cubic B-spline is then used to approximate these points. The purpose is to simulate a curve obtained from discrete data points. This B-spline curve is then fitted with a Biarc spline using the procedure described in previously section. The parameters used for the optimization problem 6–11 are given in *Table 1*.

We have used $n = 11$ as determined from the initialization procedure. The results are depicted in *Figure 6*. Since it is difficult to distinguish the biarc curve from the original curve when they are superimposed on the same graph, we magnify the error 100 times. The biarc curves in *Figure 6* are $\bar{\mathbf{R}}^a(s)$ and $\bar{\mathbf{R}}^b(s)$ where $\bar{\mathbf{R}}^a(s) = \mathbf{C}(s) + 100(\mathbf{R}^a(s) - \mathbf{C}(s))$ and $\bar{\mathbf{R}}^b(s) = \mathbf{C}(s) + 100(\mathbf{R}^b(s) - \mathbf{C}(s))$. *Figure 7* depicts the functions $\Delta^b(s)$ and $\Delta^a(s)$ against $s$ on the same plot. It is easy to see the advantage gain. It is of interest to note that there is a region ($s \in [0.4, 0.6]$) where $\Delta^a(s)$ is zero. This is possible since this portion corresponds to a circular arc (from Equation B1 where $-0.25\pi \le \alpha \le 0.25\pi$). It also

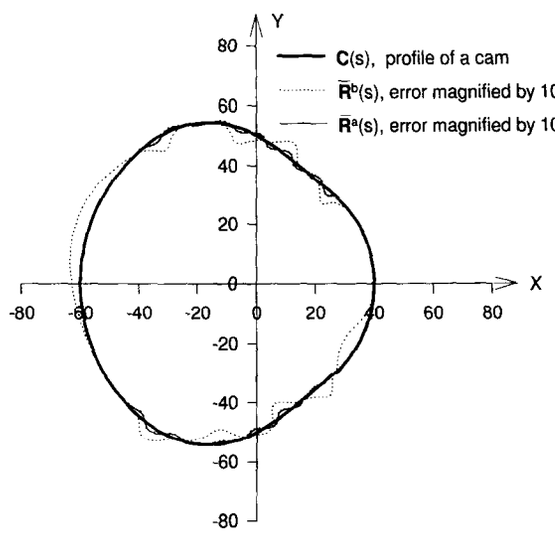


**Figure 6** The biarc segments distribution along the B-spline curve before and after optimization for the cam example

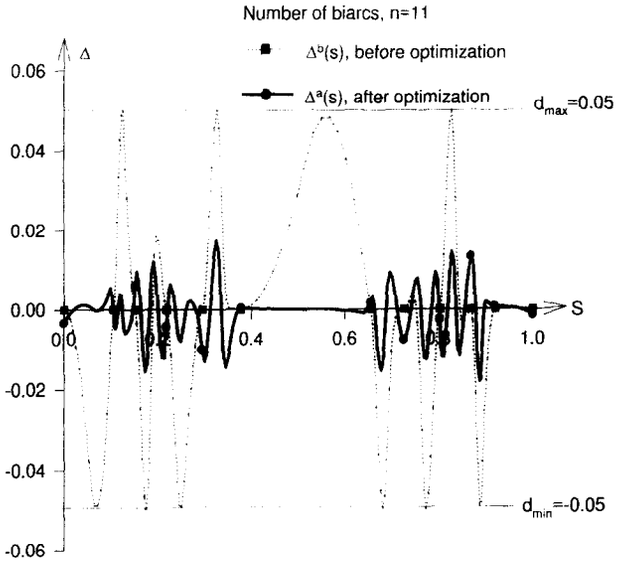


**Figure 7** The fitting error distribution along $s$ before and after optimization for the cam example

**Table 2** Values of $d_{min}$ and $d_{max}$ for additional examples

| Curves | Cases | $d_{min}$ | $d_{max}$ | $n$ |
|---|---|---|---|---|
| Parabola | I | −0.05 | 0.05 | 8 |
| | II | 0 | 0.05 | 9 |
| Sine | I | −0.05 | 0.05 | 11 |
| | II | 0.05 | 0.10 | 12 |

shows the effectiveness of the algorithm in achieving its objective.

Another interesting feature is the lack of symmetry in $\mathbf{R}^a(s)$. While $\mathbf{C}(s)$ is symmetrical about the $x$ axis, $\mathbf{R}^a(s)$ is not. This effect is common among numerical optimization and shows the influence of $\mathbf{R}^b(s)$ on $\mathbf{R}^a(s)$. Since $\mathbf{R}^b(s)$ is not symmetrical, the optimization algorithm searches for a local minimum in the vicinity of the initial guess. The resultant minimum is also non-symmetrical. This effect is also evident in *Figures 6* and 7.

Two other examples are included. The first is a parabola while the second is a sine curve. The formulae for these two curves are shown in Appendix B. The parabola is chosen as it represents a special class of quadratic function while the sine curve can only be represented exactly by an infinite number of biarcs. Their inclusion in this discussion is to show the versatility of the algorithm as well as the effectiveness of the algorithm when subject to various tolerance controls. The parameters used for Equations 6–11 are similar to those of the cam example as given in *Table 1* except for the values of $d_{min}$ and $d_{max}$. For each example, two set of values of $d_{min}$ and $d_{max}$ are investigated. These two set of values for both examples are given in *Table 2*.

The values of $n$ for all the cases in the two examples using the initialization procedure are also indicated in *Table 2*. The functions $\Delta^b(s)$ and $\Delta^a(s)$ against $s$ of these examples are illustrated in *Figures 8–11*.

A summary of the measures of performance of the three examples is given in *Table 3*. In general, the $c_{area}$ factor is in the range of 19.47–27.8% (excluding Sine II case). This represents a reduction in the absolute area of

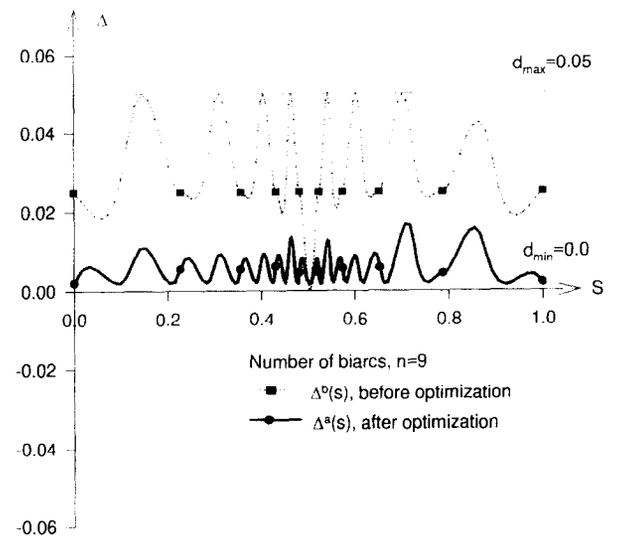


**Figure 8** The fitting error distribution along $s$ before and after optimization for the Parabola I example

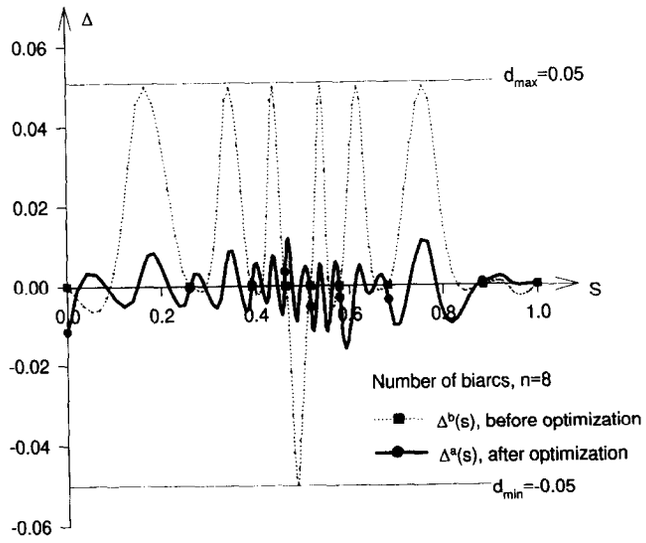


**Figure 9** The fitting error distribution along $s$ before and after optimization for the Parabola II example

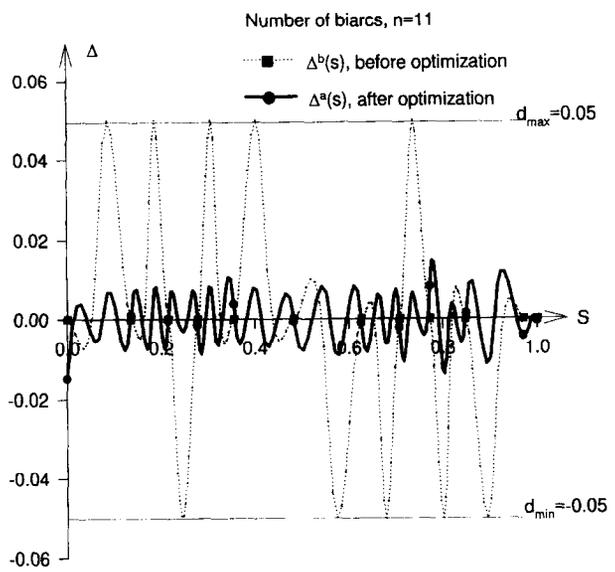


**Figure 10** The fitting error distribution along $s$ before and after optimization for the Sine I example

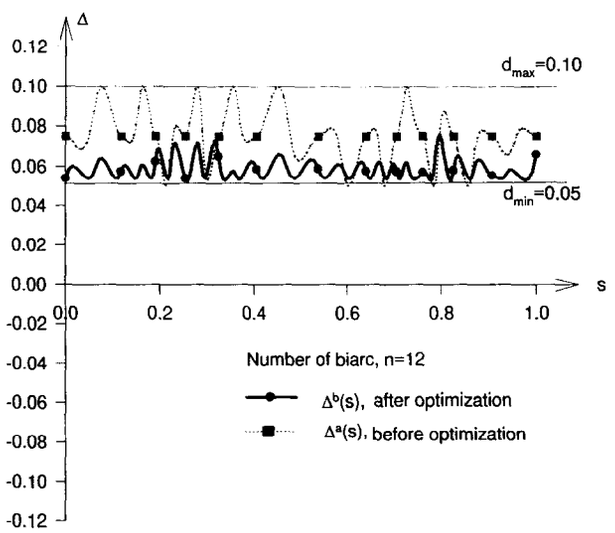


**Figure 11** The fitting error distribution along $s$ before and after optimization for the Sine II example

**Table 3** The optimal biarc-spline results

| Curve | $(\Delta^b_{min}, \Delta^b_{max})$ | $(\Delta^a_{min}, \Delta^a_{max})$ | $c_{area}$ | $c_l$ |
|---|---|---|---|---|
| Cam | (−0.05, 0.05) | (−0.01876, 0.01761) | 19.4% | 36.37% |
| Parabola I | (−0.05, 0.05) | (−0.01588, 0.01215) | 25.98% | 28.03% |
| Parabola II | (0.0, 0.05) | (0.00136, 0.01720) | 19.65% | 31.68% |
| Sine I | (−0.05, 0.05) | (−0.01499, 0.01539) | 27.80% | 30.38% |
| Sine II | (0.05, 0.10) | (0.05236, 0.07619) | 78.00% | 47.66% |

about 4–5 times. The Sine II case has a much higher value of $c_{area}$ because of the choice of $d_{min}$. Setting $d_{min} = 0.05$ imposes a lower bound on the achievable $A^a$ and this is reflected in the high value of $c_{area}$. The $c_l$ shows a range of 28.03–47.66%. It is interesting to note that none of the examples has a $c_l$ of 100%. It is easy to verify this from the figures as none of the $\mathbf{R}^a(s)$ touches the constraints. However, this indicates that it is possible to further reduce the number of segments.

The examples above are computed on a DEC 7000 machine with a LINPACK benchmark[14] of 44 Mflop/s. For $\epsilon = 10^{-5}$ with $d_{min} = -0.05$ and $d_{max} = 0.05$, the CPU times used for the cam, parabola and sine examples are 21.34, 4.24 and 19.72 s, respectively. In all examples, the initialization phase of getting $n$ took less than 1% of the overall CPU time.

## CONCLUSIONS

A formulation of the optimal curve-fitting of biarc-spline to a given B-spline curve is presented. The objective is to have minimum absolute area between the biarc-spline curve and the B-spline curve subject to some maximum and minimum deviation constraints. The formulation looks at the curve-fitting problem from a 'global' point of view. All the parameters which define the biarc-spline are used in the formulation to achieve a good fit. Tolerance control, such as overcut or undercut requirement, is easily incorporated in this formulation. Numerical examples on several problems show that significant reduction in the absolute area is achievable with a corresponding reduction in the maximum deviation.

## REFERENCES


1 Bolton, K M 'Biarc curves' *Comput.-Aided Des.* Vol 7 No 2 (1975) pp 89–92
2 Schonherr, J 'Smooth biarc curves' *Comput.-Aided Des.* Vol 25 No 6 (1993) pp 365–370
3 Parkinson, D B 'Optimized biarc curves with tension' *Comput. Aided Geom. Des.* Vol 9 (1992) pp 207–218
4 Su, B-Q and Liu, D-Y *Computational Geometry — Curve and Surface Modeling* Academic Press, USA (1989)
5 Piegl, L 'Curve fitting algorithm for rough cutting' *Comput.-Aided Des.* Vol 18 No 6 (1986) pp 79–82
6 Moreton, D N and Parkinson, D B 'The Application of a biarc technique in CNC machining' *Comput.-Aided Engng Des. J.* Vol 8 (1991) pp 54–60
7 Meek, D S and Walton, D J 'Approximation of discrete data by $G^1$ arc splines' *Comput-Aided Des.* Vol 24 No 6 (1992) pp 411–419
8 Makinouchi, S, Okamoto, M and Yamagata, K 'Optical curve fitting for NC machining by dynamic programming technology' *Report* Osaka University, Vol 16 No 2 (1976)
9 Cantoni, A 'Optimal curve fitting with piecewise linear functions' *IEEE Trans. Comput.* Vol C-20 No 1 (1971) pp 59–67
10 Sandel, G 'Zur Geometrie der Korbbogen' *Z. Ang. math. Mech.* Vol 15 (1937) pp 301–302
11 Krutz, M *Handbook of Applied Mathematics for Engineers and Scientists* McGraw-Hill (1991)
12 Hettich, R and Kortanek, K O 'Semi-infinite programming: theory, methods and applications' *SIAM Rev.* Vol 35 No 3 (1993) pp 380–429
13 Schittkowski, K 'NLPQL: A FORTRAN subroutine solving constrained nonlinear programming problem' *Ann. Oper. Res.* Vol 5 (1986) pp 485–500
14 Dongarra, J 'Performance of various computers using standard linear equations software' *Dept. Comp. Sci. Report No CS-89-85* University of Tenn. (Sept 1994)


## APPENDIX A

From Equation 4.3 on page 190 of Reference 4, it is easy to show that the locus of the common intersection point $\mathbf{Q}_i$ is an arc. The centre of this arc $\mathbf{C}(ulo_i, vlo_i)$ and the radius of the arc are given by

$$ulo_i = 0.5L_i, \quad vlo_i = -\frac{L_i}{2\tan(0.5(\alpha_i - \beta_i))}$$

and $$rlo_i = \frac{L_i}{2\sin(0.5(\alpha_i - \beta_i))}.$$

Figure A1 shows the relationship among the various quantities. From figure A1, it can be shown that

$$M_iQ_i = \sqrt{\left(\frac{L_i}{2\sin\left(\frac{\alpha_i - \beta_i}{2}\right)}\right)^2 - (0.5 - q_i)^2 L_i^2} \tag{A1}$$

$$M_iF_i = \frac{L_i}{2\tan\left(\frac{\alpha_i - \beta_i}{2}\right)} \tag{A2}$$

and

$$p_iF_i = q_iL_i \tag{A3}$$

The expressions for $\omega_i$ and $\Psi_i$ in Equation 1 follows from Equations A1–A3.

## APPENDIX B

The formula for the curve of cam in polar coordinate

$$r(\alpha) = \begin{cases} R & -\pi/4 \le \alpha \le \pi/4 \\ R + H\left(\dfrac{\alpha - \pi/4}{\beta} - \dfrac{1}{2\pi}\sin\left(2\pi\dfrac{\alpha - \pi/4}{\beta}\right)\right) & \pi/4 \le \alpha \le 3\pi/4 \\ R + H & 3\pi/4 \le \alpha \le 5\pi/4 \\ R + H\left(\left(1 - \dfrac{\alpha - 5\pi/4}{\beta}\right) + \dfrac{1}{2\pi}\sin\left(2\pi\dfrac{\alpha - 5\pi}{\beta}\right)\right) & 5\pi/4 \le \alpha \le 7\pi/4 \end{cases} \tag{B1}$$

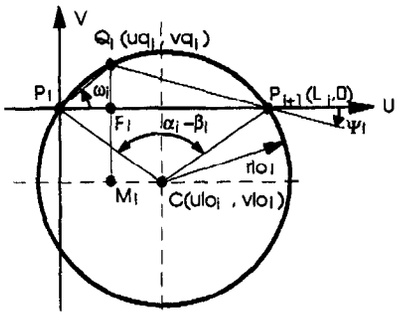


**Figure A1**

$\beta = \pi/2.0$

$R = 40$

$H = 20$

The formula for the curve of parabola

$$x = 2pt^2$$
$$y = 2pt \qquad \text{(B2)}$$
$$p = 10, -2 \leq t \leq 2$$

The formula for the curve of sine

$$x = 60\pi t$$
$$y = 20\sin(2\pi t) \qquad \text{(B3)}$$
$$0 \leq t \leq 1$$

*Chong-Jin Ong is a lecturer in the Department of Mechanical and Production Engineering at the National University of Singapore (NUS). After receiving his BEng (Hons) and MEng from NUS, he pursued postgraduate programmes at the University of Michigan, Ann Arbor, where he obtained his MS(Eng) and PhD for his work on robotic path planning. His research interests are in distance computation of geometrical objects, path planning algorithm, control and optimization theory and their applications.*

*Yoke-San Wong is a senior lecturer in the Department of Mechanical and Production Engineering at NUS. He received his BEng (Hons) and MEng from NUS and his PhD from the University of Manchester Institute of Science and Technology. His research interests are in special applications of CNC systems, CAD/CAM integration, modelling, analysis and control of automated manufacturing systems.*

*Han-Tong Loh is a senior lecturer in the Department of Mechanical and Production Engineering at NUS. He received his BEng (Hons) from the University of Adelaide. Subsequently, he joined NUS as a Senior Tutor and obtained his MEng for his work on a user-friendly robotics language. He then enrolled at the University of Michigan, Ann Arbor, where he obtained his MS(Eng) and PhD for his work on design optimization. His research interests are in optimization and CAD/CAM.*

*Xin-Gui Hong is a postgraduate research student in the Department of Mechanical and Production Engineering at NUS. He received his BEng and MEng from Nanjing Aeronautical Institute, People's Republic of China. Prior to joining NUS as a research scholar, he was a lecturer at Nanchang Institute of Aeronautical Technology. His research interests are in CAD/CAM, geometric modelling and optimization.*