ELSEVIER

Computer-Aided Design 33 (2001) 35–43

COMPUTER-AIDED DESIGN

www.elsevier.com/locate/cad

Planar point set fairing and fitting by arc splines

Xunnian Yang*, Guozhao Wang

Department of Applied Mathematics, Zhejiang University, Yuquan, Hangzhou 310027, People's Republic of China

Received 11 May 1999; received in revised form 19 February 2000; accepted 1 March 2000

Abstract

We fair and fit planar point sets by minimal-energy arc splines. The fairing process consists of two steps: computing the optimal tangents for curve interpolation and adjusting the point positions by smoothing the discrete curvatures. To fit the point set with minimal-energy arc curve, a simple linear algorithm is given for computing the optimal tangents. The discrete curvatures derived from the optimal tangents can be made smooth by low-pass filtering. These two linear and local algorithms are combined to generate a fair point set together with a fair G^1 arc curve within a given tolerance of the original data. The method can be used for fair shape design and measured data processing. Numerical examples are given to show the efficiency of this method. © 2000 Elsevier Science Ltd. All rights reserved.

Keywords: Point set fairing; Minimal-energy curve; Arc splines; Data fitting

1. Introduction

A curve is said to be fair if the curve has few inflexions and the curvature plot consists of relatively few monotone pieces. A set of points is fair if there is a fair curve passing through all these points (see the definitions in Refs. [1,2]). Though the definition of fairness is application dependent, a curve with minimal energy, i.e. a curve along which the integral of the square of the curvature is minimal, is considered to be a fair one [3–5]. Curve fairing and point set fairing play important roles in designs and some manufacturing processes [6–10]. This paper addresses the problem of planar point set fairing and fitting by minimal-energy arc splines. The method serves two purposes: generating fair point set for further applications [6,7] and generating fair smooth-arc curve for direct applications such as fair shape design, CNC machining and robot path planning [4,11,12].

The problem of curve fairing and point set fairing have attracted much attention in the fields of computer-aided design during the past few decades, and most fairing algorithms are designed to fair parametric spline curves. We refer interested readers to Refs. [8–10] and references therein. In fact, designing efficient algorithms for point set fairing has great significance; Feldman [6] has introduced some backgrounds of this topic. Algorithms for point set fairing are based either on divided difference of the point set [13] or on smoothed discrete curvatures that are computed based on fitting circles [6,7]. Optimization techniques are often employed to solve the fairing problem by existing algorithms. The result of the fairing process is still a set of points, which may not imply a fair smooth passing curve. Our work differs from these algorithms in that we estimate the discrete curvatures based on minimal-energy arc curve model and obtain a smooth interpolating curve simultaneously.

Instead of estimating the discrete curvatures based on fitting circles, we fit the point set with a set of constrained arc segments. The energy function of the fitting arc segments is quadratic and the optimal tangents can be obtained by solving a system of linear equations. The discrete curvatures can then be computed based on a smooth interpolating biarc curve. We smooth the discrete curvatures by low-pass filtering principle, and adjust the point positions to obtain a fair curve simultaneously. Even more, constraints such as with fixed points, or with permitted error tolerance can be considered in the fairing process.

In the following text, we will discuss the three steps for point set fairing and fitting in detail: optimal tangents computing, discrete fairing algorithm and fair arc curve construction.

2. Optimal tangents computation

A curve of minimal energy is a curve along which the integral of the square of the curvature is minimal. The strain

* Corresponding author. Tel.: + 86-571-7951609; fax: + 86-571-7952353.

E-mail addresses: yxn@math.zju.edu.cn (X. Yang), wgz@math.zju.edu.cn (G. Wang).

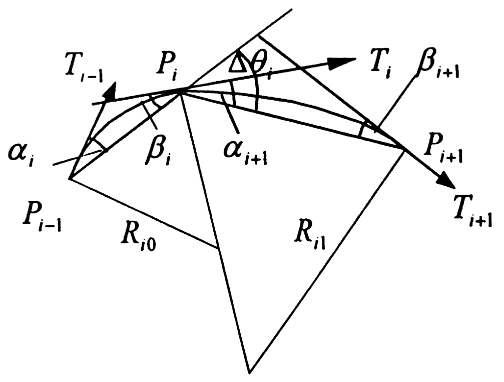


Fig. 1. Two smooth connected arc segments interpolate points P_{i-1}, P_i and P_{i+1} and the candidate tangent vector T_i at point P_i.

energy E of the curve is given by

$$E = \int k^2 \, ds, \tag{1}$$

where k is the curvature function of the arc length s.

Let $P_0, P_1, \ldots, P_n$ be an ordered set of planar points with given end tangents T_0 and T_n. By integrating the square of the curvature along the curve, the strain energy of an interpolating biarc spline becomes

$$E = \sum_{i=1}^{n} \left(\frac{\phi_{1i}}{r_{1i}} + \frac{\phi_{2i}}{r_{2i}} \right), \tag{2}$$

where ϕ_{1i} and ϕ_{2i} are the angles in radians and r_{1i} and r_{2i} are the radii for the ith biarc panel. To minimize E is equivalent to finding a set of suitable values for all the angles and radii of the biarc panels. Optimization procedures are needed to solve this non-linear problem. However, it requires a suitable library algorithm and is subject to long run times and frequent failures due to discontinuity of the energy function and difficulties associated with defining suitable constraints on the gradients (see Ref. [4]).

Instead of constructing a biarc spline with minimal energy directly, we will first estimate the optimal tangents for a set of points using energy minimization technique. Let T_i be the unit tangent vector at the point P_i, and let the angle from tangent T_{i-1} to the chord $P_{i-1}P_i$ and the angle from the same chord to the tangent T_i be α_i and β_i, respectively; $i = 1, \ldots, n$ (see Fig. 1). The angle is positive if it is counter-clockwise and negative otherwise. The end angles α_1 and β_n are known since the end tangents T_0 and T_n are given. Let $\Delta\theta_i$ be the turning angle from the chord $P_{i-1}P_i$ to the chord P_iP_{i+1}, then for $i = 1, \ldots, n-1$, we have

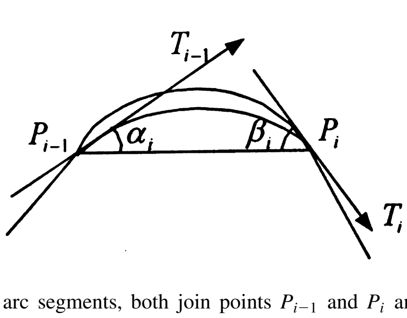


Fig. 2. Two arc segments, both join points P_{i-1} and P_i and match two tangents at the points, respectively.

$$\beta_i + \alpha_{i+1} = \Delta\theta_i. \tag{3}$$

To compute the intermediate tangent T_i, we first fit a biarc panel interpolating the points P_{i-1}, P_i and P_{i+1} and match the tangent T_i at point P_i (see Fig. 1). Let the two arc segments of the biarc be C_{i0} and C_{i1} with radii R_{i0} and R_{i1}, respectively. Then the radii of these two arc segments are

$$R_{i0} = \frac{\|P_i - P_{i-1}\|}{2 \sin \beta_i}, \tag{4}$$

$$R_{i1} = \frac{\|P_{i+1} - P_i\|}{2 \sin \alpha_{i+1}}, \tag{5}$$

where $\| \, \|$ denotes the Euclidean norm of a vector. The angles, in radians, for the arcs C_{i0} and C_{i1} are $2\beta_i$ and $2\alpha_{i+1}$, respectively. From Eq. (2), the strain energy of these two adjacent arc segments is

$$E_i = 2\left(\frac{\beta_i}{R_{i0}} + \frac{\alpha_{i+1}}{R_{i1}} \right). \tag{6}$$

By substituting Eqs. (4) and (5) into Eq. (6), we get

$$E_i = 4\left(\frac{\beta_i \sin \beta_i}{\|P_i - P_{i-1}\|} + \frac{\alpha_{i+1} \sin \alpha_{i+1}}{\|P_{i+1} - P_i\|} \right) \tag{7}$$

Hence the total energy of all the fitting arc segments is $E_{\mathrm{arc}} = \sum_{i=1}^{n-1} E_i$.

For $i = 2, \ldots, n-1$, there is one biarc panel interpolating points P_{i-2}, P_{i-1} and P_i, and another panel interpolating points P_{i-1}, P_i and P_{i+1}. Then there are two arc segments joining points P_{i-1} and P_i simultaneously and matching tangents T_{i-1} and T_i, respectively (see Fig. 2). For the first panel, there is only one arc segment interpolating P_0 and P_1 that matches T_1. An additional arc segment is obtained by interpolating P_0 and P_1 that matches the given tangent T_0. In the same way, there are two arc segments interpolating P_{n-1} and P_n, one of which is obtained by matching the given tangent T_n. The difference of the curvatures of the two segments joining P_{i-1} and P_i is $2|\sin \alpha_i - \sin \beta_i| / \|P_i - P_{i-1}\|$. This difference vanishes if α_i and β_i are equal. In this case, the two arc segments coincide.

The total energy of the fitting arcs E_{arc} will be roughly two times the energy of a smooth-fitting arc curve when the curvature differences for every two arc segments joining consecutive points are small enough. Then the tangents derived in this way will resemble the optimal tangents by a smooth curve with minimal energy. To eliminate the absolute operator in the curvature difference and establish a quadratic objective function, we compute the integral of the square of the curvature difference along the fitting arcs. By integrating along the arc segments piecewise, we have the total integral as

$$\sum_{i=1}^{n} \int_{P_{i-1}}^{P_i} \frac{4(\sin \alpha_i - \sin \beta_i)^2}{\|P_i - P_{i-1}\|^2} \, ds. \tag{8}$$

Since the curvature difference of every two arc segments joining points P_{i-1} and P_i is constant, and approximating the length of an arc segment joining these two points as the chord length $\|P_i - P_{i-1}\|$, integral (8) can be approximated by

$$E_{\mathrm{crv}} = 4 \sum_{i=1}^{n} \frac{(\sin \alpha_i - \sin \beta_i)^2}{\|P_i - P_{i-1}\|}. \tag{9}$$

The tangents for fair curve fitting can then be obtained when E_{arc} and E_{crv} are both low values. Consequently, we use a linear combination of these two functions as the objective function for optimal tangent computation. The optimal tangents can be found by minimizing the following formula:

$$E_{\mathrm{arc}} + \lambda E_{\mathrm{crv}} = \min, \tag{10}$$

where λ is a scalar factor. Although we cannot assert which λ can yield the optimal result, the experiments show that tangents with different values of λ in the range of 1–2 cannot be distinguished explicitly. Then the suggested choice is $\lambda = 1.5$, which can serve for the optimal tangents and construction of low-energy and fair arc spline curves.

To simplify and speed up the computation, we use only the linear term in the Taylor series expansion of the sine function. If the turning angles of the chords joining the points are relatively small, i.e. when the angles $\Delta\theta_i$ are small, then the higher terms in the series are negligible. This is the case in many practical applications. Then objective function (10) can be approximated by

$$E = 4 \sum_{i=1}^{n-1} \left(\frac{\beta_i^2}{\|P_i - P_{i-1}\|} + \frac{\alpha_{i+1}^2}{\|P_{i+1} - P_i\|} \right) + 4\lambda \sum_{i=1}^{n} \frac{(\alpha_i - \beta_i)^2}{\|P_i - P_{i-1}\|}. \tag{11}$$

Let $l_i = \|P_i - P_{i-1}\|$, $i = 1, \ldots, n$. Since the multiple 4 does not play any role in the minimization, Eq. (11) can be simplified to

$$U = \sum_{i=1}^{n-1} \left(\frac{\beta_i^2}{l_i} + \frac{\alpha_{i+1}^2}{l_{i+1}} \right) + \lambda \sum_{i=1}^{n} \frac{(\alpha_i - \beta_i)^2}{l_i}. \tag{12}$$

Then the optimal tangents can be obtained by minimizing the quadratic objective function U.

From Eq. (3) we have $\beta_i = \Delta\theta_i - \alpha_{i+1}$. By substituting $\beta_i = \Delta\theta_i - \alpha_{i+1}$ into Eq. (12), we get a quadratic function U with $(n-1)$ unknowns $\alpha_2, \ldots, \alpha_n$. Minimizing the objective function U is equivalent to the solution of the system of equations $\partial U / \partial \alpha_i = 0$. For $i = 2, \ldots, n-1$, these yield the linear equations

$$\frac{\lambda}{l_{i-1}} \alpha_{i-1} + \left(\frac{\lambda+1}{l_{i-1}} + \frac{\lambda+1}{l_i} \right) \alpha_i + \frac{\lambda}{l_i} \alpha_{i+1} = \frac{\lambda+1}{l_{i-1}} \Delta\theta_{i-1} + \frac{\lambda}{l_i} \Delta\theta_i. \tag{13}$$

For $i = n$ we have the linear equation

$$\frac{\lambda}{l_{n-1}} \alpha_{n-1} + \left(\frac{\lambda+1}{l_{n-1}} + \frac{\lambda+1}{l_n} \right) \alpha_n = \frac{\lambda+1}{l_{n-1}} \Delta\theta_{n-1} + \frac{\lambda}{l_n} \beta_n. \tag{14}$$

These combine into a system of $(n-1)$ linear equations in $(n-1)$ unknowns. This system of linear equations can be written in the matrix form

$$MZ = H, \tag{15}$$

where Z is a vector of $(n-1)$ unknown variables and H is a constant vector associated with the lengths and the turning angles of the chords. The coefficient matrix M is a tridiagonal matrix. The α_i can be solved by LU factorization. Once the α_i is known, the β_i can be computed immediately.

3. Automatic fairing algorithm

The process of point set fairing or curve fairing often consists of two phases: inflexion reduction and convexity preserving fairing or fine fairing [6,8]. To reduce redundant inflexions, the "tight string method" proposed by Feldman [6] is an efficient method for discrete point set. In this section, we will focus on the fine fairing of point set using discrete curvatures that are derived from minimal-energy arc curves. The following algorithm is based on the assumption that the redundant inflexions of the point set have been eliminated.

3.1. Discrete curvature computation

With the optimal tangents obtained by Eq. (15), we can now estimate the discrete curvatures at every point by a spline of the biarc curve interpolating the point set and the tangents at the points. To decide a biarc curve interpolating points P_{i-1} and P_i and the tangents T_{i-1} and T_i, we cite here the formula proposed in Ref. [2]. If the angles α_i and β_i have the same signs, a C-shaped biarc panel can be constructed, and the radii for two arc segments are

$$r_1 = \frac{l_i}{2 \sin((\alpha_i + \beta_i)/2)} \frac{\sin(\beta_i/2)}{\sin(\alpha_i/2)}, \tag{16}$$

$$r_2 = \frac{l_i}{2 \sin((\alpha_i + \beta_i)/2)} \frac{\sin(\alpha_i/2)}{\sin(\beta_i/2)}. \tag{17}$$

By approximating the sine value of an angle with the angle itself, the curvatures for the left and the right arc segment abutting point P_i are $k_i^- = (\alpha_i + \beta_i)\beta_i / l_i \alpha_i$ and $k_i^+ = (\alpha_{i+1} + \beta_{i+1})\alpha_{i+1} / l_{i+1} \beta_{i+1}$. If pairs of angles α_i and β_i or

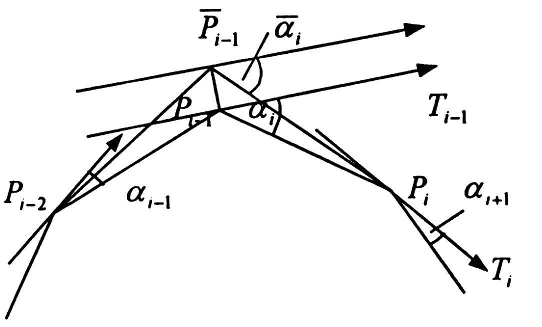


Fig. 3. Adjusting point position based on smoothed curvature.

α_{i+1} and β_{i+1} have different signs, there are inflexions in a smooth-fitting curve at the left- or at the right-side of point P_i. The left-curvature at point P_i can be computed using the curvature of the circle owning tangent T_i at point P_i and interpolating P_{i-1}, while the right-curvature can be computed by another circle interpolating point P_{i+1}. From Eqs. (4) and (5) we have $k_i^- = 2\beta_i/l_i$ and $k_i^+ = 2\alpha_{i+1}/l_{i+1}$. The discrete curvature at point P_0 or P_n can be chosen as k_0^+ or k_n^-, and at the other point can be chosen as $k_i = (k_i^- + k_i^+)/2$.

3.2. Low-pass filtering

Smooth discrete curvatures can help in yielding a fair fitting curve. The discrete curvatures can be made smoother by the low-pass filtering principle.

Supposing that the sequence of discrete curvatures is $k_1, k_2, \ldots, k_n$, by applying the Discrete Fourier Transformation (DFT), the frequency of the curvature sequence is

$$f_r = \sum_{i=1}^{n} k_i\, \mathrm{e}^{-j2\pi ir/n}, \qquad r = 1, 2, \ldots, n. \tag{18}$$

where $j = \sqrt{-1}$. The discrete curvatures can be recovered by the inverse DFT

$$k_i = \frac{1}{n}\sum_{r=1}^{n} f_r\, \mathrm{e}^{j2\pi ir/n}, \qquad i = 1, 2, \ldots, n. \tag{19}$$

Now the discrete curvature sequence k can be made smoother by discarding some high-frequency terms of the sum. When implemented, the method of DFT is computationally expensive; even with fast Fourier transformation the computational complexity is in the order of $n\log(n)$ operations.

An alternative efficient smoothing algorithm is to do the projection onto the space of low frequencies only approximately. According to Lindeberg [14], a set of discrete signals can be smoothed based on diffusion equation. Let $\Delta k_i = ((k_{i-1} - k_i)/2) + ((k_{i+1} - k_i)/2)$; in the simplest form, the curvature can be smoothed by the following iteration formula

$$k_i' = k_i + \mu\Delta k_i, \tag{20}$$

where $0 < \mu \le 1/2$ is a scale factor.

When the discrete curvatures have been smoothed, we can keep the tangents unchanged and disturb the point positions to obtain a set of fair points with these new curvatures. We may fair the point set by choosing the discrete curvature at point P_i as $k_i = k_i^-$, $k_i = k_i^+$ or $k_i = (k_i^- + k_i^+)/2$. The last choice may give better results, but the computation is somewhat complex and costly. For the convenience and efficiency of computation, we choose here $k_i = k_i^+$. The tests we have experimented show that this choice will give satisfying results. Since α_i are far less than the chord length l_i, the direction of tangent T_{i-1} at point P_{i-1} can be kept unchanged. The smoothed discrete curvatures will yield modified tangent chord angles $\{\bar{\alpha}_i\}$ and the point P_{i-1} can be moved to a new position with tangent chord angle $\bar{\alpha}_i$ $(i = 2, \ldots, n)$.

If every two elements in angle pairs $(\alpha_{i-1}, \beta_{i-1})$, (α_i, β_i) and $(\alpha_{i+1}, \beta_{i+1})$ have the same sign, the discrete curvatures can be estimated by interpolating biarc curves, i.e. $k_i = (\alpha_{i+1} + \beta_{i+1})\alpha_{i+1}/l_{i+1}\beta_{i+1}$; otherwise, we compute the curvatures by fitting circles and $k_i = 2\alpha_{i+1}/l_{i+1}$. If the first case holds, let $\omega_{i0} = ((\alpha_{i-1} + \beta_{i-1})\beta_i/(\alpha_i + \beta_i)\beta_{i-1})(l_i/l_{i-1})$ and $\omega_{i1} = ((\alpha_{i+1} + \beta_{i+1})\beta_i/(\alpha_i + \beta_i)\beta_{i+1})(l_i/l_{i+1})$, else let $\omega_{i0} = l_i/l_{i-1}$ and $\omega_{i1} = l_i/l_{i+1}$. The new angle $\bar{\alpha}_i$ corresponding to angle α_i can be derived from Eq. (20):

$$\bar{\alpha}_i = \alpha_i + (\omega_{i0} * \alpha_{i-1} + \omega_{i1} * \alpha_{i+1} - 2\alpha_i) * \frac{\mu}{2}. \tag{21}$$

As for the scalar coefficient μ, the tests we have experimented show that $\mu = 0.2$ is the suggested choice.

When the tangent chord angle α_i has been modified, the point P_{i-1} should be adjusted in the permitted range simultaneously. If V is the unit vector rotated by $+\pi/2$ from the tangent T_{i-1} and the tangent T_{i-1} is supposed to be unchanged, we can move point P_{i-1} in the direction V for $\mathrm{d}H_0$ to get the desired new tangent chord angle $\bar{\alpha}_i$ (see Fig. 3), where $\mathrm{d}H_0$ can be formulated as

$$\mathrm{d}\,H_0 = -l_i(\cos\alpha_i * tan(\bar{\alpha}_i) - \sin\alpha_i). \tag{22}$$

To guarantee that the mth faired point $P_{i-1}^{(m)}$ is adjusted within tolerance τ of initial point $P_{i-1}^{(0)}$, the upper and lower bounds for $\mathrm{d}H$ should be computed. Let $A = P_{i-1}^{(m-1)} - P_{i-1}^{(0)}$, then $\mathrm{d}H$ should satisfy the inequality

$$(A + \mathrm{d}H\cdot V)^2 \le \tau^2. \tag{23}$$

From Eq. (23), we can conclude that $\mathrm{d}H_1 \le \mathrm{d}H \le \mathrm{d}H_2$, where $\mathrm{d}H_{1,2} = -A\cdot VF\sqrt{(A\cdot V)^2 - A^2 + \tau^2}$. Now, if $\mathrm{d}H_0 < \mathrm{d}H_1$ or $\mathrm{d}H_0 > \mathrm{d}H_2$, then $\mathrm{d}H$ can be chosen as $\mathrm{d}H_1$ or $\mathrm{d}H_2$, otherwise $\mathrm{d}H = \mathrm{d}H_0$. The new position for point P_{i-1} is

$$P_{i-1}^{(m)} = P_{i-1}^{(m-1)} + \mathrm{d}H\cdot V \tag{24}$$

We can fair the point set by adjusting the point positions one by one. When position for point P_{i-1} has been changed,

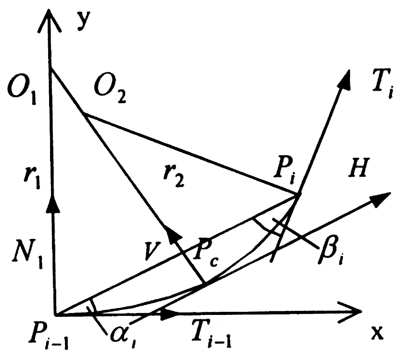


Fig. 4. Biarc curve construction.

the angle α_i should be replaced by $\bar{\alpha}_i$. If point P_k is a fixed one, no movement should be done.

3.3. The algorithm

There are two strategies for fairing a planar point set by the above two algorithms. The first one is to compute the tangents only once and smooth the discrete curvatures using the low-pass filtering algorithm repeatedly, and the other method is to compute the optimal tangents and filter the discrete curvatures alternately. It seems that the second method has more computational costs than the first one, but the convergence of the first method is much slower than the second one. In fact, computing the new optimal tangents, when the points position have been changed, will give a more concise estimate of the discrete curvatures for every iteration. The curvature plot of the faired point set by the second method always consists of fewest monotone segments while the first method does not. Then, the second strategy is the preferred one.

When fairing a set of points, the objective function U will decrease steadily just after a few iterations. The function U will reach a smallest value and waver around it after tens of iterations. For uniform data we can obtain satisfying results with about 20 iterations, and for other data only 60–80 iterations will give a set of faired points together with a fair arc spline curve. To fair a set of points, we can choose the iteration number manually according to the quality of the curvature plot or stop the fairing procedure automatically. To decide when to stop, we can check if the objective function U is still decreasing from the 10th iteration or stop the iteration procedure when the iteration number has exceeded a predefined number such as 80. Because there are only linear operations in the fairing procedures, we can always fair a point set in real-time. The fairing algorithm is as follows:

```
Geometric fairing algorithm
input: a set of planar points and the error tolerance $\tau$;
output: the faired point set and the optimal tangents at the points.
Step 0. Remove redundant inflexions;
Step 1. Compute the optimal tangents;
Step 2. Compute the discrete curvature;
Step 3. Curvature smoothing and points adjusting;
Step 4. If the termination condition has been achieved stop else go to Step 1.
```

4. Fair arc spline curve fitting

With the faired point set associated with the optimal tangents at these points, we can now fit the point set with a smooth-arc spline curve by choosing one of the biarc fitting methods. There are many ways to construct a biarc curve interpolating two points and two tangents at the points [2,15]. This section will introduce a brief idea for fast biarc curve construction for fair curve fitting purpose.

An arc spline interpolating the point set can be obtained by constructing biarc curves interpolating every two points P_{i-1} and P_i and tangents T_{i-1} and T_i, $(i = 1,\ldots,n)$. The angle from the tangent T_{i-1} to the chord $P_{i-1}P_i$ and the angle from the same chord to the tangent T_i are α_i and β_i, respectively. Since we will have to decide three variables to decide a circle (two components of the center and the radius of the circle), there are six freedoms to decide two circles. But, there are only five constraints for the construction of two connected and mutually tangent arc segments matching the above data (see Fig. 4). One additional free parameter has to be set aside for the construction of a biarc panel. A plethora of methods have been given to choose this free variable [2,15].

Let the angle θ from the direction T_{i-1} to the tangent line U at the connecting point be the free variable. Once the angle θ is decided, a biarc curve interpolating the points and the tangents can be constructed. We here employ the fast algorithm proposed by Jin [16] for biarc curve construction. The biarc curve is supposed to be constructed in a local right-handed coordinate system with origin at P_{i-1} and x coordinate axis paralleling T_{i-1} (see Fig. 4). The conversion of points or vectors in one coordinate system to another can be achieved by direct transformation. Let O_1 and O_2 be the centers of two arc segments with radii r_1 and r_2, respectively, and let P_c be the contact point of these two arcs. In this local coordinate system the unit normal vector N_1 at point P_{i-1} is $N_1 = (0\ 1)$, and the unit tangent vector U at point P_c is $U = (\cos\theta \sin\theta)$, then the unit normal vector V at point P_c is $V = (-\sin\theta \cos\theta)$. Let $L = P_i - P_{i-1}$, $G = U - T_2$ and $W = N_1 - V$, then the radius and center for the first arc are, respectively,

$$r_1 = L{\cdot}G/W{\cdot}G,$$

$$O_1 = P_{i-1} + r_1 N_1.$$

The contact point is $P_c = P_{i-1} + r_1 W$. Let $Q = P_i - P_c$,

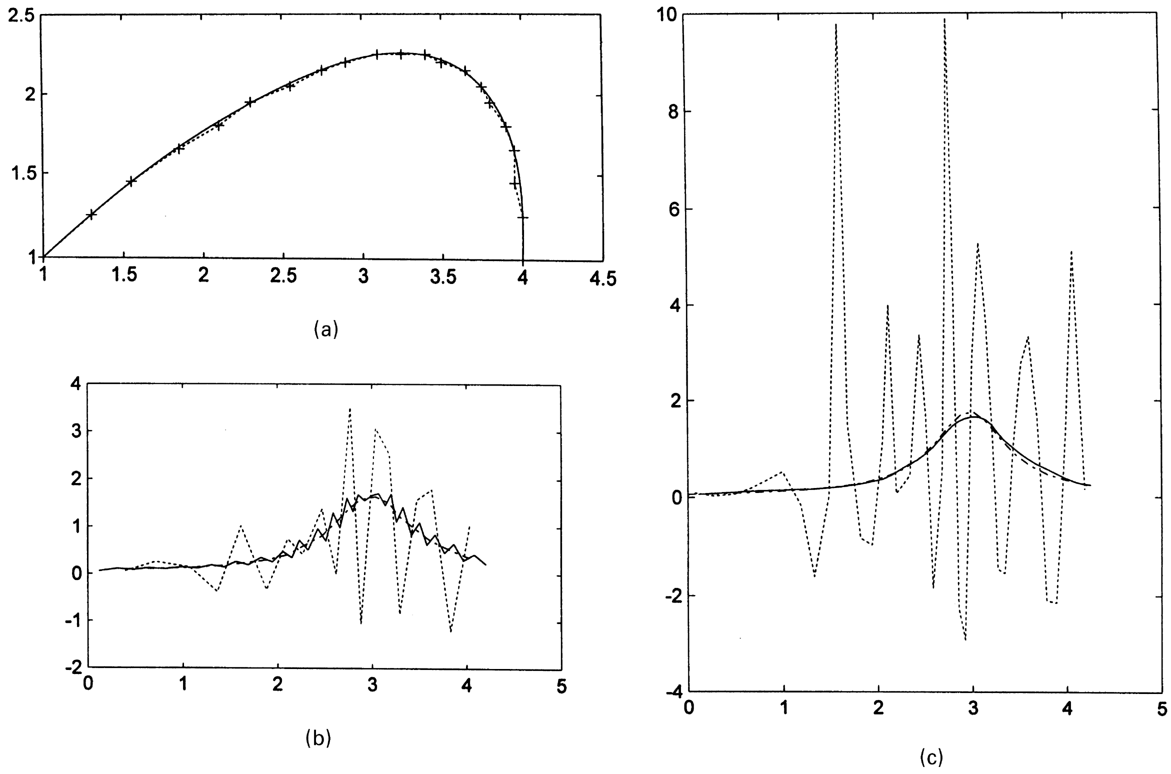


Fig. 5. Fairing the disturbed points sampled from a Bézier curve: (a) the fitting arc spline within tolerance 0.04; (b) circle curvature plots before (uniform dash) and after (non-uniform dash) fairing and curvature plot for arc spline (solid) by the circle-based method; (c) circle curvature plot after fairing (non-uniform dash) and curvature plots of fitting arc splines before (uniform dash) and after (solid) fairing by the arc-based method.

then the second arc can be determined as:

$$r_2 = Q^2/2Q\cdot V,$$

$$O_2 = P_c + r_2V.$$

If angles α_i and β_i have the same signs, we can construct a C-shaped biarc curve, otherwise construct an S-shaped biarc curve by choosing appropriate values for the angle θ. As demonstrated by Su and Liu [2], we choose $\theta = \alpha_i$ for C-shaped biarc curve construction. A key reason for this choice is that the choice can keep the monotony of the curvature plot of a cubic curve. This property will guarantee the global fairness of the fitting curve. Let the curvatures of a cubic curve at two points P_{i-1} and P_i be k_1 and k_2, and the radii of two arcs of a biarc curve that interpolates points P_{i-1} and P_i and the tangents at these two points be r_1 and r_2, respectively. If $k_1 < k_2$, then $k_1 < 1/r_1 < 1/r_2 < k_2$. If $k_1 > k_2$ then the inequality should be turned over. For the S-shaped biarc curve the angle θ can be chosen as $\theta = (3\alpha_i - \beta_i)/2$ (see Ref. [16]).

5. Examples

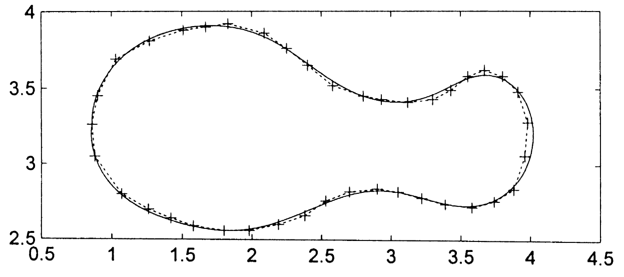
We have done a lot of experiments to test the new fairing algorithm and the fair arc curve fitting method. The program

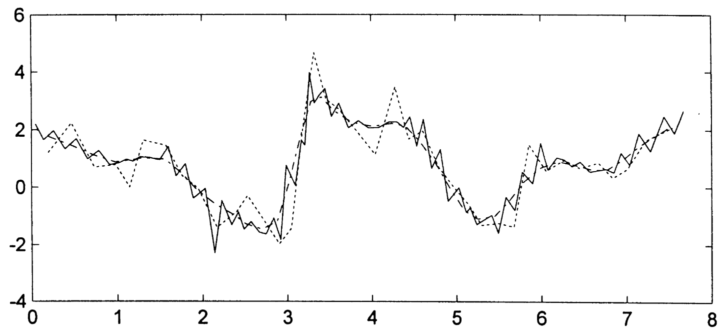
Table 1
The unit tangent vectors at the sampled points estimated by our method and the circle-based method. ρ_i is the angle between unit tangent vectors of the estimated and the Bézier curves at $\mathbf{b}(t_i)$

Parameter (t_i)	Bézier tangents	Arc-based tangents	ρ_I (rad) ($\times 10^{-2}$)	Circle-based tangents	ρ_i (rad) ($\times 10^{-2}$)
$t_0 = 0$	0.7682, 0.6402	0.7682, 0.6402	0	0.7682, 0.6402	0
$t_1 = 0.125$	0.8137, 0.5812	0.8126, 0.5828	0.189	0.8192, 0.5735	0.9415
$t_2 = 0.25$	0.8742, 0.4856	0.8731, 0.4876	0.227	0.8825, 0.4702	1.7529
$t_3 = 0.375$	0.9487, 0.3162	0.9465, 0.3227	0.680	0.9588, 0.2840	3.3763
$t_4 = 0.5$	1, 0	0.9999, 0.0048	0.481	0.9987, −0.0506	5.0632
$t_5 = 0.625$	0.8742, −0.4856	0.8737, −0.4864	0.086	0.8717, −0.4899	0.4944
$t_6 = 0.75$	0.5145, −0.8575	0.5124, −0.8587	0.239	0.5568, −0.8306	5.0115
$t_7 = 0.875$	0.1961, −0.9806	0.1877, −0.9822	0.861	0.2315, −0.9728	3.6270
$t_8 = 1$	0, −1	0, −1	0	0, −1	0

Table 2
The discrete curvatures at the sampled points estimated by our method and the circle-based method. δ_i is the curvature difference between the discrete curvatures of the estimated and the Bézier curves at $\mathbf{b}(t_i)$

Parameter (t_i)	Bézier curvature	Arc-based curvature	δ_i ($\times 10^{-2}$)	Circle-based curvature	δ_i ($\times 10^{-2}$)
$t_0 = 0$	0.0629	0.0706	0.762	0.0756	1.269
$t_1 = 0.125$	0.1117	0.1139	0.223	0.1127	0.098
$t_2 = 0.25$	0.2199	0.2254	0.547	0.2221	0.219
$t_3 = 0.375$	0.4857	0.4969	1.123	0.4879	0.226
$t_4 = 0.5$	1.1111	1.1238	1.266	1.0797	−3.142
$t_5 = 0.625$	1.7593	1.7616	0.233	1.6289	−13.04
$t_6 = 0.75$	1.2106	1.2223	1.169	1.1684	−4.22
$t_7 = 0.875$	0.5364	0.5503	1.393	0.5379	0.15
$t_8 = 1$	0.24	0.2847	4.475	0.3116	7.156


(a)


(b)

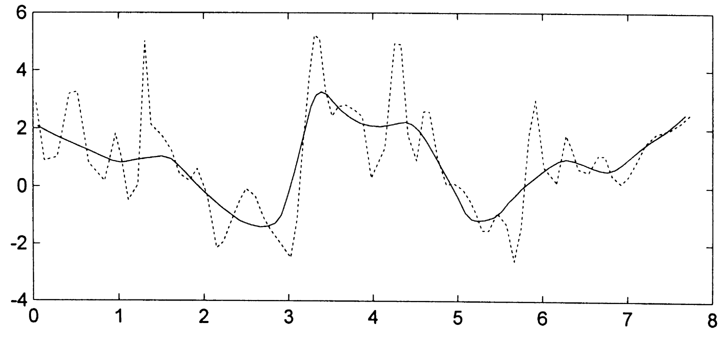

(c)

Fig. 6. Shoe-like shape modeling: (a) the fitting arc spline within tolerance 0.04; (b) circle curvature plots before (uniform dash) and after fairing (non-uniform dash) and curvature plot for arc spline (solid) by the circle-based method; and (c) arc spline curvature plots before (dash) and after fairing (solid) by the arc-based method.

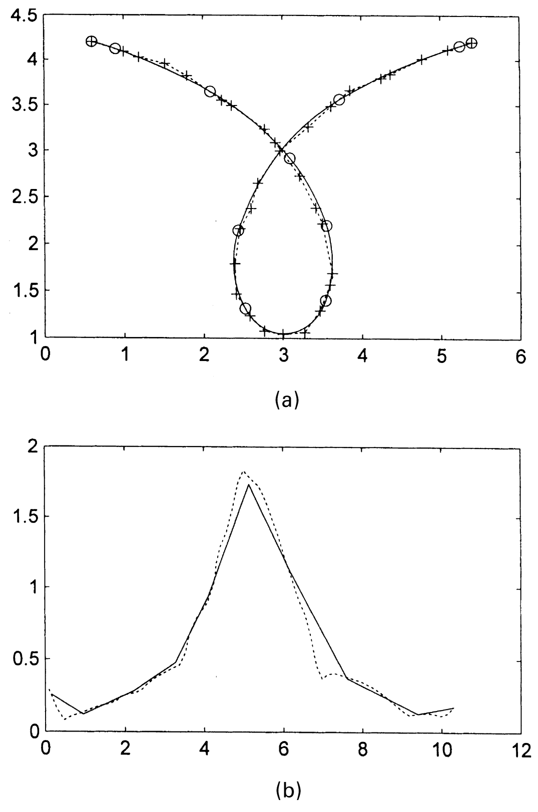


Fig. 7. Strophoid curve reconstruction: (a) the smooth-fitting arc spline consisting of 10 segments within tolerance 0.06; and (b) curvature plot of fair fitting arc spline before (dash) and after (solid) data reduction by the arc-based fairing method.

is implemented on an SGI Indigo2 workstation with MIPS R4000 and 96 Mb memory. The noisy data are generated by several different ways. A few examples dealing with convex polyline or polylines with inflexions are cited here. Discrete curvature plots before and after fairing have been drawn to show the results of fairing. To illustrate the curvature plots more clearly, we have connected the discrete curvatures at the points or the curvatures at the midpoints of arc segments into continuous polylines. The horizontal coordinate of the curvature plot is the accumulated length of the polyline or the arc length of the curve while the vertical value is the curvature of every arc segment.

To illustrate the efficiency of the arc-based fairing method, we have compared our method with the circle-based fairing method. The way we fair a set of points by the circle-based method is similar to that of the arc-based method. We modify the point positions based on smoothed discrete curvatures that have been estimated by fitting circles, and construct the fitting arc spline curve using the tangents of the fitting circles at the points. The curvatures of the fitting circles for the faired points by these two methods cannot be distinguished explicitly; but, the curvature plots show that the fitting arc spline for a point set faired by the biarc-based method is superior to the arc spline by the circle-based method.

The first example employs a Bézier curve (Fig. 5) with control points (1,1), (4,3.5) and (4,1); nine points are sampled uniformly with parameter step $\Delta t = 0.125$. Data in Tables 1 and 2 show that the approximate tangents and curvatures by the arc-based method are more concise and accurate than by the circle-based method.

The second example uses 21 points sampled from the same Bézier curve as in example 1. We first sample the points uniformly with parameter step $\Delta t = 0.05$ and then disturb the coordinates of point P (P_x, P_y) as $((\mathrm{int})(P_x * 20 + 0.5)/20, (\mathrm{int})(P_y * 20 + 0.5)/20)$, where function $\mathrm{int}(x)$ is equal to the integer part of number x. The point sets are faired within tolerance 0.04. A set of faired points together with a fair arc spline curve is obtained after 18 iterations within 0.03 s.

For the third example, we would like to design a shoe-like shape (Fig. 6). We first design the raw shape by sampling a set of points on the screen using the mouse. To form a closed shape, the last and the first points are defined coincident. It is clear that there are several inflexions in the polyline; we disturb all point positions within tolerance 0.04 using formula (21). Though the initial points are so noisy, we obtain a set of faired points together with a fair arc spline after 18 iterations within 0.06 s.

In the last example we test our program by a set of non-uniform sampled points. We have sampled 33 points on the strophoid curve (Fig. 7)

$$X(t) = \frac{t^2 - 1}{t^2 + 1}\begin{pmatrix} t \\ 1 \end{pmatrix}, \qquad t \in [-2, 2].$$

The sampling parameter interval is $\Delta t = 0.125$, but the parameters of the inner 29 points are disturbed randomly with magnitude 0.05. Even more, the positions of each of the inner 29 sampled point have been disturbed randomly inside the circle centered at the sampled point and with radius 0.05. After 80 iterations, we obtain a fair fitting arc spline in 0.17 s. There are 64 arc segments in the final biarc spline, and because the curvature plot consists of just a few monotone pieces, we can simplify the fair arc spline further. By employing our latest result for arc spline optimization in Ref. [17], we can simplify the fair arc spline into 10 segments within tolerance 0.01. A smooth fair arc spline fitting the original noisy data within tolerance 0.06 is achieved.

6. Conclusions and future work

In this paper, we present a very simple and efficient method for planar point set fairing and fitting by arc splines. The main contribution of the paper lies in three aspects: (a) efficient algorithm for planar point set fairing; (b) simple formula for optimal tangents computation for biarc curve interpolation; and (c) fitting noisy data by fair arc curve within the prescribed tolerance. The advantages of the algorithm are that the discrete curvatures estimated by minimal-energy curve model are more accurate and reliable than the circle-based method; a set of faired points and a fair fitting arc curve within a given tolerance are obtained in real-time. The result of the algorithm can be used directly for fair shape design, CNC machining, robot path planning, etc.

There are still some problems that deserve further study in the future. The first problem is under what conditions can the algorithm achieve the fewest monotone segments for the curvature plot. The second problem is how to construct arc splines with curvature plots having the prescribed properties. Finally, the method presented in this paper should be extended to fair a sequence of spatial points.

Acknowledgements

This work is supported by National Natural Science Foundation (No: 19971079), Foundation of State Key Basic Research 973 Development Programming Item (No: G1998030600) of China, and Natural Science Foundation of Zhejiang Province. Finally, the authors own thanks to two anonymous referees for their critical comments which help to improve the paper.

References

[1] Farin G, Sapidis N. Curvature and the fairness of curves and surfaces. IEEE Computer Graphics and Application 1989;9(2):52–7.
[2] Su BQ, Liu DY. Computational geometry. Shanghai: Shanghai Science and Technology Press, 1981 (in Chinese).
[3] Mehlum E. Nonlinear splines. In: Barnhill RE, Riesenfeld RF, editors. Computer aided geometric design, New York: Academic Press, 1974. p. 173–207.
[4] Parkinson DB, Moreton DN. Optimal biarc curve fitting. Computer Aided Design 1991;23(6):411–9.
[5] Brunnett G, Kiefer J. Interpolation with minimal-energy splines. Computer Aided Design 1994;26(2):137–44.
[6] Feldman M. Tight string method to fair piecewise linear curves. In: Sapidis NS, editor. Designing fair curves and surfaces, Philadelpha, PA: SIAM, 1994.
[7] Eck M, Jaspert R. Automatic fairing of point sets. In: Sapidis NS, editor. Designing fair curves and surfaces, Philadelpha, PA: SIAM, 1994.
[8] Pigounakis KG, Kaklis PD. Convexity-preserving fairing. Computer Aided Design 1996;28:981–94.
[9] Kjellander JAP. Smoothing of cubic parametric splines. Computer Aided Design 1983;15:288–93.
[10] Sapidis N, Farin G. Automatic fairing algorithm for B_spline curves. Computer Aided Design 1990;22:121–9.
[11] Meek DS, Walton DJ. Approximating of discrete data by G^1 arc splines. Computer Aided Design 1992;24:301–6.
[12] Parkinson DB. Optimised biarc curves with tension. Computer Aided Geometric Design 1992;9:207–18.
[13] Renz W. Interactive smoothing of digitized point data. Computer Aided Design 1982;14:267–9.
[14] Lindeberg T. Scale-space for discrete signals. IEEE Transactions on Pattern Analysis and Machine Intelligence 1990;12(3):234–54.
[15] Bolton KM. Biarc curves. Computer Aided Design 1975;7(2):89–92.
[16] Jin TG. A spline of biarc. Journal of Zhejiang University 1981;3:82–91.
[17] Yang X. Approximating NURBS curves by arc splines. In: Martin R, Wang N, editors. Geometric modeling and processing 2000: theory and applications, California: IEEE Computer Society Press, 2000. p. 175–83.

Xunnian Yang is a lecturer in the department of applied mathematics at Zhejiang Universisity, China. He obtained a BS in applied mathematics from Anhui University and a PhD in CAGD and Computer graphics from Zhejiang University in 1993 and 1998, respectively. His research interests include geometric modeling, CAD/CAM, computer graphics, image processing and computational geometry.

Guozhao Wang is a professor at the department of applied mathematics at Zhejiang University, China. He obtained a MS in applied mathematics from Zhejiang University. His research interests include computer aided geometric design, computer graphics and medical visualization.