



NURBS curve blending using extension^{*#}

Yong-jin LIU^{†1}, Rong-qi QIU¹, Xiao-hui LIANG²

¹*Tsinghua National Lab for Information Science and Technology, Tsinghua University, Beijing 100084, China*

²*State Key Lab of Virtual Reality Technology and Systems, Beihang University, Beijing 100191, China*

[†]E-mail: liuyongjin@tsinghua.edu.cn

Received Nov. 26, 2008; Revision accepted Feb. 16, 2009; Crosschecked Feb. 16, 2009

Abstract: Curve and surface blending is an important operation in CAD systems, in which a non-uniform rational B-spline (NURBS) has been used as the de facto standard. In local corner blending, two curves intersecting at that corner are first made disjoint, and then the third blending curve is added-in to smoothly join the two curves with G^1 - or G^2 -continuity. In this paper we present a study to solve the joint problem based on curve extension. The following nice properties of this extension algorithm are exploited in depth: (1) The parameterization of the original shapes does not change; (2) No additional fragments are created. Various examples are presented to demonstrate that our solution is simple and efficient.

Key words: Curve blending, Curve fairing, Curve extension, Non-uniform rational B-spline (NURBS)

doi:10.1631/jzus.A0820819

Document code: A

CLC number: TN391.72

INTRODUCTION

Curve blending between two sharply intersected curves has a wide range of applications in CAD systems. Local curve blending is referred to as replacing a chosen corner intersected by two curves with a new intermediate curve that smoothly joins the two original curves with G^1 - or G^2 -continuity. Usually a small portion around the corner is removed first and leaves the two curves disjoint from each other. Then the two curves are smoothly joined by the third blending curve. Many methods, in parametric or implicit form, have been proposed (Vida *et al.*, 1994; Mo and Zhao, 2006). In this paper we propose an efficient method for local curve blending, with the following distinct feature: one original curve is extended to smoothly join the other one, such that no additional blending curve is created. Another nice

property is that the shape and the parameterization of the two original curves will not be changed.

This paper is organized as follows. First, Sections 2 and 3 review the unclamping algorithm (Piegl and Tiller, 1997) and the extension algorithm (Hu *et al.*, 2002), respectively. Section 4 presents our proposed NURBS curve blending method using curve extension. Parameter optimization in the proposed blending method is discussed in Section 5. Some practical examples are presented in Section 6. Section 7 provides some discussion and also analyzes the limitations of the method. Finally Section 8 concludes this paper.

UNCLAMPING ALGORITHM

A p -degree B-spline curve is

$$C(u) = \sum_{i=0}^n N_{i,p}(u)P_i, \quad 0 \leq u \leq 1,$$

with non-periodic knot vector

$$U = [0, \underbrace{\dots, 0}_{p+1}, u_{p+1}, \dots, u_n, \underbrace{1, \dots, 1}_{p+1}].$$

* Project supported by the National Natural Science Foundation of China (Nos. 60603085 and 60736019), the Hi-Tech Research and Development (863) Program of China (No. 2007AA01Z336), and Tsinghua Basic Research Foundation, China

Expanded based on "Note on industrial applications of Hu's surface extension algorithm" by Yu Zang, Yong-Jin Liu, *et al.*, which appeared in GMP2008, LNCS, **4975**:304-314

The knot vector can be unclamped at either end (Piegl and Tiller, 1997). For example, for unclamping the vector at the right end, the curve

$$C(u) = \sum_{i=0}^n N_{i,p}(u)P_i,$$

$$U = [0, \dots, 0, \underbrace{u_{p+1}, \dots, u_n}_{p+1}, \underbrace{u_{n+1}, \dots, u_{n+1}}_{p+1}]$$

can be represented by

$$C(u) = \sum_{i=0}^n N_{i,p}(u)P'_i,$$

$$U = [0, \dots, 0, \underbrace{u_{p+1}, \dots, u_n}_{p+1}, \underbrace{u_{n+1}, \dots, u_{n+p+1}}_{p+1}],$$

where P'_i 's are determined by

- (1) Set $P_j^{-1} = P_j, j = n - p + 1, \dots, n + p + 1$;
- (2) $P_j^i = \begin{cases} P_j^{i-1}, & j = n - p + 1, \dots, n - i - 1, \\ \frac{P_j^{i-1} - (1 - \alpha_{i,j})P_{j-1}^i}{\alpha_{i,j}}, & j = n - i, \dots, n, \end{cases}$
- $\alpha_{i,j} = \frac{u_{n+1} - u_j}{u_{i+j+2} - u_j}, i = 0, 1, \dots, p - 2$;
- (3) $P_j^i = \begin{cases} P_j, & j = 0, 1, \dots, n - p, \\ P_j^{p-2}, & j = n - p + 1, \dots, n. \end{cases}$

CURVE EXTENSION ALGORITHM

To extend the curve

$$C(u) = \sum_{i=0}^n N_{i,p}(u)P_i,$$

$$U = [0, \dots, 0, \underbrace{u_{p+1}, \dots, u_n}_{p+1}, \underbrace{1, \dots, 1}_{p+1}]$$

to a point p_1 , two steps are involved:

Step 1: Unclamp the curve at the right end by

$$C^1(u) = \sum_{i=0}^n N_{i,p}(u)P_i^1,$$

$$U^1 = [0, \dots, 0, \underbrace{u_{p+1}, \dots, u_n}_{p+1}, \underbrace{1, t_1, \dots, t_1}_p].$$

Step 2: Add one knot t_1 into the right end of the knot vector. Accordingly point p_1 is added as the last control point $P_{n+1}^{(1)}$ of the extended curve

$$C^{(1)}(u) = \sum_{i=0}^{n+1} N_{i,p}(u)P_i^{(1)},$$

$$U_1 = [0, \dots, 0, \underbrace{u_{p+1}, \dots, u_n}_{p+1}, 1, \underbrace{t_1, \dots, t_1}_{p+1}],$$

where t_1 is estimated by the chord-length approximation

$$t_1 = 1 + \|P_n - p_1\| \left/ \sum_{i=p}^n \|C(u_{i+1}) - C(u_i)\| \right.$$

Note that the newly added curve segment in the interval $(1, t_1]$ is influenced by the control point p_1 only.

By repeating the above process, we let the curve go through two more points p_2 and p_3 . Finally the curve is

$$C^{(3)}(u) = \sum_{i=0}^{n+3} N_{i,p}(u)P_i^{(3)},$$

$$U_3 = [0, \dots, 0, \underbrace{u_{p+1}, \dots, u_n}_{p+1}, 1, \underbrace{t_1, t_2, t_3, \dots, t_3}_{p+1}].$$

The curve segment in the interval $(1, t_3]$ is influenced by the control points p_1, p_2 and p_3 only.

Note that the knot vector U_3 can be normalized without a change of curve geometry as

$$U_3 = [0, \dots, 0, \underbrace{u_{p+1}}_{p+1}, \dots, \underbrace{u_n}_{t_3}, \underbrace{1}_{t_3}, \underbrace{t_1}_{t_3}, \underbrace{t_2}_{t_3}, \underbrace{t_3}_{t_3}, \dots, \underbrace{1}_{p+1}].$$

CURVATURE-CONTINUOUS CONNECTION

Basic principle

Consider two rational B-spline curves $C(u)$ and $\bar{C}(\bar{u})$ with a small gap in between. Let $H(u) = \sum_{i=0}^n N_{i,p}Q_i, \bar{H}(\bar{u}) = \sum_{i=0}^{\bar{n}} N_{i,\bar{p}}\bar{Q}_i$ be the homogeneous counterparts of the rational curves $C(u), \bar{C}(\bar{u})$ of degrees p, \bar{p} , respectively. We first extend the curve $H(u)$ to pass through three points q_1, q_2, q_3 . Then the three newly added control points $Q_{n+1}, Q_{n+2}, Q_{n+3}$ of $C(u)$ offer the necessary degrees of freedom

(DOFs) to curvature-continuously connect two curves $C(u)$, $\bar{C}(\bar{u})$.

Let $C(u)$ and $\bar{C}(\bar{u})$ be parameterized in $[0, t_3]$ and $[0, 1]$, respectively, and the two curves be connected between $C(t_3)$ and $\bar{C}(0)$. We choose the initial positions of q_1, q_2, q_3 by linear interpolation:

$$q_3 = \bar{H}(0), q_2 = \frac{1}{3}H(t_3) + \frac{2}{3}q_3, q_1 = \frac{2}{3}H(t_3) + \frac{1}{3}q_3. \quad (1)$$

For the three newly created control points Q_{n+1} , Q_{n+2} and Q_{n+3} , we adjust their positions to achieve two goals:

- (1) Make two curves $C(u)$ and $\bar{C}(\bar{u})$ curvature-continuous.
- (2) Optimize the curve by minimizing a fairing criterion.

The advantage of the method is that adjusting the positions of Q_{n+1} , Q_{n+2} , Q_{n+3} only affects the curve segment parameterized in $(1, t_3]$, and thus the original parts of the two curves are not changed.

In the following, let

$$C(u) = \sum_{i=0}^{n+3} N_{i,p}(u)w_i P_i / \sum_{i=0}^{n+3} N_{i,p}(u)w_i \quad (2)$$

and its homogeneous version

$$H(u) = \sum_{i=0}^{n+3} N_{i,p}(u)Q_i = \begin{pmatrix} A(u) \\ w(u) \end{pmatrix}, Q_i = \begin{pmatrix} w_i P_i \\ w_i \end{pmatrix}. \quad (3)$$

Denote that the curve after extension is parameterized with the following knot vector interval:

$$U = [0, \dots, 0, \underbrace{u_{p+1}, \dots, u_n}_{p+1}, 1, t_1, t_2, t_3, \dots, t_3].$$

Curvature continuity between two rational curves

The conditions of curvature continuity between $C(u)$ and $\bar{C}(\bar{u})$ are shown in (Wang et al., 2001) to be

$$\begin{cases} H(t_3) = \alpha_0 \bar{H}(0), \\ H'(t_3) = \alpha_1 \bar{H}(0) + \alpha_0 \beta_1 \bar{H}'(0), \\ H''(t_3) = 2\alpha_2 \bar{H}(0) + 2(\alpha_0 \beta_2 + \alpha_1 \beta_1) \bar{H}'(0) \\ \quad + \alpha_0 \beta_1^2 \bar{H}''(0), \end{cases} \quad (4)$$

where $\alpha_0, \alpha_1, \alpha_2, \beta_1, \beta_2$ are constant, which give too many unnecessary DOFs. To make a simple and intuitive solution, we choose $\alpha_0=1, \alpha_1=\alpha_2=\beta_2=0, \beta_1=\beta$. Then

$$H(t_3) = \bar{H}(0), H'(t_3) = \beta \bar{H}'(0), H''(t_3) = \beta^2 \bar{H}''(0). \quad (5)$$

Since $q_3 = \bar{H}(0)$, $H(t_3) = \bar{H}(0)$ is guaranteed automatically. We leave one DOF β for curve optimization by fairness.

The first derivative in Eq.(5) can be expanded by

$$H'(u) = \sum_{i=0}^{n+2} N_{i,p-1}(u)Q_i^{(1)}, Q_i^{(1)} = p \frac{Q_{i+1} - Q_i}{u_{i+p+1} - u_{i+1}}. \quad (6)$$

By the property of endpoint interpolation

$$H'(t_3) = Q_{n+2}^{(1)} = p \frac{Q_{n+3} - Q_{n+2}}{u_{n+p+3} - u_{n+3}} = \lambda(Q_{n+3} - Q_{n+2}),$$

where λ is a constant. Similarly,

$$\begin{cases} \bar{H}'(\bar{u}) = \sum_{i=0}^{\bar{n}-1} N_{i,\bar{p}-1}(\bar{u})\bar{Q}_i^{(1)}, \\ \bar{Q}_i^{(1)} = \bar{p} \frac{\bar{Q}_{i+1} - \bar{Q}_i}{\bar{u}_{i+\bar{p}+1} - \bar{u}_{i+1}}, \\ \bar{H}'(0) = \bar{Q}_0^{(1)} = \bar{p} \frac{\bar{Q}_1 - \bar{Q}_0}{\bar{u}_{\bar{p}+1} - \bar{u}_1} = \bar{\lambda}(\bar{Q}_1 - \bar{Q}_0). \end{cases} \quad (7)$$

For the second derivative in Eq.(5), we have

$$\begin{cases} H''(u) = \sum_{i=0}^{n+1} N_{i,p-2}(u)Q_i^{(2)}, \\ H''(t_3) = Q_{n+1}^{(2)} = (p-1) \frac{Q_{n+2} - Q_{n+1}}{u_{n+p+2} - u_{n+2}} \\ \quad = \chi_{n+3}Q_{n+3} - \chi_{n+2}Q_{n+2} + \chi_{n+1}Q_{n+1}, \end{cases} \quad (8)$$

$$\begin{cases} \bar{H}''(\bar{u}) = \sum_{i=0}^{\bar{n}-2} N_{i,\bar{p}-2}(\bar{u})\bar{Q}_i^{(2)}, \bar{Q}_i^{(2)} = (\bar{p}-1) \frac{\bar{Q}_{i+1} - \bar{Q}_i}{\bar{u}_{i+\bar{p}+1} - \bar{u}_{i+2}}, \\ \bar{H}''(0) = \bar{Q}_0^{(2)} = (\bar{p}-1) \frac{\bar{Q}_1 - \bar{Q}_0}{\bar{u}_{\bar{p}+1} - \bar{u}_2} = \bar{\chi}_2 \bar{Q}_2 - \bar{\chi}_1 \bar{Q}_1 + \bar{\chi}_0 \bar{Q}_0. \end{cases} \quad (9)$$

With Eqs.(6)~(9), the system of Eq.(5) becomes

$$\begin{aligned} \mathbf{P}_{n+3} &= \bar{\mathbf{P}}_0, \quad w_{n+3} = \bar{w}_0, \\ \begin{pmatrix} w_{n+3} \mathbf{P}_{n+3} \\ w_{n+3} \end{pmatrix} - \begin{pmatrix} w_{n+2} \mathbf{P}_{n+2} \\ w_{n+2} \end{pmatrix} &= \beta \frac{\bar{\lambda}}{\lambda} \begin{pmatrix} \bar{w}_1 \bar{\mathbf{P}}_1 - \bar{w}_0 \bar{\mathbf{P}}_0 \\ \bar{w}_1 - \bar{w}_0 \end{pmatrix}, \\ \chi_{n+3} \begin{pmatrix} w_{n+3} \mathbf{P}_{n+3} \\ w_{n+3} \end{pmatrix} - \chi_{n+2} \begin{pmatrix} w_{n+2} \mathbf{P}_{n+2} \\ w_{n+2} \end{pmatrix} + \chi_{n+1} \begin{pmatrix} w_{n+1} \mathbf{P}_{n+1} \\ w_{n+1} \end{pmatrix} \\ &= \beta^2 \left[\bar{\chi}_2 \begin{pmatrix} \bar{w}_2 \bar{\mathbf{P}}_2 \\ \bar{w}_2 \end{pmatrix} - \bar{\chi}_1 \begin{pmatrix} \bar{w}_1 \bar{\mathbf{P}}_1 \\ \bar{w}_1 \end{pmatrix} + \bar{\chi}_0 \begin{pmatrix} \bar{w}_0 \bar{\mathbf{P}}_0 \\ \bar{w}_0 \end{pmatrix} \right]. \end{aligned}$$

The above system can be solved by

$$\begin{cases} w_{n+3} = \bar{w}_0, \quad \mathbf{P}_{n+3} = \bar{\mathbf{P}}_0, \\ w_{n+2} = w_{n+3} - \beta \frac{\bar{\lambda}}{\lambda} (\bar{w}_1 - \bar{w}_0) = c_1 \beta + c_2, \\ \mathbf{P}_{n+2} = \frac{w_{n+3}}{w_{n+2}} \mathbf{P}_{n+3} - \beta \frac{\bar{\lambda}}{\lambda w_{n+2}} (\bar{w}_1 \bar{\mathbf{P}}_1 - \bar{w}_0 \bar{\mathbf{P}}_0) = \mathbf{v}_1 \beta + \mathbf{v}_2, \\ w_{n+1} = -\frac{\chi_{n+3}}{\chi_{n+1}} w_{n+3} + \frac{\chi_{n+2}}{\chi_{n+1}} w_{n+2} \\ \quad + \frac{\beta^2}{\chi_{n+1}} (\bar{\chi}_2 \bar{w}_2 - \bar{\chi}_1 \bar{w}_1 + \bar{\chi}_0 \bar{w}_0) = c_3 \beta^2 + c_4, \\ \mathbf{P}_{n+1} = -\frac{\chi_{n+3} w_{n+3}}{\chi_{n+1} w_{n+1}} \mathbf{P}_{n+3} + \frac{\chi_{n+2} w_{n+2}}{\chi_{n+1} w_{n+1}} \mathbf{P}_{n+2} + \frac{\beta^2}{\chi_{n+1} w_{n+1}} \\ \quad \cdot (\bar{\chi}_2 \bar{w}_2 \bar{\mathbf{P}}_2 - \bar{\chi}_1 \bar{w}_1 \bar{\mathbf{P}}_1 + \bar{\chi}_0 \bar{w}_0 \bar{\mathbf{P}}_0) = \mathbf{v}_3 \beta^2 + \mathbf{v}_4, \end{cases} \quad (10)$$

where c_i ($i=1, 2, 3, 4$) are scalar constants and \mathbf{v}_i ($i=1, 2, 3, 4$) are constant vectors.

CONNECTION OPTIMIZATION BY FAIRNESS

We use the value of β to optimize the connection by minimizing a fairness objective. Note that this optimization is done in the local area of connection; the original parts of the two connected curves are not changed. This is much desired in industrial applications. We regard it as the most distinct advantage of our method.

The second order energy (Wallner, 2007)

$$E = \int \| \mathbf{C}''(u) \|^2 du \quad (11)$$

is used to measure the fairness of the extended curve

$$\mathbf{C}(u) = \sum_{i=0}^{n+3} N_{i,p}(u) w_i \mathbf{P}_i / \sum_{i=0}^{n+3} N_{i,p}(u) w_i$$

with the knot vector

$$U = [0, \dots, 0, \underbrace{u_{p+1}, \dots, u_n}_{p+1}, 1, t_1, t_2, t_3, \dots, t_3, \underbrace{t_3, \dots, t_3}_{p+1}].$$

The system of Eq.(10) shows that $\mathbf{P}_{n+3}, w_{n+3}$ are fixed, and that $\mathbf{P}_{n+2}, w_{n+2}, \mathbf{P}_{n+1}, w_{n+1}$ are polynomials of β . From Eqs.(2) and (3), we have

$$w(u) = \beta^2 c_1(u) + \beta c_2(u) + c_3(u),$$

$$w'(u) = \beta^2 c_4(u) + \beta c_5(u) + c_6(u),$$

$$w''(u) = \beta^2 c_7(u) + \beta c_8(u) + c_9(u),$$

where c_i ($i=1, 2, \dots, 9$) denote known piecewise polynomials of u . After some tedious but simple calculations, we also have

$$\mathbf{C}(u) = \frac{\beta^4 \mathbf{v}_1(u) + \beta^2 \mathbf{v}_2(u) + \mathbf{v}_3(u)}{\beta^2 c_{10}(u) + \beta c_{11}(u) + c_{12}(u)},$$

where $\mathbf{v}_1(u), \mathbf{v}_2(u)$ and $\mathbf{v}_3(u)$ denote known vector polynomials of u , and

$$\mathbf{C}'(u) = \sum_{i=0}^6 \beta^i \mathbf{v}_{4+i}(u) / \sum_{j=0}^4 \beta^j c_{13+j}(u).$$

Let $\mathbf{A}(u)=w(u)\mathbf{C}(u)$. Then it is immediately seen that

$$\mathbf{A}''(u) = \sum_{i=0}^4 \beta^i \mathbf{v}_{11+i}(u).$$

Since

$$\mathbf{A}''(u) = w''(u)\mathbf{C}(u) + 2w'(u)\mathbf{C}'(u) + w(u)\mathbf{C}''(u),$$

we have

$$\begin{aligned} \mathbf{C}''(u) &= \frac{\mathbf{A}''(u) - (2w'(u)\mathbf{C}'(u) + w''(u)\mathbf{C}(u))}{w(u)} \\ &= \sum_{i=0}^8 \beta^8 \mathbf{v}_{16+i}(u) / \sum_{j=0}^6 \beta^6 c_{18+j}(u). \end{aligned}$$

Then

$$\begin{aligned} \| \mathbf{C}''(u) \|^2 &= (\mathbf{C}''(u))^T \mathbf{C}''(u) \\ &= \sum_{i=0}^{16} \beta^i c_{25+i}(u) / \sum_{j=0}^{12} \beta^j c_{42+j}(u), \end{aligned} \quad (12)$$

which is a rational polynomial function. With the decomposition of rational functions into a sum of

partial fractions, the evaluation of energy Eq.(11) has a closed-form solution, which is a function $E(\beta)$ of β , and the minimum value of $E(\beta)$ is readily obtained by numerical methods (Press et al., 2002).

EXAMPLES

Fig.1 shows a G^2 -connection of two disjoint circular arcs. Only NURBS can exactly represent circular arcs. The coordinates of three control points of the first arc are (438, 450), (438, 250) and (638, 250), their weights being 1, 1 and 2, respectively. The coordinates of three control points of the second arc are (750, 370), (878, 370) and (878, 250), their weights being 1, 1 and 2, respectively. Figs.2a~2d show G^2 -connections with $\beta=0.01, 0.1, 1.0$ and 5.0 , respectively. The fairness energy with $\beta=0.1$ (Fig.2b) is 1979.26. By applying the optimization method proposed in the previous section, the optimal β is found to be 0.084, where the fairness energy is 1978.92. Some portions of the minimization curve of β are shown in Fig.3.

With the tensor product property, one distinct feature of our method is its natural extension to NUBRS surfaces. Figs.4 and 5 give two industrial examples, where an isophote model (Poeschl, 1984) was used to inspect the surface quality.

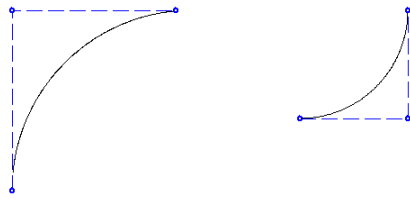


Fig.1 Two disjoint circular arcs needed to be G^2 joined
The control polygon with control points is shown in dashed lines

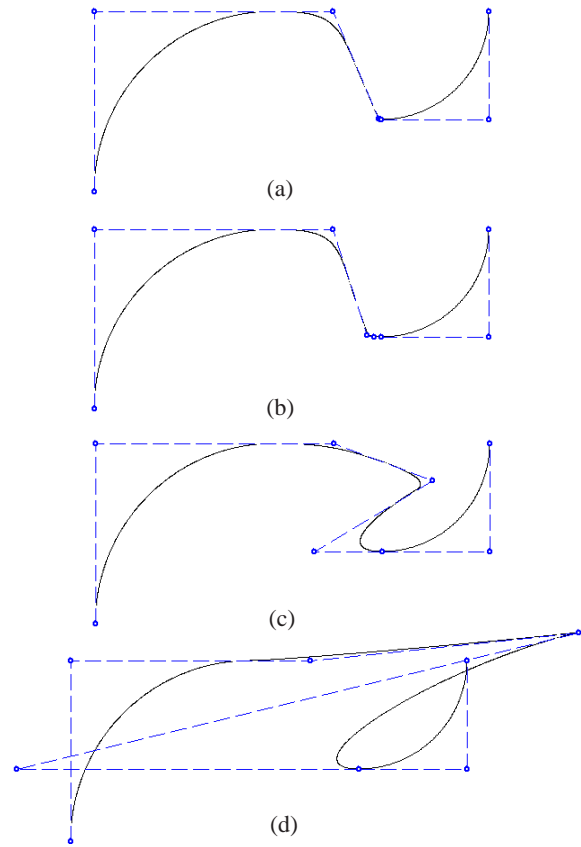


Fig.2 G^2 -connection of the two arcs in Fig.1 with different β . (a) $\beta=0.01$; (b) $\beta=0.1$; (c) $\beta=1.0$; (d) $\beta=5.0$
The control polygon with control points is shown in dashed lines

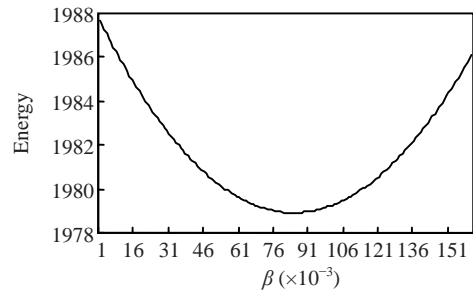


Fig.3 The extension energy function in terms of β

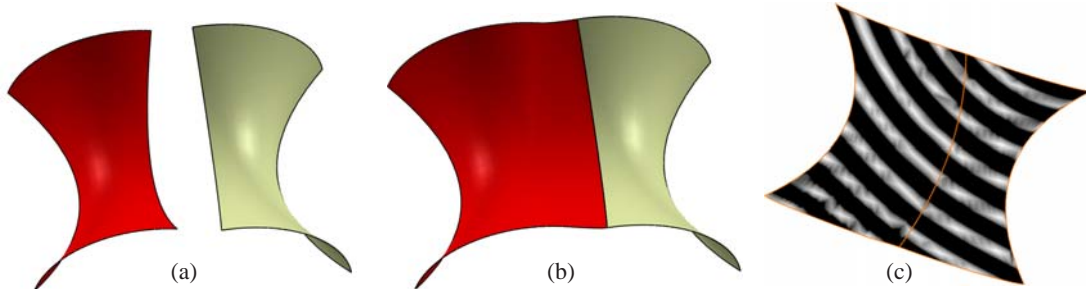


Fig.4 G^2 -connection of two NURBS surfaces
(a) Two disjoint patches; (b) Extending the red patch to G^2 -connect the gray patch; (c) The isophote model of the smoothly joined surfaces

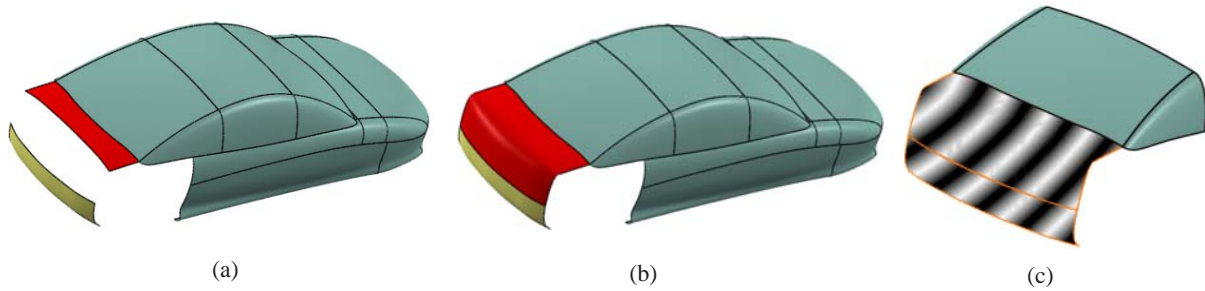


Fig.5 G²-connection of two disjoint portions in a car body

(a) Two disjoint patches shown in yellow and red, in a car body model; (b) Extending the red patch to G²-connect the yellow patch; (c) The isophote model of the smoothly joined surface

DISCUSSION

Rather different from traditional blending methods that usually plug-in the third curve segment in the gap of two curves (Vida *et al.*, 1994), our method extends one curve to smoothly connect the other curve. The extension is achieved in a strictly local way that is much desired in practical applications. So, compared with previous blending work, our method has the distinct advantage that it avoids creating unnecessary fragments of the small curve segment in between two disjoint curves.

The limitation of the proposed method lies in the fact that the computation of the exact integral of the rational function is very difficult. Currently we use the software Mathematica™ to analyze the exact integral. Things become much simpler if a B-spline curve is used instead. That is, by putting $w_i=1$ in Eq.(2), the system of Eq.(10) reduces to

$$\begin{cases} P_{n+3} = \bar{P}_0, & P_{n+2} = P_{n+3} - \beta \frac{\bar{\lambda}}{\lambda} (\bar{P}_1 - \bar{P}_0), \\ P_{n+1} = -\frac{\chi_{n+3}}{\chi_{n+1}} P_{n+3} + \frac{\chi_{n+2}}{\chi_{n+1}} P_{n+2} \\ \quad + \frac{\beta^2}{\chi_{n+1}} (\bar{\chi}_2 \bar{P}_2 - \bar{\chi}_1 \bar{P}_1 + \bar{\chi}_0 \bar{P}_0), \end{cases} \quad (13)$$

and curve $C(u)$ becomes

$$C(u) = \sum_{i=0}^{n+3} N_{i,p}(u) P_i$$

with knot vector

$$U = [\underbrace{0, \dots, 0}_{p+1}, u_{p+1}, \dots, u_n, 1, t_1, t_2, t_3, \dots, t_3] \cdot \underbrace{}_{p+1}$$

Then Eq.(11) becomes

$$E(\beta) = \int \left\| \sum_{i=n-1}^{n+1} N_{i,p-2}(u) P_i^{(2)} \right\|^2 du + \text{const.}, \quad (14)$$

where $P_{n-1}^{(2)}$, $P_n^{(2)}$, and $P_{n+1}^{(2)}$ are all linear functions of P_{n+1} and P_{n+2} . Eq.(14) is much simpler—it can be shown that $E(\beta)$ is a polynomial of β of order four:

$$E(\beta) = c_4 \beta^4 + c_3 \beta^3 + c_2 \beta^2 + c_1 \beta + c_0, \quad (15)$$

where c_i ($i=0, 1, \dots, 4$) are all constant. Finding the minimum of function $E(\beta)$ in Eq.(15) is very simple—it is equal to solving a cubic equation:

$$E'(\beta) = \beta^3 + a\beta^2 + b\beta + c = 0,$$

whose closed-form solution exists. This simplified solution was first studied in (Zang *et al.*, 2008) by a different way.

CONCLUSION

In this paper we present a study on connecting two disjoint rational B-spline curves with G²-continuity. Different from previous work that inserts the third curve fragment, our method possesses the following two distinct advantages:

(1) The shape and parameterization of two original curves do not change. This is much desired in many industrial applications.

(2) No additional fragments are created between two disjoint curves, so the CAD data are kept as small as possible.

One more advantage is that our method is naturally extended to make a G^n -connection between two rational B-spline curves and surfaces.

ACKNOWLEDGEMENT

The authors would like to thank Mr. Yu Zang for generating the results shown in Figs.4 and 5.

References

- Hu, S.M., Tai, C.L., Zhang, S., 2002. An extension algorithm for B-splines by curve unclamping. *Computer-Aided Design*, **34**(5):415-419. [doi:10.1016/S0010-4485(01)00108-7]
- Mo, G.L., Zhao, Y.N., 2006. A new extension algorithm for cubic B-splines based on minimal strain energy. *J. Zhejiang Univ. Sci. A*, **7**(12):2043-2049. [doi:10.1631/jzus.2006.A2043]
- Piegl, L., Tiller, W., 1997. *The NURBS Book*. Springer-Verlag, New York, NY.
- Poeschl, T., 1984. Detecting surface irregularities using isophotes. *Computer Aided Geometric Design*, **1**(2):163-168. [doi:10.1016/0167-8396(84)90028-1]
- Press, W., Teukolsky, S., Vetterling, W., Flannery, B., 2002. *Numerical Recipes in C++ (2nd Ed.)*. Cambridge University Press, Cambridge, UK.
- Vida, J., Martin, R.R., Varady, T., 1994. A survey of blending methods that use parametric surfaces. *Computer-Aided Design*, **26**(5):341-365. [doi:10.1016/0010-4485(94)90023-X]
- Wallner, J., 2007. Note on curve and surface energies. *Computer Aided Geometric Design*, **24**(8-9):494-498. [doi:10.1016/j.cagd.2007.05.007]
- Wang, J., Wang, G., Zheng, J., 2001. *Computer Aided Geometric Design*. China Higher Education Press, Beijing, China (in Chinese).
- Zang, Y., Liu, Y., Lai, Y., 2008. Note on industrial applications of Hu's surface extension algorithm. *LNCS*, **4975**:304-314.