Computer-Aided Design 44 (2012) 824–834

Contents lists available at SciVerse ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

G^1 continuous approximate curves on NURBS surfaces

Yi-Jun Yang [a], Wei Zeng [b], Cheng-Lei Yang [a,*], Xiang-Xu Meng [a], Jun-Hai Yong [c], Bailin Deng [d]

[a] School of Computer Science and Tech., Shandong Univ., Jinan, China
[b] Computer Science Department, State Univ. of New York at Stony Brook, USA
[c] School of Software, Tsinghua Univ., Beijing, China
[d] Computer Graphics and Geometry Laboratory, EPFL, Switzerland

ARTICLE INFO

Article history:
Received 2 November 2011
Accepted 11 April 2012

Keywords:
Approximation
Curves on surfaces
Reparameterization
Parabola

ABSTRACT

Curves on surfaces play an important role in computer aided geometric design. In this paper, we present a parabola approximation method based on the cubic reparameterization of rational Bézier surfaces, which generates G^1 continuous approximate curves lying completely on the surfaces by using iso-parameter curves of the reparameterized surfaces. The Hausdorff distance between the approximate curve and the exact curve is controlled under the user-specified tolerance. Examples are given to show the performance of our algorithm.

1. Introduction

Curves lying on free form surfaces play an important role in surface blending, surface–surface intersection, surface trimming and NC tool path generation for machining surfaces. An explicit and control point based representation of a curve on a surface is needed in many circumstances such as using the curve as a boundary curve of another surface [1,2]. Also many geometric properties derived from the control polygon, like the convex hull, can be computed directly from the explicit representation.

In the past 20 years, curves on surfaces have been studied extensively in the literature such as [3–11,1,12–14,2], whose methods can be generally categorized into two approaches. The first approach is to compute exact curves on surfaces directly [3,4,6–8,10,11,1] while the second approach generates an approximation of the exact curves [5,9,11,1,12–14]. The degree of exact curves is considerably high, which results in computationally demanding evaluations and introduces numerical instability in practice. The approximation approach uses a relatively low degree curve to approximate the exact curve, with some constraints imposed on the approximate curve. However, most approximation algorithms [5,9,11,1,12–14] generate curves not lying completely on the surfaces. If such a curve is used as a boundary curve of another surface, gaps may occur between the two surfaces, which is not acceptable in many Computer Aided Design (CAD) applications such as surface blending and surface trimming (see Fig. 1). To overcome the problem of approximate curves not completely on surfaces, a polyline approximation method was presented in [2]. The Hausdorff distance between the approximate curve and the exact curve is controlled under the user-specified distance tolerance. The approximate curve is ε_T–G^1 continuous where ε_T is the user-specified angle tolerance. The bottleneck of the polyline approximation method is the large number of subdivisions introduced to satisfy the user-specified distance and angle tolerances, which is problematic in many applications. To decrease the number of subdivisions, a hyperbola approximation method was presented in [15]. Iso-parameter curves of the quadratic reparameterized surfaces are utilized to approximate the target curves such that the number of subdivisions is reduced while the degree of the approximate curves is preserved compared with the polyline approximation method. Although the hyperbola approximation method can reduce the subdivisions to some extent, both methods can only generate ε_T–G^1 (only G^0) continuous projection curves and suffer from numerous subdivisions introduced to satisfy the angle tolerance.

The principal objective of this paper is to construct G^1 continuous approximate curves of low degree, which are lying completely on the NURBS surfaces. To the authors' knowledge, there is no literature related to constructing G^1 smooth approximate curves lying completely on the NURBS surfaces, which is necessary in applications such as tangent continuous surface blending and surface trimming [16–21] (see Fig. 1). G^1 continuity describes the tangent smoothness between stitched curves and surfaces, which is indispensable in practical high quality surface and model designs. One straightforward method to construct G^1 continuous approximate curves is to extend the polyline approximation method in [2]. The 2D domain curve in the parameter domain is first approximated by

* Correspondence to: No. 1500, Shunhua Road, Gaoxin District, Shandong University, 250101, Jinan, China. Tel.: +86 13854167071; fax: +86 0 10 62795460.
E-mail address: ycl.sdu@gmail.com (C.-L. Yang).

doi:10.1016/j.cad.2012.04.004

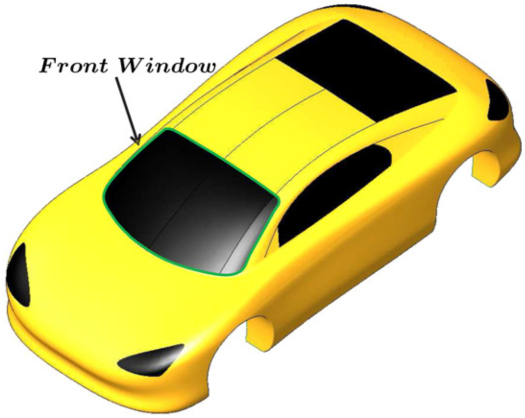

Fig. 1. Surface trimming operations on a car model: the front window boundary (the green curve) is represented as a G^1 continuous curve on the top surface of the car. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

G^1 continuous conic curves [22], which are then projected to the located NURBS surface by substituting the conic equations into the located NURBS surface. However the resulting approximate curve will have a relatively high degree $2(m+n)$, where m and n are the degrees of the NURBS surface in the u-direction and v-direction, respectively.

In order to generate lower-degree approximate curves, which are G^1 continuous and lie completely on the NURBS surface, we first approximate the domain curve with parabolic curves and subdivide the NURBS surface into rational Bézier surfaces. The domain curve is then subdivided such that the Hausdorff distance between the approximate curve and the exact curve is controlled under the tolerance ε_D, where ε_D is the user-specified distance tolerance. Finally, the parabolic curves are projected onto the NURBS surface using cubic reparameterization techniques. The degree of the approximate curve is $\max(2m+n, m+2n)$, which is lower than that $2m+2n$ of approximate curves obtained by the conic approximation method mentioned above. The main technique used in the parabola approximation method is that iso-parameter curves of the cubic reparameterized surfaces are utilized to generate G^1 continuous approximate curves. The main contributions of this paper are as follows.

1. We give the explicit representation of cubic reparameterizations for rational Bézier surfaces.
2. The exact curves are approximated by G^1 continuous low-degree curves, which are composed of iso-parameter curves of the cubic reparameterized surfaces and satisfy the given distance tolerance.

The organization of this paper is as follows. In Section 2, input and output handling is discussed. Section 3 describes how to preprocess the domain curves. Section 4 gives the explicit representation of the cubic reparameterized surfaces. Section 5 describes how to generate the G^1 continuous parabolas and control the Hausdorff distance between the approximate curve and the exact curve under the user-specified distance tolerance. Results are given in Sections 6 and 7 concludes the paper.

2. Algorithm overview

A NURBS curve is defined by

$$\mathbf{D}(t) = \frac{\sum_{k=0}^{n_d-1} N_k^d(t)\omega_k \mathbf{D}_k}{\sum_{k=0}^{n_d-1} N_k^d(t)\omega_k}, \quad t \in [0, 1],$$

where $\mathbf{D}_k$ are the control points, ω_k are the weights and N_k^d are the dth-degree B-spline basis functions defined on the knot vector

$$\mathbf{T} = \{\underbrace{0, \dots, 0}_{d+1}, t_{d+1}, \dots, t_{n_d-1}, \underbrace{1, \dots, 1}_{d+1}\}.$$

A NURBS surface in three dimensional space is defined by

$$\mathbf{S}(u, v) = \frac{\sum_{i=0}^{n_u-1}\sum_{j=0}^{n_v-1} N_i^p(u)N_j^q(v)\omega_{i,j}\mathbf{P}_{i,j}}{\sum_{i=0}^{n_u-1}\sum_{j=0}^{n_v-1} N_i^p(u)N_j^q(v)\omega_{i,j}}, \quad u, v \in [0, 1],$$

where $\mathbf{P}_{i,j}$ are the control points, $\omega_{i,j}$ are the weights and $N_i^p(u)$, $N_j^q(v)$ are the pth-degree and qth-degree B-spline basis functions defined on the knot vectors

$$\mathbf{U} = \{\underbrace{0, \dots, 0}_{p+1}, u_{p+1}, \dots, u_{n_u-1}, \underbrace{1, \dots, 1}_{p+1}\},$$

and

$$\mathbf{V} = \{\underbrace{0, \dots, 0}_{q+1}, v_{q+1}, \dots, v_{n_v-1}, \underbrace{1, \dots, 1}_{q+1}\},$$

respectively. Assume that we have a NURBS curve $\mathbf{D}(t)$ lying in the parameter domain of the surface $\mathbf{S}$. Let $\widetilde{\mathbf{D}}(t)$ denote the image curve obtained by substituting $\mathbf{D}(t)$ into the surface equation of $\mathbf{S}$. We attempt to obtain a spatial low-degree G^1 continuous curve $\widetilde{\mathbf{C}}(t)$ lying completely on the surface $\mathbf{S}$ to approximate the curve $\widetilde{\mathbf{D}}(t)$. The main algorithm flow is described as follows.

1. Divide the NURBS surface into rational Bézier surfaces by knots insertion and the domain curve into monotonic rational Bézier curves.
2. Approximate each monotonic rational Bézier curve by a parabolic curve with same end points and same end tangent directions in the parameter domain, which corresponds to a vertical or horizontal segment in the parameter domain of the cubic reparameterized surface.
3. Subdivide the domain curve so that the Hausdorff distance between the mapped curves of the parabolas and that of the NURBS curve is under the user-specified tolerance ε_D.
4. Compute the mapped curves of the parabolas by evaluating the iso-parameter curves of the cubic reparameterized Bézier surfaces.

The following sections illustrate how to generate the approximate curve.

3. Domain curves

Given a NURBS surface, we subdivide it into rational Bézier surfaces by knots insertion (see Fig. 2). Given a NURBS curve $\mathbf{D}(t)$ in the parameter domain of the surface (see Fig. 3), we use parabolas to approximate it. First, the NURBS curve $\mathbf{D}(t)$ is subdivided as follows. Subdivide the NURBS curve (see Fig. 4) at

- the knot positions of $\mathbf{D}(t)$;
- the parameter values where $\mathbf{D}(t)$ crosses a knot value in the u-direction or a knot value in the v-direction.

After the above subdivision, each segment of curve $\mathbf{D}(t)$ is a rational Bézier curve defined by

$$\mathbf{C}_1(t) = \frac{\sum_{i=0}^{n} B_i^n(t)\omega_i\mathbf{P}_i}{\sum_{i=0}^{n} B_i^n(t)\omega_i}, \quad 0 \le t \le 1,$$

where $\mathbf{P}_i = (u_i, v_i)^T$. Each segment is approximated by parabolic curves in the parameter domain with same end points and same end tangent directions in the following sections.

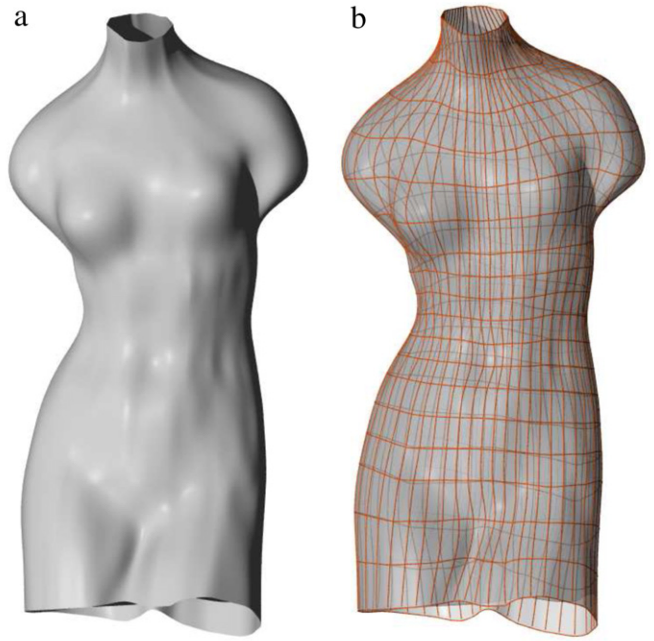

Fig. 2. NURBS surface subdivision: (a) the original NURBS surface (venus); (b) the resulting rational Bézier surfaces.

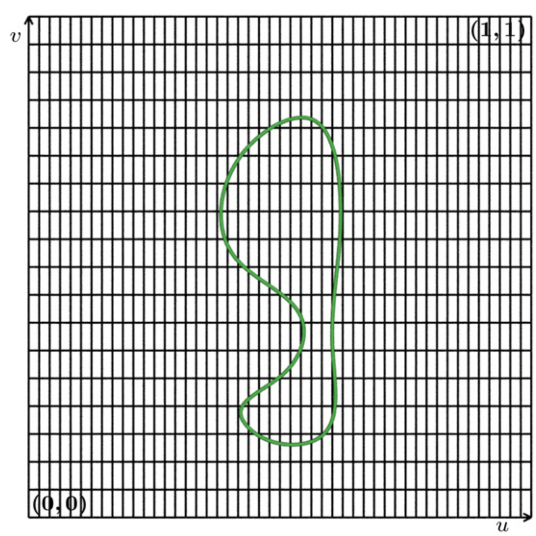

Fig. 3. A NURBS curve in the venus parameter domain.

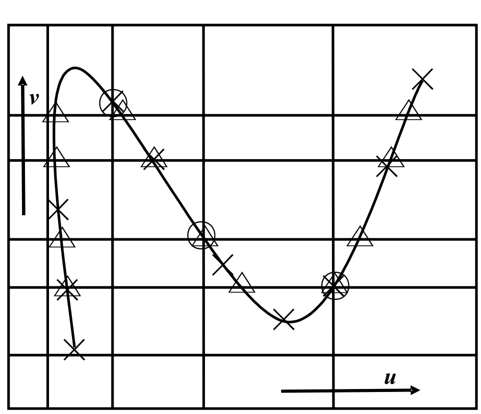

Fig. 4. NURBS curve division: $\times$ denotes knot position of $\mathbf{D}(t)$, $\bigcirc$ denotes the position where the curve crosses a knot value in the u-direction and Δ denotes the position where the curve crosses a knot value in the v-direction.

4. Transformations of rational Bézier surfaces

Before we describe how to approximate the 2D NURBS curve with parabolas, we first introduce the reparameterizations of rational Bézier surfaces in this section, which is the key technique for the parabola construction.

4.1. Möbius transformations of rational Bézier surfaces

A rational Bézier surface can be represented in the following form

$$\mathbf{X}(u,v) = \frac{\sum_{i=0}^{m}\sum_{j=0}^{n} B_i^m(u)B_j^n(v)\omega_{i,j}\mathbf{P}_{i,j}}{\sum_{i=0}^{m}\sum_{j=0}^{n} B_i^m(u)B_j^n(v)\omega_{i,j}}, \quad u \in [0,1],\ v \in [0,1], \tag{1}$$

where $\mathbf{P}_{i,j}$ are the control points, $\omega_{i,j}$ are the weights, $B_i^m(u)$ and $B_j^n(v)$ are the Bernstein polynomials. For a rational Bézier surface, each parameter is subjected to a Möbius transformation as follows.

$$u = u(s) = \frac{(\alpha-1)s}{2\alpha s - s - \alpha}, \quad \alpha \in (0,1) \tag{2}$$

and

$$v = v(t) = \frac{(\beta-1)t}{2\beta t - t - \beta}, \quad \beta \in (0,1). \tag{3}$$

Applying the transformations (2) and (3) to surface (1) results in the rational Bézier surface

$$\mathbf{X}(s,t) = \frac{\sum_{i=0}^{m}\sum_{j=0}^{n} B_i^m(s)B_j^n(t)\widetilde{\omega}_{i,j}\mathbf{P}_{i,j}}{\sum_{i=0}^{m}\sum_{j=0}^{n} B_i^m(s)B_j^n(t)\widetilde{\omega}_{i,j}}, \quad s \in [0,1],\ t \in [0,1],$$

where $\widetilde{\omega}_{i,j} = \omega_{i,j}(1-\alpha)^i\alpha^{m-i}(1-\beta)^j\beta^{n-j}$ [23]. An example of the Möbius transformation of a rational Bézier surface is given in Fig. 5(b), where the distribution of the vertical iso-parameter curves is more uniform compared with the original surface parameterization shown in Fig. 5(a). Linear Möbius transformations cannot change the shape of the iso-parameter curves. What changes is the distribution of the iso-parameter curves.

4.2. Quadratic transformations of rational Bézier surfaces

To introduce the quadratic reparameterization [24], α and β in Eqs. (2) and (3) are not constants any more. They are redefined as linear interpolations of another set of parameters as follows.

$$\begin{cases} \alpha = \alpha_1 t + \alpha_2(1-t), & \alpha_1, \alpha_2 \in (0,1) \\ \beta = \beta_1 s + \beta_2(1-s), & \beta_1, \beta_2 \in (0,1) \end{cases} \tag{4}$$

α in Eq. (2) is defined as a linear function of t, with coefficients α_1 and α_2, and β in Eq. (3) is defined as a linear function of s, with coefficients β_1 and β_2. As a result of more freedom, the surface degree will be raised accordingly. Applying the quadratic transformations (2)–(4) to surface (1) results in a rational Bézier surface of degree $m+n$ in both s-direction and t-direction [15]. As a consequence of introducing additional free parameters, quadratic reparameterization can change the shape of the iso-parameter curves as well as the distribution of the iso-parameter curves. In fact the quadratic reparameterization is a linear interpolation of Möbius transformations imposed on the opposite boundaries. An example is given in Fig. 5(c), where the horizontal iso-parameter curves are more uniform compared with the Möbius transformation results in Fig. 5(b). A vertical/horizontal segment in the parameter domain of the reparameterized surface corresponds to a hyperbola in the parameter domain of the original surface,

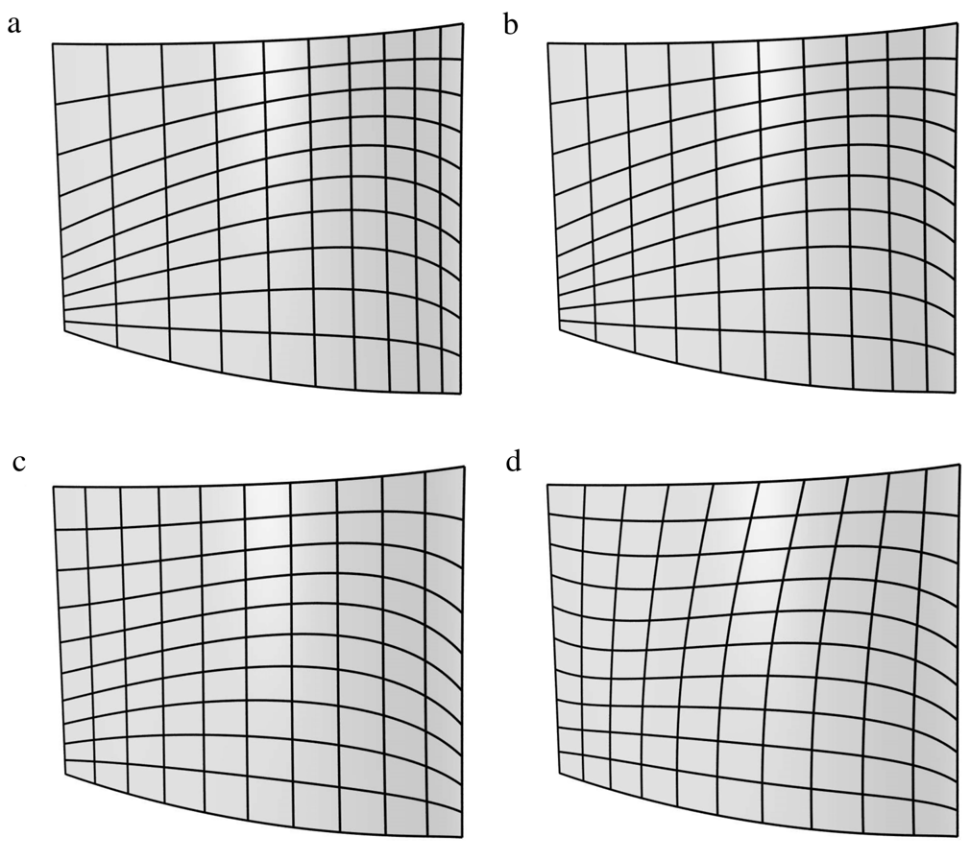

Fig. 5. Transformations of a rational Bézier surface: (a) the original surface parameterization; (b) optimal Möbius transformation with coefficients $\alpha = 0.5$ and $\beta = 0.6$; (c) optimal quadratic transformation with coefficients $\alpha_1 = 0.39, \alpha_2 = 0.60, \beta_1 = 0.65$ and $\beta_2 = 0.64$; (d) optimal cubic transformation with coefficients $\alpha_1 = 0.7, \alpha_2 = 0.4, \alpha_3 = 0.4, \beta_1 = 0.6, \beta_2 = 0.7$ and $\beta_3 = 0.7$.

which is utilized to reduce the curve subdivisions in [15]. A s-hyperbola curve, which corresponds to a s iso-parameter curve, only has three variables $\alpha_1, \alpha_2, \beta$ while the G^1 continuity will impose four constraints (two end points and two end tangent directions) on the hyperbola. Same statement holds for a t-hyperbola. Thus we cannot obtain G^1 continuous approximate curves by only utilizing iso-parameter curves of quadratic reparameterized surfaces. To achieve G^1 continuity, cubic reparameterization of rational Bézier surfaces is introduced in the next subsection.

4.3. Cubic transformations of rational Bézier surfaces

To introduce the cubic reparameterization, α and β in Eqs. (2) and (3) are redefined as quadratic interpolations of another set of parameters as follows.

$$\begin{cases} \alpha(t) = B_0^2(t)\alpha_1 + B_1^2(t)\alpha_2 + B_2^2(t)\alpha_3, & \alpha_1, \alpha_2, \alpha_3 \in (0, 1) \\ \beta(s) = B_0^2(s)\beta_1 + B_1^2(s)\beta_2 + B_2^2(s)\beta_3, & \beta_1, \beta_2, \beta_3 \in (0, 1) \end{cases} \quad (5)$$

where $B_i^2(s)$ and $B_i^2(t)$, $i = 0, 1, 2$, are the ith quadratic Bernstein polynomials of s and t, respectively. α in Eq. (2) is redefined as a quadratic function of t, with coefficients α_1, α_2 and α_3 and β in Eq. (3) is redefined as a quadratic function of s, with coefficients β_1, β_2 and β_3. As a result of more freedom, the surface degree will be raised accordingly. Applying the cubic transformations (2), (3) and (5) to surface (1) results in the rational Bézier surface

$$\mathbf{X}(s,t) = \frac{\sum_{k_1=0}^{m+2n}\sum_{k_2=0}^{2m+n} B_{k_1}^{m+2n}(s)B_{k_2}^{2m+n}(t)\widehat{\omega}_{k_1,k_2}\mathbf{Q}_{k_1,k_2}}{\sum_{k_1=0}^{m+2n}\sum_{k_2=0}^{2m+n} B_{k_1}^{m+2n}(s)B_{k_2}^{2m+n}(t)\widehat{\omega}_{k_1,k_2}},$$

$$s \in [0,1],\ t \in [0,1],$$

which is of degree $(m+2n)$ in the s-direction and degree $(2m+n)$ in the t-direction. The control points $\mathbf{Q}_{k_1,k_2}$ and their weights $\widehat{\omega}_{k_1,k_2}$ of the reparameterized surface are as follows.

$$\mathbf{Q}_{k_1,k_2} = \frac{\sum_{i=\max(k_1-2n,0)}^{\min(k_1,m)}\sum_{j=\max(k_2-2m,0)}^{\min(k_2,n)} c_{k_2-j,i}d_{k_1-i,j}R_{i,j}\omega_{i,j}\cdot\mathbf{P}_{i,j}}{\sum_{i=\max(k_1-2n,0)}^{\min(k_1,m)}\sum_{j=\max(k_2-2m,0)}^{\min(k_2,n)} c_{k_2-j,i}d_{k_1-i,j}R_{i,j}\omega_{i,j}}$$

and

$$\widehat{\omega}_{k_1,k_2} = \sum_{i=\max(k_1-2n,0)}^{\min(k_1,m)}\sum_{j=\max(k_2-2m,0)}^{\min(k_2,n)} c_{k_2-j,i}d_{k_1-i,j}R_{i,j}\omega_{i,j},$$

where

$$R_{i,j} = \frac{\binom{m}{i}\binom{n}{j}}{\binom{m+2n}{k_1}\binom{2m+n}{k_2}},$$

$$\begin{aligned} c_{l,k} = & \sum_{e_1=\max(0,l+k-2m)}^{\min\left(\left\lfloor\frac{l}{2}\right\rfloor,k\right)} \sum_{e_2=\max(0,l+2k-2m-2e_1)}^{\min(k-e_1,l-2e_1)} \sum_{e_3=\max(l+k-m-e_2-2e_1,0)}^{\min\left(\left\lfloor\frac{l-2e_1-e_2}{2}\right\rfloor,m-k\right)} \\ & \times \binom{k}{e_1}\binom{k-e_1}{e_2}\binom{m-k}{e_3}\binom{m-k-e_3}{l-2e_1-e_2-2e_3} \\ & \times (-1)^{m-k}2^{l-2e_1-2e_3}(\alpha_3-1)^{e_1}(\alpha_2-1)^{e_2}(\alpha_1-1)^{k-e_2-e_1} \\ & \times (\alpha_2)^{l-2e_1-e_2-2e_3}\alpha_3^{e_3}\alpha_1^{m+e_2+2e_1+e_3-k-l} \end{aligned}$$

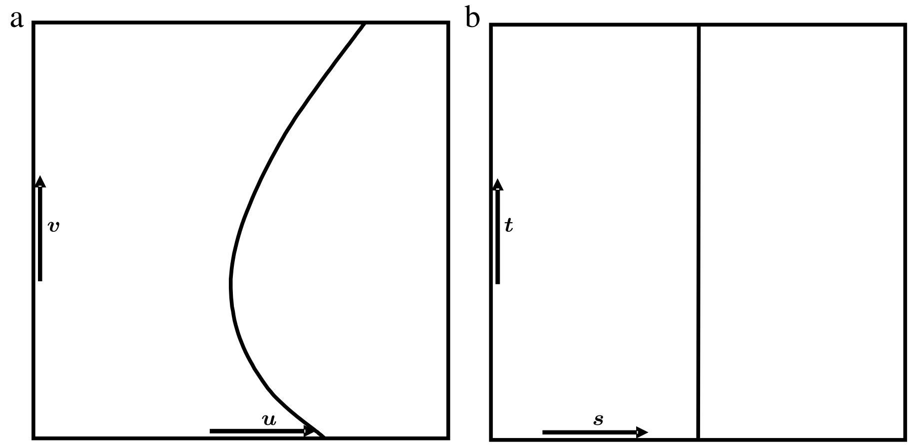

Fig. 6. Domain curves lying in the parameter domains of the original surface and the cubic reparameterized surface with the coefficients $\alpha_1 = 0.2$, $\alpha_2 = 0.8$, $\alpha_3 = 0.3$, $\beta_1 = 0.4$, $\beta_2 = 0.4$ and $\beta_3 = 0.4$: (a) the corresponding parabolic curve in the original surface parameter domain; (b) the s iso-parameter line with the coefficient $s = 0.5$ in the parameter domain of the reparameterized surface.

and

$$\begin{aligned} d_{l,k} = & \sum_{e_1=\max(0,l+k-2n)}^{\min\left(\left\lfloor \frac{l}{2}\right\rfloor,k\right)} \sum_{e_2=\max(0,l+2k-2n-2e_1)}^{\min(k-e_1,l-2e_1)} \sum_{e_3=\max(l+k-n-e_2-2e_1,0)}^{\min\left(\left\lfloor \frac{l-2e_1-e_2}{2}\right\rfloor,n-k\right)} \\ & \times \binom{k}{e_1}\binom{k-e_1}{e_2}\binom{n-k}{e_3}\binom{n-k-e_3}{l-2e_1-e_2-2e_3} \\ & \times (-1)^{n-k}2^{l-2e_1-2e_3}(\beta_3-1)^{e_1}(\beta_2-1)^{e_2}(\beta_1-1)^{k-e_2-e_1} \\ & \times (\beta_2)^{l-2e_1-e_2-2e_3}\beta_3^{e_3}\beta_1^{n+e_2+2e_1+e_3-k-l}. \end{aligned}$$

The cubic reparameterization is a quadratic interpolation of the Möbius reparameterizations imposed on the two opposite boundaries and the surface interior. To better illustrate how surface parameterization changes differently in these three types of transformations, one example is given in Fig. 5, where we try to optimize the uniformity of the surface parameterization by these three transformations. To compute the coefficients of the optimal transformation, we sample the coefficient space uniformly and select the coefficients with the minimal uniformity energy. From Fig. 5, we can see that the cubic transformation is more powerful than the Möbius and quadratic transformations in changing surface parameterizations. All the three reparameterizations mentioned above would not change the shape of the surface. What changes is the surface parameterization. By introducing cubic reparameterization, a certain form of parabolic curves can be utilized to approximate the target curves in the original surface parameter domain, which can be transformed into vertical or horizontal line segments in the parameter domain of the reparameterized surface by properly choosing the reparameterization coefficients (see Fig. 6). The details of generating such a parabola are described in the next section.

5. Approximate parabolic curves

A good approximation algorithm should control the Hausdorff distance between the approximate curve and the exact curve. To control the Hausdorff distance between the mapped curves of the approximate parabolic curves and that of the NURBS curve under the user-specified tolerance ϵ_D, the target curve is first subdivided into monotonic segments $\{\mathbf{C}_i(t) = (u_i(t), v_i(t))\}$ in the (u, v) parameter domain, where the coordinate functions $u_i(t)$ and $v_i(t)$ are monotonic with respect to t for each segment $\mathbf{C}_i(t)$. To obtain monotonic sub-segments, we split the target curve at points where the derivative vectors there have vanishing u or v components. Then s or t iso-parameter lines lying in the parameter domain of the cubic reparameterized surface, which correspond to parabolas in the parameter domain of the original surface, are utilized to approximate each monotonic segment. For simplicity, we denote parabolas corresponding to s or t iso-parameter lines in the reparameterized parameter domain by s-parabolas or t-parabolas. Given a monotonic curve with end points $\mathbf{C}(0) = \mathbf{P}_0(u_0, v_0)$, $\mathbf{C}(1) = \mathbf{P}_1(u_1, v_1)$ and end derivatives $\mathbf{C}'(0) = \mathbf{V}_0(m_0, n_0)$, $\mathbf{C}'(1) = \mathbf{V}_1(m_1, n_1)$, our aim is to find a s/t-parabola which has the same end points and same derivative directions as the given curve. To give a brief statement, we always suppose that the unmentioned components of the end derivatives are not zero in the following theorems and deductions. To construct a satisfying s/t-parabola for a given monotonic curve, we have the following theorem.

Theorem 1. *To construct an approximate parabola which has the same end points and same tangent directions as the given monotonic curve, we have the following statement.*

Type 1. *If $m_0 = 0$ or $m_1 = 0$, there exists a s-parabola which has the same end points and same tangent directions as the target curve.*

Type 2. *If $n_0 = 0$ or $n_1 = 0$, there exists a t-parabola which has the same end points and same tangent directions as the target curve.*

Type 3. *If*

$$\frac{(u_0-u_1)^2}{(v_0-v_1)^2} - \frac{m_0 m_1}{n_0 n_1} \geq 0,$$

there exists a s-parabola which has the same end points and same tangent directions as the target curve.

Type 4. *If*

$$\frac{(v_0-v_1)^2}{(u_0-u_1)^2} - \frac{n_0 n_1}{m_0 m_1} \geq 0,$$

there exists a t-parabola which has the same end points and same tangent directions as the target curve.

Theorem 1, which will be demonstrated later, serves as a guide for the parabola constructions. Each given monotonic curve will fall into one of the above four types. Curves of Type 1 and Type 2 could be seen as the degenerate monotonic cases, where the end derivative vectors have vanishing components. For the sake of computation simplicity, the rational Bézier surface is first subdivided or/and extended according to the two end points of the target curve such that the two end points lie on opposite boundary edges of the new surface parameter domain.

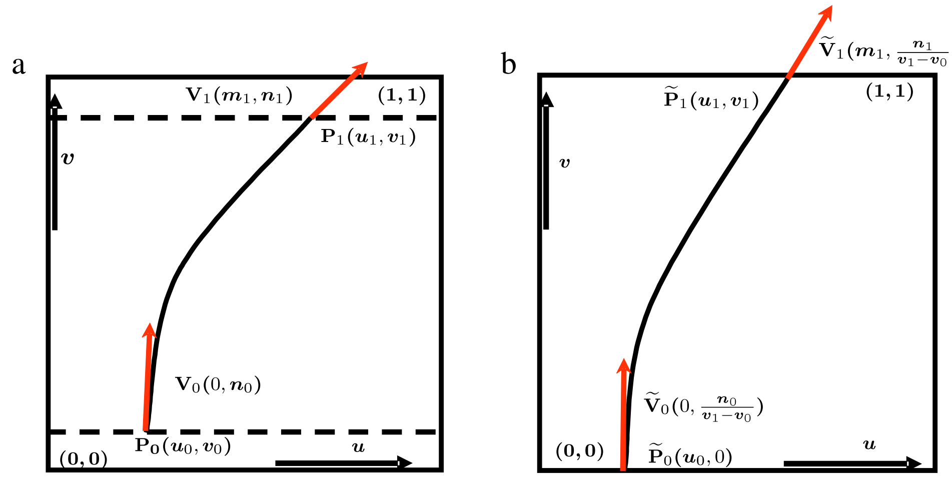

Fig. 7. A curve of Type 1 (a) in the parameter domain and the new curve (b) after v subdivisions.

After surface extensions or/and subdivisions, the two end points of the target curve lie on opposite boundaries of the new surface parameter domain (see Figs. 7(b) and 8(b)). The procedure of how to approximate such a target curve with an appropriate parabola is described as follows. Supposing the 3D mapped curve fixed, each horizontal/vertical line (see Fig. 6(b)) in the parameter domain of the cubic reparameterized surface corresponds to a parabola (see Fig. 6(a)) in the parameter domain of the original surface, which is utilized to approximate the target monotonic curve. Here we give the solution of how to approximate a curve of Type 1/Type 3 with a parabolic curve. Approximating the curve of Type 2/Type 4 with a parabolic curve can be handled similarly. For simplicity, let the coefficients β_3, β_2 and β_1 in Eq. (4) be the same and denote them as β. Without loss of generality, we always use the iso-parameter line s or $t = 0.5$ in the parameter domain of the cubic reparameterized surface to approximate the target curve in the following computations. The implicit representation of the approximate parabolic curve in the original surface parameter domain is as follows.

$$u(v) = \frac{av^2 + bv + c}{dv^2 + ev + f}, \tag{6}$$

where

$$\begin{cases} a = (4 - \alpha_1 - 2\alpha_2 - \alpha_3)\beta^2 + (2\alpha_1 + 2\alpha_2 - 4)\beta + 1 - \alpha_1 \\ b = 2(\beta - 1)(\alpha_1\beta - 2\beta + \alpha_2\beta - \alpha_1 + 1) \\ c = (\beta - 1)^2(1 - \alpha_1) \\ d = (2\beta - 1)^2 \\ e = -2(\beta - 1)(2\beta - 1) \\ f = (\beta - 1)^2. \end{cases} \tag{7}$$

For the curve denoted by Eq. (6), we have the following theorem.

Theorem 2. *The curve in Eq. (6) is a parabola.*

Proof. We make a Möbius transformation to v as follows.

$$v = \frac{(\beta - 1)t}{2\beta t - t - \beta}. \tag{8}$$

Substituting Eq. (8) into Eq. (6), we have

$$gu = hv^2 + pv + q, \tag{9}$$

where

$$\begin{cases} g = \beta^2(1 - \beta) \\ h = 2\alpha_1 + 8\beta^3 - 14\beta^2 + 9\beta - 2\alpha_2\beta + 4\alpha_2\beta^2 - \alpha_3\beta^3 \\ \quad + 11\alpha_1\beta^2 + \alpha_3\beta^2 - 5\alpha_1\beta^3 - 2\alpha_2\beta^3 - 8\alpha_1\beta - 2 \\ p = \beta(5\beta - 4\beta^2 + 4\alpha_1\beta^2 - 2 - 6\alpha_1\beta + 2\alpha_1) \\ q = -(1 - \alpha_1)\beta^2(1 - \beta). \end{cases}$$

Then Eq. (9) can be rewritten in the following form.

$$\left(v + \frac{p}{2h}\right)^2 = \frac{g\left(u + \frac{p^2}{4gh}\right)}{h}.$$

Let $y = v + \frac{p}{2h}$, $x = u + \frac{p^2}{4gh}$ and $r = \frac{g}{h}$. Finally we have

$$y^2 = rx,$$

which is a parabola. □

Let k_1 and k_2 be the start and end derivative ratios $\frac{m_0(v_1 - v_0)}{n_0}$ and $\frac{m_1(v_1 - v_0)}{n_1}$, respectively. To construct G^1 continuous curves, we apply the four G^1 constraints on the parabola expressed in Eq. (6) as follows.

$$u(0) = 1 - \alpha_1 = u_0 \tag{10}$$

$$u(1) = 1 - \alpha_3 = u_1 \tag{11}$$

$$u'(0) = \frac{2\beta(\alpha_1 - \alpha_2)}{1 - \beta} = k_1 \tag{12}$$

$$u'(1) = \frac{2(1 - \beta)(\alpha_2 - \alpha_3)}{\beta} = k_2. \tag{13}$$

How to approximate the target curve corresponding to four types of curves in Theorem 1 with parabolic curves is described as follows.

Type 1. For the curve of Type 1 (see Fig. 7(a)), if $m_0 = 0, k_1 = 0$ holds. As a valid coefficient β always lies in the open interval (0,1), from Eq. (12), we have

$$\alpha_2 = \alpha_1. \tag{14}$$

Substituting Eq. (14) into Eq. (13), we have

$$\beta = \frac{2(u_1 - u_0)}{2(u_1 - u_0) + k_2},$$

where β always lies in the open interval (0,1) for monotonic curves. The other cases of Type 1 can be verified similarly. Thus the demonstration of Theorem 1 for curves of Type 1 follows.

Type 2. Similarly we can construct the approximate parabolic curves for curves of Type 2 as for curves of Type 1 by exchanging u/v, s/t and α/β.

Type 3. From Eqs. (10) and (11), we can get the coefficients α_1 and α_3 as follows.

$$\alpha_1 = 1 - u_0$$
$$\alpha_3 = 1 - u_1.$$

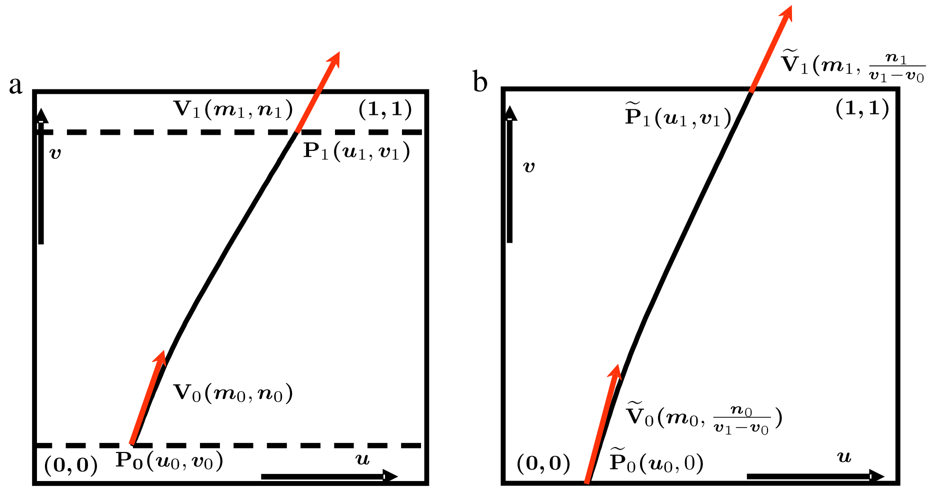

Fig. 8. A curve of Type 3 (a) in the parameter domain and the new curve (b) after v subdivisions.

From Eq. (12), we have

$$\beta = \frac{2(\alpha_2 - \alpha_3)}{2(\alpha_2 - \alpha_3) + k_2}. \tag{15}$$

Substituting Eq. (15) into Eq. (13), we obtain the following quadratic equation of the coefficient α_2.

$$4\alpha_2^2 - 4(\alpha_1 + \alpha_3)\alpha_2 + 4\alpha_3\alpha_1 + k_1 k_2 = 0,$$

whose roots can be expressed as follows.

$$\alpha_2 = \frac{(\alpha_1 + \alpha_3) \pm \sqrt{\Delta}}{2},$$

where

$$\Delta = (\alpha_1 - \alpha_3)^2 - k_1 k_2.$$

To define a valid cubic transformation, Δ must be no less than zero and α_2 and β both lie in the interval (0, 1). Here we always choose

$$\alpha_2 = \frac{(\alpha_1 + \alpha_3) + \sqrt{\Delta}}{2} \tag{16}$$

in the subsequent computations. As the curve is of Type 3 (see Fig. 8), the term Δ can be rewritten as follows.

$$\Delta = (u_0 - u_1)^2 - \frac{m_1 m_0 (v_0 - v_1)^2}{n_1 n_0}. \tag{17}$$

Substituting Eq. (17) into Eq. (16), we have

$$\alpha_2 = \frac{2 - (u_0 + u_1) + \sqrt{(u_0 - u_1)^2 - k_1 k_2}}{2}.$$

To guarantee that α_2 lies in the interval (0, 1), we have the following condition

$$0 < \alpha_2 = \frac{2 - (u_0 + u_1) + \sqrt{(u_0 - u_1)^2 - k_1 k_2}}{2} < 1,$$

which always holds for a monotonic curve. Also we can verify that the corresponding β also lies in the interval (0, 1). From the above deductions, the demonstration of Theorem 1 for curves of Type 3 follows.

Type 4. Similarly we can construct the approximate parabolic curves for curves of Type 4 as for curves of Type 3 by exchanging u/v, s/t and α/β.

Thus Theorem 1 is demonstrated. After we determine the coefficients, the curve in Eq. (6) can be expressed as a cubic rational Bézier curve as follows.

$$\mathbf{C_2}(t) = \frac{\omega_0 \mathbf{P}_0 B_0^3(t) + \omega_1 \mathbf{P}_1 B_1^3(t) + \omega_2 \mathbf{P}_2 B_2^3(t) + \omega_3 \mathbf{P}_3 B_3^3(t)}{\omega_0 B_0^3(t) + \omega_1 B_1^3(t) + \omega_2 B_2^3(t) + \omega_3 B_3^3(t)},$$

where the four control points and their weights of the cubic curve are as follows.

$$\begin{cases} \mathbf{P}_0 = (1 - \alpha_1, 0) \\ \mathbf{P}_1 = \left(\dfrac{\alpha_1 \beta - 2\beta\alpha_2 + \beta + 1 - \alpha_1}{\beta + 1}, \dfrac{1 - \beta}{\beta + 1} \right) \\ \mathbf{P}_2 = \left(\dfrac{\beta - 2 - 2\beta\alpha_2 + 2\alpha_2 + \beta\alpha_3}{\beta - 2}, \dfrac{2(\beta - 1)}{\beta - 2} \right) \\ \mathbf{P}_3 = (1 - \alpha_3, 1). \end{cases}$$

$$\begin{cases} \omega_1 = (\beta - 1)^2 \\ \omega_2 = \dfrac{1 - \beta^2}{3} \\ \omega_3 = \dfrac{\beta(2 - \beta)}{3} \\ \omega_4 = \beta^2. \end{cases}$$

A good approximation algorithm should control the Hausdorff distance between the approximate curve and the exact curve. To control the Hausdorff distance between the mapped curves of the parabolas and the 3D exact curve under the user-specified tolerance ε_D, we subdivide the target rational Bézier curves whose Hausdorff distance to their approximate hyperbolas are larger than the 2D tolerance d in the parameter domain, which can be obtained by a rational extension of the method presented in Section 4 of [2] as in Box I. In Box I, ε_D is the user-specified 3D tolerance, $\mathbf{P}_{i,j}$ are the control points of the located Bézier surface, $\omega_{i,j}$ are the weights of the Bézier surface, m and n are the degrees of the Bézier surface in the u-direction and v-direction, respectively. The Hausdorff distance $H(\mathbf{C}_1(t), \mathbf{C}_2(t))$ between the approximate curve (a parabola $\mathbf{C}_2(t)$) and the target curve (a rational Bézier curve $\mathbf{C}_1(t)$) can be obtained by an iterative method [25]. If $H(\mathbf{C}_1(t), \mathbf{C}_2(t)) \le d$, the Hausdorff distance between $\widetilde{\mathbf{C}}_2(t)$ and $\widetilde{\mathbf{C}}_1(t)$ is controlled under the user-specified tolerance ε_D where $\widetilde{\mathbf{C}}_2(t)$ and $\widetilde{\mathbf{C}}_1(t)$ are the mapped curves of $\mathbf{C}_2(t)$ and $\mathbf{C}_1(t)$ on their located surfaces, respectively. If this condition holds for all rational Bézier curves in the parameter domain, the Hausdorff distance between the approximate curve and the exact curve is controlled under ε_D. If the Hausdorff distance between the parabolic curves and the rational Bézier curve is larger than d, subdivide the rational Bézier curve at the shoulder point $\mathbf{C}_1(0.5)$. The subdivision

$$d = \frac{\varepsilon_D}{\left(\frac{\max\limits_{i,j} \omega_{i,j}}{\min\limits_{i,j} \omega_{i,j}}\right)^2 \left(n \max\limits_{i,h,k} \|\mathbf{P}_{h,i+1} - \mathbf{P}_{k,i}\| + m \max\limits_{i,h,k} \|\mathbf{P}_{i+1,h} - \mathbf{P}_{i,k}\|\right)} \tag{18}$$

Box I.

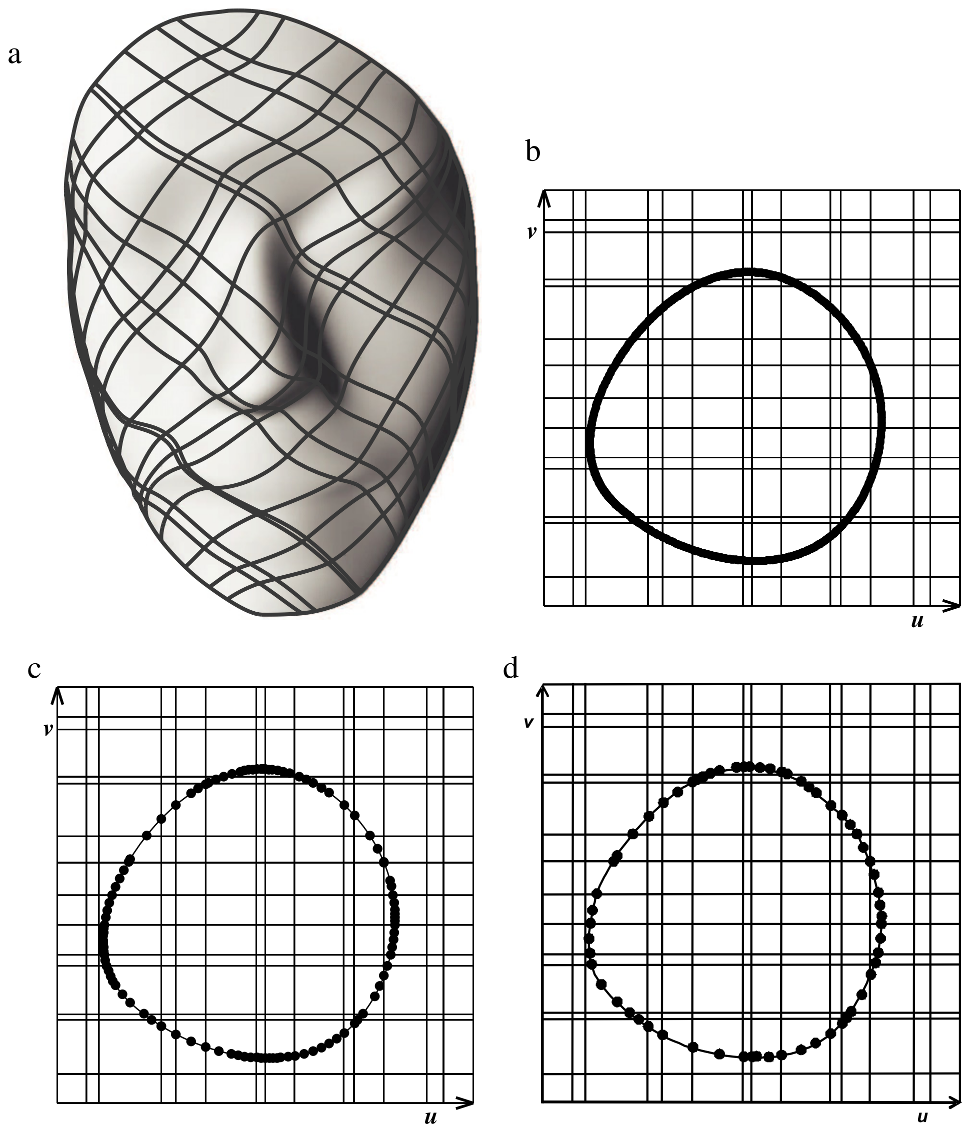

Fig. 9. Approximate parabolic, hyperbolic and polyline curves in the parameter domain of a face model: (a) the face model and its rational Bézier patches; (b) polylines for $\varepsilon_D = 0.0001$; (c) hyperbolae for $\varepsilon_D = 0.0001$; (d) parabolas for $\varepsilon_D = 0.0001$.

procedure is performed repeatedly until the distance tolerance is met for all sub-curves. For the NURBS curve which is located in the parameter domain of the surface shown in Fig. 9(a), Fig. 9 shows the approximate parabolic, hyperbolic [15] and polyline [2] curves for $\varepsilon_D = 0.0001$. Also the number of the approximate curves shown in Fig. 9 is listed in Table 1. From Table 1, we can see that the parabola approximation method decreases the number of subdivisions evidently compared with the polyline and hyperbola approximation methods. Furthermore, the parabola approximation method has no tangent discrepancies between adjacent 3D projection curves of the parabolas.

Table 1
Subdivision comparison of domain curves.

Distance tolerance	Polyline	Hyperbola	Parabola
1×10^{-1}	443	63	43
1×10^{-2}	448	69	44
1×10^{-3}	459	70	59
1×10^{-4}	700	110	77
1×10^{-5}	2118	230	131

6. Results

To show the performance of the algorithm presented in this paper, three examples are given below, which are all implemented in the environment with Intel Pentium IV CPU 2.0 GHZ, 1 G Memory, Microsoft Windows 7, and Microsoft Visual Studio 2008.

In the first example, the cubic, closed NURBS curve with 8 control points lying in the parameter domain of the bicubic human face model in Fig. 9(a) is mapped onto the human face model with 17 by 17 control points (see Fig. 10). The exact

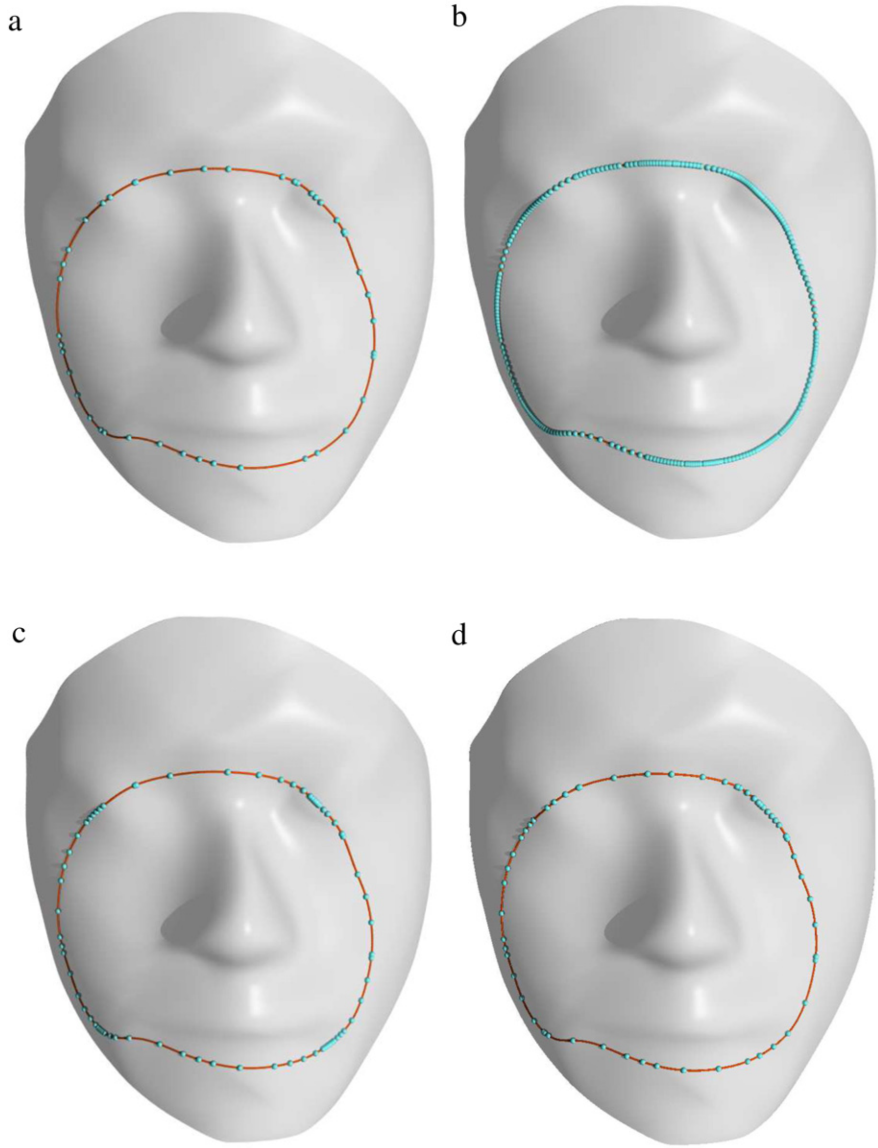

Fig. 10. Exact and approximate images of a NURBS curve on a face surface: (a) the exact curve and its subdivision; the approximate curve and its subdivisions generated by (b) the polyline approximation method and (c) the hyperbola approximation method and (d) the parabola approximation method.

Table 2
Results for a curve on a face surface.

	Exact	Polyline	Hyperbola	Parabola
Tolerance	–	$1\times10^{-3}/1°$	$1\times10^{-3}/1°$	1×10^{-3}
Degree	18	6	6	9
Number of segments	43	275	82	58
Distance to **S**	0	0	0	0
Continuity	G^1	$1°$-G^1	$1°$-G^1	G^1
Processor time (ms)	2.1	40	36	31

curve and its segmentation [1] are shown in Fig. 10(a). The results of the polyline and hyperbola approximation methods are shown in Fig. 10(b) and (c), respectively, where ε_D is set to 10^{-3} and ε_T is set to 1°. Also we show the result of our parabola approximation method in Fig. 10(d) where the distance tolerance ε_D is set to 10^{-3}. Results of the four algorithms are given in Table 2. The curves generated by the four algorithms all lie completely on the surface. The polyline approximation method and the hyperbola approximation method only generate ε_T–G^1 continuous curves, while the curves generated by the parabola approximation method is G^1 continuous. The degree of the curves generated by our parabola approximation method is 9, which is lower than that of the exact 18th-degree curve. Compared with the polyline and hyperbola approximation methods, less subdivisions are introduced in the parabola approximation method.

In the second example, a cubic NURBS curve with 12 control points in Fig. 3 lying in the parameter domain of the bicubic venus model in Fig. 2(a) is mapped onto the venus surface with 145 by 55 control points (see Fig. 11). The exact curve and its segmentations [1] are shown in Fig. 11(a). The results of the polyline and hyperbola approximation methods are shown in Fig. 11(b) and (c), respectively, where ε_D is set to 10^{-3} and ε_T is set to 1°. Also we show the result of our parabola approximation method in Fig. 11(d) where the distance tolerance ε_D is set to 10^{-3}. Results of the four algorithms are given in Table 3.

In the third example, a cubic curve consisting of 19 segments is mapped onto a top surface of the car model. The exact curve on the surface has more than 20 subdivisions, as shown in Fig. 12(a). The results of the polyline and hyperbola approximation methods are given in Fig. 12(b) and (c), respectively, where the distance tolerance to the exact curve is set to 10^{-3} and the angle tolerance is set to 1°. The result of our parabola approximation method is given in Fig. 12(d). Results of the four algorithms are given in Table 4.

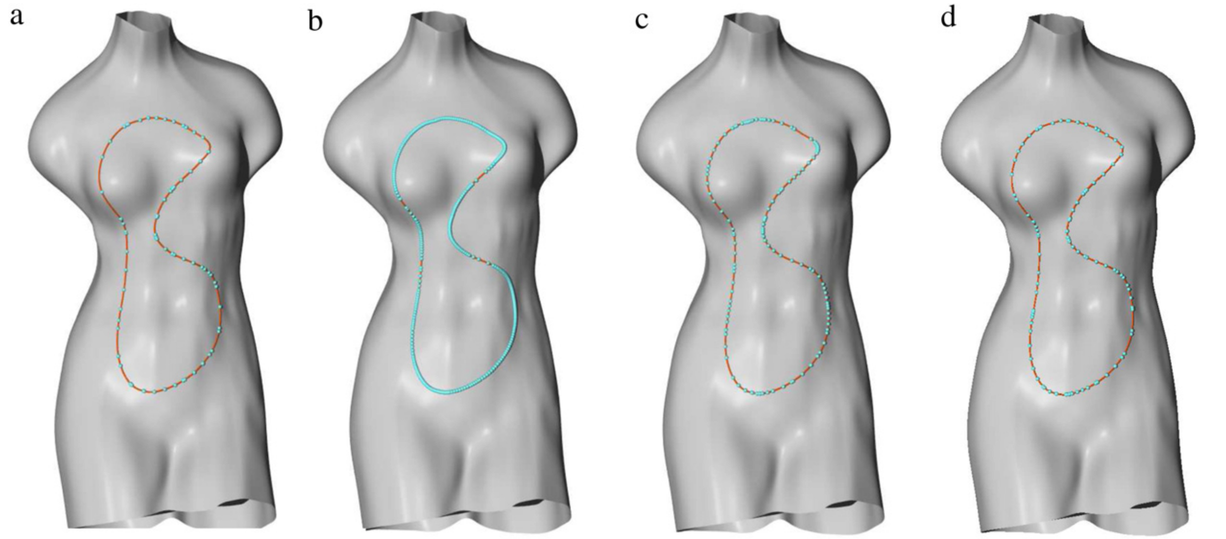

Fig. 11. Exact and approximate images of a NURBS curve on a venus surface: (a) the exact curve and its subdivision; the approximate curve and its subdivisions generated by (b) the polyline approximation method and (c) the hyperbola approximation method and (d) the parabola approximation method.

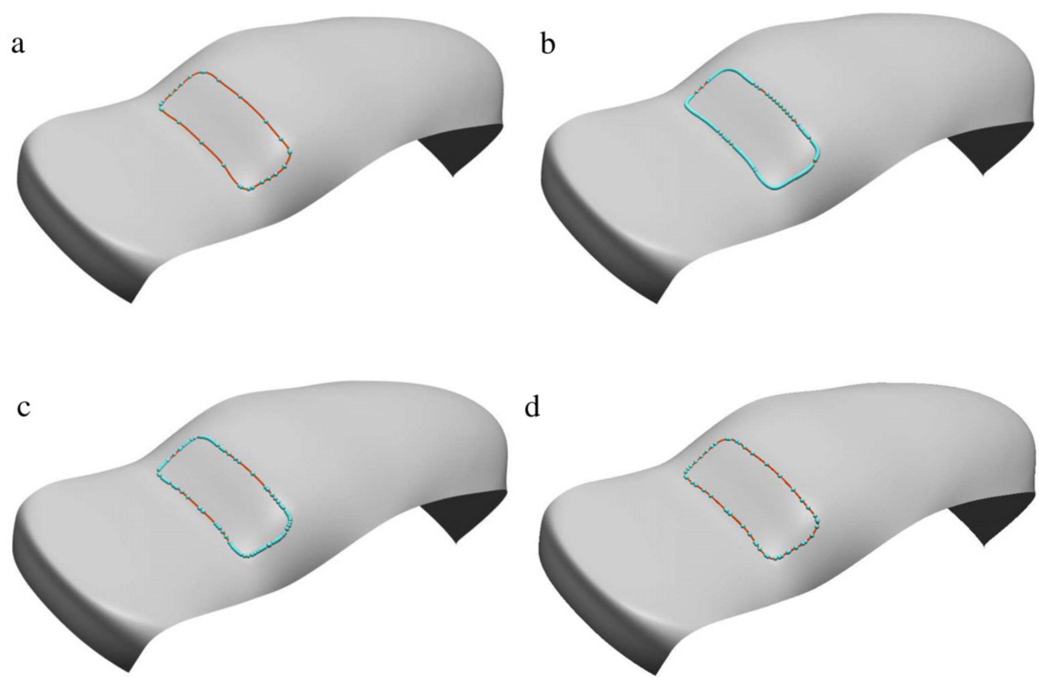

Fig. 12. Exact and approximate images of a NURBS curve on a car surface: (a) the exact curve and its subdivision; the approximate curve and its subdivisions generated by (b) the polyline approximation method and (c) the hyperbola approximation method and (d) the parabola approximation method.

Table 3
Results for a curve on a venus surface.

	Exact	Polyline	Hyperbola	Parabola
Tolerance	–	$1 \times 10^{-3}/1°$	$1 \times 10^{-3}/1°$	1×10^{-3}
Degree	18	6	6	9
Number of segments	70	367	149	96
Distance to **S**	0	0	0	0
Continuity	G^1	$1°$-G^1	$1°$-G^1	G^1
Processor time (ms)	3.6	73	59	47

Table 4
Results for a curve on a car surface.

	Exact	Polyline	Hyperbola	Parabola
Tolerance	–	$1 \times 10^{-3}/1°$	$1 \times 1^{-3}/1°$	1×10^{-3}
Degree	18	6	6	9
Number of segments	28	993	127	50
Distance to **S**	0	0	0	0
Continuity	G^1	$1°$-G^1	$1°$-G^1	G^1
Processor time (ms)	1.5	189	54	21

The four methods mentioned above all generate curves lying completely on the surface. From Tables 2–4, compared with the polyline and hyperbola approximation methods, the parabola approximation method reduces the number of subdivisions to an acceptable extent, which can be comparable with that of the exact method. Furthermore, by using a cubic reparameterization technique, the parabola approximation method generates G^1 composite rational Bézier curves lying completely on the NURBS surface, which can be further converted to a C^1 NURBS curve using Zheng's method [26]. G^1 continuity is indispensable for many CAD applications, such as G^1 continuous surface blending and surface trimming. Compared with the exact method, the degree of the approximate curve generated by our parabola approximation method is $\max(2m+n, m+2n)$, which is much lower than that $(m+n)d$ of the exact curve where d is the degree of the domain curve, m and n are the degrees of the NURBS surface in the u-direction and v-direction, respectively.

7. Conclusions

Based on the cubic reparameterization of rational Bézier surfaces, a parabola approximation algorithm for computing a curve on a NURBS surface has been presented in this paper. First the initial parabolic approximation of the domain curve is generated. The domain curve is then subdivided repeatedly until the Hausdorff distance between the approximate curve and the exact curve is under the distance tolerance. The main technique of our method is that we utilize iso-parameter curves of the cubic reparameterized surfaces to generate G^1 continuous approximate curves lying completely on the NURBS surfaces, which have a much lower degree than that of the exact curves. Compared with the polyline and hyperbola approximation methods, both the curve smoothness and the curve subdivision are improved.

Acknowledgments

We thank Prof. Wenping Wang for his useful comments and suggestions that motivated this work. This work was partly supported by the China National Natural Science Foundation (U1035004, 60703028, and 61070093), Shandong Province Outstanding Young Scientist Research Award Fund (BS2009DX026). The fifth author was supported by Chinese 973 Program (2010CB328001), Chinese 863 Program (2012AA040902) and the ANR-NSFC (60911130368).

References

[1] Renner G, Weiß V. Exact and approximate computation of *B*-spline curves on surfaces. Computer-Aided Design 2004;36(4):351–62.
[2] Yang YJ, Cao S, Yong JH, Zhang H, Paul JC, Sun JG, et al. Approximate computation of curves on *B*-spline surfaces. Computer-Aided Design 2008; 40(2):223–34.
[3] DeRose T. Composing Bézier simplexes. ACM Transactions on Graphics 1988; 7(3):198–221.
[4] DeRose T, Goldman R, Hagen H, Mann S. Functional composition algorithm via blossoming. ACM Transactions on Graphics 1993;12(2):113–35.
[5] Flöry S, Hofer M. Constrained curve fitting on manifolds. Computer-Aided Design 2008;40(1):25–34.
[6] Hu Y-P, Sun T-C. Moving a *B*-spline surface to a curve—a trimmed surface matching algorithm. Computer-Aided Design 1997;29(6):449–55.
[7] Knuth D. The art of computer programming-fundamental, Vol. 1. Reading (MA): Addison-Wesley; 1969.
[8] Lee E. Computing a chain of blossom, with application to products of splines. Computer Aided Geometric Design 1994;11(6):597–620.
[9] Lee I-K. Curve reconstruction from unorganized points. Computer Aided Geometric Design 2000;17(2):161–77.
[10] Liu W, Mann S. An optimal algorithm for expanding the composition of the polynomials. ACM Transactions on Graphics 1997;16(2):155–78.
[11] Renner G, Weiß V. Curves on surfaces: anlysis and new solutions. In: Cripps R, editor. The mathematics of surfaces VIII. Information geometers. 1998. p. 57–71.
[12] Saux E, Daniel M. An improved Hoschek intrinsic parametrization. Computer Aided Geometric Design 2003;20(8–9):513–21.
[13] Wang WP, Pottmann H, Liu Y. Fitting *B*-spline curves to point clouds by curvature-based squared distance minimization. ACM Transactions on Graphics 2006;25(2):214–38.
[14] Yang HP, Wang WP, Sun JG. Control point adjustment for *B*-spline curve approximation. Computer-Aided Design 2004;36(7):639–52.
[15] Yang YJ, Zeng W, Zhang H, Paul JC, Yong JH. Projection of curves on *B*-spline surfaces using quadratic reparameterization. Journal of Graphical Models 2010;72(5):47–59.
[16] Braid IC. Non-local blending of boundary models. Computer-Aided Design 1997;29(2):89–100.
[17] Liu D. GC^1 continuity conditions between two adjacent rational Bézier surface patches. Computer Aided Geometric Design 1990;7:151–63.
[18] Piegl LA, Tiller W. The NURBS book. 2nd ed. New York: Springer; 1997.
[19] Piegl LA, Tiller W. Filling *n*-sided regions with NURBS patches. The Visual Computer 1999;15(2):77–89.
[20] Yang YJ, Yong JH, Zhang H, Paul JC, Sun JG. A rational extension of Piegl's method for filling *n*-sided holes. Computer-Aided Design 2006;38(11): 1166–78.
[21] Shi X, Wang T, Yu P. A practical construction of G^1 smooth biquintic *B*-spline surfaces over arbitrary topology. Computer-Aided Design 2004;36(5):414–24.
[22] Ahn YJ. Conic approximation of planar curves. Computer-Aided Design 2001; 33(12):867–72.
[23] Farin G. Curves and surfaces for CAGD: a practical guide. 4th ed. San Diego: Academic Press; 1999.
[24] Yang YJ, Yong JH, Zhang H, Paul JC, Sun JG. Optimal parameterizations of Bézier surfaces. In: 2nd international symposium on visual computing. ISVC'06. 2006. p. 672–81.
[25] Elber Gershon, Thomas A. Grandine: hausdorff and minimal distances between parametric freeforms in R2 and R3. In: Geometric modeling and processing. 2008. p. 191–204.
[26] Zheng JM. C^1 NURBS representation of G^1 composite rational Bezier curves. Computing 2009;86(2):257–68.