PERGAMON Computers and Mathematics with Applications 41 (2001) 39–50

An International Journal
computers & mathematics
with applications

www.elsevier.nl/locate/camwa

Quadratic B-Spline Curve Interpolation

FUHUA CHENG
Department of Computer Science, University of Kentucky
Lexington, KY 40506-0046, U.S.A.

XUEFU WANG
SoftImage, Montreal, Quebec, H2X 2V2, Canada

B. A. BARSKY
Computer Science Division-EECS, University of California
Berkeley, CA 94720-1776, U.S.A.

(Received August 1999; revised and accepted April 2000)

Abstract—Traditional approach in performing even-degree B-spline curve/surface interpolation would generate undesired results. In this paper, we show that the problem is with the selection of interpolation parameter values, not with even-degree B-spline curves and surfaces themselves. We prove this by providing a new approach to perform quadratic B-spline curve interpolation. This approach generates quadratic B-spline curves whose quality is comparable to that of cubic interpolating B-spline curves. This makes quadratic B-spline curves better choices than cubic B-spline curves in some applications in graphics and geometric modeling, since it is cheaper to render/subdivide a quadratic curve and it is easier to find the intersection of two quadratic curves.

Keywords—B-spline curves/surfaces, Interpolation, Constrained minimization, Digital filter.

1. INTRODUCTION

A problem with the current B-spline curve/surface interpolation techniques is that even-degree B-spline curve/surface interpolation is completely overlooked, because it is believed that even-degree B-spline curves and surfaces are not suitable for the curve and surface interpolation problem [1]. Indeed, given the joints of an even-degree B-spline curve, one may not be able to determine the control vertices of the curve [2]. This follows from the observation that when viewed as *digital filters*, the *transfer functions* of even-degree and odd-degree B-splines behave differently. The transfer function of a filter is obtained from the Fourier transform of the filter and describes the behavior of the filter under all input frequencies.

The odd-degree B-splines behave like a low-pass filter, passing zero-frequency signals with no attenuation and other signals with more and more attenuation as the frequency of the signal increases. The even-degree B-splines attenuate the low and high frequencies only slightly. Therefore, even-degree B-splines generate curves and surfaces containing high magnitude high frequencies. This means that odd-degree B-splines generate smoother curves.

Moreover, even-degree B-splines considerably attenuate the midfrequencies. Actually, they attenuate certain frequencies completely, such as frequency 1/4 for B-splines of degree 2. In

This work is supported by NSF Grant (DMI-9400823).

0898-1221/01/$ - see front matter Typeset by AMS-TEX
PII: S0898-1221(00)00254-6

terms of B-spline curves, this means that given the joints of an even-degree B-spline curve, we may not be able to determine the control vertices of the curve, since information about the original signal—at least at one frequency—is lost. The odd-degree B-splines do not have this problem. Hence, odd-degree B-spline curves and surfaces are more appropriate for interpolation problems, while even-degree B-splines should be avoided. This is consistent with the comment made by Ahlberg, Nilson and Walsh: "polynomial splines of even degree interpolating to a prescribed function at mesh points need not exist" [1].

A recent result by Rabut [3], however, shows that the above observation does not hold if the uniform even-degree B-splines are defined with joints at half integer values instead of integer values. According to his result, even-degree uniform B-spline curves and surfaces defined this way, when viewed as digital filters, behave in a similar fashion to odd-degree B-spline curves and surfaces (with joints at the integer values). The study was based on *central* B-splines [4] instead of B-splines defined using the Cox-de Boor recurrence relation or divided differences [5,6]. Note that central B-splines have their joints at half integer values when the degrees are even, while B-splines defined using the Cox-de Boor recurrence relation or divided differences will always have their joints at integer values.

Rabut's result is interesting but does not conflict with our previous observation which was based on even-degree curves and surfaces defined with joints at integer values. However, his result does raise an important point: *the problem with traditional even-degree B-spline curve/surface interpolation is probably with the way it is performed, not with even-degree B-spline curves and surfaces themselves since they are constructed with equally good basis functions as odd-degree curves and surfaces.* For instance, by performing interpolation at midpoints of the knots, Marsden presents a good solution to quadratic interpolation when the knots are uniform [7]. Note that when the knots are uniform, performing interpolation at midpoints of the knots would generate a B-spline curve as considered by Rabut. This approach has been extended to knot average for higher even-degree B-spline curves by de Boor [6]. But midpoint method does not work well when the knots are not uniform. (See (c) and (d) in Figures 4–7.)

In this paper, we show that the above point is indeed the case by presenting a quadratic B-spline curve interpolation technique that works well. The new approach tries to perform the interpolation process at the parameter values where the maxima of quadratic B-spline basis functions occur. It covers the midpoint method when the knots are uniform and works well for nonuniform case. It generates quadratic interpolating B-spline curves as good as cubic interpolating B-spline curves in most of the cases (actually in some cases, the results are better). Conics and quadrics, two important primitives in geometric modeling, can be represented precisely by even-degree rational spline curves and surfaces, and representing conics and quadrics using even-degree (rational) B-spline curves and surfaces requires less time and effort in computing, rendering, and performing intersection operation than using cubic NURBS curves and surfaces. Thus, this shows that even-degree spline curves and surfaces should play a more significant role in various applications rather than just being "introduced principally in situations in which they combine with odd-degree splines to help clarify the total picture", as Ahlberg, Nilson and Walsh put it [1].

2. QUADRATIC B-SPLINE CURVES

Let (t_i) be a set of given knots. A quadratic (order three) B-spline basis function defined on (t_i) is of the following form:

$$B_{i,3}(t) = \begin{cases} \dfrac{t-t_i}{t_{i+2}-t_i}\dfrac{t-t_i}{t_{i+1}-t_i}, & t_i \le t < t_{i+1}, \\ \dfrac{t-t_i}{t_{i+2}-t_i}\dfrac{t_{i+2}-t}{t_{i+2}-t_{i+1}} + \dfrac{t_{i+3}-t}{t_{i+3}-t_{i+1}}\dfrac{t-t_{i+1}}{t_{i+2}-t_{i+1}}, & t_{i+1} \le t < t_{i+2}, \\ \dfrac{t_{i+3}-t}{t_{i+3}-t_{i+1}}\dfrac{t_{i+3}-t}{t_{i+3}-t_{i+2}}, & t_{i+2} \le t < t_{i+3}, \end{cases} \tag{2.1}$$

and zero elsewhere. If the knots are from t_{-2} to t_{n+3}, then for a set of $n+3$ control points $\mathbf{P}_0, \ldots, \mathbf{P}_{n+2}$, one can construct a quadratic B-spline curve $\mathbf{C}(t)$ as follows:

$$\mathbf{C}(t) = \sum_{i=-2}^{n} \mathbf{P}_{i+2} B_{i,3}(t), \qquad t_0 \le t \le t_{n+1}. \tag{2.2}$$

On the other hand, if a set of interpolation points $\mathbf{Q}_0, \ldots, \mathbf{Q}_{n+1}$ are given, one might not be able to construct a quadratic B-spline curve that interpolates $\mathbf{Q}_i$ at t_i, $i = 0, 1, \ldots, n+1$ in a desired fashion [2]. The reason for this was depicted from the viewpoint of digital filters in the previous section. Nevertheless, it is also possible to provide intuitive reasoning from the geometric point of view. Consider the interpolation points A, B, C, D, E, F, and G and a desired interpolating curve in Figure 1. These points form a zigzag polyline. Therefore, the desired curve would have inflection points between B and C, C and D, D and E, and E and F, such as the interpolating cubic B-spline curve in Figure 2 (the thinner one, width = 1). The desired curve cannot be generated using an even-degree B-spline curve with joints at the interpolation points. This is because the segments of an even-degree B-spline curve are either concave-up or concave-down. An even-degree B-spline curve with joints at these points would be forced to use these points as inflection points and overshoot, such as the interpolating quadratic B-spline curve in Figure 2 (the wider one, width = 3).

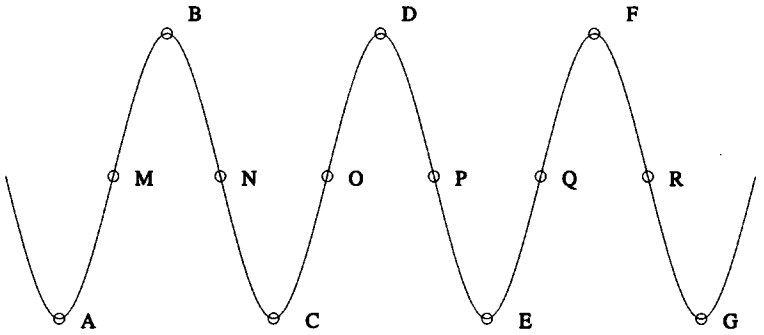

Figure 1. Zigzagged data points and a desired interpolation curve.

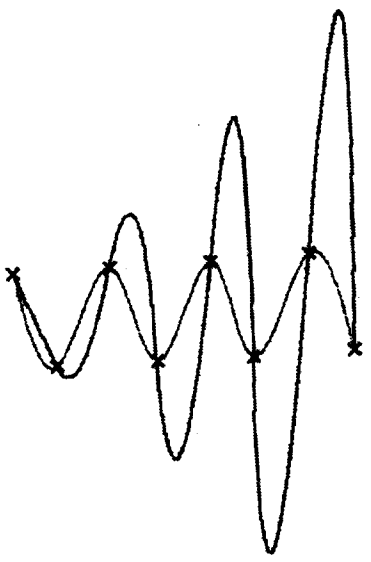

Figure 2. Ordinary cubic and quadratic B-spline curve interpolation.

In the next section, we will show that this problem can be overcome by changing the way interpolation is performed. The interpolation will be performed so that the joints of the quadratic B-spline curve (such as M, N, O, P, Q, and R in Figure 1), instead of the interpolation points, would act as inflection points. In this case, the shape characteristic of even-degree curve segments fits the situation well, since the shape of the segments MBN, NCO, ODP, PEQ, QFR is either concave-up or concave-down. The questions then are: how should the knots be defined and what should be interpolated? We will solve these questions for the general case, i.e., nonuniform quadratic B-spline curves, in the next section.

3. QUADRATIC B-SPLINE CURVE INTERPOLATION

Rabut's result indicates one ought to find features that are common to both uniform odd-degree B-spline curves and central even-degree B-spline curves for clues in performing even-degree interpolation. One notices that in both representations, the maxima of their basis functions occur at parameter knots. This indicates that even-degree central B-splines would also behave like low-pass filters and, consequently, would generate similar results as cubic B-splines when performing interpolation at the parameter knots. For ordinary uniform quadratic B-splines, since the maxima of the basis functions occur at the midpoints of the parameter knots, this implies that one ought to perform interpolation at the midpoint of the parameter knots, as Marsden has proposed, instead of the parameter knots. For nonuniform quadratic B-splines, the maxima of the basis functions do not necessarily occur at the midpoints, one has to find the parameter values where the maxima occur and perform interpolation there.

The maximum of the quadratic B-spline basis function $B_{i,3}(t)$ defined in (2.1) occurs at

$$t = t_{i+1} + \frac{(t_{i+3} - t_{i+1})\,(t_{i+2} - t_{i+1})}{(t_{i+2} - t_i) + (t_{i+3} - t_{i+1})}, \tag{3.1}$$

a point between t_{i+1} and t_{i+2}. Denoting this point by s_{i+1}, it can be seen that

$$\frac{t_{i+2} - s_{i+1}}{s_{i+1} - t_{i+1}} = \frac{s_{i+1} - t_i}{t_{i+3} - s_{i+1}} = \frac{t_{i+2} - t_i}{t_{i+3} - t_{i+1}}. \tag{3.2}$$

The geometric meaning of this equation is shown in Figure 3.

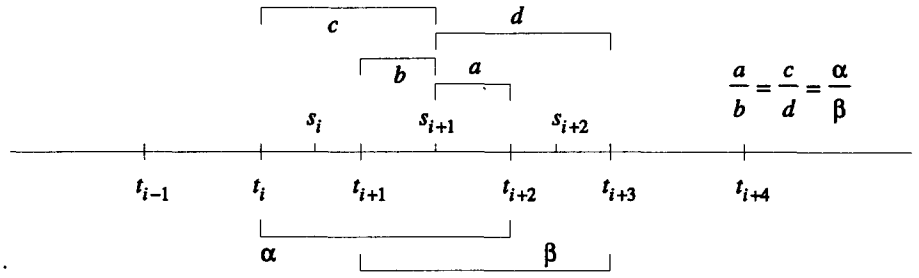

Figure 3. Relationship between t_i and s_i.

For given interpolation points $\mathbf{Q}_i$, $i = 0, 1, \ldots, n$, we will construct a quadratic B-spline curve $\mathbf{C}(t)$ with knots $t_{-2}, \ldots, t_{n+3}$ and control points $\mathbf{P}_0, \ldots, \mathbf{P}_{n+2}$ such that if s_i satisfy condition (3.2), then $\mathbf{C}(t)$ interpolates $\mathbf{Q}_i$ at s_i for i from 0 to n.

$$\mathbf{C}(s_i) = \mathbf{Q}_i, \qquad i = 0, 1, \ldots, n. \tag{3.3}$$

Since the interpolation is to be performed at s_i, the relationship between the interpolation points can be used to define s_i as follows:

$$s_0 = 0, \qquad s_n = 1,$$
$$s_i - s_{i-1} = \frac{|\mathbf{Q}_i - \mathbf{Q}_{i-1}|^{1/2}}{\sum_{j=1}^{n} |\mathbf{Q}_j - \mathbf{Q}_{j-1}|^{1/2}}, \qquad 1 \le i \le n. \tag{3.4}$$

Equation (3.4) follows the centripetal model defined by Lee [8].

To define the knots t_i for $i = -2, \ldots, n+3$, note that s_i satisfies (3.2) for i from 0 to n. On the other hand, one has two choices for setting up (t_{-2}, t_{-1}, t_0) and $(t_{n+1}, t_{n+2}, t_{n+3})$:

$$t_{-2} = t_{-1} = t_0, \qquad t_{n+1} = t_{n+2} = t_{n+3}, \tag{3.5}$$

or

$$t_{-1} - t_{-2} = t_0 - t_{-1} = \Delta s_0, \qquad t_{n+3} - t_{n+2} = t_{n+2} - t_{n+1} = \Delta s_{n-1}. \tag{3.6}$$

In either case, this yields a system of $n+1$ equations with $n+2$ unknowns $t_0, t_1, \ldots, t_{n+1}$. If boundary condition (3.5) is used, this yields

$$\begin{aligned} t_2 t_1 - s_0 t_2 - s_0 t_1 + 2 s_0 t_0 - t_0^2 &= 0, \\ t_{i+2} t_{i+1} - t_i t_{i-1} - s_i t_{i+2} - s_i t_{i+1} + s_i t_i + s_i t_{i-1} &= 0, \qquad i = 1, \ldots, n-1, \\ t_{n+1}^2 - t_n t_{n-1} - 2 s_n t_{n+1} + s_n t_n + s_n t_{n-1} &= 0. \end{aligned} \tag{3.7}$$

If boundary condition (3.6) is used, this yields a slightly different system of $n+1$ equations:

$$\begin{aligned} t_2 t_1 - s_0 t_2 - s_0 t_1 + 2 s_0 t_0 - t_0^2 - s_0 \Delta s_0 + t_0 \Delta s_0 &= 0, \\ t_{i+2} t_{i+1} - t_i t_{i-1} - s_i t_{i+2} - s_i t_{i+1} + s_i t_i + s_i t_{i-1} &= 0, \qquad i = 1, \ldots, n-1, \\ t_{n+1}^2 - t_n t_{n-1} + (\Delta s_{n-1} - 2 s_n)\, t_{n+1} + s_n t_n + s_n t_{n-1} - s_n \Delta s_{n-1} &= 0. \end{aligned} \tag{3.8}$$

These systems cannot be solved using ordinary approaches even with a given initial value for t_0 or t_{n+1}. One needs to use a minimization technique to solve for $t_0, t_1, \ldots, t_{n+1}$ in this case.

Let f_i be a function of $t_0, t_1, \ldots, t_{n+1}$ for i from 0 to n. f_i is defined as the left side of the corresponding equation in, say (3.7). Namely,

$$\begin{aligned} f_0 &\equiv t_2 t_1 - s_0 t_2 - s_0 t_1 + 2 s_0 t_0 - t_0^2, \\ f_i &\equiv t_{i+2} t_{i+1} - t_i t_{i-1} - s_i t_{i+2} - s_i t_{i+1} + s_i t_i + s_i t_{i-1}, \qquad i = 1, \ldots, n-1, \\ f_n &\equiv t_{n+1}^2 - t_n t_{n-1} - 2 s_n t_{n+1} + s_n t_n + s_n t_{n-1}. \end{aligned}$$

Define F as the sum of the squared values of f_i,

$$F \equiv \sum_{i=0}^{n} f_i^2. \tag{3.9}$$

F is a nonnegative function of $t_0, t_1, \ldots, t_{n+1}$. The value of t_i for i between 1 and n is bounded above by s_i and below by s_{i-1}. The value of t_0 is bounded above by $s_0 = 0$ and the value of t_{n+1} is bounded below by $s_n = 1$. Note that a solution to equation (3.7) or (3.8) satisfies the equation $F = 0$ and vice versa (depending on the boundary condition chosen). However, since equation (3.7) or (3.8) does not always have a solution, the minimum of F is not guaranteed to be zero. We will find a solution where the minimum of F occurs. In Section 4, we will show an example where there is no solution to (3.7) or (3.8).

The computation process for identifying the minimum point of F can be done by minimizing the objective function (3.9) subject to the following constraints:

$$\begin{aligned} &\alpha \le t_0 \le s_0, \\ &s_i \le t_{i+1} \le s_{i+1}, \qquad i = 0, \ldots, n-1, \\ &s_n \le t_{n+1} \le \beta, \end{aligned} \tag{3.10}$$

where $\alpha \le -1$ and $\beta \ge 2$. Actually, anything smaller than $-(s_1 s_2)^{1/2}$ may be used for α and anything bigger than $1 + ((1 - s_{n-1})(1 - s_{n-2}))^{1/2}$ may be used for β. (Note that from equation (3.7) and the assumption that $s_0 = 0$, we have $t_0^2 = t_1 t_2$. The value for α then follows

from the fact that $t_1 < s_1$ and $t_2 < s_2$. The value for β can be derived similarly.) This constrained minimization problem may be solved using the sequential quadratic programming (SQP) method [9–11]. The E04UCF subroutine of the *NAG Fortran Library* [12] is used here in the minimization process.

Once the values of t_{-2}, $t_{-1}, \ldots, t_{n+3}$ are available, then based on (3.3), one can set up the interpolation conditions as follows:

$$\begin{aligned}
\mathbf{P}_0 B_{-2,3}(s_0) + \mathbf{P}_1 B_{-1,3}(s_0) + \mathbf{P}_2 B_{0,3}(s_0) &= \mathbf{Q}_0, \\
\mathbf{P}_1 B_{-1,3}(s_1) + \mathbf{P}_2 B_{0,3}(s_1) + \mathbf{P}_3 B_{1,3}(s_1) &= \mathbf{Q}_1, \\
&\vdots \\
\mathbf{P}_{n-1} B_{n-3,3}(s_{n-1}) + \mathbf{P}_n B_{n-2,3}(s_{n-1}) + \mathbf{P}_{n+1} B_{n-1,3}(s_{n-1}) &= \mathbf{Q}_{n-1}, \\
\mathbf{P}_n B_{n-2,3}(s_n) + \mathbf{P}_{n+1} B_{n-1,3}(s_n) + \mathbf{P}_{n+2} B_{n,3}(s_n) &= \mathbf{Q}_n.
\end{aligned} \tag{3.11}$$

This is a system of $n+1$ equations in $n+3$ unknowns. One needs two extra conditions to solve this system. If the values of $\mathbf{C}'(t)$ at s_0 and s_n are known, say

$$\mathbf{C}'(s_0) = \mathbf{D}_0, \qquad \mathbf{C}'(s_n) = \mathbf{D}_n, \tag{3.12}$$

then $\mathbf{P}_0$ and $\mathbf{P}_{n+2}$ can be expressed as

$$\mathbf{P}_0 = \frac{\mathbf{D}_0 - \mathbf{P}_2 B'_{0,3}(s_0)}{B'_{-2,3}(s_0)}, \tag{3.13}$$

$$\mathbf{P}_{n+2} = \frac{\mathbf{D}_n - \mathbf{P}_n B'_{n-2,3}(s_n)}{B'_{n,3}(s_n)}. \tag{3.14}$$

Equations (3.13) and (3.14) follow from the fact that $B'_{-1,3}(s_0) = B'_{n-1,3}(s_n) = 0$. Hence, (3.11) can be expressed as

$$M \cdot P = Q, \tag{3.15}$$

where

$$P = (\mathbf{P}_1, \ldots, \mathbf{P}_{n+1})^{\mathsf{T}},$$

$$Q = \left(\mathbf{Q}_0 - \frac{B_{-2,3}(s_0)}{B'_{-2,3}(s_0)} \mathbf{D}_0, \mathbf{Q}_1, \ldots, \mathbf{Q}_{n-1}, \mathbf{Q}_n - \frac{B_{n,3}(s_n)}{B'_{n,3}(s_n)} \mathbf{D}_n \right)^{\mathsf{T}},$$

and M is a *totally positive* [6], $(n+1) \times (n+1)$ tridiagonal matrix satisfying the Schoenberg-Whitney conditions [6]. M is of the following form:

$$M = \begin{bmatrix}
B_{-1,3}(s_0) & \Delta_2 & 0 & & \cdot & 0 \\
B_{-1,3}(s_1) & B_{0,3}(s_1) & B_{1,3}(s_1) & 0 & & \cdot \\
0 & \cdot & \cdot & 0 & & \cdot \\
\cdot & 0 & \cdot & \cdot & \cdot & 0 \\
\cdot & & 0 & B_{n-3,3}(s_{-1}) & B_{n-2,3}(s_{n-1}) & B_{n-1,3}(S_{n-1}) \\
0 & \cdot & \cdot & 0 & \Delta_n & B_{n-1,3}(s_n)
\end{bmatrix},$$

with

$$\Delta_2 = B_{0,3}(s_0) - \frac{B'_{0,3}(s_0)}{B'_{-2,3}(s_0)} B_{-2,3}(s_0),$$

$$\Delta_n = B_{n-2,3}(s_n) - \frac{B'_{n-2,3}(s_n)}{B'_{n,3}(s_n)} B_{n,3}(s_n).$$

Solving (3.15) for P yields the control points of a quadratic B-spline curve $\mathbf{C}(t)$ that interpolates $\mathbf{Q}_i$ at s_i for $i = 0, \ldots, n$. Finding the solution of (3.15) is a typical application of a linear time Gaussian elimination process [5] which will not be repeated here.

4. IMPLEMENTATION

The above quadratic B-spline curve interpolation technique has been implemented in C on the following platform: HPUX level 9.05 on a Hewlett Packard HP 735 machine using a SUN Sparc 20 machine as the display device. The E04UCF subroutine of the *NAG Fortran Library* [12] was used for the constrained optimization process. The boundary conditions (3.5) and (3.6) have both been used in the construction of the interpolation parameters (s_i) and the parameter knots (t_j).

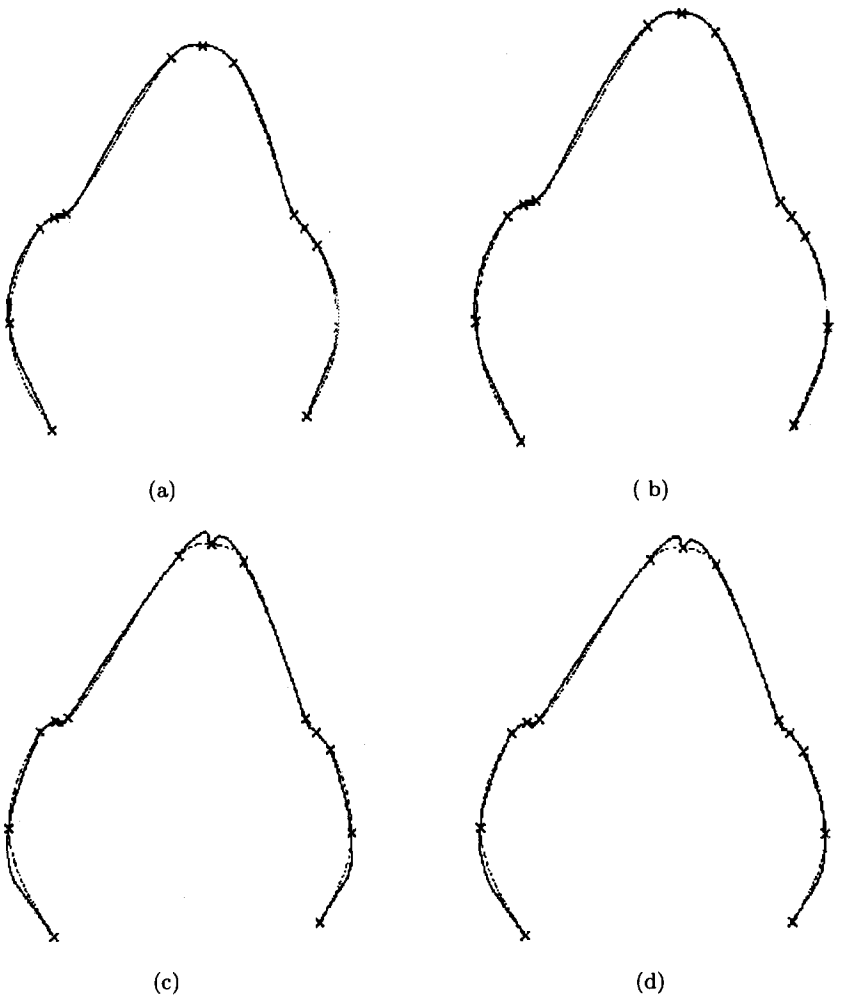

Figure 4. Quadratic B-spline curve interpolation: Case 1.

A number of test cases have been carried out. Four of them are included here (Figures 4–7). The fourth case is an example in which there is no solution to equation (3.2). In this case, $n = 5$, and the numbers in (3.5) have the values $s_0 = 0$, $s_1 = 1/6$, $s_2 = 1/3$, $s_3 = 2/3$, $s_4 = 5/6$, and $s_5 = 1$. There are six interpolation points $\mathbf{P}_0, \ldots, \mathbf{P}_5$, with $2|\mathbf{P}_0\mathbf{P}_1| = 2|\mathbf{P}_1\mathbf{P}_2| = |\mathbf{P}_2\mathbf{P}_3| = 2|\mathbf{P}_3\mathbf{P}_4| = 2|\mathbf{P}_4\mathbf{P}_5|$.

In each case, the proposed method and the midpoint method [7] are both tested. The proposed method with boundary condition (3.5) is shown in (a), the proposed method with boundary condition (3.6) is shown in (b), the midpoint method with boundary condition (3.5) is shown in (c), and the midpoint method with boundary condition (3.6) is shown in (d). These curves are shown in width 3. To compare the performance of these curves with odd-degree B-spline curves, a cubic B-spline curve that interpolates the same interpolation points is also generated for each test case. (We have implemented boundary conditions similar to (3.5) and (3.6) for the cubic

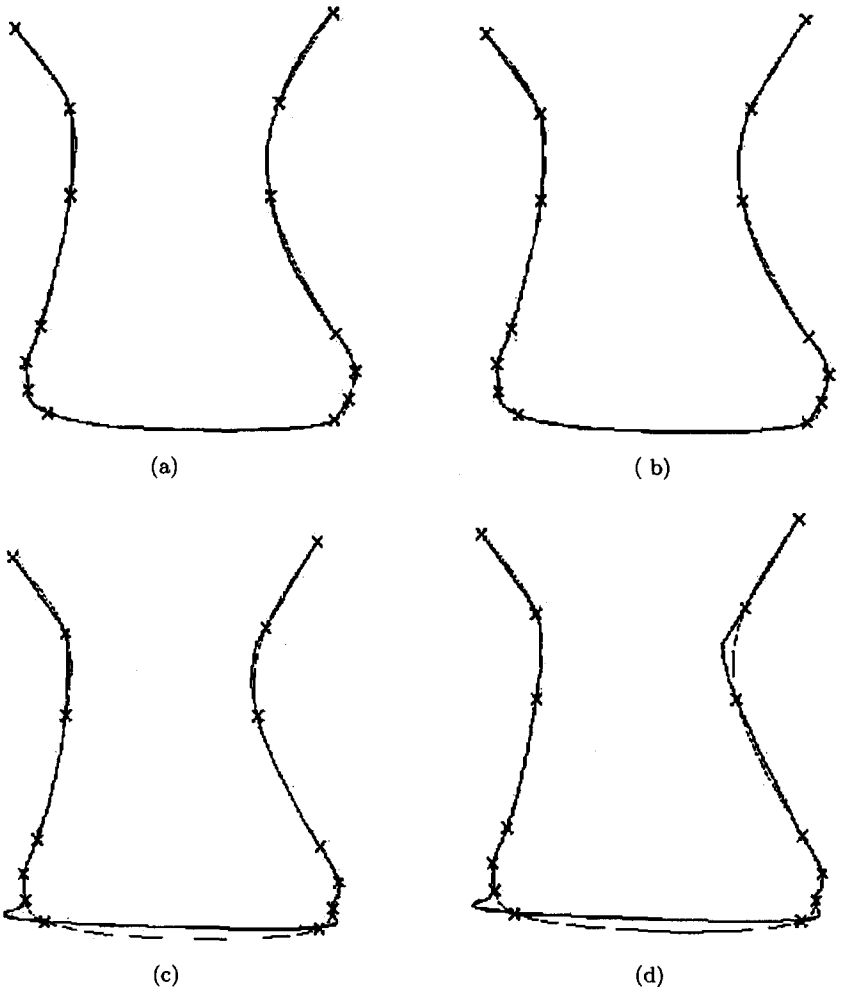

Figure 5. Quadratic B-spline curve interpolation: Case 2.

case as well. The resulting curves turn out to be exactly the same.) The cubic B-spline curve is shown in width 1.

For each case, the parameter values (s_i) where the interpolation is to be performed for the quadratic curves are computed using the centripetal model [8] first. The corresponding parameter knots (t_i) are then calculated using the E04UCF subroutine of the *NAG Fortran Library*. For the sake of space, these values are shown in Tables 1 and 2 for test case 1 only. In these tables (and subsequent tables as well), the term "maximum-1" refers to the proposed method with boundary condition (3.5), "maximum-2" refers to the proposed method with boundary condition (3.6), "midpoint-1" refers to the midpoint method with boundary condition (3.5), and "midpoint-2" refers to the midpoint method with boundary condition (3.6). The interpolation curves with appropriate boundary conditions are then computed and generated. Their strain energies and computation time are shown in Tables 3 and 4, respectively. The strain energy of a curve $\mathbf{C}(t)$ is defined as follows:

$$E(\mathbf{C}) = \int_{\mathbf{I}} \left(\frac{|\mathbf{C}' \times \mathbf{C}''|}{|\mathbf{C}'|^3} \right)^2 |\mathbf{C}'| \, dt.$$

It determines the smoothness of a curve: the smaller the energy, the smoother the curve.

It can be seen that in most of the cases, the quadratic B-spline curves generated by the proposed method are no worse than the corresponding cubic B-spline curves (Figures 4–7). The quadratic B-spline curves generated by the midpoint method tend to produce undesired oscillations (Figures 4–7). The midpoint method functions well only if consecutive interpolation

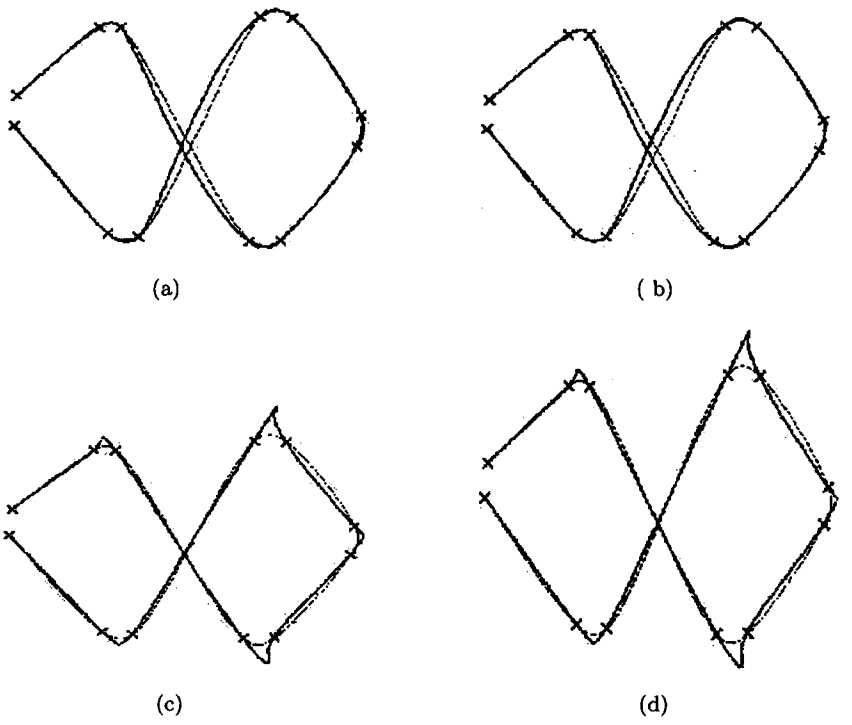

Figure 6. Quadratic B-spline curve interpolation: Case 3.

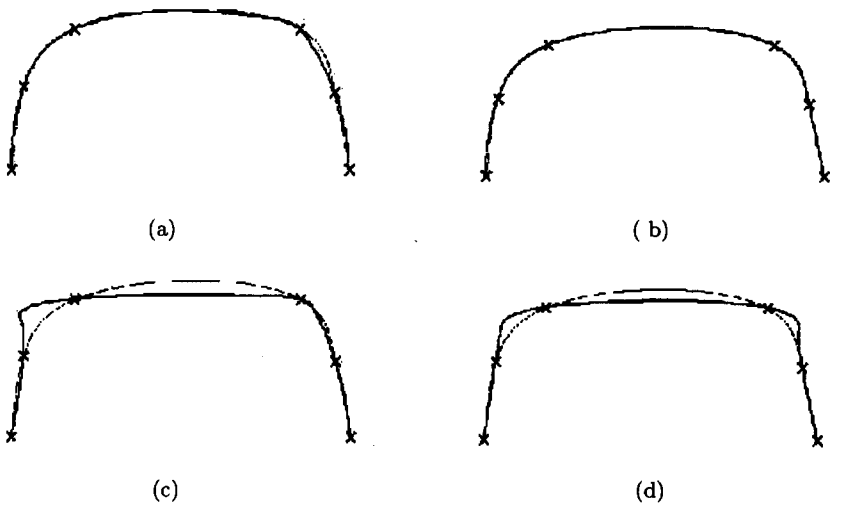

Figure 7. Quadratic B-spline curve interpolation: Case 4.

points are distributed rather uniformly, i.e., consecutive interpolation points are of approximately uniform distance and the curvature of the original curve does not change dramatically. This phenomenon is also justified by the strain energies of the resulting curves: the curves generated by the proposed method have smaller energies (hence, smoother shape) than the curves generated by the midpoint methods in most of the cases, and their energies are close to the energies of the corresponding cubic B-spline curves (Table 3).

The reason that the quadratic B-spline curves generated by the midpoint method tend to generate undesired oscillation can be explained as follows. Note that if consecutive interpolation points are not distributed uniformly, then the parameter values where the maxima of the quadratic B-spline basis functions occur would be quite different from the midpoints of the knots for some of the spans. For each of these spans, performing quadratic interpolation at the midpoint of the

Table 1. Parameter knots and interpolation parameters in Case 1a and 1c.

Parameter Knots	Interpolation Parameters for Maximum-1	Interpolation Parameters for Midpoint-1	Difference
$t_0 = -0.119939$	$s_0 = 0.000000$	$s_0 = -0.026513$	0.026513
$t_1 = 0.066914$	$s_1 = 0.112916$	$s_1 = 0.140876$	−0.027960
$t_2 = 0.214838$	$s_2 = 0.217241$	$s_2 = 0.216495$	0.003411
$t_3 = 0.218152$	$s_3 = 0.262778$	$s_3 = 0.259367$	0.000746
$t_4 = 0.300582$	$s_4 = 0.301353$	$s_4 = 0.322016$	−0.020663
$t_5 = 0.343451$	$s_5 = 0.447167$	$s_5 = 0.420367$	0.026800
$t_6 = 0.497283$	$s_6 = 0.509941$	$s_6 = 0.511944$	−0.002003
$t_7 = 0.526605$	$s_7 = 0.574426$	$s_7 = 0.598919$	−0.024493
$t_8 = 0.671234$	$s_8 = 0.709191$	$s_8 = 0.690649$	0.018542
$t_9 = 0.710064$	$s_9 = 0.752811$	$s_9 = 0.755716$	−0.002905
$t_{10} = 0.801368$	$s_{10} = 0.802359$	$s_{10} = 0.803397$	0.024861
$t_{11} = 0.805427$	$s_{11} = 0.898568$	$s_{11} = 0.873707$	0.024861
$t_{12} = 0.941988$	$s_{12} = 1.000000$	$s_{12} = 1.024154$	−0.024154
$t_{13} = 1.106319$			

Table 2. Parameter knots and interpolation parameters in Case 1b and 1d.

Parameter Knots	Interpolation Parameters for Maximum-2	Interpolation Parameters for Midpoint-2	Difference
$t_0 = -0.063362$	$s_0 = 0.000000$	$s_0 = -0.005728$	0.005728
$t_1 = 0.051907$	$s_1 = 0.112916$	$s_1 = 0.133531$	−0.020615
$t_2 = 0.215155$	$s_2 = 0.217241$	$s_2 = 0.216653$	0.000588
$t_3 = 0.218152$	$s_3 = 0.262778$	$s_3 = 0.259367$	0.003411
$t_4 = 0.300582$	$s_4 = 0.301353$	$s_4 = 0.322022$	−0.020669
$t_5 = 0.343461$	$s_5 = 0.447167$	$s_5 = 0.420347$	0.026820
$t_6 = 0.497233$	$s_6 = 0.509941$	$s_6 = 0.511922$	−0.001981
$t_7 = 0.526610$	$s_7 = 0.574426$	$s_7 = 0.598937$	−0.024511
$t_8 = 0.671264$	$s_8 = 0.709191$	$s_8 = 0.690664$	0.018527
$t_9 = 0.710064$	$s_9 = 0.752811$	$s_9 = 0.755716$	−0.002905
$t_{10} = 0.801368$	$s_10 = 0.802359$	$s_10 = 0.803142$	−0.000783
$t_{11} = 0.804916$	$s_11 = 0.898568$	$s_11 = 0.881024$	0.017544
$t_{12} = 0.957132$	$s_12 = 1.000000$	$s_12 = 1.005499$	−0.005499
$t_{13} = 1.053866$			

Table 3. Strain energies of the interpolation curves.

Case	Maximum-1	Maximum-2	Midpoint-1	Midpoint-2	Cubic
1 (Figure 4)	0.230800	0.238385	12.817297	12.646090	0.164627
2 (Figure 5)	1.314683	1.298631	2.353258	2.479480	0.167944
3 (Figure 6)	0.373523	0.357775	6.325497	6.729463	0.425358

Table 4. Computation time (in seconds) of the interpolation curves.

Case	Maximum-1	Maximum-2	Midpoint-1	Midpoint-2	Cubic
1 (Figure 4)	0.080000	0.070000	0.100000	0.070000	0.020000
2 (Figure 5)	0.110000	0.130000	0.090000	0.100000	0.020000
3 (Figure 6)	0.120000	0.070000	0.110000	0.080000	0.020000

span means using a small value of the dominating basis function to scale the curve at the midpoint while using a larger value of the dominating basis function to scale the curve in the neighborhood of the maximum point. This would force the curve to overshoot and, consequently, generate undesired oscillation. For instance, in Table 1, the difference of the s_i values for the proposed method and the midpoint method in spans 6 and 7 are quite large and overshooting occurs in

these two spans in Figure 4c. The same situation holds in Table 2 and Figure 4d. If consecutive interpolation points are distributed uniformly, since the s_i values for the proposed method are close to the midpoints of the knots for all the spans, performing quadratic interpolation at the midpoints will be about the same as performing quadratic interpolation at the maximum points. Hence, no overshooting would be produced.

The overall performance of cubic B-spline curves is about the same as quadratic B-spline curves generated by the proposed method. Actually, in some cases (Figures 4 and 5), the quadratic curves generated by the proposed method are better than the cubic B-spline curves. Obviously, the time to compute the knots and control points for a cubic curve is smaller than the time for a quadratic curve generated by the proposed method. However, the difference in absolute value is not significant at all (Table 4).

5. CONCLUSIONS

A new quadratic B-spline curve interpolation technique is presented. Given a set of interpolation points, this approach constructs the parameter values (s_i) where the interpolation will be performed first. These parameter values are then used in a constrained optimization process to construct a set of parameter knots (t_j). The relationship between (s_i) and (t_j) is this: (s_i) are the parameter values where the quadratic B-spline basis functions defined with respect to the knot sequence (t_i) reach their maxima. The control points of the quadratic interpolating B-spline curve are then computed by solving a totally positive, tridiagonal system of equations.

According to the test results, the new quadratic B-spline curve interpolation would not only avoid oscillations which often occurred with the midpoint approach [4], but also yield quadratic interpolation curves that are no worse than the cubic B-spline curve interpolating the same points. This shows that the long time assertion that "even-degree B-spline curve/surfaces are not suitable for the curve/surface interpolation problem" is a misconception, since the problem is actually with the way the interpolation process is performed, not with even-degree B-spline curves or surfaces themselves. Furthermore, due to the fact that quadratic curves are cheaper for rendering, subdivision, and intersection computation, the new approach actually makes quadratic B-spline curves better choices than cubic B-spline curves in some applications in graphics and geometric modeling.

Extension of this approach to biquadratic B-spline surface interpolation is straightforward: simply apply the proposed technique on both parameters to find the maxima of the corresponding basis functions. This approach can be used for other even-degree B-spline curves as well. The key point is to find the parameter values where the corresponding B-spline basis functions' maxima occur. However, no efficient way to compute these parameter values for higher degree even-degree B-spline basis functions has been found yet.

REFERENCES

1. J.H. Ahlberg, E.N. Nilson and J.L. Walsh, *The Theory of Splines and Their Applications*, Academic Press, New York, (1967).
2. A. Goshtasby, F. Cheng and B.A. Barsky, B-spline curves and surfaces viewed as digital filters, *Computer Vision, Graphics, and Image Processing* **52** (2), 264–275 (1990).
3. C. Rabut, Even degree B-spline curves and surfaces, A note on 'B-spline curves and surfaces viewed as digital filters' by A. Goshtasby, F. Cheng and B.A. Barsky, *CVGIP: Graphical Models and Image Processing* **54** (4), 351–356 (July 1992).
4. I.J. Schoenberg, *Cardinal Spline Interpolation*, SIAM, Philadelphia, PA, (1973).
5. R.H. Bartels, J.C. Beatty and B.A. Barsky, *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*, Morgan Kaufmann, San Francisco, CA, (1987).
6. C. de Boor, *Practical Guide to Splines*, Springer-Verlag, New York, (1978).
7. M.J. Marsden, Quadratic spline interpolation, *Bulletin of American Mathematical Society* **80**, 903–906 (1974).
8. E.T.Y. Lee, On choosing nodes in parametric curve interpolation, *CAD* **21** (6), 363–370 (July/August 1989).
9. R. Fletcher, *Practical Methods of Optimization*, John Wiley and Sons, Chichester, (1987).

10. P.E. Gill, W. Murray and M.H. Wright, *Practical Optimization*, Academic Press, New York, (1981).
11. M.J.D. Powell, Variable metric methods for constrained optimization, In *Mathematical Programming: The State of the Art*, (Edited by A. Bachem, M. Grotschel and B. Korte), pp. 288–311, Springer-Verlag, Berlin, (1983).
12. *NAG FORTRAN Library Manual, Mark 16*, First Edition, (September 1993).