Vol. 36 No. 12 Transactions of Information Processing Society of Japan Dec. 1995

Regular Paper

An Interactive Method for Designing Smooth Convex Curves by Using a Cubic B-spline Formulation

HUI GUAN† and TATSUO TORII†

This paper addresses the problem of designing smooth convex curves in an interactive graphics environment. Our main contribution to the solution of this problem is an easy-to-use and efficient design method using a cubic B-spline formulation, in the case of given first derivatives at end-points. This method provides users with an intuitive, practical way of handling convex curves, based on a set of simple equations and a recursive subdivision manner. Because of the ease of computation and the simplicity of subdivision, this method has several obvious advantages such as intuitiveness, rapidity, and convenience.

1. Introduction

Interactive design of smooth curves with arbitrary shapes is useful in many application areas, such as computer graphics, computer-aided geometric design, and cartography. A natural approach is to decompose complex curve structures into several convex parts and then approximate each part smoothly. Decomposing the curve is generally the easier task. Thus the major challenge in this approach is how to efficiently produce convex approximation curves on the assumption that some level of continuity is maintained at all the joints of the curves.

Polynomial B-splines are usually used as approximation curves in interactive curve design, because of the simplicity of their representation and their ease of manipulation. Such a curve is generated from a control polygon defined by a set of control points, and is typically obtained in two ways: as the result of interactive manipulation of the control polygon, or as a curve that is interpolated at given points. Theoretically, by interactively changing a set of control points, it is possible to obtain an acceptable approximation curve of the intended shape after several iterations. In practice, however, such manipulations cannot produce a curve of good quality[1]. Experience shows that it is difficult to tell from the display of a control polygon on the screen whether the shape of the generated curve is acceptable or not, especially for designers without experience in the design of B-spline curves[2]. The reason for this is that the interactive tweaking of control polygons does not provide direct control over geometric properties such as position, tangency, and curvature at an arbitrary point on the curve.

An alternative approach to solving this problem is to interpolate some points on the intended curve. Since these points give more precise information about the behavior of the curve, designers can easily modify them at the terminal to generate an acceptable approximation curve. For this reason, developing effective algorithms for the approximation of such points has become an important research direction. Various methods in this area have been proposed[3)~6),10]. The most popular of them is the cubic-spline interpolation technique, in which the resulting curve passes through each given point and its continuity is everywhere C^2. However, the method suffers from two major drawbacks: the first is that it may develop unwanted wiggles or undulations[6], and the second is that it requires a time-consuming calculation to determine the control polygon of a B-spline curve because this involves solving a large system of equations. Therefore, there is a need to both guarantee the curve convexity and speed up the control polygon computation, especially in an interactive graphics environment. This is the motivation for our work.

In this paper, we present a simple yet efficient approach for B-spline-based geometric design of smooth convex curves. Our approach adopts a subdivision strategy: an intended curve is repeatedly split until it can finally be approximated by a set of acceptable B-spline curve segments[7]. To ensure the continuity of adjacent curve segments, it uses first derivatives as end-conditions, since a curve with continuous unit tangent vectors shows satisfactory

† Department of Information Engineering, Faculty of Engineering, Nagoya University

visual smoothness for most interactive graphics applications[8]. Unlike existing interpolation techniques, the new method avoids solving a large system of equations to generate a B-spline approximation curve. The basic idea is to construct a convex curve, starting from a selected point and using information about the behavior of end-points, based on the convexity-preserving property of B-spline curves. Theoretical analysis indicates that all curves generated by the method have the following features: (1) geometric continuity, (2) convexity, and (3) interpolation of the given points on the original curve. In addition, preliminary experiments with the method, some of which are described later in this paper, show that it is both accurate enough and fast enough to be used for interactive curve design.

The rest of this paper is organized as follows. The next section defines a cubic B-spline curve and describes its properties. Section 3 introduces our strategy for approximating smooth convex curves and an algorithm for subdividing the intended curve. Section 4 provides several examples illustrating how the proposed method can be used to design smooth convex curves, and Section 5 contains some concluding remarks.

2. Cubic B-spline Curves

A cubic B-spline curve in 2D is defined as a vector-valued function that maps an interval into two-dimensional space. Let $\boldsymbol{Q}(t) = (x(t), y(t))$ denote the position vectors along the curve as a function of the parameter t. Mathematically, the curve is given by

$$\boldsymbol{Q}(t) = \sum_{i=0}^{n} \boldsymbol{B}_i N_{i,4}(t) \quad t_{min} \le t < t_{max} \quad (1)$$

where $\boldsymbol{B}_i$, $i = 0, \cdots, n$, represents the position vectors of the $n+1$ control points connected in a sequence to form a control polygon, and $N_{i,4}(t)$, $i = 0, \cdots, n$, represents the normalized B-spline basis functions, defined by the Cox-deBoor recursion formulas[9] as follows:

$$N_{i,1}(t) = \begin{cases} 1 & \text{if } u_i \le t < u_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$N_{i,k}(t) = \frac{t - u_i}{u_{i+k-1} - u_i} N_{i,k-1}(t) + \frac{u_{i+k} - t}{u_{i+k} - u_{i+1}} N_{i+1,k-1}(t)$$
$$k = 2, 3, 4$$

where the parameter t varies from t_{min} to t_{max}

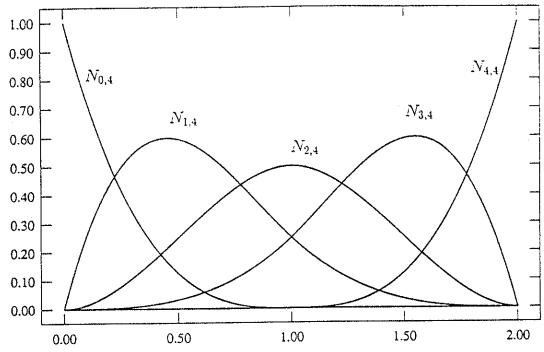

Fig. 1 B-spline basis functions.

to form a segment of the curve $\boldsymbol{Q}(t)$, and the values of u_i are elements of a knot vector

$$U = \{u_0, ..., u_i, u_{i+1}, ..., u_{n+k}\}$$

satisfying the relation $u_i \le u_{i+1}, i = 0, ..., n+k-1$.

The above basis functions have the following properties[9]:

(1) Each basis function is positive or zero for all parameter values, i.e., $N_{i,4}(t) \ge 0$,

(2) The sum of the basis functions for any parameter value t can be shown to be

$$\sum_{i=0}^{n} N_{i,4}(t) = 1$$

The curve exhibits two important geometric characteristics:

- The convex hull property: it lies entirely within the convex hull of its control points.
- The convexity-preserving property: it is convex when the control polygon is convex.

In order to make control polygons convex, we assume throughout this paper that a set of control points is $\{\boldsymbol{B}_i\}_{i=0}^{4}$, and that a knot vector takes the following form:

$$U = \{0, 0, 0, 0, 1, 2, 2, 2, 2\}$$

over which the basis functions $N_{i,4}(t), i = 0, 1, 2, 3, 4$ are defined. The values of $N_{i,4}(t)$, $i = 0, 1, 2, 3, 4$ are shown in **Fig. 1**, where $0 \le t < 2$. For convenience, we further let $N_{4,4}(2) = 1$ and $N_{i,4}(2) = 0, i = 0, 1, 2, 3$. Thus equation (1) reduces to

$$\boldsymbol{Q}(t) = \sum_{i=0}^{4} \boldsymbol{B}_i N_{i,4}(t) \qquad 0 \le t \le 2 \quad (2)$$

which is used as an approximation curve segment.

3. B-spline-based Curve Design

This section describes the details of our curve design scheme. Conceptually, the method con-

sists of two steps. The first is to generate a convex approximation curve using the cubic B-spline equation (2). The second is to subdivide the intended curve into curve segments for further approximation only when the shape of the generated curve is not acceptable. After a sufficient amount of recursive subdivision, we can finally obtain the intended curve composed of curve segments, each of which is a B-spline polynomial. The theoretical analysis at the end of this section will derive two important properties in order to characterize the graphical behavior of the generated approximation curve.

3.1 Approximation

Let us first discuss how to interactively approximate the original curve by using a convex B-spline curve segment, in the case of given first derivatives at end-points. The basic idea underlying our method is to let the point selected by the user on the curve be one of the control points of the approximation curve. This leads to a set of very simple equations for determining a convex control polygon.

Consider as an example the approximation of a curve with two end-points $\boldsymbol{P}_0$ and $\boldsymbol{P}_4$, as shown in **Fig. 2**, where $\boldsymbol{T}$ is the intersection of the two tangents at $\boldsymbol{P}_0$ and $\boldsymbol{P}_4$, and $\boldsymbol{M}$ is the midpoint of the line $\overline{\boldsymbol{P}_0\boldsymbol{P}_4}$. Denote by λ_0 and λ_4 the first derivatives at $\boldsymbol{P}_0$ and $\boldsymbol{P}_4$, respectively. Our goal is now to generate a convex curve segment that mimics the overall shape of the control polygon, but with $\boldsymbol{P}_0$ and $\boldsymbol{P}_4$ interpolated.

Assume that the point selected by the user, called the sample point, is the intersection $\boldsymbol{P}_2$ of the intended curve and the line $\overline{\boldsymbol{T}\boldsymbol{M}}$. We first determine the positions of the points $\boldsymbol{P}_0$, $\boldsymbol{P}_2$, and $\boldsymbol{P}_4$ by setting $t = 0$, 1, and 2 respectively, in equation (2). Then, according to the values of the basis functions $N_{i,4}(t), i = 0, \cdots, 4$ shown in Fig. 1, we obtain

$$\boldsymbol{P}_0 = \boldsymbol{Q}(0) = \boldsymbol{B}_0 \tag{3}$$

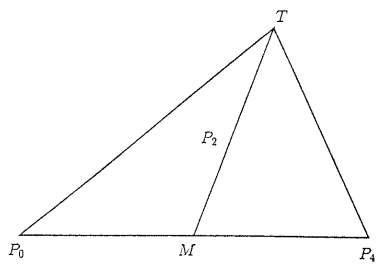

Fig. 2 Information on a segment for use in approximation.

$$\boldsymbol{P}_2 = \boldsymbol{Q}(1) = \frac{1}{4}\boldsymbol{B}_1 + \frac{1}{2}\boldsymbol{B}_2 + \frac{1}{4}\boldsymbol{B}_3 \tag{4}$$

$$\boldsymbol{P}_4 = \boldsymbol{Q}(2) = \boldsymbol{B}_4 \tag{5}$$

Furthermore, let $\boldsymbol{P}_2 = \boldsymbol{B}_2$; that is, let the sample point be the control point of the approximation curve. Equation (4) thus becomes

$$\boldsymbol{B}_1 + \boldsymbol{B}_3 = 2\boldsymbol{P}_2 \tag{6}$$

Now, it remains to discuss how to determine the unknown control points $\boldsymbol{B}_1$ and $\boldsymbol{B}_3$. Calculating the first derivatives of the basis functions at $t = 0$ and $t = 2$, we find that the tangent vectors at the two end-points are

$$d\boldsymbol{Q}/dt|_{t=0} = 3(\boldsymbol{B}_1 - \boldsymbol{P}_0) \tag{7}$$

$$d\boldsymbol{Q}/dt|_{t=2} = 3(\boldsymbol{P}_4 - \boldsymbol{B}_3) \tag{8}$$

Referring to the right-hand sides of the above two equations, we can then rewrite equation (6) in the following equivalent form:

$$(\boldsymbol{B}_1 - \boldsymbol{P}_0) - (\boldsymbol{P}_4 - \boldsymbol{B}_3) = 2\boldsymbol{P}_2 - \boldsymbol{P}_0 - \boldsymbol{P}_4 \tag{9}$$

For computational simplicity, we further set the unknown vectors $\boldsymbol{U}$ and $\boldsymbol{V}$ and the known vector $\boldsymbol{R}$ as

$$(\boldsymbol{B}_1 - \boldsymbol{P}_0) = \boldsymbol{U} = \begin{pmatrix} u_x \\ u_y \end{pmatrix}$$

$$(\boldsymbol{P}_4 - \boldsymbol{B}_3) = \boldsymbol{V} = \begin{pmatrix} v_x \\ v_y \end{pmatrix}$$

$$(2\boldsymbol{P}_2 - \boldsymbol{P}_0 - \boldsymbol{P}_4) = \boldsymbol{R} = \begin{pmatrix} r_x \\ r_y \end{pmatrix}$$

Thus, from equations (7), (8), and (9), we derive the following equation with respect to u_x, u_y, v_x, and v_y for the given finite λ_0 and $\lambda_4 (\neq \lambda_0)$:

$$\begin{cases} u_y/u_x = \lambda_0 \\ v_y/v_x = \lambda_4 \\ u_x - v_x = r_x \\ u_y - v_y = r_y \end{cases} \tag{10}$$

Solving for $\boldsymbol{U}$ and $\boldsymbol{V}$ yields

$$\begin{pmatrix} u_x \\ v_x \end{pmatrix} = \frac{1}{\lambda_0 - \lambda_4} \begin{pmatrix} -\lambda_4 & 1 \\ -\lambda_0 & 1 \end{pmatrix} \begin{pmatrix} r_x \\ r_y \end{pmatrix} \tag{11}$$

$$\begin{pmatrix} u_y \\ v_y \end{pmatrix} = \begin{pmatrix} \lambda_0 & 0 \\ 0 & \lambda_4 \end{pmatrix} \begin{pmatrix} u_x \\ v_x \end{pmatrix} \tag{12}$$

When the first derivative λ_0 or λ_4 is infinite, u_x or v_x becomes zero and, consequently, straightforward calculation gives the following results: $u_x = 0$, $u_y = r_y - \lambda_4 r_x$, $v_x = -r_x$, and $v_y = -\lambda_4 r_x$ for $|\lambda_0| = \infty$ and $|\lambda_4| \neq \infty$; $u_x = r_x$, $u_y = \lambda_0 r_x$, $v_x = 0$, and $v_y = \lambda_0 r_x - r_y$ for $|\lambda_0| \neq \infty$ and $|\lambda_4| = \infty$.

Thus, the control points $\boldsymbol{B}_1$ and $\boldsymbol{B}_3$ can be settled as

$$\boldsymbol{B}_1 = \boldsymbol{P}_0 + \boldsymbol{U} \tag{13}$$

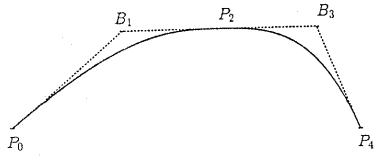

Fig. 3 A convex cubic B-spline curve segment.

$$\boldsymbol{B}_3 = \boldsymbol{P}_4 - \boldsymbol{V} \tag{14}$$

From equation (2), we finally obtain the corresponding approximation curve segment shown in **Fig. 3**, where the lines $\overline{\boldsymbol{B}_1\boldsymbol{B}_3}$ and $\overline{\boldsymbol{P}_0\boldsymbol{P}_4}$ are parallel, as mentioned below.

So far, we have dealt with the normal case, $\lambda_0 \neq \lambda_4$. When $\lambda_0 = \lambda_4$, that is, when the tangents at $\boldsymbol{P}_0$ and $\boldsymbol{P}_4$ are parallel, we also let $\overline{\boldsymbol{B}_1\boldsymbol{B}_3}$ be parallel to $\overline{\boldsymbol{P}_0\boldsymbol{P}_4}$. This gives

$$\boldsymbol{B}_1 = \boldsymbol{P}_0 + \frac{\boldsymbol{R}}{2} \tag{15}$$

$$\boldsymbol{B}_3 = \boldsymbol{P}_4 + \frac{\boldsymbol{R}}{2} \tag{16}$$

Theorem 1 The curve constructed under the given conditions has the following properties:

(1) Convexity and C^2 continuity
(2) Interpolation of the points $\boldsymbol{P}_0$, $\boldsymbol{P}_2$, and $\boldsymbol{P}_4$
(3) Zero curvature at the given sample point $\boldsymbol{P}_2$.

Proof: It is easy to see from equations (7) and (8) that $\boldsymbol{B}_1$ is on the line $\overline{\boldsymbol{P}_0\boldsymbol{T}}$, and $\boldsymbol{B}_3$ is on the line $\overline{\boldsymbol{T}\boldsymbol{P}_4}$ (see Figs. 2 and 3). Since $\boldsymbol{M}$ is the midpoint of the line $\overline{\boldsymbol{P}_0\boldsymbol{P}_4}$ and $\boldsymbol{P}_2$ is the midpoint of the line $\overline{\boldsymbol{B}_1\boldsymbol{B}_3}$ (from equation (6)), we know that $\overline{\boldsymbol{P}_0\boldsymbol{P}_4}$ is parallel to $\overline{\boldsymbol{B}_1\boldsymbol{B}_3}$. This shows that the control polygon $\boldsymbol{P}_0\boldsymbol{B}_1\boldsymbol{P}_2\boldsymbol{B}_3\boldsymbol{P}_4$ must be convex. The B-spline curve is therefore convex according to the convexity-preserving property of B-spline curves. Of course, it is also C^2 continuous, because the B-spline functions used here are cubic.

The second property is obvious from equations (3), (4), and (5). Now let us prove the third property. Use of the calculated second derivatives of the basis functions at $t = 1$ yields

$$d^2\boldsymbol{Q}/dt^2|_{t=1} = \frac{3}{2}\boldsymbol{B}_1 - 3\boldsymbol{P}_2 + \frac{3}{2}\boldsymbol{B}_3$$

Applying equation (6) to the above, we obtain

$$d^2\boldsymbol{Q}/dt^2|_{t=1} = 0$$

The curvature at the given sample point $\boldsymbol{P}_2$ is then zero. □

3.2 Subdivision

If the approximation curve generated in the previous section is acceptable, the control points are saved; otherwise, designers need to divide the intended curve into two curve segments. This section shows how to select a dividing point and determine its first derivative.

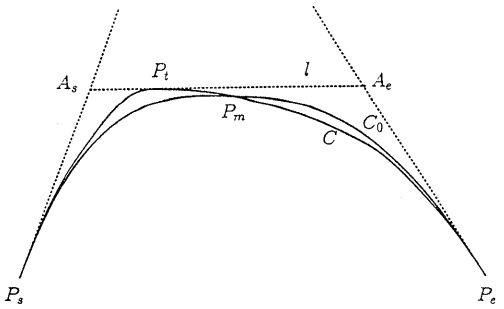

Fig. 4 Subdivision of a curve.

For illustrative purposes, consider the simple example shown in **Fig. 4**, where C is an ideal curve in the user's mind, and C_0 is the approximation curve on the screen, generated by our approximation method. Let $\boldsymbol{P}_t$ represent the highest point on the intended curve C, and let $\boldsymbol{P}_m$ represent the highest point on the approximation curve C_0, where the highest point on a curve is defined to be the point furthest from the line passing through the endpoints $\boldsymbol{P}_s$ and $\boldsymbol{P}_e$. It is easy to see that $\boldsymbol{P}_m$ is a considerable distance from $\boldsymbol{P}_t$, and that the approximation curve C_0 and the intended curve C have significant shape differences. To produce an acceptable approximation curve, we use two articulated B-spline curve segments to describe the curve C. Thus, the key problem we need to solve here is where to split C into two curve segments or how to select a dividing point on C. Clearly, the dividing point should not require alteration of the convex hull property of the entire approximation curve consisting of two articulated B-spline curve segments. For this reason, let us consider the highest point $\boldsymbol{P}_t$ on C. We know that this point should be also the highest point of the new approximation curve composed of left and right B-spline curve segments. Thus, it seems to be a natural choice for the subdivision of the intended curve. This gives rise to a simple curve subdivision process consisting of the following three steps:

(1) On the curve C, select the point $\boldsymbol{P}_t$ that is furthest from the line $\overline{\boldsymbol{P}_s\boldsymbol{P}_e}$.
(2) Draw a line l through $\boldsymbol{P}_t$ parallel to $\overline{\boldsymbol{P}_s\boldsymbol{P}_e}$.
(3) Split C into two subsegments at the di-

viding point P_t.

It is easy to see that the first derivative at the dividing point P_t is equal to the slope of the line $\overline{P_sP_e}$. By applying the method described in Section 3.1 to each of the subsegments, the corresponding C^2 convex B-spline curve segments can then be generated.

3.3 Curve Properties

After a sufficient amount of recursive subdivision and approximation, we will obtain a set of acceptable convex curve segments that can be pieced together to produce an entire curve. Next, we need to show that the curve is convex and smooth.

For convenience of presentation, in the following discussion, we use the term *joint* to refer to a dividing point, where two adjacent B-spline curve segments join.

Property 1. The entire curve is convex.

To prove this property, let us consider the curve C shown in Fig. 4, where A_s is the intersection of the line l and the tangent at P_s, and A_e is the intersection of the line l and the tangent at P_e. Since the distance from P_t to the line $\overline{P_sP_e}$ is maximum, the line l is the tangent at P_t. Thus, by the proposed approximation method, we can prove that the two convex B-spline curve segments connected by the dividing point P_t lie within the triangle $P_sA_sP_t$ and the triangle $P_tA_eP_e$, respectively. This shows that the convex property of the composed curve is not altered by our subdivision and approximation methods.

Property 2. The entire curve has visual smoothness.

For a curve to be visually smooth, it is necessary only that it is G^1. Theorem 1 indicates that each approximation curve segment generated by our method is C^2. It is thus sufficient to check the continuity at the joints. First, we can guarantee C^0 continuity at joints, since the dividing points are interpolated in the generated convex curve segments. Furthermore, we know from our subdivision algorithm that the first derivatives at dividing points are definite. Thus we can guarantee that the slope is continuous (geometric continuity G^1), because the tangent magnitudes on the left- and right-hand sides of a joint may be different. As a result, the curve has geometric continuity and hence is visually smooth.

4. Discussion

The method described in the previous section was implemented on a workstation in the C programming language. The new design tool provides more flexibility in controlling the shape of a convex curve segment, and is an independent system, because the modification of one curve segment does not influence others. In this section we show by means of three examples the features of the proposed method.

The first example compares the method with the cubic-spline interpolation technique. As mentioned previously, our method can guarantee the convexity of a B-spline approximation curve segment by constructing its convex control polygon, while the interpolation technique may not be able to generate such a convex curve segment. **Figure 5** illustrates this fact. Consider the three sample points and the first

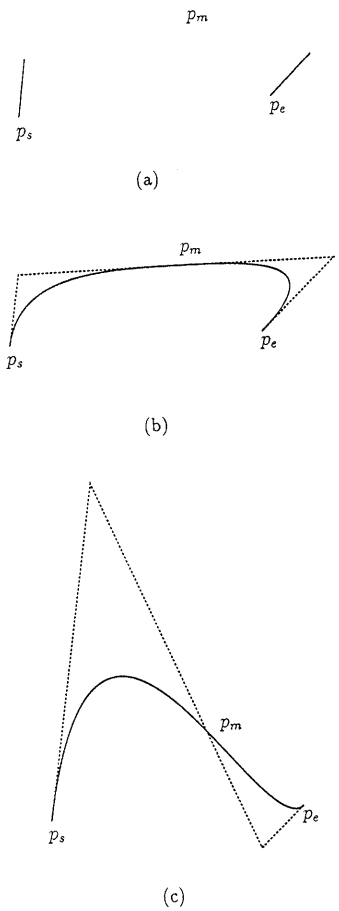

Fig. 5 An example of convexity preservation. (a) Given end-points, tangents at the end-points, and the point selected by the user. (b) Approximation curve using the proposed method. (c) Approximation curve using an interpolation method.

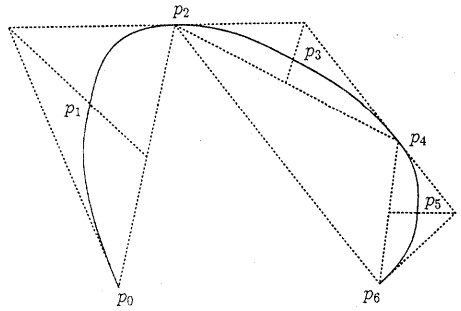

Fig. 6 An example of the algorithm's use in designing a convex curve.

derivatives at the end-points given in Fig. 5 (a). By applying our method to it, we can determine the convex control polygon (dotted lines) shown in Fig. 5 (b) and consequently, the resulting curve segment is also convex. In cubic-spline interpolation[9], however, we see that although it would be desirable to obtain a convex curve segment, the control polygon computation determines the control polygon (dotted lines) of Fig. 5 (c), which results in an undesirable deflection.

The second example illustrates how to design a smooth convex curve by the proposed method. Let us consider the construction of the curve segment shown in **Fig. 6**, where $\boldsymbol{p}_0$ and $\boldsymbol{p}_6$ are the end-points of the intended curve segment. First, we obtain a C^2 convex B-spline curve by our approximation computation. Assume that the shape of the curve is not acceptable and that the original curve thus needs to be divided into two subsegments. For this purpose, we plot a point $\boldsymbol{p}_2$ as a dividing point at the terminal and then calculate the first derivative at that point by drawing a line through it parallel to the line $\overline{p_0 p_6}$. As a result of the subdivision and approximation, convex B-spline curve segments are generated for the corresponding subsegments. In this example, the B-spline curve segment $\boldsymbol{p}_0\boldsymbol{p}_1\boldsymbol{p}_2$ is assumed to be acceptable, and hence the subsegment of the intended curve does not need to be further divided. Now, we turn to another subsegment with the end-points $\boldsymbol{p}_2$ and $\boldsymbol{p}_6$, where the approximation curve segment is not good. By specifying a new dividing point $\boldsymbol{p}_4$ and computing its first derivative, we make a further subdivision and then produce the B-spline curve segments $\boldsymbol{p}_2\boldsymbol{p}_3\boldsymbol{p}_4$ and $\boldsymbol{p}_4\boldsymbol{p}_5\boldsymbol{p}_6$. Let the two curve segments be very close to the intended shape. As a result, we obtain the acceptable G^1 convex curve shown in Fig. 6, which is composed of three cubic B-spline curve segments.

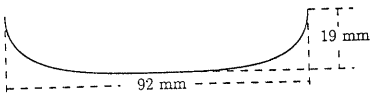

Fig. 7 An engineering drawing.

The above examples show the ability of the proposed method to produce smooth convex curves. It is easy to see from them that the shapes of B-spline approximation curves obtained by our method may be not very aesthetic, since the curvatures at the user-selected points are equal to zero. However, this is not always a drawback. It can be very useful in cases where the intended shape is required to be flat somewhere. The engineering drawing shown in **Fig. 7** can serve as such an example, where the shape in the neighborhood of the user-selected point is required to be very flat. We believe that the new method has practical value for the design of engineering drawings and curved line drawings, and is a good starting place for creating other curve-drawing algorithms.

5. Concluding Remarks

This paper has presented an interactive method for designing smooth convex curves, based on a set of simple equations and a recursive subdivision manner. The method provides a new, powerful means of interactively manipulating convex curves without solving a large system of equations. Our preliminary experiments showed that it is sufficiently accurate and fast to be used for designing smooth convex curves in interactive environments. As a natural extension of the method, we are studying its applicability to the manipulation of some feature points on a curve.

Acknowledgments We are grateful to the referees for their careful reading and valuable comments.

References

1) Farin, G. and Sapidis, N.: Curvature and the Fairness of Curves and Surfaces, *IEEE Comput. Gr. Appl.*, pp.52–57 (Mar. 1989).
2) Yamashita, J. and Fukui, Y.: DDM: A Direct Deformation Method — Two-Dimensional DDM —, *IEICE Trans.*, Vol.J76-D-II, No.8, pp.1780–1787 (1993).
3) Goshtasby, A. , Cheng, F. and Barsky, B.A.: B-spline Curves and Surfaces Viewed as Digital Filters, *Comput. Gr. Image Process.*, Vol.52, pp.264–275 (1990).

4) Hatano, K. and Ninomiya, I.: Algorithms for the Computation of Interpolating Splines by Means of B-Splines, *IPS Japan*, Vol.19, No.6, pp.538–545 (1978).
5) Kjellander, J.A.P.: Smoothing of Cubic Parametric Splines, *Comput. Aided Des.*, Vol.15, No.3, pp.175–179 (1983).
6) Barsky, B.A. and DeRose, T.D.: Geometric Continuity of Parametric Curves: Constructions of Geometrically Continuous Splines, *IEEE Comput. Gr. Appl.*, pp.60–68 (Jan. 1990).
7) Guan, H. and Torii, T.: Approximation of a Smooth Convex Curve by Cubic B-Spline Preserving the Convexity, *Proc. Symposium on Applied Mathematics*, Ryukoku Univ., pp.48.1–48.4 (1994).
8) Piegl, L.: Interactive Data Interpolation by Rational Bezier Curves, *IEEE Comput. Gr. Appl.*, pp.45–59 (Apr. 1987).
9) Rogers, D.F. and Adams, J.A.: *Mathematical Elements for Computer Graphics*, pp.305–351, International Edition, Singapore (1990).
10) Su, B.Q. and Liu, D.Y.: *Computational Geometry: Curve and Surface Modeling*, pp.9–40, Academic Press, Inc., San Diego (1989).

(Received May 11, 1995)
(Accepted October 5, 1995)

Hui Guan is a doctoral candidate in the Department of Information Engineering at Nagoya University, Japan. Her research interests include computer graphics and computer-aided geometric design. Guan received the B.S. degree in Computer Science from Beijing University of Aeronautics and Astronautics, Beijing, China, in 1987, the M.S. degree in Information Engineering from Fukui University, Fukui, Japan, in 1993. She is a member of the IPS Japan.

Tatsuo Torii is a professor of Department of Information Engineering, School of Engineering, Nagoya University. He has been engaged in research about numerical analysis and mathematical software. His special subjects are function approximation and numerical integration based on the FFT technique. He was born in Kumamoto prefecture in 1934. He graduated from Department of Electricity, Kyushu Institute of Technology in 1957. After his 7 years career being with Shin Nihon Chisso Corp., he joined Department of Applied Physics, Faculty of Engineering, Osaka University as a research assistant in 1964. He received D.E. from Osaka University in 1972. In 1975, he joined Department of Information Engineering, School of Engineering, Nagoya University as an assistant professor. He was promoted to an associate professor in 1976, and to a professor in 1985. He is a member of the IPS Japan.