

# Boundary evaluation for interval Bézier curve

Hongwei Lin<sup>a,b</sup>, Ligang Liu<sup>a,b</sup>, Guojin Wang<sup>a,b,\*</sup>

<sup>a</sup>*Institute of Computer Images and Graphics, Zhejiang University, Hangzhou, 310027, China*

<sup>b</sup>*Department of Mathematics, Zhejiang University, Hangzhou, 310027, China*

Received 18 December 2000; revised 9 April 2001; accepted 1 May 2001

---

## Abstract

The objective of this paper is to provide an efficient and reliable algorithm for representing and evaluating the boundary of the interval Bézier curve in 2- and 3-D. The boundary of the planar Bézier curve is represented by a sequence of Bézier curve segments with same degree and line segments in the order they are encountered when marching counter-clockwise along its boundary. The boundary can also be represented as a single B-spline curve having the same degree with the interval Bézier curve. The boundary of the 3-D interval Bézier curve is made up of trimmed Bézier surface patches and rectangular patches. Some examples illustrate our algorithms. © 2002 Elsevier Science Ltd. All rights reserved.

*Keywords:* Interval Bézier curves; Sweeping; Boundary evaluation

---

## 1. Introduction

A fundamental problem in CAD/CAM and computer graphics is how to represent and approximate curves and surfaces. A variety of approaches based on implicit, explicit and parametric representations have been proposed. Suppose an input curve already has a significant geometric uncertainty with an ill-posed configuration, its approximation may result in more serious ambiguities propagated during the process of approximation. On the other hand, state-of-the-art CAD/CAM systems using floating point arithmetic frequently fail in creating and interrogating curves and surfaces. The ultimate reason for this failure is the result of the practically limited precision that is inherent to the internal representation of floating-point number in geometric computation. One can keep in mind that any sequence of operations on a digital computer is essentially equivalent to a finite sequence of manipulations on a discrete grid of points.

To cope with these adversities, Sederberg and Farouki [1] introduced a new representation form of parametric curves, the interval Bézier curves, which is computed in interval arithmetic [2]. Interval arithmetic leads to numerical robustness and provides results with numerical certainty and verifiability. Mudur et al. [3] and Snyder [4] applied interval algorithms to a wide variety of problems in computer

graphics and geometry processing. Sherbrooke and Patrikalakis [5] used interval arithmetic in non-linear polynomial system solvers. Hu et al. [6,7] presented robust algorithms for curve and surface intersections by rounded interval arithmetic. Interval arithmetic is also used in solid modeling [8,9] and in robust visualization [10]. Recently, Abrams et al. [11] presented two methods for performing robust rounded interval arithmetic. The above series of works indicate that using rounded interval arithmetic will substantially provide results with verifiable numerical certainty in geometric computations, and thus enhance numerical robustness of current CAD/CAM systems.

Interval Bézier curves differ from classical Bézier curves in that the real numbers representing control point coordinates are replaced by intervals. Inspired by Sederberg and Farouki's work, such curves have been used in [12] for approximating offsets of parametric curves. Tuohy and Patrikalakis [13] used interval Bézier curves for the representation of functions with uncertainty. Interval Bézier curves were used for solving shape interrogation problems robustly [14,15]. Interval B-spline curves were used in [16] for the approximation of measured data. The numerical and geometric properties of interval B-spline curve in general are discussed in [17]. Chen and Lou [18] discussed the problem of bounding interval Bézier curves with lower degree interval Bézier curves.

In interval Bézier curves, the classical control points are replaced by 2-D rectangles or 3-D boxes. Consequently, an interval curve represents a region containing a family of

---

\* Corresponding author.

E-mail address: wgj@math.zju.edu.cn (G. Wang).

curves. This implies that, in 2-D plane, an interval Bézier curve represents a thin stripe and in 3-D space, an interval Bézier curve represents a slender tube, if the intervals are chosen sufficiently small.

In fact, interval Bézier curves belong to the class of sweep objects. Sweeping a simple object along some trajectory is one of the fundamental operations in geometric and solid modeling. Sweeps are considered to be one of the basis representation schemes in [19] and have numerous applications in graphics, geometric modeling, mechanical design and manufacturing, and motion planning. Despite their usefulness, it is non-trivial to construct the boundary of a general sweep. The problem of determining the boundary of the swept area generated by a moving planar polygonal body is discussed variously. But computing boundary representation of the sweep is widely viewed as difficult and expensive. Several methods for generating the approximated boundary of the sweep are known [20–23].

The determination of the geometry of the interval Bézier curve is one of the essential steps in the application of interval Bézier curves such as boundary representation (B-rep) model by interval Bézier curves, intersection between interval Bézier curves, numerical control of the tolerance by interval Bézier curves, etc. In this paper we will investigate the explicit boundary structure for interval Bézier curves. Both the two-dimensional and three-dimensional version of the solution are explained in detail and the implementations are discussed in this paper. Specifically, in our paper we aspire to solve the following problem: given an interval Bézier curve, what is the boundary of the region generated by this interval Bézier curve?

In this paper we first review a short summary of interval arithmetic and interval Bézier curve in Section 2. The boundary of 2-D interval Bézier curve is analyzed and an efficient and reliable algorithm for representing and evaluating the boundary of the interval Bézier curve is presented in Section 3. Next, in Section 4, boundary evaluations for interval Bézier curves in 3-D are discussed. Finally, we conclude the paper in Section 5.

## 2. Preliminaries

A scalar interval  $[a, b]$  is a closed set of real values of the form [2]

$$[a, b] = \{x | a \leq x \leq b\} \tag{1}$$

where the following interval arithmetic operations are defined by

$$[a, b] + [c, d] = [a + c, b + d], \tag{2}$$

$$[a, b] - [c, d] = [a - d, b - c], \tag{3}$$

$$[a, b] \cdot [c, d] = [\min(ac, ad, bc, bd), \max(ac, ad, bc, bd)], \tag{4}$$

$$[a, b] / [c, d] = [a, b] \cdot [1/d, 1/c], (0 \notin [c, d]). \tag{5}$$

A planar Bézier curve on the parameter interval  $[0, 1]$  is defined as follows

$$\mathbf{P}(t) = \sum_{i=0}^n \mathbf{P}_i B_i^n(t), \tag{6}$$

where  $n$  is the degree of the curve,  $\mathbf{P}_i$  is the control point and

$$B_i^n(t) = \binom{n}{i} (1-t)^{n-i} t^i$$

is the  $i$ -th Bernstein basis function of degree  $n$ . Interval Bézier curves are Bézier curves with vector-valued interval control points. We here give the definition of a planar interval Bézier curve.

**Definition 1.** A planar degree  $n$  interval Bézier curve  $[\mathbf{P}](t)$  is defined by [1]:

$$[\mathbf{P}](t) = \sum_{i=0}^n [\mathbf{P}_i] B_i^n(t), \quad 0 \leq t \leq 1, \tag{7}$$

where  $B_i^n(t)$  is the  $i$ th Bernstein basis function of degree  $n$ , the interval control points  $[\mathbf{P}_i]$  ( $i = 0, 1, \dots, n$ ) are rectangles in the plane which can also be degenerate intervals with zero-width:

$$[\mathbf{P}_i] = ([a_i, b_i], [c_i, d_i]), \quad a_i \leq b_i, \quad c_i \leq d_i, \tag{8}$$

$i = 0, 1, \dots, n.$

Interval Bézier curves differ from classical Bézier curves in that the control points are replaced by rectangles. For any Bézier curve  $\mathbf{P}(t)$  whose control points satisfy  $\mathbf{P}_i \in [\mathbf{P}_i]$  for  $i = 0, 1, \dots, n$ , we have  $\mathbf{P}(t) \in [\mathbf{P}](t)$ ; in other words, the interval Bézier curve defines a region (a thin strip) in the plane which consists of all the Bézier curves whose control points satisfy  $\mathbf{P}_i \in [\mathbf{P}_i]$  for  $i = 0, 1, \dots, n$ . An example of planar cubic interval Bézier curve is illustrated as Fig. 1. Interval control points are shown by dark solid-filled rectangles, and the interval Bézier curve is shown by light solid-filled region.

After defining the interval Bézier curves, the problem at hand is to compute and represent the geometry of the

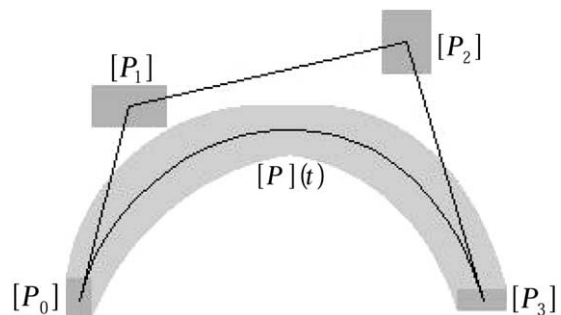


Fig. 1. A planar cubic interval Bézier curve.

interval Bézier curve, i.e. to determine a representation for the interval Bézier curve.

### 3. Boundary evaluation for 2-D interval Bézier curve

In this section we restrict our discussion to the planar interval Bézier curves, the extension to three dimension is straightforward in the next section.

The procedure we outline first identifies a set of curve segments and line segments that could lie on the boundary of the planar interval Bézier curve  $[P](t)$  based on some analysis, and later does a global trimming where the curve segments not on the boundary are discarded.

Without loss of generality the region of the interval Bézier curve lies in the right side when marching clockwise along its boundary.

#### 3.1. Curve and line segments on the boundary

Initially, in this section we will only consider the cases that there are no self-intersections for  $[P](t)$ . Further details of how to cope with the cases that  $[P](t)$  self intersects can be found in later section.

**Definition 2.** Denote the upper, lower, left, right edge of the interval point  $[P_i]$  by

$$\begin{aligned}
 [P_i^u] &= ([a_i, b_i], d_i), & [P_i^d] &= ([a_i, b_i], c_i), & [P_i^l] &= (a_i, [c_i, d_i]), \\
 [P_i^r] &= (b_i, [c_i, d_i]), & & & & 
 \end{aligned}
 \tag{9}$$

respectively. The four interval Bézier curves with degree  $n$  as

$$[P^x](t) = \sum_{i=0}^n [P_i^x] B_i^n(t), \quad x = u, d, l, r, \tag{10}$$

are respectively called upper, lower, left and right edge curves. The trajectories for the four corner vertices of the interval point  $[P_i]$

$$\begin{aligned}
 \mathbf{p}^{ul}(t) &= (\underline{x}(t), \bar{y}(t)), & \mathbf{p}^{ur}(t) &= (\bar{x}(t), \bar{y}(t)), \\
 \mathbf{p}^{dl}(t) &= (\underline{x}(t), \underline{y}(t)), & \mathbf{p}^{dr}(t) &= (\bar{x}(t), \underline{y}(t)),
 \end{aligned}
 \tag{11}$$

are called the upper-left, upper-right, lower-left, lower-right vertex curves respectively, where

$$\underline{x}(t) = \sum_{i=0}^n a_i B_i^n(t), \quad \bar{x}(t) = \sum_{i=0}^n b_i B_i^n(t), \tag{12}$$

$$\underline{y}(t) = \sum_{i=0}^n c_i B_i^n(t), \quad \bar{y}(t) = \sum_{i=0}^n d_i B_i^n(t).$$

It can be seen that it is sufficient to determine the boundaries of the four edge curves  $[P^u](t)$ ,  $[P^d](t)$ ,  $[P^l](t)$ , and  $[P^r](t)$  in order to determine the boundary of  $[P](t)$ .

**Definition 3.** A point  $\mathbf{P} = (x, y) \in [P](t)$  is called a boundary point, if every neighborhood of  $\mathbf{P}$  does not entirely lie in  $[P](t)$ . A boundary point  $\mathbf{P}$  is called

1. an upper boundary point, if for arbitrary  $\epsilon > 0$ ,  $(x, y + \epsilon) \notin [P](t)$ ;
2. a lower boundary point, if for arbitrary  $\epsilon > 0$ ,  $(x, y - \epsilon) \notin [P](t)$ ;
3. a left boundary point, if for arbitrary  $\epsilon > 0$ ,  $(x - \epsilon, y) \notin [P](t)$ ;
4. a right boundary point, if for arbitrary  $\epsilon > 0$ ,  $(x + \epsilon, y) \notin [P](t)$ .

The sets of the upper, lower, left, right boundary points of  $[P](t)$  are denoted by  $UpBn([P](t))$ ,  $LoBn([P](t))$ ,  $LeBn([P](t))$ ,  $RiBn([P](t))$ , respectively.

**Lemma 1.** For a planar interval Bézier curve  $[P](t)$  with degree  $n$ , we have

$$UpBn([P](t)) \subseteq UpBn([P^u](t)); \tag{13}$$

$$LoBn([P](t)) \subseteq LoBn([P^d](t)); \tag{14}$$

$$LeBn([P](t)) \subseteq LeBn([P^l](t)); \tag{15}$$

$$RiBn([P](t)) \subseteq RiBn([P^r](t)). \tag{16}$$

**Proof.** We only prove (13), the others are similar. Let

$$\mathbf{Q} = (x_0, y_0) \in UpBn([P](t)). \tag{17}$$

There exists a Bézier curve  $\mathbf{P}(t) = (x(t), y(t)) \in [P](t)$  passing through  $\mathbf{Q}$  at some parameter value  $t_0 \in [0, 1]$ , where

$$\begin{aligned}
 (x(t), y(t)) &= \left( \sum_{i=0}^n x_i B_i^n(t), \sum_{i=0}^n y_i B_i^n(t) \right), \quad (x_i, y_i) \in [P_i], \\
 i &= 0, 1, \dots, n.
 \end{aligned}
 \tag{18}$$

If  $y_{i_0} < d_{i_0}$  for some index  $i_0$ , then

$$\begin{aligned}
 \bar{y}_0 &= \sum_{i=0}^n y_i B_i^n(t_0) + d_{i_0} B_{i_0}^n(t_0) > y_0 \\
 i &\neq i_0
 \end{aligned}$$

It contradicts Eq. (17) by the definition of upper boundary point. Thus we have  $y_i = d_i, i = 0, 1, \dots, n$ , i.e.  $\mathbf{Q} \in [P^u](t)$ . Since  $\mathbf{Q}$  is an upper point of  $[P](t)$ , it is so an upper point of  $[P^u](t)$ , i.e.  $\mathbf{Q} \in UpBn([P^u](t))$ . This completes the proof of (13).

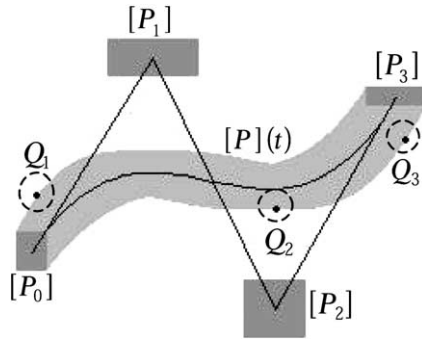


Fig. 2. Boundary points of a planar cubic interval Bézier curve.

**Remark 1.** It should be noted that some boundary point may be both the upper boundary point and the right boundary point, etc. For example, in Fig. 2,  $Q_2$  is only a lower boundary point, but  $Q_1$  is both the upper and left boundary point,  $Q_3$  is both the right and lower boundary point.

The geometric explanation for (13) is that the upper boundary of  $[P](t)$  is determined by the upper boundary of  $[P^u](t)$ . The others (14)–(16) are similar.

**Lemma 2.** For the left and right boundary of  $[P^u](t)$ , we have

$$LeBn([P^u](t)) \subseteq p^{ul}(t) \tag{19}$$

and

$$RiBn([P^u](t)) \subseteq p^{ur}(t). \tag{20}$$

*Proof.* Let

$$Q = (x_0, y_0) = \left( \sum_{i=0}^n x_i B_i^n(t_0), \sum_{i=0}^n d_i B_i^n(t_0) \right) \in LeBn([P^u](t)).$$

It can be seen that  $x_i = a_i$  ( $i = 0, 1, \dots, n$ ) by the definition of the left boundary. Thus  $Q \in p^{ul}(t)$ , which concludes  $LeBn([P^u](t)) \subseteq p^{ul}(t)$ . The proof of (20) is similar (Fig. 3).

**Definition 4.**

1. Let  $p^1(t) = (x(t), y^1(t))$  and  $p^2(t) = (x(t), y^2(t))$  be two degree  $n$  Bézier curves with the same abscissa. If  $x(t)$  attains a local extremum  $x_0$  at some  $t = t_0$ , then  $p^1(t)$  and  $p^2(t)$  both reach local extrema in the horizontal( $x$ -coordinate) direction. The two points  $(x(t_0), y^1(t_0))$  and  $(x(t_0), y^2(t_0))$  are called the corresponding local  $x$ -extremum points.
2. If  $p^1(t) = (x^1(t), y(t))$  and  $p^2(t) = (x^2(t), y(t))$  are two degree  $n$  Bézier curves with the same ordinate. If  $y(t)$  attains a local extremum  $y_0$  at some  $t = t_0$ , then  $p^1(t)$  and  $p^2(t)$  both reach local extrema in the vertical( $y$ -coordinate) direction.

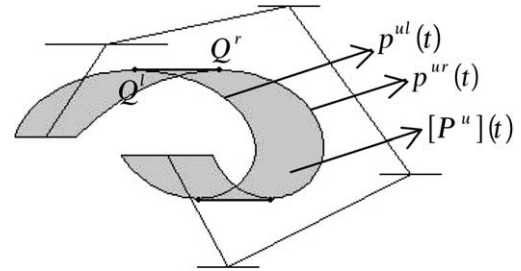


Fig. 3. The boundary structure of  $[P^u](t)$ .

direction. The two points  $(x^1(t_0), y(t_0))$  and  $(x^2(t_0), y(t_0))$  are called the corresponding local  $y$ -extremum points.

**Lemma 3.** Assume that  $[P^u](t)$  does not self intersect. The line segments connecting the corresponding local  $y$ -extremum(maximum/minimum) points of  $p^{ul}(t)$  and  $p^{ur}(t)$  are parallel to the  $x$ -axis. They are parts of the upper/lower boundary of  $[P^u](t)$ .

*Proof.* Suppose that  $p^{ul}(t)$  and  $p^{ur}(t)$  reach their local  $y$ -maximum points  $Q^l$  and  $Q^r$  at  $t = t_0$  respectively (Fig. 3). The line segment between  $Q^l$  and  $Q^r$  is parallel to  $x$ -axis as  $p^{ul}(t)$  and  $p^{ur}(t)$  have the same  $y$ -coordinates.

Let  $Q = (x^*, y^*) \in Line(Q^l, Q^r)$ ,  $y^* = \bar{y}(t_0)$ ,  $\underline{x}(t_0) \leq x^* \leq \bar{x}(t_0)$ . Since the function

$$f(x_0, x_1, \dots, x_n) = \sum_{i=0}^n x_i B_i^n(t_0)$$

is continuous with respect to  $x_0, x_1, \dots, x_n$  and  $f(a_0, a_1, \dots, a_n) = \underline{x}(t_0)$ ,  $f(b_0, b_1, \dots, b_n) = \bar{x}(t_0)$ , so there must exist  $(x_0^*, x_1^*, \dots, x_n^*) \in [a_0, b_0] \otimes [a_1, b_1] \otimes \dots \otimes [a_n, b_n]$  satisfying  $f(x_0^*, x_1^*, \dots, x_n^*) = x^*$ . That is, there is a Bézier curve

$$p^*(t) = \left( \sum_{i=0}^n x_i^* B_i^n(t), \bar{y}(t) \right)$$

passing through  $Q$ . As  $\bar{y}(t)$  reaches its local maximum at  $t_0$ , so  $p^*(t)$  reaches its local  $y$ -maximum at  $Q$ . Therefore the line segment  $Line(Q^l, Q^r)$  is the upper boundary of  $[P^u](t)$ .

The proof for the case that  $Q^l$  and  $Q^r$  are local minimum is similar.

**Remark 2.** It should be noted that the line segment  $Line(Q^l, Q^r)$  is in fact the envelope of the family of Bézier curves in  $[P^u](t)$ . And at time  $t = t_0$ , the corresponding member of the family of curves will just touch  $Line(Q^l, Q^r)$  in some point. This may also be expressed by saying that the line segment  $Line(Q^l, Q^r)$  is tangential to all members of the family of curves.

**Remark 3.** It can be shown from the proofs of Lemmas 2 and 3 that the boundary of  $[P^u](t)$  is made up of only two

types of segments, as shown in Fig. 3. First, the line segments connecting the corresponding local maximum/minimum points are the upper/lower boundary of  $[\mathbf{P}^u](t)$ . Secondly, the curve segments of  $\mathbf{p}^{ul}(t)$  and  $\mathbf{p}^{ur}(t)$ , which may intersect each other, consist of the left and right boundary of  $[\mathbf{P}^u](t)$ . That is, the boundary of  $[\mathbf{P}^u](t)$  is made up of segments of  $\mathbf{p}^{ul}(t)$  and  $\mathbf{p}^{ur}(t)$  divided by their local y-extremum points and their intersections.

To investigate the boundaries of  $[\mathbf{P}^d](t)$ ,  $[\mathbf{P}^l](t)$  and  $[\mathbf{P}^r](t)$ , we should consider  $\mathbf{p}^{dl}(t)$  and  $\mathbf{p}^{dr}(t)$  as well. Using similar analysis as above, we can obtain the similar conclusions for the boundaries structure of  $[\mathbf{P}^d](t)$ ,  $[\mathbf{P}^l](t)$  and  $[\mathbf{P}^r](t)$ . From the symmetry standpoint it is no loss of generality, if we do not pursue them further. Now it is possible to describe the boundary structure for the planar interval Bézier curve as follows.

**Theorem 1.** Assume that  $[\mathbf{P}](t)$  does not self intersect. The boundary of interval Bézier curve  $[\mathbf{P}](t)$  consists of the curve segments of  $\mathbf{p}^{ul}(t)$ ,  $\mathbf{p}^{ur}(t)$ ,  $\mathbf{p}^{dl}(t)$  and  $\mathbf{p}^{dr}(t)$ , the line segments connecting the corresponding y-extremum points between  $\mathbf{p}^{ul}(t)$  and  $\mathbf{p}^{ur}(t)$ , and between  $\mathbf{p}^{dl}(t)$  and  $\mathbf{p}^{dr}(t)$ , and the line segments connecting the corresponding x-extremum points between  $\mathbf{p}^{ul}(t)$  and  $\mathbf{p}^{dl}(t)$ , and between  $\mathbf{p}^{ur}(t)$  and  $\mathbf{p}^{dr}(t)$ .

### 3.2. Algorithm for boundary evaluation

In this section an algorithm for evaluating the boundary of 2-D interval Bézier curve  $[\mathbf{P}](t)$  is presented. We suggest using an algorithm based on the tracing of the segments of the vertex curves  $\mathbf{p}^{ul}(t)$ ,  $\mathbf{p}^{ur}(t)$ ,  $\mathbf{p}^{dl}(t)$  and  $\mathbf{p}^{dr}(t)$ , as well as the line segments of their corresponding extremum points. Here we call the extremum points and intersection points as crucial points.

The tracing algorithm proceeds as follows. First find a starting point on the boundary of the interval Bézier curve  $[\mathbf{P}](t)$  (this starting point can be selected as some vertex of  $\partial([\mathbf{P}_0])$  lying on the boundary). Then, with the interval Bézier curve to our right, we march along the corresponding vertex curve until it reaches a crucial point. At that point, we again continue to march along the next corresponding vertex curve. This is repeated until we return to the starting point.

As we march along, we store the points and the vertex curves in the order in which they are encountered. They represent the boundary of the interval Bézier curve. Furthermore, we can present the boundary as a single closed B-spline curve.

A sketch of the tracing algorithm follows:

**Algorithm 1.** Boundary evaluation for planar interval Bézier curve  $[\mathbf{P}](t)$ .

Set  $\mathbf{p}_0 = \mathbf{p}^{ul}(t)$ ,  $\mathbf{p}_1 = \mathbf{p}^{ur}(t)$ ,  $\mathbf{p}_2 = \mathbf{p}^{dr}(t)$ ,  $\mathbf{p}_3 = \mathbf{p}^{dl}(t)$ ; Calculate the intersections between  $\mathbf{p}_i$  and  $\mathbf{p}_{(i+1) \bmod 4}$

( $i = 0, 1, 2, 3$ ), and record their correspondence; Calculate the y-extremum points of  $\mathbf{p}_i$  and  $\mathbf{p}_{i+1}$  ( $i = 0, 2$ ), and record their correspondence; Calculate the x-extremum points of  $\mathbf{p}_i$  and  $\mathbf{p}_{(i+1) \bmod 4}$  ( $i = 1, 3$ ), and record their correspondence; Arrange the parameters of the extrema and intersections of  $\mathbf{p}_i$  ( $i = 0, 1, 2, 3$ ) in sequence; Determine the starting points  $A = A_0$  on  $\partial([\mathbf{P}_0])$  and its corresponding vertex curves  $\Gamma$ ; Initialize boundary segment list  $\Psi = \phi$ ;

**do**

Find the next crucial point  $B$  on  $\Gamma$ ;

Add the curve segment from  $A$  to  $B$  on  $\Gamma$  to  $\Psi$ ;

**if** ( $B$  is an extremum point of  $\Gamma$ )

Find the corresponding extremum point  $C$  and the corresponding vertex curve  $\Lambda$ ;

Add the line segment connecting  $B$  and  $C$  to  $\Psi$ ;

$\Gamma = \Lambda$ ;  $A = C$ ;

**else if** ( $B$  is an intersection point of  $\Gamma$ )

Find the corresponding vertex curve  $\Lambda$ ;

$\Gamma = \Lambda$ ;  $A = B$ ;

**else if** ( $B$  is on  $\partial([\mathbf{P}_n])$ )

Find the other starting vertex  $C$  on  $\partial([\mathbf{P}_n])$  and the corresponding vertex curve  $\Lambda$ ;

Add the line segments from  $B$  to  $C$  on  $\partial([\mathbf{P}_n])$  to  $\Psi$ ;

$\Gamma = \Lambda$ ;  $A = C$ ;

**else if** ( $B$  is on  $\partial([\mathbf{P}_0])$ )

Add the line segments from  $B$  to  $A_0$  on  $\partial([\mathbf{P}_0])$  to  $\Psi$ ;

$A = A_0$ ;

**while** ( $A! = A_0$ );

**Output:** the B-spline boundary curve joining all the segments of  $\Psi$  in sequence.

It should be noted that the intersection computations between two Bézier clipping curves are needed in Algorithm 1. We use the Bézier clipping technique [24] for calculating the intersections.

**Example 1.** A planar cubic interval Bézier curve is illustrated in Fig. 4. The corresponding y-maximum points  $\mathbf{A}_1$  and  $\mathbf{A}_2$  and intersection  $\mathbf{B}_1$  between the vertex curves  $\mathbf{p}^{ul}(t)$  and  $\mathbf{p}^{ur}(t)$  lie on the boundary. The corresponding y-minimum points  $\mathbf{A}_3$  and  $\mathbf{A}_4$  and intersection  $\mathbf{B}_2$  between the vertex curves  $\mathbf{p}^{dl}(t)$  and  $\mathbf{p}^{dr}(t)$  lie on the boundary. The

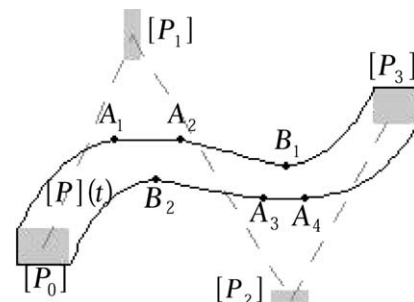


Fig. 4. The boundary of a planar cubic interval Bézier curve.

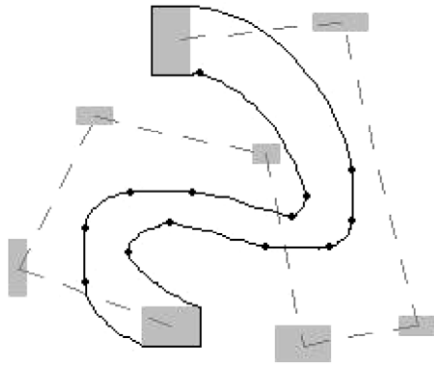


Fig. 5. The boundary of a planar degree 7 interval Bézier curve.

boundary of the interval Bézier curve is made up of 6 cubic Bézier curve segments and 6 line segments.

**Example 2.** Fig. 5 shows a planar degree 7 interval Bézier curve. The boundary of the interval Bézier curve is represented with 11 Bézier curve segments and 9 line segments.

3.3. Self-intersection elimination

One problem which may arise is that the interval Bézier curve may intersect itself. Loosely speaking, this is likely to happen if the curve passes close to earlier points, close being interpreted as meaning within a distance less than the size of the moving rectangle. It also happens if the curve self-intersect.

Like other methods for trimming problem, it can be solved by decomposing the interval Bézier curve into several monotone components. Then we calculate the boundary of each component using the previously outlined algorithm and do the trimming procedure to obtain the final boundary. Fig. 6 shows the trimming boundaries of two interval Bézier curves that have self-intersections.

4. Boundary evaluation for 3-D interval Bézier curve

In this section we briefly discuss the boundary evaluation for the interval Bézier curves in three-dimensional space. In Ref. [25] it is noted that three-dimensional geometry is considerably harder than two-dimensional geometry and many two-dimensional algorithms do not extend naturally to three or higher dimensions. We therefore expect that

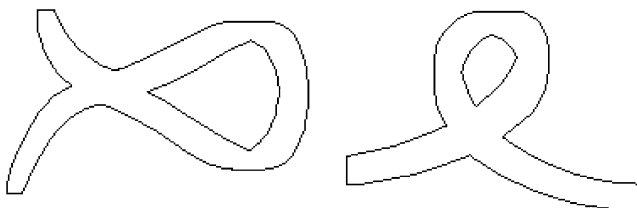


Fig. 6. Trimming boundary of interval Bézier curve.

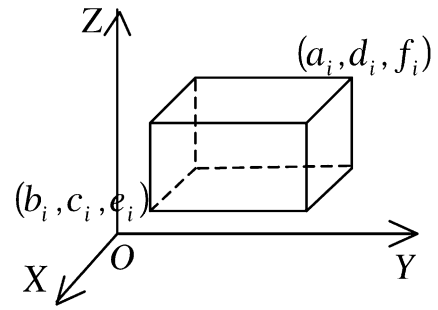


Fig. 7. A 3-D interval control box.

boundary evaluation for 3-D interval Bézier curves are more complex and difficult to obtain.

4.1. Surface patches on the boundary

**Definition 5.** A 3-D interval Bézier curve  $[P](t)$  of degree  $n$  is defined by

$$[P](t) = \sum_{i=0}^n [P_i] B_i^n(t) \quad 0 \leq t \leq 1 \quad (21)$$

where the interval control points  $[P_i] = ([a_i, b_i], [c_i, d_i], [e_i, f_i])$ ,  $i = 0, 1, \dots, n$ , are boxes in the space (Fig. 7).

A 3-D interval Bézier curve is defined by replacing the control rectangles of the 2-D interval Bézier curve by boxes in the space. A 3-D cubic interval Bézier curve is illustrated in Fig. 8.

Denote the upper, lower, front, back, left, right face of the interval point  $[P_i]$  by

$$[P_i^u] = ([a_i, b_i], [c_i, d_i], f_i), \quad [P_i^d] = ([a_i, b_i], [c_i, d_i], e_i),$$

$$[P_i^f] = (b_i, [c_i, d_i], [e_i, f_i]), \quad [P_i^b] = (a_i, [c_i, d_i], [e_i, f_i]),$$

$$[P_i^l] = ([a_i, b_i], c_i, [e_i, f_i]), \quad [P_i^r] = ([a_i, b_i], d_i, [e_i, f_i]),$$

$i = 0, 1, \dots, n$ , respectively. We obtain 6 interval Bézier

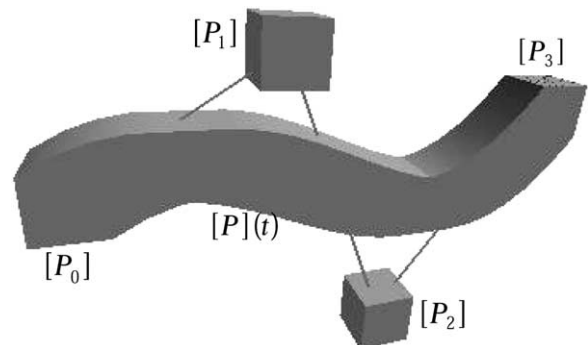


Fig. 8. A cubic 3-D interval Bézier curve.

curves with degree  $n$  as:

$$[\mathbf{P}^g](t) = \sum_{i=0}^n [\mathbf{P}_i^g] B_i^n(t), \quad g = u, d, f, b, l, r. \quad (22)$$

**Definition 6.** A point  $\mathbf{P} = (x, y, z) \in [\mathbf{P}](t)$  is called a boundary point, if every neighborhood of  $\mathbf{P}$  does not entirely lie in  $[\mathbf{P}](t)$ . A boundary point  $\mathbf{P}$  is called

1. an upper boundary point, if for arbitrary  $\epsilon > 0$ ,  $(x, y, z + \epsilon) \notin [\mathbf{P}](t)$ ;
2. a lower boundary point, if for arbitrary  $\epsilon > 0$ ,  $(x, y, z - \epsilon) \notin [\mathbf{P}](t)$ ;
3. a front boundary point, if for arbitrary  $\epsilon > 0$ ,  $(x + \epsilon, y, z) \notin [\mathbf{P}](t)$ ;
4. a back boundary point, if for arbitrary  $\epsilon > 0$ ,  $(x - \epsilon, y, z) \notin [\mathbf{P}](t)$ ;
5. a left boundary point, if for arbitrary  $\epsilon > 0$ ,  $(x, y - \epsilon, z) \notin [\mathbf{P}](t)$ ;
6. a right boundary point, if for arbitrary  $\epsilon > 0$ ,  $(x, y + \epsilon, z) \notin [\mathbf{P}](t)$ .

The sets of the upper, lower, front, back, left, right boundary points of  $[\mathbf{P}](t)$  are denoted by  $UpBn([\mathbf{P}](t))$ ,  $LoBn([\mathbf{P}](t))$ ,  $FrBn([\mathbf{P}](t))$ ,  $BaBn([\mathbf{P}](t))$ ,  $LeBn([\mathbf{P}](t))$ ,  $RiBn([\mathbf{P}](t))$ , respectively.

**Lemma 4.** For a space interval Bézier curve  $[\mathbf{P}](t)$  with degree  $n$ , we have

$$UpBn([\mathbf{P}](t)) \subseteq UpBn([\mathbf{P}^u](t)); \quad (23)$$

$$LoBn([\mathbf{P}](t)) \subseteq LoBn([\mathbf{P}^l](t)); \quad (24)$$

$$FrBn([\mathbf{P}](t)) \subseteq FrBn([\mathbf{P}^f](t)); \quad (25)$$

$$BaBn([\mathbf{P}](t)) \subseteq BaBn([\mathbf{P}^b](t)); \quad (26)$$

$$LeBn([\mathbf{P}](t)) \subseteq LeBn([\mathbf{P}^l](t)); \quad (27)$$

$$RiBn([\mathbf{P}](t)) \subseteq RiBn([\mathbf{P}^r](t)). \quad (28)$$

**Proof.** Similar to the proof of Lemma 1.

Denote the eight vertices of the control boxes by  $\mathbf{P}_i^0 = (b_i, c_i, f_i)$ ,  $\mathbf{P}_i^1 = (b_i, d_i, f_i)$ ,  $\mathbf{P}_i^2 = (a_i, d_i, f_i)$ ,  $\mathbf{P}_i^3 = (a_i, c_i, f_i)$ ,  $\mathbf{P}_i^4 = (b_i, c_i, e_i)$ ,  $\mathbf{P}_i^5 = (b_i, d_i, e_i)$ ,  $\mathbf{P}_i^6 = (a_i, d_i, e_i)$ ,  $\mathbf{P}_i^7 = (a_i, c_i, e_i)$ ,  $i = 0, 1, \dots, n$ .

We investigate the upper boundary of  $[\mathbf{P}^u](t)$  in the

following. The left edge interval curve

$$[\mathbf{P}_1^u](t) = \sum_{i=0}^n ([a_i, b_i], c_i, f_i) B_i^n(t) \quad (29)$$

is a degree  $n \times 1$  Bézier surface

$$\mathbf{Q}^1(s, t) = \sum_{i=0}^n \sum_{j=0}^1 \mathbf{Q}_{ij}^1 B_i^n(s) B_j^1(t), \quad 0 \leq s, t \leq 1, \quad (30)$$

where  $\mathbf{Q}_{i0}^1 = \mathbf{P}_i^0$ , and  $\mathbf{Q}_{i1}^1 = \mathbf{P}_i^3$ ,  $i = 0, 1, \dots, n$ . Similarly, the right, front and back edge interval curves are respectively degree  $n \times 1$  Bézier surfaces as

$$\mathbf{Q}^2(s, t) = \sum_{i=0}^n \sum_{j=0}^1 \mathbf{Q}_{ij}^2 B_i^n(s) B_j^1(t), \quad 0 \leq s, t \leq 1, \quad (31)$$

$$\mathbf{Q}^3(s, t) = \sum_{i=0}^n \sum_{j=0}^1 \mathbf{Q}_{ij}^3 B_i^n(s) B_j^1(t), \quad 0 \leq s, t \leq 1, \quad (32)$$

$$\mathbf{Q}^4(s, t) = \sum_{i=0}^n \sum_{j=0}^1 \mathbf{Q}_{ij}^4 B_i^n(s) B_j^1(t), \quad 0 \leq s, t \leq 1, \quad (33)$$

where  $\mathbf{Q}_{i0}^2 = \mathbf{P}_i^1$ ,  $\mathbf{Q}_{i1}^2 = \mathbf{P}_i^2$ ,  $\mathbf{Q}_{i0}^3 = \mathbf{P}_i^0$ ,  $\mathbf{Q}_{i1}^3 = \mathbf{P}_i^1$ ,  $\mathbf{Q}_{i0}^4 = \mathbf{P}_i^2$ , and  $\mathbf{Q}_{i1}^4 = \mathbf{P}_i^3$ ,  $i = 0, 1, \dots, n$ . It is easily seen that  $\mathbf{Q}^1(s, t)$ ,  $\mathbf{Q}^2(s, t)$ ,  $\mathbf{Q}^3(s, t)$ , and  $\mathbf{Q}^4(s, t)$  are 4 ruled surfaces.

**Lemma 5.** For the left, right, front and back boundary of  $[\mathbf{P}^u](t)$ , we have

$$LeBn([\mathbf{P}^u](t)) \subseteq \mathbf{Q}^1(s, t), \quad (34)$$

$$RiBn([\mathbf{P}^u](t)) \subseteq \mathbf{Q}^2(s, t), \quad (35)$$

$$FrBn([\mathbf{P}^u](t)) \subseteq \mathbf{Q}^3(s, t), \quad (36)$$

and

$$BaBn([\mathbf{P}^u](t)) \subseteq \mathbf{Q}^4(s, t). \quad (37)$$

**Lemma 6.** Assume that  $[\mathbf{P}^u](t)$  does not self intersect. The rectangular patches made up of the corresponding  $z$ -extremum lines of  $\mathbf{Q}^1(s, t)$ ,  $\mathbf{Q}^2(s, t)$ ,  $\mathbf{Q}^3(s, t)$ , and  $\mathbf{Q}^4(s, t)$  are perpendicular to the  $z$ -axis. They are parts of the lower/upper boundary of  $[\mathbf{P}^u](t)$ .

The proofs of Lemmas 5 and 6 are similar to the proofs of Lemmas 2 and 3, respectively, so we omit the details here.

**Remark 4.** It can be shown from Lemmas 5 and 6 that the boundary of  $[\mathbf{P}^u](t)$  consists of the trimmed patches of the ruled surfaces  $\mathbf{Q}^1(s, t)$ ,  $\mathbf{Q}^2(s, t)$ ,  $\mathbf{Q}^3(s, t)$ , and  $\mathbf{Q}^4(s, t)$  as well as the rectangular patches made up of their corresponding  $z$ -extremum lines.

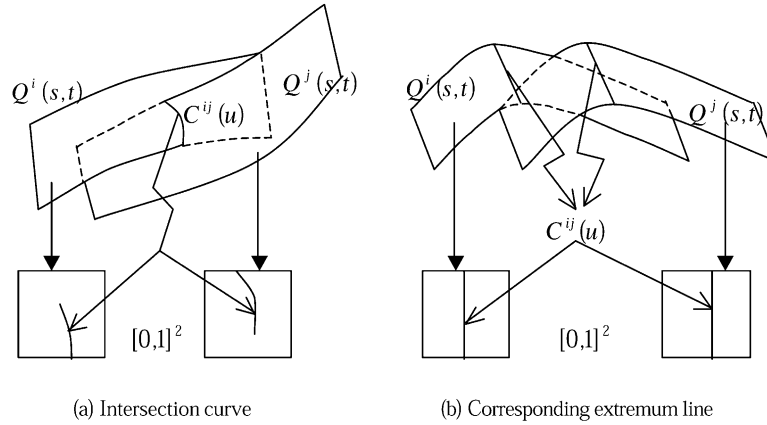


Fig. 9. The trimmed curve  $C^{ij}(u)$  between  $Q^i(s, t)$  and  $Q^j(s, t)$  (a) intersection curve; (b) corresponding extremum line.

Similarly, we can describe the boundary structure for  $[P^d](t)$ ,  $[P^f](t)$ ,  $[P^b](t)$ ,  $[P^l](t)$ , and  $[P^r](t)$ . It should further consider the following degree  $n \times 1$  Bézier surfaces:

$$Q^k(s, t) = \sum_{i=0}^n \sum_{j=0}^1 Q_{ij}^k B_i^n(s) B_j^1(t), \quad 0 \leq s, t \leq 1, \quad (38)$$

$k = 5, 6, \dots, 12,$

where  $Q_{i0}^5 = P_i^4, Q_{i1}^5 = P_i^7, Q_{i0}^6 = P_i^5, Q_{i1}^6 = P_i^6, Q_{i0}^7 = P_i^4, Q_{i1}^7 = P_i^5, Q_{i0}^8 = P_i^6, Q_{i1}^8 = P_i^7, Q_{i0}^9 = P_i^0, Q_{i1}^9 = P_i^4, Q_{i0}^{10} = P_i^1, Q_{i1}^{10} = P_i^5, Q_{i0}^{11} = P_i^2, Q_{i1}^{11} = P_i^6, Q_{i0}^{12} = P_i^3, Q_{i1}^{12} = P_i^7, i = 0, 1, \dots, n$ . The boundary structure for the 3-D interval Bézier curve is described as follows.

**Theorem 2.** Assume that  $[P](t)$  does not self intersect. The boundary of  $[P](t)$  are composed of the trimmed patches of the degree  $n \times 1$  Bézier surfaces  $Q^i(s, t)$  and the rectangular patches between the corresponding extremum lines of  $Q^i(s, t), i = 1, 2, \dots, 12$ .

4.2. Algorithm for boundary evaluation

We briefly describe a heuristic algorithm for calculating the boundary of the 3-D interval Bézier curve  $[P](t)$  in this section.

By Theorem 2 we know the boundary of 3-D interval Bézier curve  $[P](t)$  is composed of trimmed patches of Bézier surfaces and rectangular patches. A trimmed Bézier surface consists of two things: a tensor product Bézier surface, and a set of properly ordered trimmed curves lying with the parameter rectangle of the surface. The trimming curve can be hooked up to another trimming curve or to the boundaries of the surface. The trimmed surface boundaries are then obtained by mapping the 2-D trimming curves onto the surface.

First, we calculate the intersections between the degree  $n \times 1$  Bézier surfaces  $Q^i(s, t) (i = 1, 2, \dots, 12)$  as well as their corresponding  $x$ - $y$ - $z$ -extremum lines. The intersection

curves and the extremum lines are both the trimmed curves of the surfaces. The surface index  $i$  and  $j$  should be recorded in the data structure of the trimmed curve  $C^{ij}(u)$  if  $C^{ij}(u)$  is the intersection or the corresponding extremum line between the surfaces  $Q^i(s, t)$  and  $Q^j(s, t)$ (Fig. 9). The code of  $C^{ij}(u)$  also consists of corresponding information identifying the manner in which the trimmed patches in  $Q^i(s, t)$  and  $Q^j(s, t)$  lie with respect to it. The trimmed curves in parameter space  $[0, 1] \times [0, 1]$  are all properly joined to form the trimmed regions. Each of the ruled surfaces  $Q^i(s, t)$  is then subdivided into trimmed patches by the trimmed curves. Please refer to Fig.10 for an illustration.

We suggest using an algorithm based on the tracing of the trimmed patches of the surfaces  $Q^i(s, t) (i = 1, 2, \dots, 12)$  as well as the rectangular patches between their corresponding extremum lines.

The tracing algorithm proceeds as follows: starting from an initial trimmed patch on the boundary of  $[P](t)$ (this starting patch can be easily determined at  $\partial([P_0])$  or  $\partial([P_n])$ ), we march along the corresponding Bézier surface  $Q^i(s, t)$  until it reaches a trimmed curve. At that trimmed curve, we again continue to march along the next corresponding surface. This is repeated until we obtain the whole boundary patches.

The tracing procedure is performed in their parameter spaces of  $Q^i(s, t) (i = 1, 2, \dots, 12)$ . It proceeds by maintaining a ‘tracing-curve’ of the trimmed curves. The tracing-curves are recorded as a list of trimmed curves on a stack. The other trimmed curves in the trimmed region of the current surface will be pushed into the stack. The trimmed

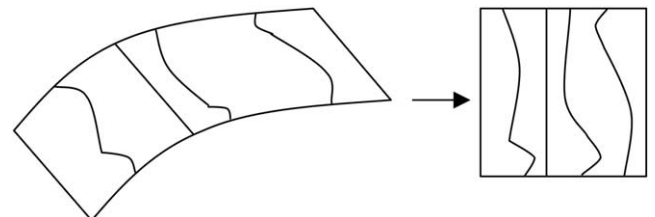


Fig. 10. The trimmed patches in the surface and its parameter domain.

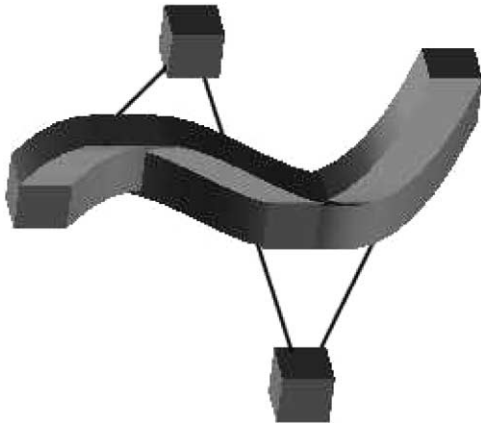


Fig. 11. The boundary of a cubic interval Bézier curve in 3-D.

region of the current surface is then stored as a part of the boundary  $\partial([\mathbf{P}](t))$ . The tracing-curve is then updated popping from the stack, and the procedure repeated. The procedure terminates when the stack is empty.

The trimmed curve is traced in parameter space, however, the trimmed patches on the boundary  $\partial([\mathbf{P}](t))$  are obtained in model space. Along with tracing the patches, the algorithm produces a compact database for browsing in the boundary patches network, e.g. finding all neighbours of a given boundary patch.

The main steps of the algorithm are as follows:

**Algorithm 2.** Boundary evaluation for 3-D interval Bézier curve  $[\mathbf{P}](t)$ .

Calculate the intersections and the corresponding extremum lines of  $\mathbf{Q}^i(s, t)$ ,  $i = 1, 2, \dots, 12$ ; Store the trimmed curves in the parameter domain of each surface  $\mathbf{Q}^i(s, t)$  and build their corresponding information; Determine the starting tracing-curve and push it on the stack;

**while** (stack is not empty) **do**

Pop stack to get current tracing-curve and get the corresponding surface;

**if** (the current tracing-curve has been traced) **continue**;

**if** (the current tracing-curve is an intersection curve)

Extract trimmed region over current surface's domain and store it;

Put the other trimmed curves on the boundary of the region on the stack;

**else** // the current tracing-curve is an extremum line

Store the extremum rectangular patch;

Put the other extremum lines of the rectangle on the stack;

**Output** all the trimmed patches and the rectangular patches as the boundary of  $[\mathbf{P}](t)$ .

**Example 3.** A cubic 3-D interval Bézier curve is illu-

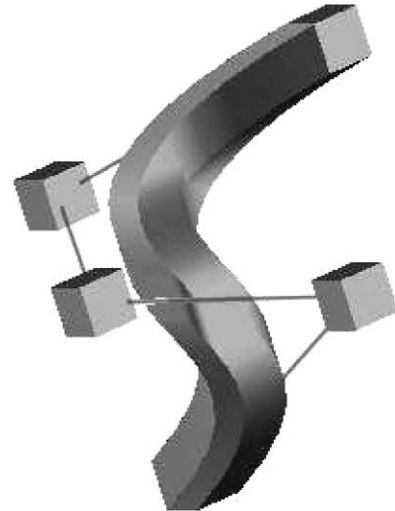


Fig. 12. The boundary of a degree 4 interval Bézier curve in 3-D.

strated in Fig. 11. The boundary is made up of 17 trimmed Bézier surface patches and nine rectangles.

**Example 4.** Fig. 12 shows a degree 4 interval Bézier curve in 3-D. The boundary is made up of 21 trimmed Bézier surface patches and 10 rectangles.

## 5. Conclusions

This paper discusses the boundary evaluation for the interval Bézier curves in 2- and 3-D. A reliable algorithm for calculating the exact boundary of the 2-D interval Bézier curve is presented. The boundary of the 2-D interval Bézier curve is represented by a sequence of Bézier curve segments and line segments in the order in which they are encountered while traversing the boundary. Then the algorithm determines the boundary of an interval Bézier curve in a plane by a single closed curve, which can be presented by a B-spline curve. The boundary structure of 3-D interval Bézier curve is also discussed. The boundary of 3-D interval Bézier curve consists of trimmed degree  $n \times 1$  Bézier surface patches and rectangular patches. And the algorithm for evaluating the boundary of 3-D interval Bézier curve is presented.

Our algorithm may be easily extended to interval rational Bézier curves and interval NURBS curves. The only difference is the polynomial equations which determine the local extremum points of curves are of higher degree and the intersection calculation is more complex. Furthermore, how to deal with self-intersection more efficiently is a topic of future research. Finally, the boundary evaluation for interval Bézier surface should be studied in the future.

## Acknowledgements

This research is supported by the National Natural

Science Foundation of China under grant no. 69973041, Zhejiang Provincial Natural Science Foundation of China under grant no.698025, and Foundation of State Key Basic Research 973 Item under grant no. G1998030600. Thanks are also due to the reviewers, whose perceptive comments led to an improvement of the original manuscript.

## References

- [1] Sederberg TW, Farouki RT. Approximated by interval Bézier curves. *IEEE Computer Graphics and its Application* 1992;15(2):87–95.
- [2] Moore RE. Interval analysis. Englewood Cliffs, NJ: Prentice-Hall, New Jersey, 1966.
- [3] Mudur SP, Koparkar PA. Interval methods for processing geometric objects. *IEEE Computer Graphics and its Applications* 1984;4(2):7–17.
- [4] Snyder JM. Interval analysis for computer graphics. *SIGGRAPH'92, Computer Graphics* 1992;26(2):121–30.
- [5] Sherbrooke EC, Patrikalakis NM. Computation of the solutions of nonlinear polynomial systems. *Computer Aided Geometric Design* 1993;10(5):379–405.
- [6] Hu CY, Maekawa T, Sherbrooke EC, Patrikalakis NM. Robust interval algorithm for curve intersections. *Computer Aided Design* 1996;28(6/7):495–506.
- [7] Hu CY, Maekawa EC, Patrikalakis NM, Ye XZ. Robust interval algorithm for surface intersections. *Computer Aided Design* 1997;29(9):617–27.
- [8] Hu CY, Patrikalakis NM, Ye XZ. Robust interval solid modelling—Part I: representations. *Computer Aided Design* 1996;28(10):807–17.
- [9] Hu CY, Patrikalakis NM, Ye XZ. Robust interval solid modelling—Part I: boundary evaluation. *Computer Aided Design* 1996;28(10):819–30.
- [10] Tuohy ST, Yoon JW, Patrikalakis NM. Reliable interrogation of 3-D non-linear geophysical databases. In: Vince JA, Earnshaw RA, editors. *Computer Graphics: Developments in Virtual Environments, Proceedings of CG International'95*, Leeds, UK, London: Academic Press, 1995. p. 327–41.
- [11] Abrams SL, Cho W, Hu CY, Maekawa T, Patrikalakis NM, Sherbrooke EC, Ye XZ. Efficient and reliable methods for rounded-interval arithmetic. *Computer Aided Design* 1998;30(8):657–65.
- [12] Sederberg TW, Buehler DB. Offsets of polynomial Bézier curves: Hermite approximation with error bounds. In: Lyché T, Schumaker LL, editors. *Mathematical methods in computer aided geometric design*, vol. II. Academic Press, Boston, 1992. p. 549–58.
- [13] Tuohy ST, Patrikalakis NM. Representation of geographical maps with uncertainty. In: Magnenat-Thalmann N, Thalmann D, editors. *Communicating with Virtual Worlds, Proceedings of CG International '93*, Tokyo: Springer, 1993. p. 179–92.
- [14] Maekawa T, Patrikalakis NM. Computation of singularities and intersections of offsets of planar curves. *Computer Aided Geometric Design* 1993;10(5):407–29.
- [15] Maekawa T, Patrikalakis NM. Interrogation of differential geometry properties for design and manufacture. *The Visual Computer* 1994;10(4):216–37.
- [16] Tuohy ST, Maekawa T, Shen G, Patrikalakis NM. Approximation of measured data with interval B-splines. *Computer Aided Design* 1997;29(11):791–9.
- [17] Shen G, Patrikalakis NM. Numerical and geometric properties of interval B-splines. *International Journal of Shape Modeling* 1998;4(1/2):35–62.
- [18] Chen FL, Lou WP. Degree reduction of interval Bézier curves. *Computer Aided Design* 2000;32(5):571–82.
- [19] Foley J, Dam A, Feiner S, Hughes J. *Computer graphics: principles and practice*. 2nd ed. Reading, MA: Addison-Wesley, 1990.
- [20] Wang WP, Wang KK. Geometric modeling for swept volume of moving solids. *IEEE Computer Graphics and its Applications* 1986;6:8–17.
- [21] Blackmore D, Leu MC, Shin F. Analysis and modeling of deformed swept volumes. *Computer Aided Design* 1994;26:315–26.
- [22] Parida L, Mudur SP. Computational methods for evaluating swept object boundaries. *The Visual Computer* 1994;10:266–76.
- [23] Lee JH, Hong SJ, Kim MS. Polygonal boundary approximation for a 2-D general sweep based on envelope and boolean operations. *The Visual Computer* 2000;16:208–40.
- [24] Sederberg TW, Nishita T. Curve intersection using Bézier clipping. *Computer Aided Design* 1990;22(9):538–49.
- [25] Forrest AR. Computational geometry and software engineering: towards a geometric computing environment. In: Rodgers D, Earnshaw R, editors. *Techniques for computer graphics*, New York: Springer, 1987.