The Visual Computer

Convolution surfaces for arcs and quadratic curves with a varying kernel

Xiaogang Jin[1],
Chiew-Lan Tai[2]

[1] State Key Lab of CAD&CG, Zhejiang University, Hangzhou, 310027, P. R. China
E-mail: jin@cad.zju.edu.cn
[2] Department of Computer Science, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong
E-mail: taicl@cs.ust.hk

Published online: November 20, 2002

A convolution surface is an isosurface in a scalar field defined by convolving a skeleton, comprising of points, curves, surfaces, or volumes, with a potential function. While convolution surfaces are attractive for modeling natural phenomena and objects of complex evolving topology, the analytical evaluation of integrals of convolution models still poses some open problems. This paper presents some novel analytical convolution solutions for arcs and quadratic spline curves with a varying kernel. In addition, we approximate planar higher-degree polynomial spline curves by optimal arc splines within a prescribed tolerance and sum the potential functions of all the arc primitives to approximate the field for the entire spline curve.

Key words: Geometric modeling – Convolution surface – Analytical – Quadratic curve

1 Introduction

Field-based implicit surfaces have become an increasingly popular modeling approach in recent years (Bloomenthal et al. 1997; Cani-Gascuel and Desbrun 1997). Their implicit representations, which have smooth-blending properties, make them convenient for modeling and animating smooth objects of complex topology that may change over time. Examples of such objects are liquids, clouds, plants, sea-life forms, and other organic shapes. In addition to object modeling, implicit surfaces have gained acceptance in other applications, namely, shape morphing (Turk and O'Brien 1999), surface reconstruction (Savchenko et al. 1995), natural phenomena simulation (Dobashi et al. 2000; Nishita et al. 1997), and space deformation (Jin et al. 2000). Since implicit surface can produce visually striking special effects, they have become a powerful tool for animators to convert their imaginations into reality.

Several types of implicit surfaces have appeared in the literature. They include metaballs, (Wyvill et al. 1989), distance surfaces (Bloomenthal and Wyvil 1990), convolution surfaces (Bloomenthal and Shoemake 1991), R-functions (Pasko et al. 1995), variational surfaces (Savchenko et al. 1995) and blob trees (Wyvill et al. 1999). The implicit functions in metaballs (or blobs, soft objects) are defined as a summation of point fields (Blanc and Schlick 1995), which are widely implemented in commercial modeling packages and are supported by many public-domain ray-tracers. However, despite their simplicity and popularity, point-based field surfaces have some drawbacks: modeling flat surfaces requires many closely packed metaballs to avoid bumps; incompact representation for objects whose skeletons are not points, but curves. Distance surface overcomes these drawbacks since it allows higher dimensional skeletons (Bloomenthal and Wyvill 1990; Bloomenthal 1995). Unfortunately, distance surfaces have their own weakness in that when a skeleton is not convex, the surface may have bulges, creases, and curvature discontinuity (Bloomenthal 1997).

Convolution surface was proposed by Bloomenthal and Shoemake (1991) as a natural and powerful extension to point-based field surfaces. A convolution surface is obtained by convolving a skeleton – which in principle can comprise points, line segments, curves, polygons, or other geometrical primitives – with a three-dimensional, low-pass Gaussian filter kernel. This approach over-

Digital Object Identifier (DOI) 10.1007/s00371-002-0161-3

comes the drawback of bulges and curvature discontinuity in distance surfaces. Convolution surfaces offer many desirable advantages, such as intuitive shape design, well-behaved blending and fluid topology changes with the underlying skeleton. Thus, they provide a powerful and flexible representation for modeling more complex objects than point-based field surfaces. Computer vision research has shown that any 3D object can be defined entirely from a geometric skeleton (Attali and Montanvert 1997), which implies that skeletons are natural abstractions for 3D objects. Convolution surfaces provide us with a means to control the shape of an underlying modeling object by controlling its skeleton, just as controlling a parametric surface by manipulating its control vertices. The lower dimension of skeleton also means simpler manipulation.

The mathematical formulations of convolution surfaces still pose some open problems because there are limited choices of kernel functions and skeletal primitives that can be convolved together analytically. By using the superposition property of convolution surfaces and the separable property of Gaussian filters, Bloomenthal and Shoemake (1991) calculated field functions numerically based on a point-sampling method, which unfortunately implies potential under-sampling artifacts and storage problems. McCormack and Sherstyuk (1998) addressed this weakness by employing a new kernel function, called Cauchy function, and were able to deduce analytical solutions for several useful primitives, namely, points, line segments, polygons, arcs, and planes. Since many complex skeletons can be divided into these simpler primitives, their work enables convolution surfaces of many general shapes to be modeled accurately and robustly.

The analytical models for line segments and arcs derived by McCormack and Sherstyuk assume constant weight distribution along a primitive, and therefore can only produce constant-radius convolution surfaces from each primitive. Furthermore, no analytical model for spline curves was proposed. We note that, unlike varying-radius distance surfaces, which can be produced by simply changing the distance function in the field computation (Ferley 1997), varying-radius convolution surfaces cannot be produced in the same way. Although implicit surfaces with varying radius based on curves and lines can be easily faked by convolving a constant kernel, and then multiplying the result by some scalar function $s(t)$, where t is the distance along the curve (Sherstyuk 1999b) this approach has several weaknesses. Firstly, as the resultant surface is no longer a true convolution surface, the superposition property of convolution surfaces no longer holds. Secondly, when determining the t parameter value for the scalar function $s(t)$ for a curve skeleton, ambiguity may arise since there may be more than one closest point to the skeleton for a space point. As a result, potential bulges and discontinuities may occur in the resultant surfaces.

In this paper, we propose some new analytical convolution solutions for arc and quadratic spline skeleton with polynomial weight distributions. For planar, higher-degree polynomial splines, such as Bezier, B-spline, or NURBS curves, we approximate them by GC^1 optimal arc splines to within a prescribed tolerance. The field for the entire curve can then be computed by summing the potential function of all arc primitives.

The remainder of this paper is organized as follows. The polynomial-weighted convolution surface model is introduced in Sect. 2. The field function computation for line segments, arcs, and quadratic curves with polynomial weight distribution are discussed in Sects. 3, 4 and 5 respectively. Section 6 presents the field computation for planar higher-degree polynomial spline curves via an optimal arc-spline approximation. Section 7 describes implementation details and results. Conclusions and future work are discussed in Sect. 8.

2 Polynomial-weighted convolution surfaces

Let $\boldsymbol{P}$ be a space point, and let f be a potential function describing the field generated by a single point $\boldsymbol{Q}$ in a curve skeleton g, then the field function of the convolution surface for the curve skeleton is

$$F(\boldsymbol{P}) = \int_g f(\boldsymbol{P} - \boldsymbol{Q}) \mathrm{d}s. \tag{1}$$

where $\mathrm{d}s$ is the differential arc length of the curve skeleton, and f is called the convolution kernel function.

An important property that makes convolution surfaces suitable for modeling is superposition, which

implies that summing the convolution surfaces generated by two separate skeletons yields the same surface as that generated by their combined skeleton. This independent evaluation property ensures that a skeleton can be arbitrarily subdivided into sub-skeletons whose field functions can be simply summed to evaluate the final convolution surface.

Several kernel functions have been adopted in implicit surface modeling, such as Gaussian, inverse linear, inverse squared, Cauchy, and polynomial functions. Polynomial functions have been proven to be very effective and are widely used because of their small computation cost. For example, Nishimura et al. use piecewise quadratic polynomials to define metaballs (Bloomenthal et al. 1997); Wyvill and Wyvill (1999) introduced a six-degree polynomial to model soft objects; quartic polynomials are used in ray-tracing software such as Rayshade and POV-Ray. In this paper, we adopt the quartic polynomial as the kernel function because it leads to simplest computation. The quartic polynomial kernel is defined as

$$f(r^2) = \begin{cases} \left(1 - \frac{r^2}{R^2}\right)^2, & r \le R \\ 0 & r > R \end{cases}, \tag{2}$$

where R is the effective radius of the kernel.

The ability to design convolution surfaces of varying profiles from curve skeletons would facilitate the modeling of many shapes. By using a cubic control curve to define a polynomial distribution function $q(\boldsymbol{Q}):R^3 \rightarrow R$ along a curve skeleton, and multiplying the field function of a point $\boldsymbol{Q}$ in the skeleton by $q(\boldsymbol{Q})$, the convolution model in Eq. (1) now becomes a weighted convolution surface model with polynomial weight distribution:

$$F(\boldsymbol{P}) = \int_g q(\boldsymbol{Q}) f(\boldsymbol{P} - \boldsymbol{Q}) \mathrm{d}s. \tag{3}$$

We represent the cubic control curve as a one-dimensional Bezier curve (Farin 1997), which is an intuitive and computationally efficient tool for shape control. Assuming that the control points are $\left(\frac{i}{n}, q_i\right), \quad i = 0, 1, 2, 3$, and based on the identity $\sum_{i=0}^{n} \frac{i}{n} B_{i,n}(u) = u[(1-u)+u]^n = u$, where $B_{i,n}(u)$are Bernstein polynomials, the curve can be rewritten as $q(u) = \sum_{i=0}^{n} q_i B_{i,n}(u)$. Figure 1 shows our interface for controlling the polynomial distribution curve. The user can move the Bezier control vertices q_0, q_1, q_2, q_3 vertically to adjust the curve.

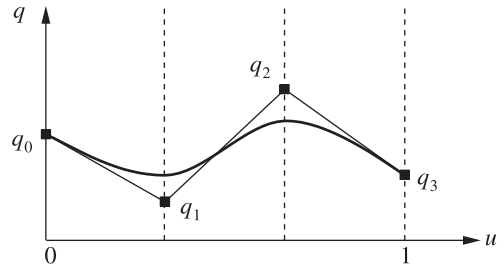


Fig. 1. A cubic curve for controlling the polynomial function distribution

With the kernel and the polynomial weight distribution function defined, for a one-dimensional skeleton with parameter t, we can now write the field of a point $\boldsymbol{P}$ of interest as

$$F(\boldsymbol{P}) = \int \left\{ \sum_{i=0}^{3} q_i B_{i,3}(u(t)) \right\} f(r^2(t)) \mathrm{d}t. \tag{4}$$

By modulating the weight of the integration kernel along the skeleton curve, a convolution surface with varying radius can be achieved.

3 Field computation for line segments with varying kernels

A line segment of length l with start point $\boldsymbol{b}$ and unit direction $\boldsymbol{n}$ can be represented parametrically as

$$\boldsymbol{L}(t) = \boldsymbol{b} + t\boldsymbol{n}, 0 \le t \le l. \tag{5}$$

Letting $\boldsymbol{d} = \boldsymbol{P} - \boldsymbol{b}$, the squared distance from the point $\boldsymbol{P}$ to a point on the line $\boldsymbol{L}(t)$ is given by

$$r^2(t) = \|\boldsymbol{d}\|^2 + t^2 - 2t\boldsymbol{d} \cdot \boldsymbol{n} = (t-h)^2 + (d^2 - h^2), \tag{6}$$

where $d = \|\boldsymbol{d}\|$ and $h = \boldsymbol{d} \cdot \boldsymbol{n}$.

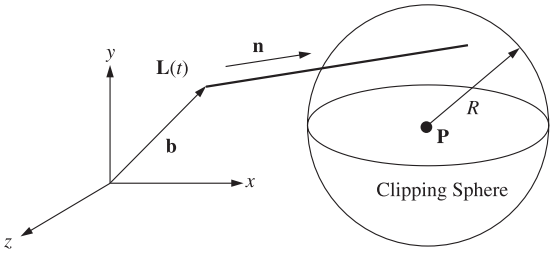


Fig. 2. Calculating the integration domain of a line segment

3.1 Effective-span computation for line segments

The quartic polynomial is a finite-support piecewise function; that is, for any point on a line segment whose distance from $\boldsymbol{P}$ is larger than R, its field contribution to $\boldsymbol{P}$ is zero. Thus, when calculating the field at the point $\boldsymbol{P}$, we must use a sphere centered at $\boldsymbol{P}$ with radius R to clip the line segment to find the effective span that contributes to that field (Fig. 2). We call this sphere the *clipping sphere* of $\boldsymbol{P}$.

The equation of the clipping sphere is

$$(\boldsymbol{X}-\boldsymbol{P})^2 = R^2. \tag{7}$$

Substituting the line segment equation (5) into the equation for the clipping sphere (7) gives

$$(t-h)^2 = R^2 - d^2 + h^2. \tag{8}$$

If the discriminant $\Delta = R^2 - d^2 + h^2 < 0$, then there is no intersection between the line segment $\boldsymbol{L}(t)$ and the clipping sphere, thus the field contribution from the line segment to $\boldsymbol{P}$ is zero, and there is no effective span. If $\Delta \geq 0$, then there are two intersection points, which correspond to

$$t_1 = h - \sqrt{R^2 - d^2 + h^2},\ t_2 = h + \sqrt{R^2 - d^2 + h^2}. \tag{9}$$

If $t_2 < 0$ or $t_1 > l$, then there is no valid intersection, and thus no effective span. Otherwise, the effective span is $t \in [l_1, l_2]$, where $l_1 = \max(0, t_1)$ and $l_2 = \min(l, t_2)$.

3.2 Field computation for line segments

If the effective span is $[l_1, l_2]$, where $l_1 < l_2$, the field of a point $\boldsymbol{P}$ is

$$\begin{aligned} F_{\text{line}}(\boldsymbol{P}) &= \int_0^l \left\{ \sum_{i=0}^{3} q_i B_{i,3}\left(\frac{t}{l}\right) \right\} f(r^2(t))\mathrm{d}t \\ &= \sum_{i=0}^{3} \left\{ q_i' \frac{1}{R^4} \int_{l_1}^{l_2} t^i \left(t^2 - 2ht - (R^2 - d^2)\right)^2 \mathrm{d}t \right\} \\ &= q_0' F^1_{\text{line}}(\boldsymbol{P}) + q_1' F^t_{\text{line}}(\boldsymbol{P}) + q_2' F^{t^2}_{\text{line}}(\boldsymbol{P}) + q_3' F^{t^3}_{\text{line}}(\boldsymbol{P}) \end{aligned} \tag{10}$$

where $q_0' = q_0$, $q_1' = \frac{1}{l}(-3q_0 + 3q_1)$, $q_2' = \frac{1}{l^2}(3q_0 - 6q_1 + 3q_2)$, $q_3' = \frac{1}{l^3}(-q_0 + 3q_1 - 3q_2 + q_3)$, and $F^{t^i}_{\text{line}}(\boldsymbol{P})$, $i = 0, 1, 2, 3$, are the field functions of the line segment $\boldsymbol{L}(t)$ with weight distribution t^i defined as

$$F^{t^i}_{\text{line}}(\boldsymbol{P}) = \frac{1}{R^4} \int_{l_1}^{l_2} t^i \left(t^2 - 2ht - (R^2 - d^2)\right)^2 \mathrm{d}t. \tag{11}$$

By applying integration techniques, $F^{t^i}_{\text{line}}(\boldsymbol{P})$can be calculated using the following formulae,

$$\begin{aligned} F^1_{\text{line}}(\boldsymbol{P}) = \frac{1}{R^4}\Bigg(& \frac{1}{5}\left(l_2^5 - l_1^5\right) - \left(l_2^4 - l_1^4\right) h \\ & + \frac{2}{3}\left(l_2^3 - l_1^3\right)\left(2h^2 - (R^2 - d^2)\right) + 2\left(l_2^2 - l_1^2\right) \\ & \times h(R^2 - d^2) + (l_2 - l_1)\left(R^2 - d^2\right)^2 \Bigg), \end{aligned} \tag{12}$$

$$\begin{aligned} F^t_{\text{line}}(\boldsymbol{P}) = \frac{1}{R^4}\Bigg(& \frac{1}{6}\left(l_2^6 - l_1^6\right) - \frac{4}{5}\left(l_2^5 - l_1^5\right) h \\ & + \frac{1}{2}\left(l_2^4 - l_1^4\right)\left(2h^2 - (R^2 - d^2)\right) + \frac{4}{3}\left(l_2^3 - l_1^3\right) \\ & \times h(R^2 - d^2) + \frac{1}{2}\left(l_2^2 - l_1^2\right)\left(R^2 - d^2\right)^2 \Bigg), \end{aligned} \tag{13}$$

$$\begin{aligned} F^{t^2}_{\text{line}}(\boldsymbol{P}) = \frac{1}{R^4}\Bigg(& \frac{1}{7}\left(l_2^7 - l_1^7\right) - \frac{2}{3}\left(l_2^6 - l_1^6\right) h \\ & + \frac{2}{5}\left(l_2^5 - l_1^5\right)\left(2h^2 - (R^2 - d^2)\right) + \left(l_2^4 - l_1^4\right) \end{aligned}$$

$$\times h(R^2-d^2)+\frac{1}{3}\left(l_2^3-l_1^3\right)(R^2-d^2)^2\Bigg), \quad (14)$$

$$F_{\text{line}}^{t^3}(\boldsymbol{P})=\frac{1}{R^4}\Bigg(\frac{1}{8}\left(l_2^8-l_1^8\right)-\frac{4}{7}\left(l_2^7-l_1^7\right)h + \frac{1}{3}\left(l_2^6-l_1^6\right)(2h^2-(R^2-d^2))+\frac{4}{5}\left(l_2^5-l_1^5\right) \times h(R^2-d^2)+\frac{1}{4}\left(l_2^4-l_1^4\right)(R^2-d^2)^2\Bigg). \quad (15)$$

Due to the high-degree polynomial terms of l_1 and l_2 in the field computation, numerical errors may be large for long line segments. Hence, to improve the stability of the algorithm, after computing the effective span, we reparameterize the line segment as

$$\boldsymbol{L}'(t)=(\boldsymbol{b}+l_1\boldsymbol{n})+t\boldsymbol{n},\ 0\le t\le l_2-l_1. \quad (16)$$

The weight distribution curve over the interval $[l_1, l_2]$ can be obtained by Bezier subdivision algorithm (Farin 1997). Obviously, the effective span of $\boldsymbol{L}'(t)$ for the point $\boldsymbol{P}$ is $[0, l_2-l_1]$. Since l_2-l_1 is independent of the length of $\boldsymbol{L}(t)$, this field computation is more stable.

Linear weight distribution is the most frequently used in applications. Since Eqs. (14) and (15) are zero in this case, the computation can be simplified to only calculating Eqs. (12) and (13).

4 Field computation for arcs with varying kernels

Let $\boldsymbol{A}(t)$ be an arc defined in the arc's local z-aligned coordinate system,

$$\boldsymbol{A}(t)=(R_0\cos t, R_0\sin t, 0),\ \varphi_1\le t\le\varphi_2 \quad , \quad (17)$$

where R_0 is the radius of the arc, and φ_1 and φ_2 are the starting and ending angles of the arc (Fig. 3). The squared distance from $\boldsymbol{P}(x, y, z)$ to a point on the arc is then given by

$$r^2(t)=(x-R_0\cos t)^2+(y-R_0\sin t)^2+z^2 = d^2+R_0^2-2R_0(x\cos t+y\sin t). \quad (18)$$

where $d^2=x^2+y^2+z^2$.

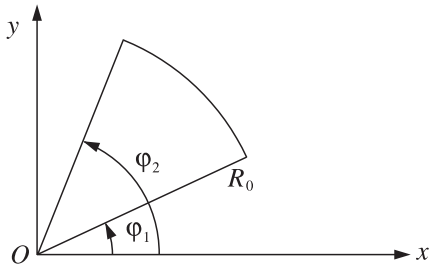


Fig. 3. An arc primitive

4.1 Effective-span computation for arcs

As with line-segment primitives, before computing the field of a point of interest $\boldsymbol{P}(x, y, z)$contributed by an arc, we use the clipping sphere to clip the arc to obtain the effective span of the arc contributing to $\boldsymbol{P}$. Clearly, the intersection points between the clipping sphere and the circle on which the arc lies satisfy the following equations:

$$\begin{cases}(\tilde{x}-x)^2+(\tilde{y}-y)^2+(\tilde{z}-z)^2=R^2\\ \tilde{x}^2+\tilde{y}^2=R_0^2\\ \tilde{z}=0\end{cases}. \quad (19)$$

Eliminating $\tilde{x}$ gives the following quadratic equation with unknown $\tilde{y}$:

$$4(x^2+y^2)\tilde{y}^2-4y(R_0^2+d^2-R^2)\tilde{y} + (R_0^2+d^2-R^2)^2-4x^2R_0^2=0. \quad (20)$$

The discriminant of Eq. (20) is

$$\Delta=16x^2\left(4(x^2+y^2)R_0^2-(R_0^2+d^2-R^2)^2\right). \quad (21)$$

If $4(x^2+y^2)R_0^2<(R_0^2+d^2-R^2)^2$, then there is no intersection between the clipping sphere and the circle (see case *a* in Fig. 4) and thus no effective span; otherwise there are two intersection points (see cases *b* and *c* in Fig.4).

Singularity will arise if $|x|<\varepsilon$ and $|y|<\varepsilon$, where ε is a specified small number. In this case, if $R^2-z^2>R_0^2$, then the effective span is the entire parameter range $[\varphi_1, \varphi_2]$ of the arc; otherwise there is no effective span. If it is not a singular case, in order to avoid the division-by-zero error and to reduce numerical error, we use the following formulae to calculate the intersection points $(\tilde{x}_1, \tilde{y}_1)$ and $(\tilde{x}_2, \tilde{y}_2)$ if

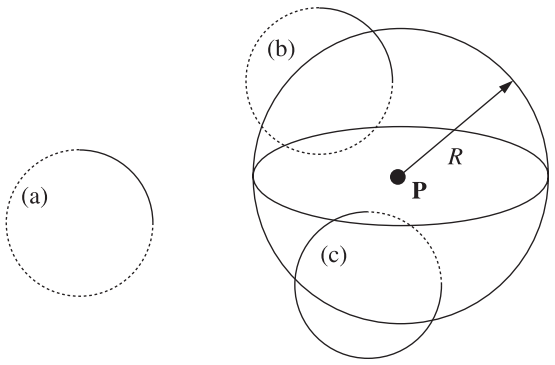


Fig. 4. A clipping sphere, and three possible arcs with the circles containing them

$|x| \geq |y|$:

$$\tilde{y}_1 = \frac{4y(R_0^2+d^2-R^2)-\sqrt{\Delta}}{8(x^2+y^2)},$$
$$\tilde{x}_1 = -\frac{y}{x}\tilde{y}_1 + \frac{R_0^2+d^2-R^2}{2x}, \quad (22)$$
$$\tilde{y}_2 = \frac{4y(R_0^2+d^2-R^2)+\sqrt{\Delta}}{8(x^2+y^2)},$$
$$\tilde{x}_2 = -\frac{y}{x}\tilde{y}_2 + \frac{R_0^2+d^2-R^2}{2x}. \quad (23)$$

If $|y| > |x|$, we eliminate $\tilde{y}$ in Eq. (19). The new discriminant is

$$\Delta = 16y^2\left(4(x^2+y^2)R_0^2 - (R_0^2+d^2-R^2)^2\right),$$

and the intersection points are calculated using the following formulae:

$$\tilde{x}_1 = \frac{4x(R_0^2+d^2-R^2)-\sqrt{\Delta}}{8(x^2+y^2)},$$
$$\tilde{y}_1 = -\frac{x}{y}\tilde{x}_1 + \frac{R_0^2+d^2-R^2}{2y}, \quad (24)$$
$$\tilde{x}_2 = \frac{4x(R_0^2+d^2-R^2)+\sqrt{\Delta}}{8(x^2+y^2)},$$
$$\tilde{y}_2 = -\frac{x}{y}\tilde{x}_2 + \frac{R_0^2+d^2-R^2}{2y}. \quad (25)$$

To find the effective parameter span of the arc after obtaining the intersection points, we examine the intersection between the clipping sphere and the plane $\tilde{z} = 0$, which produces a *clipping circle*. The radius of the clipping circle is $R_1 = \sqrt{R^2 - z^2}$. Based on the relative position of the clipping circle and the arc $\boldsymbol{A}(t)$, their intersection may produce zero, one, or two effective spans (Fig. 5). Let the two intersection points between the clipping circle and the circle containing the arc be $P_1(\tilde{x}_1, \tilde{y}_1)$ and $P_2(\tilde{x}_2, \tilde{y}_2)$. We determine which of the arcs OP_1P_2 and OP_2P_1 is inside the clipping circle (i.e., which one of them is the *intersection arc*) and swap P_1 and P_2 if it is the latter: If $R_1 < R_0$ (i.e., $R^2 - z^2 < R_0^2$), then the angle subtended by the intersection arc is less than π(Fig. 5a). In this case, if $OP_1 \times OP_2 > 0$, i.e., $\tilde{x}_1\tilde{y}_2 - \tilde{x}_2\tilde{y}_1 > 0$, then OP_1P_2 is the intersection arc; otherwise OP_2P_1 is the intersection arc, so we swap P_1 and P_2.

If $R_1 \geq R_0$, then P and O must lie on the same side of the line P_1P_2 (Fig. 5c). If P_1PP_2 is counterclockwise, then OP_1P_2 is the intersection arc; otherwise OP_2P_1 is the intersection arc, so we swap P_1 and P_2.

Now we have obtained the intersection arc OP_1P_2. The corresponding angles of P_1 and P_2 with respect to the positive x-axis are

$$\theta_1 = \text{atan2}(\tilde{y}_1, \tilde{x}_1), \theta_2 = \text{atan2}(\tilde{y}_2, \tilde{x}_2).$$

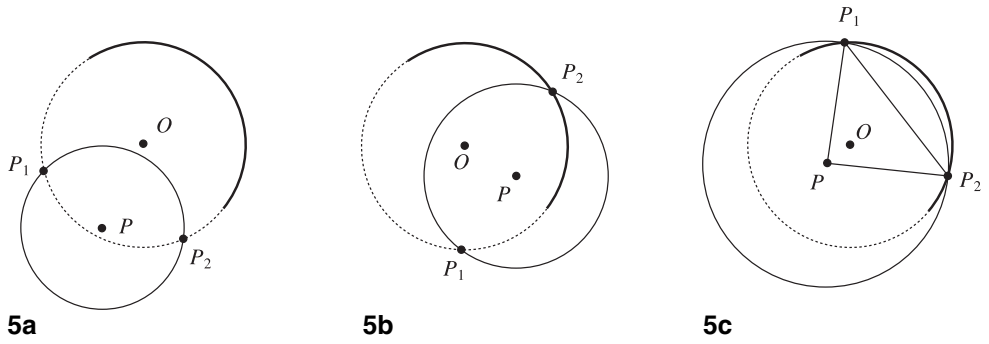
If $\theta_2 < \theta_1$, we set $\theta_2 = \theta_2 + 2\pi$ to ensure that θ_2 is larger than θ_1. In order to calculate the intersecting parameter interval(s) for the intersection arc and arc $\boldsymbol{A}(t)$, we set $\varphi = \varphi_1$ and subtract φ from $\varphi_1, \varphi_2, \theta_1, \theta_2$, which give

$$\varphi_1 = 0, \varphi_2 = \varphi_2 - \varphi, \theta_1 = \theta_1 - \varphi, \theta_2 = \theta_2 - \varphi.$$

If $\theta_1 < -2\pi$, we set $\theta_1 = \theta_1 + 2\pi$ and $\theta_2 = \theta_2 + 2\pi$; if $\theta_1 > 2\pi$, we set $\theta_1 = \theta_1 - 2\pi$ and $\theta_2 = \theta_2 - 2\pi$. After these transformations, we have $\theta_1 \in (-2\pi, 2\pi)$, $\theta_2 \in (-2\pi, 2\pi)$, and $\varphi_2 \in (0, 2\pi]$, and the effective span(s) can be computed according to the following cases:

Case 1: $\theta_1 > 0$ and $\theta_2 \leq 2\pi$ (arc a in Fig. 6). If $\theta_1 > \varphi_2$, there is no effective span; otherwise there is one effective span $[\theta_1 + \varphi, \min(\varphi_2, \theta_2) + \varphi]$.

Case 2: $\theta_1 > 0$ and $\theta_2 > 2\pi$ (arc b in Fig. 6). We subdivide the span $[\theta_1, \theta_2]$ into two sub-spans $[\theta_1, 2\pi]$ and $[0, \theta_2 - 2\pi]$, and compute the intersecting intervals between these two sub-spans and $[\varphi_1, \varphi_2]$. The union of the two intersecting intervals is the effective span.



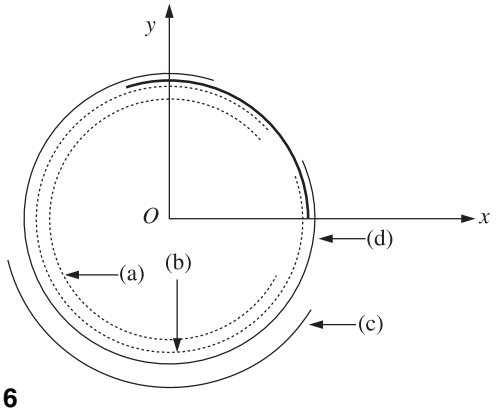


Fig. 5a–c. Three possible cases of intersections between the clipping circle of P and an arc with center O: **a** no effective span; **b** one effective span; **c** two effective spans
Fig. 6. Computation of the effective spans

Case 3: $\theta_1 < 0$ and $\theta_2 \le 0$ (arc c in Fig. 6). The effective span is the intersecting interval between $[\theta_1 + 2\pi, \theta_2 + 2\pi]$ and $[\varphi_1, \varphi_2]$.

Case 4: $\theta_1 < 0$ and $\theta_2 > 0$ (arc d in Fig. 6). We subdivide the span $[\theta_1, \theta_2]$ into two sub-spans $[0, \theta_2]$ and $[\theta_1 + 2\pi, 2\pi]$, and compute the intersecting intervals between these two sub-spans and $[\varphi_1, \varphi_2]$. The union of the resulting two intersecting intervals is the effective span.

4.2 Field computation for arcs

Let $\Delta\varphi = \varphi_2 - \varphi_1$, and let *SpanNum* be the number of effective spans. If there is only one effective span, let it be $[\theta_1, \theta_2]$; if there are two of them, let them be $[\theta_1, \theta_2]$ and $[\theta_3, \theta_4]$. The arc's analytical field function for a point $\boldsymbol{P}(x, y, z)$ is then

$$F_{\text{arc}}(\boldsymbol{P}) = \int_{\varphi_1}^{\varphi_2} \left\{ \sum_{i=0}^{3} q_i B_{i,3}\left(\frac{t-\varphi_1}{\Delta\varphi}\right) \right\} f(r^2(t))\mathrm{d}t$$

$$= \sum_{j=1}^{\text{SpanNum}} \sum_{i=0}^{3} \left\{ q_i' \frac{R_0}{R^4} \int_{\theta_{2j-1}}^{\theta_{2j}} t^i \left(R^2 - d^2 - R_0^2 + 2R_0(x\cos t + y\sin t)\right)^2 \mathrm{d}t \right\}$$

$$= \sum_{j=1}^{\text{SpanNum}} \left\{ q_0' \; {}_jF_{\text{arc}}^1(\boldsymbol{P}) + q_1' \; {}_jF_{\text{arc}}^t(\boldsymbol{P}) + q_2' \; {}_jF_{\text{arc}}^{t^2}(\boldsymbol{P}) + q_3' \; {}_jF_{\text{arc}}^{t^3}(\boldsymbol{P}) \right\}, \tag{26}$$

where

$$q_0' = \frac{1}{\Delta\varphi^3}\left(q_0\varphi_2^3 - 3q_1\varphi_1\varphi_2^2 + 3q_2\varphi_1^2\varphi_2 - q_3\varphi_1^3\right),$$

$$q_1' = \frac{1}{\Delta\varphi^3}\left(-3q_0\varphi_2^2 + 3q_1(\varphi_2^2 + 2\varphi_1\varphi_2) - 3q_2(\varphi_1^2 + 2\varphi_1\varphi_2) + 3q_3\varphi_1^2\right),$$

$$q_2' = \frac{1}{\Delta\varphi^3}\big(3q_0\varphi_2 - 3q_1(2\varphi_2+\varphi_1) + 3q_2(2\varphi_1+\varphi_2) - 3q_3\varphi_1\big),$$
$$q_3' = \frac{1}{\Delta\varphi^3}(-q_0+3q_1-3q_2+q_3),$$

and ${}_jF^{t^i}_{\text{arc}}(\boldsymbol{P})$, $i = 0, 1, 2, 3$ are the field functions of the arc $\boldsymbol{A}(t)$ for span j with weight distribution t^i defined as follows:

$${}_jF^{t^i}_{\text{line}}(\boldsymbol{P}) = \int_{\theta_{2j-1}}^{\theta_{2j}} t^i \left(R^2 - d^2 - R_0^2 + 2R_0(x\cos t + y\sin t)\right)^2 \mathrm{d}t. \tag{27}$$

By applying integration techniques, we obtain the following formulae for computing the various field functions for the first span $[\theta_1, \theta_2]$, the corresponding field functions can be calculated using the following formulae by applying integration techniques,

$$\begin{aligned}\frac{R^4}{R_0}F^1_{\text{arc}}(\boldsymbol{P}) &= 2R_0^2(x^2+y^2)(\theta_2-\theta_1)\\ &+ R_0^2(x^2-y^2)(\sin 2\theta_2 - \sin 2\theta_1)\\ &- 2R_0^2xy(\cos 2\theta_2 - \cos 2\theta_1)\\ &+ 4R_0(R^2-d^2-R_0^2)(x(\sin\theta_2-\sin\theta_1)\\ &\quad - y(\cos\theta_2-\cos\theta_1))\\ &+ \left(R^2-d^2-R_0^2\right)^2(\theta_2-\theta_1),\end{aligned} \tag{28}$$

$$\begin{aligned}\frac{R^4}{R_0}F^t_{\text{arc}}(\boldsymbol{P}) &= R_0^2(x^2+y^2)(\theta_2^2-\theta_1^2)\\ &+ R_0^2(x^2-y^2)(\theta_2\sin 2\theta_2 - \theta_1\sin 2\theta_1)\\ &+ \frac{1}{2}R_0^2(x^2-y^2)(\cos 2\theta_2 - \cos 2\theta_1)\\ &+ R_0^2xy(\sin 2\theta_2 - \sin 2\theta_1)\\ &- 2R_0^2xy(\theta_2\cos 2\theta_2 - \theta_1\cos 2\theta_1)\\ &+ 4R_0(R^2-d^2-R_0^2)x(\cos\theta_2-\cos\theta_1)\\ &+ 4R_0(R^2-d^2-R_0^2)x(\theta_2\sin\theta_2-\theta_1\sin\theta_1)\\ &+ 4R_0(R^2-d^2-R_0^2)y(\sin\theta_2-\sin\theta_1)\\ &- 4R_0(R^2-d^2-R_0^2)y(\theta_2\cos\theta_2-\theta_1\cos\theta_1)\\ &+ \frac{1}{2}\left(R^2-d^2-R_0^2\right)^2(\theta_2^2-\theta_1^2),\end{aligned} \tag{29}$$

$$\begin{aligned}\frac{R^4}{R_0}F^{t^2}_{\text{arc}}(\boldsymbol{P}) &= \frac{2}{3}R_0^2(x^2+y^2)(\theta_2^3-\theta_1^3)\\ &+ R_0^2(x^2-y^2)(\theta_2\cos 2\theta_2 - \theta_1\cos 2\theta_1)\\ &+ R_0^2(x^2-y^2)\left(\left(\theta_2^2-\frac{1}{2}\right)\sin 2\theta_2 - \left(\theta_1^2-\frac{1}{2}\right)\sin 2\theta_1\right)\\ &+ 2R_0^2xy(\theta_2\sin 2\theta_2 - \theta_1\sin 2\theta_1)\\ &- R_0^2xy\left((2\theta_2^2-1)\cos 2\theta_2 - (2\theta_1^2-1)\cos 2\theta_1\right)\\ &+ 8R_0(R^2-d^2-R_0^2)x(\theta_2\cos\theta_2-\theta_1\cos\theta_1)\\ &+ 4R_0(R^2-d^2-R_0^2)x\big((\theta_2^2-2)\sin\theta_2 - (\theta_1^2-2)\sin\theta_1\big)\\ &+ 8R_0(R^2-d^2-R_0^2)y(\theta_2\sin\theta_2-\theta_1\sin\theta_1)\\ &- 4R_0(R^2-d^2-R_0^2)y\big((\theta_2^2-2)\cos\theta_2 - (\theta_1^2-2)\cos\theta_1\big)\\ &+ \frac{1}{3}\left(R^2-d^2-R_0^2\right)^2(\theta_2^3-\theta_1^3),\end{aligned} \tag{30}$$

$$\begin{aligned}\frac{R^4}{R_0}F^{t^3}_{\text{arc}}(\boldsymbol{P}) &= \frac{1}{2}R_0^2(x^2+y^2)(\theta_2^4-\theta_1^4)\\ &+ \frac{3}{4}R_0^2(x^2-y^2)\big((2\theta_2^2-1)\cos 2\theta_2 - (2\theta_1^2-1)\cos 2\theta_1\big)\\ &+ \frac{1}{2}R_0^2(x^2-y^2)\big((2\theta_2^3-3\theta_2)\sin 2\theta_2 - (2\theta_1^3-3\theta_1)\sin 2\theta_1\big)\\ &+ \frac{3}{2}R_0^2xy\left((2\theta_2^2-1)\sin 2\theta_2 - (2\theta_1^2-1)\sin 2\theta_1\right)\\ &- R_0^2xy\big((2\theta_2^3-3\theta_2)\cos 2\theta_2 - (2\theta_1^3-3\theta_1)\cos 2\theta_1\big)\\ &+ 4R_0(R^2-d^2-R_0^2)x\big((3\theta_2^2-6)\cos\theta_2 - (3\theta_1^2-6)\cos\theta_1\big)\\ &+ 4R_0(R^2-d^2-R_0^2)x\big((\theta_2^3-6\theta_2)\sin\theta_2 - (\theta_1^3-6\theta_1)\sin\theta_1\big)\\ &+ 4R_0(R^2-d^2-R_0^2)y\big((3\theta_2^2-6)\sin\theta_2 - (3\theta_1^2-6)\sin\theta_1\big)\\ &- 4R_0(R^2-d^2-R_0^2)y\big((\theta_2^3-6\theta_2)\cos\theta_2 - (\theta_1^3-6\theta_1)\cos\theta_1\big)\\ &+ \frac{1}{4}\left(R^2-d^2-R_0^2\right)^2(\theta_2^4-\theta_1^4).\end{aligned} \tag{31}$$

The field functions for the second span $[\theta_3, \theta_4]$ can be calculated similarly.
For an arbitrary arc in space, since the field is coordinate system independent, we may first transform a point $\boldsymbol{P}(x, y, z)$ into the arc's local z-aligned coordinate system and then perform the field computation.

5 Field computation for quadratic curves with varying kernels

Let the quadratic curve primitive be represented as

$$\boldsymbol{Q}(t) = (x(t), y(t), z(t)) = \sum_{i=0}^{2} \boldsymbol{Q}_i t^i, \; 0 \le t \le 1 \quad (32)$$

where $\boldsymbol{Q}_i = (Q_{ix}, Q_{iy}, Q_{iz})$ are vector coefficients. Quadratic curves that are represented in other parametric schemes, such as Bezier or B-spline, can be easily converted into this power basis form.

5.1 Effective-span computation for quadratic curves

As with line segments and arcs, we first use the clipping sphere of $\boldsymbol{P}$ to clip the quadratic curve to obtain the effective span(s) of the curve. To compute the intersection points between the clipping sphere and the quadratic curve, we substitute $\boldsymbol{Q}(t)$ in Eq. (32) into Eq. (7), the clipping sphere equation, to obtain

$$(\boldsymbol{Q}_2 t^2 + \boldsymbol{Q}_1 t + \boldsymbol{Q}_0 - \boldsymbol{P})^2 = R^2. \quad (33)$$

This equation can be converted to a quartic polynomial

$$a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t + a_0 = 0$$

where

$$a_4 = \boldsymbol{Q}_2 \cdot \boldsymbol{Q}_2, \quad a_3 = 2\boldsymbol{Q}_2 \cdot \boldsymbol{Q}_1,$$
$$a_2 = \boldsymbol{Q}_1 \cdot \boldsymbol{Q}_1 + 2\boldsymbol{Q}_2 \cdot (\boldsymbol{Q}_0 - \boldsymbol{P}),$$
$$a_1 = 2\boldsymbol{Q}_1 \cdot (\boldsymbol{Q}_0 - \boldsymbol{P}),$$
$$a_0 = (\boldsymbol{Q}_0 - \boldsymbol{P}) \cdot (\boldsymbol{Q}_0 - \boldsymbol{P}) - R^2.$$

For quartic polynomials, their real roots can be found analytically (Schwarze 1990).
The effective spans of the quadratic curve can be calculated according to the number of roots of the quartic equation:

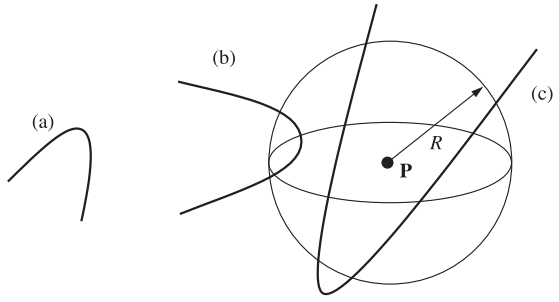


Fig. 7. A clipping sphere and three possible quadratic curves

Case 1: The quartic equation has no real root (curve *a* in Fig. 7). The effective span is empty and the field of the point $\boldsymbol{P}$ is zero.
Case 2: The quartic equation has two real roots (curve *b* in Fig. 7). Let the two roots be $\tilde{t}_1$ and $\tilde{t}_2$ in increasing order. Let $t_1 = \max(0, \tilde{t}_1)$, $t_2 = \min(1, \tilde{t}_2)$, then
if ($t_2 < 0$ or $t_1 > 1$) there is no effective span;
else there is one effective span $[t_1, t_2]$.
Case 3: The quartic equation has four real roots (curve *c* in Fig. 7). Let the four roots be $\tilde{t}_1, \tilde{t}_2, \tilde{t}_3$ and $\tilde{t}_4$ in increasing order. Obviously, the intersecting spans between the quadratic curve and the clipping sphere are $[\tilde{t}_1, \tilde{t}_2]$ and $[\tilde{t}_3, \tilde{t}_4]$. Let $t_1 = \max(0, \tilde{t}_1)$, $t_2 = \min(1, \tilde{t}_2)$, $t_3 = \max(0, \tilde{t}_3)$, $t_4 = \min(1, \tilde{t}_4)$, then
if ($t_4 \le 0$ or $t_1 \ge 1$), there is no effective span;
else if ($t_3 \ge 1$), there is one effective span $[t_1, t_2]$;
else if ($t_2 \le 0$), there is one effective span $[t_3, t_4]$;
else there are two effective spans $[t_1, t_2]$and $[t_3, t_4]$.

5.2 Field computation for quadratic curves

Now we have obtained the effective spans of a quadratic curve contributing to a point $\boldsymbol{P}(x, y, z)$. Let the effective span be $[t_1, t_2]$ if there is only one such span, and be $[t_1, t_2]$and $[t_3, t_4]$ if there are two. The squared distance from $\boldsymbol{P}$ to a point on $\boldsymbol{Q}(t)$ is

$$r^2(t) = (\boldsymbol{Q}_2 t^2 + \boldsymbol{Q}_1 t + \boldsymbol{Q}_0 - \boldsymbol{P})^2$$
$$= a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t + a_0 + R^2. \quad (34)$$

For a parametric curve, we must use the arc length as the integral parameter in the convolution model so as

to guarantee that each point on the curve contributes a correct amount of field to the resulting integral. The differential arc length of a quadratic curve is

$$\begin{aligned}\mathrm{d}s &= \sqrt{(\dot{x}(t))^2+(\dot{y}(t))^2+(\dot{z}(t))^2}\,\mathrm{d}t \\ &= \sqrt{b_2}\sqrt{\left(t+\frac{b_1}{2b_2}\right)^2+\frac{4b_0b_2-b_1^2}{4b_2^2}}\,\mathrm{d}t, \end{aligned} \tag{35}$$

where

$$\begin{aligned} b_2 &= 4\left(Q_{2x}^2+Q_{2y}^2+Q_{2z}^2\right)=4a_4,\\ b_1 &= 4\left(Q_{2x}Q_{1x}+Q_{2y}Q_{1y}+Q_{2z}Q_{1z}\right)=2a_3,\\ b_0 &= \left(Q_{1x}^2+Q_{1y}^2+Q_{1z}^2\right). \end{aligned}$$

Obviously,

$$\begin{aligned} &4b_0b_2-b_1^2\\ &\quad=16\left(Q_{2x}^2+Q_{2y}^2+Q_{2z}^2\right)\left(Q_{1x}^2+Q_{1y}^2+Q_{1z}^2\right)\\ &\qquad-16\left(Q_{2x}Q_{1x}+Q_{2y}Q_{1y}+Q_{2z}Q_{1z}\right)^2\geq 0 \end{aligned}$$

Let $c_0=\sqrt{b_2}$, $c_1=-\frac{b_1}{2b_2}$, $c=\frac{4b_0b_2-b_1^2}{4b_2^2}$, then the differential arc length can be rewritten as

$$\mathrm{d}s = c_0\sqrt{(t-c_1)^2+c}\,\mathrm{d}t,\quad c\geq 0. \tag{36}$$

The quadratic curve's analytical field function for a point $\boldsymbol{P}(x, y, z)$ is then

$$\begin{aligned} F_{\mathrm{curve}}(\boldsymbol{P}) &= \int_0^1\left\{\sum_{i=0}^{3}q_iB_{i,3}(t)\right\}f(r^2(t))\mathrm{d}t\\ &= \sum_{j=1}^{\mathrm{SpanNum}}\sum_{i=0}^{3}\left\{q_i'\frac{c_0}{R^4}\int_{t_{2j-1}}^{t_{2j}}\left(a_4t^4+a_3t^3+a_2t^2\right.\right.\\ &\qquad\left.\left.+a_1t+a_0\right)^2\sqrt{(t-c_1)^2+c}\,\mathrm{d}t\right\}\\ &= \sum_{j=1}^{\mathrm{SpanNum}}\left\{q_0'\ {}_jF_{\mathrm{curve}}^1(\boldsymbol{P})+q_1'\ {}_jF_{\mathrm{curve}}^t(\boldsymbol{P})\right.\\ &\qquad\left.+q_2'\ {}_jF_{\mathrm{curve}}^{t^2}(\boldsymbol{P})+q_3'\ {}_jF_{\mathrm{curve}}^{t^3}(\boldsymbol{P})\right\}, \end{aligned} \tag{37}$$

where

$$\begin{aligned} &q_0'=q_0,\quad q_1'=-3q_0+3q_1,\\ &q_2'=3q_0-6q_1+3q_2,\quad q_3'=-q_0+3q_1-3q_2+q_3, \end{aligned}$$

and ${}_jF_{\mathrm{curve}}^{t^i}(\boldsymbol{P})$, $i=0, 1, 2, 3$ are the field functions of the quadratic curve for span j with weight distribution t^i defined by

$$\begin{aligned} {}_jF_{\mathrm{curve}}^{t^i}(\boldsymbol{P}) &= \frac{c_0}{R^4}\int_{t_{2j-1}}^{t_{2j}}t^i\left(a_4t^4+a_3t^3+a_2t^2\right.\\ &\qquad\left.+a_1t+a_0\right)^2\sqrt{(t-c_1)^2+c}\,\mathrm{d}t. \end{aligned} \tag{38}$$

For the first effective span $[t_1, t_2]$, we obtain

$$\begin{aligned} F_{\mathrm{curve}}^1(\boldsymbol{P}) &= \frac{c_0}{R^4}\int_{t_1}^{t_2}\left(a_4t^4+a_3t^3+a_2t^2+a_1t+a_0\right)^2\\ &\qquad\times\sqrt{(t-c_1)^2+c}\,\mathrm{d}t\\ &= C\sum_{i=0}^{8}A_i\left(\int_{t_1-c_1}^{t_2-c_1}t^i\sqrt{t^2+c}\,\mathrm{d}t\right), \end{aligned} \tag{39}$$

where

$$\begin{aligned} &C=\frac{c_0}{R^4},\\ &A_8=a_4^2,\\ &A_7=0,\\ &A_6=-12a_4^2c_1^2+2a_4a_2,\\ &A_5=-16a_4^2c_1^3+4a_4a_2c_1+2a_4a_1,\\ &A_4=30a_4^2c_1^4-10a_4a_2c_1^2+2a_4a_1c_1+(a_2^2+2a_4a_0),\\ &A_3=96a_4^2c_1^5-40a_4a_2c_1^3-12a_4a_1c_1^2\\ &\qquad+4a_2^2c_1+2a_2a_1,\\ &A_2=100a_4^2c_1^6-50a_4a_2c_1^4-28a_4a_1c_1^3\\ &\qquad+6(a_2^2-2a_4a_0)c_1^2+6a_2a_1c_1+(a_1^2+2a_2a_0),\\ &A_1=48a_4^2c_1^7-28a_4a_2c_1^5-22a_4a_1c_1^4\\ &\qquad+4(a_2^2-4a_4a_0)c_1^3+6a_2a_1c_1^2\\ &\qquad+2(a_1^2+2a_2a_0)c_1+2a_1a_0,\\ &A_0=9a_4^2c_1^8-6a_4a_2c_1^6-6a_4a_1c_1^5+(a_2^2-6a_4a_0)c_1^4\\ &\qquad+2a_2a_1c_1^3+(a_1^2+2a_2a_0)c_1^2+2a_1a_0c_1+a_0^2. \end{aligned}$$

Let $l_1=t_1-c$, $l_2=t_2-c$, and

$$I_n=\int_{l_1}^{l_2}t^n\sqrt{t^2+c}\,\mathrm{d}t,$$

then we obtain

$$F^1_{\text{curve}}(\boldsymbol{P}) = C\sum_{i=0}^{8} A_i I_i, \tag{40}$$

$$F^t_{\text{curve}}(\boldsymbol{P}) = C\left\{\sum_{i=0}^{8} A_i I_{i+1} + c_1\sum_{i=0}^{8} A_i I_i\right\}, \tag{41}$$

$$F^{t^2}_{\text{curve}}(\boldsymbol{P})\mathrm{d}t = C\left\{\sum_{i=0}^{8} A_i I_{i+2} + 2c_1\sum_{i=0}^{8} A_i I_{i+1} + c_1^2\sum_{i=0}^{8} A_i I_i\right\}, \tag{42}$$

$$F^{t^3}_{\text{curve}}(\boldsymbol{P}) = C\left\{\sum_{i=0}^{8} A_i I_{i+3} + 3c_1\sum_{i=0}^{8} A_i I_{i+2} + 3c_1^2\sum_{i=0}^{8} A_i I_{i+1} + c_1^3\sum_{i=0}^{8} A_i I_i\right\}. \tag{43}$$

Let

$$J = \int_{l_1}^{l_2} \frac{\mathrm{d}t}{\sqrt{t^2+c}}.$$

By applying integration techniques, we obtain

$$J = \ln\left|\frac{l_2+\sqrt{l_2^2+c}}{l_1+\sqrt{l_1^2+c}}\right|, \tag{44}$$

$$\begin{aligned} I_0 &= \int_{l_1}^{l_2} \sqrt{t^2+c}\,\mathrm{d}t \\ &= \frac{1}{2}\left(l_2\sqrt{l_2^2+c} - l_1\sqrt{l_1^2+c}\right) + \frac{1}{2}cJ, \end{aligned} \tag{45}$$

$$\begin{aligned} I_1 &= \int_{l_1}^{l_2} t\sqrt{t^2+c}\,\mathrm{d}t \\ &= \frac{1}{3}\left[(l_2^2+c)\sqrt{l_2^2+c} - (l_1^2+c)\sqrt{l_1^2+c}\right]. \end{aligned} \tag{46}$$

When $n \geq 2$, since we have

$$\begin{aligned} I_n &= \int_{l_1}^{l_2} t^{n-2}(t^2+c)\sqrt{t^2+c}\,\mathrm{d}t - c\int_{l_1}^{l_2} t^{n-2}\sqrt{t^2+c}\,\mathrm{d}t \\ &= \frac{1}{n-1}\int_{l_1}^{l_2} (t^2+c)\sqrt{t^2+c}\,\mathrm{d}t^{n-1} - cI_{n-2} \\ &= \frac{t^{n-1}}{n-1}(t^2+c)\sqrt{t^2+c}\Big|_{l_1}^{l_2} \\ &\quad - \frac{3}{n-1}\int t^n\sqrt{t^2+c}\,\mathrm{d}t - cI_{n-2} \\ &= \frac{1}{n-1}\left[l_2^{n-1}(l_2^2+c)\sqrt{l_2^2+c}\right. \\ &\quad \left. - l_1^{n-1}(l_1^2+c)\sqrt{l_1^2+c}\right] - \frac{3}{n-1}I_n - cI_{n-2}, \end{aligned}$$

we obtain the following recursive formula:

$$\begin{aligned} I_n = \frac{1}{n+2}&\left[l_2^{n-1}(l_2^2+c)\sqrt{l_2^2+c}\right. \\ &\left. - l_1^{n-1}(l_1^2+c)\sqrt{l_1^2+c}\right] - \frac{n-1}{n+2}cI_{n-2}. \end{aligned} \tag{47}$$

From I_0, I_1, we can recursively calculate the values of $I_2, I_3, \ldots, I_{11}$, and compute $F^1_{\text{curve}}(\boldsymbol{P})$, $F^t_{\text{curve}}(\boldsymbol{P})$, $F^{t^2}_{\text{curve}}(\boldsymbol{P})$ and $F^{t^3}_{\text{curve}}(\boldsymbol{P})$.

As the arc length function $s(t)$ is not linear in t, the presented model is not ideal. However, our experiments show that this model still produces good results.

5.3 Field computation for quadratic spline curves

Without loss of generality, let the quadratic spline curve be a B-spline curve. B-spline is a versatile tool for designing curves with local control. It is a vector-valued piecewise polynomial function of the form

$$\boldsymbol{C}(t) = \sum_{i=0}^{L+1} \boldsymbol{P}_i N_i^p(t), \tag{48}$$

where L is the number of polynomial segments in the curve, $\boldsymbol{P}_i$ are the control points, and N_i^p are the normalized B-spline basis function of degree p defined over the knot vector $\{t_0, t_1, \cdots, t_{L+2p}\}$ (Piegl 1999). We assume a clamped knot vector so that the curve interpolates the endpoints.

Since a quadratic B-spline curve is in fact a piecewise quadratic curve, we can convert each quadratic

curve segment into the form in Eq. (32). We first convert the B-spline curve into the Bezier form, and then compute the following for each quadratic Bezier curve with control points $\boldsymbol{P}_0$, $\boldsymbol{P}_1$, $\boldsymbol{P}_2$:

$$\boldsymbol{Q}_0 = (\boldsymbol{P}_0 - 2\boldsymbol{P}_1 + \boldsymbol{P}_2),\ \boldsymbol{Q}_1 = 2(\boldsymbol{P}_1 - \boldsymbol{P}_0),\ \boldsymbol{Q}_2 = \boldsymbol{P}_0.$$

To compute the control points of the weight distribution curve for each quadratic curve segment, we need the arc length at the junction points of the B-spline curve. The arc length of a quadratic curve segment is

$$\begin{aligned} l &= c_0 \int_{-c_1}^{1.0-c_1} \sqrt{t^2 + c}\,\mathrm{d}t \\ &= \frac{1}{2}c_0\left[(1-c_1)\sqrt{(1-c_1)^2 + c} + c_1\sqrt{c_1^2 + c}\right] \\ &\quad + \frac{1}{2}c_0 c \ln\left|\frac{(1-c_1)\sqrt{(1-c_1)^2 + c}}{-c_1 + \sqrt{c_1^2 + c}}\right|. \end{aligned} \tag{49}$$

Let $l_i, i = 0, 1, \cdots, L-1$ denote the arc length of the ith quadratic segment, then the normalized cumulative arc lengths are

$$u_0 = 0,\ u_i = u_{i-1} + \left(\sum_{j=0}^{i-1} l_j \Big/ \sum_{j=0}^{L-1} l_j\right).$$

The control points of the weight distribution curve for the ith quadratic segment can then be computed using the Bezier subdivision algorithm over interval $[u_i, u_{i+1}]$.

6 Field computation for planar higher-degree polynomial spline curves

We have presented analytical convolution models for line segments, arcs, and quadratic spline curves; we now consider planar higher-degree polynomial spline curves. Motivated by research on biarc curve fitting, we employ an optimization approach to approximate any given planar polynomial parametric curves, represented in Bezier, B-spline, NURBS, or other parametric form, by an arc spline with fewest segments and within a prescribed tolerance. An arc spline is a G^1 continuous curve consisting of arcs and line segments. Since the field function of an arc and a line segment can be derived analytically, the field function for the arc spline can be obtained by summing the potential functions of all skeletal primitives. As most well-behaved planar spline curves can be approximated by only a few arcs and line segments, our algorithm runs quite efficiently.

To approximate a planar parametric spline by an arc spline, we adopt the method proposed by Yang (2000), which constructs an arc spline via optimizing a biarc spline that interpolates sample points on the given curve. That is, his scheme consists of two steps:

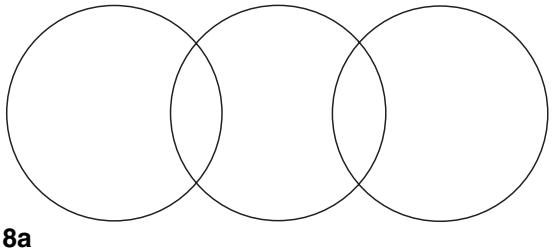
First, sample and interpolate the given spline curve by biarcs to within a tolerance τ_1, which is usually smaller than 10^{-6} and is insignificant.

Iteratively merge consecutive arc splines into optimal arc splines with fewest segments and within tolerance τ_2. The total deviation is $\tau_1 + \tau_2$.

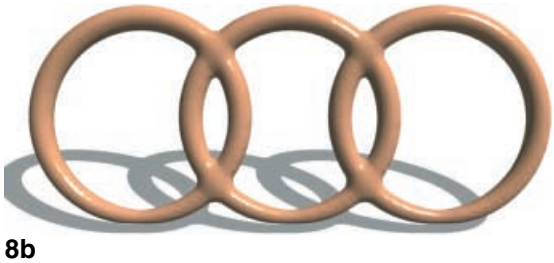
The method has several advantages over other existing arc spline approximation methods. Firstly, it can approximate any type of planar parametric curves. Secondly, constructing the interpolating biarcs by sampling the original curve is efficient and the error is controllable. Finally, the number of segments is smaller compared with other existing methods and is proven to be near optimal within the prescribed tolerance. Since the arc length of each arc can be calculated trivially, the control points of the weight distribution curve for each arc can be calculated by Bezier subdivision algorithm using the normalized arc length parameter.

7 Results

We have implemented our algorithm on a Pentium III 400E PC with 128 M main memory. When calculating the fields, optimizations have been performed to reduce the amount of computation. For example, for the arc primitive, after $\sin\theta$ and $\cos\theta$ have been calculated, $\sin 2\theta$ and $\cos 2\theta$ are computed using only multiplication, addition, and subtraction operations based on trigonometric principles. We show some rendered modeling examples to demonstrate the capabilities of our method. For uniform processing, all the convolution models are first polygonized into

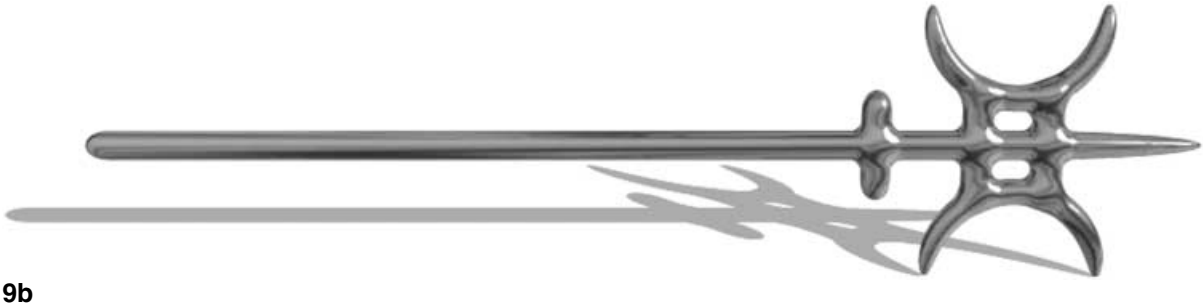


8a



8b

9a



9b

Fig. 8a,b. Blending tori: **a** control skeleton; **b** convolution surface
Fig. 9a,b. Chinese old weapon "ji": **a** control skeleton; **b** convolution surface

polygon meshes (Bloomenthal 1988; Bloomenthal 1994), which are then ray-traced.

Figures 8–10 are examples of convolution surfaces using only line segments and arcs as the skeletons, and Figs. 11–14 are examples that include quadratic B-spline curves (in addition to line segments and arcs) in their skeletons. The number of line segments, arcs, and quadratic curve segments in these examples are listed in Table 1. The table also shows the field computation cost for polygonizing each convolution surface; specifically, it gives the time cost for each type of skeletal primitive, the total field computation time, and the number of triangles generated in the implicitization. The results show that the field for these skeletal primitives can be competitively computed.

As an example of modeling with planar cubic spline curves, we model the characters "CAD&CG" in Fig. 15. The letter *A* uses four line segments, and the other letters use planar cubic NURBS curves which are converted to arc splines. The numbers of arcs and line segments in the approximate arc splines are shown in Table 2, together with the field computation timing results.

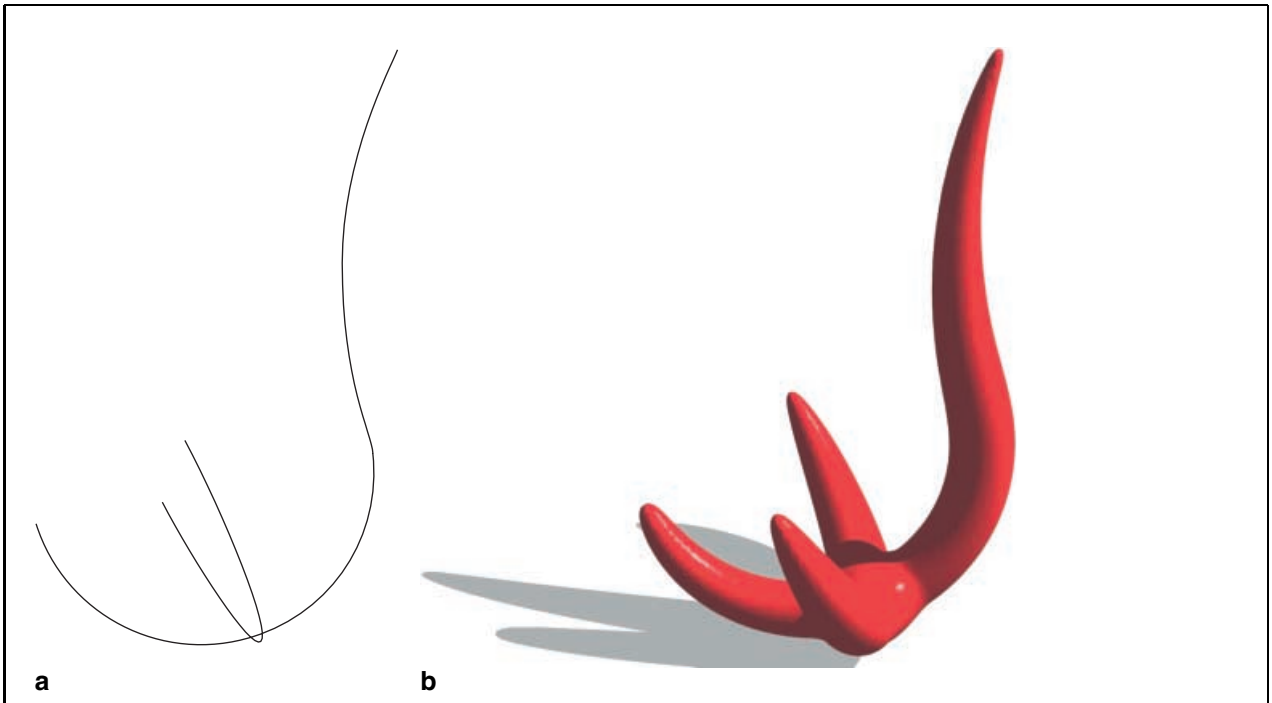


Fig. 10a,b. The red-bird logo of Hong Kong University of Science and Technology: **a** control skeleton; **b** convolution surface

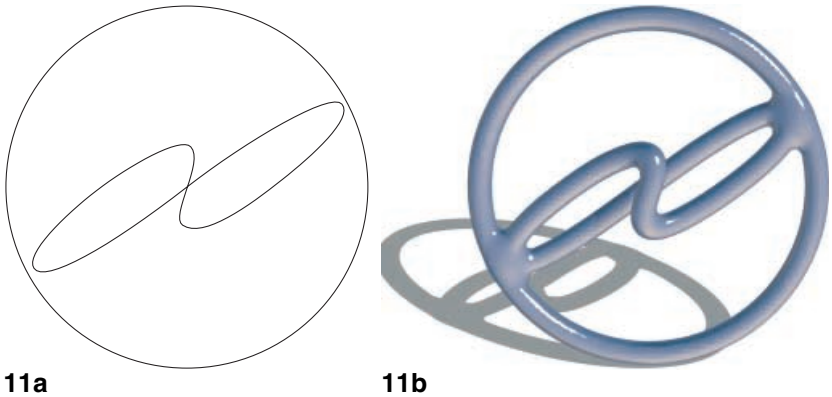
Table 1. Field computation time for polygonizing the convolution surfaces in Figs. 8–14

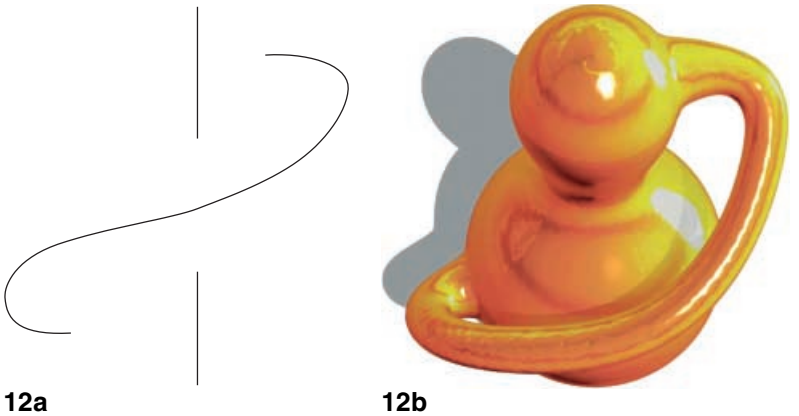
Model	Lines		Arcs		Quadratic curves		Total time (s)	Number of Triangles
	No.	Time (s)	No.	Time (s)	No.	Time (s)		
Blending tori	–	–	3	2.304	–	–	2.304	132 144
"ji" weapon	5	0.922	2	1.223	–	–	2.145	140 220
Red-bird logo	–	–	4	3.373	–	–	3.373	132 888
Art shape	–	–	1	1.121	11	16.273	17.394	138 048
Gourd shape	2	1.422	–	–	5	7.683	9.105	148 620
Dinosaur	15	1.741	–	–	6	9.627	11.368	158 796
Enforcer	12	1.432	–	–	10	15.639	17.071	156 132

8 Conclusions and future work

Curve-skeleton-based convolution surfaces are useful for modeling many objects, such as plants, logo characters, and sea-life forms. We have presented some efficient analytical convolution surface models for arc and quadratic spline curves skeletons with polynomial-weighted distributions. For planar higher-degree parametric spline curves, rather than directly calculating the convolution surface integral for the curve skeleton, we construct an approximate GC^1 arc spline by optimizing a biarc spline that interpolates sample points on the given curve. The field of each arc or line segment in the arc spline can then be calculated analytically and summed. Since non-uniform tapering effects and varying-radius tubular shapes are prevalent in many organic shapes, the exact evaluation of our model provides an effective solution to model these shapes. Although the span computation requires lots of efforts for the finite-support polynomial kernel, our study shows that such computation is worthwhile as the closed-form solutions for arcs and quadratic curves using varying infinite-support kernels seems impossible.

Since curve skeletons are good abstractions for a wide variety of natural forms, our method is considerably general in its applicability. Experimen-





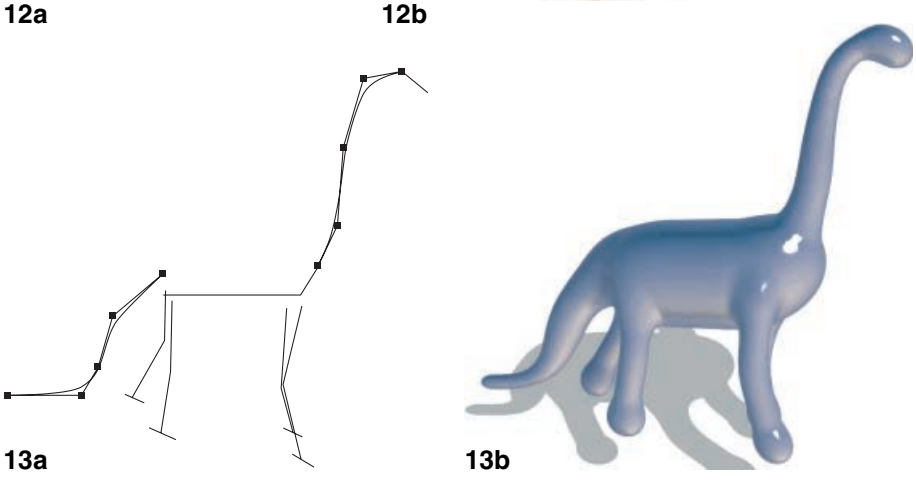
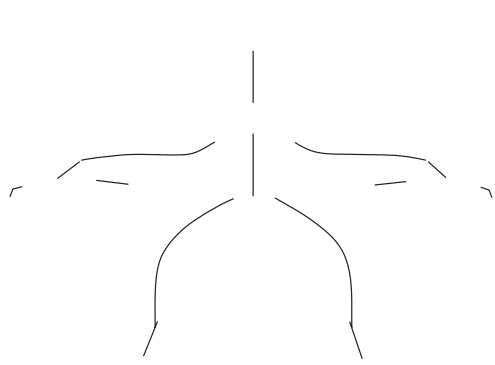


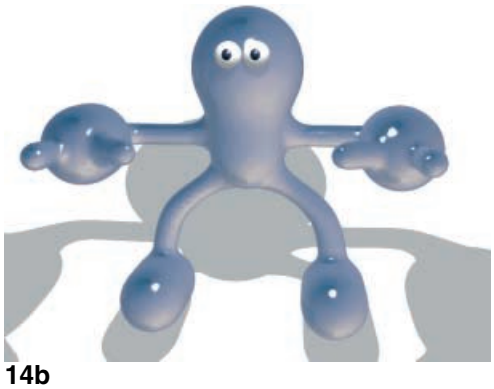
Fig. 11a,b. An art shape: **a** control skeleton; **b** convolution surface
Fig. 12a,b. A gourd shape: **a** control skeleton; **b** convolution surface
Fig. 13a,b. A dinosaur: **a** control skeleton; **b** convolution surface

tal results demonstrate that our method can create many aesthetically pleasing branching effects and possesses potential applications in both geometric modeling and computer animation.

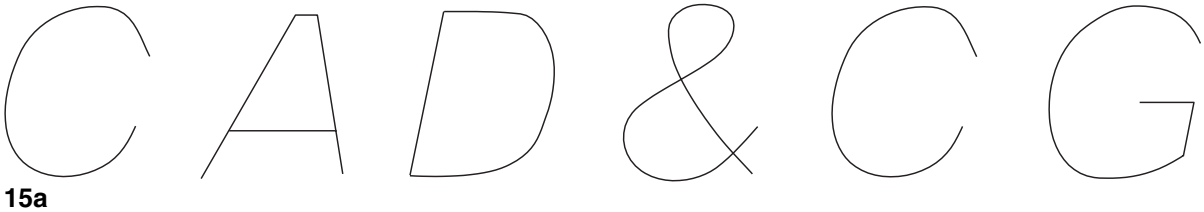
The analytical solutions for higher-degree polynomial curves without using arc spline approximation still remains an open problem. Our attempts do not produce closed-form solutions. To find an efficient



14a



14b



15a

15b

Fig. 14a,b. An enforcer: **a** control skeleton; **b** convolution surface
Fig. 15a,b. Characters "CAD&CG": **a** control skeleton; **b** convolution surface

Table 2. Field computation timing results for "CAD&CG"

Model	Arc spline approximation time (s)	Lines		Arcs		Total time (s)	Number of Triangles
		No.	Time (s)	No.	Time (s)		
C	0.102	–	–	11	2.114	2.114	36 456
A	–	4	0.420	–	–	0.42	36 504
D	0.085	1	0.190	7	1.472	1.662	42 540
&	0.121	2	0.040	14	2.520	2.560	42 828
G	0.109	2	0.201	8	1.761	1.962	44 028

method to approximate an arbitrary space polynomial curve by an arc spline to within a prescribed tolerance appears to be an interesting future research direction. The possibility of using degree reduction algorithms to convert high-degree B-spline curves to quadratic B-spline curves within prescribed tolerance for field computation may be another interesting alternative.

Acknowledgements. Part of this research work was conducted while the first author was a visiting researcher at the Hong Kong University of Science and Technology. The authors are grateful to Andrei Sherstyuk for his help on convolution surfaces, Dr. Xunnian Yang for his help on

arc spline approximation, and the anonymous reviewers for their constructive suggestions. This work received support from the Hong Kong Research Grant Council (HKUST6215/99E), National Natural Science Foundation of China (grant no. 69973040), Zhejiang Provincial Natural Science Foundation (grant no. 698022), and Innovative Research Groups (60021201).

References

1. Attali D, Montanvert A (1997) Computing and simplifying 2D and 3D semi-continuous skeletons of 2D and 3D shapes. Comput Vision Image Understanding 67(3):261–273
2. Blanc C, Schlick C (1995) Extended field functions for soft objects. In: Proceedings of implicit surfaces '95, Grenoble, France, pp 21–32
3. Bloomenthal J (1988) Polygonization of implicit surfaces. Comput Aided Geom Des 5(4):341–355
4. Bloomenthal J (1994) An implicit surface polygonizer. In: Heckbert P (ed) Graphics gems IV. Academic Press, pp 324–349
5. Bloomenthal J (1995) Skeletal design of natural forms. Ph.D Thesis, Department of Computer Science, University of Calgary
6. Bloomenthal J (1997) Bulge elimination in convolution surfaces. Comput Graph Forum 16(1):31–41
7. Bloomenthal J, Bajaj C, Blinn J, Cani-Gascuel M, Rockwood A, Wyvill B, Wyvill G (1997) An introduction to implicit surfaces. Morgan Kaufmann, Los Altos
8. Bloomenthal J, Shoemake K (1991) Convolution surfaces. In: SIGGRAPH '91 conference proceedings. Computer graphics annual conference series, pp 251–256
9. Bloomenthal J, Wyvill B (1990) Interactive techniques for implicit modeling. Comput Graph 24(2):109–116
10. Cani-Gascuel M, Desbrun M (1997) Animation of deformable models using implicit surfaces. IEEE Trans Visual Comput Graph 3(1):39–50
11. Dobashi Y, Kaneda K, Yamashita H, Okita T, Nishita T (2000) A simple, effcient method for realistic animation of clouds. In: SIGGRAPH '00 conference proceedings. Computer graphics annual conference series, pp 19–28
12. Farin G (1997) Curves and surfaces for computer aided geometric design: a practical guide, 4th edn. Academic Press, New York
13. Ferley E, Cani-Gascuel M, Attali D (1997) Skeletal reconstruction of branching shapes. Comput Graph Forum 16(5):283–293
14. Jin X, Li Y, Peng Q (2000) General constrained deformation based on generalized metaballs. Comput Graph 24(2):219–231
15. McCormack J, Sherstyuk A (1998) Creating and rendering convolution surfaces. Comput Graph Forum 17(2):113–120
16. Nishita T, Iwasaki H, Dobashi Y, Nakamae E (1997) A modeling and rendering method for snow by using metaballs. Comput Graph Forum 16(3):357–364
17. Pasko A, Adzhiev V, Sourin A, Savchenko V (1995) Function representation in geometric modeling: concepts, implementation and applications. Vis Comput 11(8):429–446
18. Piegl L (1999) On NURBS: a survey. IEEE Comput Graph Appl 11(1):55–71
19. Savchenko V, Pasko A, Okunev O, Kunii T (1995) Function representation of solids reconstructed from scattered surface points and contours. Comput Graph Forum 14(4):181–188
20. Schwarze J (1990) Cubic and quartic roots. In: Glassner A (ed) Graphics gems. Academic Press, pp 404–406
21. Sherstyuk A (1999a) Kernel functions in convolution surfaces: a comparative analysis. Vis Comput 15(4):171–182
22. Sherstyuk A (1999b) Convolution surfaces in computer graphics. PhD Thesis, School of Computer Science and Software Engineering, Monash University, Australia
23. Turk G, O'Brien J (1999) Shape transformation using variational implicit functions. In: SIGGRAPH '99 conference proceedings. Computer graphics annual conference series, pp 335–342
24. Wyvill B, Wyvill G (1989) Field functions for implicit surfaces. Vis Comput 5(1/2):75–82
25. Wyvill B, Galin E, Guy A (1999) Extending the CSG tree: warping, blending and boolean operations in an implicit surface modeling system. Comput Graph Forum 18(2):149–158
26. Yang X (2000) Approximating NURBS curves by arc splines. In: Proceedings of geometric modeling and processing '00, Hong Kong, pp 10–12

XIAOGANG JIN is a professor of the State Key Lab of CAD&CG, Zhejiang University, People's Republic of China. He received his BSc degree in Computer Science in 1989, MSc and PhD degrees in Applied Mathematics in 1992 and 1995, all from Zhejiang University. His research interests include implicit surface modeling, space deformation, computer animation and realistic image synthesis.

CHIEW-LAN TAI is an assistant professor in the Department of Computer Science, Hong Kong University of Science & Technology. She received her BSc and MSc in mathematics from the University of Malaya, and her MSc in computer and information sciences from the National University of Singapore. She earned her DSc in information science from the University of Tokyo in 1997. Her research interests include geometric modeling, computer graphics, and interpretation of engineering drawings.