PERGAMON

Computers & Graphics 26 (2002) 437–447

COMPUTERS & GRAPHICS

www.elsevier.com/locate/cag

Technical Section

Analytical methods for polynomial weighted convolution surfaces with various kernels

Xiaogang Jin[a,*], Chiew-Lan Tai[b]

[a] *State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou 310027, People's Republic of China*

[b] *Department of Computer Science, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong*

Abstract

Convolution surface has the advantage of being crease-free and bulge-free over other kinds of implicit surfaces. Among the various types of skeletal elements, line segments can be considered one of the most fundamental as they can approximate curve skeletons. This paper presents analytical solutions for convolving line segments with varying kernels modulated by polynomial weighted functions. We derive the closed-form formulae for most classical kernel functions, namely Gaussian, inverse linear, inverse squared, Cauchy, and quartic functions, and compare their computational complexity. These analytical solutions can be incorporated into existing implicit surface modeling systems for more convenient modeling of generalized cylindrical shapes. We demonstrate their high potentials for modeling and animating branching and tubular organic objects with some examples. We also propose a new competitive kernel function that has a smoothness control parameter. © 2002 Elsevier Science Ltd. All rights reserved.

Keywords: Convolution surface; Polynomial weighted distribution; Implicit surface; Analytical solution

1. Introduction

Geometric objects are often modeled using parametric surfaces and polygon meshes; however, many smooth deformable objects with complex and time-varying topology are more conveniently modeled as field-based implicit surfaces. Examples of such objects are liquids, clouds, tree branches and other organic shapes. Consequently, implicit surfaces have gained acceptance in shape morphing, surface reconstruction, natural phenomena simulation, and space deformation [1–7].

Metaballs, distance surfaces, convolution surfaces, and variational surfaces are some well-known types of implicit surfaces. Metaballs (or blobs, soft objects) [8–11] are defined in terms of point fields, and are widely implemented in commercial modeling packages (e.g., Softimage, 3D Studio Max) and supported by many ray-tracing software (e.g., *Rayshade*, *POV-Ray*). Nevertheless, point-based field surfaces suffer from some drawbacks: flat surfaces have to be approximated by many closely packed metaballs to avoid bumps, causing expensive computation; curve skeletons, which naturally abstract many shapes, have to be converted to points, causing incompact representations. Distance surface offers a solution to this problem by generalizing point-based skeletons to higher dimensional ones [12–13]. Unfortunately, distance surfaces have a major weakness; wherever a skeleton is not convex, the surface may have bulges, creases and curvature discontinuity [14].

Bloomenthal and Shoemake presented convolution surfaces [15] (Fig. 1) as a natural and powerful extension to point-based field surfaces to include line segments, curves, polygons as skeletal elements. By convolving these skeletons with a three-dimensional (3D) low-pass Gaussian filter kernel, convolution surfaces overcome the problem of bulges and curvature discontinuity in distance surfaces. Their other desirable advantages include intuitive shape design, well-behaved blending and fluid topology changes in accordance with the underlying skeleton. Computer vision research has shown that any 3D object can be defined entirely from

*Corresponding author. Tel.: +86-571-879-51045; fax: +86-571-879-51780.

E-mail addresses: jin@cad.zju.edu.cn (X. Jin), taicl@cs.ust.hk (C.-L. Tai).

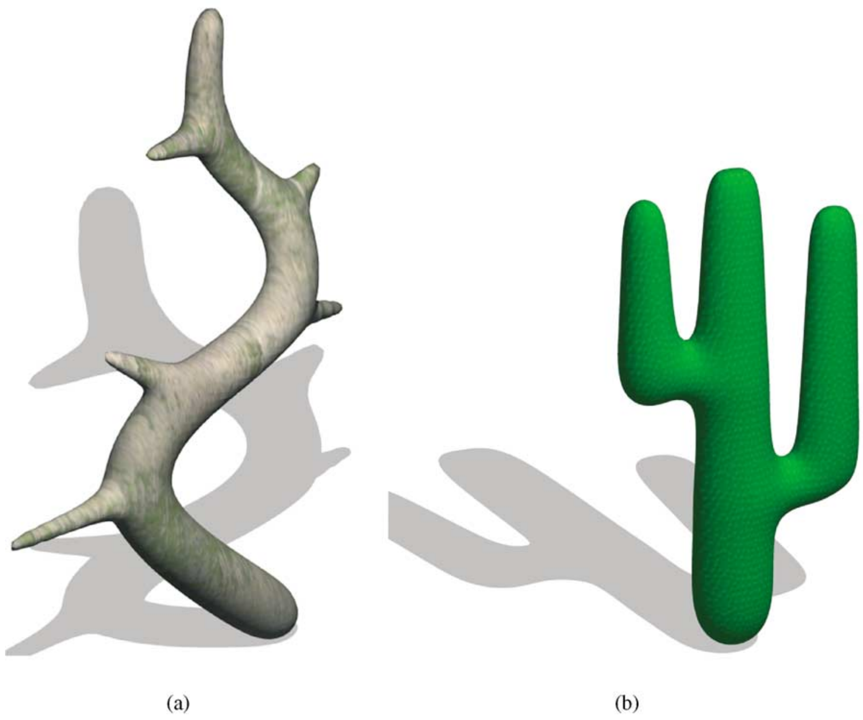

Fig. 1. Convolution surface modeling examples: (a) the trunk of a potted plant, (b) a cactus.

a geometric skeleton [16]; that is, skeletons are natural abstractions for 3D objects. Convolution surfaces also offer a means of controlling the shape of an underlying modeling object by controlling its skeleton, which is easier to manipulate due to its lower dimension.

While the modeling potentials of convolution surfaces are very attractive, their mathematical formulations still pose some open problems, stemming from the fact that convolution integrals seldom yield closed-form solutions. There are limited choices of kernel functions and skeletal primitives that can be convolved together analytically. By using the superposition property of convolution surface and separable property of the Gaussian filter, Bloomenthal and Shoemake calculated the field function numerically based on point-sampling method [15], which unfortunately suffers from potential under-sampling artifacts and large storage. The existence of closed-form solutions depends on both the skeletal element and the kernel function. By employing a kernel function called Cauchy function McCormack and Sherstyuk deduced analytical solutions for points, line segments, polygons, arcs and planes [17–19].

In convolution surface modeling, line segments can be viewed as one of the most fundamental ones because many objects can be abstracted into curve skeletons, and curve skeletons can in turn be subdivided into line segments. The analytical model for line-segment primitive derived by McCormack and Sherstyuk treats the weight distribution along the skeleton uniformly, thus modeling tapering or generalized cylindrical shapes requires specifying multiple line segments. Unlike generalized cylindrical distance surfaces, which can be produced by simply varying the distance in the field computation, this approach fails for generalized cylindrical convolution surfaces [20].

In an earlier work, we have presented an analytical solution for line-segment skeletons convolved with the Cauchy function modulated by polynomial weighted distributions [21]. This paper further proves that analytical solutions for line-segment skeletons with polynomial weighted distributions can be derived for most other classical kernel functions, namely Gaussian, inverse linear, inverse squared, inverse cubic, inverse quintic, and quartic polynomial functions. In addition, we propose a new competitive kernel function that has a smoothness control parameter and derive the corresponding analytical solutions. Our model can also be applied to curve skeletons by subdividing the skeletons into polylines as well as subdividing the given polynomial weighted distribution. With the analytical formulae derived for all the commonly used kernel functions of implicit surfaces, our method can be incorporated into any software.

2. Convolution surfaces

A convolution surface is determined by a skeleton consisting of 3D points, each of which contributes to the field function according to its distance to a space point in question. Let $\mathbf{P}(x, y, z)$ be a space point in R^3, and let $g : R^3 \to R$ be a tri-variate *geometry function* that represents a modeling skeleton V:

$$g(\mathbf{P}) = \begin{cases} 1, & \mathbf{P} \in \text{skeleton } V, \\ 0 & \text{otherwise.} \end{cases}$$

Let $f : R^3 \to R$ be a potential function that describes the field generated by a single point in the skeleton, and $\mathbf{Q}$ be a point in the skeleton V, then the total field contributed by the skeleton at a point $\mathbf{P}$ is the convolution of functions f and g, as defined by McCormack and Sherstyuk [17]

$$F(\mathbf{P}) = \int_V g(\mathbf{Q}) f(\mathbf{P} - \mathbf{Q})\, \mathrm{d}V.$$

Thus, f is also called the *convolution kernel*. The geometry function gives the distribution of the skeleton, and the field function describes the potential of the skeleton at a specific position.

One of the most important properties of convolution surfaces is superposition, which means summing the convolution surfaces generated by two separate skeletons yields the same surface as that generated by their combined skeleton. Therefore, when designing a convolution surface, the user only has to be concerned with the shape of the skeleton; the independent evaluation property ensures that the skeletons can be arbitrarily subdivided and the field function of the sub-skeletons can be simply summed to evaluate the final convolution surface.

3. Polynomial field computation for line segments with various kernel functions

A line segment $\mathbf{L}(t)$ of length l, with start point $\mathbf{b}$ and unit direction $\mathbf{n}$, can be represented parametrically as

$$\mathbf{L}(t) = \mathbf{b} + t\mathbf{n}, \quad 0 \leqslant t \leqslant l.$$

The squared distance from a point $\mathbf{P}$ to a point on the line $\mathbf{L}(t)$ is then

$$r^2(t) = d^2 + t^2 - 2t\mathbf{d} \bullet \mathbf{n} = (t - h)^2 + (d^2 - h^2),$$

where $\mathbf{d} = \mathbf{P} - \mathbf{L}(0)$, $d = \|\mathbf{d}\|$, and $h = \mathbf{d} \bullet \mathbf{n}$.

The idea of using weight functions in defining convolution surfaces was introduced in [15], but no closed-form solutions were developed. Our primary purpose is to develop analytical convolution surface solutions for line-segment skeletons with polynomial weighted distributions. Let $F^{t^i}_{\text{line}}(\mathbf{P})$ denote the field function of the line segment $\mathbf{L}(t)$ with weight distribution t^i. For $i = 0, 1, 2, 3$, we can obtain the following:

$$F^1_{\text{line}}(\mathbf{P}) = \int_0^l f(r)\, \mathrm{d}t, \quad F^t_{\text{line}}(\mathbf{P}) = \int_0^l t f(r)\, \mathrm{d}t$$

$$F^{t^2}_{\text{line}}(\mathbf{P}) = \int_0^l t^2 f(r)\, \mathrm{d}t, \quad F^{t^3}_{\text{line}}(\mathbf{P}) = \int_0^l t^3 f(r)\, \mathrm{d}t.$$

3.1. Gaussian function

Gaussian function is one of the earliest used field functions in implicit surface modeling. It is given by

$$f(r) = \mathrm{e}^{-a^2 r^2}, \quad r > 0,$$

where the parameter a is related to the standard deviation of the function and is used to control the degree of decay of the field function. Gaussian function is attributed to Jim Blinn [8], and was used as the kernel function of convolution surfaces by Bloomenthal and Shoemake [15]. It is also used by Muraki to fit simple blobby primitives to the range data [22]. From the point of view of signal processing, Gaussian function is a low-pass filter imposing on skeletons.

To obtain the field function of a line segment with constant distribution, we substitute $r^2(t)$ into $F^1_{\text{line}}(\mathbf{P})$ and integrate

$$\begin{aligned} F^1_{\text{line}}(\mathbf{P}) &= \frac{1}{a} \mathrm{e}^{-a^2(d^2 - h^2)} \int_{-ah}^{a(l-h)} \mathrm{e}^{-t^2}\, \mathrm{d}t \\ &= \frac{\sqrt{\pi}}{2a} \mathrm{e}^{-a^2(d^2 - h^2)} [\operatorname{sgn}(l - h) \operatorname{erf}(a|l - h|) \\ &\quad + \operatorname{sgn}(h) \operatorname{erf}(a|h|)], \end{aligned}$$

where $\operatorname{erf}(x)$ returns the error function integrated between 0 and x,

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \mathrm{e}^{-t^2}\, \mathrm{d}t, \quad x > 0,$$

and $\operatorname{sgn}(x)$ is a sign function defined as

$$\operatorname{sgn}(x) = \begin{cases} 1, & x > 0, \\ 0, & x = 0, \\ -1, & x < 0. \end{cases}$$

Similarly, by applying integration techniques, we can deduce the following formulae for line segments with weight distributions t, t^2, and t^3, respectively:

$$F^t_{\text{line}}(\mathbf{P}) = hF^1_{\text{line}}(\mathbf{P}) + \frac{1}{2a^2} \mathrm{e}^{-a^2(d^2 - h^2)} [\mathrm{e}^{-a^2 h^2} - \mathrm{e}^{-a^2(l-h)^2}],$$

$$\begin{aligned} F^{t^2}_{\text{line}}(\mathbf{P}) &= 2hF^t_{\text{line}}(\mathbf{P}) + \left(\frac{1}{2a^2} - h^2\right) F^1_{\text{line}}(\mathbf{P}) \\ &\quad - \frac{1}{2a^2} \mathrm{e}^{-a^2(d^2 - h^2)} [(l - h)\mathrm{e}^{-a^2(l-h)^2} + h\mathrm{e}^{-a^2 h^2}], \end{aligned}$$

$$F^{t^3}_{\text{line}}(\mathbf{P}) = 3hF^{t^2}_{\text{line}}(\mathbf{P}) - 3h^2F^{t}_{\text{line}}(\mathbf{P}) + h^3F^{1}_{\text{line}}(\mathbf{P}) - \frac{1}{2a^4}e^{-a^2(d^2-h^2)}[(a^2(l-h)^2+1)e^{-a^2(l-h)^2} - (a^2h^2+1)e^{-a^2h^2}].$$

3.2. Inverse linear function

The inverse linear function is given by

$f(r) = 1/r, \quad r>0$

and was used in implicit surface modeling by Wyvill and van Overveld [23].

When modeling implicit surfaces using the inverse type of functions, the point at $r=0$ causes the division-by-zero error. In order to clip the singular point, we calculate the squared distance from the point $\mathbf{P}$ to the closest point on the line segment $\mathbf{L}(t)$ as follows:

$$r^2 = \begin{cases} d^2, & h<0, \\ (\mathbf{d}-h\mathbf{n})\bullet(\mathbf{d}-h\mathbf{n}), & 0\leqslant h \leqslant l, \\ (\mathbf{d}-l\mathbf{n})\bullet(\mathbf{d}-l\mathbf{n}), & h>l. \end{cases}$$

If $r^2<\varepsilon$, where ε is a specified small number ($\varepsilon = 10^{-6}$), we set the potential function of the point $\mathbf{P}$ to be a very large value.

Let $m = \sqrt{d^2-2hl+l^2}$, the field functions due to the line segment with $1, t, t^2, t^3$ weight distributions can be integrated to

$$F^{1}_{\text{line}}(\mathbf{P}) = \ln\left(\frac{m+(l-h)}{d-h}\right),$$

$$F^{t}_{\text{line}}(\mathbf{P}) = hF^{1}_{\text{line}}(\mathbf{P}) + (m-d),$$

$$F^{t^2}_{\text{line}}(\mathbf{P}) = 2hF^{t}_{\text{line}}(\mathbf{P}) - h^2F^{1}_{\text{line}}(\mathbf{P}) + \frac{1}{2}((l-h)m+hd) - \frac{d^2-h^2}{2}\ln\left(\frac{m+(l-h)}{d-h}\right),$$

$$F^{t^3}_{\text{line}}(\mathbf{P}) = 3hF^{t^2}_{\text{line}}(\mathbf{P}) - 3h^2F^{t}_{\text{line}}(\mathbf{P}) + h^3F^{1}_{\text{line}}(\mathbf{P}) + (l-h)^2m - h^2d - \frac{2}{3}m^3 + \frac{2}{3}d^3.$$

3.3. Inverse squared function

For the inverse squared function

$f(r) = 1/r^2,$

after the integration, we obtain

$$F^{1}_{\text{line}}(\mathbf{P}) = \frac{1}{\sqrt{d^2-h^2}}\left(\arctan\frac{l-h}{\sqrt{d^2-h^2}} + \arctan\frac{h}{\sqrt{d^2-h^2}}\right),$$

$$F^{t}_{\text{line}}(\mathbf{P}) = hF^{1}_{\text{line}}(\mathbf{P}) + \frac{1}{2}\ln\frac{d^2-2hl+l^2}{d^2},$$

$$F^{t^2}_{\text{line}}(\mathbf{P}) = 2hF^{t}_{\text{line}}(\mathbf{P}) - d^2F^{1}_{\text{line}}(\mathbf{P}) + l,$$

$$F^{t^3}_{\text{line}}(\mathbf{P}) = 3hF^{t^2}_{\text{line}}(\mathbf{P}) - (2h^2+d^2)F^{t}_{\text{line}}(\mathbf{P}) + h^3F^{1}_{\text{line}}(\mathbf{P}) + \frac{l^2}{2} - lh.$$

3.4. Inverse cubic function

The inverse cubic function $f(r) = 1/r^3$ was introduced by Marie-Paule Cani et al. for fast convolution computation, and was successfully used for subdivision curve skeletons in implicit surface modeling [24]. The analytical fields for line segments with weight distributions $t^i (i = 0, 1, 2, 3)$ are:

$$F^{1}_{\text{line}}(\mathbf{P}) = \frac{1}{d^2-h^2}\left(\frac{l-h}{m} + \frac{h}{d}\right),$$

$$F^{t}_{\text{line}}(\mathbf{P}) = hF^{1}_{\text{line}}(\mathbf{P}) + \frac{1}{d} - \frac{1}{m},$$

$$F^{t^2}_{\text{line}}(\mathbf{P}) = 2hF^{t}_{\text{line}}(\mathbf{P}) - h^2F^{1}_{\text{line}}(\mathbf{P}) + \ln\left(\frac{l-h+m}{d-h}\right) - \left(\frac{l-h}{m} + \frac{h}{d}\right),$$

$$F^{t^3}_{\text{line}}(\mathbf{P}) = 3hF^{t^2}_{\text{line}}(\mathbf{P}) - 3h^2F^{t}_{\text{line}}(\mathbf{P}) + h^3F^{1}_{\text{line}}(\mathbf{P}) + m - d + (d^2-h^2)\left(\frac{1}{m} - \frac{1}{d}\right).$$

3.5. Inverse 2n+1 degree function

From the derivation for the inverse cubic function, the log function in the analytical fields can be avoided by using higher degree inverse functions. For the inverse $2n+1$ degree function $f(r) = 1/r^{(2n+1)}$, $n\geqslant 2$, we have

$$F^{1}_{\text{line}}(\mathbf{P}) = \frac{1}{(d^2-h^2)^n}\sum_{k=0}^{n-1}\frac{(-1)^k}{2k+1}\binom{n-1}{k} \times \left.\frac{t^{2k+1}}{(t^2+(d^2-h^2))^{(2k+1)/2}}\right|^{l-h}_{-h},$$

$$F^{t}_{\text{line}}(\mathbf{P}) = hF^{1}_{\text{line}}(\mathbf{P}) - \left.\frac{1}{(2n-1)(t^2+(d^2-h^2))^{(2n-1)/2}}\right|^{l-h}_{-h},$$

$$F^{t^2}_{\text{line}}(\mathbf{P}) = 2hF^{t}_{\text{line}}(\mathbf{P}) - h^2F^{1}_{\text{line}}(\mathbf{P}) + \frac{1}{(d^2-h^2)^{n-1}}\sum_{k=0}^{n-2}\frac{(-1)^k}{2k+3}\binom{n-2}{k} \times \left.\frac{t^{2K+3}}{(t^2+(d^2-h^2))^{(2K+3)/2}}\right|^{l-h}_{-h},$$

$$F^{t^3}_{\text{line}}(\mathbf{P}) = 3hF^{t^2}_{\text{line}}(\mathbf{P}) - 3h^2F^{t}_{\text{line}}(\mathbf{P}) + h^3F^{1}_{\text{line}}(\mathbf{P}) - \frac{1}{(2n-3)(t^2+(d^2-h^2))^{(2n-3)/2}}\Bigg|^{l-h}_{-h} + \frac{d^2-h^2}{(2n-1)(t^2+(d^2-h^2))^{(2n-1)/2}}\Bigg|^{l-h}_{-h},$$

which does not require log computation. When n equals 2, we have the simplest inverse quintic function $f(r) = 1/r^5$, and obtain

$$F^{1}_{\text{line}}(\mathbf{P}) = \frac{1}{(d^2-h^2)^2}\left(\frac{l-h}{m} + \frac{h}{d} - \frac{(l-h)^3}{3m^3} - \frac{h^3}{3d^3}\right),$$

$$F^{t}_{\text{line}}(\mathbf{P}) = hF^{1}_{\text{line}}(\mathbf{P}) - \frac{1}{3}\left(\frac{1}{m^3} - \frac{1}{d^3}\right),$$

$$F^{t^2}_{\text{line}}(\mathbf{P}) = 2hF^{t}_{\text{line}}(\mathbf{P}) - h^2F^{1}_{\text{line}}(\mathbf{P}) + \frac{1}{3(d^2-h^2)}\left(\frac{(l-h)^3}{m^3} + \frac{h^3}{d^3}\right),$$

$$F^{t^3}_{\text{line}}(\mathbf{P}) = 3hF^{t^2}_{\text{line}}(\mathbf{P}) - 3h^2F^{t}_{\text{line}}(\mathbf{P}) + h^3F^{1}_{\text{line}}(\mathbf{P}) - \left(\frac{1}{m} - \frac{1}{d}\right) + \frac{d^2-h^2}{3}\left(\frac{1}{m^3} - \frac{1}{d^3}\right).$$

3.6. Blended inverse function

A disadvantage of the above inverse type of kernel functions is that it does not provide any control parameter. Since the field decays slowly for the inverse linear function, and faster for the inverse quintic function, we propose a new kernel function by forming a weighted average of them as follows:

$$f(r) = \frac{1-s^4}{r} + \frac{s^4}{r^5}, \quad 0 \leqslant s \leqslant 1,$$

where s is a smoothness control parameter. We call it the *blended inverse function*. We use s^4, instead of s, so as to compensate for the different decaying rates such that when s equals 0.5, the resulting convolution surface is about a half-way blend between the surfaces generated by the two inverse functions. Fig. 2 illustrates the effect of different s values on some convolution surfaces, which have two crossed line segments as their underlying skeleton.

The analytical field formulae for line segments with weight distributions $t^i (i = 0, 1, 2, 3)$ are

$$F^{1}_{\text{line}}(\mathbf{P}) = (1-s^4)\ln\left(\frac{l-h+m}{d-h}\right) + \frac{s^4}{(d^2-h^2)^2}\left(\frac{l-h}{m} + \frac{h}{d} - \frac{(l-h)^3}{3m^3} - \frac{h^3}{3d^3}\right),$$

$$F^{t}_{\text{line}}(\mathbf{P}) = hF^{1}_{\text{line}}(\mathbf{P}) + (1-s^4)(m-d) - \frac{s^4}{3}\left(\frac{1}{m^3} - \frac{1}{d^3}\right),$$

$$F^{t^2}_{\text{line}}(\mathbf{P}) = 2hF^{t}_{\text{line}}(\mathbf{P}) - h^2F^{1}_{\text{line}}(\mathbf{P}) + \frac{1-s^4}{2}((l-h)m + hd) - (1-s^4)\frac{d^2-h^2}{2}\ln\left(\frac{m+(l-h)}{d-h}\right) + \frac{s^4}{3(d^2-h^2)}\left(\frac{(l-h)^3}{m^3} + \frac{h^3}{d^3}\right),$$

$$F^{t^3}_{\text{line}}(\mathbf{P}) = 3hF^{t^2}_{\text{line}}(\mathbf{P}) - 3h^2F^{t}_{\text{line}}(\mathbf{P}) + h^3F^{1}_{\text{line}}(\mathbf{P}) + (1-s^4)\left[(l-h)^2m - h^2d - \frac{2}{3}m^3 + \frac{2}{3}d^3\right] - s^4\left(\frac{1}{m} - \frac{1}{d}\right) + \frac{s^4(d^2-h^2)}{3}\left(\frac{1}{m^3} - \frac{1}{d^3}\right).$$

3.7. Cauchy function

Cauchy function was first introduced to implicit surface modeling as a convolution kernel by McCormack and Sherstyuk [17],

$$f(r) = \frac{1}{(1+s^2r^2)^2}, \quad r > 0,$$

where s is a parameter used to control the width of the kernel. The main advantage of this kernel is that it can produce closed-form solutions for the most number of skeletal primitives, namely points, line segments, polygons, arcs and planes. We have presented the analytical solution for line-segment skeleton with polynomial weight distribution for Cauchy kernel function and its performance in [21].

3.8. Quartic polynomial

Several polynomial potential functions have been adopted in implicit surface modeling; for example, the

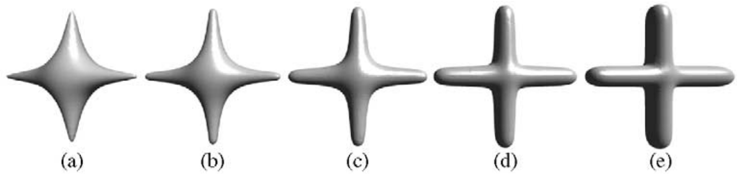

Fig. 2. Convolution surfaces with blended inverse kernel and various s values: (a) $s = 0$, (b) $s = 0.25$, (c) $s = 0.5$, (d) $s = 0.75$, (e) $s = 1.0$.

piecewise quadratic polynomials used in metaball [9], the six-degree polynomials used in soft object modeling [10], and the quartic polynomial used in ray-tracing software such as *Rayshade* and *POV-Ray*. Among these polynomials, the quartic polynomial is the simplest, thus we consider it here; the analytical formulae for the other polynomial potential functions can be analogously deduced, albeit with more computation cost.

The quartic polynomial kernel is represented by

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

where R is the effective radius of the kernel. Due to the kernel's finite support, for any point on the skeletal line segment whose distance to a point $\mathbf{P}$ is larger than R, its field contribution to $\mathbf{P}$ is zero. Thus when calculating the field at the point $\mathbf{P}$, we must use a sphere centered at $\mathbf{P}$ with radius R to clip the line segment to find the sub-line-segment that contributes to the field function (Fig. 3).

The equation of the sphere centered at $\mathbf{P}$ with radius R is

$$(\mathbf{X} - \mathbf{P})^2 = R^2.$$

Substituting $\mathbf{L}(t)$ into the sphere equation gives

$$(t - h)^2 = R^2 - d^2 + h^2.$$

If the discriminant $\Delta = R^2 - d^2 + h^2 < 0$, there is no intersection between $\mathbf{L}(t)$ and the sphere, and the field contribution from the line segment to $\mathbf{P}$ is zero; otherwise, i.e., $\Delta \geqslant 0$, the intersection points between the line and the sphere are

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

If $t_2 < 0$ or $t_1 > l$, then there is no valid intersections between $\mathbf{L}(t)$ and the sphere, and the line segment contributes zero field to $\mathbf{P}$. Otherwise, the sub-line-segment of $\mathbf{L}(t)$ that contributes positive field to $\mathbf{P}$ is $[l_1, l_2]$, where $l_1 = \max(0, t_1)$ and $l_2 = \min(l, t_2)$.

By integrating in domain $[l_1, l_2]$, we can obtain the analytical field formulae for $\mathbf{L}(t)$ with weight distributions $t^i (i = 0, 1, 2, 3)$ as follows:

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

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

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

$$\begin{aligned} F^{t^3}_{\text{line}}(\mathbf{P}) = \frac{1}{R^4}\Big(&\frac{1}{8}(l_2^8 - l_1^8) - \frac{4}{7}(l_2^7 - l_1^7)h + \frac{1}{3}(l_2^6 - l_1^6) \\ &\times (2h^2 - (R^2 - d^2)) + \frac{4}{5}(l_2^5 - l_1^5)h(R^2 - d^2) \\ &+ \frac{1}{4}(l_2^4 - l_1^4)(R^2 - d^2)^2\Big). \end{aligned}$$

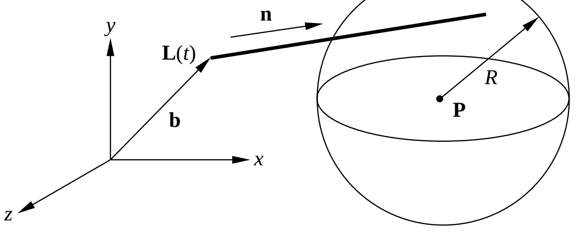

Fig. 3. Calculating the integration domain for a line segment.

4. Weight distribution control with cubic profile curves

As it is difficult to imagine the shape of a polynomial curve only from its coefficients, we present an intuitive interface for defining a cubic spline curve to control the weight distribution along a line-segment skeleton. The idea of this cubic control curve is similar to that of Kochanek et al's interpolating splines [25], which are widely used for designing keyframe animation in commercial animation software such as Softimage, Alias|Wavefront, and Maya.

A degree n Bezier curve $q(u)$ with control points $(i/n, q_i), i = 0, 1, 2, \cdots, n$ is given by

$$q(u) = \sum_{i=0}^{n} \left(\frac{i}{n}, q_i\right) B_{i,n}(u),$$

where $B_{i,n}(u)$ are the Bernstein polynomials. From the following identity

$$\sum_{i=0}^{n} \frac{i}{n} B_{i,n}(u) = u[(1 - u) + u]^n = u,$$

it is clear that the abscissa of any point $q(u)$ on the curve equals u, and thus the curve can be rewritten as $q(u) = \sum_{i=0}^{n} q_i B_{i,n}(u)$. This special type of Bezier curve is called non-parametric Bezier curve [26] and is an intuitive and computationally efficient shape control tool.

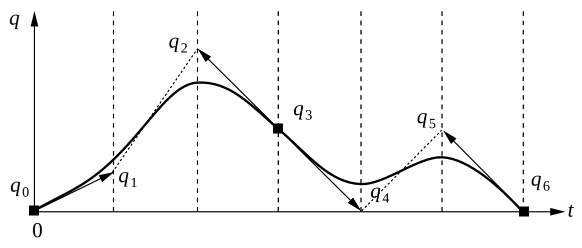

Fig. 4. A cubic control curve for defining a weight distribution.

Fig. 4 shows our interface. Those points marked as solid squares (q_0, q_3, q_6) are the points to be interpolated, which can be added or removed freely according to user's requirement. The control points between two adjacent interpolated points (e.g., q_1 and q_2) are automatically added for locally controlling the shape of the curve. They are restricted to move vertically and maintain geometric continuity at the interpolated points. Clearly, a user may break the slopes to introduce discontinuity in the weight distribution.

In order to calculate the field function of the entire line-segment skeleton, the profile curve is subdivided at the interpolated points and the field functions of individual sub-line-segments are summed. The correctness is guaranteed by the superposition property of convolution surfaces. Take the first span of the curve in Fig. 4 as an example. Let the length of the sub-line-segment between q_0 and q_3 be l_0, and let $u = t/l_0$, then the weight at parameter t is

$$q'(t) = q(u) = \sum_{i=0}^{3} q_i B_{i,3}(u) = \sum_{i=0}^{3} q'_i t^i,$$

where

$$\begin{aligned} q'_0 &= q_0, \\ q'_1 &= 1/l_0(-3q_0 + 3q_1), \\ q'_2 &= 1/l_0^2(3q_0 - 6q_1 + 3q_2), \\ q'_3 &= 1/l_0^3(-q_0 + 3q_1 - 3q_2 + q_3), \end{aligned}$$

thus the field function for this sub-line-segment is

$$\begin{aligned} F_{\text{line}}(\mathbf{P}) = {} & q'_0 F^1_{\text{line}}(\mathbf{P}) + q'_1 F^t_{\text{line}}(\mathbf{P}) \\ & + q'_2 F^{t^2}_{\text{line}}(\mathbf{P}) + q'_3 F^{t^3}_{\text{line}}(\mathbf{P}). \end{aligned}$$

The total field can be obtained by summing the fields of all sub-line-segments.

5. NURBS curve skeletons

Many natural objects can be abstracted as curves. Since NURBS curve is a standard for representing freeform curves, we now consider generating convolution surfaces from NURBS curve skeletons. The NURBS curves are assumed to be cubic with clamped knot vectors (i.e., the curves interpolate the two end control points). By specifying a dense parameter that controls the number of points into which each polynomial segment is to be divided [26], and applying the deBoor algorithm, we can approximate the NURBS curve into a polyline $P_0P_1 \cdots P_n$.

To compute the control points of the Bezier weight distribution curve for each sub-line-segment in the polyline, we need the arc-length parametrization of the polyline. Let the length of the ith segment P_iP_{i+1} of the polyline be $l_i = \|P_{i+1} - P_i\|$, $i = 0, 1, \cdots, n-1$. By using the chord length parametrization, we obtain the following parameters at the joints of the polyline:

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

For each sub-line-segment P_iP_{i+1}, $i = 0, 1, 2, \cdots, n-1$, we apply the de Casteljau algorithm [26] to subdivide the profile curve $q(u)$ to compute the control points of the weight distribution curve over interval $[u_i, u_{i+1}]$. The field for the entire NURBS curve is then evaluated by summing the fields of all sub-line-segments.

6. Computational efficiency and results

We have implemented our algorithm on a Pentium III 400E PC with 128M main memory. To analyze the computational complexity of the various kernels, we use a skeleton consisting of two line segments: from (−4,0,0) to (4,0,0), and from (0,−4,0) to (0,4,0). The space volume is organized into a uniform grid of 150 × 150 × 150 nodes, yielding 3.375 million function evaluations. The time taken to evaluate the skeleton with cubic polynomial distribution and the various kernels is given in Table 1. When evaluating the field functions, we make full use of previously calculated results with optimizations to reduce the number of operations. The kernels are ranked according to their evaluation time, and the ranking is found to be close to Sherstyuk's speed ratings for line segments with uniform distributions [19]. Table 1 also lists the special functions needed, which can be regarded as a rough indication of the kernel's performance.

The results show that polynomial kernel function is still the most efficient. One reason is that polynomial kernel involves the least number of special functions; only one square root is needed. Another reason is that polynomial function has finite support: the field of any point further from the line segment than the effective radius is zero, and thus further computations can be eliminated. Obviously, when choosing a kernel function, computational cost is not the only factor to consider.

Table 1
Time taken and speed ranking

Kernel function	Special functions	Time (s)	Rating
Quartic polynomial	1 sqrt	2.390	1
Inverse linear	2 sqrt, 1 log	5.378	2
Inverse cubic	2 sqrt, 1 log	6.764	3
Inverse quintic	2 sqrt	7.734	4
Inverse squared	1 sqrt, 2 atan, 1 log	7.913	5
Blended inverse	2 sqrt, 1 log	9.552	6
Cauchy	1 sqrt, 1 log, 2 atan	10.332	7
Gauss	2 erf, 3 exp	13.432	8

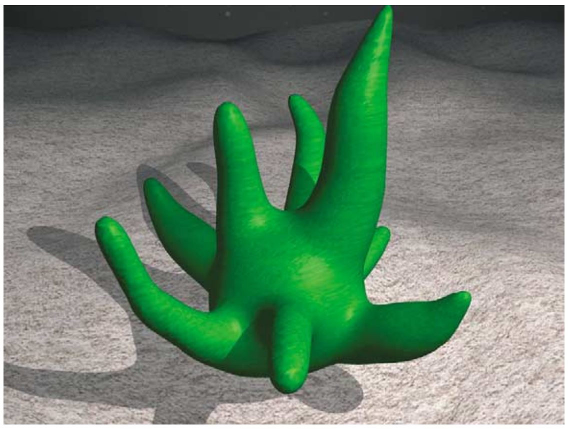
Fig. 5. An alien plant.

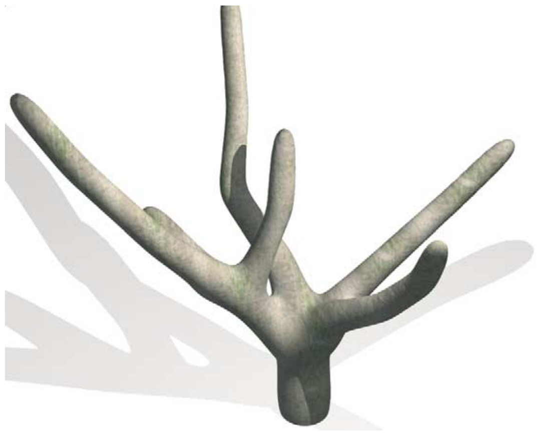
Fig. 6. The trunk of a joshua tree.

Compatibility between kernels and primitives, and the controllability of the resulting surface, should also be taken into account. From the table, our proposed blended inverse kernel appears as a competitive one compared with the Cauchy kernel. For the inverse type of kernels, the field of any point in the skeleton is infinite; this feature guarantees that the skeleton is

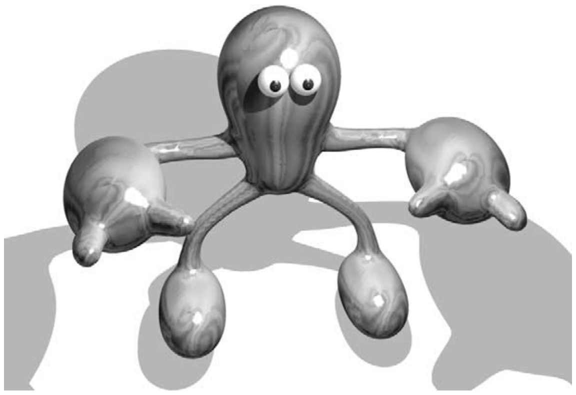
Fig. 7. A cartoon enforcer.

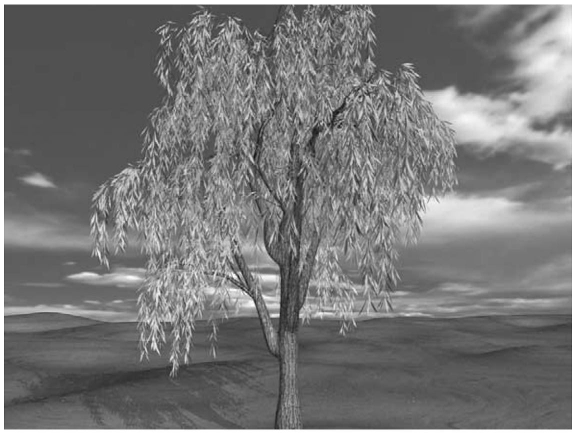
Fig. 8. A willow tree.

always preserved regardless of the threshold chosen, which can be regarded as an advantage. For other kernel functions, such as polynomial, Cauchy and Gaussian kernels, improperly chosen threshold may ‘‘eat away’’ part of the skeleton, which may be annoying for designing a shape using one-dimensional skeletons.

We have rendered several modeling examples to demonstrate the modeling capabilities of our method. For the sake of uniform processing, all the convolution models are first polygonized into polygon meshes using Bloomenthal's polygonizer algorithm [27,28], and a ray-tracing algorithm is then employed to render the polygon meshes with solid or projective texture mapping. The trunk of the potted plant shown in Fig. 1(a) is convolved with 1 NURBS curve and 5 line segments. The cactus shown in Fig. 1(b) is convolved with 8 line segments. Fig. 5 is a virtual alien plant; its skeleton is made up of 24 line segments. The joshua tree trunk in Fig. 6 is modeled with 20 line segments, while the cartoon enforcer in Fig. 7 with 32 line segments. Fig. 8 is an example of a willow tree, with the stem and the main branches modeled with convolution surfaces, and the twigs and leaves modeled with traditional modeling methods. These examples show that the convolution surfaces with non-uniform density distributions can be used to model tapering effects conveniently and efficiently.

7. Conclusions

Convolution surfaces have the advantage of producing pleasing crease-free and bulge-free features over other kinds of implicit surfaces. One drawback that prevents its more extensive use is the expensive computation and high memory cost incurred by numerical methods if closed-form solution does not exist for calculating the field function. In this paper, we present the closed-form solutions for line-segment skeletons with polynomial weighted distributions for most of the commonly used kernel functions. The polynomial distribution along a line-segment skeleton can be intuitively specified using a cubic Bezier spline curve. The skeletons may also be NURBS curves, in which case they are automatically subdivided into line segments. Since tapering effects and tubular shapes of varying radius are prevalent in organic shapes, our exact evaluation formulae provide an effective solution to model these shapes. We also propose a new competitive kernel that has a smoothness control parameter. Finally, we examine the performance of the various kernels for modeling convolution surfaces with 1D skeletons.

Implicit surface is an increasingly popular approach for modeling smooth and complex topology objects, which may be difficult to model with parametric or mesh-based geometric methods. Since curve skeletons are good abstractions for a wide variety of natural forms, and curve skeletons can be approximated by polylines, our method can be used to model and animate a wide variety of complex organic objects. This method can also be used in skeleton-based space deformation and shape morphing. Experimental results demonstrate that our method can create many aesthetically pleasing branching effects and possesses many potential applications in both geometric modeling and computer animation.

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. We are grateful to Andrei Sherstyuk for his help on convolution surfaces and to anonymous reviewers for their constructive suggestions. This work received support from 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] Bloomenthal J, Bajaj C, Blinn J, Cani M, Rockwood A, Wyvill B, Wyvill G. An introduction to implicit surfaces. Los Altos, CA: Morgan Kaufmann Publishers, 1997.
[2] Dobashi Y, Kaneda K, Yamashita H, Okita T, Nishita T. A simple, efficient method for realistic animation of clouds. Proceedings of SIGGRAPH'00, 2000. p. 19–28.
[3] Cani-Gascuel M, Desbrun M. Animation of deformable models using implicit surfaces. IEEE Transactions on Visualization and Computer Graphics 1997;3(1):39–50.
[4] Nishita T, Iwasaki H, Dobashi Y, Nakamae E. A modeling and rendering method for snow by using metaballs. Computer Graphics Forum 1997;16(3):357–64.
[5] Jin X, Li Y, Peng Q. General constrained deformation based on generalized metaballs. Computers & Graphics 2000;24(2):219–31.
[6] Savchenko V, Pasko A, Okunev O, Kunii T. Function representation of solids reconstructed from scattered surface points and contours. Computer Graphics Forum 1995;14(4):181–8.
[7] Turk G, O'Brien J. Shape transformation using variational implicit functions. Proceedings of SIGGRAPH'99, 1999. p. 335–42.
[8] Blinn J. A generalization of algebraic surface drawing. ACM Transactions on Graphics 1982;1(3):235–56.
[9] Nishimura H, Hirai M, Kawai T. Object modeling by distribution function and a method of image generation. Transactions on IECE 1985;68-D(4):718–25.
[10] Wyvill G, McPheeters C, Wyvill B. Data structure for soft objects. The Visual Computer 1986;2(4):227–34.
[11] Wyvill B, Wyvill G. Field functions for implicit surfaces. The Visual Computer 1989;5(1/2):75–82.
[12] Bloomenthal J, Wyvill B. Interactive techniques for implicit modeling. Computer Graphics 1990;24(2): 109–16.
[13] Bloomenthal J. Skeletal design of natural forms, Ph.D. dissertation, University of Calgary, Department of Computer Science, 1995.
[14] Bloomenthal J. Bulge elimination in convolution surfaces. Computer Graphics Forum 1997;16(1):31–41.
[15] Bloomenthal J, Shoemake K. Convolution surfaces. Computer Graphics 1991;25(4):251–6.
[16] Attali D, Montanvert A. Computing and simplifying 2d and 3d semi-continuous skeletons of 2d and 3d shapes. Computer Vision and Image Understanding 1997;67(3): 261–73.
[17] McCormack J, Sherstyuk A. Creating and rendering convolution surfaces. Computer Graphics Forum 1998; 17(2):113–20.
[18] Sherstyuk A. Convolution surfaces in computer graphics. Ph.D. dissertations, Monash University, Australia, 1999.
[19] Sherstyuk A. Kernel functions in convolution surfaces: a comparative analysis. The Visual Computer 1999; 15(4):171–82.

[20] Ferley E, Cani M, Attali D. Skeletal reconstruction of branching shapes. Computer Graphics Forum 1997;16(5):283–93.

[21] Jin X, Tai CL, Feng J, Peng Q. An analytical convolution surface model for line skeletons with polynomial weighted distributions. Journal of Graphics Tools 2001;6(3):1–12.

[22] Muraki S. Volumetric shape description of range data using blobby model. Computer Graphics 1991;25(4): 227–35.

[23] Wyvill B, van Overveld K. Tiling techniques for implicit skeletal models. SIGGRAPH Course Notes 1996;11:1–26.

[24] Cani M, Hornus S. Subdivision curve primitives: a new solution for interactive implicit modeling. Proceedings of the Shape Modeling International'01, IEEE Computer Society, Geneva, Italy, 2001.

[25] Kochanek DH, Bartels R. Interpolating splines with local tension, continuity and basis Control. Computer Graphics 1984;18(3):33–41.

[26] Farin G. Curves and surfaces for computer aided geometric design: a practical guide, 4th ed. New York: Academic Press, 1997.

[27] Bloomenthal J. Polygonization of implicit surfaces. Computer Aided Geometric Design 1988;5(4):341–55.

[28] Bloomenthal J. An implicit surface polygonizer. In: Graphics gems IV. New York: Academic Press, 1994. p. 324–49.

Xiaogang Jin

CURRICULUM VITAE

Xiaogang Jin is a professor of the State Key Lab of CAD&CG, Zhejiang University, People's Republic of China. He received his B.Sc. degree in Computer Science in 1989, M.Sc. and Ph.D. 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

CURRICULUM VITAE

Chiew-Lan Tai is an Assistant Professor in the Department of Computer Science, Hong Kong University of Science & Technology. She received her B.Sc. and M.Sc. in Mathematics from the University of Malaya, and her M.Sc. in Computer and Information Sciences from the National University of Singapore. She earned her D.Sc. in Information Science from the University of Tokyo in 1997. Her research interests include geometric modeling, computer graphics, digital Chinese art, and interpretation of engineering drawings.