Vol. 6, No. 4: 41–54

An Efficient and Controllable Blob Function

Gladimir V. G. Baranoski
University of Waterloo

Jon Rokne
University of Calgary

Abstract. A tunable blob function is proposed which is computationally simple since it is a rational function of low degree. The blob function is dependent on a parameter d which can be used to control the blob function making it suitable for computer graphics applications such as implicit surface generation. The main properties of the blob function are described and graphed. When the parameter d is large, the blob function can be used to generate a three-dimensional spike by rotating it around the z-axis.

1. Introduction

In *Tricks of the Trade: Andrew Glassner's Notebook* [Glassner 01], Glassner discusses some functions that he calls *blob functions.* These functions are "radially symmetric, smooth shapes—the sort of thing you'd get if you plopped a spoonful of pudding onto a dish" [Glassner 01]. These functions are useful in computer graphics, for example, for implicit surface generation and general filtering (see [Bloomenthal 95], [Hartman 01], [Muraki 91]).

The original work on blob functions was done by Blinn [Blinn 82] and Perlin [Perlin 01] in 1980–1982 under the name of *field functions.*

Here we desire a blob function to have certain properties that ensure smoothness of blends. Hence we define a blob function to be a function $b(r)$ where

$$\begin{aligned} b(0) &= 1, \\ b'(0) &= 0, \\ b(1) &= 0, \\ b'(1) &= 0, \\ b'(r) &< 0, \text{ when } 0 < r < 1, \\ b(-r) &= b(r), \\ b(r) &= 0, \text{ when } |r| \notin [0,1], \\ b &\in C^2[0,1] \\ \int_0^1 b(r)dr &= 1 \text{ (an optional normalizing condition).} \end{aligned}$$

These conditions restrict the possible blob functions to functions that have a reversed "s"-shape. They also imply that there are three numbers $\xi < \theta < \eta$, $\xi, \theta, \eta \in [0,1]$ such that $b'(\xi) = b''(\theta) = b'(\eta) = -1$. In other words, the slope of the function is equal to the slope of the line between $(0,1)$ and $(1,0)$ at two points, and the second derivative has the slope of the same line at one point.

The blob function

$$b_w(r) = \begin{cases} 1 - (22r^2 - 17r^4 + 4r^6)/9 & \text{if } |r| < 1, \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

which was originally published in [Wyvill et al. 86], satisfies these requirements. Additionally, this blob function satisfies $b_w(1/2) = 1/2$. It closely approximates another blob function,

$$b_1(r) = \begin{cases} (1 + \cos(r\pi))/2 & \text{if } |r| < 1, \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

which also has the required properties. Glassner graphs both of these blob functions [Glassner 01].

Bloomenthal notes [Bloomenthal 97] that it might be desirable to be able to control blends in implicit surface based computer animation. This can be done by blob functions with tunable parameters. An interpolating Catmull-Rom spline with three control points is suggested by Bloomenthal [Bloomenthal 97] where the center control point controls the slope at that point.

Glassner also notes [Glassner 01] that Equation (1) has no tunable parameters, and two alternate possibilities for blob functions

$$b_p(r) = \begin{cases} (1 + \cos(r\pi))/2)^p & \text{if } |r| < 1, \\ 0 & \text{otherwise;} \end{cases} \tag{3}$$

$$c_p(r) = \begin{cases} (e^{-(rp)^2} - e^{-p^2})/(1 - e^{-p^2}) & \text{if } |r| < 1, \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

both of which have a tunable parameter p, are provided (where e is Euler's constant). Glassner gives examples for several values of p [Glassner 01].

Graphs comparing several blob functions can be found in [Bourke 97], and an early survey of blob functions can be found in [Wyvill, Wyvill 89].

2. The Blob Function

When blob functions are used for implicit surface blending [Bloomenthal 97], they are evaluated possibly several times whenever a surface point is required. Small changes in the computational cost of a blob function might therefore affect the computational cost of generating an implicit surface. Therefore we provide another blob function which is tunable and which is computationally simple since it is a rational function of low degree. Let

$$b_d(r) = \begin{cases} (a + br^2 + cr^4)/(1 + dr^2) & \text{if } |r| < 1, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Then Equation (5) has 4 tunable parameters a, b, c, and d. The requirement that $b_d(0) = 1$ means that $a = 1$ and $b'_d(0) = 0$ are identically satisfied. Conditions $b_d(1) = b'_d(1) = 0$ result in

$$b + c = -1, \tag{6}$$

$$2b + 4c = 0, \tag{7}$$

from which $b = -2$ and $c = 1$. From this we get the new blob function

$$b_d(r) = \begin{cases} ((r-1)^2(r+1)^2)/(1 + dr^2) & \text{if } |r| < 1, \\ 0 & \text{otherwise,} \end{cases} \tag{8}$$

which has one tunable parameter. The parameter d has to satisfy $d > -1$ since $d < -1$ results in singular values in $(0, 1)$ by the fact that the denominator of Equation (8) is $g = 1 + dr^2$ and $g = 0$ at $r^2 = -1/d$.

As noted above, Equation (8) will have to be optionally normalized so that the area under the curve from zero to one is unity. This is accomplished by dividing by the scaling factor:

$$s = \frac{\sqrt{d}(3 + 5d) + (1 + d)^2 atan\sqrt{d}}{3d^{5/2}}. \tag{9}$$

(Note that Equation (9) is indeterminate for $d = 0$. However, applying l'Hopital's rule to Equation (9) results in $s = 8/15$ which is the same result as found by direct integration of Equation (8) with $d = 0$).

If the selection of an appropriate blob function is made by varying the parameter d, then we obtain the graph in Figure 1. In the graph, the parameter $d = (2i)^2$ with $i = 0, \ldots, 5$.

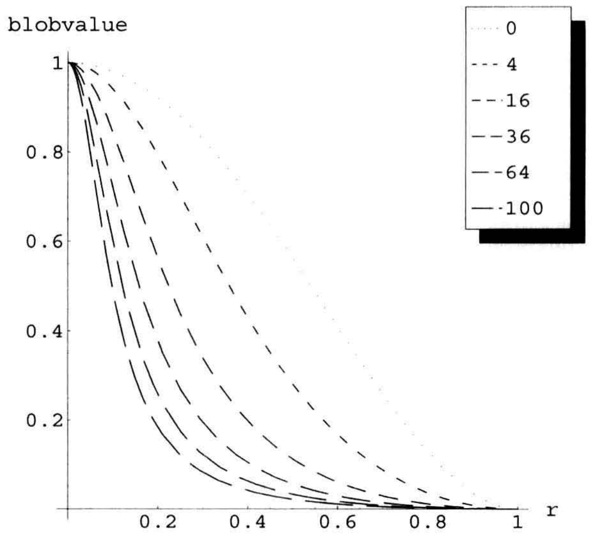

Figure 1. The parameter $d = (2i)^2$ with $i = 0, \ldots, 5$.

Similarly, Figure 2 shows the progression of blob function shape when the parameter $d = (2i)^3$ with $i = 0, \ldots, 5$. We can also let the parameter d be negative, varying it as $d = -0.19 - 0.2i$ with $i = 0, \ldots, 4$, obtaining the graphs in Figure 3.

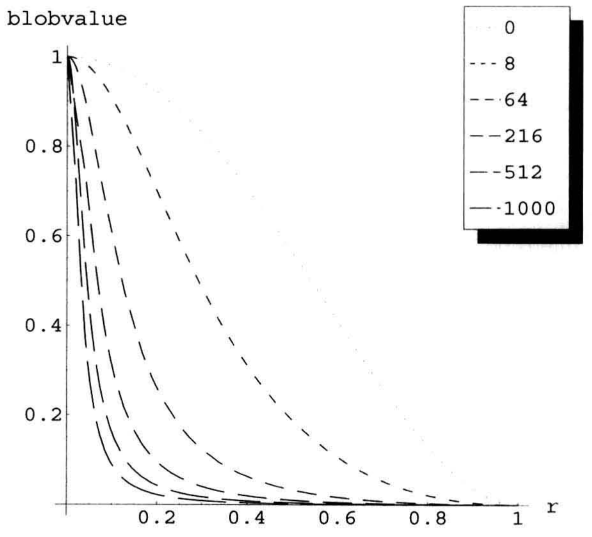

Figure 2. Plotted with $d = (2i)^3$ with $i = 0, \ldots, 5$.

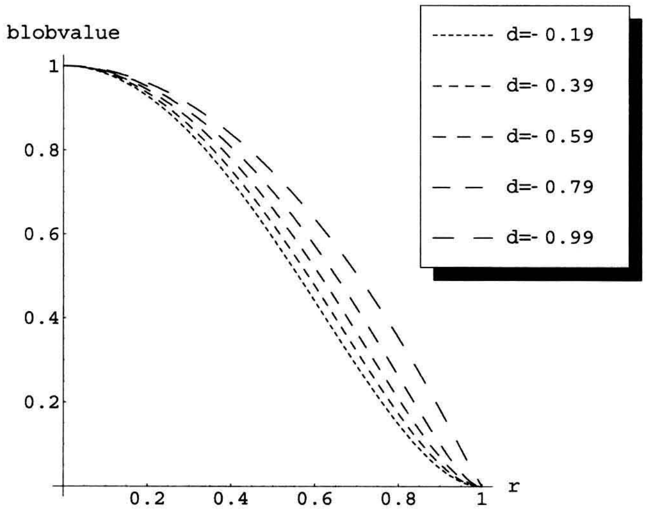

Figure 3. Negative values of *d*.

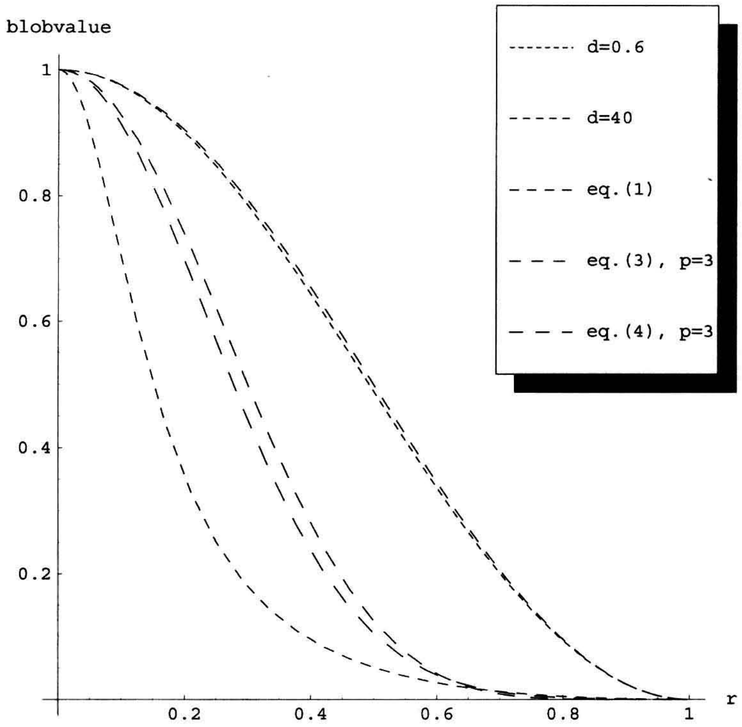

Figure 4. Comparing the blob function (8) to (1),(3) and (4).

The parameter d therefore enables control over blob function (8), making it suitable for computer graphics applications such as implicit surface generation.

The new blob function (8) is now compared visually to the blob functions (1), (3), and (4) in Figure 4. Two values, $d = 0.6$ and $d = 40$, were selected for the comparison. It should be noted that for $d = 0.6$, the new blob function is very close to blob function (1). The functions (3) and (4) are close to each other, but differ in shape from the new blob function with the parameter $d = 40$ (and for other values as well). The blob function (2) was not graphed since it was very close to blob function (1).

When blob functions are used to generate implicit surfaces, the contributions for each of the blob functions are added and compared to a fixed cutoff value. This value is normally $x = 0.5$, but it can be any value in the range $(0, 1]$. The shape at x determines the "hardness" or the "softness" of the blend (see also [Bloomenthal 95] pp. 247–248) where the more negative the slope is at x, the harder the blend.

The parameter d can be used to control the slope of blob function (8) at, for example, $x = 1/2$. This slope is

$$b'_d(0.5) = -1.5/(1 + d/4) - 0.5625d/(1 + d/4)^2. \tag{10}$$

Setting $b'_d(0.5) = p$ and solving for d we get two roots, one of which is

$$d = \frac{-15 - 8p + 12\sqrt{1.5625 + p}}{2p}, \tag{11}$$

the other resulting in an impossible value of d less than -1.

Equation (11) also shows that the slope can never be less than -1.5625 at $x = 0.5$. This restricts the control of Equation (8) to a limited range of derivative values at $x = 0.5$, as shown in Figure 5 as a function of the parameter d.

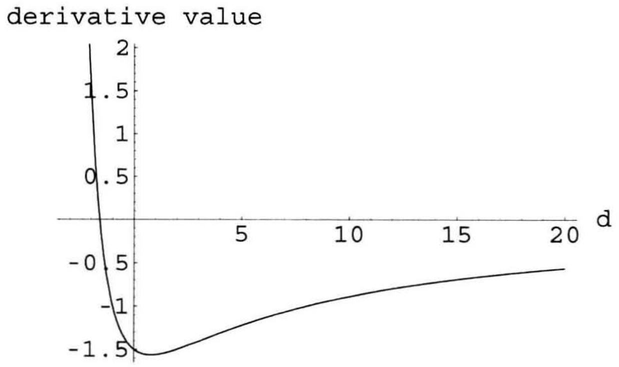

Figure 5. The curve for $b'_d(0.5)$ as a function of d.

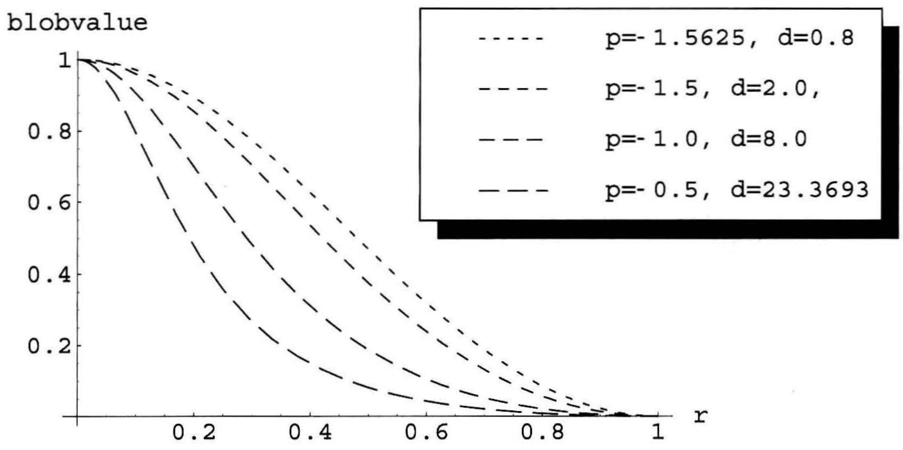

Figure 6. Varying the slope at $x = 1/2$.

In Figure 6, the plots are given for the blob functions generated with slopes $p = -1.5625, -1.5, -1.0, -0.5$ at $x = 0.5$. Among these curves the hardest blend for an implicit surface is given by $d = 23.3693$, and the softest with $d = 0.8$.

In Figure 7, the derivative is varied at the point x so that $x = 0.2+0.1i$, $i = 0, 1, 2, 3$. This point corresponds to the cutoff point for implicit blends. We note that the range of variation runs from about -2.5 to -1.5 as x ranges from 0.2 to 0.5. That is, the closer we choose the point of evaluation to the origin, the steeper the slope can be chosen. The range of possible curves does not change; however, the range of variation in hardness of blend does change by the fact that the cutoff point is changed.

As another example of the use of these blob functions, a rotation surface is shown in Figure 8 where the parameter $d = 23.3693$.

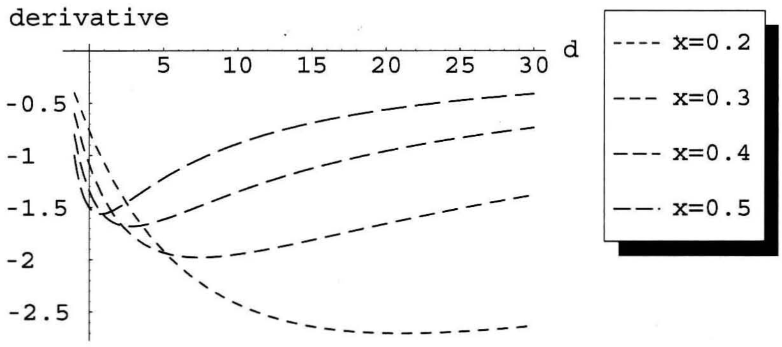

Figure 7. Varying the point of derivative control.

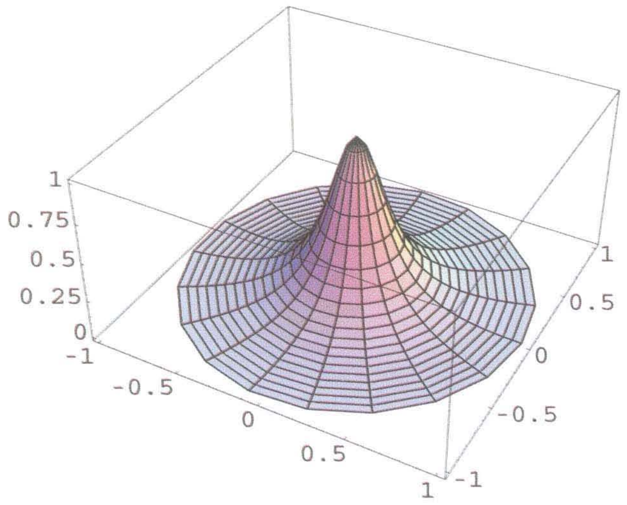

Figure 8. A three-dimensional symmetric surface generated with a blob function.

The surface in Figure 8 may be normalized so that the integral under the surface is equal to 1 dividing by the constant 1.47218.

For an implementation, Equation (8) is changed to

$$b_d(r) = \begin{cases} ((r^2-1)^2)/(1+dr^2) & \text{if } |r| < 1, \\ 0 & \text{otherwise,} \end{cases} \tag{12}$$

which is implemented as

$$\begin{aligned} r_1 &= r^2, \\ r_2 &= r_1 - 1, \\ r_3 &= r_2^2, \\ r_4 &= dr_1, \\ r_5 &= 1 + r_4, \\ r_6 &= r_2/r_5, \end{aligned}$$

resulting in four multiply/divide and two add (increment/decrement by 1). In comparing the cost of the new blob function to the other blob functions, Equation (1) is rewritten as

$$b_w(r) = \begin{cases} 1 - r^2((4/9r^2 - 17/9)r^2 + 22/9) & \text{if } |r| < 1, \\ 0 & \text{otherwise,} \end{cases} \tag{13}$$

with the constants 4/9, 17/9 and 22/9 being precalculated.

cost of evaluation				
	$b_d(r)$	$b_w(r)$	$b_p(r)$	$c_p(r)$
mpy/div	4	4	2	3
add/sub	2	3	1	2
fcn calls			1	2
remark	tunable	not tunable	tunable	tunable

Table 1. Comparison of evaluation costs for Equations (8), (1), (3), and (4).

The cost of evaluating the four different blob functions is given in Table 1 for the four blob functions (8), (1), (3), and (4) (excluding the test for inclusion/exclusion in $[0, 1]$ which is the same for all of the functions).

The comparison shows that Equation (8) is slightly less expensive to calculate than Equation (1), with the added advantage of being tunable. It is assumed that Equations (3) and (4) are more expensive due to the function calls.

Some attempts were made to generalize Equation (8), however, they were not successful because the resulting functions tended to have singularities in the domain [0,1].

3. Implicit Surface Examples

Three sets of examples of three-dimensional implicit surfaces were generated using the new blob functions based on point primitives. With the point primitive located at (x_0, y_0, z_0), the implicit surfaces are defined by $f(x, y, z) - 0.5 = 0$ where $f(x, y, z) = b_d(\sqrt{(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2})$, i.e., the Euclidean distance from the point primitive is used as the parameter for the blob function. An implicit surface generated from several point primitives is similarly defined by adding the contributions from each primitive. All of the test images are of peanut-like shapes.

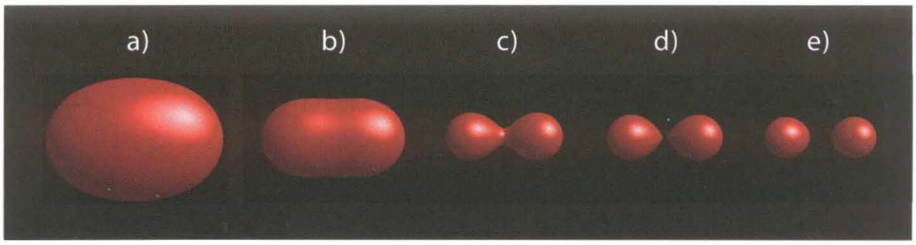

Figure 9. The effect of varying the d value for two point primitives. (a) 2, (b) 8, (c) 24, (d) 26, and (e) 32.

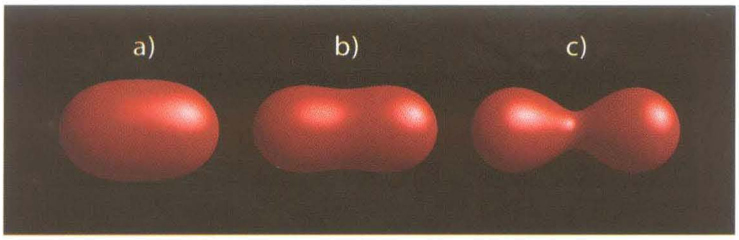

Figure 10. Varying the separation between two point primitives. (a) separation 0.6, (b) separation 0.8, and (c) separation 1.0.

In the first set of images in Figure 9, the point primitives are 0.6 units apart. Five images are displayed with d values $2.0, 8.0, 24.0, 26.0, 32.0$. The effect of increasing d is that the blob functions drop off more rapidly (see also Figure 3), which means that the 0.5 cutoff will become closer to the point primitives.

From Figure 9 c) to 9 d) the blobs separate. The separation occurs when at the distance 0.3, the value of the blob functions drop below the threshold 0.5 because the configuration is symmetric. The d value at the separation is therefore found by solving $2b_d(0.3) = 0.5$ for d resulting in $d = 24.88$.

In the second set of images in Figure 10, the d value is kept constant, whereas the separation between the point primitives is $0.6, 0.8$, and 1.0.

The third set of images in Figure 11 uses different d values for the two point primitives. The left point primitive uses $d = 1$, whereas the right point primitives have $d = 16.0, 32.0, 1024.0, 2048.0$.

4. Spike

Another potential use for function (8) for large values of d is to generate spikes which have smooth peaks. The graphs in Figure 12 are computed by

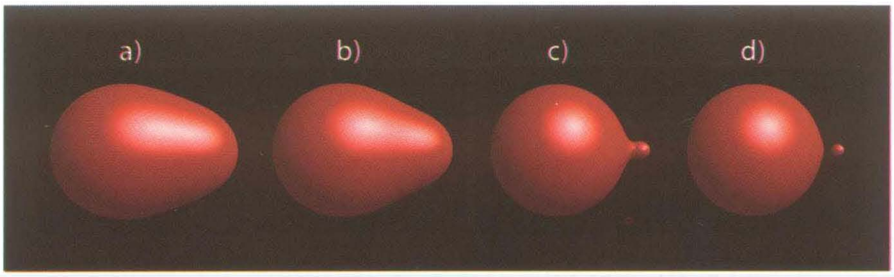

Figure 11. The effect of different d values for the right point primitives. (a) $d = 16$, (b) $d = 32$, (c) $d = 1024$, and (d) $d = 2048$.

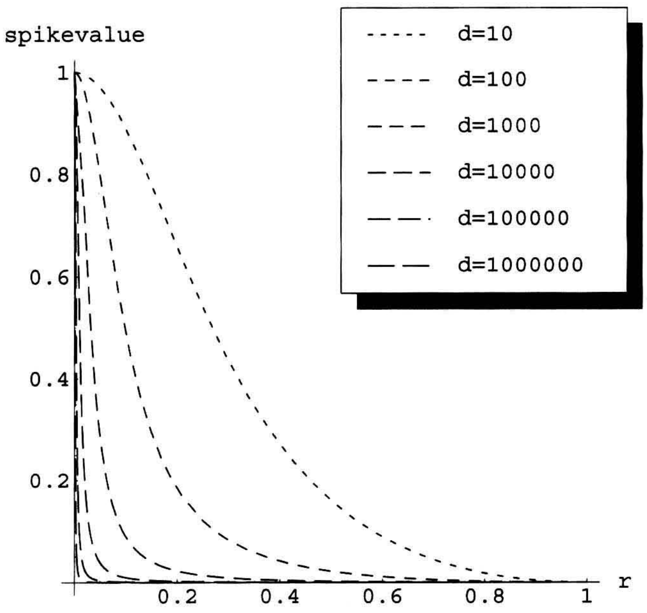

Figure 12. Increasing the parameter by multiplies of 10.

letting d increase by multiples of 10. When these functions are rotated about the y-axis, surfaces, which we call *spikes*, are generated. Spikes can be used for a number of modeling purposes such as creating textures by scaling and repeating the spike, modeling slime molds, branching processes, etc.

Note that the higher values of d produce a shape that "hugs" the y-axis and drops quickly towards the x-axis, then asymptotically goes to zero as r tends to 1.

Clearly, any rational form $p(x)/(1 + d^n x^m)$ will have this property when $n, m > 0$ as long as d is large enough. function (8) is distinguished from these functions because it is computationally simple and has the smoothness conditions stated in the previous section. This cannot be seen in Figure 12. We therefore use a mathematical magnifying glass and display a closer look at the situation around $x = 0$ in Figure 13, showing that even with very large values of d, the spike will have the required smoothness properties.

As a final example, we display a spike where $d = 1000$ in Figure 14.

Acknowledgements. Thanks are due to the anonymous referees who provided valuable criticism of the first draft of the paper and to Ryan Schmidt who generated the implicit surface examples. The authors are also grateful for funding through the National Sciences and Engineering Council of Canada.

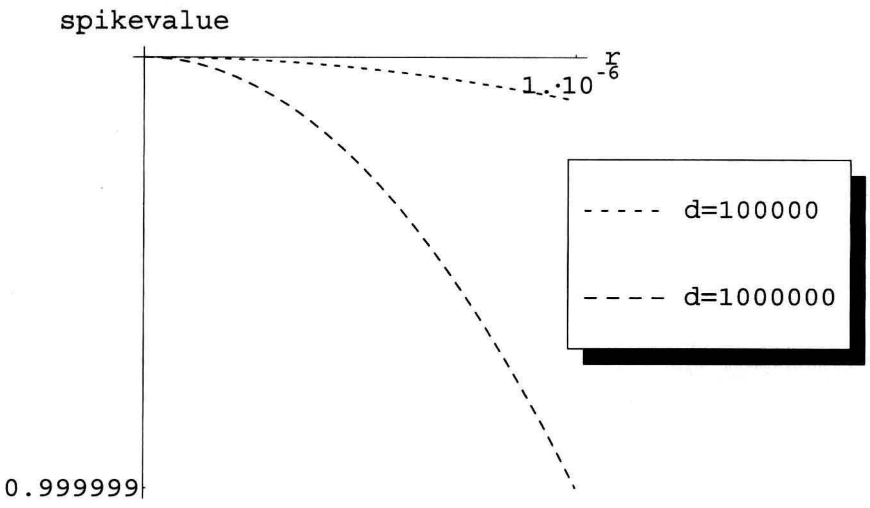

Figure 13. A closer look at the top of spikes.

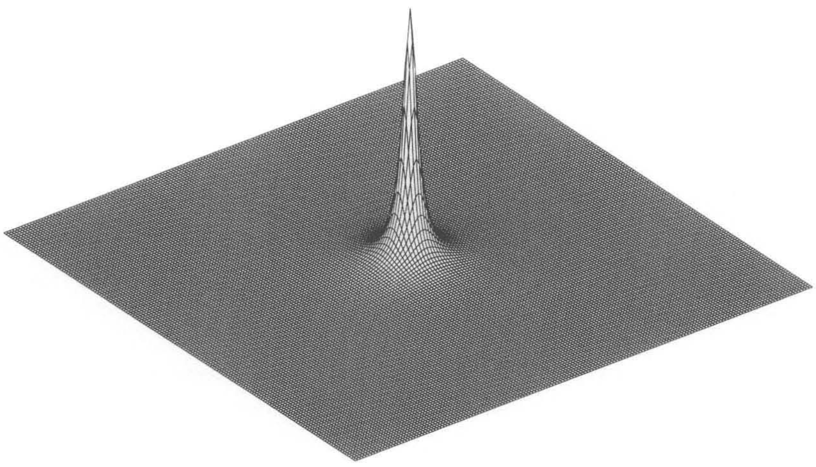

Figure 14. A spike with $d = 1000$.

References

[Blinn 82] Jim Blinn. "A generalization of algebraic surface drawing." *ACM Transactions on Graphics* 1(3): 235–256 (1982).

[Bloomenthal 95] Jules Bloomenthal. "Bulge elimination in implicit surface blends." In*Implicit Surfaces'95, The First International Workshop on Implicit Surfaces, Grenoble, France*, pp. 7–20, 1995.

[Bloomenthal 97] Jules Bloomenthal. *Introduction to Implicit Surfaces.* San Francisco: Morgan Kaufmann Publishers, Inc., 1997.

[Bourke 97] P. Bourke. http://astronomy.swin.edu.au/~pbourke/modelling/implicitsurf/ (1997).

[Glassner 01] A. Glassner. "Tricks of the trade. Andrew Glassner's Notebook." *IEEE Computer Graphics and Applications* 21(2): 80–87 (2001).

[Hartman 01] E. Hartmann. "Parametric G^n blending of curves and surfaces." *The Visual Computer* 17: 1–13(2001).

[Muraki 91] M. Muraki. "Volumetric shape description of range data using 'Blobby Model."' *Computer Graphics* 25: 227–235 (1991).

[Perlin 01] K. Perlin. *Personal communication* (2001).

[Wyvill et al. 86] G. Wyvill, McPheeters, B. Wyvill. "Data structure for soft objects." *The Visual Computer* 2: 227–234 (1986).

[Wyvill, Wyvill 89] B. Wyvill, G. Wyvill. "Field functions for implicit surfaces." *The Visual Computer* 5:75–82 (1989).

Web Information: http://www.acm.org/jgt/papers/BaranoskiRokne01

Gladimir V. G. Baranoski, Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1 (gvgbaranoski@cgl.uwaterloo.ca)

Jon Rokne, Department of Computer Science, The University of Calgary, Calgary, Alberta, Canada T2N 1N4 (rokne@cpsc.ucalgary.ca)

Received May 21, 2001; accepted in revised form February 1, 2002.