Advanced modeling

Andreas H. König

March 9, 2009

1 Sweep Representations

Solid-modeling packages often provide a number of construction techniques. Sweep representations are useful for constructing three-dimensional objects that possess translational, rotational, or other symmetries. We can represent such objects by specifying a two-dimensional shape and a sweep that moves the shape though a region of space. A set of two-dimensional primitives, such as circles and rectangles, can be provided for sweep representations as menu options. Other methods for obtaining two-dimensional figures include closed spline-curve constructions and cross-sectional slices of solid objects.

Figure 1 illustrates a translational sweep. The periodic spline curve in Fig. 1(a) defines the object cross section. We then perform a translational sweep by

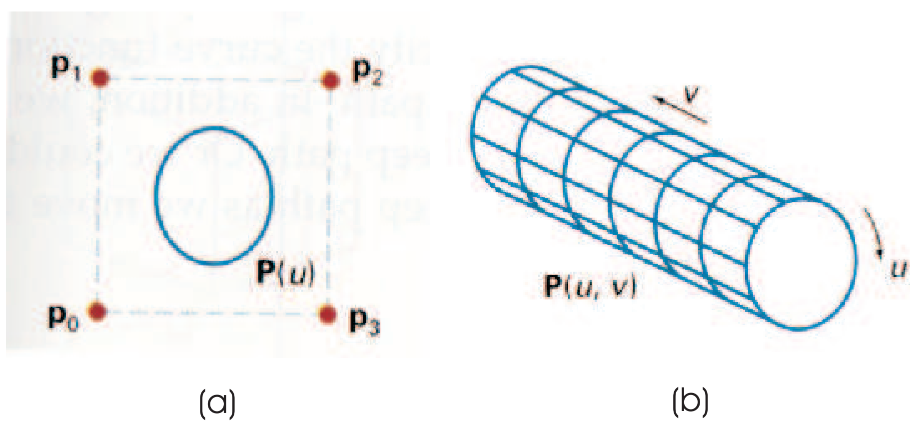

Figure 1: Constructing a solid with a translational sweep. Translating the control points of the periodic spline curve in (a) generates the solid shown in (b), whose surface can be described with point function P(u,v).

moving the control points p_0 through p_3 a set distance along a straight-line path perpendicular to the plane of the cross section. At intervals along this path, we replicate the cross-sectional shape and draw a set of connecting lines in the direction of the sweep to obtain the wireframe representation shown in 1(b).

An example of object design using a rotational sweep is given in Figure 2. This time, the periodic spline cross section is rotated about an axis of rotation

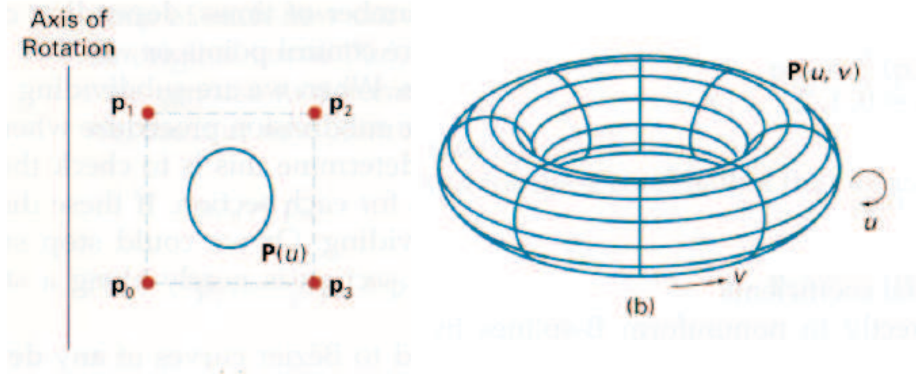

Figure 2: Constructing a solid with a rotational sweep. Rotating the control points of the periodic spline curve in (a) about the given rotation axis generates the solid shown in (b), whose surface can be described with point function P(u,v).

specified in the plane of the cross section to produce the wireframe representation shown in Figure 2(b). Any axis can be chosen for a rotational sweep. If we use a rotation axis perpendicular to the plane of the spline cross section in Figure 2(a), we generate a two-dimensional shape. But if the cross section shown in this figure has depth, then we are using one three-dimensional object to generate another.

In general, we can specify sweep constructions using any path. For rotational sweeps, we can move along a circular path through any angular distance from 0 to 360 degrees. For noncircular paths, we can specify the curve function describing the path and the distance of travel along the path. Figure 3 shows an example for this approach. In addition, we can vary the shape or size of the cross section along the sweep path. Or we could vary the orientation of the cross section relative to the sweep path as we move the shape through a region of space.

2 Blobby Objects

Some objects do not maintain a fixed shape, but change their surface characteristics in certain motions or when in proximity to other objects. Examples in this class of objects include molecular structures, water droplets and other liquid effects, melting objects, and muscle shapes in the human body. These objects can described as exhibiting ”blobbiess” and are often simply referred to as blobby objects, since their shapes show a certain degree of Liquidity. A molecular shape, for example, can be described as spherical in isolation but this shape changes when the molecule approaches another molecule. Distortion of the shape of the electron density cloud is due to the ”bonding” that occurs between the two molecules. Figure 4 illustrates the stretching, snapping, and contracting effects on molecular shapes when two molecules move apart. These characteristics cannot be adequately described simply with spheres or elliptical

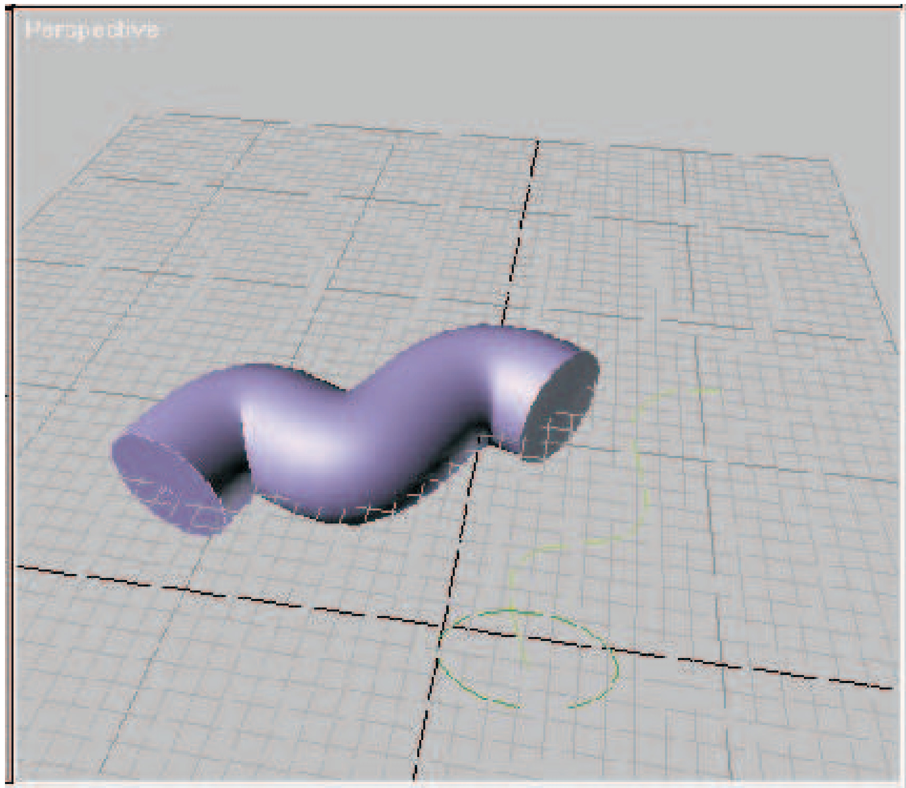

Figure 3: Example for a spline-based sweep object.

shapes. Similarly, Fig. 5 shows muscle shapes in a human arm which exhibit similar characteristics. In this case, we want to model surface shapes so that the total volume remains constant. Several models have been developed for representing blobby objects as distribution functions over a region of space. One way to do this is to model as combinations of Gaussian density functions, or "bumps" (Fig. 6). A surface function is then defined as

$$f(x, y, z) = \sum kb_k e^{-a_k r_k^2} - T = 0$$

where

$$r_k^2 = \sqrt{x_k^2 + y_k^2 + z_k^2}$$

and parameter T is some specified threshold, and parameters a and b are used to adjust the amount of blobbiness of the individual objects. Negative values for parameter b can be used to produce dents instead of bumps. Figure 6 illustrates the surface structure of a composite object modeled with four Gaussian density functions. At the threshold level, numerical root-finding techniques are used to locate the coordinate intersection values. The cross sections of the individual objects are then modeled as circles or ellipses. If two cross sections are near to each other, they are merged to form one blobby shape, as in Figure 4, whose structure depends on the separation of the two objects.

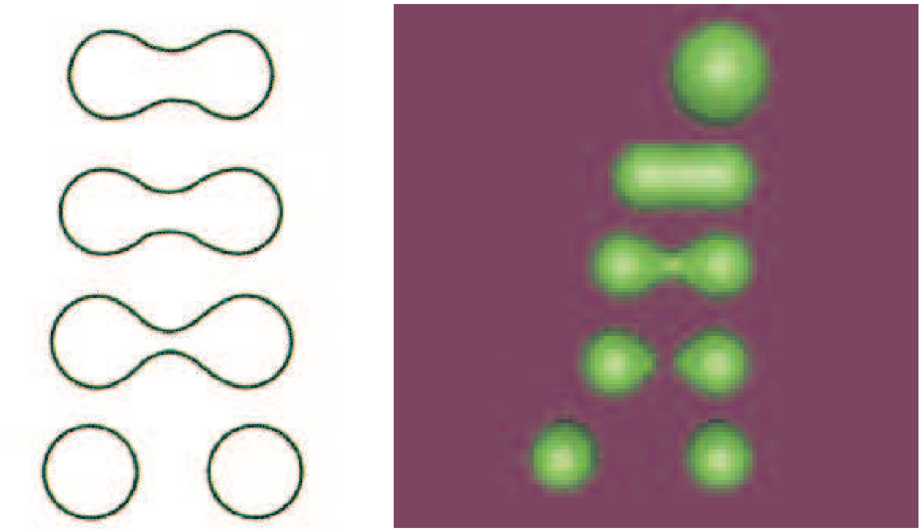

Figure 4: Molecular bonding. As two molecules move away from each other, the surface shapes stretch, snap, and finally contract into spheres.

Other methods for generating blobby objects use density functions that fall off to 0 in a finite interval, rather than exponentially. The "metaball" model describes composite objects as combinations of quadratic density functions of the form

$$f(r) = \begin{cases} b(1 - 3r^2/d^2), & if 0 < r \leq d/3 \\ \frac{3}{2}b(1 - r/d)^2, & if d/3 < r \leq d \\ 0, & if r > d \end{cases}$$

And the "soft object" model uses the function

$$f(r) = \begin{cases} 1 - \frac{22r^2}{9d^2} + \frac{17r^4}{9d^4} + \frac{4r^6}{9d^6} & if 0 < r \leq d \\ 0, & if r > d \end{cases}$$

Some design and painting packages now provide blobby function modeling for handling applications that cannot be adequately modeled with polygon or spline functions alone. Figure 7 shows a user interface for a blobby object modeler using metaballs.

3 Superquadrics

This class of objects is a generalization of the quadric representations. Superquadrics are formed by incorporating additional parameters into the quadric equations to provide increased flexibility for adjusting object shapes. The number of additional parameters used is equal to the dimension of the object: one parameter for curves and two parameters for surfaces.

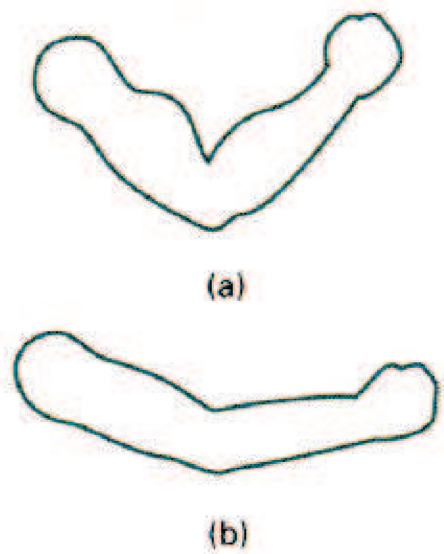

Figure 5: Blobby muscle shapes in a human arm.

3.1 Superellipse

We obtain a Cartesian representation for a superellipse from the corresponding equation for an ellipse by allowing the exponent on the x and y terms to be variable. One way to do this is to write the Cartesian superellipse equation in the form

$$\left(\frac{x}{r_x}\right)^{2/s} + \left(\frac{y}{r_y}\right)^{2/s} = 1$$

where parameter s can be assigned any real value. When s = 1, we get an ordinary ellipse.

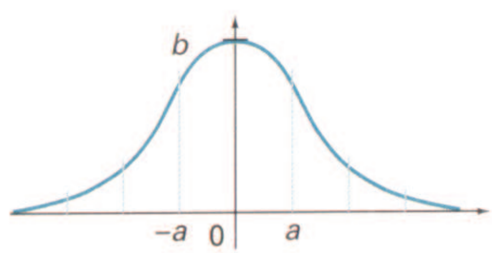

Figure 6: 3D Gaussian bump centered at position 0, with height b and standard deviation a.

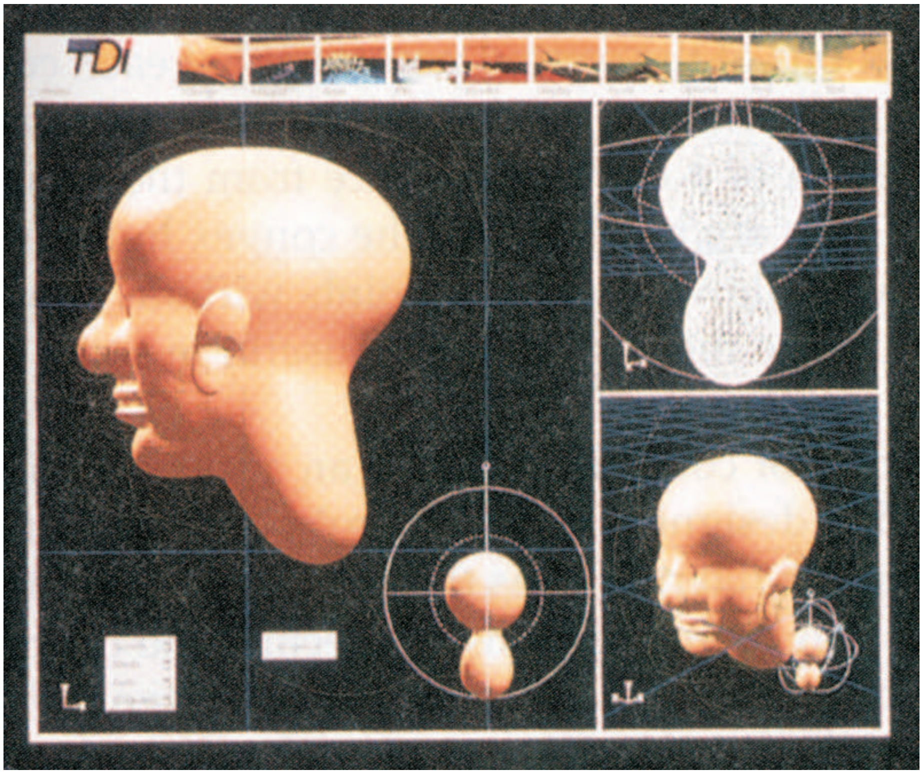

Figure 7: A screen layout, used the Blob Modeler and the Blob Animator packages, for modeling objects with metaballs. (Courtesy og Thomson Digital Image.) .

Corresponding parametric equations for the superellipse can be expressed as

$$x = r_x \cos^s \kappa, \quad -\pi \leq \kappa \leq \pi$$
$$y = r_y \sin^s \kappa$$

Figure 8 illustrates supercircle shapes that can be generated using various values for parameter s.

3.2 Superellipsoid

A Cartesian representation for a superellipsoid is obtained from the equation for an ellipsoid by incorporating two exponent parameters:

$$\left[\left(\frac{x}{r_x}\right)^{2/s_2} + \left(\frac{y}{r_y}\right)^{2/s_2}\right]^{s_2/s_1} = 1$$

For $s_1 = s_2 = 1$, we have an ordinary ellipsoid.

We can then write the corresponding parametric representation for the su-

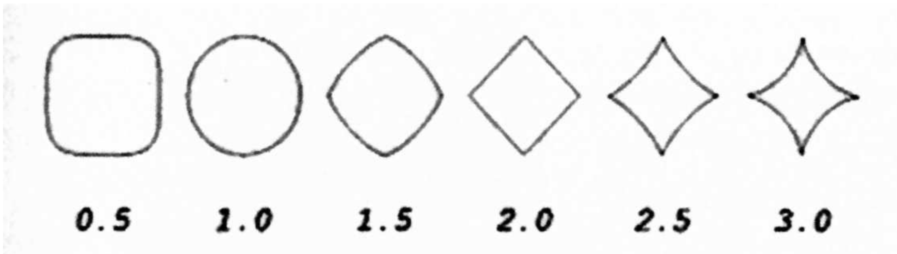

Figure 8: Superellipses plotted with different values for parameter s and with $r_x = r_y$.

perellipsoid as

$$\begin{aligned} x &= r_x \cos^{s_1}\phi \, cos^{s_2}\kappa, \quad -\pi/2 \le \phi \le \pi/2 \\ y &= r_y \sin^{s_1}\phi \, cos^{s_2}\kappa, \quad -\pi \le \kappa \le \pi \\ z &= r_z \sin^{s_1}\phi \end{aligned}$$

Figure 9 illustrates supersphere shapes that can be generated using various values for parameters s_1, and s_2. These and other superquadric shapes can be combined to create more complex structures, such as furniture, threaded bolts, and other hardware.

4 Particle Systems

A method for modeling natural objects, or other irregularly shaped objects, that exhibit "fluid-like" properties is particle systems. This method is particularly good for describing objects that change over time by flowing, billowing, spattering, or expanding. Objects with these characteristics include clouds, smoke, fire, fireworks, waterfalls, water spray, and clumps of grass. For example, particle systems were used to model the planet explosion and expanding wall of fire due to the "genesis bomb" in the motion picture *Star Trek II: The Wrath of Khan.*

Random processes are used to generate objects within some defined region of space and to vary their parameters over time. At some random time, each object is deleted. During the lifetime of a particle, its path and surface characteristics may be color-coded and displayed.

Particle shapes can be small spheres, ellipsoids, boxes, or other shapes. The size and shape of particles may vary randomly over time. Also, other properties such as particle transparency, color, and movement all can vary randomly. In some applications, particle motion may be controlled by specified forces, such as a gravity field.

As each particle moves, its path is plotted and displayed in a particular color. For example, a fireworks pattern can be displayed by randomly generating particles within a spherical region of space and allowing them to move radially, outward, as in Figure 10. The particle paths can be color-coded from red to yellow, for instance, to simulate the temperature of the exploding particles. Similarly, realistic displays of grass clumps have been modeled with "trajectory" particles (Fig. 11) that are shot up from the ground and fall back to earth under

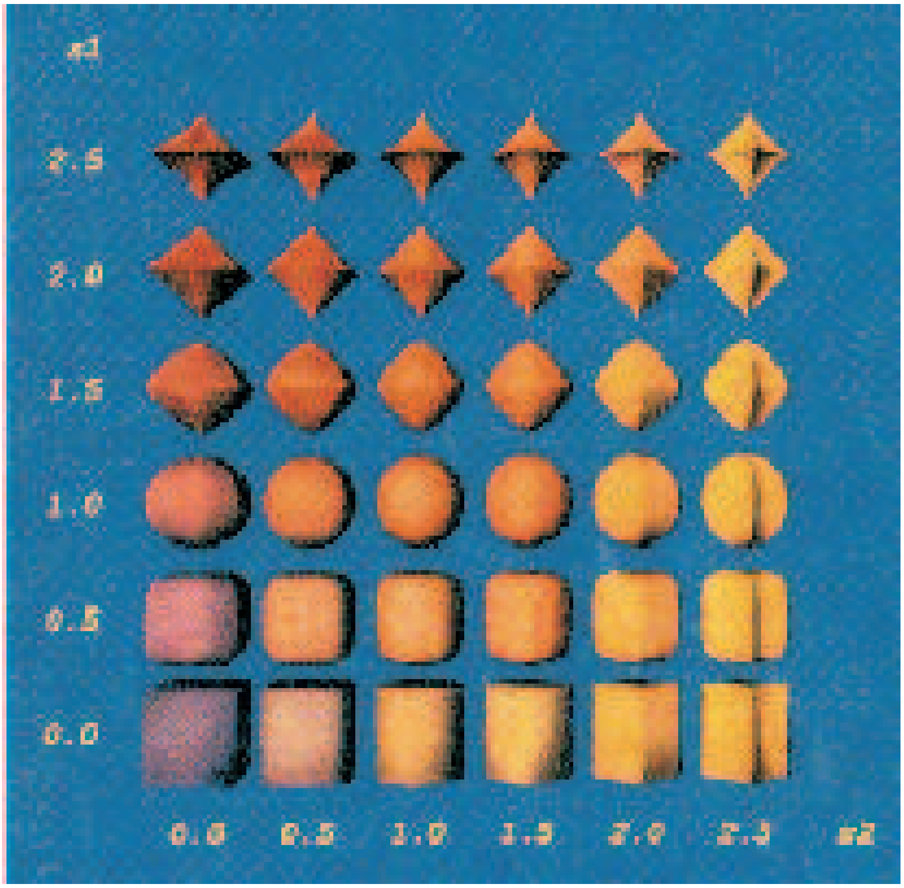

Figure 9: Superellipsoids plotted with different values for parameters $s_!$ and s_2, and with $r_x = r_y = r_z$.

gravity. In this case, the particle paths can originate within a tapered cylinder, and might be color-coded from green to yellow.

Figure 12 illustrates a particle-system simulation of a waterfall. The water particles fall from a fixed elevation, are deflected by an obstacle, and then lash up from the ground. Different colors are used to distinguish the particle paths at each stage. An example of an animation simulating the disintegration of an object is shown in Figure 13. The object on the left disintegrates into the particle distribution on the right. A composite scene formed with a variety of representations is given in Figure 14. The scene is modeled using particle-system grass, fractal mountains, and texture mapping and other surface-rendering procedures.

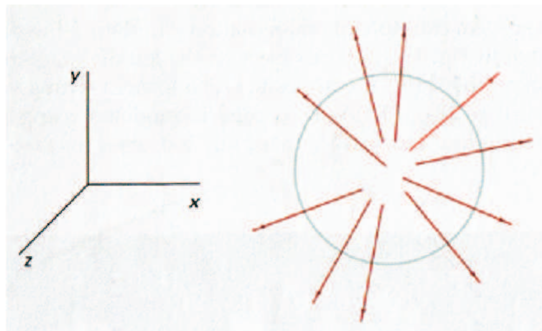

Figure 10: Modeling fireworks as a particle system with particles traveling radially outward from the center of the sphere.

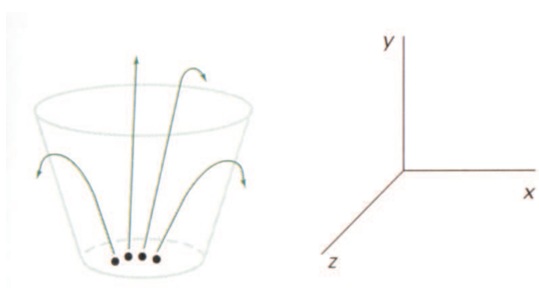

Figure 11: Modeling a clump of grass by firing particles upward within a tapered cylinder. The particle paths are parabolas due to the downward force of gravity.

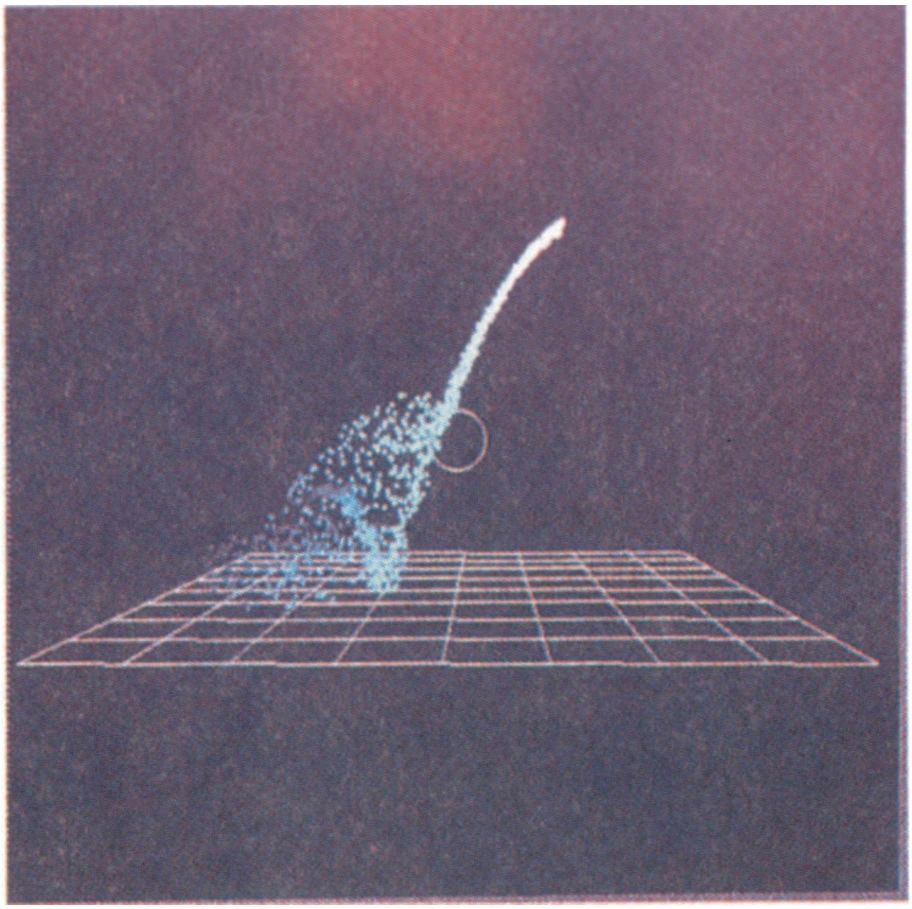

Figure 12: Simulation of the behaviour of a waterfall hitting a stone (circle). The water particles are deflected by the stone and then splash up from the ground. (Courtesy of M.Brooks and T.L.J. Howard, Department of Computer Science, University of Manchester.)

Figure 13: An object disintegrating into a cloud of particles. (Courtesy of Autodesk, Inc.)

Figure 14: A scene, entitled Road to Point Reyes, showing particle system grass, fractal mountains, and texture mapped surfaces. (Courtesy of Pixar. Copyright 1983 Pixar.)