IMPLICIT SURFACE PAINTING

Ergun Akleman

Visualization Laboratory,
216 Langford Center,
College of Architecture
Texas A&M University
College Station, TX 77843-3137
ergun@viz.tamu.edu

ABSTRACT

Implicit painting is a non-photorealistic rendering method for painting implicit surfaces. The method is based on the fact that when a difference equation is applied to a set of particles, these particles will move in 3D space. The motion of the particles is viewed as the motion of the hands of several painters and the trajectories of the particles as long unbroken brush strokes, over the implicit surfaces. These surfaces are used as if they are the canvases of painters. We consider implicit surface painting as a creative or 'artistic' process in which the resulting artwork can be an image, a stereo image or even an animation that shows the painting process.

1. INTRODUCTION

One of the most important goals in computer graphics is the development of new concepts and techniques for creating artistic [9, 10] or natural looking [6, 8] images and shapes. We believe the process of oil painting gives us a good paradigm to obtain both artistic or natural looking images. Some of the richness of painted images comes from the application of many layers of different paint over the canvas. Since each layer of paint goes on over the previous layers, the appearance of a painting implicitly contains the history of painting process. In other words, D'arcy Thompson's comments [12] that organic forms are *events in space-time* apply equally well to the appearance of paintings. Our goal is to obtain the same kind of richness that comes from the hard work of the artist by creating continuous motion of brushes.

Meier obtained a painting effect by attaching brush images to static particles [10]. In this work, we attach objects, instead of images, to brushes (particles) relying on the continuous motion of brushes to produce the desired look. We have already developed a method for painting boundaries of CSG solids [1]. In this paper, we generalize this earlier idea into general implicit surfaces and simplify and improve the motion equations. The new equations provide more intuitive control to the painter.

2. IMPLICIT PAINTING

Painters work in highly individual ways. They do not stop painting until they are satisfied; they may use different brushes, different paints. These decisions of painters are subjective. Some painters even leave most of the canvas unpainted by stopping after a few brush strokes. On the other hand, some painters continuously apply new layers covering the canvas with several layers of paint. Our goal is to paint implicit surfaces like oil painters paint their canvases. We assume that implicit surfaces are provided to painters as if they are custom-made canvases. The painters control only the motion of a set of virtual brushes. These brushes continuously move on the implicit surface until the painters are satisfied with the results.

The choice of implicit surfaces as canvases is not arbitrary. Characteristics of implicit representations are extremely suitable for development of equations for moving brushes. We must first introduce the implicit surfaces $\mathcal{S}(f)$, by relating them to their implicit equations as

$$\mathcal{S}(f) = \{\ p\ \mid\ f(p) = 0\ \}.$$

In other words, the surface $\mathcal{S}(f)$ consists of all points $p = (x, y, z)$ that are zeros of the function f from $\Re^3$ to $\Re$.

We need a set of equations to control the motion of the virtual brushes on the surface of $\mathcal{S}(f)$. We have observed that the power and simplicity of turtle geometry [2] comes from that separation of velocity vector into speed and direction. We suggest using a similar approach to define the motion. In other words, the current state of brush will be defined as a triplet $(p(t), s(t), n[g(t)])$ where the integer t denotes the current time, the point $p(t)$ denotes the current position of the brush, unit vector $n[g(t)]$ denotes the current direction of the motion and the real number $s(t)$ denotes the current speed and $g(t)$ denotes

all the control parameters that affect the direction of the brush, including the layered surface on which the brush moves. We adopt the convention that time advances in unit steps, so that $t-1$ denotes the previous time and $t+1$ denotes next time. The next position of the brush, $p(t+1)$, is given by

$$p(t+1) = p(t) + s(t)\ n[g(t)]. \qquad (1)$$

If $s(t)$ and $n[g(t)]$ are arbitrarily chosen by painters, the brushes will roam in three-dimensional space without any goal. Instead, they need to be attracted by a implicit surface and should stay on the implicit surface when they reach it. To move the brushes towards the implicit surface, we need to provide equations for computing $s(t)$ and $n[x(t)]$. In addition, the painter should be able to control the speed and direction of motion of the brushes in a simple and intuitive way.

Like an actual brush, these *virtual* brushes, when they move, have to leave some paint behind them. This paint will be created by attaching *paint* objects, such as cylinders or quadrilaterals, to each moving brush. The attributes of the paint objects, such as sizes and material properties, will be functions of position and time. Painters will also be able to control these attributes. By changing these attributes, it is possible to obtain several *paint* effects. For instance, depending on the radius and connectivity, cylinders create effects such as wire, toothpaste, rusted wire or tree branches. On the other hand, quadrilaterals can create the appearance of ribbon or paper and several cylinders together create classical oil paint effect.

To compute the orientations of the paints, it is necessary to have a coordinate system attached to each brush. Let three orthonormal vectors n_0, n_1 and n_2 describe this coordinate system for a given time t. Examples of attachment of different paint objects to the brushes by using these local coordinate systems are shown in Figure 1. A good choice of these vectors also simplifies implicit surface painting equations as shown in the following section.

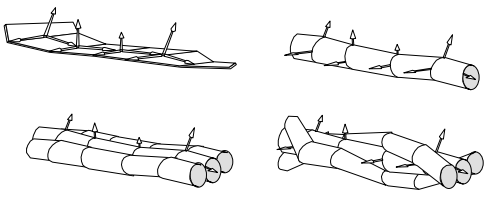

Figure 1: Examples of attachment of paint objects to the brushes.

2.1. Implicit Surface Painting Equations

Characteristics of implicit representations [5, 15, 4] are quite suitable for development of equations for brush motion.

- The gradient vector ∇f gives the normal vector to the surfaces $\mathcal{S}(f)$. We can use this normal vector to move in three-dimensional space to reach $\mathcal{S}(f)$. This idea is used by Bloomenthal [5] to sample implicit surface, $\mathcal{S}(f)$.

- Any vector perpendicular to ∇f will be on the tangent surface of $\mathcal{S}(f)$. By using this vector, it is possible to write an equation to make the brushes float over surfaces. Witkin and Heckbert [14] used this fact to achieve a uniform sample of a complex surface using local repulsion between *floating* particles and letting particles be born and die.

To develop implicit painting equations which will give control to the painter, we also use the gradient vector ∇f.

Let three orthonormal vectors n_0, n_1 and n_2 describe the coordinate system attached to the brush. In our painting system, we choose the gradient vector ∇f as one of these normal vectors

$$n_0 = \frac{\nabla f}{|\nabla f|}.$$

We compute the orthonormal vectors n_1 and n_2 by using the previous direction of brush motion $n(x(t-1))$ as

$$n_2 = \frac{n_0 \times n[g(t-1)]}{|n_0 \times n[g(t-1)]|},$$

$$n_1 = n_2 \times n_0.$$

The orthonormal vectors n_1 and n_2 are used to compute a goal direction r by using a rotation angle $\theta(t)$:

$$r = \cos\theta(t)\ n_1 + \sin\theta(t)\ n_2.$$

If we choose the current direction of brush motion $n[g(t)] = r$, the virtual brushes can never reach the implicit surface. We need another vector to attract the brushes towards the implicit surface. The vector $v(f)$ that is given as

$$v(f) = -\frac{f}{|\nabla f|} n_0 = -\frac{f \nabla f}{\nabla f \bullet \nabla f}$$

attracts the brushes towards the implicit surface at the direction of surface normal ∇f. This vector will be zero when the brushes are on the implicit surface. Thus, the current direction vector can be computed simply by normalizing the linear combination of the

vectors r and $v(f)$[1]

$$n[g(t)] = \frac{r + b(t)\ v(f)}{|r + b(t)\ v(f)|}, \tag{2}$$

where scaling factor $b(t)$ is a positive real number that can be controlled by the painter. It is possible to view $b(t)\ v(f)$ as a feedback term to make the particles attract and stay on the implicit surface $\mathcal{S}(f)$.

If we rewrite the equation 1, we obtain

$$p(t+1) = p(t) + s(t)\ n[f,\ \theta(t),\ b(t)]. \tag{3}$$

Equation 3 shows all variables that can be controlled by the painter for computing the next position of the brush. In equation 3, the value of $b(t)$ is not extremely critical, but there exists an interdependency between surface curvature, $s(t)$ and $b(t)$. Depending on the values of $s(t)$, $b(t)$ and the surface curvature, the brushes either converge towards the target surface, move in an orbit around the surface, or make a periodic motion as illustrated in Figure 2. Even if the brushes move on a distant orbit, the painters will not immediately notice any difference. However, if a periodic behavior occurs, it will be noticed immediately as a ripple. Such ripples which are the result of high curvature regions can be eliminated by choosing smaller values for either $s(t)$, $b(t)$ or both.

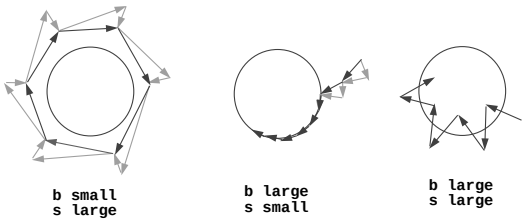

Figure 2: The relationship between $s(t)$ and $b(t)$ for a given curvature.

2.2. Motion Control

On a planar surface $\theta(t) = 0$ gives a straight line, while $0 < |\theta(t)| < \pi/2$ with constant $s(t)$ draws a regular pattern which approximates a circle. Since brushes have states and the length of $s(t)$ can change, even parametric L-systems [11] can be used to control brush motion to draw a tree-like curve over the implicit surface. Instead, we use two types of simple controls in our applications.

[1]The equation 2 can also be obtained by using Witkin and Heckbert's approach [14] and simplifying their motion equation. Since for implicit surface painting the motion of one brush does not have to be dependent on the positions and motions of other brushes, we do not use a local repulsion. In addition, shape parameters and time derivatives of the shape parameters can also be ignored since we do not have to change the shapes of the implicit surfaces. These eliminations make our painting equations extremely simple.

2.2.1. Random Walk

We obtain random walk by using the following equation

$$\theta(t+1) = \theta(t) + c\ \delta\theta, \tag{4}$$

where c is a uniform random number between 1 and -1 and scale factor $\delta\theta$ is a positive real number. The values $\delta\theta$ and initial angle $\theta(0)$ are provided by the painter. Likewise, $\delta\theta$ and $\theta(t)$ can be changed during the painting process. The value of scale factor $\delta\theta$ is essential for a good coverage of $\mathcal{S}(f)$. If $\delta\theta$ is very small, the brush draws similar curves without covering most of the surface. On the other hand, if $\delta\theta$ values are bigger than $\pi/4$, the brushes tend to stay in a local area. Figure 3 shows how the values of $\delta\theta$ can affect the uniformity of trajectory distribution. As known in sweeping surfaces, the angle $\theta(t)$ should never be bigger than $\pi/2$, since obtuse angles create overlapping and twisted paint objects as shown in Figure 4.

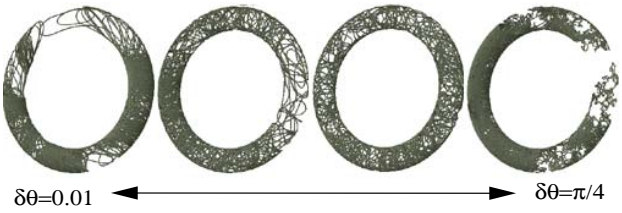

Figure 3: Trajectory of only one brush over a torus for different $\delta\theta$ values.

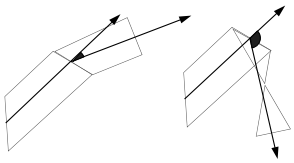

Figure 4: Angles bigger than $\pi/2$ create overlapping and twisted paint objects.

2.2.2. Periodic Curves

The effects different than random walk can be obtained by using time derivative function of a periodic curve in 2-dimensions. Let this derivative function be given by a parametric function $(x(t), y(t))$. Based on this parametric function our control parameters will be computed as

$$s(t) = \sqrt{x(t)^2 + y(t)^2}, cos\theta = \frac{x(t)}{s(t)}, \text{ and } sin\theta = \frac{y(t)}{s(t)}.$$

In our applications, we obtain derivative functions from periodic spline equations [3]. We also let the painter change $s(t)$ value. Changing $s(t)$ value results in scaling the periodic curve. Note that since the surface is not planar the motion will not be truly

periodic and the brush will not move on the surface drawing the same periodic curve unless $s(t)$ is very small. Currently, the periodic curves are hard-coded and users do not have explicit control over the shape of curve. In the near future, we want to provide such a control. An example of using periodic curves is shown in Figure 5. In this case, there are only ten brushes and each image is generated by a different set of initial conditions for brushes. The brushes eventually cover the surface, since $s(t)$ chosen is not small enough.

Figure 5: Curves on the surface of a toroid.

2.3. Paint control

Additional paint effects can be achieved by changing the sizes, the material properties and the types of the primitive objects with time. The sizes of the primitive objects can be changed simply by an additional equation. Material properties require a more involved process. One of the essential element in implicit surface painting is determination of the material properties of the paint objects that are left behind. In our applications, we only determine the diffuse color left behind by paint objects. The determination process consists of four stages.

- The first stage is to attach a diffuse color to each brush when brushes are initially created. The diffuse colors of brushes will be used to generate material properties of paint objects.

- The second stage is to compute the *original diffuse colors of paints* by adding noise to the material property of the brush. For instance, if the diffuse color of the brush is green, the diffuse color of a paint object attached to this particular brush may be generated by adding a little bit of blue or red to the green.

- The third stage is to compute the diffuse color of the three-dimensional position where the graphical object to be left. This diffuse color is computed using a function $C_d(p)$.

- The last stage is to compute the diffuse color of the paint object is to be left behind. This diffuse color is computed as a weighted average of the diffuse color of the three-dimensional position and the original diffuse color of the paint objects. The painter can change the weights of the interpolation function to control the amount of noise.

Although the original diffuse color of the paints may differ, when they pass thorough the neighborhood of a particular point they will all share some of the diffuse color of that point. The layers of paints which share the same point's diffuse color create a mix material whose diffuse color tends toward the diffuse color of that point. The process above can easily be extended to include other shader coefficients such as those for specular-reflection k_s and diffuse-reflection k_d.

3. IMPLEMENTATION

Earlier figures shows the viability of the method for painting toroidal implicit surfaces. We also used this method to paint more complicated surfaces. In our examples in this paper, we considered only blobby surfaces and spline based implicit surfaces.

The blobby surfaces we use are given by a generalized version of Blinn's exponentials [4]. For blending of several simple surfaces, we use

$$\text{Blend}(\cup_{j=0}^{J}\mathcal{S}(f_j) - \cup_{l=J}^{J+L}\mathcal{S}(f_l)) =$$

$$\mathcal{S}(1 - \sum_{j=0}^{J} e^{-f_j} + \sum_{l=J}^{J+L} e^{-f_l}).$$

If we know the derivatives of f_j's and f_k's, finding derivatives of these combined functions is straightforward. Since $\mathcal{S}(F)$'s are generally primitive surfaces, it is easy to implement these operations. If the blended surfaces have different material properties, material property of a given point is determined by blending their material properties using a weighted average of the diffuse colors of primitive implicit surfaces. The Figures 6 and 7 show implicit painting of exponential surfaces by using random walk. Different painting styles are obtained by changing the control parameters in equations 3 and 4. We also changed the sizes and types of paints and viewing positions. The figure 8 shows examples of the cases when brush motion is given by periodic curves.

The spline based implicit surfaces are obtained by using β-spline functions [3] which approximate randomly generated volume data. Since β-splines provide additional controls β_1 and β_2, we find them useful to approximate volume data.

Let $h(x, y, z)$ be β-spline function, the surface is described as $\mathcal{S}(h(x, y, z) - \epsilon)$ where ϵ is a given threshold value. Diffuse colors of voxels are also randomly generated and diffuse color of a given point, $C_d(p)$, is computed by using β-spline approximation of material properties of voxels. Examples of painting β-spline surfaces with random walk are shown in Figure 9.

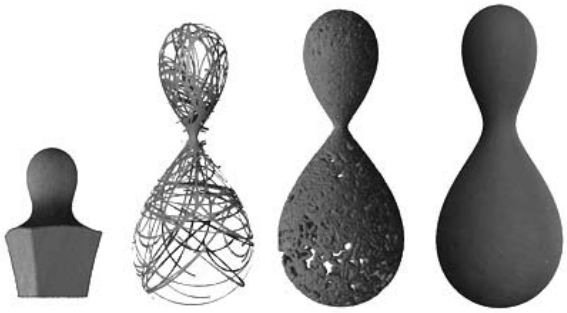

Figure 6: Blending the union of cube and sphere and and union of two spheres.

4. DISCUSSION

Finishing one painting depends on the chosen control parameters of equations 3 and 4. Small $s(t)$ values provide a better approximation of the surface but it takes more time to cover the whole surface. Likewise, choice of $\delta\theta$ in equation 4 effects the time to cover the surface. In addition, the surface area of implicit surfaces and paint size effect the coverage time. Our prototype painter is a simple menu driven interactive system which is implemented in C using the GL graphics library. The system also supports stereo viewing with stereo glasses. Each one of the images in the figures in this paper were created in less than one minute on an SGI O2 workstation. They are direct screen captures during the painting process.

Since we have provided complete control of brush speed, $s(t)$, to the painter, the implicit surface painting equations presented in this paper do not perform well around the sharp edges resulting from set operations. The early equations we have developed for painting CSG solids give better results in such situations [1].

In our system, it is possible to get a well-finished image by covering the surface completely with slower speed and smaller brush and paint sizes. However, exactly like classical painting, the results will look like overworked paintings. Although there is a close relationship between volume painting and classical painting, volume paintings are different than classical paintings in three ways.

- When painters do not paint a portion of the canvas we can still see the canvas. However, in a implicit surface painting, if a painter does not paint a certain portion of a surface it leaves a hole. A similar effect has been achieved by Marcel Duchamp by painting a large glass [7].
- These paintings can be considered as sculptures which can be viewed in three-dimensions using stereo views. Our prototype system provides stereo viewing. Such stereo views especially reveal the potential of these paintings as a new painting paradigm. The paintings can also be transformed into stereo photographic prints.

5. FUTURE WORK

In the current implementation of our painting system, we do not preserve geometry information for later use. Instead, we only preserve the images we like. In the future, we want to preserve the scene descriptions with detailed geometry and shading information instead of images. These scene descriptions could be rendered by using different renderers to get additional effects.

In the current system each brush behaves individually. To obtain well distributed brush strokes or constrained motion paths, the motion of different brushes must be coupled by using constraint based techniques similar to the techniques of Witkin and Heckbert [14]. Interrante [13] uses principal curvature to define paths for line integral. We also plan to provide a similar option to use principal curvature to control the path of the brush motion.

We plan to use only spline based implicit surfaces to unify surface representations. Other functions will simply be sampled and represented as volume data. Then, regardless of implicit function, the shape information will be provided as volume data and the painting program will simply use a spline function. In this work, we do not mix paint. However, when we work on volume data Interrante's [13] 3D line integral convolution may be applied to create mixture of paints.

6. ACKNOWLEDGMENTS

This work is supported by *Texas A&M University Program to Enhance Scholarly and Creative Activities.* We are also grateful to Don House, John Ferguson, Lori Cagle and reviewers for their helpful suggestions.

7. REFERENCES

[1] Ergun Akleman, "Implicit Painting of CSG Solids", *Proceedigns of CSG'98,* April 1998.

[2] H. Abelson and A. A. diSessa, "Turtle Geometry", *M.I. T. Press,* Cambridge, 1982.

[3] R. H. Bartels, J. C. Beatty and B. A. Barsky, "An Introduction to Splines for use in Computer Graphics and Geometric Modeling", *Morgan Kaufman Publishers,* California, 1987.

[4] J. I. Blinn, "A Generalization of algebraic surface drawing", *ACM Transactions on Graphics,* 1, 3, (235-256), 1982.

[5] J. Bloomenthal, "Techniques for implicit modeling", *ACM SIGGRAPH Tutorial 23: Modeling and Animating with Implicit Surfaces,* eds. B. Wyvill and J. Bloomenthal, (13.1-13.18), 1990.

[6] J. Dorsey and P. Hanrahan, "Modeling and rendering of metallic patinas", *Computer Graphics,* 30, 3, (387-396), 1996.
[7] M. Duchamp, "The Bride Stripped Bare by Her Bachelors, Even", *Painting made by oil, varnish, lead wire, lead foil and dust on two glass panels,* 1915-1923.
[8] K. Fleisher, D. Laidlaw, B. Currin and A. Barr, "Cellular Texture Generation", *Computer Graphics,* vol 29, no. 3, pp. 239-248, 1995.
[9] P. E. Haeberli, "Paint by Numbers", *Computer Graphics,* vol 24, no. 3, pp. 207-214, 1990.
[10] B. J. Meier, "Painterly rendering for animation", *Computer Graphics,* 30, 3, (477-486), 1996.
[11] P. Prusinkiewicz and A. Lindenmayer, *Algorithmic Beauty of Plants,* Springer Verlag, New York, 1990.
[12] d'Arcy Thomson, *On Growth and Form,* Universty Press, Cambridge, 1952.
[13] V. Interrante, "Illustrating Surface Shape in Volume Data via Principal Direction-Driven 3D Line Integral Convolution", *Computer Graphics,* 30, 3, (109-116), 1997.
[14] A. P. Witkin and P. S. Heckbert, "Using particles to sample and control implicit surfaces", *Computer Graphics,* 27, 3, (269-277), 1994.
[15] G. Wyvill and A. Trotman, "Ray tracing soft objects", *ACM SIGGRAPH Tutorial 23: Modeling and Animating with Implicit Surfaces,* eds. B. Wyvill and J. Bloomenthal, (13.1-13.18), 1990.

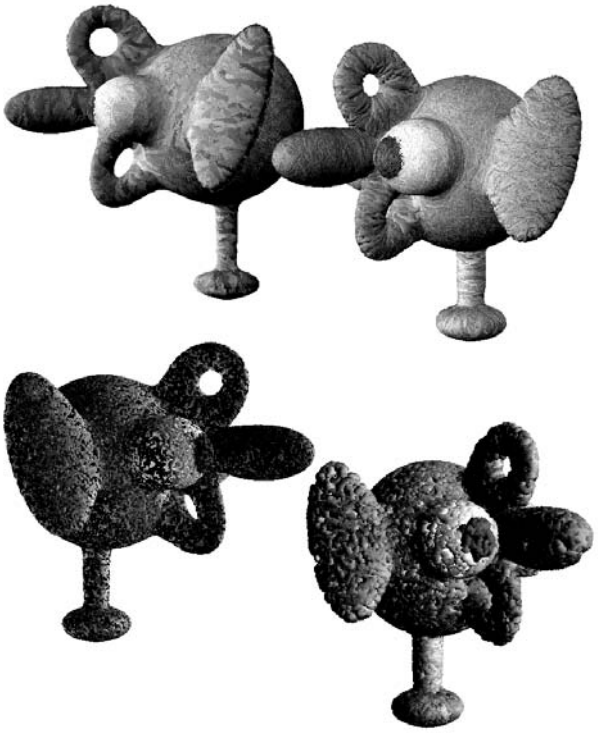

Figure 7: Blobby surreal chicken painted by random walk.

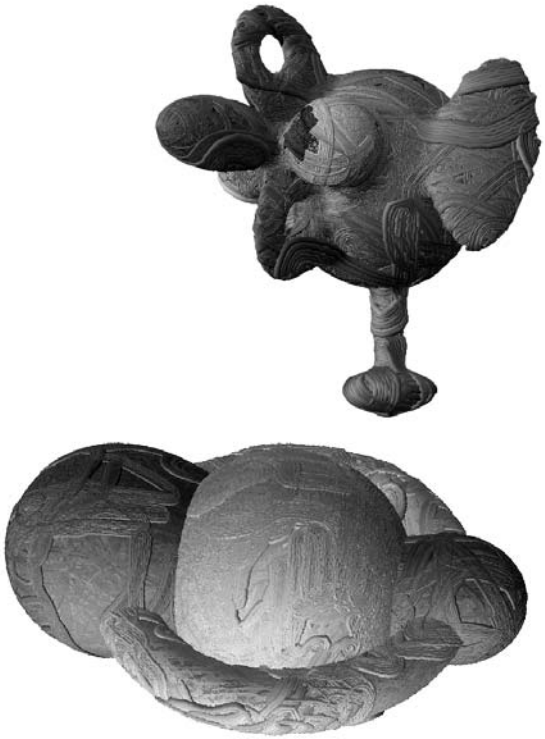

Figure 8: Examples of the effects obtained by using periodic curves to control brush motion.

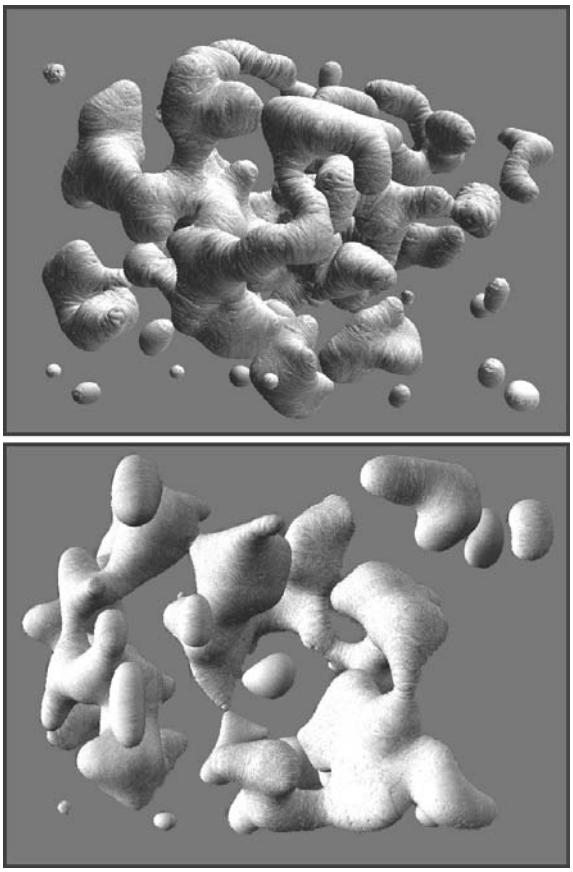

Figure 9: Examples of spline based random implicit surfaces painted by random walk.