Particle-Based Transparent Rendering of Implicit Surfaces and its Application to Fused Visualization

S. Tanaka1, K. Hasegawa1, Y. Shimokubo1, T. Kaneko1, T. Kawamura2, S. Nakata1, S. Ojima1, N. Sakamoto1, H. T. Tanaka1, K. Koyamada3

1College of Information Science and Engineering, Ritsumeikan University, Japan
2Center for Computational Science & e-Systems, Japan Atomic Energy Agency, Japan
3Institute for the Promotion of Excellence in Higher Education, Kyoto University, Japan

Abstract

We present particle-based surface rendering (PBSR) that realizes precise rendering of transparent implicit surfaces and flexible fused visualization. The PBSR uses small opaque particles as rendering primitives and creates transparent images with the correct depth feel without the necessity of particle sorting. The nonnecessity of sorting enables that the rendering stage of the PBSR can be executed with interactive frame rates. The formula derived based on binomial probability works well to control the surface opacity. The PBSR realizes precise and quick transparent rendering of highly nonlinear implicit surfaces, such as those constructed by interpolating 3D scattered points. The PBSR also realizes various types of fused visualization such as surface-surface fusion, surface-volume fusion, and fusion of a slice plane with a volume and a polygon mesh.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Display algorithms I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Visible line/surface algorithms

1. Introduction

This paper proposes a method for precise and flexible rendering of transparent implicit surfaces. We call our method “particle-based surface rendering” (PBSR). It uses small opaque particles as rendering primitives, and the particles are stochastically generated uniformly on the target surfaces.

The PBSR realizes precise rendering of transparent complex implicit surfaces without the necessity of extracting intermediate polygon meshes. The PBSR does not require sorting of the rendering primitives (particles) to realize the correct depth feel. Therefore, it is free from artifacts coming from sorting errors such as patchy patterns, and the rendering time is proportional to the number of particles. The PBSR is inspired by the particle-based volume rendering (PBVR) [KST08, SKKN10]. But the method to generate the particles and the theory to control opacity are very different.

A unique feature of the PBSR is that it is applicable to various types of precise “fused visualization”. Surface-surface fusion, surface-volume fusion, etc. are quite straightforward. To achieve fusion, we simply generate particles for each constituent object and merge the particles before going onto the rendering stage.

Let us briefly review the related works. One choice to render a transparent implicit surface is to extract an approximate polygon mesh. Once the surface is decomposed into a set of polygons, we can use one of the many established methods for transparent rendering, including the A-buffer method [Car84], the screen-door method [FGH∗85, MGvW98, ND93, ESSL11, SCG03], the depth-peeling method [Eve01, LXW06, BM08], the k-buffer method [BCL∗07], and the per-pixel linked list [YHGT10]. Another choice to render a transparent implicit surface is to use the ray tracing method. This choice is suitable for precise rendering due to the calculations of exact ray-surface intersections. Recent rapid development of the GPU enables real-time rendering of implicit surfaces defined by comparatively simple mathematical equations [PBMH02, GPSS07, KHK∗09]. Nevertheless, the GPU-based acceleration of ray tracing does not work sufficiently for more complex implicit surfaces, such as the ones constructed by interpolating scattered 3D points [TO02, OBA∗03, LB06].

The particle-based approach has been studied for render-
26

S. Tanaka, K. Hasegawa, et al. / Particle-Based Transparent Rendering of Implicit Surfaces

ing of “opaque” implicit surfaces [WH94, dFG96, MGW05, TSYK01, OGG09, OAG10, BWWM10]. A pioneer work that applies the particle-based approach to transparent surfaces is the elliptical weighted average (EWA) splatting, which is also applicable to fused visualization [ZPvBG02]. EWA splatting is different from our PBSR in that it requires sorting of finite-size particles. Therefore it cannot be free from artifacts for very complex surfaces.

2. General view of the PBSR

A particle used in the PBSR is defined as small opaque 3D object with the following properties: (1) 3D position of the center, (2) particle color, (3) particle cross section, and (4) normal vector. The particle cross section is tuned so that an image of a particle just overlaps one pixel on the image plane. The normal vector is used for shading.

The following three steps realize transparent surface visualization. The origin of the transparency is the probabilistic determination of pixel colors.

STEP 1. Particle generation: The first step is to generate particles on the target surface. The particles should be distributed with uniform density on the surface. We repeat the particle distribution until \( L_k \) statistically independent particle sets are prepared. We call \( L_k \) the “repeat level”.

STEP 2. Particle projection with occlusion: The second step is to project the generated particles to the image plane. Since the opaque property is assigned to the particles, we perform particle occlusion during the projection. Namely, at each pixel, we save the color of the one projected particle closest to the camera position. Such particle projection with the occlusion effect is executed for each particle set prepared in STEP 1. As the result, \( L_k \) similar images are created.

STEP 3. Averaging images: The final step is to make an average image of the \( L_k \) images created in STEP 2. At each pixel on the image plane, pixel colors of the \( L_k \) images are averaged to determine the final pixel color.

Here we explain how to execute "fused visualization". Usually, it is complicated to execute fused visualization of different types of objects. Because we need to use a different visualization algorithm for each type. In the PBSR, however, we can use the same common procedure for all objects. Namely, a visualized object is converted to a set of particles (STEP 1) and then rendered (STEP 2 and STEP 3). Therefore, once the particle sets are created for constituent objects in STEP 1 as preprocessing, we can easily create a combined set and use it in STEP 2 and STEP 3 to create the fused view.

3. Probabilistic determination of surface opacity

In this section we explain a theory to control surface opacity. The theory is based on the assumption that particles are generated with uniform density on the visualized surface.

First, let us summarize mathematical notation and related setup. We consider a small local surface segment with area \( S \) on the visualized surface (see Figure 1). We presume that the surface segment is approximately a flat plane. Let \( n \) be the number of particles generated uniformly within the surface segment, and let \( N \) be the number of pixels inside the image of the segment. That is, for each independent particle set created in STEP 1, we generate \( n \) particles uniformly, and each particle is projected onto one of the \( N \) pixels on the image plane. For each particle, we assign particle cross section \( s \), which is tuned so that its image just overlaps one pixel. (In perspective projection, \( s \) is made dependent on view depth of each particle.) Then \( N \) is related to \( S \) and \( s \) as \( N = S/s \).

For simplicity, we consider the case that the surface segment is parallel to the image plane. Let us focus on an arbitrary pixel taken from the \( N \) pixels. For the focused pixel, we investigate the number of projected particles, \( x \). Note that \( x \) is an integer random variable in the range \( 0 \leq x \leq n \). Because of the uniformity, the probability that a particle on the surface segment is projected to this pixel is \( 1/N \). Then \( x \) should obey the probability function of the binomial distribution \( B(n, 1/N) \): \( p(x) = nC_x(1/N)^x(1 - 1/N)^{n-x} \). The probability that at least one particle is projected to the focused pixel is

\[
\alpha \equiv 1 - p(0) = 1 - (1 - 1/N)^n = 1 - (1 - s/S)^n. \quad (1)
\]

This \( \alpha \) is the probability that the focused pixel becomes the particle color. In the procedure of the PBSR, we prepare \( L_k \) images by projecting statistically independent \( L_k \) particle sets. The final pixel colors are determined by calculating the averages of these images. Thus, \( \alpha \) of Equation (1) functions as the opacity of the focused pixel. Since the focused pixel is arbitrary, this \( \alpha \) becomes opacity of the surface segment.

For a surface segment nonparallel to the image plane, its image size becomes smaller. Then the surface opacity is the following straightforward generalization of Equation (1):

\[
\alpha(\theta) \equiv 1 - (1 - s/S \cos \theta)^n, \quad \text{where } \theta \text{ is the angle between the surface segment and the image plane.}
\]

4. Uniform sampling of visualized surfaces

Here we explain how to execute uniform sampling of implicit surfaces. (Here “sampling” means particle generation.) There are many algorithms for this purpose [WH94,
MGW05, TSYK01, OGG09, OAG10, BWWM10]. Here we adopt stochastic sampling [TMN00, TSYK01] that is improved for the PBSR.

We consider an implicit surface \( I_f \) defined by the mathematical equation, \( f(x) = c \), where \( f(x) \) is a continuous scalar field, \( c \) is an isovalue, and \( x = (x_1, x_2, x_3) \) is a position vector. In the case that \( I_f \) is an isosurface of volume data, \( f(x) \) is created by properly interpolating the data.

Stochastic sampling uses a hypothetical Brownian motion confined to \( I_f \) (see Figure 2). This Brownian motion is defined by the following stochastic differential equation:

\[
dx[i]{x}(t) = dx[i]{x}(T)(t) + dx[i]{x}(S)(t), \quad f(x(t = 0)) = c,
\]

where \( t \) is a fictitious time variable. Each term on the right-hand side of Equation (2) is defined as follows:

\[
dx[i]{x}(T) = \sum_{j=1}^{3} P_{ij} dw[j], \quad dx[i]{x}(S) = -\frac{dt}{\| \nabla f \|^2} \left( \frac{\partial f}{\partial x_i} \right) Tr\{H}\}
\]

The term \( dx[i]{x}(T) \) generates random movement in the tangential direction of \( I_f \), and the term \( dx[i]{x}(S) \) incorporates the effects of surface curvature. \( P = \{P_{ij}\} \), \( P_{ij} \equiv \delta_{ij} - \left( \frac{\partial f}{\partial x_i} \right) \left( \frac{\partial f}{\partial x_j} \right) / \| \nabla f \|^2 \) is the projection matrix, which confines solutions of Equation (2) to \( I_f \). The vector \( (dw[1], dw[2], dw[3]) \) is the Gaussian white noise with the statistical properties, \( \langle dw[i](t) \rangle = 0, \langle dw[i](t) dw[j](t) \rangle = 2\delta_{ij} dt \).

By solving a discretized version of Equation (2) with Euler’s method, we collect its solutions as sampling points. Deviation of the discretized \( x(t + \Delta t) \) out of the curved \( I_f \) is always corrected using Newton’s method or the bisection method. For open surfaces, we introduce a proper boundary condition [TSYK01]. We tune the time step \( \Delta t \) so that the average size of \( dw[i] \), which is \( \sqrt{2\Delta t} \), is sufficiently small. For an analytically defined implicit surface, we normalize its approximate width to unity. Then we tune the value of \( \sqrt{2\Delta t} \) in the range between \( 10^{-2} \) and \( 10^{-3} \), depending on complexity of the surface. For an isosurface of volume data, we tune \( \Delta t \) so that \( \sqrt{2\Delta t} \) is 30% of the cell width.

We also need to define a termination condition of the stochastic sampling. As the number of generated particles, i.e., numerical solutions of Equation (2), increases, the opacity \( \alpha \) becomes larger (see Equation (1)). We terminate the sampling when the user-required opacity is realized in \( \alpha \).

For statistical accuracy we prepare 100 counting spheres at random positions on \( I_f \), and use an average \( n \).
Next we demonstrate the surface-volume fused visualization. Figure 5 is an example of visualizing a simulation result of radiation physics. The human breast (simulation target) is visualized as a volume by using the PBVR [KST08]. In addition, the dose distribution (simulation result) is visualized as multiple isosurfaces of the Catmull-Rom spline interpolation field [CR74] of the dose volume. The rendering parameters are $\alpha = 0.1$, $LR = 500$. The dose distribution is often visualized as contour lines drawn on a 2D medical slice image. Figure 5 is its natural 3D extension.

Suppose that a volume is the main visualization target and the isosurface need not be visualized very precisely. In such a case, we can execute STEP 1 (uniform surface sampling) through the Marching Cubes polygonization and the polygon-by-polygon sampling, which is easier than the stochastic sampling. Figure 6 (left) is an example that visualizes a human skull (a volume) and superposes a polygonized isosurface to show the human body boundary. The rendering parameters for the isosurface are $LR = 100$, $\alpha = 0.1$. Figure 6 (right) shows similar fused visualization of three different types of objects: a slice plane (the middle yellow plane) that is visualized with higher opacity, a volume (the red cloud), and a polygonized isosurface (the blue surface). This kind of visualization is useful, for example, in analyzing a slice image in detail, confirming its position in the volume space at the same time. The slice plane is sampled stochastically, treating it as an implicit surface that is defined by bilinearly interpolating original pixel colors.

A major drawback of the PBSR is long sampling time (STEP 1). It takes 3294, 3708, and 1449 sec for the sampling to create Figures 3 (right) ($6.3 \times 10^7$ particles), 4 (left) ($9.5 \times 10^7$ particles), and 4 (right) ($5.0 \times 10^7$ particles), respectively. Once the particles are prepared, the rendering stages (STEP 2 and STEP 3) can be executed with interactive frame rates. Rendering speeds are 3.2, 2.5, and 3.6 fps for Figures 3 (right), 4 (left), and 4 (right), respectively. The rendering time is proportional to the total number of particles that is proportional to $LR$ and the image resolution. (Our experiments used a Linux PC with a Core i7 980 processor, 3.0 GHz, 24 GB main memory, and Nvidia Geforce GTX580 GPU. The sampling was executed with the CPU computation, and the rendering with the GPU computation.)

6. Conclusions

We presented particle-based surface rendering (PBSR) for precise visualization of transparent implicit surfaces. The PBSR does not require sorting of rendering primitives. It makes the method free from unwanted artifacts and also realizes that the rendering stage of very complex surfaces is executed at least with interactive frame rates. These features of the PBSR are suitable for visualizing fine 3D inner structures of highly nonlinear implicit surfaces such as those constructed by interpolating 3D scattered points. The PBSR also realizes various types of fused visualization. Once a particle set is prepared for each candidate constituent object, we can select the proper particle sets to create their fused view. We demonstrated surface-surface, surface-volume, and slice-volume-polygon fusions. We can also quickly create animations, where constituent objects are changing their relative positions, without re-sampling. In such animations created by the PBSR, no moving artifacts should appear. We are now developing a parallel sampling scheme using GPU or multi-core processors. It is known that stochastic (Monte Carlo) simulation is suitable for parallel processing. It is also interesting to apply the PBSR to feature extraction based on non-uniform surface sampling.

The authors thank John Allison, Joseph Perl, Akinori Kimura, and Masaru Komori for their valuable suggestions.

\[\text{Figure 4: Surface-surface fused visualization of overlapping (left) and intersecting (right) implicit surfaces.}\]

\[\text{Figure 5: Surface-volume fused visualization (example 1).}\]

\[\text{Isosurfaces of dose distribution are fused with a human-breast volume.}\]

\[\text{Figure 6: Surface-volume fused visualization (example 2).}\]

\[\text{In the right figure, a slice plane is also fused additionally.}\]
References


© The Eurographics Association 2012.