Eurographics Conference on Visualization (EuroVis) (2012)
M. Meyer and T. Weinkauf (Editors)



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

S. Tanaka[1], K. Hasegawa[1], Y. Shimokubo[1], T. Kaneko[1], T. Kawamura[2], S. Nakata[1], S. Ojima[1], N. Sakamoto[3], H. T. Tanaka[1], K. Koyamada[3]

[1]College of Information Science and Engineering, Ritsumeikan University, Japan
[2]Center for Computational Science & e-Systems, Japan Atomic Energy Agency, Japan
[3]Institute 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, LWX06, 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-



DOI: 10.2312/PE/EuroVisShort/EuroVisShort2012/025-029

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 $L_R$ statistically independent particle sets are prepared. We call $L_R$ 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, $L_R$ similar images are created.

**STEP 3. Averaging images:** The final step is to make an average image of the $L_R$ images created in STEP 2. At each pixel on the image plane, pixel colors of the $L_R$ 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 size $S$ 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 $N = S/s$.

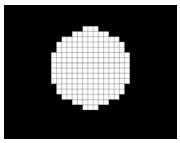

**Figure 1:** *The white circle shows an example of the local surface segment. Its area size is S and its image consists of N pixels (squares). Each particle has size s and its image just overlaps one of the squares.*

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 $0 \le x \le n$. 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 $\mathrm{B}(n, 1/N)$: $p(x) = {}_n\mathrm{C}_x (1/N)^x (1-1/N)^{n-x}$ . 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 $L_R$ images by projecting statistically independent $L_R$ 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): $\alpha(\theta) \equiv 1 - (1 - s/S\cos\theta)^n$ , where $\theta$ 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 [TMN*00, TSYK01] that is improved for the PBSR.

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

Stochastic sampling uses a hypothetical Brownian motion confined to $\mathrm{I}_f$ (see Figure 2). This Brownian motion is defined by the following stochastic differential equation:

$$dx_i(t) = dx_i^{(\mathrm{T})}(t) + dx_i^{(\mathrm{S})}(t)\ , \quad f(\mathbf{x}(t=0)) = c\ , \qquad (2)$$

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

$$dx_i^{(\mathrm{T})} = \sum_{j=1}^{3} P_{ij}\, dw_j\ , \quad dx_i^{(\mathrm{S})} = -\frac{dt}{\|\nabla f\|^2}\left(\frac{\partial f}{\partial x_i}\right)\mathrm{Tr}\{\mathbf{HP}\}. \qquad (3)$$

The term $dx_i^{(\mathrm{T})}$ generates random movement in the tangential direction of $\mathrm{I}_f$, and the term $dx_i^{(\mathrm{S})}$ incorporates the effects of surface curvature. $\mathbf{P} = \{P_{ij}\}$, $P_{ij} \equiv \delta_{ij} - (\partial f/\partial x_i)(\partial f/\partial x_j)/\|\nabla f\|^2$ is the projection matrix, which confines solutions of Equation (2) to $\mathrm{I}_f$. The vector $(dw_1, dw_2, dw_3)$ is the Gaussian white noise with the statistical properties, $\langle dw_i(t)\rangle = 0$, $\langle dw_i(t) dw_j(t)\rangle = 2\delta_{ij}dt$. $\mathbf{H} = \{H_{ij}\}$, $H_{ij} \equiv \partial^2 f/\partial x_i \partial x_j$ is the Hessian matrix.

By solving a discretized version of Equation (2) with Euler's method, we collect its solutions as sampling points. Deviation of the discretized $\mathbf{x}(t+\Delta t)$ out of the curved $\mathrm{I}_f$ 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 $\alpha$. We consider a small hypothetical sphere whose center is on $\mathrm{I}_f$. We call this sphere the "counting sphere" (see Figure 2). The great circle overlapping $\mathrm{I}_f$ is adopted as the local surface segment (see Section 3), as in Figure 1. During the sampling we count the number of particles generated inside the counting sphere. The counted number is used as $n$ and the great-circle area of the counting sphere is $S$ in Equation (1). For statistical accuracy we prepare 100 counting spheres at random positions on $\mathrm{I}_f$, and use an average $n$.

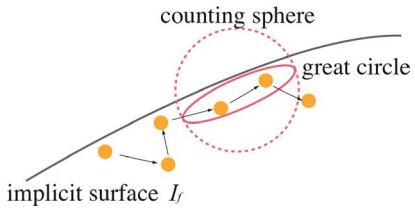


**Figure 2:** *Stochastic sampling and the counting sphere.*

## 5. Experiments

Here we demonstrate the effectiveness of the PBSR. Image resolution is set to $1024^2$ for all the presented images.

The PBSR realizes precise visualization of nonlinearly interpolated volume data. Figure 3 (left) visualizes five isosurfaces of the trilinear interpolation field within one cube. The curved isosurfaces are visualized directly without polygonization. Figure 3 (right) visualizes non-polygonized iso-

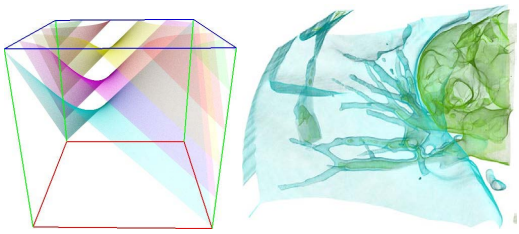

**Figure 3:** *Non-polygonized isosurface visualization of nonlinearly interpolated scalar fields.*

surfaces of a human breast with $\alpha = 0.1$, $L_R = 500$. The Catmull-Rom spline interpolation [CR74] is adopted.

Next we demonstrate the surface-surface fused visualization. Figure 4 (left) is an example of fusing surfaces of internal human organs. The rendering parameters are $\alpha = 0.1$, $L_R = 500$. For each organ, a non-polygonized zero isosurface of a proper scalar field $f(\mathbf{x})$ is constructed from scattered 3D points by using the multi-level partition of unity (MPU) method [OBA*03]. Each constructed isosurface is sampled with the stochastic sampling. The time step is tuned as $\Delta t = 1.0 \times 10^{-5}$ in units of the diagonal-line length of the bounding box of the input scattered 3D points. The created particle sets are merged for the fused visualization. In Figure 4 (left) each organ surface partially overlaps neighboring ones. In such cases, the conventional methods often suffer from artifacts due to ambiguity in sorting rendering primitives. The PBSR is, however, free from such artifacts. The rendered surfaces can even intersect (see Figure 4 (right) ). The Asian Dragon surface (the Stanford 3D Scanning Repository) is constructed from scattered 3D points by using the MPU method. The wave surface is defined using cosine functions analytically. The repeat level is $L_R = 500$, and the total number of particles is $5.0 \times 10^7$.

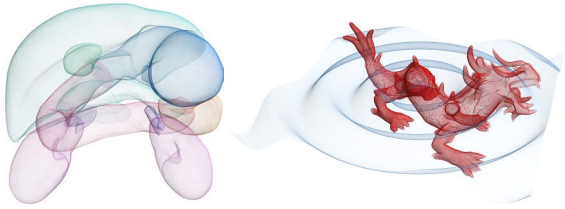

**Figure 4:** *Surface-surface fused visualization of overlapping (left) and intersecting (right) implicit surfaces.*

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 $\alpha = 0.1$, $L_R = 500$. The dose distribution is often visualized as contour lines drawn on a 2D medical slice image. Figure 5 is its natural 3D extension.

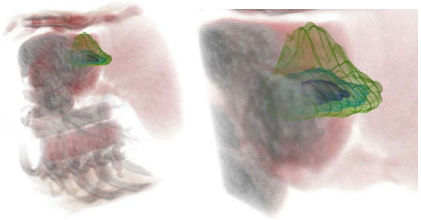

**Figure 5:** *Surface-volume fused visualization (example 1). Isosurfaces of dose distribution are fused with a human-breast volume.*

Suppose that a volume is the main visualization target and that 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 $L_R = 100$, $\alpha = 0.1$. 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

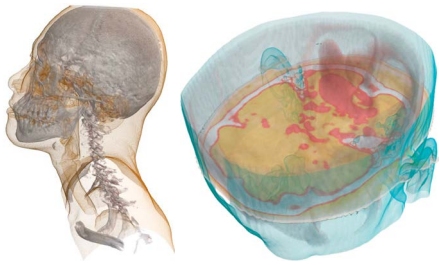

**Figure 6:** *Surface-volume fused visualization (example 2). In the right figure, a slice plane is also fused additionally.*

(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 $L_R$ 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.

## References


[BCL*07] BAVOIL L., CALLAHAN S. P., LEFOHN A., COMBA J. L. D., SILVA C. T.: Multi-fragment effects on the GPU using the k-buffer. In *Proc. 2007 Symposium on Interactive 3D Graphics and Games* (2007), pp. 97–104. 1

[BM08] BAVOIL L., MEYERS K.: *Order independent transparency with dual depth peeling*. Tech. rep., NVIDIA Corporation, Feb. 2008. 1

[BWWM10] BOWERS J., WANG R., WEI L.-Y., MALETZ D.: Parallel poisson disk sampling with spectrum analysis on surfaces. *ACM Transactions on Graphics (TOG) (Proceedings of ACM SIGGRAPH Asia 2010) 29*, 6 (2010). 1, 2

[Car84] CARPENTER L.: The A-buffer, an antialiased hidden surface method. In *Proc. SIGGRAPH '84* (1984), pp. 103–108. 1

[CR74] CATMULL E., ROM R.: A class of local interpolating splines. In *Computer Aided Geometric Design* (1974), Barnhill R. E., Reisenfeld R. F., (Eds.), Academic Press, pp. 317–326. 3, 4

[dFG96] DE FIGUEIREDO L. H., GOMES J.: Sampling implicit objects with physically-based particle systems. *Computers & Graphics 20*, 3 (1996), 365–375. 1

[ESSL11] ENDERTON E., SINTORN E., SHIRLEY P., LUEBKE D.: Stochastic transparency. *IEEE TVCG 17*, 8 (2011), 1036–1047. 1

[Eve01] EVERITT C.: *Interactive order-independent transparency*. Tech. rep., NVIDIA Corporation, 2001. 1

[FGH*85] FUCHS H., GOLDFEATHER J., HULTQUIST J. P., SPACH S., AUSTIN J. D., BROOKS JR. F. P., EYLES J. G., POULTON J.: Fast spheres, shadows, textures, transparencies, and image enhancements in pixel-planes. In *Proc. SIGGRAPH '85* (1985), pp. 111–120. 1

[GPSS07] GÜTHER J., POPOV S., SEIDEL H.-P., SLUSALLEK P.: Realtime ray tracing on GPU with BVH-based packet traversal. In *Proc. IEEE/Eurographics Symposium on Interactive Ray Tracing 2007* (2007), pp. 113–118. 1

[KHK*09] KNOLL A., HIJAZI Y., KENSLER A., SCHOTT M., HANSEN C., HAGEN H.: Fast ray tracing of arbitrary implicit surfaces with interval and affine arithmetic. *Computer Graphics Forum 28*, 1 (2009), 26–40. 1

[KST08] KOYAMADA K., SAKAMOTO N., TANAKA S.: A particle modeling for rendering irregular volumes. In *Proc. UKSIM 2008* (2008), pp. 372–377. 1, 4

[LB06] LOOP C., BLINN J.: Real-time GPU rendering of piecewise algebraic surfaces. In *Proc. SIGGRAPH 2006* (2006), pp. 664–670. 1

[LWX06] LIU B., WEI L. Y., XU Y. Q.: *Multi-layer depth peeling via fragment sort*. Tech. rep., Microsoft Research 2006-81, 2006. 1

[MGvW98] MULDER J. D., GROEN F. C. A., VAN WIJK J. J.: Pixel masks for screen-door transparency. In *Proc. IEEE Visualization '98* (1998), pp. 351–358. 1

[MGW05] MEYER M. D., GEORGE P., WHITAKER R. T.: Robust particle systems for curvature dependent sampling of implicit surfaces. In *Proc. the International Conference on Shape Modeling and Applications 2005 (SMI'05)* (2005), pp. 124–133. 1, 2

[ND93] NEIDER J., DAVIS T.: *OpenGL Programming Guide, Release 1*. Addison-Wesley, 1993. 1

[OAG10] ÖZTIRELI A. C., ALEXA M., GROSS M.: Spectral sampling of manifolds. *ACM Transactions on Graphics (TOG) (Proceedings of ACM SIGGRAPH Asia 2010) 29*, 6 (2010). 1, 2

[OBA*03] OHTAKE Y., BELYAEV A., ALEXA M., TURK G., SEIDEL H. P.: Multi-level partition of unity implicits. In *Proc. SIGGRAPH 2003* (2003), pp. 463–470. 1, 3

[OGG09] ÖZTIRELI A. C., G. GUENNEBAUD M. G.: Feature preserving point set surfaces based on non-linear kernel regression. *Computer Graphics Forum 28*, 2 (2009), 493–501. 1, 2

[PBMH02] PURCELL T. J., BUCK I., MARK W. R., HANRAHAN P.: Ray tracing on programmable graphics hardware. In *Proc. SIGGRAPH 2002* (2002), pp. 703–712. 1

[SCG03] SEN O., CHEMUDUGUNTA C., GOPI M.: Silhouette-opaque transparency rendering. In *Proc. Computer Graphics and Imaging 2003* (2003), pp. 153–158. 1

[SKKN10] SAKAMOTO N., KAWAMURA T., KOYAMADA K., NOZAKI K.: Improvement of particle-based volume rendering for visualizing irregular volume data sets. *Computers & Graphics 34* (2010), 34–42. 1

[TMN*00] TANAKA S., MORISAKI A., NAKATA S., FUKUDA Y., YAMAMOTO H.: Sampling implicit surfaces based on stochastic differential equations with converging constraint. *Computers & Graphics 24*, 3 (2000), 419–431. 2

[TO02] TURK G., O'BRIEN J. F.: Modelling with implicit surfaces that interpolate. *ACM Transactions on Graphics (TOG) 21*, 4 (2002), 855–873. 1

[TSYK01] TANAKA S., SHIBATA A., YAMAMOTO H., KOTSURU H.: Generalized stochastic sampling method for visualization and investigation of implicit surfaces. *Computer Graphics Forum 20*, 3 (2001), 359–367. (Proc. Eurographics 2001). 1, 2, 3

[WH94] WITKIN A., HECKBERT P.: Using particles to sample and control implicit surfaces. In *Proc. SIGGRAPH '94* (1994), pp. 269–277. 1, 2

[YHGT10] YANG J. C., HENSLEY J., GRÜN H., THIBIEROZ N.: Real-time concurrent linked list construction on the GPU. *Computer Graphics Forum 29*, 4 (2010), 1297–1304. 1

[ZPvBG02] ZWICKER M., PFISTER H., VAN BAAR J., GROSS M.: EWA splatting. *IEEE TVCG 8*, 3 (2002), 223–238. 2