High-quality Particle-based Volume Rendering for Large-scale Unstructured Volume Datasets

Naohisa Sakamoto, Naoya Maeda, Takuma Kawamura, Koji Koyamada[1]

Abstract In this article, we propose a technique for improving the image quality of particle-based volume rendering (PBVR). A large-scale unstructured volume dataset often contains multiple sub-volumes, which cannot be ordered by visibility. PBVR can handle this type of volume dataset. Sampling misses often occur when the transfer function undergoes drastic changes, which can result in poor image quality. To reduce sampling misses caused by the high-frequency transfer function, we develop a new sampling technique called "layered sampling". To confirm the effectiveness of our technique, we apply the proposed technique to a large-scale unstructured volume dataset subdivided into multiple sub-volumes.

Keywords large-scale unstructured volume datasets, layered sampling, volume rendering

1 Introduction

Developing a sorting-free technique for rendering unstructured volume datasets has been a major challenge for the visualization community. Such datasets consist mainly of scalar data, defined as collections of irregularly ordered cells with shapes that are not necessarily orthogonally cubic. Roettger et al. (2003) noted that the memory bandwidth required for visibility sorting becomes the limiting factor, and they proposed an algorithm that requires no visibility sorting for cells with unstructured volumes. However, because their optical model considers only emissions, its application is limited to visualizing gaseous phenomena. Csebfalvi (2004) proposed a sorting-free volume rendering technique that can be categorized as an X-ray volume rendering approach, though their optical model considers only absorption. Zhou et al. (2006) proposed a sorting-free rendering technique that is implemented with additional terms to help provide enhanced depth cues without visibility sorting. Though their technique has achieved 20 fps for 17.6 million tetrahedra, their optical model does not consider absorption effects.

To solve the above problems of sorting-free volume rendering techniques, Sakamoto et al. (2010b) returned to the density emitter model, and they presented a basic idea for this approach. The proposed particle-based volume rendering (PBVR) technique represents the 3-dimensional scalar field as a set of particles, and it considers both emission and absorption effects (Sakamoto et al. 2010a). The particle density is derived from a user-specified transfer function and is utilized to estimate the number of particles to be generated in a given volume dataset. Because the particles can be considered fully opaque, no visibility sorting processing is required during the rendering process. This is advantageous from a distributed processing perspective where we often face a large-scale unstructured volume that is subdivided into multiple sub-volumes. When we visualize such a large volume, a common approach is to generate sub-images from the sub-volumes and compose sub-images in a visibility order into a final image. For the regular volume, the visibility order becomes apparent. For an unstructured volume, however, the visibility order is difficult to calculate because the shape of the sub-volume can be non-convex. Thus, a sorting-free volume rendering is indispensable for processing multiple sub-volumes.

N.Sakamoto, K.Koyamada
Institute for the Promotion of Excellence in Higher Education, Kyoto University
Yoshida Nihonmatsu-cho, Sakyo-ku, Kyoto-shi, Kyoto 606-8501, Japan
E-mail: naohisas@viz.media.kyoto-u.ac.jp
Tel: +81-75-753-9372

N.Maeda
Graduate School of Engineering, Kyoto University
Kyoto daigaku tasura, Nishikyo-ku, Kyoto-shi, Kyoto 615-8530, Japan

T.Kawamura
Center for Computational Science and e-Systems, Japan Atomic Energy Agency
5-1-5 Kashiwanoha, Kashiwa-shi, Chiba 277-8587, Japan

The first proposed PBVR limits the locations where a set of particles is generated from an unstructured volume dataset to the regular grids (Sakamoto et al. 2010a). This limitation results in poor image quality. The second version introduces a more precise particle model to improve the image quality (Sakamoto et al. 2010b). The current weakness of PBVR occurs while generating a low-quality image due to an under-estimated number of particles when the distribution of the transfer function has a peak in which the opacity value rises steeply in a narrow interval. Such transfer function is often used to represent a given volume dataset as a set of semi-transparent isosurfaces. The under-estimation is caused because the number of particles is evaluated by using only the particle density at the center of the tetrahedral cell. To solve this problem, we develop a layered sampling technique in which a volume cell is subdivided into a number of regions, which are related to the intervals. This technique generates particles by evaluating the number of particles at each interval of a volume cell.

2 Related works

High-performance parallel computers have recently created huge unstructured volumes subdivided into multiple sub-volumes and preserved in each machine in a distributed computing environment. Though several Parallel Volume Rendering (PVR) techniques for unstructured volumes exist (Silva 1996; Chen et al. 2002), achieving an interactive frame rate remains difficult.

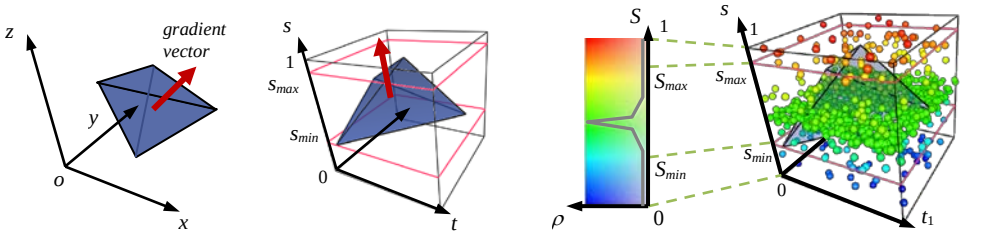
PVR for unstructured volumes requires traversal cell order, as the unstructured sub-volumes are often non-convex, and the rendering thus cannot employ image composition. In a single volume, several effective techniques are available. Meredith and Ma (2001) developed hardware-assisted splats that manage tetrahedral cells using octrees. Muigg et al. (2007) proposed a hybrid ray-casting technique for large unstructured volumes. Though this method can render various unstructured meshes with region of interest (ROI) control by converting the non-interesting region to a regular mesh, it cannot handle meshes containing deformed elements. Callahan et al. (2005) developed an integrated visibility ordering technique, called Hardware Assisted Visibility Sorting (HAVS), in which the centroids of the cell faces are first sorted to create a rough visibility ordering. The pixel fragments generated from rasterized faces then increase the accuracy with a k-buffer. However, integrating these methods into PVR is difficult because the communication between machines during cell tracing creates a bottleneck.

The previous PBVR comprised three processes: particle generation, particle projection and sub-pixel processing (Sakamoto et al. 2010b). In this technique, the required frame buffer size becomes large depending on the sub-pixel level when performing sub-pixel processing. To solve this problem, the ensemble average process was developed (Sakamoto et al. 2010; Kawamura et al. 2010). Repetition processing required for the ensemble averaging can reduce the memory cost but requires more rendering time than sub-pixel processing. This technique was implemented using the GPU, which improves PBVR performance, and outperformed HAVS in rendering speed and scalability (Ding et al. 2010).

Sakamoto et al. developed a distributed PBVR implementation to visualize a large-scale unstructured volume dataset generated from a distributed finite element method (FEM) simulation (Sakamoto et al. 2010a). Because the computational mesh was too large for a single computational node to handle, it was divided into multiple regions. The computational result thus contained multiple unstructured volume datasets. Sakamoto et al. first applied the developed technique to the FEM simulation results; other volume rendering techniques had never visualized such a large-scale dataset. On the basis of the experimental results, they constructed a performance model as a function of the number of CPU cores. They found that the overall performance time improved as the number of CPU cores increased to 250.

3 Layered sampling

Metropolis sampling for PBVR (Kawamura et al. 2010) first calculates the particle density values at each cell vertex and then interpolates for arbitrary positions in the volume cell. The density values are smoothed even if they change drastically in the volume cell, which results in a low-quality image. In layered sampling, we assume that the transfer function is described by a piecewise linear function with respect to scalar values. The scalar values are evenly subdivided into pieces, i.e., intervals. Particles are generated at each interval volume.



(a) A tetrahedral cell in object coordinates

(b) Affine transformation to the tetrahedral cell

(c) The piecewise linear density function $\rho(S)$

(d) Pre-generated particles in the regular box, which includes the tetrahedral cell

Fig. 1 Layered sampling overview.

If we assume that we process a volume dataset composed of tetrahedral cells and employ a linear interpolation within the cell, we can develop an efficient sampling technique. With that assumption, the scalar gradient becomes constant in the tetrahedral cell, and the cell is segmented into layer of the scalar interval. If we examine the cell along the gradient vector, the distribution pattern is identical: the particle distribution becomes constant at each layer. This observation provides a clue to creating an efficient sampling technique. This technique first prepares a cubic box, hereafter called a regular box, sufficiently large to include the largest tetrahedral cell. The technique then considered one of the axes to be the scalar data axis, and it calculates particle density by referring to the transfer function. Finally, the technique generates particles in layers vertical to the scalar axis with intersections identical to the endpoints of the scalar intervals (Figure 1).

After pre-generating particles in the regular box, we can improve the performance of the generated particles in each tetrahedral cell. When we generate particles in a tetrahedral cell, we apply an affine transformation to the cell so that the scalar gradient vector becomes parallel to the scalar axis, and the cell is included in the regular box. After the transformation, we can specify the particles to be generated inside the tetrahedral cell instead of actually generating them. Thus, we expect that image-quality degradation can be suppressed because the density distribution can reflect the transfer function even if its change becomes drastic.

This method employs three stages. The first stage is pre-generating the particles using the density function calculated from the opacity function, and the second stage estimating the number of particles according to the integral value of the density function. The last stage is generating particles in each tetrahedral cell using the estimated number of particles from the pre-generated particles.

3.1 Particle pre-generation

A sampling process in each tetrahedral cell determines pre-generated particles. Before the sampling process, particles are generated in the regular box using the piecewise linear transfer function (Figures 1 (c), (d)). In this regular box, the pre-generated particles are sampled at each interval volume in a tetrahedral cell. Sampling the linear density field at each interval volume for particle generation is easy. We employ the rejection method (Neumann 1951) for this sampling. Before beginning the sampling, we assume the total number (N) of pre-generated particles, which is a user-specified number. To estimate the number of particles at each interval volume, the density field in the regular box must be integrated. The integration value determines the division ratio of the total number of pre-generated particles.

Within the regular box, the coordinate axes are described as (t_1, t_2, s). The value of the s axis is identical to the scalar value S ($S = s$), and the coordinate axes are normalized into the [0, 1] region. The pre-generated particles are distributed as if they were slabs in which the particles are uniformly distributed in the t_1, t_2 axes while stacked in the s axis. The coordinate (t_1, t_2, s) is limited to the region $[0, 1]^3$.

We assume a piecewise linear function of $\rho(S)$ by calculating a density function value $\alpha(S)$. The density function is calculated by substituting both ends of the opacity interval for the density estimation. In an interval $[S_i, S_{i+1}]$, the density is represented as:

$$\rho(S) = \frac{\rho_{i+1} - \rho_i}{S_{i+1} - S_i} S + \frac{\rho_i S_{i+1} - \rho_{i+1} S_i}{S_{i+1} - S_i}, \quad (1)$$

where S_i and S_{i+1} are both ends of the scalar interval. ρ_i and ρ_{i+1} are the corresponding density values.

The following integral calculates the division ratio m_i of the total pre-generated particles at the interval $[S_i, S_{i+1}]$:

$$m_i = \int_{S_i}^{S_{i+1}} \rho(S)\, ds = \frac{1}{2}(\rho_i + \rho_{i+1})(S_{i+1} - S_i) \quad . \tag{2}$$

The ratio becomes m_i/M, where M is total value of m_i, and the number of particles in an interval is calculated as $N \times m_i/M$.

3.2 Particle number estimation

The particle number estimation stage is a sequence of processes. First, we calculate an affine transformation of the cell to the regular box (*Transformation*). The cell is inserted into the regular box, and the inclusion of the pre-generated particles inside the tetrahedral cell is determined (*Insertion*). The included particles are then utilized to calculate the number of generated particles N_{tet} by the volume integration (*Integration*).

Each tetrahedral cell is fitted into the regular box via an affine transform to calculate volume integration and generate particles. The affine transform contains matrices **L** and **A**. Matrix **L** rotates the gradient vector of a cell, so it becomes parallel to the *z* direction of the object space where particles are pre-generated. Matrix **A** is a scaling and translation matrix that fits the rotated tetrahedral cell into the regular box, where the direction of the gradient vector is identical to the *s* axis. Here, *Tet* indicates the original tetrahedral cell. *Tet'* indicates the tetrahedral cell transformed by matrix **L**, and *Tet''* indicates a tetrahedral cell in the regular box. Then, *Tet''* = **A** *Tet'* = **A L** *Tet*.

Matrix **L** is constructed from the normalized gradient vector g and vector u, which is an arbitrary unit vector that is linearly independent of g. To determine u, the absolute value of each component in g is compared with the other components, and the component with the smallest absolute value becomes 1. The other components become zero. The orthogonal basis vectors are calculated as

$$l_1 = u \times g, \quad l_2 = g \times (u \times g). \tag{3}$$

Then matrix **L** is defined as follows:

$$\mathbf{L} = \begin{bmatrix} l_1 & l_2 & g & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^T \quad . \tag{4}$$

Because the gradient vector is constant in a tetrahedral cell, the *z* coordinate of the new coordinate system transformed by matrix **L** is identical to the scalar data value. Thus, matrix **A**, which transforms the *z* coordinate into the scalar data value, is defined as follows:

$$\mathbf{A} = \begin{bmatrix} a_1 & 0 & 0 & b_1 \\ 0 & a_2 & 0 & b_2 \\ 0 & 0 & a_3 & b_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

$$a_1 = \frac{1}{x'_{\max} - x'_{\min}} \quad b_1 = -a_1 x'_{\min}$$

$$a_2 = \frac{1}{y'_{\max} - y'_{\min}} \quad b_2 = -a_2 y'_{\min}$$

$$a_3 = \frac{S_{\max} - S_{\min}}{z'_{\max} - z'_{\min}} \quad b_3 = -a_3 z'_{\min} + S_{\min}$$

Here, $S_{\min}$ and $S_{\max}$ are the minimum and maximum scalar values, respectively, in a tetrahedral cell. (x', y', z') denotes the coordinates of *Tet'*. The minimum and maximum coordinates of a bounding

box of *Tet'* are x'_{min}, x'_{max}. y'_{min}, y'_{max}, z'_{min}, and z'_{max}, respectively. The scaling factors of matrix **A** are determined, so each length of the bounding box of the tetrahedral cell is normalized to [0,1].

The volume integration of the density function estimates number of particles N_{tet} in each cell. In layered sampling, N_{tet} is numerically obtained by counting the pre-generated particles in a tetrahedral cell:

$$N_{tet} = M \frac{1}{|a_1 a_2 a_3|} \frac{N_{in}}{N} \tag{6}$$

where N_{in} is the number of pre-generated particles included in the tetrahedral cell and M is the integration value of the density function $\rho(S)$.

In our implementation, N_{in} particles included within *Tet''* are first selected in the pre-generated particles. Then, Equation 6 estimates N_{tet}. When the scalar value range represented by $S_{max}-S_{min}$ is smaller than δ, N_{in} becomes too small to serve as a reference, as *Tet''* becomes very thin in the regular box. δ is a pre-defined small number between [0, 1] and is determined to be smaller than a width of the sharp peak in the transfer function. In this case, we can therefore estimate the number of particles by single point integration.

3.3 Particle generation

In the particle generation stage, when the scalar range is greater than δ, N_{tet} particles are randomly selected from the N_{in} particles included in *Tet''* if N_{tet} does not exceed N_{in} (*Selection*). The selected particles are inversely mapped to the object coordinate by multiplying $\mathbf{L}^{-1}\mathbf{A}^{-1}$ (*Inverse Transformation*). If not, first the N_{in} particles are inversely mapped to the object coordinate, and additional particles must be generated by using rejection sampling to compensate for the shortage (*Rejection Sampling*). We cannot generate particles using the abovementioned sampling method when the scalar value range in a tetrahedral cell is smaller than δ. In this case, we generate particles using uniform sampling, because we can assume that the particles are uniformly distributed in the cell. The implementation of the particle generation process is as follows:

if $S_{max} - S_{min} > \delta$
  Calculate the transformation matrices **A** and **L** (*Transformation*)
  Determine the inclusion of the pre-generated particles in *Tet''* and obtain N_{in} (*Insertion*)
  Estimate N_{tet} by Equation 6 (*Integration*)

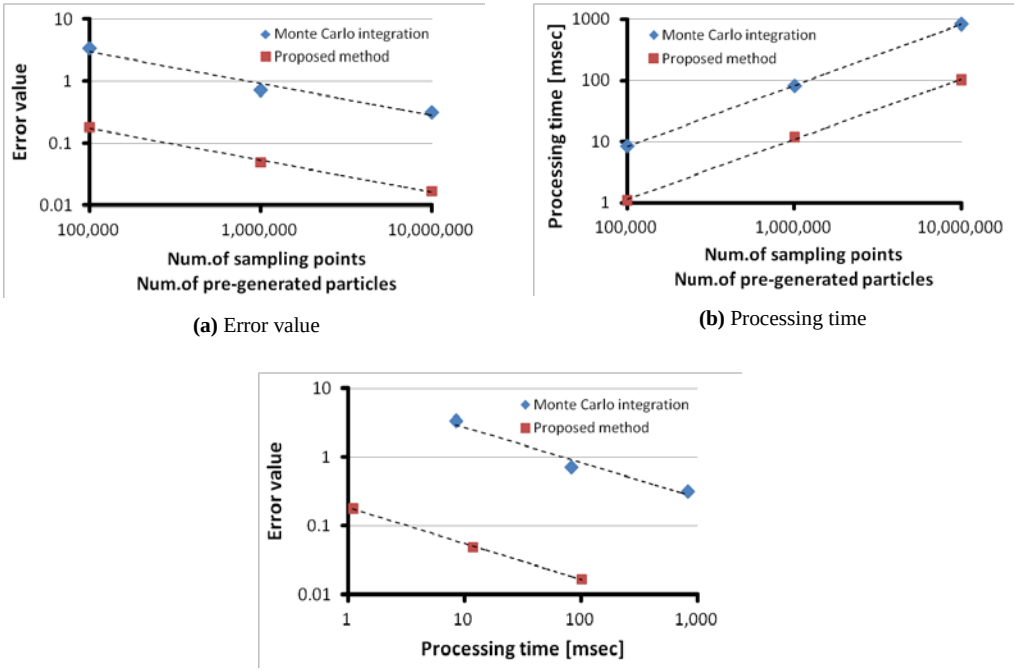
  if $N_{tet} <= N_{in}$
    Select N_{tet} particles included in *Tet''* (*Selection*)
    Transform the selected particles into *Tet* by multiplying $\mathbf{L}^{-1}\mathbf{A}^{-1}$ (*Inverse Transformation*)
  else
    Transform the selected particles into *Tet* by multiplying $\mathbf{L}^{-1}\mathbf{A}^{-1}$ (*Inverse Transformation*)
    Generate $N_{tet} - N_{in}$ particles by rejection sampling (*Rejection Sampling*)

else
  Estimate N_{tet} by single point integration
  Generate N_{tet} particles by rejection sampling

4 Experimental results and discussion

4.1 Layered sampling evaluation

We conducted two experiments to evaluate the effectiveness of the layered sampling technique. One experiment is the performance and computational accuracy evaluation of our method by comparing it with Monte Carlo integration. The other experiment is the image quality evaluation when applying our proposed sampling method and the Metropolis sampling method (Kawamura et



(a) Error value

(b) Processing time

(c) Error value and processing time

Fig. 2 Experimental results of computational accuracy and performance.

al. 2010). We used a PC with an Intel Core2 Duo 2.4 GHz CPU and 2.9 GB of RAM for these experimental evaluations.

4.1.1 Performance and computational accuracy

To calculate the analytical value, we used an unstructured volume data composed of a single tetrahedral cell. Figure 2 (a) shows the integration value errors calculated using our proposed method and Monte Carlo integration when changing the number of pre-generated particles and sampling points, respectively. From this result, we can verify that the accuracy of the integration value proportionally increases as the number of particles increases in the double logarithmic plot graph, and the approximated curves of the error value can be expressed as follows:

$$E_l = 64.7 N_l^{-0.513}, \tag{7}$$

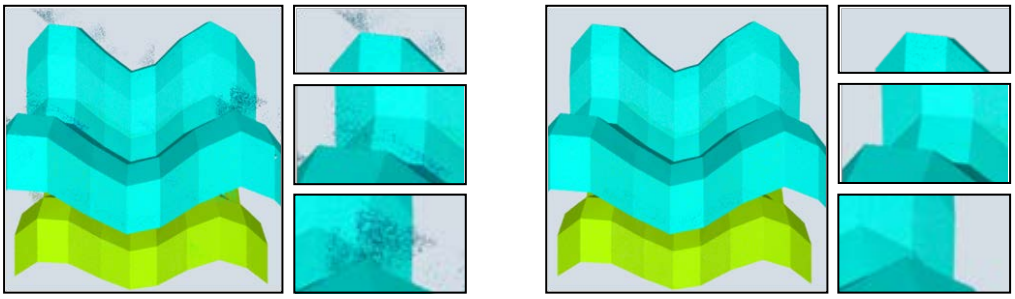
$$E_m = 1082.0 N_m^{-0.515}. \tag{8}$$

where E_l and E_m represent the error values calculated by using our proposed sampling method and Monte Carlo integration, respectively. N_l and N_m denote the number of pre-generated particles and sampling points. Equations 7 and 8 show that the error of both methods can be assumed to be $O(N^{-1/2})$, and the accuracy of our method can be confirmed to be about 17 times higher than Monte Carlo integration.

We also measured the averaged computational times of the integration value. Figure 2 (b) shows the relationship between the computational time and integration value accuracy in the double logarithmic plot graph. This figure confirms that the computation time of our proposed method outperforms Monte Carlo integration by nearly ten times. We can also verify that our proposed method estimates the number of particles for the particle generation stage with higher accuracy and speed than the Monte Carlo integration, as shown in Figure 2 (c).

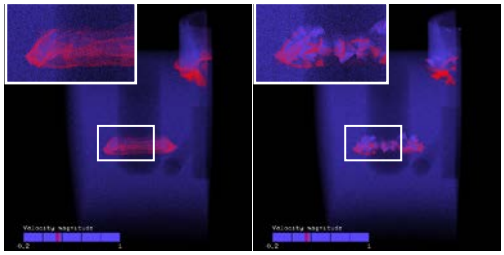
4.1.2 Image quality

As test data for evaluating image quality, we used a simple regular volume dataset (10x10x10). In this experiment, we generated particles for test data using our proposed method and the metropolis method according to the number of particles, which Equation 6 can estimate. We used a transfer function with two opacity value peaks for particle generation. The particle generation times of our method and the metropolis method were 153.0 and 127.7 seconds, respectively. Figure 3 shows the



(a) Metropolis method **(b)** Layered sampling method

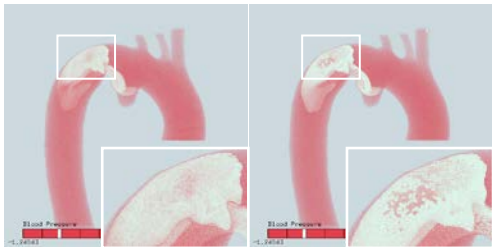
Fig. 3 Rendering results of the test data by using the metropolis method and the layered sampling method.



(a) Layered sampling **(b)** Previous technique

(c) Transfer Function.

Fig. 4 Rendering results of SPX and employed transfer function.



(a) Layered sampling **(b)** Previous technique

(c) Transfer Function.

Fig. 5 Rendering results of Aorta and employed transfer function.

test data rendering results. Though our method is about 20 % more costly than the metropolis method, Figure 3 (b) eliminates the supposedly impermissible particles that appear in Figure 3 (a).

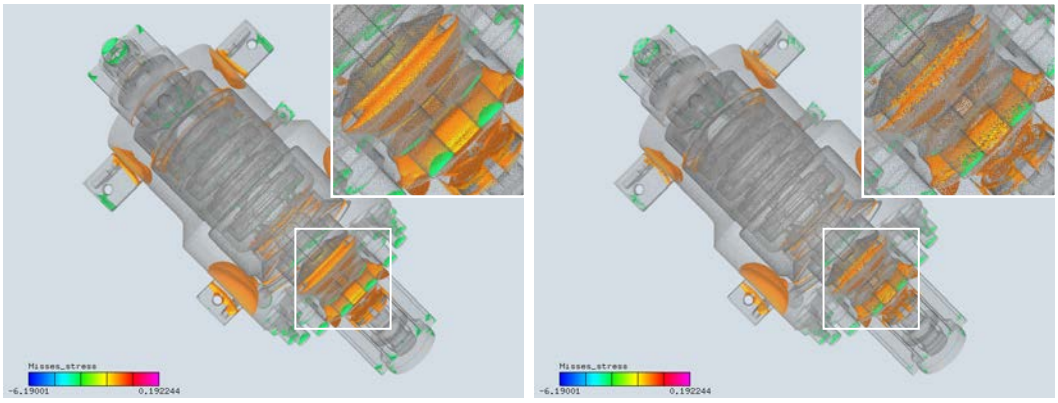
4.1.3 Comparison with previous PBVR

To evaluate the effectiveness of layered sampling, we applied it to the tetrahedral volumes "SPX" (2,896 vertices, 12,936 cells) and "Aorta" (248,992 vertices, 1,386,882 cells). We used the transfer functions for SPX and Aorta, as shown in Figures 4 and 5 (c), to visualize the isosurfaces because they have a drastic change in opacity value as a single-peak function. We compared layered sampling to previous PBVR sampling techniques in both performance and image quality.

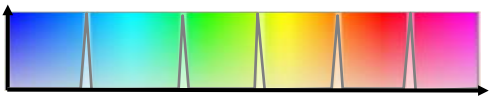
To evaluate the image quality, we compared the rendering results using the two techniques, as shown in Figures 4 and 5. In Figure 4 (b), which was generated by the previous technique, we can observe a discontinuous pattern caused by insufficient sampling around the red-colored portion. In Figure 4 (a), which was generated by layered sampling, we can confirm that the red-colored surface was visualized. In Figure 5 (b), we can see the perforated white surface. In Figure 5 (a), we can see an improved image quality. Table 1 shows statistics on the performance data. We confirm that the sampling time of the previous technique outperformed layered sampling. For both layered sampling and the previous technique (where the latter outperforms the former), the sampling time increases in proportion to the number of cells.

Table 1 Performance comparison. The repeat level is 144, and the number of pre-generated particles is 100,000. 'N/A' means there is no corresponding process. T1 means layered sampling and T2 means the previous technique.

	SPX		**Aorta**	
	T1	**T2**	**T1**	**T2**
Pre-gen. time [msec]	703	N/A	750	N/A
Sampling time [sec]	5.6	1.1	290.7	4.7
Num. particles [M]	5.16	5.12	17.74	17.76
Frame-rate [fps]	18.6	18.8	7.6	7.6



(a) Layered sampling **(b)** Previous technique



(c) Transfer Function

Fig. 6 Rendering results of Pump and employed transfer function. The gray colored semi-transparent surface is the boundary surface of Pump.

5.2 Application to large-scale volume dataset

The unstructured volume dataset “Pump” from a computational structural mechanics (CSM) simulation is the result of an elastostatic weight analysis with one hundred million degrees of freedom using a PC cluster. The Pump dataset contained 26,289,770 quadratic tetrahedral cells with 36,728,129 nodes, and it was divided into 32 datasets as outputs generated from the distributed CSM computation with a finite element method solver.

We applied layered sampling to the Pump dataset. Because the sampling technique assumes a volume dataset composed of linear tetrahedral cells, we subdivided the quadratic tetrahedral cell into eight linear cells. Thus, PBVR with layered sampling rendered 210,318,160 tetrahedral cell volume datasets. For this experiment, we used a PC with an Intel Core2 Quad CPU running at 2.83 GHz with 8.0 GB of RAM and an NVIDIA GeForce GPX280 GPU with 1.5 GB VRAM. We successfully tuned the layered sampling process with four threads. We handled the divided volume data by making each thread process every four cells. Generating the particles took 1,162.9 seconds. Figure 6 shows the rendering results with and without the technique. In the rendering, we employed a transfer function in which the opacity values change drastically at some scalar values. A transfer function, as shown in Figure 6 (c), was referenced. Figure 6 shows the boundary surface rendering of the Pump data, which was independently processed in layered sampling. The boundary surface of Pump was converted to particles by uniform sampling on the polygon patch. A total of 7.0 mega particles were generated on the boundary surface, and 9.8 mega particles were generated with layered sampling. Table 2 shows the sampling time, generated particles, and rendering performance. From Table 2, we can confirm that the number of particles generated by layered sampling is larger than that generated by the previous technique. The previous technique could not calculate the volume integration of the density field accurately because of sampling misses for the isosurface region. We may conclude that the layered sampling technique can reveal isosurfaces clearly.

Table 2 Performance comparison. The shared repeat level is 144, and the number of pre-generated particles is 20,000.

	Layered sampling	Previous technique
Pre-gen. time [msec]	3.9	
Num. particles [M]	9.8	5.2
Sampling time [sec]	1163.9	27.7
Frame-rate [fps]	7.9	10.2

5 Conclusion

We have proposed a high-quality PBVR technique that includes layered sampling. The previous PBVR created low quality images when handling high-frequency transfer functions. Layered sampling is applied to a tetrahedral mesh and can handle high-frequency transfer functions. To confirm the scalability, we applied the proposed technique to the CSM simulation results of a size never previously visualized using a volume rendering technique.

In layered sampling, particles are generated in the regular box space using the density function. The volume integration and particle generation are performed using pre-generated particles. The effectiveness of this technique was confirmed by comparing the pre-integrated volume rendering in the quality of images generated by a transfer function in which the opacity values change drastically. PBVR with layered sampling clearly rendered the isosurface-like interval volume, which is difficult to render with any previous sampling technique. We confirmed that the previous sampling technique is sufficient when the opacity function changes gradually. To improve the sampling time, we plan to develop a hybrid technique that combines layered sampling and the previous sampling with opacity function variation detection. The sampling time was strongly influenced by the number of pre-generated particles. However, determining the optimal number is difficult, and solving this problem is one for future work. We must reduce the pre-generation time to apply the technique to larger unstructured volumes.

The current implementation supports a tetrahedral volume dataset. When we process other types of volume datasets, subdividing the volume cell into multiple tetrahedral cells to optimize our proposed technique is possible. In the future, we will develop a technique for pre-generating particles for other cells.

Reference

Callahan SP, Ikits M, Comba JLD, Silva CT (2005) Hardware-Assisted Visibility Sorting for Unstructured Volume Rendering, IEEE Trans. on Visualization and Computer Graphics, 11(3):285-295

Chen L, Fujishiro I, Nakajima K (2002) Parallel Performance Optimization of Large-scale Unstructured Data Visualization for the Earth Simulator, In Proc. of the Fourth Eurographics Workshop on Parallel Graphics and Visualization, pp. 133-140

Csebfalvi B (2004) Interactive Transfer Function Control for Monte Carlo Volume Rendering, In Proc. of IEEE Symposium on Volume Visualization and Graphics 2004, pp. 33-38

Ding Z, Kawamura T, Sakamoto N, Koyamada K (2010) Particle-based Multiple Irregular Volume Rendering on CUDA, Simulation Modelling Practice and Theory, 18(8):1172-1183

Kawamura T, Sakamoto N, Koyamada K (2010) A Level-of-Detail Rendering of a Large-Scale Irregular Volume Dataset Using Particles, Journal of Computer Science and Technology, 25(5):905-915

Meredith J, Ma KL (2001) Multiresolution View-Dependent Splat-based Volume Rendering of Large Irregular Data. In Proc. of IEEE Symposium on Parallel and Large-Data Visualization and Graphics, pp. 93-155

Muigg P, Hadwiger M, Doleisch H, Hauser H (2007) Scalable Hybrid Unstructured and Structured Grid Raycasting, IEEE Trans. on Visualization and Computer Graphics, 13(6): 1592-1599

Neumann J (1951) Various Technique used in Connection with Random digits, Journal of Research of the National Bureau of Standards, Applied Mathematics Series, 12:36-38

Roettger S, Ertl T (2003) Cell Projection of Convex Polyhedra, In Proc. of Volume Graphics 2003, pp. 103-107

Sakamoto N, Kawamura T, Koyamada K (2010) Improvement of Particle-based Volume Rendering for Visualizing Irregular Volume Data Sets, Computers & Graphics, 34(1):34-42

Sakamoto N, Kuwano H, Kawamura T, Koyamada K, Nozaki K (2010) Visualization of Large-scale CFD Simulation Results Using Distributed Particle-Based Volume Rendering, International Journal of Emerging Multidisciplinary Fluid Sciences, 2(2):73-86

Silva CT, (1996) Parallel Volume Rendering of Irregular Grids. Ph.D. thesis, State University of New York at Stony Brook

Zhou Y, Garland M (2006) Interactive Point-Based Rendering of Higher-Order Tetrahedral Data, IEEE Trans. on Visualization and Computer Graphics, 12(5):1229-1236