Computational Visual Media
https://doi.org/10.1007/s41095-019-0137-0 Vol. 5, No. 2, June 2019, 151–160
Research Article

# A method for estimating the errors in many-light rendering with supersampling

**Hirokazu Sakai**[1], **Kosuke Nabata**[1], **Shinya Yasuaki**[1], **and Kei Iwasaki**[1,2] (✉)



**Abstract** In many-light rendering, a variety of visual and illumination effects, including anti-aliasing, depth of field, volumetric scattering, and subsurface scattering, are combined to create a number of virtual point lights (VPLs). This is done in order to simplify computation of the resulting illumination. Naive approaches that sum the direct illumination from many VPLs are computationally expensive; scalable methods can be computed more efficiently by clustering VPLs, and then estimating their sum by sampling a small number of VPLs. Although significant speed-up has been achieved using scalable methods, clustering leads to uncontrollable errors, resulting in noise in the rendered images. In this paper, we propose a method to improve the estimation accuracy of many-light rendering involving such visual and illumination effects. We demonstrate that our method can improve the estimation accuracy by a factor of 2.3 over the previous method.

**Keywords** anti-aliasing; depth of field; many-light rendering; participating media

## 1 Introduction

Many-light rendering methods simplify the complex computation of global illumination into a simple summation of the contributions from many virtual point lights (VPLs) [1]. In many-light rendering, the incident radiance at a point to be shaded (the *shading point*) is calculated using VPLs. Because the number of VPLs used is generally quite large, previous methods [2–4] have clustered VPLs to streamline the computation. These methods, however, cannot control the errors produced by clustering, resulting in noise in the rendered images. To eliminate noise in such systems, users must adjust clustering parameters in tedious trial-and-error processes.

To address this problem, Nabata et al. [5] proposed a method to estimate errors in the pixel values due to VPL clustering. This method, however, estimates each pixel value using a single shading point and cannot be applied to visual effects such as anti-aliasing and depth-of-field (DOF), which require multiple shading points to estimate the pixel value. Walter et al. [6] proposed a method called *multidimensional lightcuts* (*MDLC*), which attempts to control the clustering errors in many-light rendering of various visual effects. This method, however, does not estimate the error in each pixel value, which appears as noise in the rendered images. This paper proposes a method to improve the accuracy of pixel values for visual effects such as anti-aliasing, DOF, and volumetric effects, as shown in Fig. 1. Our method automatically partitions clusters and continues sampling them until the relative errors in the pixel values are smaller than a user-specified threshold.

## 2 Related work

### 2.1 Scalable VPL rendering

Recent advances in many-light rendering have demonstrated that global illumination effects can be more than adequately approximated using many virtual lights [1, 7]. Keller [8] introduced the instant radiosity method, which calculates the indirect illumination from virtual point lights (VPLs). Walter et al. [2, 6] proposed a scalable solution to many-light

1 Wakayama University, Wakayama, Wakayama, 640-8510, Japan.
2 Dwango CG Research, KADOKAWA Hongo Bldg. 5245 Hongo, Bunkyo-ku, Tokyo, 1130033, Japan. E-mail: iwasaki@sys.wakayama-u.ac.jp (✉).
Manuscript received: 2018-12-21; accepted: 2019-02-15


**Fig. 1** Rendering with anti-aliasing (256 spp): (a) reference, (b) our method, (c) multidimensional lightcuts (MDLC). (b) and (c) are rendered so that the relative errors are less than 2%. (d) and (e) indicate relative errors (error color-code shown at far right). Values in (d) and (e) give the percentages of pixels with relative errors of less than 2%. Our method improves the estimation accuracy of pixel values by approximately 50% compared to MDLC. The low estimation accuracy of MDLC results in noise as shown in (c).

methods using a hierarchical representation of VPLs, called *cuts*. Hašan et al. [9] represented many-light rendering via a large matrix, and explored the matrix structure by using row–column sampling. Ou and Pellacini [3] clustered the shading points into groups called slices and performed matrix row–column sampling for each slice. Georgiev et al. [10] proposed an importance sampling method for VPLs by recording the contributions of the VPLs at each cache point. Yoshida et al. [11] proposed an adaptive cache insertion method to improve VPL sampling. Wu and Chuang [12] proposed the *VisibilityCluster* algorithm, which approximates the visibilities between each cluster of VPLs and those of the shading points by estimating the averaged visibility. Huo et al. [4] proposed a matrix sampling-and-recovery method to efficiently gather contributions from the VPLs by sampling a small number of them. Although these methods can significantly accelerate many-light rendering, the results rely on user-specified parameters, and finding the optimal parameter values remains a challenging task.

### 2.2 Participating media

VPL rendering suffers from splotches that stem from the singularity of the geometry term relating shading points and VPLs. Clamping is often used to avoid these artifacts. Engelhardt et al. [13] proposed a method to compensate for energy loss due to clamping and proposed a rendering method for participating media using VPLs. Novák et al. [14] proposed virtual ray lights (VRLs) to alleviate this singularity, and several groups have proposed acceleration methods [15, 16] that use VPLs and VRLs. These methods, however, do not estimate the errors due to clustering of the VPLs and VRLs.

### 2.3 Subsurface scattering

Arbree et al. [17] proposed a scalable rendering method for translucent materials using VPLs. Walter et al. [18] proposed bidirectional lightcuts that supports complex illumination effects, including subsurface scattering. Wu et al. [19] formulated a radiance computation method for subsurface scattering by sampling light-to-surface and surface-to-camera matrices. Although these methods can render translucent materials efficiently by clustering VPLs, they cannot control the errors due to clustering.

To address this problem, we provide an error estimation method that can handle a wide variety of illumination and visual effects. After specifying the relative error tolerance $\epsilon$ and the probability $\alpha$, our method stochastically estimates the relative errors with probability $\alpha$, i.e., using our method, the relative errors of a proportion $\alpha$ of the pixels in the rendered image is likely to be less than $\epsilon$.

## 3 Background

In many-light rendering, the outgoing radiance $L(\boldsymbol{x}, \boldsymbol{x}_\mathrm{v})$ at shading point $\boldsymbol{x}$ toward the viewpoint $\boldsymbol{x}_\mathrm{v}$ is calculated by

$$L(\boldsymbol{x}, \boldsymbol{x}_\mathrm{v}) = \sum_{\boldsymbol{y} \in \mathbb{L}} I(\boldsymbol{y}) f(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{x}_\mathrm{v}) V(\boldsymbol{y}, \boldsymbol{x}) G(\boldsymbol{y}, \boldsymbol{x}) \quad (1)$$

where $\mathbb{L}$ is the set of VPLs, $I(\boldsymbol{y})$ is the intensity of VPL $\boldsymbol{y}$, and $f(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{x}_\mathrm{v})$ is the material function that encompasses the bidirectional reflectance distribution function (BRDF) on the surface and the phase

function within the volume. $V(\boldsymbol{y},\boldsymbol{x})$ and $G(\boldsymbol{y},\boldsymbol{x})$ are the visibility and geometry terms [1, 7] relating $\boldsymbol{y}$ and $\boldsymbol{x}$, respectively. To estimate the pixel value for the rendering of the participating media, anti-aliasing, and DOF, the outgoing radiances from multiple shading points are required, as shown in Fig. 2. For example, when multiple shading points are generated via supersampling for anti-aliasing, the pixel values are calculated using the weighted sum of the outgoing radiances from the shading points. The pixel value $\mathcal{I}$ produced by supersampling is calculated as

$$\begin{aligned}\mathcal{I} &= \sum_{\boldsymbol{x}\in\mathbb{G}} W(\boldsymbol{x})L(\boldsymbol{x},\boldsymbol{x}_{\mathrm{v}}) \\ &= \sum_{\boldsymbol{x}\in\mathbb{G}}\sum_{\boldsymbol{y}\in\mathbb{L}} I(\boldsymbol{y})W(\boldsymbol{x})f(\boldsymbol{y},\boldsymbol{x},\boldsymbol{x}_{\mathrm{v}})V(\boldsymbol{y},\boldsymbol{x})G(\boldsymbol{y},\boldsymbol{x})\end{aligned} \tag{2}$$

where $\mathbb{G}$ is the set of shading points, and $W$ is the weighting function. The weighting function $W$ and material function $f$ depend on the visual effects and the rendered objects. Details of $W$ and $f$ are provided in Sections 4.4–4.6.

The computational cost of Eq. (2) is proportional to the product of the number of VPLs $|\mathbb{L}|$ and the number of shading points $|\mathbb{G}|$. Since, in general, many VPLs are used, the computational cost of Eq. (2) is prohibitive. MDLC is an efficient scalable many-light rendering method that clusters VPLs and shading points [6]. MDLC implicitly represents the hierarchy of clusters with a *product graph* that consists of pairs of clusters for VPLs and shading points. MDLC estimates the upper bound for clustering error, but does not estimate the error in the pixel value (i.e., the sum of the errors due to clustering). We improve the estimation accuracy of the pixel value by combining a method for estimating errors [5] with MDLC [6].

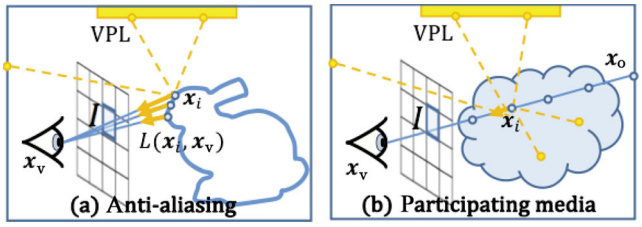



**Fig. 2** Pixel value estimation using multiple shading points.

## 4 Proposed method

### 4.1 Overview

Figure 3 shows an overview of our method, which estimates image pixel values by clustering VPLs and shading points. Clusters of VPLs and shading points are referred to as VPL clusters and shading clusters, respectively. VPL clusters and shading clusters are represented by binary trees, whose leaf nodes correspond to VPLs (or shading points), and whose inner nodes correspond to clusters of VPLs (or shading points). As shown in Fig. 3(b), we refer to the binary trees for the VPL clusters and the shading clusters as the *light tree* and the *shading tree*, respectively. Following MDLC, we denote a cluster pair consisting of a VPL cluster $\mathbb{C}^L$ and a shading cluster $\mathbb{C}^G$ by $(\mathbb{C}^L,\mathbb{C}^G)$.

We create the VPL clusters and build the light tree, used for all pixels, in a preprocess. For each pixel, we first generate shading points and build the shading tree by clustering the shading points. We prepare a priority queue $Q$ that stores the cluster pairs in descending order of standard deviation of each pair. $Q$ is initialized with the pair $(\mathbb{C}^L_{\mathrm{r}},\mathbb{C}^G_{\mathrm{r}})$ of root nodes of the light tree and the shading tree. By using this pair, the estimate $\hat{\mathcal{I}}$ of the pixel value and

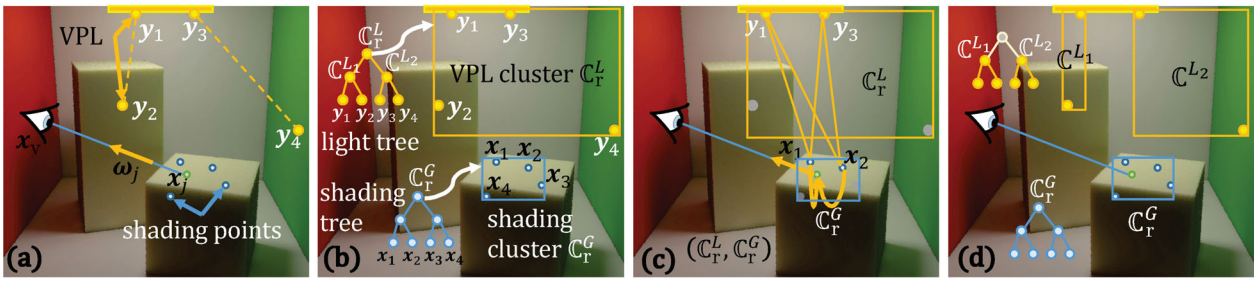



**Fig. 3** Overview of our method, explained for rendering translucent materials. (a) Subsurface scattering of light at $\boldsymbol{x}_j$ is calculated by generating shading points $\boldsymbol{x}_i$, which are importance sampled. (b) A VPL cluster $\mathbb{C}^L_{\mathrm{r}}$ and shading cluster $\mathbb{C}^G_{\mathrm{r}}$ are constructed. (c) Two VPLs and shading points from each cluster are sampled from the pair $(\mathbb{C}^L_{\mathrm{r}},\mathbb{C}^G_{\mathrm{r}})$, then the pixel value $\hat{\mathcal{I}}$, error $\Delta\hat{\mathcal{I}}$, and standard deviation are estimated. (d) One of the clusters is subdivided if the estimated error $\Delta\hat{\mathcal{I}}$ is greater than tolerance $\epsilon\hat{\mathcal{I}}$. A cluster is selected based on the length of the diagonal of its bounding box weighted by the numbers of VPLs and shading points in each cluster. Here, the VPL cluster $\mathbb{C}^L_{\mathrm{r}}$ is selected and replaced with two child nodes $\mathbb{C}^{L_1}$ and $\mathbb{C}^{L_2}$; two new pairs $(\mathbb{C}^{L_1},\mathbb{C}^G_{\mathrm{r}})$ and $(\mathbb{C}^{L_2},\mathbb{C}^G_{\mathrm{r}})$ are created. The pixel value is estimated from the two pairs $(\mathbb{C}^{L_1},\mathbb{C}^G_{\mathrm{r}})$ and $(\mathbb{C}^{L_2},\mathbb{C}^G_{\mathrm{r}})$.

the estimated error $\Delta\hat{\mathcal{I}} = |\hat{\mathcal{I}} - \mathcal{I}|$ are initialized, and the standard deviation $\sigma$ is calculated (see Section 4.2). Then we repeat the following processes:

1. Choose the pair $(\mathbb{C}_i^L, \mathbb{C}_i^G)$ in $Q$ with the maximum standard deviation.
2. Subdivide one of the two clusters, $\mathbb{C}_i^L$ or $\mathbb{C}_i^G$, and move down one step in the corresponding binary tree. Cluster selection is described in Section 4.3. Replace the selected cluster with two child nodes and create two new pairs.
3. Update $\hat{\mathcal{I}}$ and $\Delta\hat{\mathcal{I}}$, and calculate the standard deviations for the two new pairs. Push the two new pairs into $Q$.
4. Terminate the process if the estimated error, $\Delta\hat{\mathcal{I}}$, is smaller than the tolerance, $\epsilon\hat{\mathcal{I}}$, where $\epsilon$ is the relative error tolerance. Otherwise, return to Step 1.

### 4.2 Estimating $\mathcal{I}$ and $\Delta\mathcal{I}$

Outgoing radiances $\mathcal{I}_i$ due to the shading cluster $\mathbb{C}_i^G$ illuminated from VPL cluster $\mathbb{C}_i^L$ are calculated by

$$\mathcal{I}_i = \sum_{\boldsymbol{x}\in\mathbb{C}_i^G} \sum_{\boldsymbol{y}\in\mathbb{C}_i^L} I(\boldsymbol{y})W(\boldsymbol{x})f(\boldsymbol{y},\boldsymbol{x},\boldsymbol{x}_\mathrm{v})V(\boldsymbol{y},\boldsymbol{x})G(\boldsymbol{y},\boldsymbol{x}) \tag{3}$$

As summing over all the VPLs and shading points in clusters $\mathbb{C}_i^L$ and $\mathbb{C}_i^G$ is computationally expensive, our method estimates $\mathcal{I}_i$ by sampling a small number of VPLs and shading points as follows:

$$\hat{\mathcal{I}}_i = \frac{1}{K}\sum_{k=1}^{K} \frac{I(\boldsymbol{y}_k)W(\boldsymbol{x}_k)f(\boldsymbol{y}_k,\boldsymbol{x}_k,\boldsymbol{x}_\mathrm{v})V(\boldsymbol{y}_k,\boldsymbol{x}_k)G(\boldsymbol{y}_k,\boldsymbol{x}_k)}{p_G(\boldsymbol{x}_k)p_L(\boldsymbol{y}_k)} \tag{4}$$

where $\boldsymbol{x}_k$ and $\boldsymbol{y}_k$ are the $k$-th sample of the shading point and VPL, respectively, $K$ is the number of samples, and $p_G$ and $p_L$ are probability mass functions for sampling, calculated as follows:

$$p_G(\boldsymbol{x}) = \frac{W(\boldsymbol{x})}{\sum_{\boldsymbol{x}'\in\mathbb{C}_i^G} W(\boldsymbol{x}')}$$

$$p_L(\boldsymbol{y}) = \frac{I(\boldsymbol{y})}{\sum_{\boldsymbol{y}'\in\mathbb{C}_i^L} I(\boldsymbol{y}')}$$

Substituting these functions into Eq. (4), $\hat{\mathcal{I}}_i$ is calculated from:

$$\hat{\mathcal{I}}_i = \frac{I_{\mathbb{C}_i^L} W_{\mathbb{C}_i^G}}{K} \sum_{k=1}^{K} f(\boldsymbol{y}_k,\boldsymbol{x}_k,\boldsymbol{x}_\mathrm{v})V(\boldsymbol{y}_k,\boldsymbol{x}_k)G(\boldsymbol{y}_k,\boldsymbol{x}_k) \tag{5}$$

where $I_{\mathbb{C}_i^L} = \sum_{\boldsymbol{y}'\in\mathbb{C}_i^L} I(\boldsymbol{y}')$ and $W_{\mathbb{C}_i^G} = \sum_{\boldsymbol{x}'\in\mathbb{C}_i^G} W(\boldsymbol{x}')$. The estimated pixel value $\hat{\mathcal{I}}$ is calculated by summing $\hat{\mathcal{I}}_i$ over all pairs.

Computing the error $\Delta\mathcal{I} = |\hat{\mathcal{I}} - \mathcal{I}|$ necessitates knowing the true value of $\mathcal{I}$ in Eq. (2), but the computational cost of obtaining this value is high. Therefore, our method estimates $\Delta\hat{\mathcal{I}}$ using the following equation [5]:

$$\Delta\hat{\mathcal{I}} = t_\alpha \sqrt{\sum_{i=1}^{N} s_i^2} \tag{6}$$

where $t_\alpha$ is the $\alpha$ quantile of the $t$-distribution, $N$ is the number of pairs, and $s_i^2$ is the sample variance of $\mathcal{I}_i$ for the $i$-th pair of $(\mathbb{C}_i^L, \mathbb{C}_i^G)$; Eq. (6) is derived in the Appendix.

To select a pair of clusters to be subdivided, the standard deviation $\sigma_i$ for each pair $(\mathbb{C}_i^L, \mathbb{C}_i^G)$ is required. Although the sample variance $s_i^2$ can be used to estimate $\sigma_i$, the accuracy of this estimate is low, since our method estimates $\hat{\mathcal{I}}_i$ with $K = 2$ samples, as does the previous method [5]. Instead of using the sample variance, our method calculates the standard deviation $\sigma_i$ for the $i$-th pair using the upper bounds of $f$ and $G$, as follows:

$$\sigma_i = I_{\mathbb{C}_i^L} W_{\mathbb{C}_i^G} f_\mathrm{ub}(\mathbb{C}_i^L, \mathbb{C}_i^G) G_\mathrm{ub}(\mathbb{C}_i^L, \mathbb{C}_i^G) \sqrt{\mathrm{Var}\,[V]} \tag{7}$$

where $f_\mathrm{ub}$ and $G_\mathrm{ub}$ are the upper bounds of $f$ and $G$, respectively, within the VPL cluster $\mathbb{C}_i^L$ and shading cluster $\mathbb{C}_i^G$. $\sqrt{\mathrm{Var}\,[V]}$ is the standard deviation of the visibility function; a maximum value of 0.5 is used. $f_\mathrm{ub}$ and $G_\mathrm{ub}$ for $\mathbb{C}_i^L$ and $\mathbb{C}_i^G$ are calculated in a similar way to those in MDLC.

### 4.3 Cluster selection

We now determine which cluster of the two clusters in the pair with maximum standard deviation is to be subdivided. In MDLC, the cluster is chosen by a refinement heuristic; it basically selects the cluster with the largest bounding box. Although this works well in some cases, it predominantly selects the VPL cluster $\mathbb{C}^L$, since VPLs are usually distributed throughout a scene, whereas the shading points are distributed locally, as shown by Fig. 2. In addition, the numbers of VPLs and shading points are often substantially different (e.g., the number of VPLs in Fig. 1 is 35.1k while the number of shading points is only 256). These facts cause unequal subdivisions between VPL clusters and shading clusters, which results in lower estimation accuracy. To address this problem, we propose a cluster selection method that accounts for the size difference between the bounding boxes of the root nodes of the VPL cluster and the

shading cluster. We also consider the number of VPLs and shading points when selecting the cluster to be subdivided. Specifically, the bounding box for each shading cluster is scaled using the following two coefficients:

$$c_l = \frac{l_{\mathbb{C}_\mathrm{r}^L}}{l_{\mathbb{C}_\mathrm{r}^G}} \tag{8}$$

$$c_d = \frac{l_{\mathbb{C}_\mathrm{r}^L}/|\mathbb{L}|}{l_{\mathbb{C}_\mathrm{r}^G}/|\mathbb{G}|} \tag{9}$$

where $\mathbb{C}_\mathrm{r}^L$ and $\mathbb{C}_\mathrm{r}^G$ are the root nodes of the VPL and shading clusters, respectively, $l_{\mathbb{C}_\mathrm{r}^L}$ and $l_{\mathbb{C}_\mathrm{r}^G}$ are the diagonal lengths of the bounding boxes for $\mathbb{C}_\mathrm{r}^L$ and $\mathbb{C}_\mathrm{r}^G$, respectively, and $|\mathbb{L}|$ and $|\mathbb{G}|$ are the numbers of VPLs and shading points, respectively.

### 4.4 Anti-aliasing and DOF

In many-light rendering with anti-aliasing or DOF, $N_\mathrm{spp}$ viewing rays are generated via sampling on the screen pixel or camera lens, and the point of intersection between each viewing ray and the surface in the scene is calculated. The material function $f$ at each shading point $\boldsymbol{x}_i$ is represented by a BRDF $f_\mathrm{r}$. As we use a simple box filter, the weighting function is calculated by $W(\boldsymbol{x}_i) = 1/N_\mathrm{spp}$; other filters (e.g., Gaussian filters) can also be applied.

Here, for simplicity, we describe methods for rendering translucent materials and participating media with a single ray per pixel; however, with simple modifications, our method can render these materials using multiple rays, as shown in Fig. 8.

### 4.5 Rendering translucent materials

The outgoing radiance at $\boldsymbol{x}_\mathrm{o}$ on the surface of a translucent material in viewing direction $\boldsymbol{\omega}_\mathrm{o}$ due to multiple scattered light within the translucent material is calculated using the diffuse bidirectional scattering surface reflectance distribution function (BSSRDF) $R_\mathrm{d}$ [20]: subsurface scattering of light at $\boldsymbol{x}_j$ in viewing direction $\boldsymbol{\omega}_j$ is given by

$$\frac{T_\eta(\boldsymbol{\omega}_j)}{\pi}\int_A\int_\Omega R_\mathrm{d}(r_{ij})L(\boldsymbol{x}_i,\boldsymbol{\omega}_i)T_\eta(\boldsymbol{\omega}_i)(\boldsymbol{n}_i\cdot\boldsymbol{\omega}_i)\mathrm{d}\boldsymbol{\omega}_i\mathrm{d}A(\boldsymbol{x}_i)$$

where $T_\eta$ is the Fresnel transmittance, $\boldsymbol{x}_i$ and $\boldsymbol{n}_i$ are a point and the normal to the surface of the translucent material at $\boldsymbol{x}_i$, respectively, $r_{ij} = \| \boldsymbol{x}_i - \boldsymbol{x}_j \|$, $A$ is the set of points on the surface of the translucent material, and $\Omega$ is a set of directions over the hemisphere.

Our method traces $N_\mathrm{spp}$ rays through each pixel and defines the point of intersection between the $j$-th ray and the surface of the translucent material as $\boldsymbol{x}_j$. The pixel value due to the subsurface scattering of light is calculated by

$$\frac{T_\eta(\boldsymbol{\omega}_j)}{\pi}\int_A R_\mathrm{d}(r_{ij})\sum_{\boldsymbol{y}\in\mathbb{L}} I(\boldsymbol{y})T_\eta(\boldsymbol{\omega}_i)V(\boldsymbol{y},\boldsymbol{x}_i)\cdot G(\boldsymbol{y},\boldsymbol{x}_i)\mathrm{d}A(\boldsymbol{x}_i) \tag{10}$$

where $\boldsymbol{\omega}_i = (\boldsymbol{y}-\boldsymbol{x}_i)/\| \boldsymbol{y}-\boldsymbol{x}_i \|$. Our method samples $|\mathbb{G}|$ shading points around $\boldsymbol{x}_j$ using the probability density function $p_\mathrm{d}$, which is as proportional as possible to the diffuse BSSRDF $R_\mathrm{d}$ [21]. Using importance-sampled shading points, Eq. (10) becomes

$$\frac{T_\eta(\boldsymbol{\omega}_j)}{\pi|\mathbb{G}|}\sum_{\boldsymbol{x}_i\in\mathbb{G}}\frac{R_\mathrm{d}(r_{ij})}{p_\mathrm{d}(\boldsymbol{x}_i)}\sum_{\boldsymbol{y}\in\mathbb{L}} I(\boldsymbol{y})T_\eta(\boldsymbol{\omega}_i)V(\boldsymbol{y},\boldsymbol{x}_i)G(\boldsymbol{y},\boldsymbol{x}_i) \tag{11}$$

By comparing Eqs. (11) and (2), the weighting function $W$ and material function $f$ are represented as follows:

$$W(\boldsymbol{x}_i) = \frac{T_\eta(\boldsymbol{\omega}_j)}{\pi|\mathbb{G}|}\frac{R_\mathrm{d}(r_{ij})}{p_\mathrm{d}(\boldsymbol{x}_i)}$$

$$f(\boldsymbol{y},\boldsymbol{x}_i,\boldsymbol{x}_\mathrm{v}) = T_\eta\left(\frac{\boldsymbol{y}-\boldsymbol{x}_i}{\| \boldsymbol{y}-\boldsymbol{x}_i \|}\right)$$

### 4.6 Rendering participating media

Like MDLC, our method assumes homogeneous participating media and an isotropic phase function. To calculate a pixel value in the presence of homogeneous participating media, the scattered radiances are integrated along the viewing ray, as shown in Fig. 2(b). Let $\boldsymbol{x}_\mathrm{o}$ be the point of intersection of the viewing ray and the surface in the scene. Thus, the outgoing radiance from $\boldsymbol{x}_\mathrm{o}$ is calculated using the sum of the reflected radiance $L_\mathrm{s}$ at $\boldsymbol{x}_\mathrm{o}$ and the scattered radiance $L_\mathrm{m}$ along the viewing ray, as follows:

$$L_\mathrm{s}(\boldsymbol{x}_\mathrm{o},\boldsymbol{x}_\mathrm{v}) = \tau(\boldsymbol{x}_\mathrm{o},\boldsymbol{x}_\mathrm{v})L(\boldsymbol{x}_\mathrm{o},\boldsymbol{x}_\mathrm{v}) \tag{12}$$

$$L_\mathrm{m}(\boldsymbol{x}_\mathrm{o},\boldsymbol{x}_\mathrm{v}) = \sigma_\mathrm{s}\int_0^{\|\boldsymbol{x}_\mathrm{o}-\boldsymbol{x}_\mathrm{v}\|}\tau(\boldsymbol{x}(t),\boldsymbol{x}_\mathrm{v})L(\boldsymbol{x}(t),\boldsymbol{x}_\mathrm{v})\mathrm{d}t \tag{13}$$

where $\tau(\boldsymbol{x}_\mathrm{o},\boldsymbol{x}_\mathrm{v}) = \exp(-\sigma_\mathrm{t}\| \boldsymbol{x}_\mathrm{o}-\boldsymbol{x}_\mathrm{v} \|)$ is the transmittance, and $\sigma_\mathrm{t}$ and $\sigma_\mathrm{s}$ are the extinction and scattering coefficients, respectively. $\boldsymbol{x}(t)$ is the point on the viewing ray parameterized by the distance $t$ from viewpoint $\boldsymbol{x}_\mathrm{v}$. $L(\boldsymbol{x}_\mathrm{o},\boldsymbol{x}_\mathrm{v})$ in Eq. (12) is calculated using Eq. (1). The weighting function $W$ and material function $f$ for $L_\mathrm{s}$ are represented by $W(\boldsymbol{x}_\mathrm{o}) = \tau(\boldsymbol{x}_\mathrm{o},\boldsymbol{x}_\mathrm{v})$ and the BRDF $f_\mathrm{r}(\boldsymbol{y},\boldsymbol{x}_\mathrm{o},\boldsymbol{x}_\mathrm{v})$, respectively. The scattered radiance $L(\boldsymbol{x}(t),\boldsymbol{x}_\mathrm{v})$ at

$\boldsymbol{x}(t)$ is calculated using:

$$L(\boldsymbol{x}(t), \boldsymbol{x}_\mathrm{v}) = \sum_{\boldsymbol{y}\in\mathbb{L}} I(\boldsymbol{y}) f_\mathrm{p}(\boldsymbol{y}, \boldsymbol{x}(t), \boldsymbol{x}_\mathrm{v}) V(\boldsymbol{y}, \boldsymbol{x}(t)) G(\boldsymbol{y}, \boldsymbol{x}(t))$$

where $f_\mathrm{p}$ is the (isotropic) phase function.

To compute the integral in Eq. (13), shading points $\boldsymbol{x}_i$ are generated by uniformly subdividing the viewing ray with step size $\Delta t$; $L_\mathrm{m}$ is calculated by summing over the shading points and VPLs as follows:

$$\sum_{\boldsymbol{x}_i\in\mathbb{G}} \sigma_\mathrm{s}\tau(\boldsymbol{x}_i, \boldsymbol{x}_\mathrm{v})\Delta t \sum_{\boldsymbol{y}\in\mathbb{L}} I(\boldsymbol{y}) f_\mathrm{p}(\boldsymbol{y}, \boldsymbol{x}_i, \boldsymbol{x}_\mathrm{v}) \cdot V(\boldsymbol{y}, \boldsymbol{x}_i) G(\boldsymbol{y}, \boldsymbol{x}_i)$$

Comparing this equation with Eq. (2), the material function $f$ is represented by the phase function $f_\mathrm{p}$, and the weighting function $W(\boldsymbol{x}_i)$ is calculated using $W(\boldsymbol{x}_i) = \sigma_\mathrm{s}\tau(\boldsymbol{x}_i, \boldsymbol{x}_\mathrm{v})\Delta t$.

## 5 Results

Figures 1 and 4–8 show our results. Table 1 shows the numbers of VPLs and shading points, as well as the computational time for our method and MDLC measured using a PC with an Intel Xeon E5-2650 v4 2.20 GHz CPU. In all the calculations, the relative tolerance $\epsilon$ was set to 2%, and the $\alpha$ quantile for Eq. (6) was set to 95%. To measure the estimation accuracy, we defined $R_\epsilon$ as the percentage of pixels satisfying $\| \hat{\mathcal{I}} - \mathcal{I} \| < \epsilon\mathcal{I}$; the estimation is accurate when $R_\epsilon$ is close to $\alpha$ (95% in our experiments).

Figure 1 shows results of our rendering method with anti-aliasing, where the insets compare (a) the reference solution, (b) our method, and (c) MDLC. The reference solution is rendered by summing up all the contributions from the VPLs at all the shading points to calculate the pixel values. Insets (d) and (e) illustrate the relative errors and $R_\epsilon$ for our method and MDLC. This figure demonstrates that our method can improve upon the estimation accuracy achieved using MDLC, with corresponding improvements in noise, as shown in Fig. 1(c). Figure 4 shows a San Miguel scene rendered with DOF. Figure 5 shows a Cornell box scene filled with homogeneous participating media. As shown in insets (c) of Figs. 4 and 5, the stochastic noise is perceptible when using MDLC, whereas it is imperceptible using our error estimation method (see insets (b)). Figure 6 shows a Sponza scene filled with homogeneous participating media. Figure 7 shows a Cornell box scene where the two boxes consist of translucent material. Figure 8 shows a human head model rendered with a diffuse BSSRDF and anti-aliasing. In all of these experiments, our method showed improved estimation accuracy $R_\epsilon$ over MDLC.

**Table 1** Statistics for our method and MDLC

| Fig. | Image resolution | $\lvert\mathbb{L}\rvert$ | $\lvert\mathbb{G}\rvert$ | Computational time | |
|---|---|---|---|---|---|
| | | | | Our | MDLC |
| 1 | $640 \times 480$ | 35.1k | 256 | 42.5 min | 6.2 min |
| 4 | $640 \times 480$ | 35.1k | 128 | 45.3 min | 5.3 min |
| 5 | $512^2$ | 539k | 256 | 35.6 s | 21.5 s |
| 6 | $512^2$ | 657k | 256 | 27.2 s | 19.8 s |
| 7 | $512^2$ | 124k | 1024 | 58.5 s | 52.9 s |
| 8 | $512^2$ | 33.2k | 2048 | 35.2 s | 29.1 s |

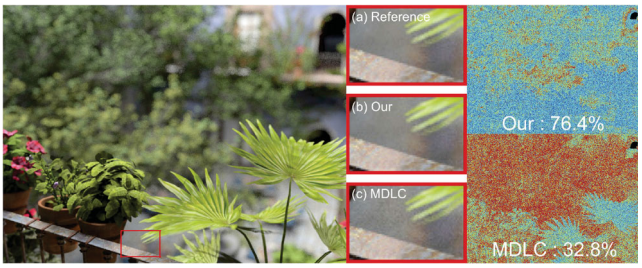



**Fig. 4** San Miguel scene with DOF.

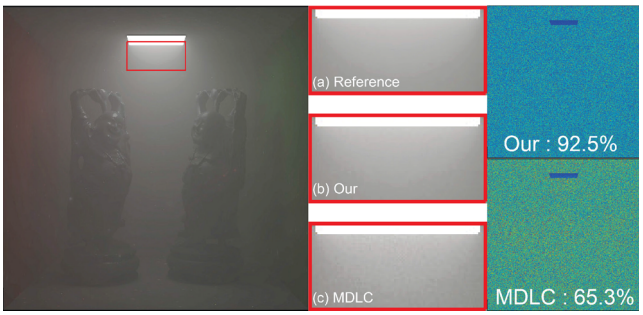



**Fig. 5** Cornell box scene with participating media.

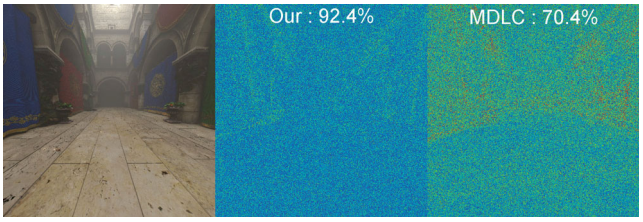



**Fig. 6** Sponza scene with participating media.

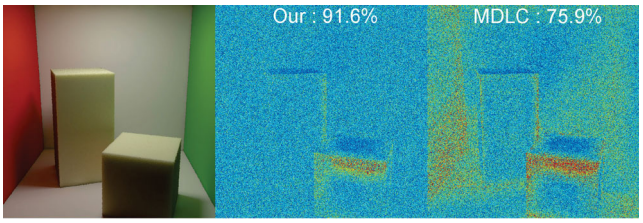



**Fig. 7** Cornell box scene with two boxes of translucent materials.

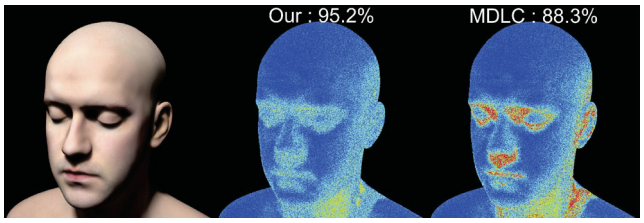



**Fig. 8** Human head scene (subsurface scattering with anti-aliasing (8 spp)).

In Figs. 9 and 10, cluster selections without scaling (the second column), scaling by $c_l$ (the third column), and scaling by $c_d$ (the fourth column) are compared. Scaling by $c_d$ yields the best estimation accuracy $R_\epsilon$ for five out of six of these scenes. Moreover, comparing results for these scenes without scaling with those with scaling demonstrates that scaling by $c_d$ consistently improves the estimation accuracy. In Fig. 9(a), the first row of the San Miguel scene with anti-aliasing shows relative errors greater than 5%, especially around the ivy-covered wall in the first column, but are reduced in the third column. In Fig. 9(d), the fourth row of the kitchen scene with DOF, relative errors greater than 5% can be seen, especially around the shadow boundaries and outlines. We think that these low estimation accuracies arise from the uneven cluster subdivisions (i.e., shading clusters are not subdivided, whereas VPL clusters are predominantly subdivided). Comparing scalings by $c_l$ and $c_d$ shows that estimation accuracies in the three scenes deteriorate when using $c_l$, although $c_l$ yields the best estimation accuracy in the San Miguel scene with DOF (c). Based on these experiments, we used the scaling coefficient $c_d$ to select the clusters.

While our method outperformed MDLC in estimation accuracy, its computational time was from 1.11 to 8.55 times greater. We attempted to adjust

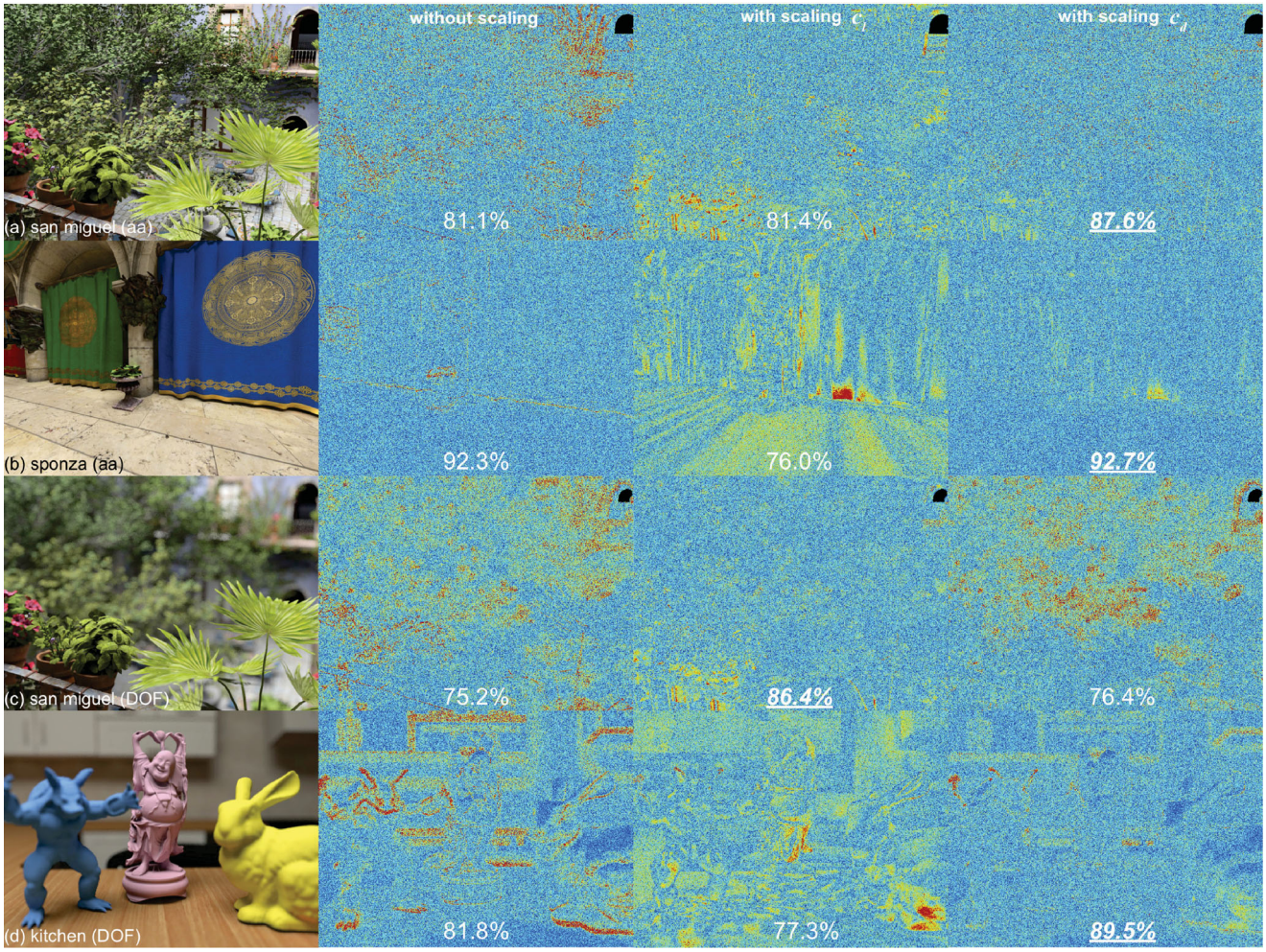



**Fig. 9** Cluster selection without scaling (the second column), scaling by $c_l$ (the third column), and scaling by $c_d$ (the fourth column). The left column shows the reference solutions and the other images show the relative errors in false color. The values in each relative error image show $R_\epsilon$. Scaling using $c_d$ yields better estimation accuracies than not scaling or scaling by $c_l$.

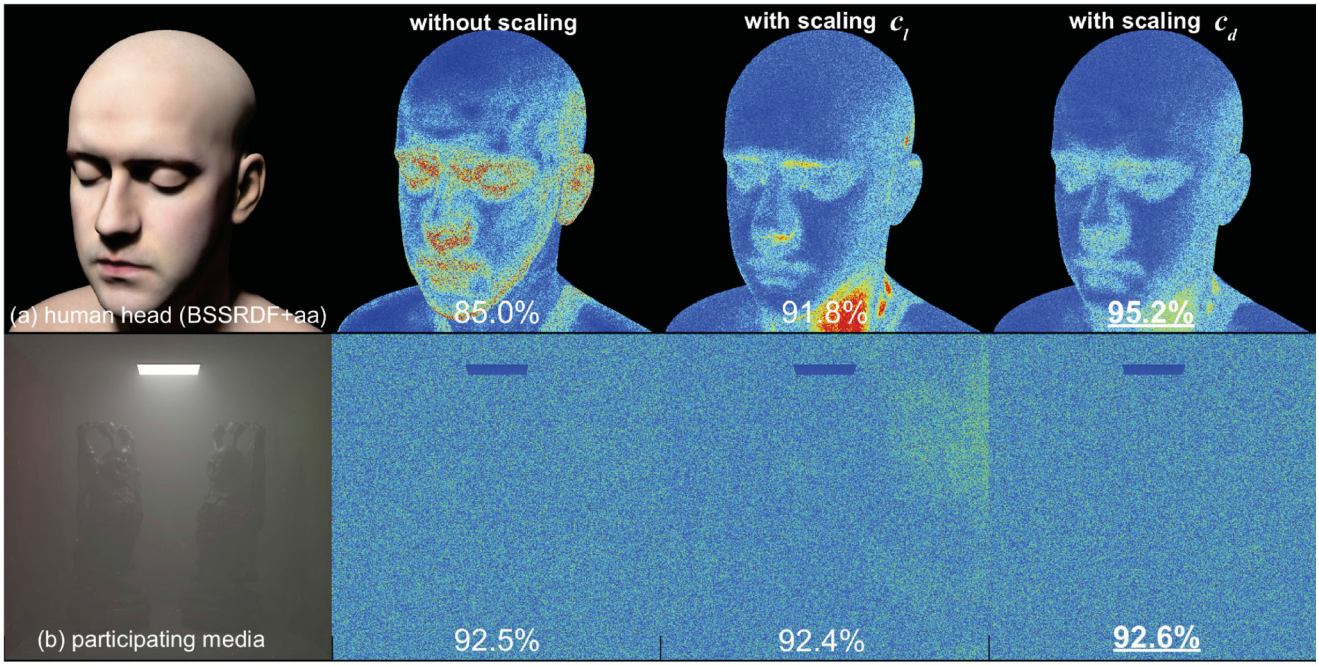



**Fig. 10** Cluster selection without scaling (the second column), with scaling by $c_l$ (the third column), and scaling by $c_d$ (the fourth column) for subsurface scattering (above) and participating media (below). Although scaling by $c_l$ provides better estimation accuracy $R_\epsilon$ than not scaling in the human head scene (above), relative errors greater than 5% can be seen in the neck. Scaling using $c_d$ provides the best estimation accuracy and relative errors greater than 5% are not seen. In the Cornell-box scene with homogeneous participating media (below), since VPLs and shading points are distributed throughout the scene, the improvements due to the use of $c_d$ are subtle, but its use does not impair the estimation accuracy.

the parameter $\epsilon$ so that the estimation accuracy of MDLC was similar to our result in Fig. 5. After several trial-and-error processes and re-renderings, $R_\epsilon$ of MDLC became 91.2% with $\epsilon = 0.5\%$. The computational time for MDLC with $\epsilon = 0.5\%$ was 33.5 s, which is comparable to our result (35.6 s). However, our method does not require the tedious trial-and-error processes needed for MDLC.

## 6 Conclusions and future work

We have presented a scalable many-light rendering method that can improve the estimation accuracy for various visual and illumination effects. Our method automatically partitions VPL and shading clusters so that pixel errors are smaller than the relative tolerance $\epsilon$.

Currently, our method is limited to homogeneous participating media with isotropic scattering. We would like to lift this limitation in future work. Moreover, we intend to apply our method to motion blur.

### Acknowledgements

This work was partially supported by JSPS KAKENHI 15H05924 and 18H03348.

## Appendix Derivation of the error estimate $\Delta\mathcal{I}$

If the samples in each pair follow a normal distribution and Neyman allocation is used for the number of samples for each pair, their statistics $T$ follow a $t$-distribution with $(n - N)$ degrees of freedom:

$$T = \sqrt{\frac{n-N}{n}} \frac{\hat{\mathcal{I}} - \mathcal{I}}{\sqrt{\sum_{i=1}^{N} s_i^2/n_i}} \tag{14}$$

where $N$ is the number of pairs, $n_i$ is the number of samples for the $i$-th pair $(\mathbb{C}_i^L, \mathbb{C}_i^G)$, $n$ is the total number of samples (i.e., $n = \sum_{i=1}^{N} n_i$), and $s_i$ is the sample variance of the $i$-th pair. The error $\Delta\mathcal{I} = \hat{\mathcal{I}} - \mathcal{I}$ is calculated using the $\alpha$ quantile in the $t$-distribution, $t_\alpha$, as

$$\Delta\mathcal{I} = t_\alpha \sqrt{\frac{n}{n-N} \sum_{i=1}^{N} \frac{s_i^2}{n_i}} \tag{15}$$

To estimate the pixel value, it is known to be more efficient to use a large number of pairs with a small number of samples than a small number of pairs with many samples. Since at least two samples for each pair is required to calculate the sample variance $s_i$, we sample two VPLs and shading points from the

pair ($K = 2$ in Eq. (4)). By substituting $n_i = 2$ and $n = 2N$ in Eq. (15), the error $\mathcal{I}$ can be simplified to give Eq. (6)

## References


[1] Dachsbacher, C.; Křivánek, J.; Hašan, M.; Arbree, A.; Walter, B.; Novák, J. Scalable realistic rendering with many-light methods. *Computer Graphics Forum* Vol. 33, No. 1, 88–104, 2014.

[2] Walter, B.; Fernandez, S.; Arbree, A.; Bala, K.; Donikian, M.; Greenberg, D. P. Lightcuts: A scalable approach to illumination. *ACM Transactions on Graphics* Vol. 24, No. 3, 1098–1107, 2005.

[3] Ou, J.; Pellacini, F. LightSlice: Matrix slice sampling for the many-lights problem. *ACM Transactions on Graphics* Vol. 30, No. 6, Article No. 179, 2011.

[4] Huo, Y.; Wang, R.; Jin, S.; Liu, X.; Bao, H. A matrix sampling-and-recovery approach for many-lights rendering. *ACM Transactions on Graphics* Vol. 34, No. 6, Article No. 210, 2015.

[5] Nabata, K.; Iwasaki, K.; Dobashi, Y.; Nishita, T. An error estimation framework for many-light rendering. *Computer Graphics Forum* Vol. 35, No. 7, 431–439, 2016.

[6] Walter, B.; Arbree, A.; Bala, K.; Greenberg, D. P. Multidimensional lightcuts.*ACM Transactions on Graphics* Vol. 25, No. 3, 1081–1088, 2006.

[7] Křivánek, J.; Georgiev, I.; Kaplanyan, A. S.; Cañada, J. Recent advances in light transport simulation: Theory and practice. In: Proceedings of the ACM SIGGRAPH 2013 Courses, Article No. 4, 2013.

[8] Keller, A. Instant radiosity. In: Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, 49–56, 1997.

[9] Hašan, M.; Pellacini, F.; Bala, K. Matrix row-column sampling for the many-light problem. *ACM Transactions on Graphics* Vol. 26, No. 3, Article No. 26, 2007.

[10] Georgiev, I.; Křivánek, J.; Popov, S.; Slusallek, P. Importance caching for complex illumination. *Computer Graphics Forum* Vol. 31, No. 2, 701–710, 2012.

[11] Yoshida, H.; Nabata, K.; Iwasaki, K.; Dobashi, Y.; Nishita, T. Adaptive importance caching for many-light rendering. *Journal of WSCG* Vol. 23, No. 1, 65–72, 2015.

[12] Wu, Y.-T.; Chuang, Y.-Y. VisibilityCluster: Average directional visibility for many-light rendering. *IEEE Transactions on Visualization and Computer Graphics* Vol. 19, No. 9, 1566–1578, 2013.

[13] Engelhardt, T.; Novák, J.; Schmidt, T.-W.; Dachsbacher, C. Approximate bias compensation for rendering scenes with heterogeneous participating media. *Computer Graphics Forum* Vol. 31, No. 7, 2145–2154, 2012.

[14] Novák, J.; Nowrouzezahrai, D.; Dachsbacher, C.; Jarosz, W. Virtual ray lights for rendering scenes with participating media. *ACM Transactions on Graphics* Vol. 31, No. 4, Article No. 60, 2012.

[15] Frederickx, R.; Bartels, P.; Dutré, P. Adaptive LightSlice for virtual ray lights. In: Proceedings of Eurographics 2015 Short Paper, 61–64, 2015.

[16] Huo, Y.; Wang, R.; Hu, T.; Hua, W.; Bao, H. Adaptive matrix column sampling and completion for rendering participating media. *ACM Transactions on Graphics* Vol. 35, No. 6, Article No. 167, 2016.

[17] Arbree, A.; Walter, B.; Bala, K. Single-pass scalable subsurface rendering with lightcuts. *Computer Graphics Forum* Vol. 27, No. 2, 507–516, 2008.

[18] Walter, B.; Khungurn, P.; Bala, K. Bidirectional lightcuts. *ACM Transactions on Graphics* Vol. 31, No. 4, Article No. 59, 2012.

[19] Wu, Y. T.; Li, T. M.; Lin, Y. H.; Chuang, Y. Y. Dual-matrix sampling for scalable translucent material rendering. *IEEE Transactions on Visualization and Computer Graphics* Vol. 21, No. 3, 363–374, 2015.

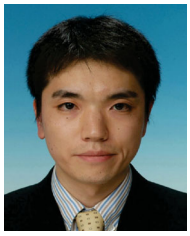
[20] Jensen, H. W.; Marschner, S. R.; Levoy, M.; Hanrahan, P. A practical model for subsurface light transport. In: Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, 511–518, 2001.

[21] King, A.; Kulla, C.; Conty, A.; Fajardo, M. BSSRDF importance sampling. In: Proceedings of the ACM SIGGRAPH 2013 Talks, Article No. 48, 2013.


**Hirokazu Sakai** received his B.S. degree from Wakayama University in 2017. He is currently an M.S. student at Wakayama University.

**Kosuke Nabata** received his B.S. and M.S. degrees from Wakayama University in 2013 and 2015, respectively. He is currently a Ph.D. student at Wakayama University.

**Shinya Yasuaki** received his B.S. and M.S. degrees from Wakayama University in 2015 and 2017, respectively. He is currently working at Square Enix Co., Ltd.



**Kei Iwasaki** received his B.S., M.S., and Ph.D. degrees from the University of Tokyo, in 1999, 2001, and 2004, respectively. He is currently an associate professor at Wakayama University.