Volume 39 (2020), Number 6 pp. 192–203 COMPUTER GRAPHICS *forum*

Spherical Gaussian-based Lightcuts for Glossy Interreflections

Y.C. Huo[1], S.H. Jin[1], T. Liu[2], W. Hua[1], R. Wang[†1] and H.J. Bao[1]

[1]State Key Lab of CAD&CG, Zhejiang University
[2]College of Transport & Communications, Shanghai Maritime University

Abstract

It is still challenging to render directional but non-specular reflections in complex scenes. The SG-based (Spherical Gaussian) many-light framework provides a scalable solution but still requires a large number of glossy virtual lights to avoid spikes as well as reduce clamping errors. Directly gathering contributions from these glossy virtual lights to each pixel in a pairwise way is very inefficient. In this paper, we propose an adaptive algorithm with tighter error bounds to efficiently compute glossy interreflections from glossy virtual lights. This approach is an extension of the Lightcuts that builds hierarchies on both lights and pixels with new error bounds and new GPU-based traversal methods between light and pixel hierarchies. Results demonstrate that our method is able to faithfully and efficiently compute glossy interreflections in scenes with highly glossy and spatial varying reflectance. Compared with the conventional Lightcuts method, our approach generates light cuts with only one fourth to one fifth light nodes therefore exhibits better scalability. Additionally, after being implemented on GPU, our algorithms achieves a magnitude of faster performance than the previous method.

CCS Concepts

• ***Computing methodologies*** → ***Ray tracing;***

1. Introduction

Efficiently and accurately rendering glossy scenes is a common demand. However, the directional, but non-specular reflections of glossy materials make such task challenging. Many existing techniques are able to generate images with high realism, but require a long time, especially for scenes with many glossy surfaces. The Monte Carlo-based path tracing methods [Vea98] simulate the light transports in a stochastic way and produce unbiased results. However, the convergence of Monte Carlo path tracing usually comes at a significant cost. The photon mapping [Jen01] and its extension [HOJ08] are capable of generating high-quality image, but require a large number of samples. The instant radiosity [Kel97] and many follow-up methods based on virtual point lights are efficient at handling diffuse materials. However, when applied to glossy materials, these methods produce inaccurate material perception unless dense sampling is performed. While there are some methods to represent the interreflections of glossy material, they have poor scalability because of poor performance or huge memory consumption.

In this paper, we propose a GPU-based and error-bounded adaptive algorithm for efficient rendering glossy interreflections. Our approach is extended from the Lightcuts methods [WFA*05a, WABG06] but introduces two main new features. First, our method is able to handle glossy interreflections with SG virtual lights and pixels in contrast to only handling omni, oriented and directional lights and diffuse pixels. Second, we propose new error bounds on

Figure 1: *King's Treasure. The scene contains about 50k direct lights, 1600k diffuse lights, 1600k glossy lights, 300k diffuse pixels and 290k glossy pixels. Our method is able to quickly and accurately computed the illumination in 30 seconds using approximated error bound E_a proposed in our paper.*

† Corresponding Author

integrals of a group of pixels and lights instead of evaluating one single integral for one pixel. In such a way, our method not only utilizes the coherence of lights but also explores the reflectance and visibility coherence among pixels. This enables extra efficiency and scalability, especially for glossy interreflections, the directional reflections or reflections that can be estimated by the contributions of a small portion of pairs of lights and pixels.

Our adaptive algorithm constructs two kinds of global hierarchies on lights and pixels. In each node, a Spherical Gaussian-based representative is positioned to represent the contribution of lights or pixels in the node. By defining new error bounds on clustered lights and pixels, we use these representatives in shading computation to accelerate the actual pairwise integrals of lights and pixels. For different kinds of light and pixel pairs, we derive different error bounds, especially the one for computing glossy to glossy interreflections. To better utilize the coherence of reflectance and visibilities across pixels, we further propose two kinds of error bounds, the conservative and approximate error bounds. Based on these error bounds, a GPU-based traversal algorithm across light and pixel hierarchies is developed to efficiently accumulate the illuminations. Figure 1 shows a result of using our method to render glossy interreflective effects including highly glossy and spatial varying reflectance. Our method faithfully captures these effects and completes the rendering in 30 seconds.

2. Related Works

Many lights methods. Instant radiosity [Kel97] generates a number of virtual point lights (VPLs) from light sources. It replaces the computation of indirect diffuse illumination by direct diffuse illumination from these virtual point lights. The same key idea was then formulated as a many-light problem and inspired a series of works. Walter et al. [WFA*05a] propose Lightcuts method that uses hierarchy on virtual point lights to reduce the cost of gathering contributions from light to pixels from linear to sublinear. Then it was extended to multidimensional Lightcuts [WABG06] with bidirectional hierarchy for solving high dimensional rendering problems, such as rendering volume scattering, depth of field or motion blur. However, the error metrics of these methods are all designed for evaluating the integral of a pixel and unable to bound multiple integrals of pixels. IlluminationCut [BMB15] extends the bidirectional hierarchy to accelerate the gathering process between many pixels and diffuse VPLs. Hašan et al. [HPB07] formulate the many-light problem as a matrix sampling problem and suggests sampling a small number of rows and columns from the original, large matrix of light-pixel pairs. Some improvements have been achieved by using the matrix slicing and sparse matrix sampling method [OP11, HWJ*15]. Nabata et al. [NIDN16] estimates the error due to clustering in a way of analyzing stratified sampling with confidence intervals. Stochastic Lightcuts [Yuk19] uses stochastic sampling to eliminate sampling correlation of Lightcuts. Liu et al. [LXY19] propose to sample lights based on importance values estimated using the BRDF's contributions.

However, these methods can handle diffuse illuminations well but fail at applying for scenes with interreflections between glossy surfaces.

For scenes with glossy materials, Hašan et al. [HKWB09] extend the VPLs to virtual spherical light (VSLs) but fail to capture some local glossy interreflections. Multidimensional Lightcuts and RichVPLs propose to use cube map to represent the directional emisson on glossy surfaces but the high memory consumption make them inappropriate for a large number of VPLs [WKB12, SHD15]. Davidovič et al. [DKH*10] introduce an alternative solution that combines global and local virtual lights to compensate for the loss of clamping and offers better approximation for sharp glossy reflections. However, it still needs to connect local lights to global lights, which is not scalable for scenes with many global lights and local lights. Some point-based models are used to represent the glossy interreflections. Laurijssen et al. [LWDB10] propose to model the glossy BRDF at a surface point as a directional distribution, using a spherical von Mises-Fisher (vMF) distribution. Spherical Gaussian is a more common model. Xu et al. [XCM*14] approximate the accurate glossy interreflections with facet to facet spherical Gaussian integration, and Tokuyoshi et al. [Tok15a, Tok15b, Tok16] approximate the total radiant intensity and positional distribution of VPLs using Spherical Gaussians and Gaussian distribution. Even anisotropic glossy material can be efficiently represented by anisotropic spherical Gaussians [XSD*13]. These methods are able to efficiently render glossy interreflections with small number of VPLs but are unsuitable for high-quality many-light rendering because they are limited to only one bounce. Walter et al. [WKB12] propose a high-quality hybrid algorithm that introduces path tracing-based eye paths to extend [WABG06] in glossy scenes. However, it relies on many multiple bounce eye paths to capture glossy effect, therefore cannot explore the glossy virtual lights to efficiently capture multiple-bounce glossy interreflections, especially those from light sources.

In this paper, we extend the hierarchical framework proposed by the Lightcuts [WFA*05a, WABG06] and propose an approach to utilize the coherence on pixels and further apply it for glossy interreflections.

Path tracing methods. Path tracing methods have been widely used for simulating global illumination. The original method is proposed by Whitted [Whi80]. Later, many methods have been developed by introducing stochastic merits [CPC84], applying to rendering equation [Kaj86] and employing Monte Carlo sampling techniques. The bidirectional path tracing combines the light path tracing and eye path tracing to combine the advantages of the path tracing as well as the light tracing [Vea98]. The main problem of Monte Carlo path tracing is that its variance may be seen as noise in the rendered images. To eliminate the noises, those path tracing methods usually cost a long time.

Photon mapping proposed by Jensen et al. [Jen01] traces photons in the scene and then caches and reuses them for final gathering of eye path illuminations. However, the photon mapping usually requires a large number of photons to reduce noise, especially for glossy interreflections, which results in high memory consumption. The progressive photon mapping [HOJ08] alleviates such a problem by multi-pass processing. But the convergence of the method is still quite slow.

Cache-based methods. Irradiance caching [WRC88] is a popular technique that progressively caches diffuse irradiance samples

as an octree, and reuses them during the computation. Radiance caching [KG05] extends it by recording directional radiance using spherical harmonics. Bala et al. [BDT99] present a general approach to exploiting both spatial and temporal coherence of rays for radiance interpolation. Okan Arikan et al. [AFO05] introduce a fast approximation to global illumination by decomposing radiance fields into far- and near-field components, which are computed separately to improve efficiency. For glossy interreflections, Václav et al. [GKB09] propose a spatial-directional cache method for glossy to glossy reflections. However, they still require sequentially inserting spatial sample points into a data structure, which has to be frequently queried and updated during the computation. Thus, they are hard to be implemented in the current parallel hardware framework. Wang et al. [WWZ*09] propose a GPU-based irradiance caching solution. However, it is only able to handle diffuse to diffuse interreflections.

Cache-based methods have some limitations. First, the error metric used to determine cache points is usually only on local geometric variations, which is unreliable for scenes with glossy reflections. Second, the interpolation scheme may introduce error when the resolution is not enough. Our method also utilizes the spatial coherence among pixels, but we consider the illumination difference as the error bound to directly computing instead of interpolating the illumination.

3. Algorithm

According to the rendering equation in area formulation, the outgoing radiance L at a shading point $\mathbf{x}$ to the eye is computed by the following integral:

$$L(\mathbf{x},\omega_o) = \int_S L_{\mathbf{y}}(\overrightarrow{\mathbf{yx}}) M_{\mathbf{x}}(\omega_o,\overrightarrow{\mathbf{yx}}) V(\mathbf{x},\mathbf{y}) G(\mathbf{x},\mathbf{y}) dA_y, \quad (1)$$

where S is the set of all surface points, $L_{\mathbf{y}}(\overrightarrow{\mathbf{yx}})$ is the incident radiance from a surface point $\mathbf{y}$ to $\mathbf{x}$, $V(\mathbf{x},\mathbf{y})$ is the visibility, $M_{\mathbf{x}}(\omega_o,\overrightarrow{\mathbf{yx}})$ is the BRDF of $\mathbf{x}$ and $G(\mathbf{x},\mathbf{y})$ is the geometry term that depends on the relative geometric relationship between $\mathbf{x}$ and $\mathbf{y}$. To simplify the notation, we will drop the ω_o since it's determinate given a camera position and a surface point $\mathbf{x}$. In the many-light framework, Equation 1 can be solved by creating a number of virtual point lights then gathering their contributions to pixels. To avoid per pixel-light pair computation, the Lightcuts [WFA*05a] approximates the contributions of all lights with the contributions of some representative lights. We follow the idea of light hierarchy, then further extend it to pixel hierarchy. Let us define the L_q to be the light representative of a set of point lights, $\{L_{\mathbf{y}_i}\} = \mathbf{D}$, and M_p to be the representative of a set of pixels, $\{M_{\mathbf{x}_j}\} = \mathbf{C}$. The illumination from a light cluster $\mathbf{D}$ to pixel $\mathbf{x}$ can be approximated by using the representative light, material, and the geometric and visibility terms between these two representatives. Formally, the equation is

$$\begin{aligned} L(\mathbf{x}) &= \sum_{\mathbf{y}_i \in \mathbf{D}} L_{\mathbf{y}_i}(\overrightarrow{\mathbf{y}_i\mathbf{x}}) M(\mathbf{x},\overrightarrow{\mathbf{y}_i\mathbf{x}}) G(\mathbf{x},\mathbf{y}_i) V(\mathbf{x},\mathbf{y}_i) \quad (2) \\ &\approx L_{\mathbf{p}}(\overrightarrow{\mathbf{qp}}) M_{\mathbf{p}}(\overrightarrow{\mathbf{qp}}) G(\mathbf{p},\mathbf{q}) V(\mathbf{p},\mathbf{q}) = L_{\mathbf{pq}}. \quad (3) \end{aligned}$$

Such an approximation in shading leads us to construct two kinds of hierarchies both on lights and pixels. For each node in these hierarchies, the representative is computed beforehand and used at run-time to approximate all leaf nodes (lights or pixels) under the node. The error incurred by such an approximation is

$$E_{\mathbf{pq}}(\mathbf{x}) = \| L_{\mathbf{pq}} - L(\mathbf{x}) \| . \quad (4)$$

The goal of our approach is to find a tight and efficient bound on the error. Thus, it can guide the traversal in two hierarchies and reduce the shading cost. According to their types of material, we categorize pixels into ones with diffuse reflectance and with glossy reflectance. We use spherical Gaussians (SGs) to represent non-diffuse BRDFs as they have been proved to be good approximations for empirical BRDF models or captured spatial varying BRDFs [WRG*09]. The glossy reflectance, $M_{\mathbf{x}}(\omega)$, and glossy VPL, $L_{\mathbf{y}}(\omega)$, are formulated in SG using N_k bases as:

$$M_{\mathbf{x}}(\omega) = \sum_{k=1}^{N_k} \mu_k e^{\lambda_k(\omega\cdot\mathbf{u}_k - 1)}, L_{\mathbf{y}}(\omega) = \sum_{k=1}^{N_k} \xi_k e^{\kappa_k(\omega\cdot\mathbf{v}_k - 1)}, \quad (5)$$

where ω is a direction defined in the local coordinate of $\mathbf{x}$ or $\mathbf{y}$, μ and ξ are lobe amplitudes, λ and κ are lobe sharpness, $\mathbf{u}$ and $\mathbf{v}$ are lobe axes that are also defined in the local coordinates, respectively. In this way, the glossy hierarchy can be organized in a uniform formation. In order to handle glossy interreflections, we also extend the light types proposed in the Lightcuts method [WFA*05a] (omni, oriented, and directional lights) with glossy lights for reflections from specular surface. The glossy reflection from pixels are also represented by the same formation. Such a formation is natural for reflections from specular surfaces, as after one bounce, all reflections are in spherical Gaussians formation.

In order to apply to general material, we use both SG VPLs and diffuse VPLs in our method. As discussed in previous work [WRG*09], the SG mixture can be applied to all-frequency material by using a varying lobe sharpness $\lambda \in (0, +\infty)$. The supplemental document of Wang et.al [WRG*09] provide detailed error analysis of a wide range of material including fitting a very small lobe sharpness to nearly diffuse Blinn-Phong material. For the case where $\lambda = 0$, i.e., purely diffuse material, we simply use the classic diffuse VPLs [WFA*05b].

3.1. Representatives of Lights and Pixels

For different types of clustered lights and pixels, we use different attributes to represent them. In Table 1, we list attributes used in this paper.

Attributes are divided into two types, the representative values or functions in shading computation or the maximum and minimum bounds for error evaluation. For pixels, we use $k_{\mathbf{C}}$, $\mathbf{n}_{\mathbf{C}}$, $\mathbf{v}_{\mathbf{C}}$ to represent material coefficient, normal and lobe direction and k_m and k_n, κ_m and κ_n, to denote maximum and minimum bounds of material coefficients and lobe shininess of the cluster. $\Delta\gamma_{\mathbf{n}}$ and $\Delta\beta_{\mathbf{v}}$ are orientation angle bounds of the normal cone and the lobe direction cone of the cluster. These bounds on lobe magnitude and shininess

Lights		Pixels
Omni	$I_{\mathbf{D}}$, $\mathbf{q}$	
Directional	$I_{\mathbf{D}}$, $\mathbf{d_D}$ $\Delta\theta_{\mathbf{d}}$	
Diffuse	$I_{\mathbf{D}}$, $\mathbf{q}$, $\mathbf{n_D}$, $\Delta\theta_{\mathbf{n}}$	$k_{\mathbf{C}}$, k_m, k_n, $\mathbf{p}$, $\mathbf{n_C}$, $\Delta\gamma_{\mathbf{n}}$
Glossy	$I_{\mathbf{D}}$, $\mathbf{q}$, $\mathbf{n_D}$, $\mathbf{v_D}$, $\lambda_{\mathbf{D}}$ λ_m, λ_n, $\Delta\theta_{\mathbf{n}}$, $\Delta\alpha_{\mathbf{v}}$	$k_{\mathbf{C}}$, k_m, k_n, $\mathbf{p}$, $\mathbf{n_C}$, $\mathbf{v_C}$ $\kappa_{\mathbf{C}}$, κ_m, κ_n, $\Delta\gamma_{\mathbf{n}}$, $\Delta\beta_{\mathbf{v}}$

Table 1: *Attributes of representative of clustered lights and pixels.*

enable us to correctly handle highly glossy and spatially varying BRDFs. For lights, we use $I_{\mathbf{D}}$ to represent the total light intensity and denote $\mathbf{d_D}$, $\mathbf{n_D}$ and $\mathbf{v_D}$ to the representative direction for directional lights, normal for diffuse and glossy lights and lobe direction for glossy lights, respectively. $\Delta\theta_{\mathbf{d}}$, $\Delta\theta_{\mathbf{n}}$ and $\Delta\alpha_{\mathbf{n}}$ are orientation angle bounds of cones of these three directions for leaves. Besides these attributes, we also compute and store the bounding box for each cluster.

For omni, directional and diffuse lights, the representative intensity, $I_{\mathbf{D}}$, can be computed by summing all intensity of children. Other attributes of lights, i.e. averaging directions of orientation or directional axis, can also be computed by averaging all values of children. For diffuse pixels, the representative reflectance coefficient is computed by averaging all values of children. For glossy lights or pixels, it is non-trivial to compute the best representative that concurrently approximates the positional and directional distributions in 3D space. In this paper, we separate the positional and directional variations and approximate them individually. The position of the representative is computed by averaging all children positions. The approximated lobe of the representative is in a similar way as that proposed in Banerjee et al. [BDGS05]. The equations to compute representatives for glossy lights are as follows:

$$\mathbf{q} = \frac{1}{N}\sum_i \mathbf{y}_i, \mathbf{n_D} = \frac{1}{N}\sum_i \mathbf{n}_{\mathbf{y}_i}, \mathbf{v_D} = \frac{\mathbf{r}_p}{\| \mathbf{r}_p \|},$$
$$\lambda_{\mathbf{D}} = \frac{3 \| \mathbf{r}_p \| - \| \mathbf{r}_p \|^3}{1 - \| \mathbf{r}_p \|^2}, I_{\mathbf{D}} = \frac{1}{\lambda_{\mathbf{D}}}\sum_i \frac{I_i}{\lambda_i}, \tag{6}$$

where $\mathbf{r}_p = \frac{1}{N}\sum_i \mathbf{v}_i$, The equations for glossy pixels are in a similar way except $k_{\mathbf{C}}$ is computed by averaging all $k_{\mathbf{x}_i}$ instead of summing them.

3.2. Error Bounds

Given two clusters of pixels and lights, $\mathbf{C}$ and $\mathbf{D}$ respectively, we are looking for the error bound of the illumination error for any pixels in $\mathbf{C}$ while using the light representative $L_{\mathbf{q}}$ and material representative $M_{\mathbf{p}}$. According to Equation 4, since the $L_{\mathbf{pq}}$ can be directly computed from $L_{\mathbf{q}}$ and $M_{\mathbf{p}}$, the challenge is to find two bounds, the maximum and minimum of actual illuminations, Lm and Ln.

Thus, the error $E_{\mathbf{CD}}(\mathbf{x})$ between clusters of pixels and lights, $\mathbf{C}$ and $\mathbf{D}$, can be derived from Equation 4 as:

$$E_{\mathbf{CD}}(\mathbf{x}) = \| L_{\mathbf{pq}} - L(\mathbf{x}) \| \leq \max\{L_{\mathbf{pq}} - Ln, Lm - L_{\mathbf{pq}}\}. \tag{7}$$

Note that in the following text, we omit the subscript $\mathbf{CD}$ of E since the derivation of error bound in this section is always between clusters $\mathbf{C}$ and $\mathbf{D}$, and use subscript m and n to indicate the maximum and minimum values.

According to Equation 2, the visibility term $V(\mathbf{x}, \mathbf{y}_i)$ is included in $L(\mathbf{x})$, thus the conservative error bound is $E_c(\mathbf{x}) = \max\{L_{\mathbf{pq}}, Lm\}$. This occurs when all $\mathbf{y}_i$ are invisible, namely $Ln = 0$, and the $\mathbf{q}$ is invisible for $\mathbf{p}$, namely $L_{\mathbf{pq}} = 0$. However such a criteria is too conservative for many cases, especially for scenes where glossy reflections is the primary reason of discord rather than visibility. Thus, besides the E_c, we take an assumption in Hašan et al. [HKWB09] that visibilities of clustered lights shares the visibility of the representative $\mathbf{p}$ and propose an approximated error bound, $E_a(\mathbf{x}) = \max\{L_{\mathbf{pq}} - Ln, Lm - L_{\mathbf{pq}}\}$, where $V(\mathbf{x}, \mathbf{y}_i) = V(\mathbf{p}, \mathbf{q})$.

Under our assumptions of lights and material, we need to compute four kinds of error bounds, when computing the illuminations of diffuse lights to diffuse pixels, E^{dd}, diffuse lights to glossy pixels, E^{ds}, glossy lights to diffuse pixels, E^{sd}, and glossy lights to glossy pixels, E^{ss}. Here, we give the derivation of glossy lights to glossy pixels, E^{ss}. Other error bounds can be readily derived from it by replacing the spherical Gaussian term with diffuse term accordingly. Please refer to Appendix A for more details.

To derive the error bound, we first derive the positional and directional relations of clustered lights, $\mathbf{D}$, and pixels, $\mathbf{C}$. For simplicity, we use spheres centered at $\mathbf{p}$ and $\mathbf{q}$, $S_{\mathbf{p}}$ and $S_{\mathbf{q}}$, to bound these pixels and lights, respectively. Thus, the minimum and maximum distances, d_n and d_m, from $\mathbf{D}$ to $\mathbf{C}$ can be computed by

$$d_n = d_{\mathbf{pq}} - r_{S_{\mathbf{p}}} - r_{S_{\mathbf{q}}}, d_m = d_{\mathbf{pq}} + r_{S_{\mathbf{p}}} + r_{S_{\mathbf{q}}}. \tag{8}$$

Among all possible connections of the lights to pixels between the bounding spheres, $S_{\mathbf{p}}$ and $S_{\mathbf{q}}$, the ones with most diverse directions lie on planes that are tangent to both spheres (Figure 2 illustrates a 2D case). The maximum angle, $\Delta\delta$, between rays on tangent planes and the direction $\overrightarrow{\mathbf{pq}}$ can be computed as:

$$\Delta\delta = \arcsin\frac{r_{S_{\mathbf{p}}} + r_{S_{\mathbf{q}}}}{d_{\mathbf{pq}}}, \tag{9}$$

where $r_{S_{\mathbf{p}}}$ and $r_{S_{\mathbf{q}}}$ are radii of the bounding spheres.

Given a pixel $\mathbf{x}$ in cluster $\mathbf{C}$ with normal $\mathbf{n_x}$, the maximum and minimum angle bounds, $\gamma_{\mathbf{x},m}$ and $\gamma_{\mathbf{x},n}$, for incident light from cluster $\mathbf{D}$ can be computed by

$$\begin{aligned} \gamma_{\mathbf{x},n} &= \max(0,\ \theta_{\mathbf{C},\mathbf{pq}} - \Delta\gamma_{\mathbf{x}} - \Delta\delta), \\ \gamma_{\mathbf{x},m} &= \min(\frac{\pi}{2},\ \theta_{\mathbf{C},\mathbf{pq}} + \Delta\gamma_{\mathbf{x}} + \Delta\delta), \end{aligned} \tag{10}$$

where max and min functions assure such an incident angle is between $0 \sim \frac{\pi}{2}$, $\theta_{\mathbf{C},\mathbf{pq}}$ is the angle between the representative normal, $\mathbf{n_C}$, and the direction $\overrightarrow{\mathbf{pq}}$, $\Delta\gamma_{\mathbf{x}}$ is the angle between the normal of $\mathbf{x}$ to the representative normal, $\mathbf{n_C}$ (Figure 2).

By replacing the $\Delta\gamma_{\mathbf{x}}$ with the normal orientation angle bounds of

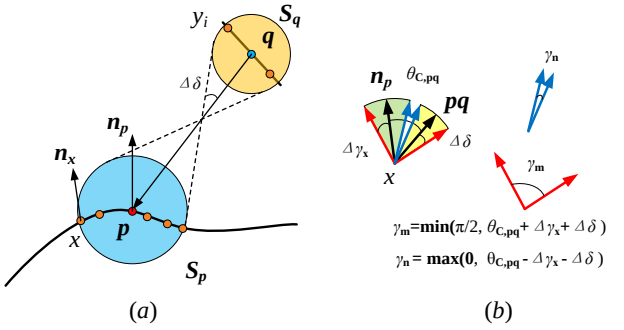


Figure 2: *Illustration of the maximum and minimum angle bounds, γ_m and γ_n, for directions of incident light from cluster **D** to **C**. (a) illustrates the maximum angle variation, $\Delta\delta$, of rays connecting lights and pixels in clusters **C** and **D**. In the 2D case, it is determined by the tangent line and the direction **pq**. (b) illustrates the computation of maximum and minimum angle bounds, γ_m and γ_n. The green region is the orientation bounding cone of cluster normal, **np**, determined by γ_n. The yellow region is the maximum angle variation, $\Delta\delta$. Thus, the maximum angle is potentially produced by two directions drawn in red, i.e.* $\min(\frac{\pi}{2},\ \theta_{\boldsymbol{C,pq}} + \Delta\gamma_{\boldsymbol{x}} + \Delta\delta)$*, and the minimum angle is produced by directions drawn in blue, i.e.* $\max(0,\ \theta_{\boldsymbol{C,pq}} - \Delta\gamma_{\boldsymbol{x}} - \Delta\delta)$

normal cone, $\Delta\gamma_{\mathbf{C}}$, we can get the maximum and minimum angle bounds for the incident light from cluster **D** to cluster **C**. We denote them γ_m and γ_n. Similarly, we can obtain the angle bounds for normal in cluster **D**, θ_m and θ_n, the angle bounds for lobe direction $\mathbf{v_C}$, α_m and α_n, and the angle bounds for lobe direction $\mathbf{v_D}$, β_m and β_n.

Given these variables, maximum and minimum bounds of the actual illuminations from glossy lights to glossy pixels, L_m^{ss} and L_n^{ss} can be computed as

$$\begin{aligned} L_m^{ss} &= I_{\mathbf{C}} k_m e^{\lambda m(\cos\beta m - 1)} e^{\kappa m(\cos\alpha m - 1)} \frac{\cos\gamma m \cos\theta m}{d_n^2}, \\ L_n^{ss} &= I_{\mathbf{C}} k_n e^{\lambda n(\cos\beta n - 1)} e^{\kappa n(\cos\alpha n - 1)} \frac{\cos\gamma n \cos\theta n}{d_m^2}. \end{aligned} \tag{11}$$

More detail derivations of Equation 11 and the formations of other value bounds, L^{dd}, L^{ds} and L^{sd} are given in Appendix A. Taking Equation 11 into Equation 7, we can get the error bounds of clustered glossy lights to glossy pixels.

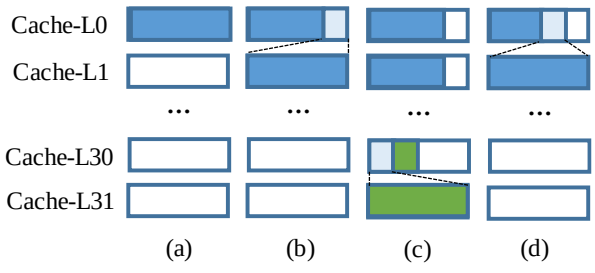


Figure 3: *An illustration of the GPU traversal process. An multi-level active node cache is maintained. (a) The initial nodes on the top levels of the tree hierarchies are inserted into the cache. (b) n_{max} active nodes are processed by the GPU at once. (c) Similar strategy of (b) is applied to lower level of the cache. If the visited nodes are leaves or the errors are smaller than the threshold, they are directly finished (marked by green). Otherwise, their children nodes are inserted into the lower level for further traversal. (d) While the lower level cache is emptied, another batch of nodes in the higher level is popped out for visiting until all nodes are visited.*

4. Light And Pixel Hierarchies

In this section, we introduce how to construct light and pixel hierarchies and how to conduct the traversal on the light and pixel hierarchies to accumulate the illuminations of light-pixel node pairs.

4.1. Construction of Hierarchies

The leaf nodes of the hierarchies store at least one element (light or pixel). Intermediate nodes are composed of two children nodes and store representative lights or pixels.

To create the hierarchy, we take a top-down clustering scheme that first cluster the entire elements (i.e. lights or pixels) into two clusters and place a root for these two clusters. Then, such a hierarchy is subsequently subdivided into subclusters until one cluster only contains less than a preset maximum number of elements or reaches a minimal error threshold.

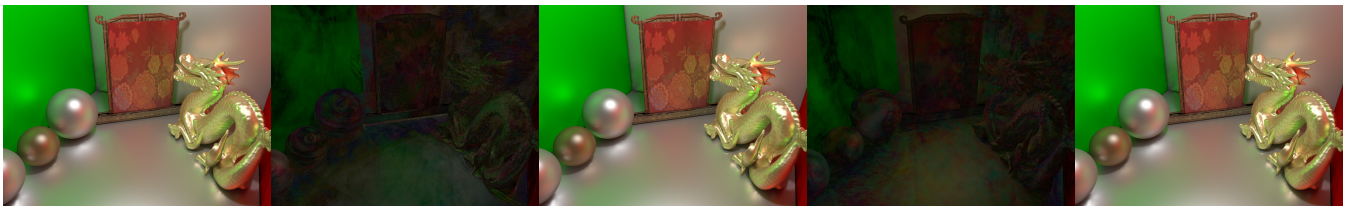
The error metric used in the clustering algorithm should reliably predict the spatial and directional changes of VPLs or the reflectance of pixels and provide tight partitions. In this paper, we define the clustering error metric as:

$$\begin{aligned} e(\mathbf{y}_i, \mathbf{y}_j) &= \| \mathbf{y}_i - \mathbf{y}_j \| + w_1 |1 - \mathbf{n}_i \cdot \mathbf{n}_j| \\ &+ w_2 (|\mu_i - \mu_j e^{\lambda_j(\mathbf{v}_i \cdot \mathbf{v}_j - 1)}| + |\mu_j - \mu_i e^{\lambda_i(\mathbf{v}_i \cdot \mathbf{v}_j - 1)}|), \end{aligned} \tag{12}$$

where w_1 and w_2 are weighting factors that determine the relative importance of the differences of positions, normals, and glossy lobes. Since the glossy lobes are distributed in the 3D space, it is hard to compute the accurate difference of two lobes as $\int_{\mathbf{H}} |\mu_i e^{\lambda_i(\mathbf{v}_i \cdot \mathbf{v} - 1)} - \mu_j e^{\lambda_j(\mathbf{v}_j \cdot \mathbf{v} - 1)}| d\mathbf{v}$. However, we observe that these two lobes have the maximum value at their own lobe directions. So we use the value difference at these two directions, the lobe directions of y_i and y_j, to approximate the error of two lobes. That is the latter part in Equation 12. Once the clusters have been subdivided, we proceed with the calculation of the representative of children and store them at the node.

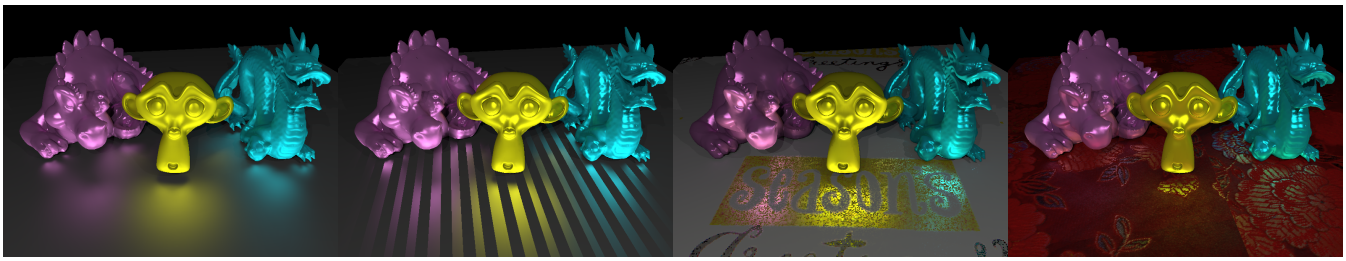
4.2. Traversal on Hierarchies

Once the light and pixel hierarchies are built, the next step is to employ them in shading. In this section, we describe how to use these error bounds to conduct hierarchical traversals.



(a) *Result using* E_a **(b)** *16x Error image*(E_a) **(c)** *Result using* E_c **(d)** *16x Error image*(E_c) **(e)** *Reference*

Figure 4: *A cornel box scene with glossy objects and spatial varying BRDF texture.*



(a) *With constant glossy shininess* **(b)** *With different BRDFs in strips* **(c)** *With SVBRDF texture, "card"* **(d)** *With SVBRDF texture, "satin"*

Figure 5: *Tableau scene*

4.2.1. Error Bounds Thresholds

To better determine the thresholds of error bounds, based on our rendering pipeline, we take a greedy error bounds detection process. In our method, illuminations from several hierarchies, i.e., the hierarchies of diffuse lights and diffuse pixels, the hierarchies of diffuse lights and glossy pixels, the hierarchies of glossy lights and diffuse pixels, and the hierarchies of glossy lights and glossy pixels, are computed sequentially. After one light hierarchy has been traversed, we use the ratio of 1% of already computed illumination of pixels as one of the error bound threshold, $E_{\mathbf{x},a}$. Because this error bound is defined in leaves, we can merge all the leaves' error bounds of node $\mathbf{p}$ as the node's error bound, $E_{\mathbf{p},a} = \min\{E_{\mathbf{x},a}\}$ for all pixels $\mathbf{x}$ belongs to node $\mathbf{p}$. While shading pixels with next light hierarchy, i.e., from the shading of diffuse lights to diffuse pixels to the shading of diffuse lights to glossy pixels, the relative error ratio, $E_{\mathbf{p},r}$, is combined with this error threshold stored at pixel to determine the error bound threshold, $E = \max\{E_{\mathbf{p},a}, E_{\mathbf{p},r}\}$. Such a greedy mechanic is useful to avoid overestimating errors in dark regions. For example, the glossy to glossy illumination can be relatively small compared to the diffuse to diffuse illumination. In that case, $E_{\mathbf{x},a}$ would help to stop the traversal between glossy lights and glossy pixels.

4.2.2. Bi-nodes Traversal on Hierarchies

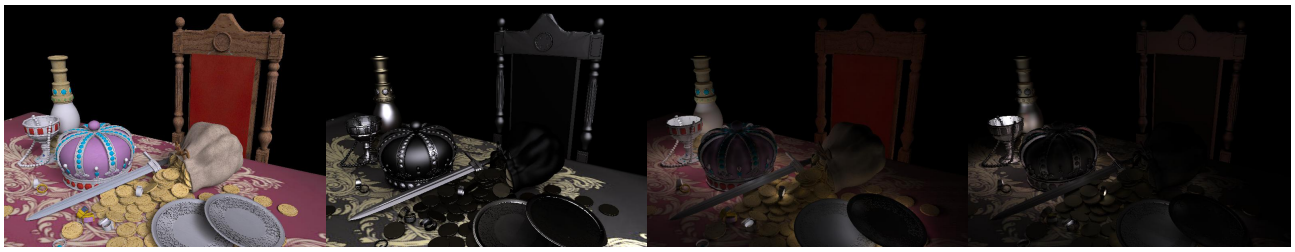
For both the light and pixel hierarchies, we take a top-down traversal scheme. For each of light node and pixel node, according to the type of these nodes, we compute the illumination from representatives (Equation 3) and the maximum and minimum bounds (Equation 11)). If its error bound is less than the error threshold, it indicates that there is no need to traverse further. Thus, we add this pair of light and pixel nodes in the output list for further visibility tests. If the error bound of this node is larger than a threshold, it suggests further traversals on the two hierarchies. To determine which hierarchy needs to be first traversed, we fetch two children of each node and compute the error bounds of the light children nodes with the parent pixel node and the counterparts of the pixel children nodes with the light parent node. The parent node generates larger error is selected as the next node to be refined and traversed. These children with parent node of the other hierarchy are made into pairs for future processes. Once the error bounds of all node pairs are less than the threshold, or both of the hierarchy has been traversed all down to the leaves, the bi-node traversal stops.

We conduct such a traversal algorithm in a parallel way and implement it on CUDA. In order to fully utilize parallel streaming processors, the traversal algorithm is organized into batches. We start with a lower level of nodes in both hierarchies (e.g., the fourth level in the implementation). Initial pairs of nodes are put into an active list. In each batch, nodes pairs in the active list are processed in parallel. To avoid the active list growing too fast, each time, we only process $n_{\max}$ (10K in our implementation) latest generated pairs in the list to traverse hierarchies in a depth-first fashion. Once the number of pairs in the output list reaches a maximum number, we suspend the traversal processing and resume it after taking the visibility tests on the pairs in the output list. Because we want to parallel and accurately estimate the illumination of image for error threshold evaluation, we unfold the first six levels of light trees as an initial active list to estimate the initial illuminations of pixels. This GPU-based traversal is based on a multi-level active node cache. Figure 3 shows an illustration of it with step-wise decomposition.

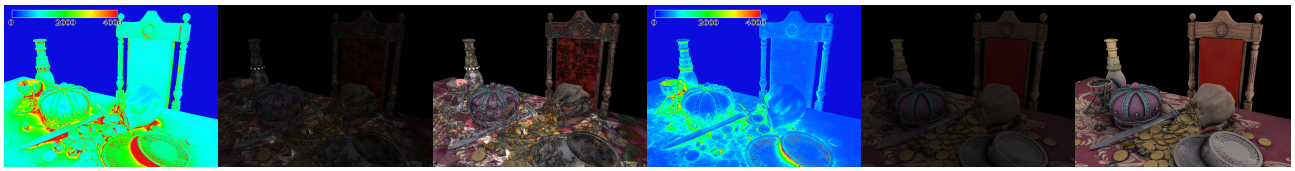
Scenes	Tris.	Lights			Pixels		Avg. Rays/Pixel	Times (s)
		#direct	#diffuse	#glossy	#diffuse	#glossy		
Box (Fig. 4)	135k	5k	2970k	2800k	479k	466k	2481(E_c),1059(E_a)	46(E_c), 25(E_a)
Tableau (Fig. 5d)	222k	3	520k	530k	397k	397k	1091(E_a)	34(E_a)
Treasure (Fig. 1)	918k	50k	1600k	1600k	300k	290k	2857(E_c), 1175(E_a)	53(E_c), 29(E_a)
Kitchen (Fig. 7)	184k	10k	2600k	2900k	480k	480k	3309(E_c), 1342(E_a)	76(E_c), 37(E_a)

Table 2: *Statistics of the test scenes. We list the number of triangles, the number of direct light sources, indirect lights and the number of diffuse and glossy pixels of scenes. We also give the average number of lights in the light nodes that are used for shading. The number indicates the efficiency of light hierarchies, the higher number, the more lights are approximated by the node representative. We also give the average number of pixels in the pixel nodes that are used in shading. The number indicates the efficiency of pixel hierarchies, the higher number, the more pixels are avoided per pixel shading computation and visibility tests. We divide the time into two part. The first part is spent on building hierarchies and the second one is used in shading computation.*

(a) *Result using the conservative error bound E_c* **(b)** *Result using approximate error bound E_a* **(c)** *Reference*



(d) *Directional & diffuse lights to diffuse pixels* **(e)** *Directional &diffuse lights to glossy pixels* **(f)** *Glossy lights to diffuse pixels* **(g)** *Glossy lights to glossy pixels*



(h) *#L-nodes/pixel, E_c* **(i)** *1x Error of E_c* **(j)** *16x Error of E_c* **(k)** *#L-nodes/pixel, E_a* **(l)** *1x Error of E_a* **(m)** *16x Error of E_a*

Figure 6: *King's treasure. (d), (e), (f) and (g) shows the decompositions of illuminations of different light and pixel combinations, which are rendered with E_a. The pseudo color figures illustrate the number of Lightcut nodes in each pixel.*

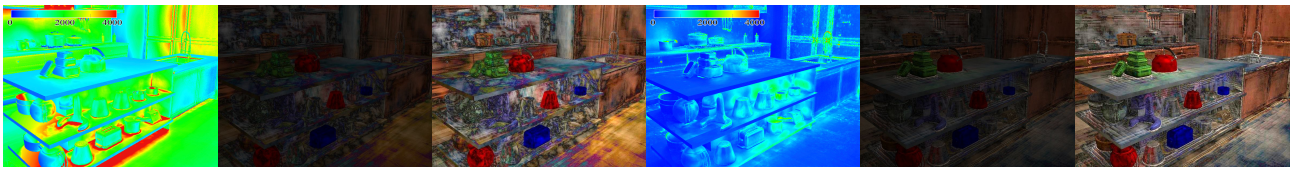
5. Results

In this section, we present our results computed on an Intel Xeon E620 2.40GHz workstation with a nVIDIA GeForce GTX TITAN graphics card. The entire algorithm is implemented on the GPU using CUDA.

5.1. Ablation Study

We test our algorithm on different scenes to verify our method. The spatial varying BRDF textures in these scenes are obtained from the websites of the authors of [WRG*09, LBAD*06]. The directional lights and indirect lights are generated beforehand. To capture these spatial varying reflections, we densely generate indirect lights on the surface with these spatial varying BRDF textures. At rendering,

(a) *Result using conservative error bound* E_c **(b)** *Result using approximate error bound* E_a **(c)** *Reference*



(d) *#L-nodes/pixel,* E_c **(e)** *1x Error of* E_c **(f)** *16x Error of* E_c **(g)** *#L-nodes/pixel,* E_a **(h)** *1x Error of* E_a **(i)** *16x Error of* E_a

Figure 7: *Kitchen. Comparison between conservative and approximate errors. The pseudo color figures illustrate the number of cut nodes per pixel. The error maps illustrate the scaled difference between various methods and the reference.*

we load all the lights with the scene. The statistics of our test scenes are reported in Table 2. Unless mentioned otherwise, all images reported are rendered at 800×600 image resolution. The reference image is generated by a brute force computation of all light and pixel pairs.

The first scene (Figure 4) contains a cornel box with spheres, dragon and a screen with spatial varying BRDFs. To light up the scene, we use one point light above these objects. The number of indirect lights and pixels are in Table 2. The lobe shininesses of spheres are 10, 40 and 80, of dragon is 80, of the floor is 200. The glossy interreflections of objects are visually significant. Though after one reflection, it is still able to recognize the shape of yellow flower patterns on the floor. Compared to the reference image, the error generated by our method is very small.

The Tableau scene (Figure 5) has three models with shininess varying from 40 to 80. The number of indirect lights and pixels and performance are reported in Table 2. We set the plane with four different materials, from constant glossy shininess, striped spatial varying BRDFs to two different spatial varying BRDF textures. All results are generated using the approximate error bound, E_a. Please note the glossy interreflections difference on the monkey and dragons.

The King's treasure scene (Figure 6) is composed of many glossy objects, the crown, the sword, the grail, dishes, gold coins, rings, etc. The shininesses of these objects vary from 20 to 200. The scene is lit by a captured HDR Kitchen environment map. Total 50k samples are generated by the method [ARBWJ03] to approximate the directional lights and one point light is added for better illumination. The number of indirect lights and pixels and performance are reported in Table 2. The total times to generate images using the conservative error bound of Lightcuts, our approximate error bound are 53 second and 29 second. From difference images between the reference image and our results, there are no visible differences. We visualize four types of interreflections in Figure 6(h)∼(k). It can be observed that each of them plays an important contribution to the final image.

The Kitchen scene (Figure 7) also consists of many objects with glossy materials. The shininesses of objects vary from 20 to 200. The scene is lit by a directional light (through the window) and twenty point lights. We also compare different approaches, E_c of Lightcuts and our method with E_a. Comparing with these results with reference, there are no visible differences.

5.2. Comparisons

We conduct an equal-time comparison between the proposed method (Ours) and other state-of-the-art methods, the Rich-VPLs [SHD15] and the IlluminationCut [BMB15]. All methods are adapted to GPU. Technically, the IlluminationCut has a bidirectional acceleration structure between pixels and VPLs but does not support glossy VPLs. Rich-VPLs discretizes incident radiance directions and stores directionally varying glossy lighting into cubemaps. Due to the large memory consumption of storing high resolution cubemaps, it cannot support a scene with a large number of glossy VPLs. Moreover, due the discretization of directions, it cannot handle high frequency but only low frequency glossy VPLs. Therefore, neither of them can produce equal-quality images as ours.

In the Kitchen 2 test scene (Figure 8), we apply different methods to render the same scene in 60 seconds and illustrate errors compared with the reference. Ours uses 2500k diffuse VPLs and 3000k glossy VPLs. The same VPLs numbers are used for testing IlluminationCut. The IlluminationCut does not natively support glossy

VPLs but we adapt it to the scene in two different ways. Figure 8 (a) uses diffuse VPLs to represent glossy VPLs for keeping energy. However, the lacking of glossy-to-glossy interreflections makes the result pale compared with the reference. Figure 8 (b) uses glossy VPLs for shading but clusters them with the technique of IlluminationCut by treating gloss VPL as diffuse VPL. It can preserve most specular features but have splotches here and there due to the clustering quality. Rich-VPLs uses cubemaps to discretize incident directions of glossy VPLs. Therefore, it has a memory consumption, e.g., 12 KB per VPL, and high overhead, whereas our SG-based glossy VPL requires only 52 bytes. The GPU implementation can only support 30k glossy VPLs for Rich-VPLs. The small number of glossy VPLs leads to spiky artifacts (Figure 8 (c) and (i)). The Rich-VPLs paper also discusses an extension by using von Mises-Fisher (vMF) to save memory and support larger number of VPLs. We also implement and test this extension (Figure 8 (d) and (j)). However, the vMF extension still requires 240 bytes per VPL and cannot accurately calculate error bounds after the approximation. Using 1500k glossy VPLs, the glossy interreflections are blurred and have high errors. In equal-time, ours has the best visual results and significantly lower errors compared with others (Figure 8 (e) and (k)). Results show that our method is suitable for scenes with a large number of VPLs. In addition, our bidirectional acceleration hierarchies and error bounds for SG-based VPLs significantly accelerate the gathering process while guaranteeing the quality.

5.3. Discussion

Error. Two error bounds are employed in our experiments, the conservative error bound, e_c, and the approximated error bound e_a. The conservative bound limits the maximum error per node to guarantee the smooth transitions of illumination with different cuts on hierarchies. As pointed out by Walter et al. [WFA*05a], such a per cluster error bound may sum up to large and visible errors. But in practice, it is unlikely to happen. The approximated error bound always assume leaves share the same visibility with the representative of the node. It may underestimate the errors of nodes with partial or complex occlusions. In our experiment, we found the error brought by such an assumption is relatively small and will not produce obvious artifacts. Using approximate error bound provides a tighter error estimation. From the Table 2, Figure 6(d) and (f) and Figure 7(d) and (f), it can be observed that the light nodes when using approximate error bound is less than that using the conservative one, which results in less rendering time.

Scalability. Compared to the naive approach to generate the reference image and the Lightcuts method, our approach has better scalability. Theoretically, given N lights and M pixels, the traversal operations of our method are $O(\log M \log N)$ vs. $O(M \log N)$ of Lightcuts method. Although Multidemensional Lightcuts also builds trees on gather points, it still relates to O(N) pixels. To demonstrate this, we varied the number of point lights and pixels in the kitchen scene. Figure 9 illustrates times vs. number of pixels for the naive solution, Lightcuts, IlluminationCut, Rich-VPLs and our method. It demonstrates the scalability of our method.

6. Conclusion

In summary, this paper presents an adaptive algorithm to render glossy interreflections. Our method can take advantage of the coherence among pairs of lights and pixels, thus computing the glossy reflections effectively. We have shown that the glossy and spatial varying BRDFs can be handled well. Compared to previous methods, our algorithm is able to control the accuracy and efficiency of rendering complex scenes with glossy materials.

There are several directions to be explored in the future. For example, the conservative error bounds proposed in this paper may be too conservative in many cases. However, the approximate error bound relying assumption on the visibility may fail in some cases. Exploring better conservative error bounds is a good future direction. Furthermore, a better approximation of glossy lobes will reduce the error bounds evaluated in shading, hence results in fast rendering speed. It will be another interesting problem worth pursuing. Another limitation of our method is that the method only support point-based virtual lights and thus cannot include VSL or other volume-based lights. We regard it as one future work.

Appendix A

In the appendix, we give detailed derivation of Equation 11 and formation of other actual illumination bounds, L^{dd}, L^{ds} and L^{sd}.

According to Equation 3, the illumination from light $\mathbf{y}$ to $\mathbf{x}$ is

$$L(\mathbf{x}) = \sum_{\mathbf{y}_i \in C} L_{\mathbf{y}_i}(\overrightarrow{\mathbf{y}_i\mathbf{x}}) M(\mathbf{x}, \overrightarrow{\mathbf{y}_i\mathbf{x}}) G(\mathbf{x}, \mathbf{y}_i) V(\mathbf{x}, \mathbf{y}_i) \tag{13}$$

Under the assumption to derive the approximate error bound that the visibilities of clustered lights shares the visibility of the representative $\mathbf{p}$, the $V(\mathbf{x}, \mathbf{y}_i)$ is 1. For each other term, $L_{\mathbf{y}_i}(\overrightarrow{\mathbf{y}_i\mathbf{x}})$, $M(\mathbf{x}, \overrightarrow{\mathbf{y}_i\mathbf{x}})$ and $G(\mathbf{x}, \mathbf{y}_i)$, we have

$$L_{\mathbf{y}_i}(\overrightarrow{\mathbf{y}_i\mathbf{x}}) = I_{\mathbf{y}_i} e^{\lambda_{\mathbf{y}_i}(\cos(\alpha_{\mathbf{y}_i})-1)}, M(\mathbf{x}, \overrightarrow{\mathbf{y}_i\mathbf{x}}) = k_{\mathbf{x}} e^{\kappa_{\mathbf{x}_i}(\cos(\beta_{\mathbf{x}_i})-1)}$$

$$G(\mathbf{x}, \mathbf{y}_i) = \frac{\cos\gamma_{\mathbf{n}_\mathbf{x}, \mathbf{x}\mathbf{y}_i} \cos\theta_{\mathbf{n}_\mathbf{y}, \mathbf{x}\mathbf{y}_i}}{d^2_{\mathbf{x}\mathbf{y}_i}}$$

$$L_{\mathbf{y}_i} e^{\lambda_n(\cos(\alpha_n)-1)} \le L_{\mathbf{y}_i}(\overrightarrow{\mathbf{y}_i\mathbf{x}}) \le L_{\mathbf{y}_i} e^{\lambda_m(\cos(\alpha_m)-1)}$$

$$k_n e^{\kappa_n(\cos(\beta_n)-1)} \le M(\mathbf{x}, \overrightarrow{\mathbf{y}_i\mathbf{x}}) \le k_m e^{\kappa_m(\cos(\beta_m)-1)}$$

$$\frac{\cos\gamma_n \cos\theta_n}{d_m^2} \le G(\mathbf{x}, \mathbf{y}_i) \le \frac{\cos\gamma_m \cos\theta_m}{d_n^2}$$

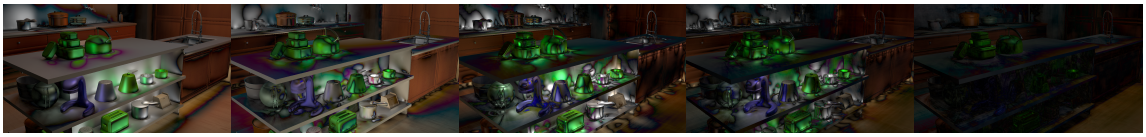
Taking these inequalities into Equation 13 and having $I_\mathbf{C} = \sum_{\mathbf{y}_i} I_{\mathbf{y}_i}$, we have the error bounds of L_m^{ss} and L_n^{ss} in Equation 11. By replacing the spherical Gaussian lobe into diffuse term, we have other formations of actual illumination bounds,

$$L_m^{dd} = I_\mathbf{C} k_m \frac{\cos\gamma_m \cos\theta_m}{d_n^2}$$

$$L_n^{dd} = I_\mathbf{C} k_n \frac{\cos\gamma_n \cos\theta_n}{d_m^2}$$

$$L_m^{ds} = I_\mathbf{C} k_m e^{\kappa_m(\cos\alpha_m - 1)} \frac{\cos\gamma_m \cos\theta_m}{d_n^2}$$

(a) *IlluminationCut w/o g.* **(b)** *IlluminationCut w/ g.* **(c)** *Rich-VPLs w/o vMF* **(d)** *Rich-VPLs w/ vMF* **(e)** *Ours* **(f)** *Reference*



(g) *Error of (a)* **(h)** *4x Error of (b)* **(i)** *4x Error of (c)* **(j)** *4x Error of (d)* **(k)** *4x Error of (e)*

Figure 8: *Kitchen 2. Equal-time (60 seconds) comparison of IlluminationCut, Rich-VPLs and Ours. The acronym g. means glossy VPLs.*

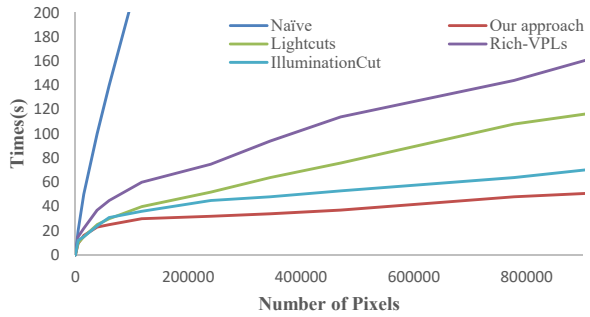


Figure 9: *Pixels scalability of the performance. We compare the scalability of the naive approach, Lightcuts, Rich-VPLs, IlluminationCut and our method. The error maps illustrate the scaled difference between various methods and the reference.*

$$
\begin{aligned}
L_n^{ds} &= I_{\mathbf{C}} k_n e^{\kappa_n(\cos\alpha_n - 1)} \frac{\cos\gamma_n \cos\theta_n}{d_m^2} \\
L_m^{sd} &= I_{\mathbf{C}} k_m e^{\lambda_m(\cos\beta_m - 1)} \frac{\cos\gamma_m \cos\theta_m}{d_n^2} \\
L_n^{sd} &= I_{\mathbf{C}} k_n e^{\lambda_n(\cos\beta_n - 1)} \frac{\cos\gamma_n \cos\theta_n}{d_m^2}
\end{aligned} \tag{14}
$$

Taking these error bounds of actual illumination into Equation 7, we have the error bounds for light and pixel clusters.

Acknowledgement

This work is partially supported by National Key R&D Program of China (No. 2017YFB1002605), NSFC (No. 61872319, 61602426), Zhejiang Provincial NSFC (No. LR18F020002), Key R&D Program of Zhejiang Province (2018C01090), Zhejiang University Education Foundation Global Partnership Fund and China Postdoctoral Science Foundation (2018M642432).

References

[AFO05] ARIKAN O., FORSYTH D. A., O'BRIEN J. F.: Fast and detailed approximate global illumination by irradiance decomposition. *ACM Trans. Graph. 24*, 3 (2005), 1108–1114. 194

[ARBWJ03] AGARWAL S., RAMAMOORTHI R., BELONGIE S., WANN JENSEN H.: Structured importance sampling of environment maps. *ACM Trans. Graph. 22*, 3 (2003), 605–612. 199

[BDGS05] BANERJEE A., DHILLON I. S., GHOSH J., SRA S.: Clustering on the unit hypersphere using von mises-fisher distributions. *J. Mach. Learn. Res. 6* (Dec. 2005), 1345–1382. 195

[BDT99] BALA K., DORSEY J., TELLER S.: Radiance interpolants for accelerated bounded-error ray tracing. *ACM Trans. Graph. 18*, 3 (1999), 213–256. 194

[BMB15] BUS N., MUSTAFA N. H., BIRI V.: Illuminationcut. In *Computer Graphics Forum* (2015), vol. 34, Wiley Online Library, pp. 561–573. 193, 199

[CPC84] COOK R. L., PORTER T., CARPENTER L.: Distributed ray tracing. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1984), SIGGRAPH '84, ACM, pp. 137–145. URL: http://doi.acm.org/10.1145/800031.808590, doi:http://doi.acm.org/10.1145/800031.808590. 193

[DKH*10] DAVIDOVIČ T., KŘIVÁNEK J., HAŠAN M., SLUSALLEK P., BALA K.: Combining global and local virtual lights for detailed glossy illumination. *ACM Trans. Graph. 29* (December 2010), 143:1–143:8. 193

[GKB09] GASSENBAUER V., KŘIVÁNEK J., BOUATOUCH K.: Spatial directional radiance caching. In *Computer Graphics Forum* (2009), vol. 28, Wiley Online Library, pp. 1189–1198. 194

[HKWB09] HAŠAN M., KŘIVÁNEK J., WALTER B., BALA K.: Virtual spherical lights for many-light rendering of glossy scenes. *ACM Trans. Graph. 28* (December 2009), 143:1–143:6. 193, 195

[HOJ08] HACHISUKA T., OGAKI S., JENSEN H. W.: Progressive photon mapping. In *SIGGRAPH Asia '08: ACM SIGGRAPH Asia 2008 papers* (New York, NY, USA, 2008), ACM, pp. 1–8. doi:http://doi.acm.org/10.1145/1457515.1409083. 192, 193

[HPB07] HAŠAN M., PELLACINI F., BALA K.: Matrix row-column sampling for the many-light problem. In *ACM SIGGRAPH 2007 papers* (New York, NY, USA, 2007), SIGGRAPH '07, ACM. URL: http://doi.acm.org/10.1145/1275808.1276410, doi:http://doi.acm.org/10.1145/1275808.1276410. 193

[HWJ*15] HUO Y., WANG R., JIN S., LIU X., BAO H.: A matrix sampling-and-recovery approach for many-lights rendering. *ACM Transactions on Graphics (TOG) 34*, 6 (2015), 210. 193

[Jen01] JENSEN H. W.: *Realistic image synthesis using photon mapping*. A. K. Peters, Ltd., 2001. 192, 193

[Kaj86] KAJIYA J. T.: The rendering equation. In *Proc. SIGGRAPH '86* (1986), pp. 143–150. 193

[Kel97] KELLER A.: Instant radiosity. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1997), SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., pp. 49–56. URL: http://dx.doi.org/10.1145/258734.258769, doi:http://dx.doi.org/10.1145/258734.258769. 192, 193

[KG05] KŘIVÁNEK J., GAUTRON P.: Radiance caching for efficient global illumination computation. *IEEE Trans. Visualization and Computer Graphics 11*, 5 (2005), 550–561. 194

[LBAD*06] LAWRENCE J., BEN-ARTZI A., DECORO C., MATUSIK W., PFISTER H., RAMAMOORTHI R., RUSINKIEWICZ S.: Inverse shade trees for non-parametric material representation and editing. *ACM Trans. Graph. 25*, 3 (2006), 735–745. 198

[LWDB10] LAURIJSSEN J., WANG R., DUTRÉ P., BROWN B. J.: Fast estimation and rendering of indirect highlights. In *Computer Graphics Forum* (2010), vol. 29, Wiley Online Library, pp. 1305–1313. 193

[LXY19] LIU Y., XU K., YAN L.-Q.: Adaptive brdf-oriented multiple importance sampling of many lights. In *Computer Graphics Forum* (2019), vol. 38, Wiley Online Library, pp. 123–133. 193

[NIDN16] NABATA K., IWASAKI K., DOBASHI Y., NISHITA T.: An error estimation framework for many-light rendering. In *Computer Graphics Forum* (2016), vol. 35, Wiley Online Library, pp. 431–439. 193

[OP11] OU J., PELLACINI F.: Lightslice: matrix slice sampling for the many-lights problem. *ACM Trans. Graph. 30*, 6 (Dec. 2011), 179:1–179:8. 193

[SHD15] SIMON F., HANIKA J., DACHSBACHER C.: Rich-vpls for improving the versatility of many-light methods. In *Computer Graphics Forum* (2015), vol. 34, Wiley Online Library, pp. 575–584. 193, 199

[Tok15a] TOKUYOSHI Y.: Fast indirect illumination using two virtual spherical gaussian lights. In *SIGGRAPH Asia 2015 Posters* (2015), ACM, p. 12. 193

[Tok15b] TOKUYOSHI Y.: Virtual spherical gaussian lights for real-time glossy indirect illumination. In *Computer Graphics Forum* (2015), vol. 34, Wiley Online Library, pp. 89–98. 193

[Tok16] TOKUYOSHI Y.: Modified filtered importance sampling for virtual spherical gaussian lights. *Computational Visual Media 2*, 4 (2016), 343–355. 193

[Vea98] VEACH E.: *Robust monte carlo methods for light transport simulation*. PhD thesis, Stanford, CA, USA, 1998. AAI9837162. 192, 193

[WABG06] WALTER B., ARBREE A., BALA K., GREENBERG D. P.: Multidimensional lightcuts. *ACM Trans. Graph. 25*, 3 (2006), 1081–1088. 192, 193

[WFA*05a] WALTER B., FERNANDEZ S., ARBREE A., BALA K., DONIKIAN M., GREENBERG D. P.: Lightcuts: a scalable approach to illumination. *ACM Trans. Graph. 24*, 3 (2005), 1098–1107. doi:http://doi.acm.org/10.1145/1073204.1073318. 192, 193, 194, 200

[WFA*05b] WALTER B., FERNANDEZ S., ARBREE A., BALA K., DONIKIAN M., GREENBERG D. P.: Lightcuts: a scalable approach to illumination. *ACM Trans. Graph. 24*, 3 (2005), 1098–1107. 194

[Whi80] WHITTED T.: An improved illumination model for shaded display. *Commun. ACM 23* (June 1980), 343–349. URL: http://doi.acm.org/10.1145/358876.358882, doi:http://doi.acm.org/10.1145/358876.358882. 193

[WKB12] WALTER B., KHUNGURN P., BALA K.: Bidirectional lightcuts. *ACM Trans. Graph. 31*, 4 (July 2012), 59:1–59:11. 193

[WRC88] WARD G. J., RUBINSTEIN F. M., CLEAR R. D.: A ray tracing solution for diffuse interreflection. In *Proc. SIGGRAPH '88* (1988), pp. 85–92. 193

[WRG*09] WANG J., REN P., GONG M., SNYDER J., GUO B.: All-frequency rendering of dynamic, spatially-varying reflectance. *ACM Trans. Graph. 28* (December 2009), 133:1–133:10. URL: http://doi.acm.org/10.1145/1618452.1618479, doi:http://doi.acm.org/10.1145/1618452.1618479. 194, 198

[WWZ*09] WANG R., WANG R., ZHOU K., PAN M., BAO H.: An efficient gpu-based approach for interactive global illumination. *ACM Trans. Graph. 28* (July 2009), 91:1–91:8. URL: http://doi.acm.org/10.1145/1531326.1531397, doi:http://doi.acm.org/10.1145/1531326.1531397. 194

[XCM*14] XU K., CAO Y.-P., MA L.-Q., DONG Z., WANG R., HU S.-M.: A practical algorithm for rendering interreflections with all-frequency brdfs. *ACM Transactions on Graphics (TOG) 33*, 1 (2014), 10. 193

[XSD*13] XU K., SUN W.-L., DONG Z., ZHAO D.-Y., WU R.-D., HU S.-M.: Anisotropic spherical gaussians. *ACM Transactions on Graphics (TOG) 32*, 6 (2013), 209. 193

[Yuk19] YUKSEL C.: Stochastic lightcuts. In *High-Performance Graphics (HPG 2019)* (2019), The Eurographics Association. doi:10.2312/hpg.20191192. 193