Scalable Spherical Harmonics Hierarchies

Jonathan B. Metzgar
University of Alaska Fairbanks
Department of Computer Science
PO Box 756670
Fairbanks, Alaska, USA, 99775-6670
jmetzgar@acm.org

Sudhanshu K. Semwal
University of Colorado Colorado Springs
Department of Computer Science
1420 Austin Bluffs Pkwy
Colorado Springs, Colorado, USA 80918
semwal@uccs.edu

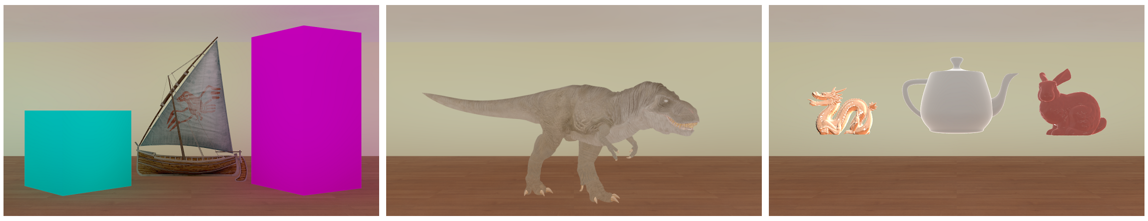


Figure 1: Scalable Spherical Harmonics Hierarchies used to generate global illumination.

ABSTRACT

Scalable Spherical Harmonics Hierarchies (SSPHH) is a real-time rendering solution to the global illumination problem. Our novel method is a system of components that enables the computation of light probes and the conversion to spherical harmonics coefficients, which are used as anisotropic time-varying light sources we call Spherical Harmonics Lights (SPHLs). The SPHLs encode irradiance information that can be used for image-based lighting. Our approach focuses on reconstructing scene lighting using diffuse illumination but is flexible to allow specular details. Furthermore, we consider the light transport from neighboring SPHLs by computing a transfer coefficient that estimates how much light from one probe is visible at another. SPHLs can be used for physically-based lighting using rendering methods similar to point lights and shadow maps. We created a reproducible testing methodology to compare our images with those of a commercial path tracer by automatically generating the appropriate scene information which gets used with absolute error metrics to determine remaining image defects. Our SSPHH method utilizes a scalable architecture to distribute the rendering of light probes between client and worker nodes using the ZeroMQ Majordomo protocol.

Keywords
spherical harmonics, anisotropic point lights, physically based rendering, path tracing, local dynamic radiance maps

1 INTRODUCTION

Real-time global illumination is a difficult problem to solve and understand for practitioners with numerous solutions proposed over the last several decades. Although new hardware is posing to change the focus to real-time path tracing, approximation methods will still have relevance for years to come due to problem complexity, power, space, and cost factors. Furthermore, it is difficult to offer solutions that solve the problem generally. Our work is a novel way of simulating global illumination using a network of light probes represented by spherical harmonics that estimates the transport of light from one node in the network to another. But let us begin by way of an illustration.

Imagine for a moment a house on a sunny day with several rooms, all with the doors closed and heavy curtains drawn. If you were to open the curtains in room A, photons immediately begin flooding the house and spreading through every crack they can find. In computer graphics, this room would be lit, and probably no light would leak through the cracks. Let us consider room B which has no direct illumination from room A, but now the door is open. If you were to open the door to room A, even though no direct light can reach B, indirect light is scattered from room A to room B. A path tracer would likely scatter photons to room B, but a typical real-time renderer using shadow maps renders no light in the second room. One typical solution for the real-time renderer uses baked global illumination, but baking forgoes the use of dynamic changes in scenery. So what can we do?

Our method supposes that you could configure light probes in each of the rooms and hallways and a set of edges connecting them. Then we will determine a visibility or transfer factor between neighboring vertices that would determine the amount of light that scatters from one light probe location to another. We use spherical harmonics to represent the visibility factor and the light probes. We use a four-part method of initialization, light probe visibility determination, light probe generation, and hierarchical spherical harmonic light generation. These four parts are components of our algorithm *Scalable Spherical Harmonics Hierarchies* (SSPHH) [Met18][1]. This paper is organized by reviewing previous research in this area before we talk about our theoretical framework, implementation, and results.

2 PREVIOUS WORK

Spherical harmonics (SH or SPH) are often used to represent low-frequency light probes because they act as a low pass filter and have some useful convolution and summation properties. In computer graphics, they represent the angular distribution of light. They are the spherical analog to the Fourier series and are a set of orthogonal functions. The spherical harmonics equation [Mer98] is given by

$$Y_m^\ell(\theta,\varphi) = K_m^\ell \, P_m^\ell(\cos\theta)\, e^{im\varphi}\,,$$

where

$$K_m^\ell = (-1)^m \sqrt{\frac{2\ell+1}{4\pi}\frac{(l-m)!}{(l+m)!}}\,,$$

$P_\ell^m(\cos\theta)$ is the associated Legendre polynomials, and $e^{im\varphi}$ can be represented in real form by

$$|e^i m\varphi| = \begin{cases}\cos m\varphi & \text{if } 0 \le m \le \ell \\ \sin m\varphi & \text{if } -\ell \le m < 0\end{cases}.$$

The spherical harmonics are orthogonal polynomials which form an orthogonal set. This means that they satisfy the property that

$$\int_{\theta=0}^{\pi}\int_{\varphi=0}^{2\pi} Y_\ell^m(\theta,\varphi)\, Y_{\ell'}^{m'}(\theta,\varphi)\, \sin\theta \, d\varphi \, d\theta = \delta_{\ell\ell'}\,\delta_{mm'}\,,$$

[1] We abbreviated our algorithm SSPHH before realizing that SPH could be confused with Smoothed Particle Hydrodynamics, so we use SH for spherical harmonics and continue to use SSPHH for our algorithm.

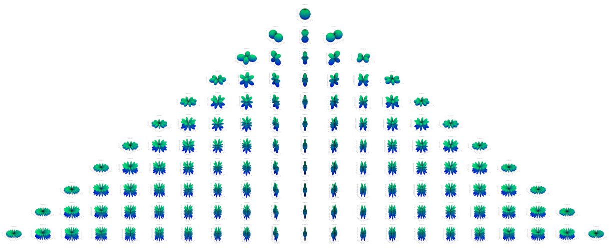


Figure 2: The degrees and bands of the spherical harmonics from $0 \le \ell \le 10$.

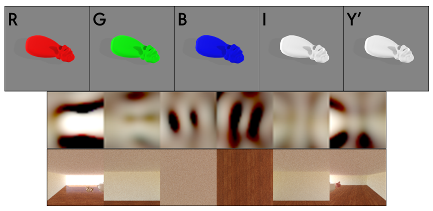


Figure 3: A SH representation of a light probe cube map.

where δ_{ij} is the Kronecker Delta

$$\delta_{ij} = \begin{cases}0 & \text{if } i \ne j, \\ 1 & \text{if } i = j.\end{cases}$$

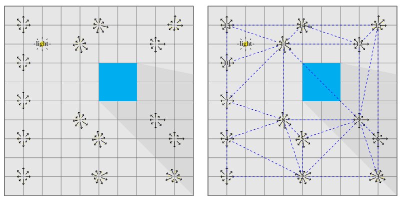
We may now approximate a spherical function $L(\theta,\varphi)$ by calculating coefficients a_ℓ^m for each degree ℓ and band m with the integral

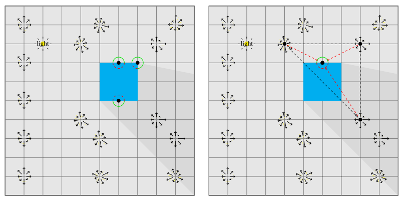
$$a_\ell^m = \int_{\theta=0}^{\pi}\int_{\varphi=0}^{2\pi} L(\theta,\varphi)\, Y_\ell^m(\theta,\varphi)\, \sin\theta\, d\varphi\, d\theta\,,$$

and we can reconstruct the original function by summing the spherical harmonics for $0 \le \ell \le N$ with the equation

$$L'(\theta,\varphi) = \sum_{\ell=0}^{N}\sum_{m=-\ell}^{\ell} a_\ell^m\, Y_\ell^m(\theta,\varphi).$$

We use the `orthopoly` package from Maxima [S⁺82] to calculate the equations for $0 \le \ell \le 10$ and optimized those equations for calculation in our mathematics library. The number of SH coefficients is $(\ell+1)^2$ so we limit the highest degree to 10 since which require 121 coefficients. Figure 2 shows degrees $0 \le \ell \le 10$ and Figure 3 shows an example of a light probe which has been converted to a SH using the sampling approach from Green [Gre03]. In this paper, we use the operator $\bigotimes$ to represent *convolution* and operator $\bigoplus$ to represent *summation* which are component wise multiplication or addition of vector components. These are common symbols used in digital signal processing diagrams. We want to avoid confusion with tensor products and set theory.


(a) Radiance probes placed in environment (left) and then Delauney triangulated for SH search (right).


(b) Visibility SH created for every vertex (left). The nearest light probes to each vertex are interpolated (right).

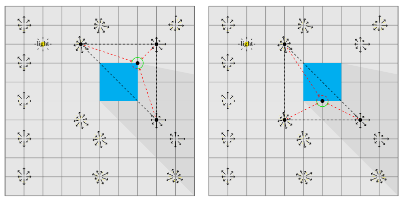

(c) Two examples show algorithm using spherical harmonics based on visibility.

Figure 4: Demonstration scene showing Irradiance Volumes and Precomputed Radiance Transfer (IR/PRT) algorithm.

The use of the terms *radiance probes*, *irradiance volumes*, *environment maps*, and *light probes* refer to the same basic idea. The simple explanation is that they are multispectral 360° photographs taken at a point in space. The most common representation of image elements are high dynamic range RGB pixels, but they can be any single or multispectral representations. Irradiance volumes may be used to describe the hemisphere or spherical angular distribution of light around a point on a surface. A surface point may be shaded by using interpolated surrounding light probes to estimate irradiance. Environment maps are used for simulating reflection or irradiance. Lastly, any of these light probe images may be projected into spherical harmonics.

2.1 Path Tracing

Path tracing [Kaj86] and ray tracing [Whi80] are general solutions to the problem. Newer classes of hardware graphics processing units such as the NVIDIA GTX 2080 Ti are capable of ray tracing a few rays per pixel and couple it with image denoising [MMBJ17] to reconstruct indirect illumination. As path tracing enjoys the most recent industry support, we employ it in our method to construct light probes using Corona Renderer [OK16]. These methods remain impractical for the foreseeable future for use with real-time graphics when dealing with low energy or low compute devices. Our method falls into the category of a hybrid approach which will be a popular approach for some time.

Kajiya's rendering equation [Kaj86]

$$\mathbf{L}_o(\mathbf{x} \to \omega_o) = \mathbf{L}_e(\mathbf{x} \to \omega_o) + \int_\Omega f_r(\omega_i, \omega_o)\, \mathbf{L}_i(\omega_i \to \mathbf{x})\, \langle \omega_i, \omega_o \rangle \, d\omega_i \, .$$

is often solved with Monte Carlo integration, but we may eliminate the integral and process specific paths of light with the Dirac-delta function

$$\delta(x) = \begin{cases} +\infty, & x = 0 \\ 0, & x \neq 0 \end{cases},$$

where $\int_{-\infty}^{\infty} \delta(x)\, dx = 1$ and the integral of $f(x)$ which is only non-zero at $x = 0$ may be directly calculated

$$f(0) = \int_{-\infty}^{\infty} f(x)\, \delta(x)\, dx = \int_{-\infty}^{\infty} f(x)\, dx \, .$$

2.2 Precomputed Radiance Transfer

Irradiance Volumes [GSHG98, RH01] and Precomputed Radiance Transfer [SKS02] (IR/PRT) modify the rendering equation by utilizing transfer functions. The first transfer function determines self-shadowing or visibility from any direction. The second transfer function determines how much light illuminates the surface. Mathematically, this is the following operation

$$T(\mathbf{x} \to \omega_o) = \frac{1}{\pi} \int_\Omega L_{SH,\omega_i} \otimes H_{SH,\omega_i} \, d\omega_i$$

where $T(\mathbf{x} \to \omega_o)$ is the exit radiance at the point $\mathbf{x}$, L_{SH,ω_i} is the incident radiation at point $\mathbf{x}$ and H_{SH,ω_i} is the cosine weighted kernel which maps visibility from angle ω_i to point $\mathbf{x}$. This particular example highlights diffuse reflection and more sophisticated examples exist for specular and refractive materials.

Figure 4 shows the process of precomputed radiance transfer using irradiance volumes. Irradiance volumes are sampled as a set of SH coefficients. Each vertex on the world geometry gets a *transfer vector* that measures the angular light that is received at infinity. The shape of the SH looks like a measurement of the visible hemisphere; hence, the transfer vector has values close to 0 for directions opposite to the normal and values close to 1 near the normal. The IV probes are placed manually or procedurally in such a way that a Delaunay triangulation can be created. Every visible vertex will be contained in a simplex of this triangulation, or extrapolation could be used otherwise. The closest IV probes are interpolated using barycentric coordinates to estimate the illumination for that vertex. Lower performing hardware often use second order spherical harmonics (9 floats per RGB channel) or order four or five spherical harmonics (25 or 36 floats per RGB channel). The irradiance volumes may be produced dynamically, but *baking* eliminates the real-time performance hit and enables the use of higher quality path traced light probes.

2.3 Virtual Point Lights

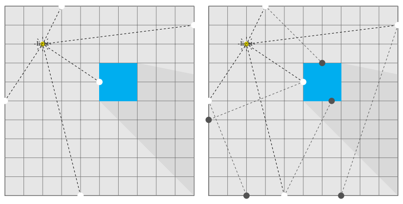
Instant radiosity—more generically *many light methods*—are used to simulate global illumination in a 3D scene by simulating a small set of "photons" and then lighting the visible scene with those photons. Ritschel et. al. [RDGK12] survey a wide class of global illumination algorithms of this sort. Typically, point lights or spherical lights [HKWB09] are used in these applications. Hot spots or fireflies can plague this class of algorithms because of discontinuities due to the nature of point lights especially if they are close to surfaces due to the $1/r^2$ inverse ratio, so clamping is used to control this problem.

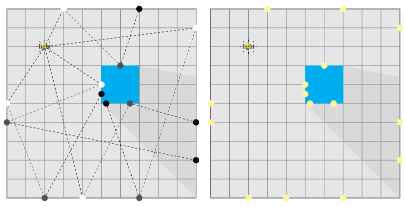
Figure 5 shows the process of virtual point lights. Virtual point light locations are determined by scattering photons into the 3D geometry. These point lights are then used to simulate the indirect illumination of the scene using shadow mapping. One key problem with VPLs are hot spots near the location of the VPLs due to the quadratic falloff of the point lights. To partially correct this defect, virtual spherical lights can be used. To reduce the number of VPLs in a scene, *light cuts* may replace close groups of lights with a single VPL.

2.4 Voxel Based Methods

Voxel based methods using a 3D Digital Differential Analyzer (3DDDA) [FTI86] are also potential solutions to the global illumination problem. Voxel cone tracing [CNS+11] is a recent method which projects lighting into a sparse octree and traces cones from surface geometry using final gathering. Other methods of 3DDDA use skip encoding to quickly march rays through the data structure [SK97]. Memory is a big problem with the voxel approach if you want high detail lighting. And light leakage may occur near geometry which does not conform to the voxel grid.

Figure 6 shows the process of voxel cone tracing. Rather than tracing rays, VCT proceeds by building a spatial subdivision data structure in which to propagate a light solution. The geometry is discretized into a voxel representation using a sparse octree. Afterwards the light is projected through the voxel grid to estimate irradiance in each cell of the octree which includes a filtering step to down-sample radiance. The camera pass then uses a normal direct illumination step and a secondary bounce step. The secondary bounces are traced through the octree using cone tracing which gathers indirect radiance.


(a) First two steps of VPL creation. Photons are scattered throughout the environment.


(b) Third iteration of VPL creation and a step showing that the lights are used as point lights near surfaces.

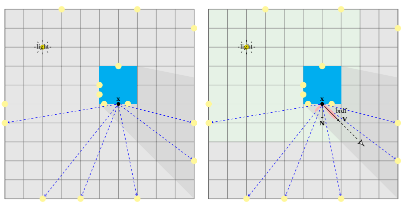
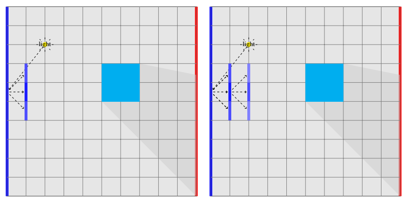

(c) Shadow maps determine visibility to **x** (left) and resulting light contribution (right).

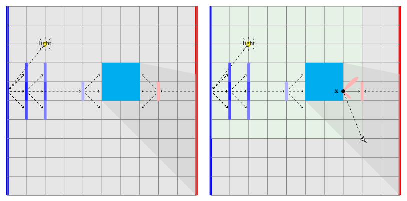
Figure 5: Demonstration scene showing Virtual Point Lights (VPL) algorithm.

3 THEORETICAL FRAMEWORK

The SSPHH method solves the problem of global illumination using *anisotropic lights*, which we call *spherical harmonic lights* (SPHL). We begin by determining the SPHL locations, or *nodes*. We need to determine the amount of light scattered around an SPHL that reaches neighboring SPHL nodes by simulating a light source and measuring the light observed by the neighbor. The graph relationship of each node to another is called a *hierarchy*. We render the scene at each of the SPHL nodes and project the images to spherical harmonics. We construct the resulting SPHL coefficients by summing its own light probe with an adjusted version of the neighbors. Figure 7 shows a visual progression of our algorithm. In this section, we will discuss this idea from several perspectives.



(a) The light source is traced through the voxel grid (left). The reflections are propagated throughout the voxel grid (right).



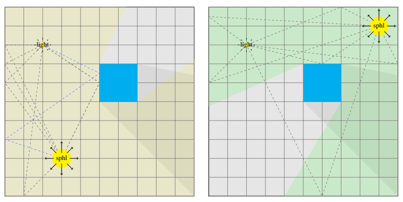
(b) Light transport between voxels eventually arrives at objects (left). During rendering, the voxels are used to shade surfaces (right).

Figure 6: Demonstration scene showing Voxel Cone Tracing (VCT) algorithm.

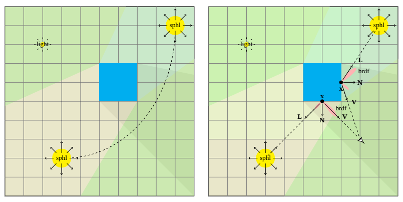
Let us return to the earlier house example. We see that if we place an SPHL in room A, place an SPHL in room B, and calculate the visibility SH coefficients from A to B, that room B will now have indirect illumination from room A without requiring a full path traced solution. This is because we can estimate how much light is transported from the SPHL in room A to the SPHL in room B. The remaining problem is time. If we slowly open the door to room A (or B), then we need to have a way of providing time-based interpolation of SH coefficients to accommodate the change in visibility between coefficients. We have not focused on this part of the problem in this paper.

If we consider our problem using regular expression notation [Hec90], we may use $L\{D|S\}E$ for direct illumination paths and $L\{D|S\}^+E$ for indirect illumination paths. The light path $L\{D|S\}^+E$ represents a chain of scattering events. We can divide this light path into three sections. The first is $L\{D|S\}^*\mathbf{S}_j$ which is the possible scattering event from a light source to the SPHL node $\mathbf{S}_j$. The second section is $\mathbf{S}_j\{D|S\}^+\mathbf{S}_i$ which is a chain of scattering events between SPHLs $\mathbf{S}_j$ and $\mathbf{S}_i$. Finally, we have the chain of scattering events $\mathbf{S}_i\{D|S\}\mathbf{x}$ which is the scattering event from SPHL $\mathbf{S}_i$ to the point $\mathbf{x}$ we are lighting.

In our method, we handle direct light paths $L\{D|S\}\mathbf{x}$ from various sources of lights including environment maps, directional lights, point lights, irradiance volumes, and SPHLs. The latter can be used to either simulate anisotropic time-varying light sources [MS12], or light probes to perform image based lighting. Later we call attention to this application, but for now we continue the discussion on light probes.



(a) Each SPHL is created using path tracer.



(b) Light transport is simulated between SPHLs before determining indirect illumination in final render.

Figure 7: Demo scene showing SSPHH algorithm.

We can use an SPHL $\mathbf{S}_i$ to estimate the irradiance at point $\mathbf{x}$ using the radiance function $\mathbf{L}_{\mathbf{S}_i}(\theta,\varphi)$ where (θ,φ) is the unit length direction $\mathbf{L}_{\mathbf{S}_i}$ from $\mathbf{x}$ to $\mathbf{S}_i$. The vector (θ,φ) is used as the input for the spherical harmonic $Y_m^\ell(\theta,\varphi)$. The limitation is that we need line of sight from $\mathbf{S}_i$ to $\mathbf{x}$ but we have a workaround for this which we discuss shortly. Since diffuse illumination of a rough surface is uniformly scattered by the BRDF $f_r(\omega_i,\omega_o)=\rho$ at point $\mathbf{x}$, the value

$$\mathbf{I}=\frac{\mathbf{L}_{\mathbf{S}_i}(\theta,\varphi)}{\mathbf{N}_\mathbf{x}\cdot\mathbf{L}_{\mathbf{S}_i}}$$

is the irradiance of $\mathbf{x}$ which accounts for the $\langle\omega_i,\omega_o\rangle$ term in the rendering equation integral.

We use a matrix $\mathbf{H}$ to store the SPHL coefficients of the SSPHH hierarchy and a matrix $\mathbf{P}$ to store the transfer coefficients $\mathbf{P}_{ij}$ between SPHLs. The SH $\mathbf{H}_{i\to i}$ is the SPHL generated by the GEN step. The SH $\mathbf{H}_{j\to i}, i\neq j$ is created by the VIZ step. The GEN light probe $\mathbf{S}_i$ is composed of the light paths $L\{D|S\}^*\mathbf{S}_i$. The visibility coefficients $\mathbf{H}_{j\to i}$ is composed of the light paths $\hat{L}_{1,j}\{D\}^*\mathbf{S}_i$ where $\hat{L}_{1,j}$ is a spherical light source of total intensity 1. We have chosen the name $\mathbf{P}_{ij}$ because we believe this represents the probability that light from direction (θ,φ) at SPHL $\mathbf{S}_j$ will appear from direction (θ,φ) at SPHL $\mathbf{S}_i$; hence, $\mathbf{P}_{ii}=1$. So by convolving the SH coefficients with the equation $\mathbf{H}_{j\to i}\otimes\mathbf{S}_j$ and summing we get the resulting SPHL

$$\mathbf{S}_i'=\mathbf{S}_i\oplus\sum_{j=0,j\neq i}^{N-1}\mathbf{H}_{j\to i}\otimes\mathbf{S}_j$$

Algorithm 1 Scalable Spherical Harmonics Hierarchies Algorithm

```
function INIT
    SPHLs ← NEWSH(0)
end function

function VIZ
    for $i \leftarrow 0, \dim \mathbf{S}$ do
        for $j \leftarrow 0, \dim \mathbf{S}, i \neq j$ do
            $\mathbf{H}_{i,j} \leftarrow$ RENDERVIZCUBE($S_i$, $S_j$)
            $\mathbf{P}_{i,j} \leftarrow$ COMPUTEVIZTRANSFER($\mathbf{H}_{i,j}$)
        end for
    end for
end function

function GEN
    for all $\mathbf{S}_i \in$ SPHLs do
        $\mathbf{S}_{i,\text{lightprobe}} \leftarrow$ RENDERCUBEMAP($\mathbf{S}_i$)
        $\mathbf{S}_i \leftarrow$ CUBEMAPTOSH($\mathbf{S}_{i,\text{lightprobe}}$)
        $\mathbf{H}_{i,i} \leftarrow \mathbf{S}_i$
        $\mathbf{P}_{i,i} \leftarrow 1$
    end for
end function

function HIER($N$)
    for $i \leftarrow 0, N'$ do
        Create sequence of pairs
        Pairs are indexes and visibilities
        $Q = \{(q,p) \in \{0,1,\ldots,N-1\} \times \mathbf{P}_i\}$
        Sort the previous sequence $Q$ by visibility
        $Q' = \{(q_j, p_j) \in Q | p_j > p_{j+1}, 0 \leq j < N\}$
        $\mathbf{S}'_{i,\text{self}} \leftarrow \mathbf{S}_i$
        Accumulate top $N'$ spherical harmonics
        $N' \leftarrow \min(N, \dim \mathbf{S})$
        $\mathbf{S}'_{i,\text{neighbor}} \leftarrow \sum_{j=0,j\neq i}^{N'} \mathbf{H}_{i,q_j} \otimes \mathbf{S}_{q_j}$
        $\mathbf{S}'_i \leftarrow \mathbf{S}'_{i,\text{self}} \oplus \mathbf{S}'_{i,\text{neighbor}}$
    end for
end function
```

where N is the number of SPHLs. This simulation of light transport from one SPHL to another is called the HIER step. We use the term hierarchy because we may sort the SPHLs by $\mathbf{P}_{ij}$ which rank the SPHLs by transport probability.

Now, this may not be not a scalable approach because we need an $N \times N$ sized matrix. We instead allow for a sparse matrix approach where the connections can be optimized to reduce memory and computational complexity. This is accomplished with a hash map instead of an array if the connection graph is not a complete graph. We create a list $\mathbf{Q}$ which contains pairs (q_i, p_i) of SH coefficients q_i and the transfer coefficients p_i which can be sorted based on p_i. We choose N' neighboring SPHLs instead and use p_i as a heuristic. If p_i is below

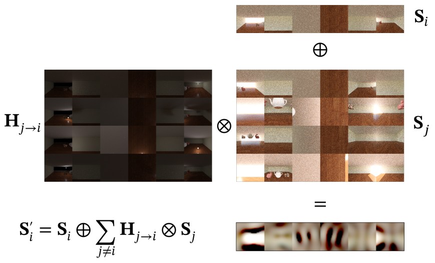

$\mathbf{S}_i$
$\oplus$
$\mathbf{H}_{j\to i}$
$\otimes$
$\mathbf{S}_j$
=
$\mathbf{S}'_i = \mathbf{S}_i \oplus \sum_{j\neq i} \mathbf{H}_{j\to i} \otimes \mathbf{S}_j$

Figure 8: Generating an SPHL $\mathbf{S}'_i$ with GEN and neighbor VIZ probes.

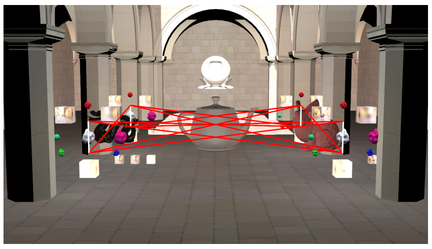


Figure 9: Image shows hierarchies and SH previews.

a certain threshold, or we choose N' to be a specific number, then our method is *scalable*. So if $p_i = 1$, as all GEN nodes are, then those are included in the sum, and if $p_i > 0$, as all enabled VIZ nodes are, then our resulting SPHL is

$$\mathbf{S}'_i = \mathbf{S}'_i \oplus \sum_{j=0,j\neq i}^{N'-1} \mathbf{H}_{q_j} \otimes \mathbf{S}_j \ .$$

Figure 8 shows a visual calculation of $\mathbf{S}'_i$ and Figure 9 shows an example set of hierarchies. Finally Algorithm 1 shows the complete four step process of the SSPHH algorithm.

4 IMPLEMENTATION DETAILS

Our research graphics engine *Fluxions* was designed for this algorithm and to work on a number of other graphics problems. There are several components worthy of mention. Our *Unicornfish* sub-library is an implementation of the majordomo protocol [Hin14] which is used to coordinate a number of worker nodes using a reliable request-reply pattern to help offload rendering tasks to other computing nodes in a local area network. We use it to send SH coefficients back to the main program using the JSON schema discussed in [MS19]. JSON schemata can be very helpful in client-server applications by adding validation to data interchange. JSON provides a blend of convenience for editing and readability. Figure 10 shows the process of converting a cube map with Corona to a SH representation which gets sent in JSON format for the renderer to use.

Processing HDR light probes require attention to data layout especially with cube maps. In practice, there is no consistency with how this data will be stored. The

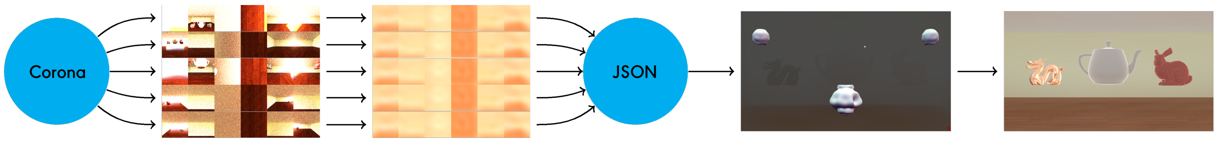


Figure 10: We outsource the light probe render to Corona, project to SH, and transmit JSON over a network.

most useful way to use a single rectangular image—rather than six individual images—is to store them in the order $(+X, -X, +Y, -Y, +Z, -Z)$ which is expected with OpenGL and Direct3D. We will mention that common layouts include the 6×1 horizontal strip, 1×6 vertical strip, 3×4 horizontal cross, or 4×3 vertical cross. The strip approaches have the benefit of not wasting image space.

Once we decided the best data formats to use in our SSPHH algorithm, we worked on a data collection plan to measure quality. There is no general consensus about how to measure image quality. No two renderers will be able to produce identical images in today's market. This is primarily due to the fact that every renderer is effectively a combination of several algorithms chosen for their application to a particular problem set. Even non-biased path tracers can generate different results if the authors chose to implement different BRDFs in their algorithms. What is best at this point is to use sound methodology in measuring image quality.

We chose to use Corona Renderer due to its usage of physically based rendering algorithms. The renderer needed to generate light probes and produce high quality images quickly—if any physically based path tracer can be described as *quick!* We also use Corona Renderer to generate ground truth images. We made some non-obvious choices about features to include in our rendered images. For example, we do not use bump maps in the reference images because not every renderer interprets bump maps equivalently. We believe it is better to use scene geometry which use the least amount of interpretable details as possible. So we limit our material exports to Lambertian diffuse surfaces, an optional specular roughness parameter, and texture maps. The reason for allowing specular roughness to be optional is to compare what highly varying BRDFs might do to an indirect illumination process.

Our testing program allows us to generate an identical ground truth image frame which we calculate *absolute error* to determine image differences. Because absolute error (or even RMSE) can have problems near polygonal edge boundaries, we also produce a low resolution image where we sum all the pixel data in smaller portions of the image. We do not take the average because we want to compare absolute differences. This approach enables us to visualize the total energy difference from reference and test images which can be see in Figure 13. This *pixelated* approach is used to help us understand where we get the most deviation. However, unlike normal pixelation, we sum up the energies in that area of the image rather than take the mean. We do not include AE numbers in our paper, but we have used them to focus our attention to why certain image areas have larger energy differences and we discuss this later in the analysis section.

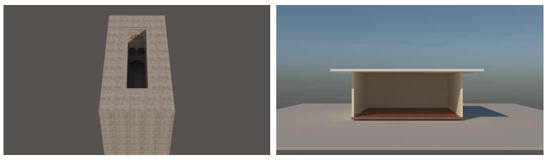


Figure 11: The two test scenes environments showing the requirement for indirect illumination.

The implementation of spherical harmonic lights is similar to point lights and point light shadow maps. A shadow map is dynamically generated for each SPHL node. The shadow map is used to determine whether to use the GI contribution. Since the SPHL stores the irradiance, only one SPHL needs to be used. If no SPHLs are available, then the closest one can be used to estimate the irradiance by assuming that the light probe is located around $\mathbf{x}$. In other words, instead of using the SPHL as a source of light, we use it as an irradiance volume instead. This is the case for surfaces where shadow mapping would cause significant amounts of occlusion. Please note that if you are using PRT or irradiance volumes, then you would want to interpolate the irradiance volumes that surround the vertex or shading fragment. The low frequency nature of spherical harmonics makes this a minor concern, and we save a deep analysis on defects arising from this usage for future work.

5 RESULTS AND ANALYSIS

Figure 1 shows two scenes with different objects. The first scene is designed to test color bleeding and the second scene is designed to test complex objects. Figure 13 show the comparison of the Corona renderer to our method. We also compare a ground truth image with a maximum ray depth of 3 since that represents second and third bounces of light. The reader will notice that there is a loss of energy (darkening) from a max ray depth of 25 in the ground truth images.

Table 1 shows the times needed to perform the Reference (REF), SPHL generation (GEN), SPHL visibility (VIZ), and hierarchical SPHL (HIER) steps. We compare both the time it takes to create reference quality

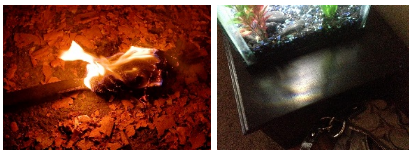


Figure 12: An SPHL could be used to represent complex emission profiles such as fire or caustics.

light probes—max ray depth 25 and pass limit 25—with normal quality light probes—max ray depth 3 and pass limit 1. For a set of four to six SPHLs with a resolution of $64 \times 64 \times 6$, less than one minute is required. This is generated on the same CPU, but in a different thread than the rendering thread. The minute time frame is due to the CPU based Corona Renderer which must load the geometry and materials and render an image. We are addressing this shortcoming by working on an internal GPU based path tracer to avoid the extra cost which we believe this can be done at 60Hz using 12 samples per pixel and a GPU based ray tracing framework.

We used our Majordomo protocol sub-library Unicornfish to launch Corona. This process converts the scene graph into the XML data format which Corona can read. Because spherical harmonics are a type of data compression, we are able to render the light probes on a separate rendering node and send them back efficiently. The size of those packets depend on the maximum degree of the SH to encode. For example, a tenth degree SH with RGB components is $3 \times (10+1)^2 = 363$. Formatting this in JSON requires under 4KB of formatted text with 5 digit accuracy. A binary form would require significantly less space. However, SPHLs do not need to be updated unless there is a significant change in the lighting context such as a door opening or closing.

We believe these would be very useful in high performance games. A secondary use would be to have a game client generate an SPHL for their own players and share that with a game server to distribute to neighboring players. SPHLs do not need to only represent global illumination, they can also represent anisotropic lights. A space ship or hot rodded automobile with custom lights could be constructed as an SPHL by sampling the emission profile of the player where

$$\mathbf{L}_o(\mathbf{x} \rightarrow \omega_o) = \mathbf{L}_e(\mathbf{x} \rightarrow \omega_o)\,.$$

Sending these to nearby players would result in object to object light interactions. Using SPHLs to represent dynamic local illumination from static objects such as torches, magical effects, or even fish tanks would result in more realistic experiences. Figure 12 shows complex emission profiles from fire and caustics.

Path Tracer Configuration

The maximum ray depth (MRD), samples per pixel, and pass count (PC) are three of the biggest factors in quality and render time required for ray tracers. Maximum ray depth deals with how many bounces of light are allowed. Using the regular expression notation, $L\{D|S\}E$ is one bounce of light, $L\{D|S\}^2E$ is two bounces of light, and so on. To prevent exponential growth of calculation as in Whitted ray tracing (e.g. shadow rays, refraction rays, reflection rays), only one path is chosen for a single sample. This is determined using Russian roulette and importance sampling. A path tracer which utilizes this technique should see maximum ray depth scale linearly with each additional bounce. The samples are scaled by the BRDF and averaged to create the final pixel. Many samples per pixel must be used to obtain a low-noise image. Pass count allows the image to be constructed progressively. That is, rather than wait for the appropriate number of samples to be recorded before producing an image, we calculate a certain number of samples per pixel per pass and update the image when each pass is finished.

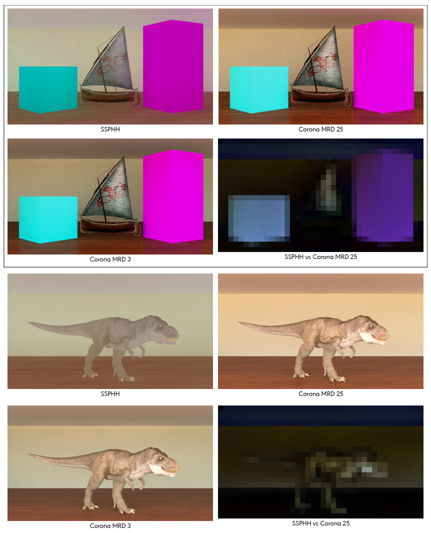


Figure 13: Additional test scenes showing color bleeding (top) and a complex model (bottom). Absolute error from reference images are also shown.

The second major factor that affects light probe render time is the quantity of geometry and light preprocessing before the image can be rendered. Intel's Embree library is a popular choice for accelerating ray-triangle intersections. And this is mostly a fixed cost, so there is not much that can be improved aside from choosing to precompute some of the lighting with a cache. Readers familiar with techniques like photon mapping [Jen96] will relate to the idea of scattering photons in a scene and reusing them to accelerate global illumination. Corona Renderer uses a similar technique called a UHD Cache which maintains similar quality with unbiased path tracing especially for indoor scenes. There is generally no benefit to turning the UHD Cache off and made no significant impact in the generation of

Table 1: Comparison of reference, GEN, VIZ, and HIER times for each scene.

Scene	*mrd*	*pl*	REF time (s)	GEN time (s)	VIZ time (s)	HIER time (s)
Gallery 1	25	25	164.46	131.54	N/A	0.62173
	3	1	15.03	48.40	197.46	0.91512
Gallery 2	25	25	176.10	141.39	N/A	0.62173
	3	1	18.35	53.08	230.81	0.61695
Gallery 3	25	25	161.43	141.76	N/A	0.60576
	3	1	16.37	56.25	234.89	0.69566
Sponza	25	25	220.20	151.12	N/A	0.47393
	3	1	21.88	55.60	178.34	0.48255

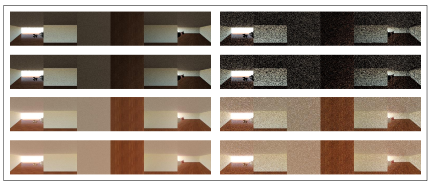


Figure 14: Varying maximum ray depth and pass limits for light probes. Left column uses a high pass count and the maximum ray depths from top to bottom are 0, 1, 3, and 25.

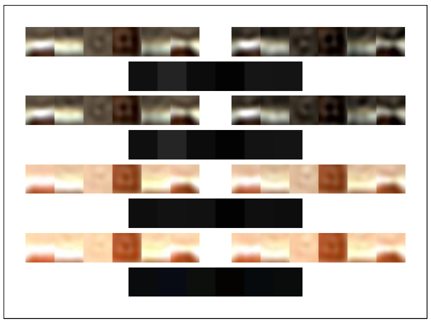


Figure 15: Varying maximum ray depth and pass limits for spherical harmonic lights. Left column uses a high pass count and the maximum ray depths from top to bottom are 0, 1, 3, and 25. The center column shows the difference between the left and right columns.

our light probe images. Instead, real-time performance was hindered by the resulting increased rendering time.

Adjusting the maximum ray depth and pass limit affects quality of the light probe images to varying degrees. Figure 14 shows the difference between path traced images using various MRD and PL. Figure 15 shows the effect of choosing different maximum ray depths and pass limits for creating SH. By comparing successive images, we determined that a practical depth of 3 provides enough light information while remaining sufficiently close in quality as a depth of 25.

Image Energy Differences

Matching images between two different renderers is challenging. We noted earlier that we use absolute error to mark differences between our images and the ground truth images as shown in Figure 13. Those deviations are primarily due to tone mapping, BRDF differences, and indirect illumination in our rendering system.

Indirect illumination in real-time renderers is often enhanced with ambient occlusion which estimates the accessibility of a surface point to ambient light. We do not simulate ambient occlusion in our images because this was out of scope and we wanted to focus on how well our algorithm worked on its own. So the reader will notice that the darkening of edges is not well handled in our approach. We expect that adding ambient occlusion will benefit a production image.

Secondly, we note that our reflection models do not match exactly with Corona Renderer. We utilize a bipolar version of normalized Blinn-Phong and GGX for specular reflection and a bipolar version of Oren-Nayar and Disney's BRDF [BS12]. We will not claim this is novel, but our bipolar approach means that the roughness parameter α maps from -1–1, where model A is chosen if $\alpha < 0$ and model B is chosen if $\alpha > 0$. This is different than Corona's BRDF which uses SGGX and Lambertian models. Our exporter simplifies the materials to a lowest common denominator of diffuse color, specular color, index of refraction, and roughness. We only support solid objects and thus do not handle transparency effects or subsurface scattering (e.g. the cloth on the the sailboat).

Lastly, tone mapping makes a huge difference in rendering and there are several approaches to make high dynamic range images fit inside the sRGB color space. We utilized gamma scaling coupled with an exposure control. Corona uses a similar adjustable gamma and exposure control along with filmic highlight compression. We chose to limit our tone mapping to something most likely to be used in real-time graphics, though we would like to experiment with more elaborate approaches. We did not modify the original tone mapping of the texture maps we loaded, while Corona does. This explains the darkening of the textures in the reference images versus ours. However, we perform tone mapping to render an sRGB image at the final stage. In the future, we are working on changing this approach to allow custom tone mapping properties on imported textures. Regardless, we always performed lighting calculations in linear space to match current best practice in physically based rendering.

6 CONCLUSION AND FUTURE WORK

In this paper, we presented the Scalable Spherical Harmonics Hierarchies method and how it can be used to approximate global illumination in real-time applications. We use spherical harmonic light probes to determine surface irradiance which allow us to simulate low frequency indirect illumination using local anisotropic point lights called SPHLs. We estimate the global illumination by determining the light transport from one SPHL to another. Hence, we can solve the problem of indirect light transport for solid objects in real-time by precomputing the visibility between SPHLs to determine how much light is transported.

In the future, we see this as a great application for client-server applications where global illumination information is dynamically updated for players in a video game context. On-line streaming platforms could utilize this concept to share global illumination between players in resource constrained environments. Lastly, we also want to investigate the use of our method for participating media and time-varying effects.

7 REFERENCES

[BS12] Brent Burley and Walt Disney Animation Studios. Physically-based shading at disney. *ACM SIGGRAPH*, 2012:1–7, 2012.

[CNS+11] Cyril Crassin, Fabrice Neyret, Miguel Sainz, Simon Green, and Elmar Eisemann. Interactive indirect illumination using voxel cone tracing. In *Computer Graphics Forum (Proc. of Pacific Graphics 2011)*, 2011.

[FTI86] A. Fujimoto, T. Tanaka, and K. Iwata. Arts: Accelerated ray-tracing system. *IEEE Computer Graphics and Applications*, 6(4):16–26, April 1986.

[Gre03] Robin Green. Spherical harmonic lighting: The gritty details. Technical report, Sony Computer Entertainment America, 2003.

[GSHG98] Gene Greger, Peter Shirley, Philip M. Hubbard, and Donald P. Greenberg. The irradiance volume. *IEEE Computer Graphics and Applications*, 18:32–43, 1998.

[Hec90] Paul S Heckbert. Adaptive radiosity textures for bidirectional ray tracing. *SIGGRAPH Comput. Graph.*, 24(4):145–154, 1990.

[Hin14] Pieter Hintjens. *ØMQ - The Guide*. iMatix Corporation, 2014.

[HKWB09] Miloš Hašan, Křivánek, Bruce Walter, and Kavita Bala. Virtual spherical lights for many-light rendering of glossy scenes. *ACM Trans. Graph.*, 28(5):143:1–143:6, 2009.

[Jen96] Henrik Wann Jensen. Global illumination using photon maps. In *Rendering Techniques 96*, pages 21–30. Springer, 1996.

[Kaj86] James T. Kajiya. The rendering equation. *SIGGRAPH Comput. Graph.*, 20(4):143–150, August 1986.

[Mer98] Eugen Merzbacher. *Quantum Mechanics*. Wiley, 3 edition, 1998.

[Met18] Jonathan B. Metzgar. *Realtime Physically Plausible Global Illumination Using Scalable Spherical Harmonics Hierarchies*. PhD thesis, University of Colorado Colorado Springs, 2018.

[MMBJ17] Michael Mara, Morgan McGuire, Benedikt Bitterli, and Wojciech Jarosz. An efficient denoising algorithm for global illumination. In *Proceedings of High Performance Graphics*, HPG '17, pages 3:1–3:7. ACM, 2017.

[MS12] Jonathan Brian Metzgar and Sudhanshu Kumar Semwal. Approximating the fire flicker effect using local dynamic radiance maps. In *International Conference in Central Europe on Computer Graphics and Visualization*. Václav Skala-UNION Agency, Pilsen, CZ, 2012.

[MS19] Jonathan B. Metzgar and Sudhanshu K. Semwal. Applying zeromq to realtime global illumination rendering. In *Symposium on Interactive 3D Graphics and Games (i3D) 2019*, 2019.

[OK16] Jaroslav Kivánek Ondej Karlík, Adam Hotový. Corona renderer. `https://corona-renderer.com/`, 2016.

[RDGK12] Tobias Ritschel, Carsten Dachsbacher, Thorsten Grosch, and Jan Kautz. The state of the art in interactive global illumination. *Comput. Graph. Forum*, 31(1):160–188, 2012.

[RH01] Ravi Ramamoorthi and Pat Hanrahan. An efficient representation for irradiance environment maps. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 497–500, New York, NY, USA, 2001. ACM.

[S+82] William Schelter et al. Maxima. `http://maxima.sourceforge.net/`, 1982.

[SK97] Sudhanshu K. Semwal and Hakan Kvarnstrom. Dual extent and directional safe zone techniques for ray tracing. In *Proceedings of Graphics Interface Conference, Kelowna*, BC, Canada, 1997.

[SKS02] Peter-Pike Sloan, Jan Kautz, and John Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph.*, 21(3):527–536, July 2002.

[Whi80] Turner Whitted. An improved illumination model for shaded display. *Commun. ACM*, 23(6):343349, June 1980.