Rendering godrays with Monte Carlo path tracing on the GPU

NOAH PITTS, University of California, Berkeley
ISAK KARLSSON, University of California, Berkeley

ENTER ABSTRACT HERE

CCS Concepts: •**Computing methodologies → Ray tracing;** *Real-time simulation; Interactive simulation; Massively parallel and high-performance simulations;*

Additional Key Words and Phrases: rendering, ray-tracing, path-tracing, volume rendering, participating media

ACM Reference format:
Noah Pitts and Isak Karlsson. 2017. Rendering godrays with Monte Carlo path tracing on the GPU. 1, 1, Article 1 (May 2017), 4 pages.
DOI: 0000001.0000001

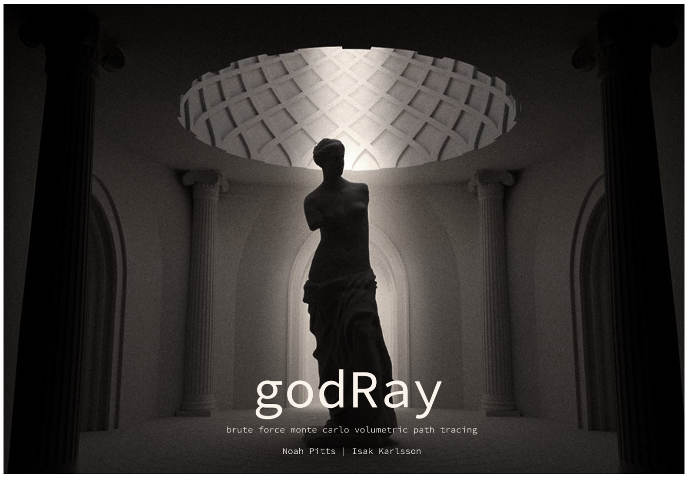
godRay
brute force monte carlo volumetric path tracing
Noah Pitts | Isak Karlsson

Fig. 1. god ray in rendered scene

1 INTRODUCTION

The god ray is a visually appealing and intense lighting effect that can be seen in the rendering of interior architectural scenes with limited direct illumination through a portal. The paper proposes a technique for using the the GPU to efficiently and progressively render god rays in these types of scenes using brute force Monte Carlo integration of a volumetric rendering equation. A CUDA rendering application called *godRay* was produced with the ability to control sun and sky lighting and participating media parameters in a interior temple scene. The discussion is focused on the caparison of these parameters with respect to the physically based rendering of god rays.

1.1 Crepuscular Rays

God rays, also known as Crepuscular Rays in atmospheric optics, are light beams that appear to radiate from an intense light source in the scene and are particularly apparent when a considerable portion of the scene is occluded from the this light source. In exterior scenes god rays may be seen as beams of light streaming through dense clouds or physical openings in geometry such as between buildings or trees. Interior scenes can also exhibit god rays such as in the image of the pantheon, a classical roman architecture that is daylit by a single occulus at the top of the dome. In these cases, the viewer of the image is drawn to the phenomenological qualities of the space and the connection to this exterior light source. In the physical environment, god rays are usually produced by the intensity sun of the sun as a single light source in contrast to the much less intense sky dome or skylight. The rays appear as if they converge to a point, but in the case of sunlight are near parallel. The tapering or slight conical appearance of these light rays is an effect often caused by perspective convergence or in the case of computer rendering linear perspective.

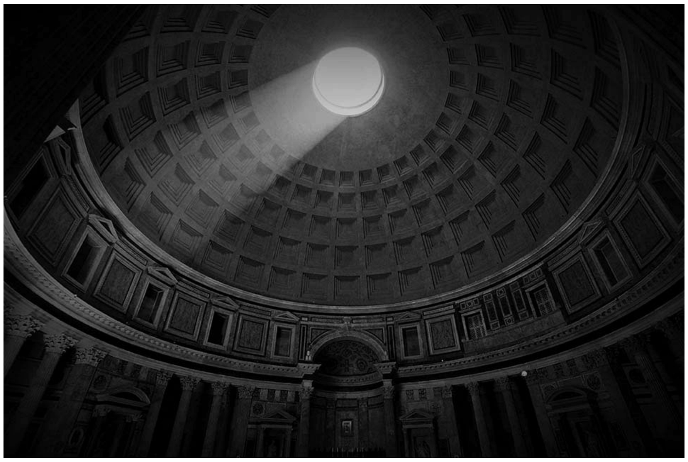

Fig. 2. Occulus of the Pantheon with Crepuscular Ray

1.2 Volumetric Rendering

God rays are made visible in the scene through the interaction of light with participating media or atmospheric particles such as smoke, fog, or haze. It is often the case that in architectural renderings of an interior scene, god rays appear in a homogeneous media as the interacting particles in the volume of the space are well mixed. However interactions with media of heterogeneous density are possible as well. In this paper we will be considering only the homogeneous case due to the efficiency it affords when sampling the media.

1.3 GPU Rendering

For interior architectural scenes with significant indirect lighting and global illumination path tracing techniques can provided excellent results. In order to maintain an unbiased solution we have used a brute force Monte Carlo integrator to numerically solve the rendering equations. The brute force approach to the rendering equation requires a significantly greater amount of pixel samples to resolve the scene, especially in areas only lit by indirect illumination. The increase in sampling can greatly increase the rendering times of the scene. In scenes with participating media, the sampling and rendering times can increase even further. To account for this, we have developed the rendering application on the GPU using Nvidia CUDA. Both the direct illumination and indirect illumination integrators are developed in CUDA kernels to take advantage of the massive core count on the GPU. The test images for this paper were rendered with two Nvidia Titan X GPUs with a combined count of 7680 CUDA cores. The scene bounding volume hierarchy (BVH) was built using the Nvidia OptiX SDK.

2 MATHEMATICAL MODEL

The rendering equation for *godRay* is formally based on the rendering equation proposed by Kajiya [2] with adaptations for volumetric rendering in homogeneous participating media discussed in chapter 4 of Wojciech Jarosz thesis [1]. This project maintains an unbiased approach using Monte Carlo techniques as presented by Veach [5].

2.1 Participating media

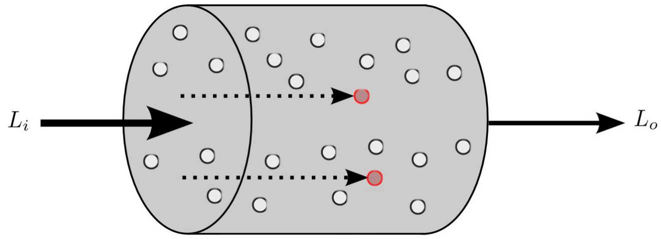

Fig. 3. Absorbtion

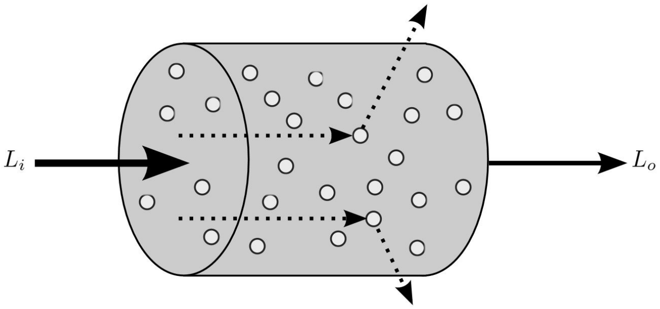

Fig. 4. Out-scattering

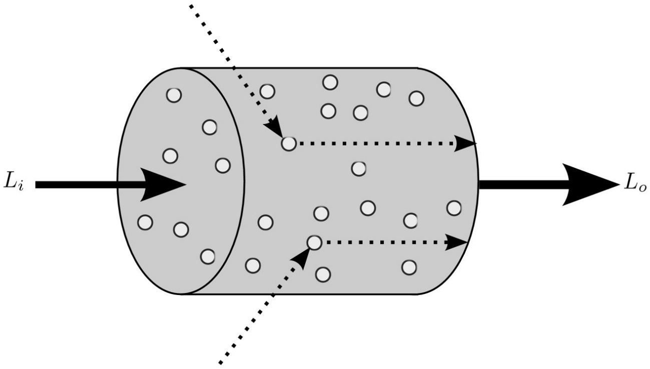

Fig. 5. In-scattering

For rendering participating media in the scene we consider three differential effects of light transport along the length of the ray path segments: absorption (σ_a) , out-scattering (σ_o) , and in-scattering (σ_s) . However absorption and out-scattering will be combined to form a single extinction coefficient (σ_t).

$$\sigma_t = \sigma_a + \sigma_o \tag{1}$$

where the differential equation of extinction can be described by

$$dL_o(x, \omega) = -\sigma_t(x, \omega)L_i(x, \omega)dt \tag{2}$$

The fraction of radiance that is transmitted between two points along the ray path, transmittance($T_r(x_1, x_2)$) is therefore the solution to the differential equation of extinction.

$$T_r(x_1, x_2) = e^{-\int_0^d \sigma_t(x_1+t\omega, \omega)dt} \tag{3}$$

However, since we are considering our *godRay* scenes to contain only homogeneous media where the extinction coefficient is constant, we can significantly simplify the transmittance equation to beer's law where d is representing the path length between points.

$$T_r(x_1, x_2) = e^{-\sigma_t d} \tag{4}$$

The model for in-scattering can be described by the differential equation where incoming radiation from all solid angles ω' is scattered in the direction of the ray ω.

$$dL_o(x, \omega) = \sigma_s(x, \omega) \int_{S_2} \rho_p(x, \omega', \omega) Li(x, \omega')d\omega' dt \tag{5}$$

2.2 Phase Functions and BSDF

The phase function, similar to the BSDF function for surface interactions, ρ_p represents the fraction of light that is scattered into the ray direction ω. For *godRay* we are using the Henyey-Greenstien phase function [4] which allows for a single asymmetry value g between the range [-1.0, 1.0] to parameterize the scattering angle. When $g = -1.0$ there is perfect back scattering while when $g = 1.0$ there is perfect forward scattering. When $g = 0.0$ scattering is isotropic with uniform probability of scattering in every direction.

$$\rho_p(x, \omega', \omega) = \frac{1}{4\pi} \frac{1 - g^2}{(1 + g^2 - 2g\cos\theta)^{3/2}} \tag{6}$$

In *godRay*, in order to focus on the volumetric effects of the god ray we are only considering a basic Lambertian bidirectional reflectance distribution function or BRDF ‘reference‘ representing an ideal matte surface. While the Lambertian model is not physically based it provides a significant simplification of the model. For a more physically based diffuse surface, the micro-facet model ‘reference‘ could be considered. In our case however ρ_r represents the BRDF while $sigma_r$ represents the diffuse reflectance coefficient of the surface interaction.

$$\rho_r(x, \omega, \omega') = \frac{\sigma_r}{\pi} \tag{7}$$

2.3 Volume Rendering Equation

The volume rendering equation for radiance $L(x, \omega)$ must consider the different effects of either scattering at a surface or within the media. Therefore along a given path segment we will sample a scattering event along the path and test whether or not the media intersection occurred before a surface intersection. If the scattering event occurred in the media, radiance, $L_{atmos}(x, \omega)$, will be calculated based off of the participating media equations and parameters set for the globally defined homogeneous atmosphere. If the scattering event occurs at a surface, the radiance, $L_{surf}(x, \omega)$,for the path will be calculated as per the Kajiya rendering equation ‘reference‘. The radiance for the path is then scaled according to the transmittance function for the atmospheric media along the path length. The following volume rendering equation adapts the Kajiya model to account for scattering events that occur within the globally defined atmospheric media.

$$L(x, \omega) = \int_0^\infty \left[t_{atmos} < t_{surf} \quad ? \quad L_{atmos}(x + s\omega, -\omega) \quad : \quad L_{surf}(x + d\omega, -\omega)\right] \sigma_t e^{-\sigma_t s} \tag{8}$$

$$L_{atmos}(x, \omega) = \int_{S_2} \frac{\sigma_s}{\sigma_t} \rho_p(x, \omega, \omega') L(x, \omega') d\omega' \tag{9}$$

$$L_{surf}(x, \omega) = L_e(x, \omega) + \int_S \rho_r(x, \omega, \omega') L(x, \omega') |\cos\theta'| d\omega' \tag{10}$$

3 SCENE

A sample scene was constructed for *godRay* where the interior of a faux classical temple interior receives illumination from a procedurally generated exterior sun and sky light model. The only light on the interior passes through a single occulus at the top of a dome centered above a sculpture.

3.1 Sun and Sky Lighting

The lighting for the sample temple scene uses two light sources procedurally generated from a Preetham Sun and Sky model [3]. While the Preetham model is not physically accurate, it provides a computationally cheap method of calculating an infinite skylight that can be parameterized through a sun azimuth and sun altitude. The sunlight was modeled as an infinite directional light with a small radius based on the suns solid angle.

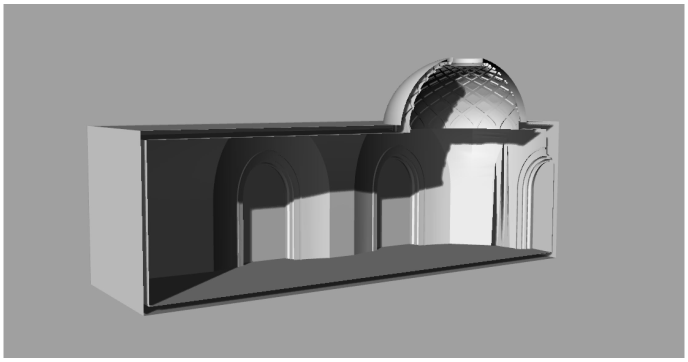

Fig. 6. geometry of temple scene

3.2 Scene Geometry

The temple geometry was modeled in NURBS and exported into an OBJ mesh format for import into the godRay rendering engine. A section of the temple scene geometry is diagrammed in ‘figure‘. Triangle intersection test are used in the BVH ray intersection accelerator. A *Venus de milo* mesh file was used as the central piece under the dome and occulus to allow for the god ray to cast onto. The mesh file is based on a 3d scan of the original sculpture ‘reference‘.

4 MONTE CARLO PATH TRACING

The rendering equation for the *godRay* engine was solved through brute force Monte Carlo integration. Pixel samples were distributed across the CUDA cores and collected into an accumulation buffer. *godRay* uses progressive rendering without tiling, where pixel samples will continue to accumulate when running the application in OpenGL with the GUI, or can be specified though arguments if run without the GUI on the command line. ‘Figure‘ describes the general path tracing procedure. Direct illumination from the sun was sampled at all scatter events while the skylight contribution was included only in the indirect illumination path when an intersection miss occurred with the geometry.

4.1 Sampling

Sampling of the of the global illumination path is done through Russian roulette termination of the recursive ray depth. The technique used follows Veach’s description closely ‘reference‘.

Sampling of the scatter events in the atmospheric media first require an intersection test of the ray with the scene geometry where t_{surf} represents the parametric value of the intersection point along the ray. t_{atmos} represent the sampled atmospheric scattering point along the same ray. t_{atmos} derived from the ‘equation‘ describing the extinction in the media where ξ is a uniformly distributed random variable.

$$t_{atmos} = \frac{ln(1 - \xi)}{\sigma_t} \tag{11}$$

with a PDF

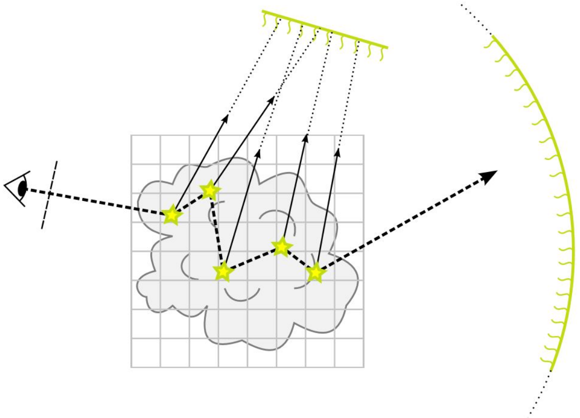

Fig. 7. direct and indirect illumination paths

$$p_t(t) = \sigma_t e^{-\sigma_t t} \tag{12}$$

Sampling of the solid angle for surface scatter events uses cosine weighted hemisphere sampling so that more sampling occurs in areas of the hemisphere where more total irrandiance is likely to come from. The PDF is: $p(x, \omega, \omega') = \cos\theta/\pi$. Sampling of the solid angle for non terminated indirect paths for atmospheric scatter events follows from solving the Henyey-Greenstien phase function for $\cos\theta$ where given some uniformly distributed random variable ξ

$$\cos\theta = \frac{1}{2g}\left(1 + g^2 - \left(\frac{1-g^2}{1-g+2g\xi}\right)^2\right) \tag{13}$$

Given that the phase function is normalized the PDF is the evaluation of the function $\rho_p(x, \omega', \omega)$.

When considering the direct illumination sampling of the sunlight is done through uniform disk sampling based off of the radius of the modeled sun.

4.2 Pathtracing in CUDA

Due to limits on the stack depth in CUDA, implementing the rendering equation for *godRay* in requires that the path tracing be done iteratively rather than recursively. This requires that the throughput of atmospheric attenuation, 'tr' and global illumination path weight, 'beta', be separately tracked through each iteration of the main global illumination rendering loop. The algorithm described on the website written in generalized pseudo code, however you can refer to the CUDA source code for specific implementation syntax and details of all functions.

5 RESULTS

Images were generated for all of the parameters concerning the rendering of the god ray in the scene. The following image sequences describe the variation of parameter values and the effect on the scene image.

NOTE TO GSI: Please see the comparison images on the website version of the write up, I am having trouble formatting them in latex, thanks

5.1 Extinction Coefficient

The extinction coefficient increases the probability of scattering, as well as increases the attenuation rate of rays as they pass through a media. In the figures below we can observe the scene becoming darker as the extinction coefficient increases.

5.2 Scatter coefficient

The Scatter coefficient affects the amount scattered into a differential segment of media. This is different than the out-scatter coefficient which is bundled together with the extinction coefficient. The in-scatter coefficient, as seen below, changes how much light is bouncing around within the scene. A larger scattering coefficient will produce a brighter image, as more light enters the dark room where the statue is.

5.3 Phase Function

The scattering phase function coefficient is the coefficient g in Henyey-Greensteein's phase function. Observe the effect of g as it increases from 0.1 to 0.99 and -0.1 to -0.99. Specifically, when the coefficient is close to 0 we observe isotropic scattering. A larger negative value produces backscattering while a larger positive value produces forward scattering. The effects of each can be observed when focusing on the face of the statue and the disk of light at its feet. Backscattering will illuminate the face while forward scattering will make the disk at the base brighter.

5.4 Helios Coefficient

The helios coefficient is an atmospheric radiance amplifier we added to the Monte Carlo integration. This helps produce a sharper contrast from areas where scattering is not taking place as often. The below images demonstrates the effect, where the coefficient is set to 1 we have a nearly dark room as scattering events in the atmosphere contribute very little radiance. A higher helios coefficient will bring out the god ray effect, as scattering events become more apparent.

ACKNOWLEDGMENTS

The authors would like to thank Professor Ren Ng of UC Berkeley's Foundations of Computer Graphics course - CS284A

REFERENCES

[1] Wojciech Jarosz. 2008. *Efficient Monte Carlo Methods for Light Transport in Scattering Media.* Ph.D. Dissertation. UC San Diego, La Jolla, CA, USA. Advisor(s) Henrik Wann Jensen and Matthias Zwicker.

[2] James T. Kajiya. 1986. The Rendering Equation. *SIGGRAPH Comput. Graph.* 20, 4 (Aug. 1986), 143–150. https://doi.org/10.1145/15886.15902

[3] A. J. Preetham, Peter Shirley, and Brian Smits. 1999. A Practical Analytic Model for Daylight. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '99).* ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 91–100. https://doi.org/10.1145/311535.311545

[4] Dominique Toublanc. 1996. Henyey–Greenstein and Mie phase functions in Monte Carlo radiative transfer computations. *Appl. Opt.* 35, 18 (Jun 1996), 3270–3274. https://doi.org/10.1364/AO.35.003270

[5] Eric Veach and Leonidas J. Guibas. 1997. Metropolis Light Transport. In *Computer Graphics (SIGGRAPH '97 Proceedings.* Addison Wesley, 65–76.

Received April 2017; accepted May 2017