Finding the Lights in a Room

(Fast Illumination Map Sampling)

Sameer Agarwal

Artificial Intelligence Laboratory, UCSD

Digital Imagery

$$\text{Geometry} + \text{Light} = \text{Image}$$

Definitions

Radiance

Radiance is the amount of energy per unit time per unit solid angle per unit area in the area of travel.

or

The number of photons striking a point from a particular direction per second.

Irradiance

Irradiance is the amount of energy per unit time per unit area.

The Rendering Equation

$$L_r(x,\vec{\omega}) = \int_\Omega f_r(x,\vec{\omega}',\vec{\omega})dE_i(x,\vec{\omega}')$$
$$= \int_\Omega f_r(x,\vec{\omega}',\vec{\omega})L_i(x,\vec{\omega}')(\vec{\omega}'\cdot\vec{n})d\vec{\omega}'$$

L_i is the incoming radiance

$\vec{n}$ is the direction of the surface normal at x

Ω is the hemisphere of incoming directions at x

$d\vec{\omega}$ is a differential solid angle

f is the BRDF

BRDF

Bi-Directional Radiance Distribution Function describes the reflection of light at a surface. It defines the relationship between the reflected radiance and irradiance.

- Diffuse A surface that reflects light in all directions. A perfectly diffuse surface reflects light equally in all directions.
- Specular Shiny and reflects like a mirror

Synthetic Lighting

- The position, size, brightness and type of each light in the scene is known apriori.
- Scenes usually have a small number of light sources.
- Solving the rendering equation, requires the evaluation of a small sum.

Image Based Lighting

Using images of real light to illuminate computer generated scenes.

Illumination Map

An omnidirectional, high dynamic range image that records the incident illumination conditions at a particular point in space.

Illumination Map

St. Peter’s Basilica

Rendering using IBL

- Solving the Rendering Equation requires numerically integration over a hemisphere.
- Full integration is extremely expensive
- Estimate Integral using random sampling.

Monte-Carlo Rendering

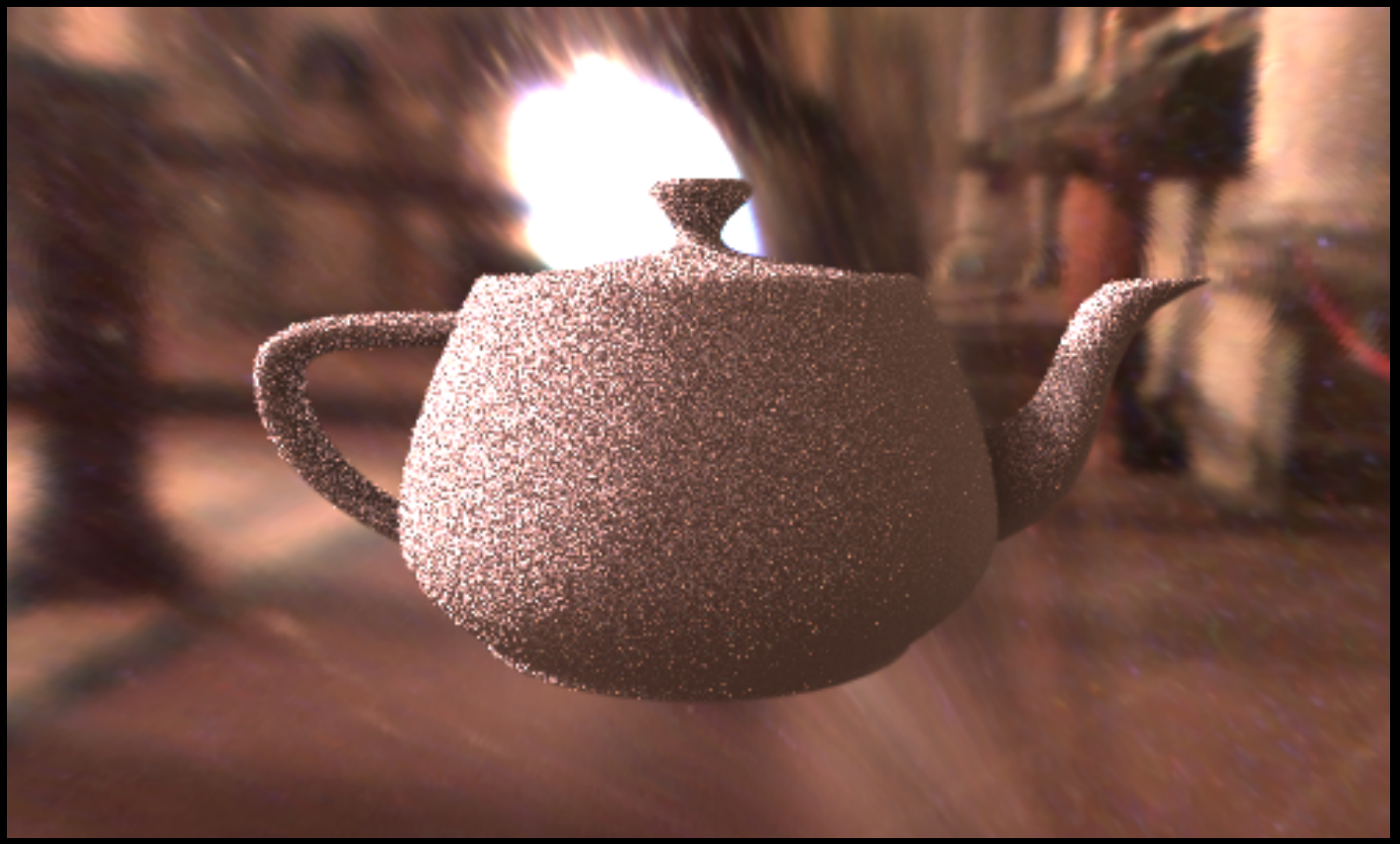
- Rendering a scene requires solving for L_r at every point in the scene.
- Too expensive

Monte-Carlo Rendering

Evaluate the integral by random sampling.

1. Choose a viewing direction
2. Find the nearest scene point in that direction
   (a) Randomly choose a direction.
   (b) Find the amount of light incident from that direction.
   (c) Accumulate
   (d) Repeat until ”convergence”
3. Repeat until image plane is covered.

The joy of variance



The joy of variance

- Illumination Maps are positive functions on a sphere
- A few peaks and mostly plain
- High Variance
- A noise free estimate requires a large number of samples.

Existing Solutions

- Low pass filtering Approximate the illumination map by the first few fourier coefficients.
- Importance Sampling Bias the sampling towards high intensity regions of the illumination map.
- Interpolation Estimate the amount of reflected radiance at a point as an average of the neighboring values.

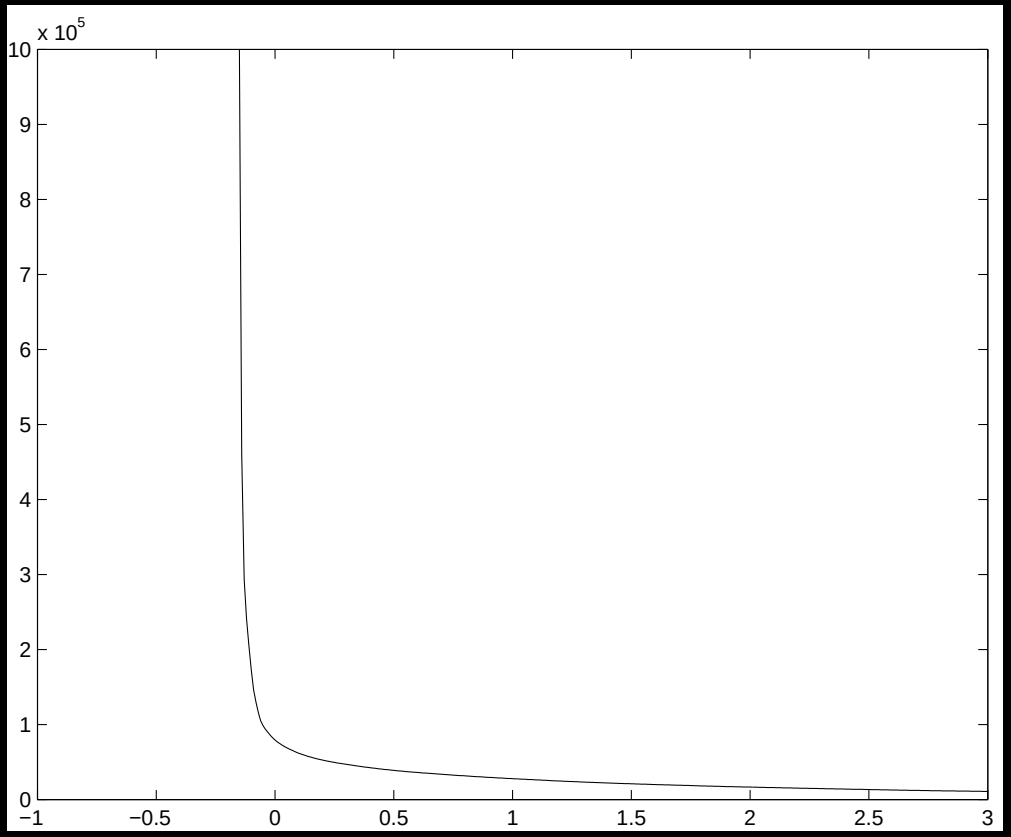
The Idea

- Approximate a light map by an equivalent collection of point light sources.
- Replace the pure Monte Carlo shader with a deterministic shader, or a monte-carlo shader based on this point source collection.

Finding the lights in a room

1. Threshold
2. Detect Connected Components
3. Cut into pieces
4. Deterministically Sample

Threshold



$$\tau = \mu + \lambda\sigma$$

Cutting up a light into pieces

Given a patch P on the surface of the unit sphere and $0 < \epsilon < 4\pi$, find the smallest sized partition

$$\underset{\sim}{C} = \{c_i\}, \quad P = \cup_i c_i$$

s.t.

$$\max_i(area(c_i)) < \epsilon$$

Equivalent to a clustering problem, with the constrain that no cluster should have area larger than ϵ.

Hochbaum-Shmoys

1. Dataset $Y = \{y_1, y_2, \ldots, y_n\}$
2. Pick a point and label it x_1.
3. For $i = 2, 3, \ldots, k$

   (a) $x_i = \underset{y_p}{argmax} \left[\underset{q}{max} \left[d(y_p, x_q) \right] \right]$
4. $C_j = \{y : y \in Y, x_j = \underset{x_l}{argmin}[d(y, x_l)]\}$
5. Return $\{(x_i, C_i)\}$

ϵ-Hochbaum-Shmoys

1. Dataset $Y = \{y_1, y_2, \ldots, y_n\}$
2. Pick a point and label it x_1.
3. Assign $C_1 = Y$
4. while ($\max(area(C_i)) < \epsilon$)
   (a) $x_i = \underset{y_p}{argmax} \left[\underset{q}{max} \left[d(y_p, x_q) \right] \right]$
   (b) $C_j = \{y : y \in Y, x_j = \underset{x_l}{argmin}[d(y, x_l)]\}$
5. Return $\{(x_i, C_i)\}$

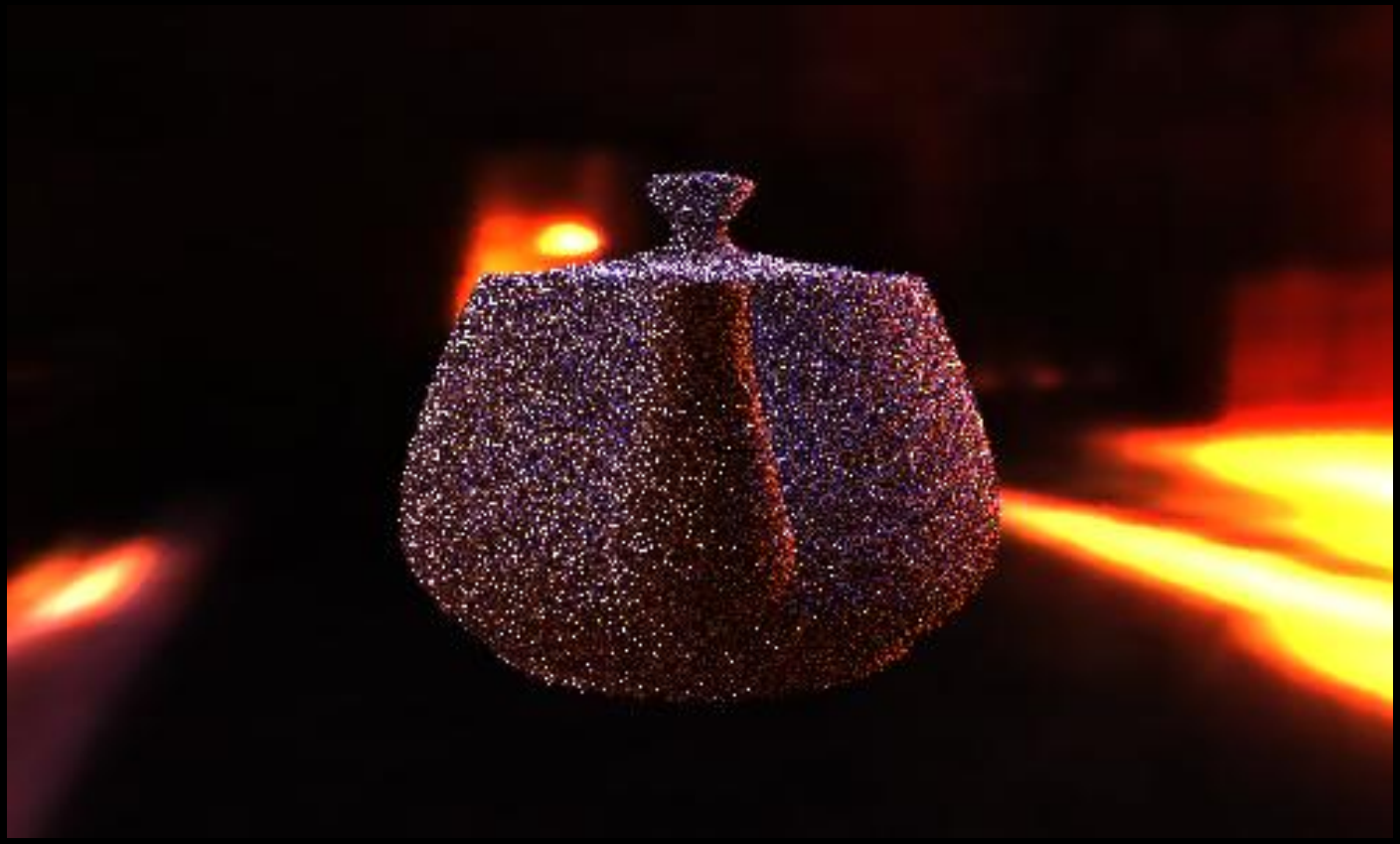
Lights Camera ...

$$l_i = \{x_i, area(C_i), R(C_i), G(C_i), B(C_i)\}$$

IMS_SHADER

1. For each l_i, calculate contribution.
2. Approximate the residual illumination using spherical harmonics.

Results: Grace Cathedral



Pure Monte Carlo Rendering

Results: Grace Cathedral

Low Pass Filtering and Interpolation Hacks

Results: Grace Cathedral

Discrete Sampling

Results: Grace Cathedral

Animation !

Results: Galileo's Tomb

Animation !

Future Work

- Compare with pure importance sampling.
- Implement a randomized sampler on top of the discrete representation.
- Explore extensions to other kind of BRDFs.
- Improve Clustering Algorithm

Acknowledgments

- Henrik Jensen
- Serge Belongie
- Salvador Dali
- The music of J5.