Microsoft DIRECTX®

Going Beyond Reflectance with Advanced Precomputed Rendering Techniques

Beyond Reflectance

- A reflectance model is a very simple model for how a scene responds to light
- More general response models exist

Beyond Reflectance

- BSSRDF (Bidirectional Surface Scattering Reflectance Distribution Function)
- Does not assume that light exits and enters at the same point
- However, only paths which enter the object, travel inside and then leave the object are considered
- Not paths which re-enter the object multiple times or interact with multiple objects

Beyond Reflectance

- The most general response model is the global illumination scene response
- Which covers all the interactions of the light stimulus with various parts of the scene
- This includes light being reflected several times from different parts of the scene, occluded by one part of the scene from another part, etc.

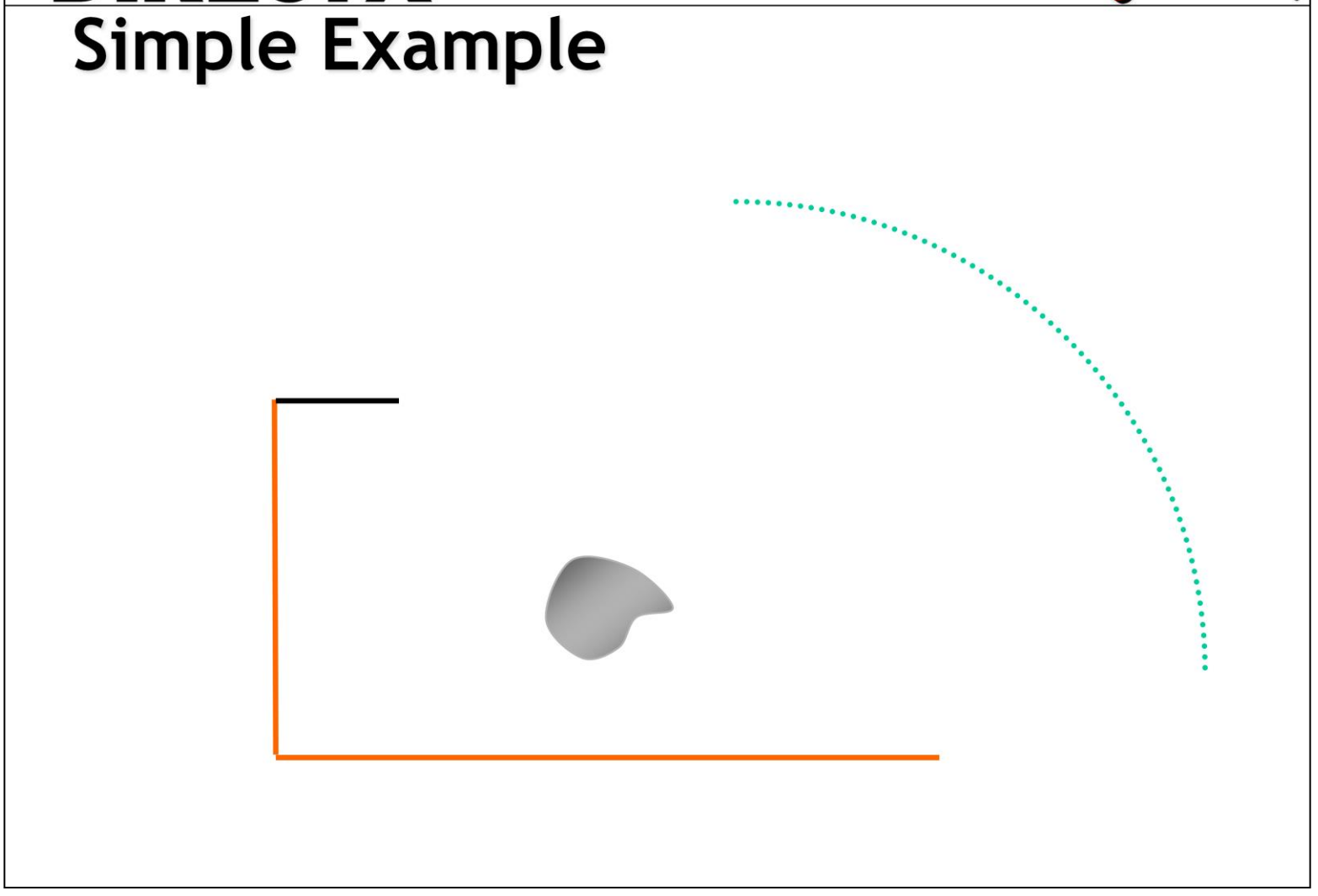
Precomputed Techniques

- Solving the rendering problem is hard off-line
- Using precomputed data in real time is still the most viable general solution
- What is practical to precompute?
- What isn't?

What are we trying to solve?

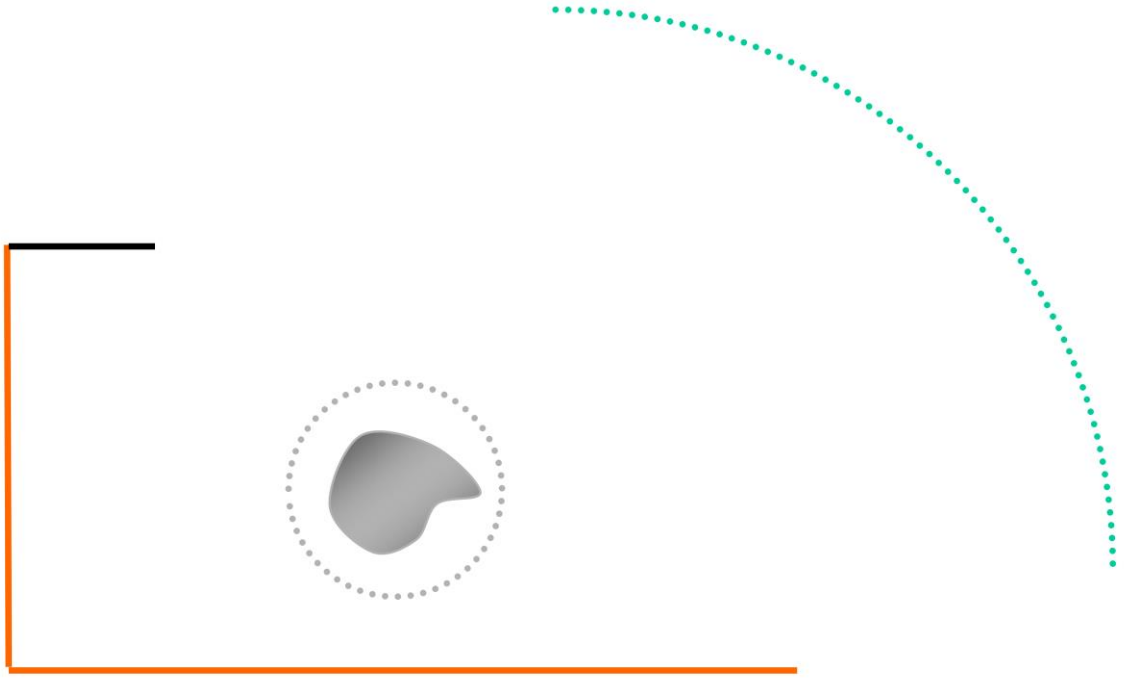
- Four basic problems need to be addressed
  - How dynamic objects are illuminated by general lighting models
  - How scenes are illuminated (GI), possibly by a parameterized model of lighting
  - How scene illuminates objects
  - How objects effect the lighting in the scene

Simple Example



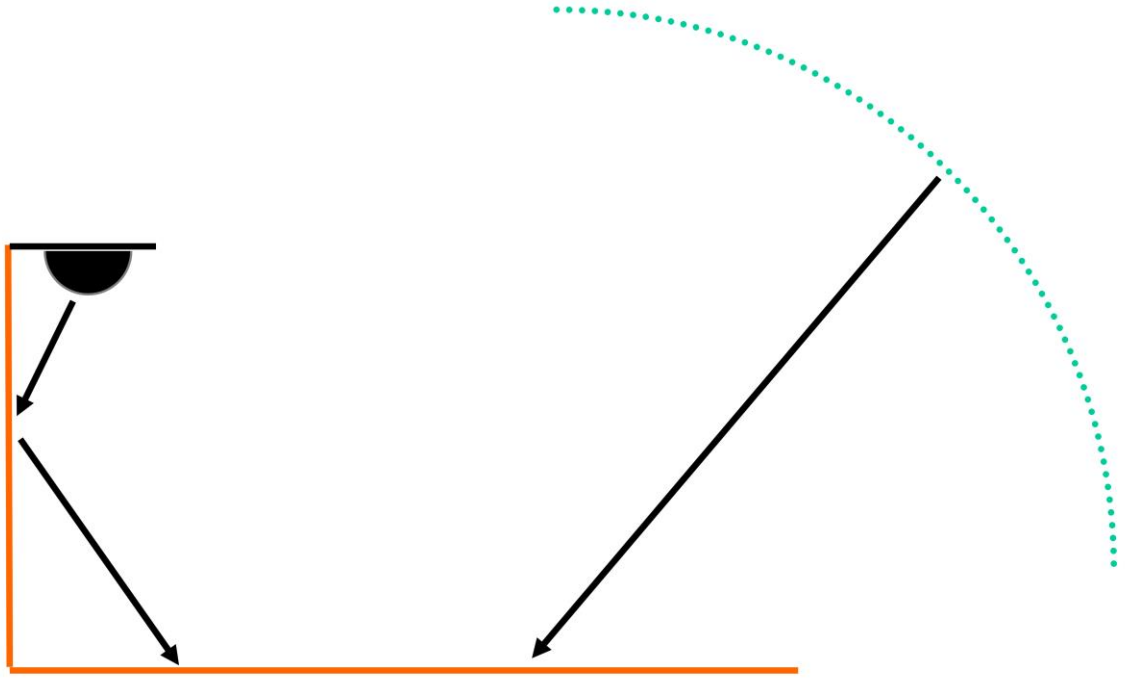
Here we have a simple scene, with a static area light and dynamic distant sky light. We now place an object in the scene, that can move around.

Object Lit by Dynamic Light



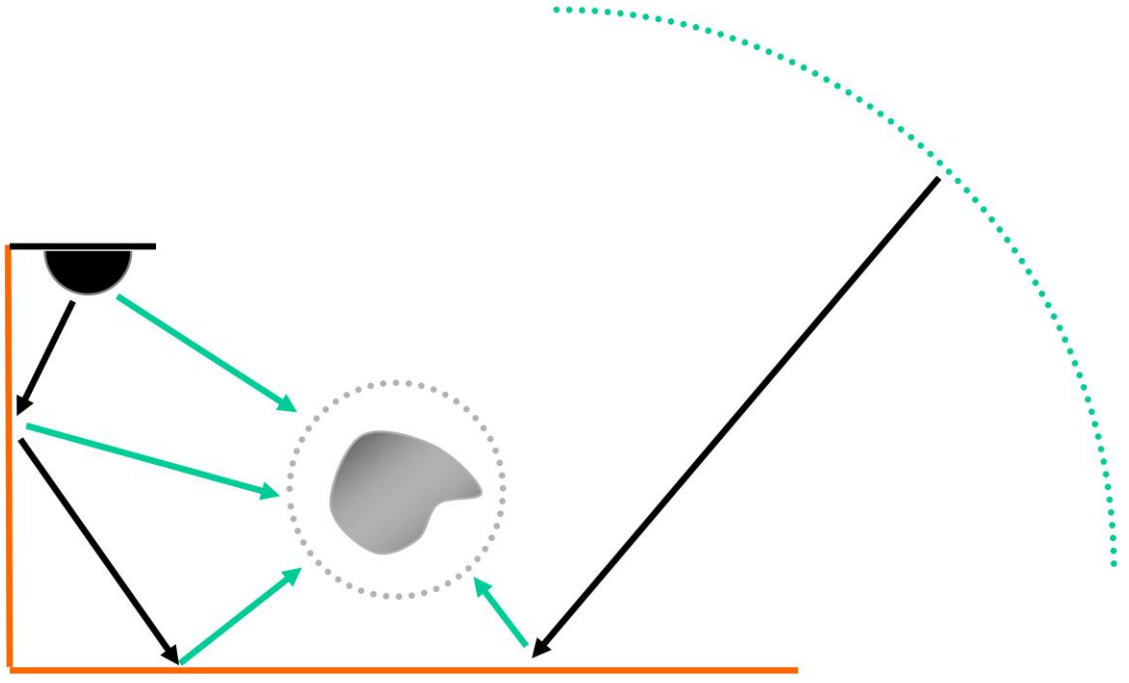
Given an object, and some model of its lighting environment, you want to render the given object with shadowing/inter-reflections/scattering from that light. This is what most of the previous PRT papers have really been centered around.

Static Scene Illuminated



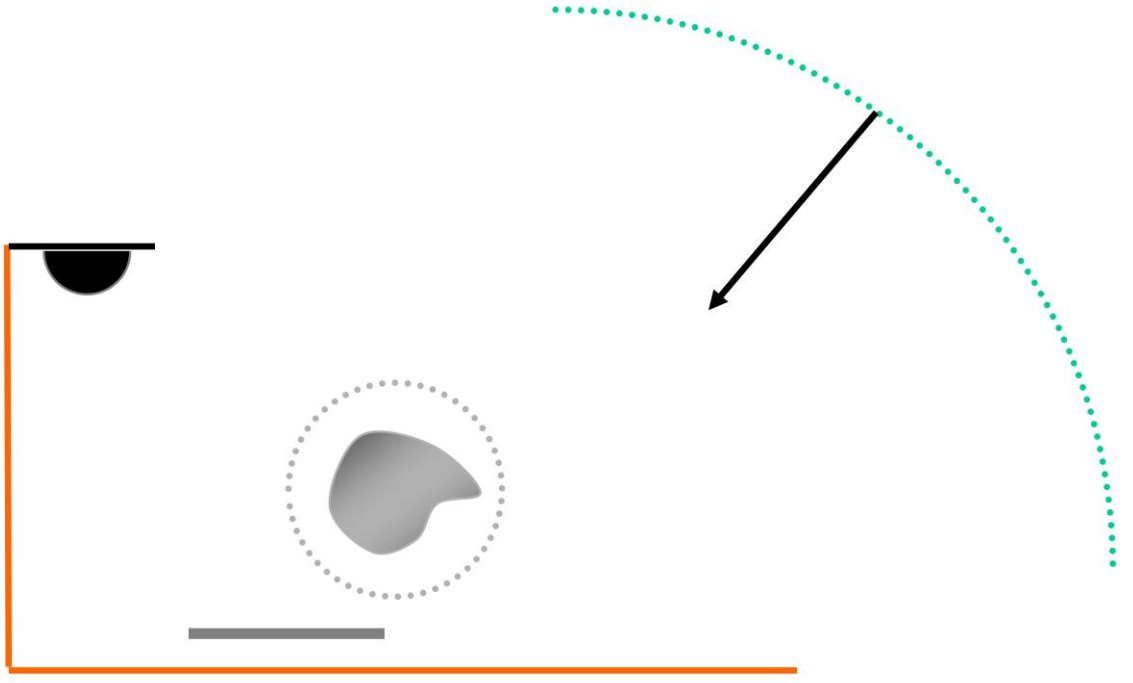
If you have a static scene, it can be illuminated by precomputed light maps, and possible transfer vectors from some model of distant lighting (a skylight model is an example.)

Object Illuminated by Scene



Given a (possibly parameterized) model of scene global illumination, it's important to be able to light objects with it. At each point in space it's a spherical function that depends on the dynamic lights & the transport paths for the lightmaps. If you can model this function, you can light an object with it.

Object Shadowing Scene

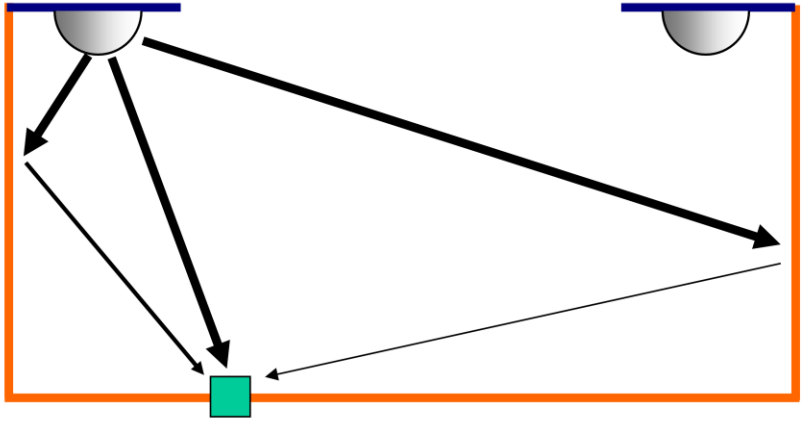


Here we see a relatively hard shadow that an object casts on the scene. As the object moves away from the receivers, the size of the shadow grows, but the details blur – shadows become much softer. This helps tie the object visually into the scene.

Scene Response

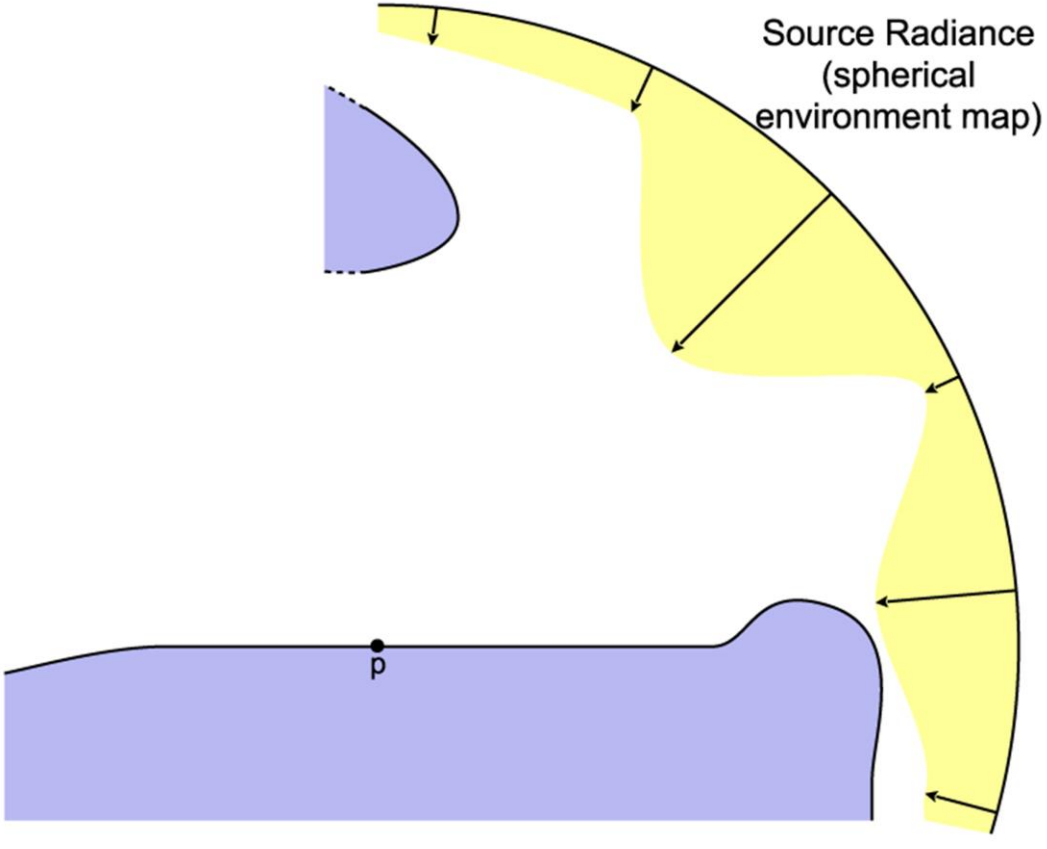
- Given some model of light, generate scene response
  - Lightmap – response to a set of “area” light sources, generally only models diffuse transport
  - Could have several light maps instead, turn them on/off independently
  - More general model of light?

General Scene Response



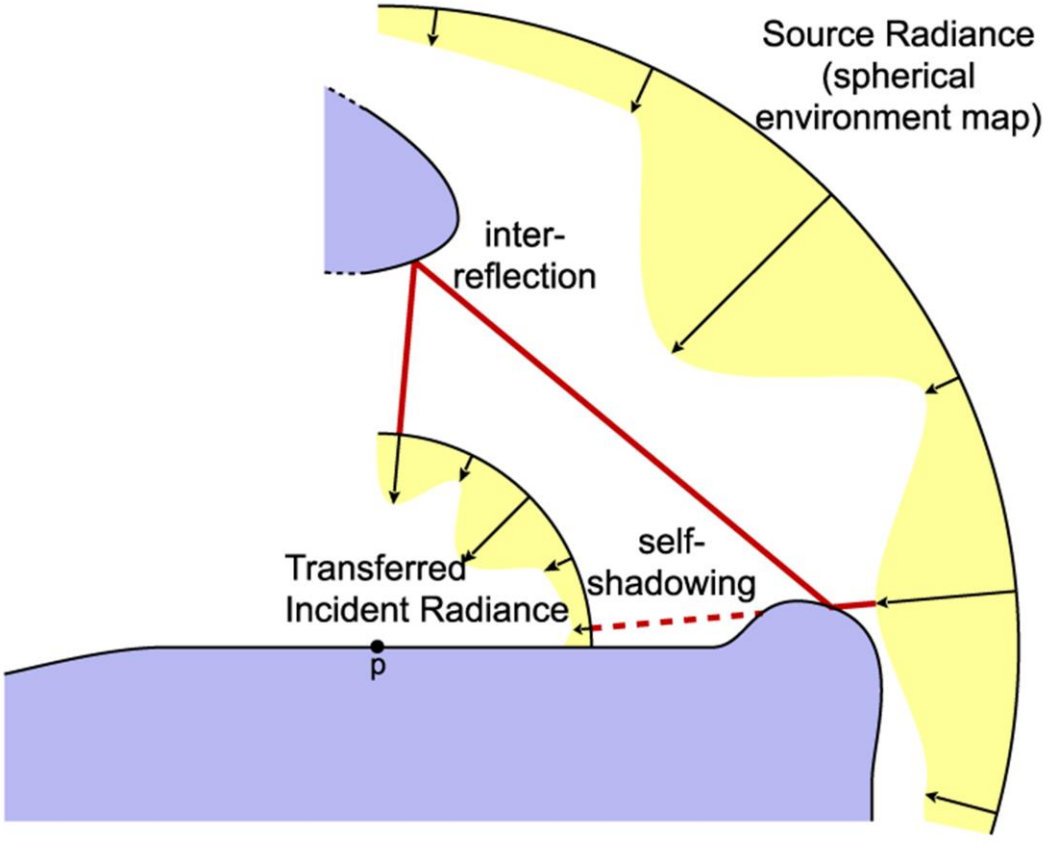
Transfer Vector

Terminology



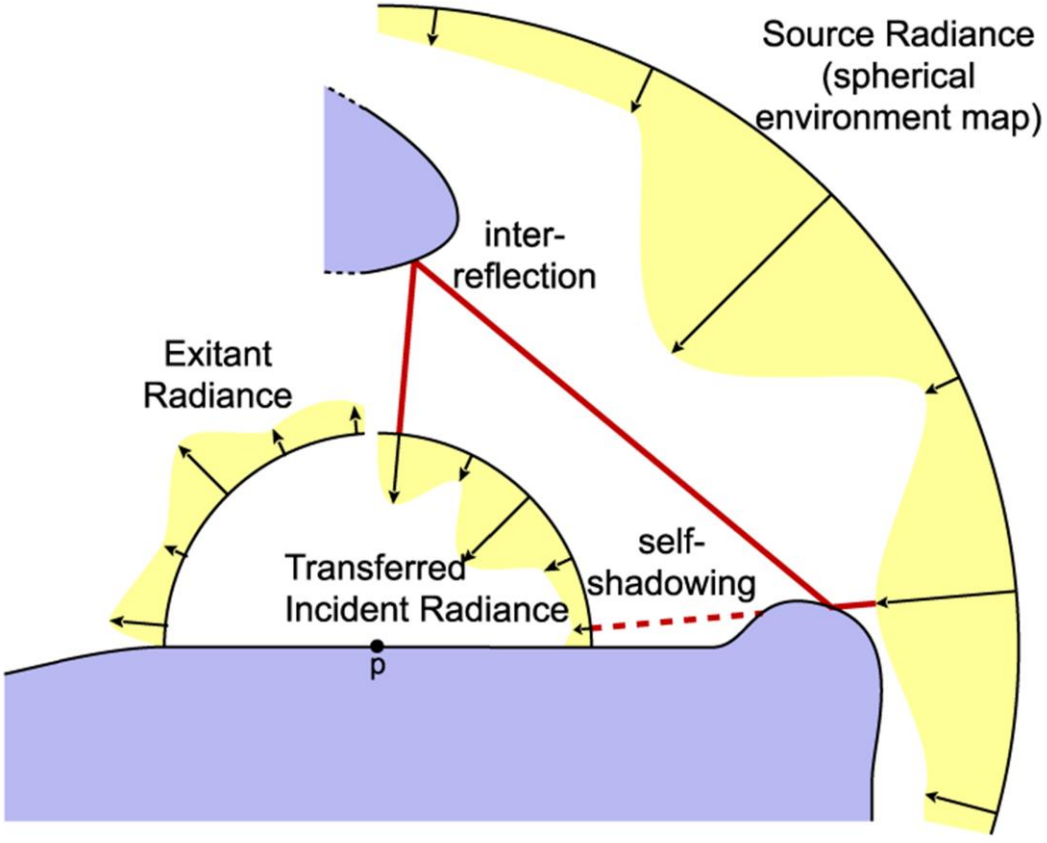
Now we are going to define some terms. We use the abbreviation “PRT” for precomputed radiance transfer. In PRT, the source radiance function represents the distant lighting that will illuminate the object. It’s a spherical function represented by a vector of coefficients with respect to a given basis.

Terminology



At every point over the surface of the object, this source radiance function is attenuated due to shadowing and increased due to inter-reflections. We call the result transferred incident radiance. It represents the local environment lighting each point “p”.

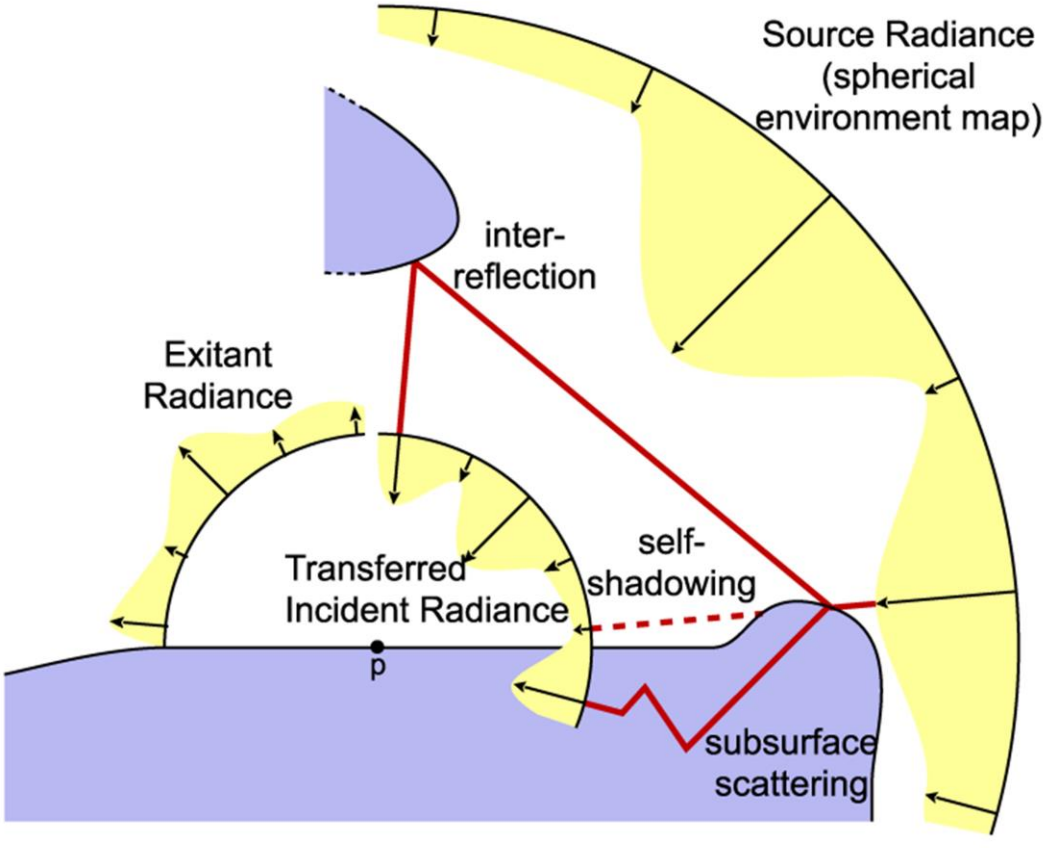
Terminology



This transferred incident radiance function can be integrated against the surface material properties to produce the exit radiance emanating from that point.

Exit radiance is a spherical function that is parameterized by view direction.

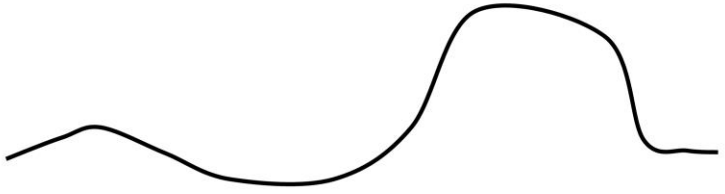
Terminology



We can also model subsurface scattering – light that propagates inside the object and leaves from the given point, also contributing to exit radiance.

Rendering Equation

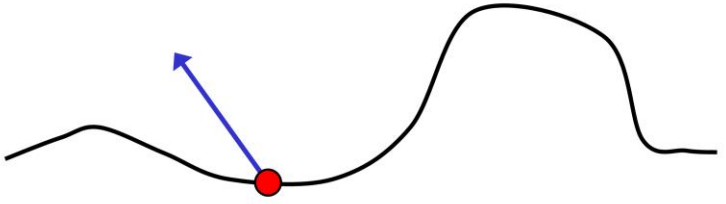
$$L\left(p \to \vec{d}\right) = L_e\left(p \to \vec{d}\right) + \int_{\Omega} f_r\left(p, \vec{s} \to \vec{d}\right) L\left(p \leftarrow \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$



Given an object illuminated in a lighting environment, the rendering equation models the equilibrium of the flow of light in the scene. We will walk through a hemispherical formulation of this equation.

Rendering Equation

$$L\left(p \rightarrow \vec{d}\right) = L_e\left(p \rightarrow \vec{d}\right) + \int_{\Omega} f_r\left(p, \vec{s} \rightarrow \vec{d}\right) L\left(p \leftarrow \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$

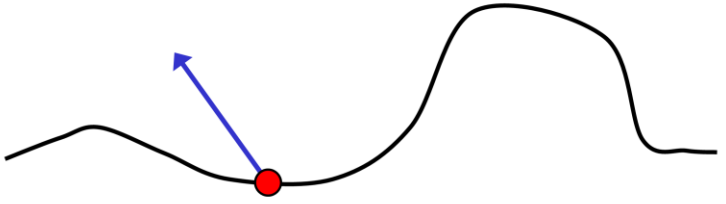


Radiance leaving point ***p*** in direction ***d***

The desired quantity is the radiance leaving a point on the object P in a given direction d.

Rendering Equation

$$L\left(p \rightarrow \vec{d}\right) = L_e\left(p \rightarrow \vec{d}\right) + \int_{\Omega} f_r\left(p, \vec{s} \rightarrow \vec{d}\right) L\left(p \leftarrow \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$

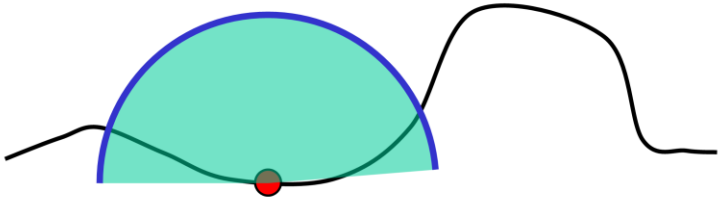


Radiance emitted from point ***p*** in direction ***d***

The first term is the radiance emitted directly from the point in the given direction. In our work we will assume that no objects emit light, they are just lit by a distant lighting environment (the source radiance function.)

Rendering Equation

$$L\left(p \to \vec{d}\right) = L_e\left(p \to \vec{d}\right) + \int_{\Omega} f_r\left(p, \vec{s} \to \vec{d}\right) L\left(p \leftarrow \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$

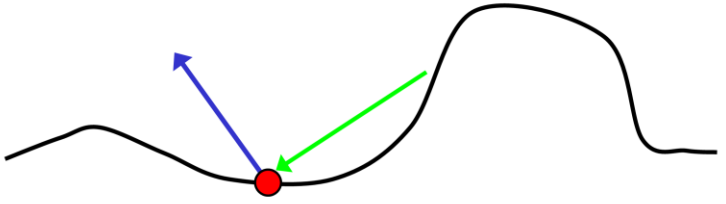


Integral over directions ***s*** on the hemisphere around ***p***

This is followed by an integral over the hemisphere around the point, where s is used to denote a direction on this hemisphere

Rendering Equation

$$L\left(p \rightarrow \vec{d}\right) = L_e\left(p \rightarrow \vec{d}\right) + \int_{\Omega} f_r\left(p, \vec{s} \rightarrow \vec{d}\right) L\left(p \leftarrow \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$

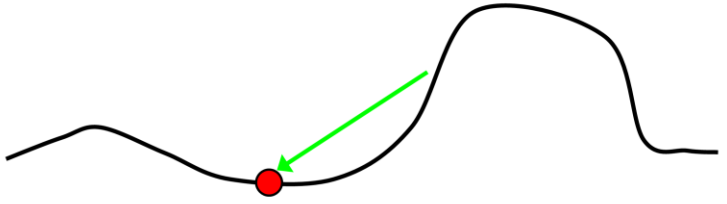


BRDF at point ***p*** evaluated for incident direction ***s*** in outgoing direction ***d***

The 1st factor inside the integral is the BRDF of the surface at point P, the BRDF is a 4D function that models what percent of light for some input direction (s) leaves in some exit direction (d).

Rendering Equation

$$L\left(p \rightarrow \vec{d}\right) = L_e\left(p \rightarrow \vec{d}\right) + \int_{\Omega} f_r\left(p, \vec{s} \rightarrow \vec{d}\right) L\left(p \leftarrow \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$

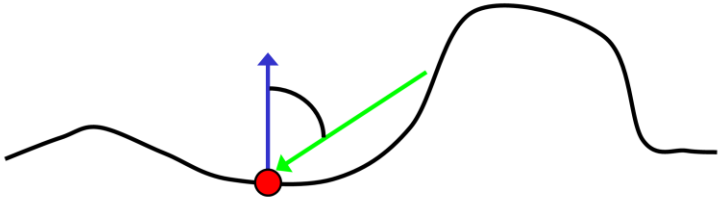


Radiance arriving at point ***p*** from direction ***s*** (also LHS)

The second term is the radiance arriving at point P from the direction S, note that this is also the variable we are solving for so this is an integral equation.

Rendering Equation

$$L\left(p \to \vec{d}\right) = L_e\left(p \to \vec{d}\right) + \int_{\Omega} f_r\left(p, \vec{s} \to \vec{d}\right) L\left(p \leftarrow \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$

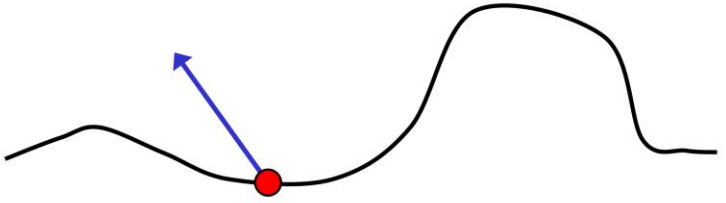


Lamberts law – cosine between normal and -s = dot(Np, -s)

The final term is the cosine term that comes from lamberts law – due to projected area.

Neumann Expansion

$$L\left(p \to \vec{d}\right) = L_0\left(p \to \vec{d}\right) + L_1\left(p \to \vec{d}\right) + \cdots$$

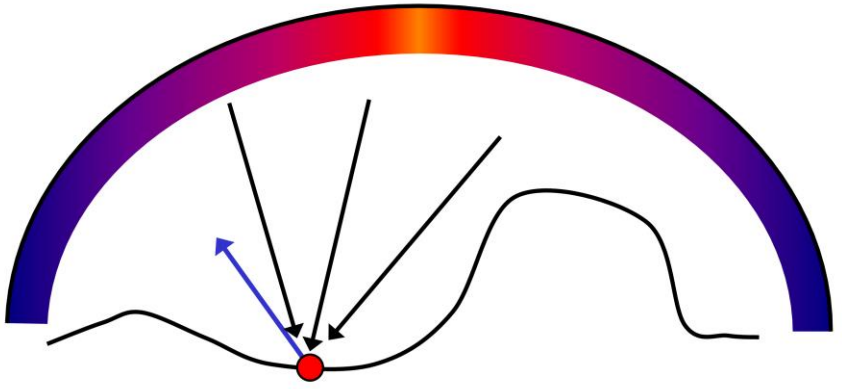


Exit radiance expressed as infinite series

One convenient way to reason about the solution to this integral equation is by using a Neumann expansion of this expression, where exit radiance is expressed as an infinite series.

Neumann Expansion

$$L\left(p \to \vec{d}\right) = L_0\left(p \to \vec{d}\right) + L_1\left(p \to \vec{d}\right) + \cdots$$



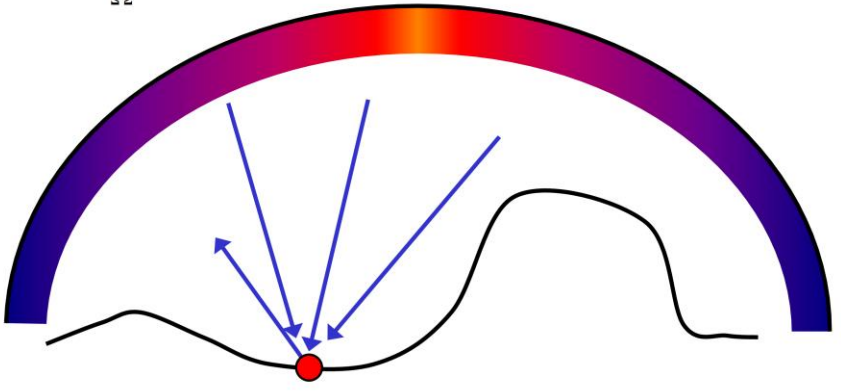
Direct lighting arriving at point ***p*** – from distant environment

The first term in this series is the direct lighting arriving at point P from a distant lighting environment – the source radiance environment we referred to earlier.

Neumann Expansion

$$L\left(p \to \vec{d}\right) = L_0\left(p \to \vec{d}\right) + L_1\left(p \to \vec{d}\right) + \cdots$$

$$L_0\left(p \to \vec{d}\right) = \int_{\Omega} f_r\left(p, \vec{s} \to \vec{d}\right) L_S\left(p \leftarrow \vec{s}\right) V\left(p \to \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$



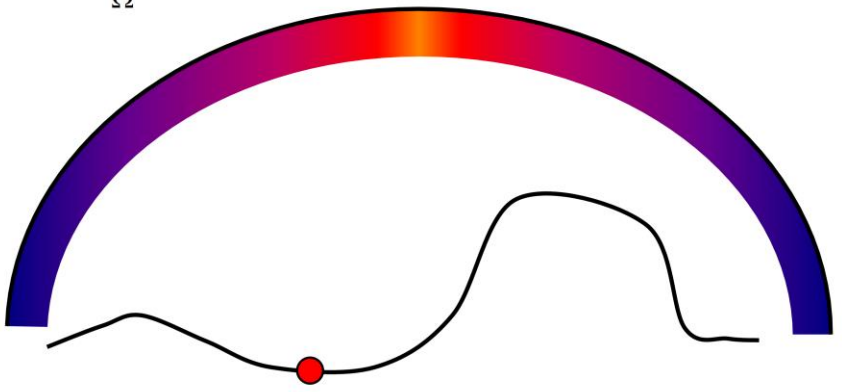
Direct lighting arriving at point ***p*** – from distant environment

This term is an integral over the hemisphere at the point p.

Neumann Expansion

$$L\left(p \to \vec{d}\right) = L_0\left(p \to \vec{d}\right) + L_1\left(p \to \vec{d}\right) + \cdots$$

$$L_0\left(p \to \vec{d}\right) = \int_{\Omega} f_r\left(p, \vec{s} \to \vec{d}\right) L_S\left(p \leftarrow \vec{s}\right) V\left(p \to \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$



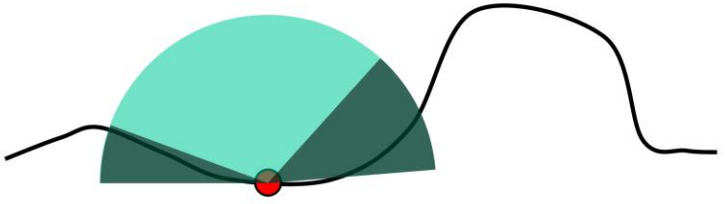
Source Radiance – distant lighting environment

One of the new factors in this expression is Ls – the source radiance function we are assuming that this is the only source of light in the scene. This is just a conventional integral, the source radiance function only exists inside the integral.

Neumann Expansion

$$L\left(p \to \vec{d}\right) = L_0\left(p \to \vec{d}\right) + L_1\left(p \to \vec{d}\right) + \cdots$$

$$L_0\left(p \to \vec{d}\right) = \int_{\Omega} f_r\left(p, \vec{s} \to \vec{d}\right) L_S\left(p \leftarrow \vec{s}\right) V\left(p \to \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$

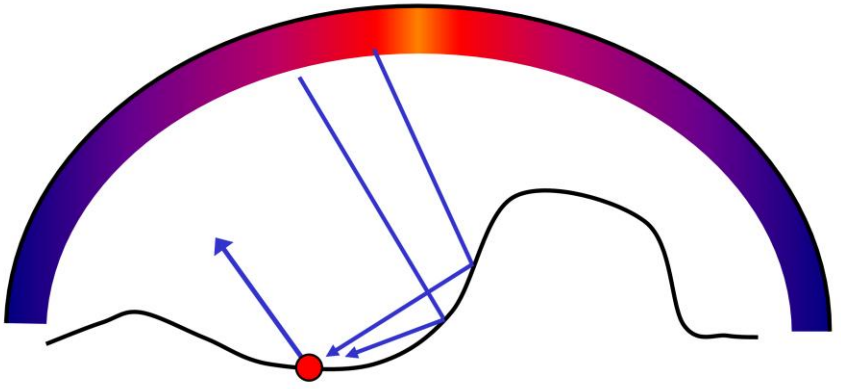


Visibility function - binary

The next new factor is the visibility function – this is a binary function that is 1 in a given direction if the point can “see” the distant lighting environment, and zero otherwise. L0 models how light that directly reaches the surface contributes to exit radiance.

Neumann Expansion

$$L\left(p \to \vec{d}\right) = L_0\left(p \to \vec{d}\right) + L_1\left(p \to \vec{d}\right) + \cdots$$



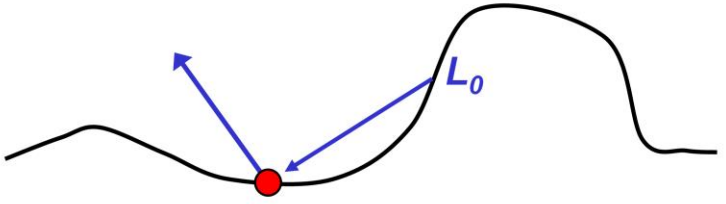
All paths from source that take 1 bounce

The next term in the expansion models all paths from the source radiance function that reach the given point after a single bounce and contribute to exit radiance in the given direction.

Neumann Expansion

$$L\left(p \to \vec{d}\right) = L_0\left(p \to \vec{d}\right) + L_1\left(p \to \vec{d}\right) + \cdots$$

$$L_1\left(p \to \vec{d}\right) = \int_{\Omega} f_r\left(p, \vec{s} \to \vec{d}\right) L_0\left(p \leftarrow \vec{s}\right) \left(1 - V\left(p \to \vec{s}\right)\right) H_{N_p}\left(-\vec{s}\right) ds$$



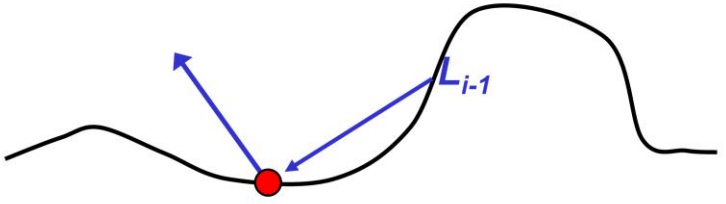
All paths from source that take 1 bounce

This is also just a conventional integral where the previous term (L0) is inside of the integral.

Neumann Expansion

$$L\left(p \to \vec{d}\right) = L_0\left(p \to \vec{d}\right) + L_1\left(p \to \vec{d}\right) + \cdots$$

$$L_i\left(p \to \vec{d}\right) = \int_{\Omega} f_r\left(p, \vec{s} \to \vec{d}\right) L_{i-1}\left(p \leftarrow \vec{s}\right) \left(1 - V\left(p \to \vec{s}\right)\right) H_{N_p}\left(-\vec{s}\right) ds$$



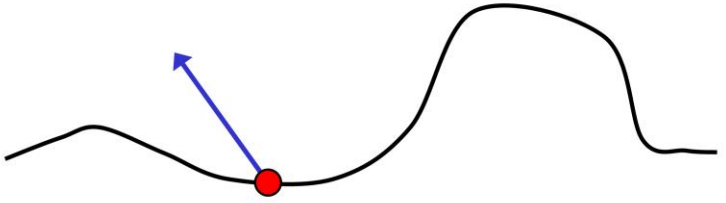
All paths from source that take i bounces

In general the ith bounce models how all of the energy from the “previous bounce” contributes to exit radiance in the given direction.

Diffuse PRT

$$L\left(p \to \vec{d}\right) = L_0\left(p \to \vec{d}\right) + L_1\left(p \to \vec{d}\right) + \cdots$$

$$L_0\left(p \to \vec{d}\right) = \int_{\Omega} f_r\left(p, \vec{s} \to \vec{d}\right) L_S\left(p \leftarrow \vec{s}\right) V\left(p \to \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$



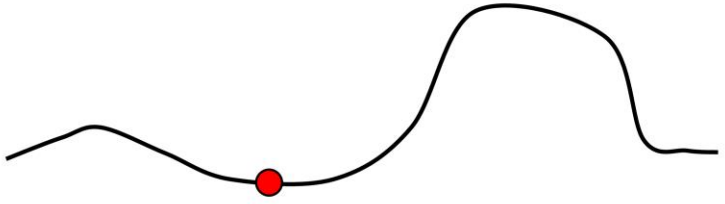
To derive PRT for the diffuse case we are going to start with just the direct term from the Neumann expansion of the rendering equation and make several simplifying assumptions.

Diffuse PRT

$$L\left(p \to \vec{d}\right) = L_0\left(p \to \vec{d}\right) + L_1\left(p \to \vec{d}\right) + \cdots$$

$$L_0\left(p \to \vec{d}\right) = \int_{\Omega} f_r\left(p, \vec{s} \to \vec{d}\right) L_S\left(p \leftarrow \vec{s}\right) V\left(p \to \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$

$$L_0\left(p\right) = \frac{\rho_d}{\pi} \int_{\Omega} L_S\left(-\vec{s}\right) V\left(p \to \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$



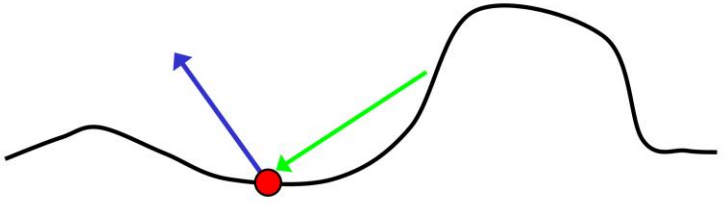
The bottom equation is the “simplified” form. First, for diffuse objects light is reflected equally in all directions, so exit radiance is independent of view direction.

Diffuse PRT

$$L\left(p \to \vec{d}\right) = L_0\left(p \to \vec{d}\right) + L_1\left(p \to \vec{d}\right) + \cdots$$

$$L_0\left(p \to \vec{d}\right) = \int_\Omega f_r\left(p, \vec{s} \to \vec{d}\right) L_S\left(p \leftarrow \vec{s}\right) V\left(p \to \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$

$$L_0\left(p\right) = \frac{\rho_d}{\pi} \int_\Omega L_S\left(-\vec{s}\right) V\left(p \to \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$



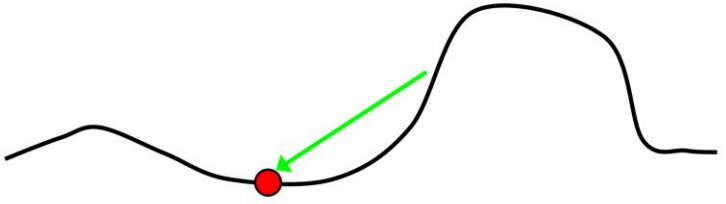
This also means the BRDF is just a constant (and independent of direction) so it can be pulled out of the integral. Rho_d represents the diffuse reflectivity of the surface, and is a number between 0 and 1. The divide by Pi enforces energy conservation.

Diffuse PRT

$$L\left(p \to \vec{d}\right) = L_0\left(p \to \vec{d}\right) + L_1\left(p \to \vec{d}\right) + \cdots$$

$$L_0\left(p \to \vec{d}\right) = \int_{\Omega} f_r\left(p, \vec{s} \to \vec{d}\right) L_S\left(p \leftarrow \vec{s}\right) V\left(p \to \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$

$$L_0\left(p\right) = \frac{\rho_d}{\pi} \int_{\Omega} L_S\left(-\vec{s}\right) V\left(p \to \vec{s}\right) H_{N_p}\left(-\vec{s}\right) ds$$

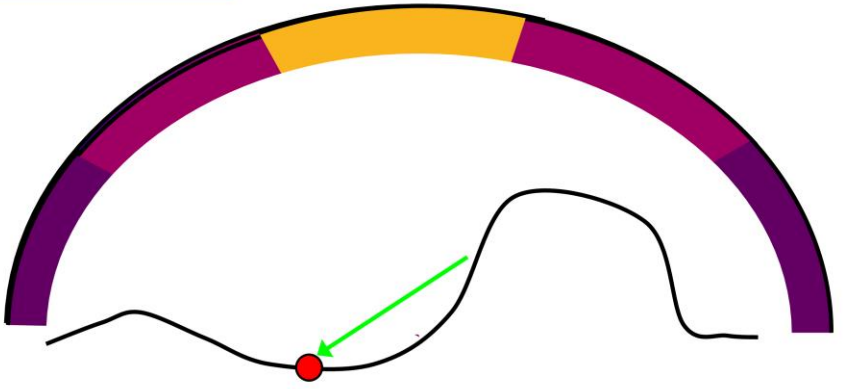


Finally, we assume the source radiance function is at infinity, this means we only need to concern ourselves with the direction s.

Diffuse PRT

$$L_0(p) = \frac{\rho_d}{\pi} \int_{\Omega} L_S(-\vec{s}) V(p \to \vec{s}) H_{N_p}(-\vec{s}) ds$$

$$L_S(-\vec{s}) \approx \sum_i l_i Y_i(-\vec{s})$$

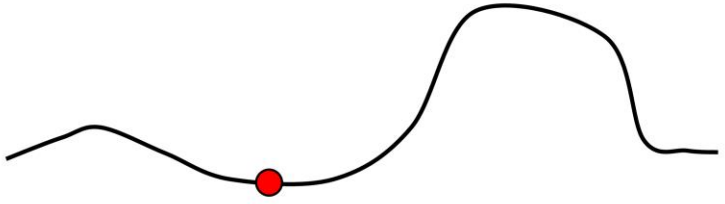


Now we are going to approximate the source radiance function with its projection into a set of basis functions on the sphere (denoted Yi in this equation). The lighting environment projection coefficients are denoted as li. For didactic purposes we use piecewise constant basis functions.

Diffuse PRT

$$L_0(p) = \frac{\rho_d}{\pi} \int_\Omega \left(\sum_i l_i Y_i(-\vec{s}) \right) V(p \to \vec{s}) H_{N_p}(-\vec{s}) \, ds$$

$$L_0(p) = \frac{\rho_d}{\pi} \sum_i l_i \int_\Omega Y_i(-\vec{s}) V(p \to \vec{s}) H_{N_p}(-\vec{s}) \, ds$$

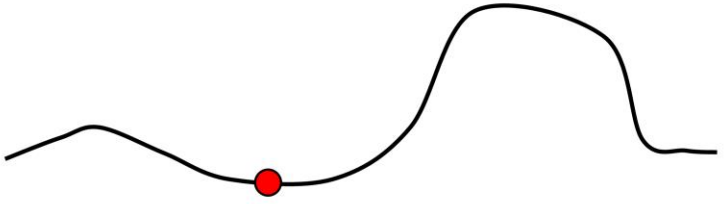


We can plug this approximation directly into the reflected radiance equation. Manipulating this expression while exploiting the fact that integration is a linear operator (sum of integrals = integral of sums), we can generate the following equivalent expression.

Diffuse PRT

$$L_0(p) = \frac{\rho_d}{\pi} \int_\Omega \left(\sum_i l_i Y_i(-\vec{s}) \right) V(p \to \vec{s}) H_{N_p}(-\vec{s})\, ds$$

$$L_0(p) = \frac{\rho_d}{\pi} \sum_i l_i \int_\Omega Y_i(-\vec{s}) V(p \to \vec{s}) H_{N_p}(-\vec{s})\, ds$$



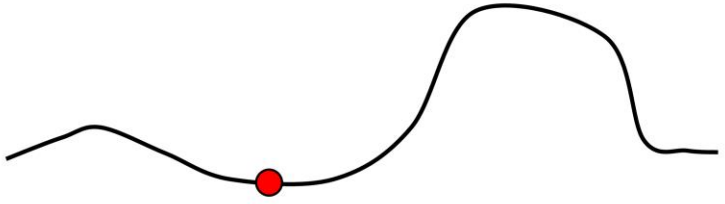
The important thing to note about the highlighted integral is that it is independent of the actual lighting environment being used, so it can be precomputed.

Diffuse PRT

$$L_0(p) = \frac{\rho_d}{\pi} \sum_i l_i \int_\Omega Y_i(-\vec{s}) V(p \to \vec{s}) H_{N_p}(-\vec{s})\, ds$$

$$L_0(p) = \frac{\rho_d}{\pi} \sum_i l_i t_{pi}^0$$

$$L_0(p) = \sum_i l_i t_{pi}^0$$



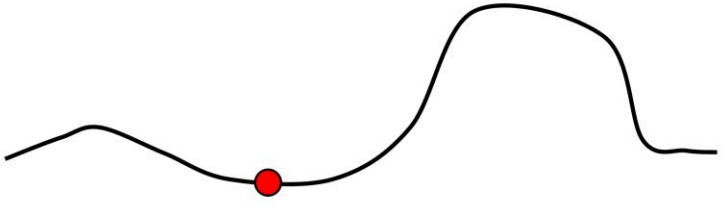
This integral represents a transfer coefficient – it maps how direct lighting in basis function I becomes exit radiance at point p. The set of transfer coefficients is a transfer vector that maps lighting into exit radiance. We can optionally fold the diffuse reflectivity into the transfer vector as well.

Diffuse PRT

$$L\left(p \to \vec{d}\right) = L_0\left(p \to \vec{d}\right) + L_1\left(p \to \vec{d}\right) + \cdots$$

$$L(p) = \sum_i l_i \left(t_{pi}^0 + t_{pi}^1 + \cdots\right)$$

$$L(p) = \sum_i l_i t_{pi}$$



A similar process can be used to model the other bounces, so that a final vector can be computed and used to map source radiance to exit radiance at every point on the object. Exit radiance is then just the dot product of the lights projection coefficients with the transfer vector.

Spherical Harmonics

- Spherical analog to Fourier transform
- Represents complex functions on the sphere, real form used in graphics

Spherical harmonics are the natural basis functions to use to represent functions on the unit sphere. They are the spherical analogue of the Fourier basis on the unit circle. They can represent complex valued functions over the sphere, but in graphics we use the form that strictly represents real valued functions.

Spherical Harmonics

- Spherical analog to Fourier transform
- Represents complex functions on the sphere, real form used in graphics
- Polynomials in R^3

Mathematically they are just polynomials in cartesean space (ie: xyz coordinates) restricted to the sphere.

Spherical Harmonics

- Spherical analog to Fourier transform
- Represents complex functions on the sphere, real form used in graphics
- Polynomials in R^3
  - Full basis through O has O^2 coeffs

The SH basis functions through order O have O^2 basis functions.

Spherical Harmonics

- Spherical analog to Fourier transform
- Represents complex functions on the sphere, real form used in graphics
- Polynomials in R^3
  - Full basis through O has O^2 coeffs
- Projection/Evaluation/Rotation are fairly straightforward

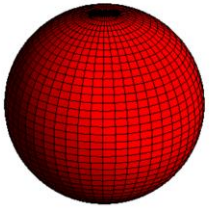
Projection/evaluation and rotation are fairly straightforward for the SH basis functions…

Spherical Harmonics

- Spherical analog to Fourier transform
- Represents complex functions on the sphere, real form used in graphics
- Polynomials in R^3
  - Full basis through O has O^2 coeffs
- Projection/Evaluation/Rotation are fairly straightforward
- Small number of bands implies "low frequency" lighting

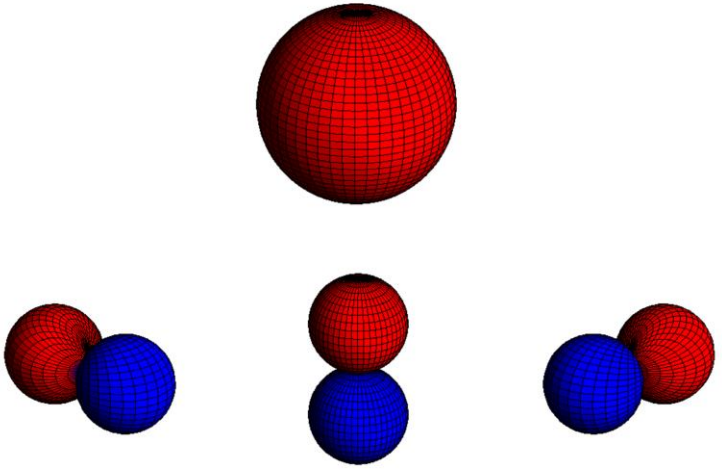
…and we use a small number of bands which implies we are limited to low frequency lighting. This is not that severe a limitation since this is exactly the kind of lighting environment that traditional interactive techniques can not handle.

Spherical Harmonics



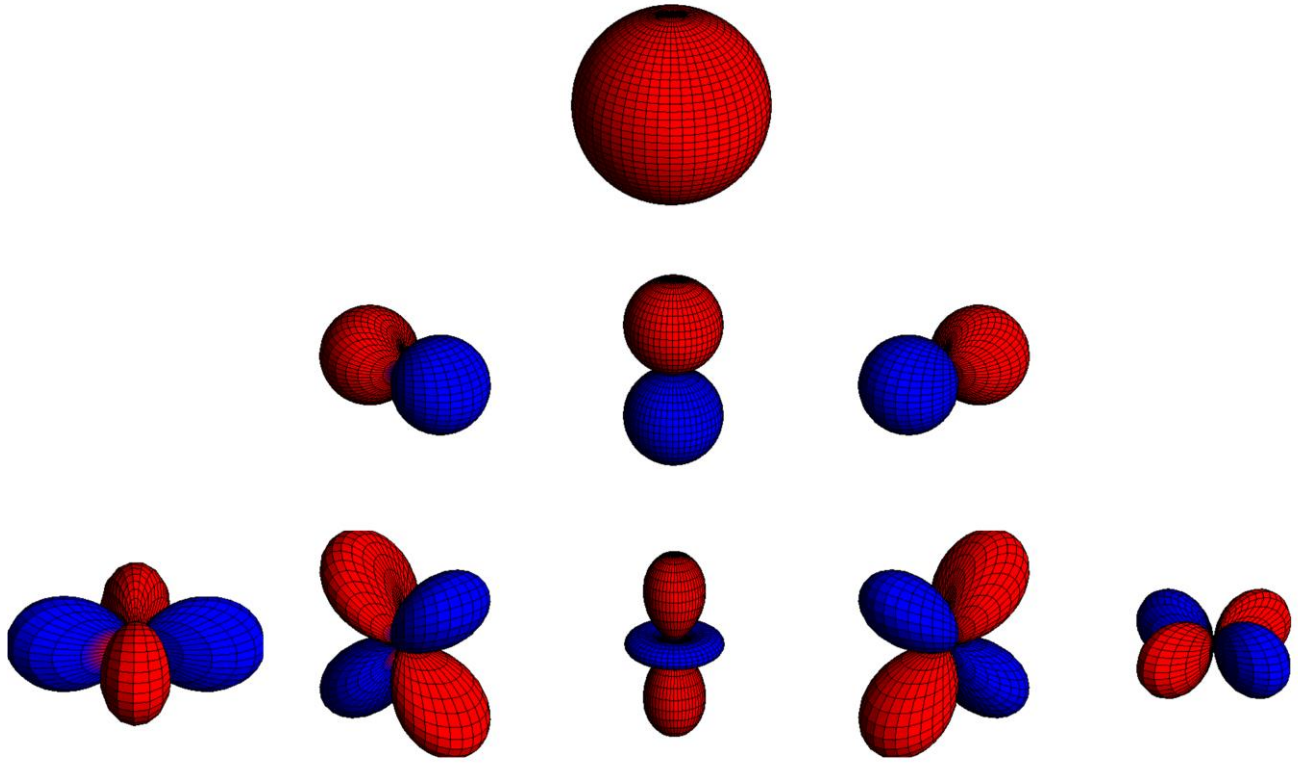
Here we will show some of the SH basis functions, visualized by evaluating them at a point on the sphere, and scaling the point based on the absolute value (sign is encoded via color; red is positive and blue is negative). This first basis function is just constant, this is like the DC term in the Fourier transform.

Spherical Harmonics



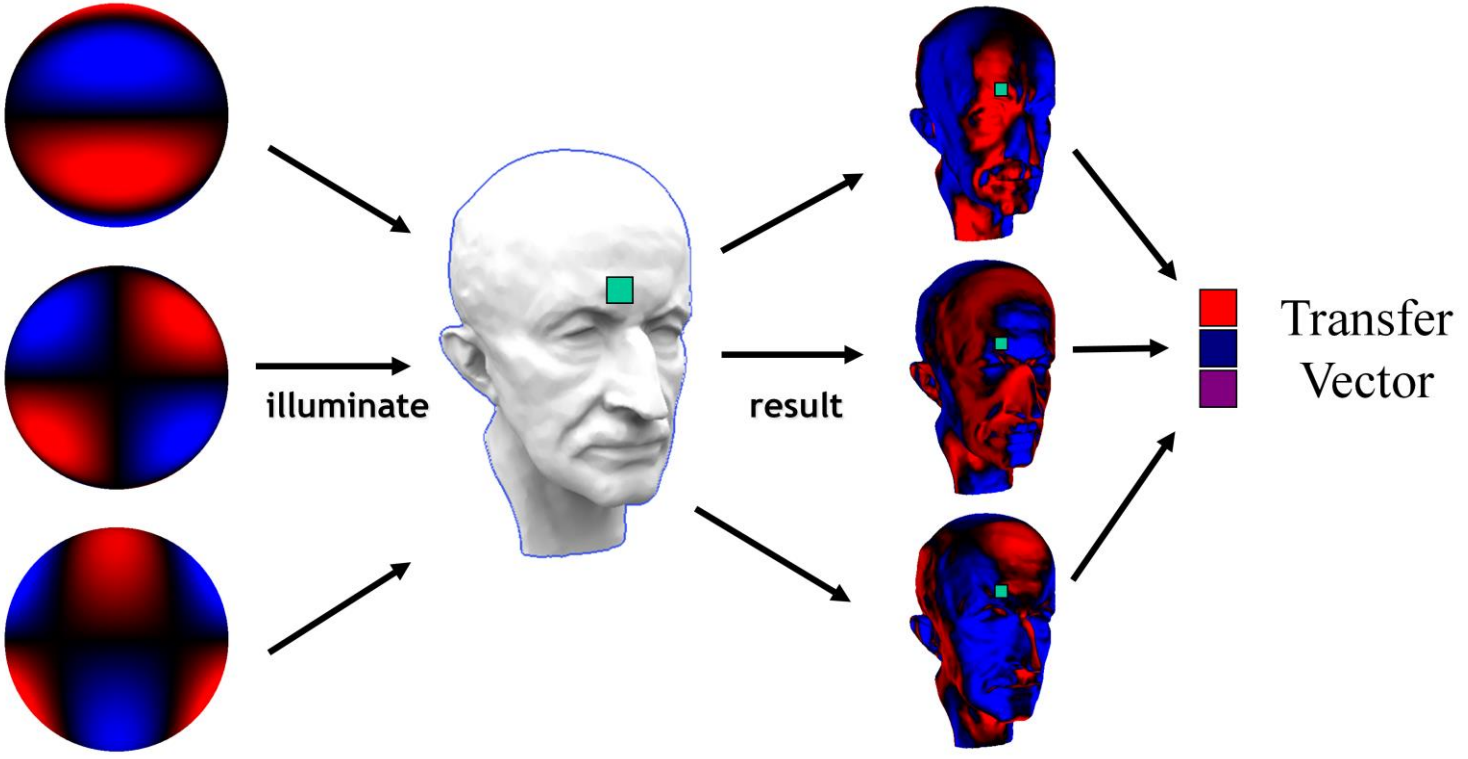
The next band are the linear functions – y followed by z followed by x.

Spherical Harmonics



The next band are the quadratic functions – each successive band is a higher degree polynomial and has more wiggles. Each band has 2 more basis functions then the one before it.

General Object Response



General Scene Response

- Rendering is scaling scene response by intensity of each “light” and summing
- Dot product generates an image
  - Factored form of an integral equation
- Relies on the spatial relationship between objects in the scene to be consistent (static scene)

Precomputed Basis

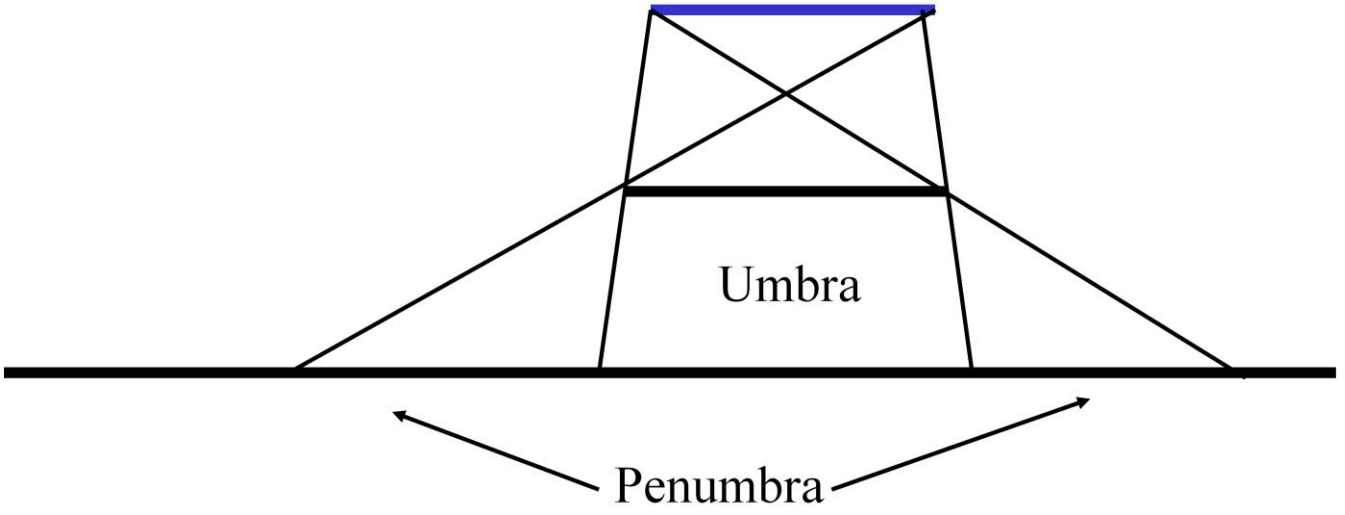
- Polynomial Texture Maps
  - [Malzbender2002]
  - Bi-quadratic polynomial (2 DOF)
- Steerable Illumination Textures
  - [Ashikhmin2002]
  - Steerable basis to model path of small area light (49 DOF)
- Directional Basis
  - [Hao2003]
  - Subsurface Scattering from directional lights

Precomputed Basis

- Spherical Harmonics
  - [Sig02,Sig03]
  - Applicable for low frequency light
  - Can model glossy materials
- Wavelets
  - [Ng03]
  - Models “all-frequencies”
  - Extended to glossy [Liu04,Ng04,Wang04]

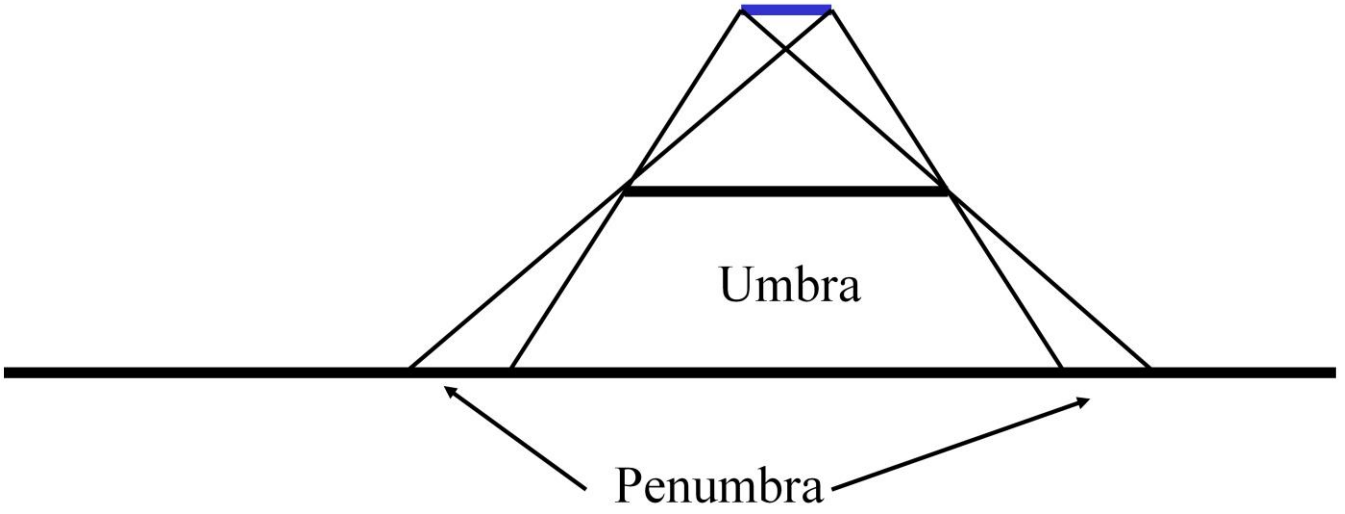
Spatial Sampling Issues

- Relationship between spatial sampling densities over objects and light frequencies



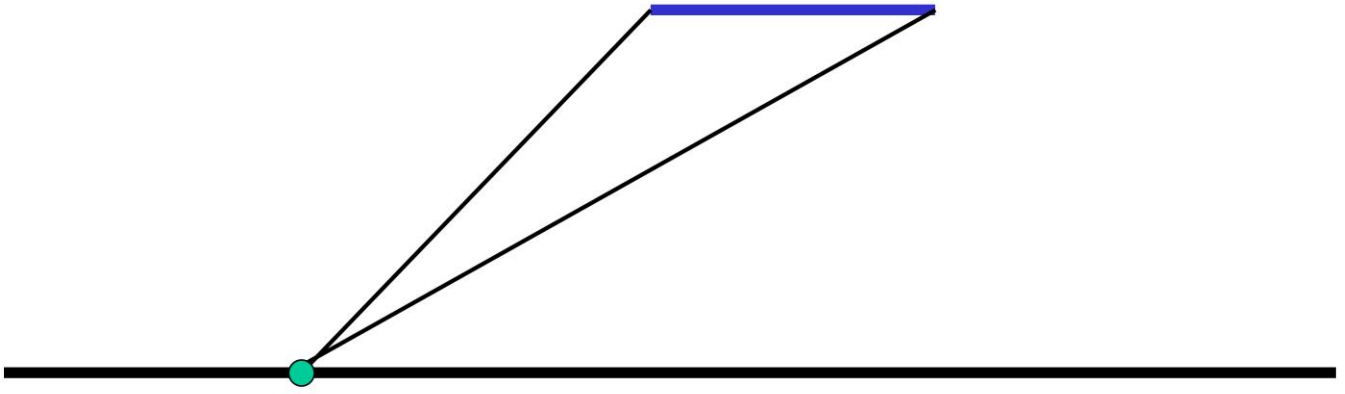
Spatial Sampling Issues

- Light shrinks -> penumbra tightens
- Higher sampling density to “move” light over object/scene



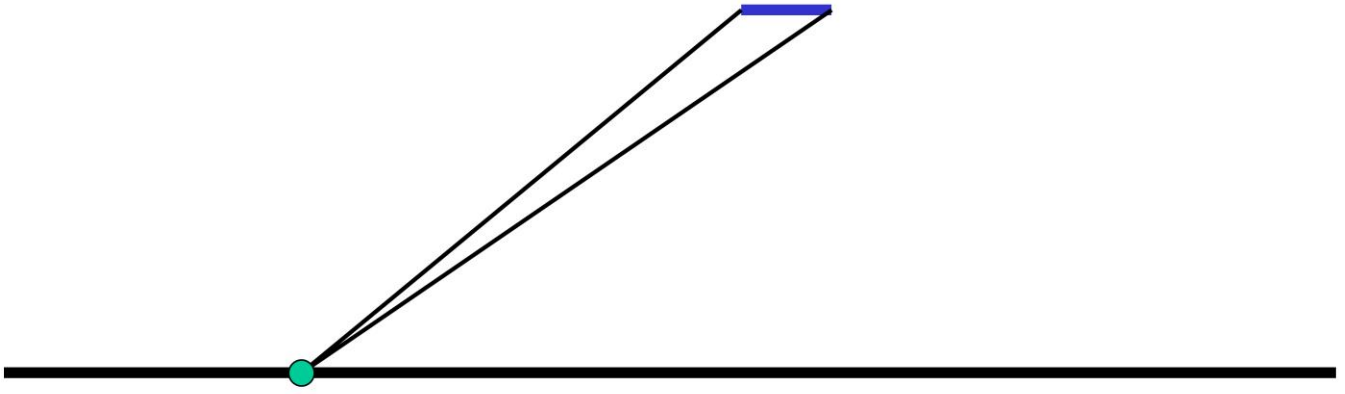
Angular Sampling Issues

- Large lights need to be sampled a lot



Angular Sampling Issues

- Small lights clearly less



Sampling Issues

- Large (low frequency) lights
  - Coarse spatial sampling
    - Not a lot of storage
  - Large solid angles
    - Run time integration would be expensive
- Small (high frequency) lights
  - Fine spatial sampling
    - High storage
  - Small solid angles
    - Run time integration isn't that bad

Sampling Issues: Transport

- Bounced light is pretty much always low frequency
  - An illuminated wall is an area light
- Do you need to use high frequency basis to model high frequency inter-reflections???
  - Similar to duality discussed in [Ramamoorthi2001], inter-reflections are implicitly large area lights

Objects

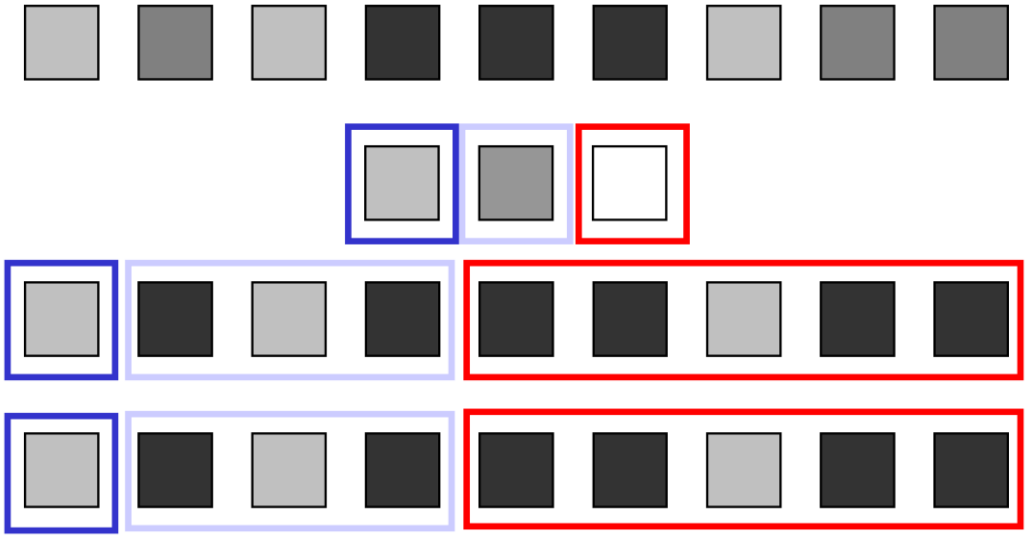
- Parameterize based on spherical basis functions
- Can rotate lighting basis for rigid objects (instead of rotating all of the transfer vectors)
- What about dynamic objects?
- What about meso-scale effects?

Local, Deformable Precomputed Radiance Transfer (LDPRT)

- Lighting coefficients and transfer vector must be in same coordinate frame
- For skinned character could rotate lighting at every bone
  - Rotating SH expensive
  - Might need more bones
- Use approximation to transfer vector that is easy to rotate

Local Deformable PRT

- Shading normal + coefficient per band



t_i

z_l

$Y_i\left(\vec{N}\right)$

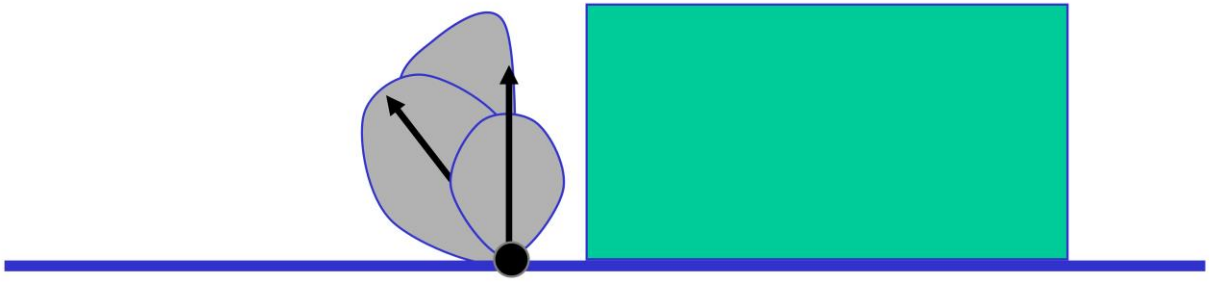
$z_l Y_i\left(\vec{N}\right)$

How Does LDPRT Work?

- Approximates transfer vector as a spherical function that has circular symmetry (around the shading normal) - zonal harmonic
- Just application of SH convolution theorem
- In between irradiance environment maps and PRT

Shading Normals

- Often used with ambient occlusion (sometimes called “bent normals”)



How to use LDPRT?

- Can be used as a replacement for normal maps (meso-scale)
  - No unique parameterization required
  - Works better for “local” surface properties
- Can be used instead of PRT
  - Less accurate transfer approximation
  - Shadows are based on “rest pose”

DEMO

Single Lobe LDPRT

LDPRT Multiple Lobes

- One lobe only can represent circularly symmetric functions
  - Works ok for some textures, not as well for others
- Fit multiple lobes to approximate transfer vector
  - More data/reconstruction costs
  - Submitted for publication

LDPRT Rendering

- **Simple Optimization:**

$$\sum_{l}\sum_{i\in l} z_l Y_i\left(\vec{N}\right)L_i$$

$$\sum_{l} z_l \boxed{\sum_{i\in l} Y_i\left(\vec{N}\right)L_i}$$

LDPRT

- Light specialized rendering
  - Complexity O(n) instead of $O(n^2)$
  - Fill a texture that contains dot(Y,L) for each band in each direction dynamically
    - Cube maps don't work well – interpolation issues between faces require too many texels
    - Lat/long seems to work well – 64x32 texture

Parameterized Models

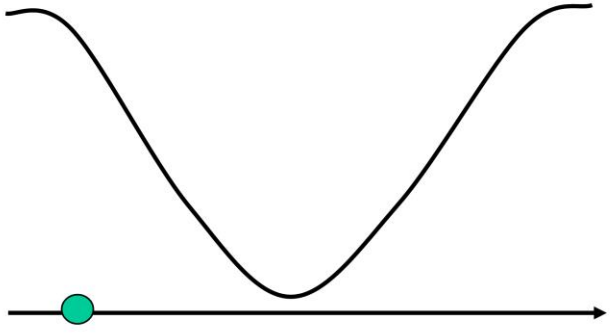
- Can just fit to results of PRT simulation
- Can build an ad-hoc model that has intuitive parameters (build transfer vector on the fly)
- Can fit an intuitive model to simulation data

Simple Translucency

- Single degree of freedom (DOF)
  - "Optical Thickness", how much light bleeds through in the negative normal direction
  - Could be based on subsurface scattering simulation, demo is ad-hoc

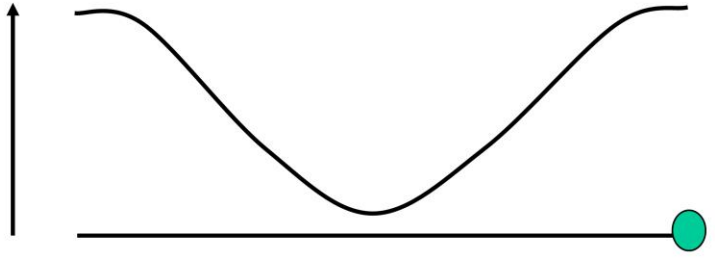
Wrinkle Model

- Two DOF
  - Phase, position along canonical wrinkle



Wrinkle Model

- Two DOF
  - Phase, position along canonical wrinkle
  - Amplitude, max magnitude of wrinkle



Fit

- Compute several simulations
  - 64 discrete amplitudes
  - 255 unique points in phase
- Fit 32x32 textures
  - DC,Linear (4 numbers) using linear least squares
  - 3 lobes (shading normal + 4 ZH coeffs), using non-linear least squares

DEMO

Multi-Lobe LDPRT Demo

Light flowing through space

- Lightmaps/PRT strictly deals with lighting response on surfaces
- What about in a volume?
- Needed to light objects moving through a scene that is modeled with transfer vectors

Representations for Lighting

- Plenoptic function [Adelson91]
  - $F(x,y,z,\theta,\phi,\lambda,t)$
    - x,y,z : Position in space
    - θ,ϕ : Direction
    - λ : Wavelength
    - t : Time
  - Common to use 3 wavelengths (RGB)
  - Time is a bit simplistic as the only means to model changes in illumination

Simple Representations

- Directional lights
  - Function of just 2 DOF (direction)
  - Only 1 direction on sphere
- Point lights
  - 3 DOF (location)
- Environment maps
  - True function of 2 DOF (any direction)
  - At infinity, so position doesn't matter

Irradiance Volumes

- Approximation of Plenoptic Function convolved with normalized cosine kernel
- Irradiance Volumes [Greger98]
  - Original paper used “diffuse cube maps”
  - SH used in later papers
  - Static lighting, diffuse objects (no transport) scene not effected by objects

SH Gradients

- "Differential" environment map at a point
- Gradients model change in radiance as a function of distance to point

SH Gradients

- Model “mid range” illumination by using a Taylor expansion (in space) of projection into SH [Annen2004]
- This is a really good idea – fairly inexpensive way to handle “local” lights
- Using N-gradient directions requires N times more work
  - 1 or 2 directional derivatives might make sense

DEMO

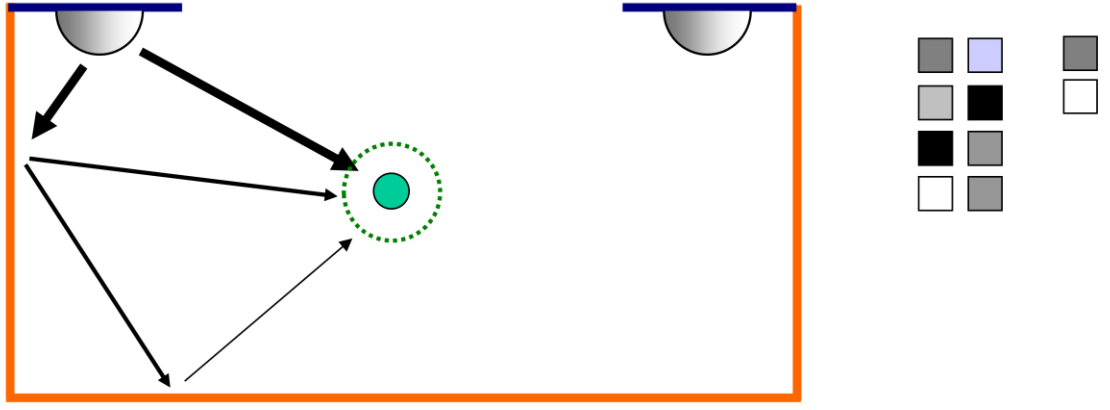
Irradiance Volumes + Gradients

Parameterized Radiance Volumes

- Extend Irradiance Volumes to handle dynamic lighting, render objects with PRT, hacks for how object effects lighting in scene
- Challenges
  - How to mix with SH Gradients
  - Compression

Parameterized Radiance Volumes

SH Light Probes from set of lights is a transfer matrix



Response in space is a SH Light Probe

PRV Representation

- Use smooth (differentiable) BF so you can easily generate gradients
- Multi-level uniform quadratic b-splines
  - Not very expensive at run time
  - C1 continuous (gradients behave better)
  - Multi-level enables more aggressive compression
- K-nearest neighbor and radial basis functions also worth investigating
  - Cost/continuity a concern

DEMO

Parameterized Radiance Volumes

Generating a PRV

- Compute transfer matrices at a moderate density in space
- Fit coarse b-spline volume to transfer matrices
  - Linear least squares
- Compute residual, threshold
- Fit finer samples to residuals

PRV Optimizations

- Only encode matrices in a single “slice”
  - Chest height in a game
  - Encode derivative out of slice explicitly
- For finer scales, only encode luminance
  - Kind of like image/video compression
  - Lower angular frequency for chroma?
- PCA transfer matrices
  - Trade off PCA decoding with less interpolation

Projected Shadows with PRV

- Vanilla PRT is “object” centric
- GI (light maps, etc.) model illumination in scene
- PRV ties scene lighting to dynamic objects
- How can objects influence scene?

Projected Shadows with PRV

- Focus on parameterized soft shadows
- Family of approximations
- “Correct” thing would be neighborhood transfer from [Sig02]
  - Expensive (transfer matrices in space)
  - Combining is difficult

Projected Shadows with PRV

- Function of distance to object that models how the given lighting environment would shadow a point in space
- Algorithms presented just produce a scalar – how much light should be attenuated for a given scene point

Projected Ambient Occlusion

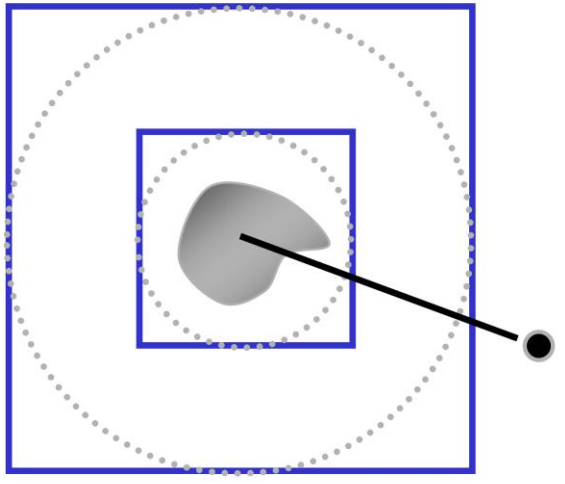
- For any point in space, what is the ambient occlusion
  - "Ambient Occlusion Fields" [Kontkanen2005]
- Linear function in $1/distance^2$
  - When distance is infinity, shadowing is zero
  - At a fixed distance (radius of bounding sphere) compute projection of visibility into first SH basis function (scalar cube map)

Projected Ambient Occlusion

- Pros:
  - Very simple and lightweight
  - Gives at least some cue for distance
- Cons:
  - Completely lighting independent
  - AO shadows are *extremely* soft

Projected Shadows with PRV

- Concentric shells (1 implicitly at infinity)
- Quadratic function of 1/distance squared



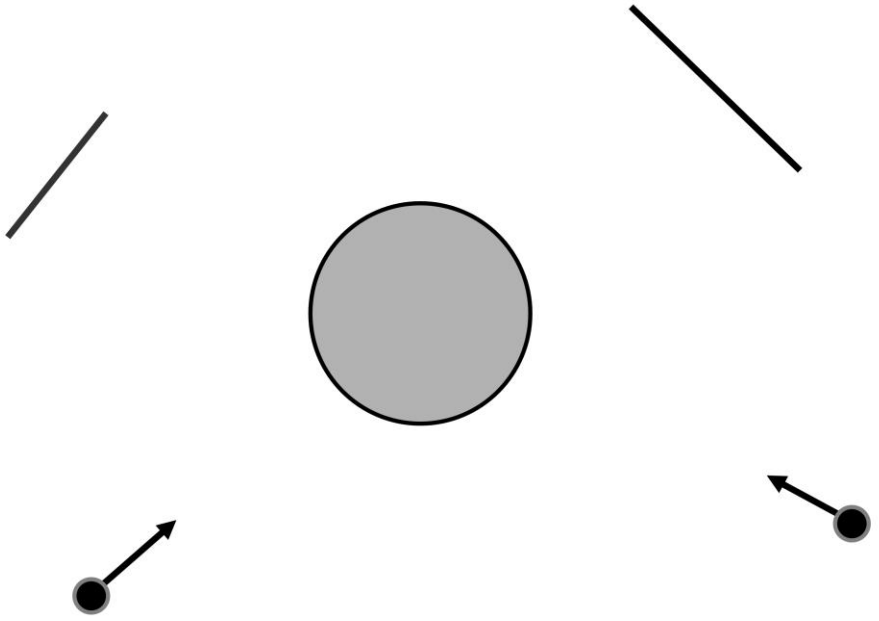
Projected Shadows with PRV

- Compute a transfer vector, normal direction pointing at center of object, at a fixed number of distances uniform in 1/distance squared
- Fit coefficients of quadratic polynomial
  - $Ax^2 + Bx + C$
  - At infinity (x = 0), transfer vector must have no shadowing, so only A and B are being fit
  - Rational polynomial in [Konktkanen2005] should be investigated as well

Projected Shadows with PRV

- Compress transfer vectors using PCA
  - For 4th order 16 PCA vectors works well (4 cube maps)
  - Could use less
- At a point in space compute shadowed irradiance (using compressed transfer vector) and divide by unshadowed irradiance

Projected Shadows Problem



Projected Shadows with PRV

- Compute approximation of maximum irradiance (max unshadowed)
- Consider unshadowed/max unshadowed
- Use Shadowed/max unshadowed
  - Only “bright” part of lighting environment will cast shadows

Projected Shadows with PRV

- If scene is more constrained (ie: race track) compute for small set of normals (vertical/horizontal) and just modulate
- Very approximate, but probably better than nothing
  - Care must be taken when mixing multiple objects

DEMO

Projected Shadows Demo

Conclusions

- Precomputed lighting techniques can be used in various scenarios
  - Normal mapping
  - Skylight models (direct + indirect)
  - Indirect lighting (from high frequency lights)
  - PRV to light object with scene
- Mix with “traditional” techniques
  - Direct lighting of “small” light sources
- Should always compress surface signals
  - Works for multiple light maps too

References - Techniques

- Experimental validation of analytical BRDF models, Ngan et. al. (SIGGRAPH 2004 Sketch)
- Normal distribution functions and multiple surfaces, Fournier (Graphics Interface 1992)
- From structure to reflectance, Fournier and Lalonde, in “Cloth Modeling and Animation” (AK Peters)
- Efficient rendering of spatial bi-directional reflectance distribution functions, McAllister et. al. (Graphics Hardware 2002)
- An efficient representation for irradiance environment maps, Ramamoorthi and Hanrahan (SIGGRAPH 2001)
- Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments, Sloan et. al. (SIGGRAPH 2002)
- The plenoptic function and the elements of early vision, Adelson and Bergen, in Computational Models of Visual Processing (MIT)
- Steerable illumination textures, Ashikhmin and Shirley (ToG)
- Polynomial Texture Maps, Malzbender et. al. (SIGGRAPH 2001)
- Interactive subsurface scattering for translucent meshes, Hao et. al. (Si3D 2003)
- Clustered principal components for precomputed radiance transfer, Sloan et. al. (SIGGRAPH 2003)
- All-frequency precomputed radiance transfer for glossy objects, Liu et. al. (EGSR 2004)
- Triple product wavelet integrals for all-frequency relighting, Ng et. al. (SIGGRAPH 2004)
- All-frequency shadows using non-linear wavelet lighting approximation, Ng et. al. (SIGGRAPH 2003)
- The irradiance volume, Greger et. al. (IEEE CG&A Mar.98)
- Spherical harmonic gradients for mid-range illumination, Annen et. al. (EGSR 2004)
- Ambient occlusion fields, Kontkanen and Laine (Si3D 2005)
- Normal Distribution Mapping, Olano and North (Tech Report, 1996)

Acknowledgements

- Shanon Drone, John Rapp, Jason Sandlin and John Steed for demo help
- Paul Debevec for light probes
- Ben Luna and John Snyder for work on LDPRT