Computer Graphics 13 - Global illumination 2

Tom Thorne

Slides courtesy of Taku Komura
www.inf.ed.ac.uk/teaching/courses/cg

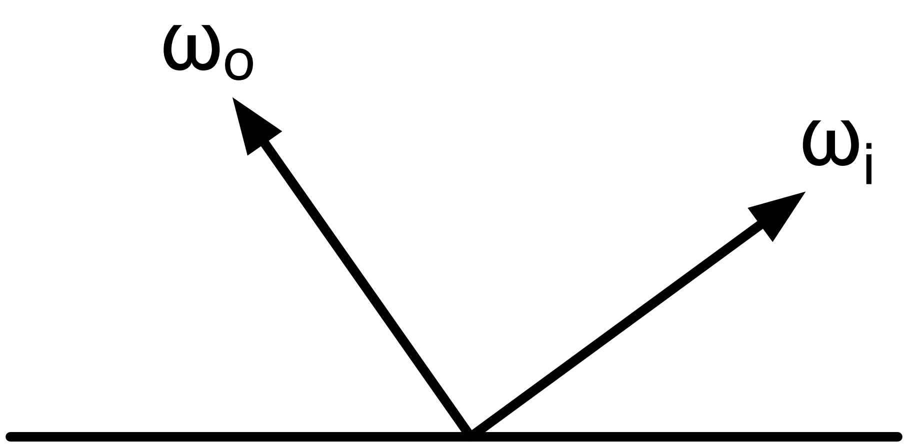
Overview

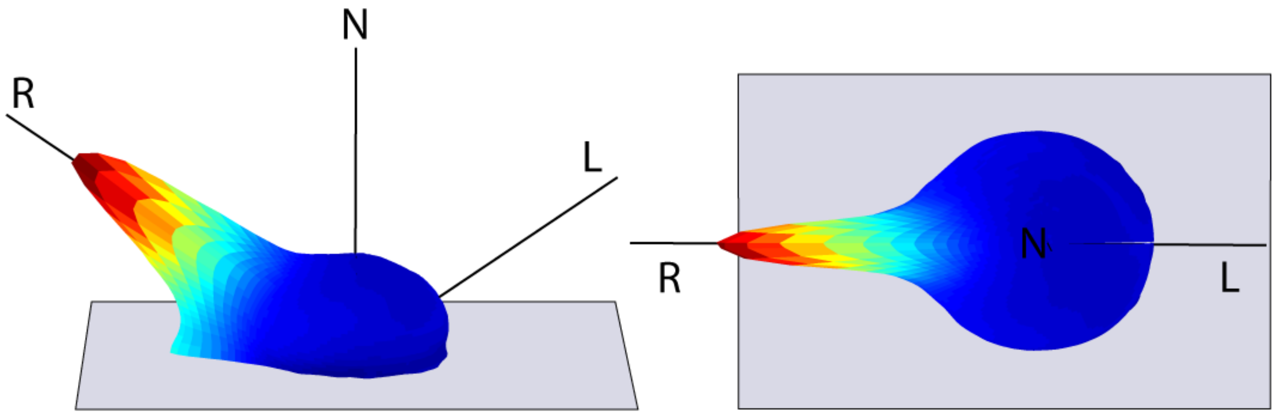
- **BRDFs**
- **Ambient Occlusion**
- **Spherical Harmonic lighting**
- Radiosity

BRDFs

- Bidirectional reflectance distribution function
- A function of the incident and outgoing angles
- Can be dependent on position

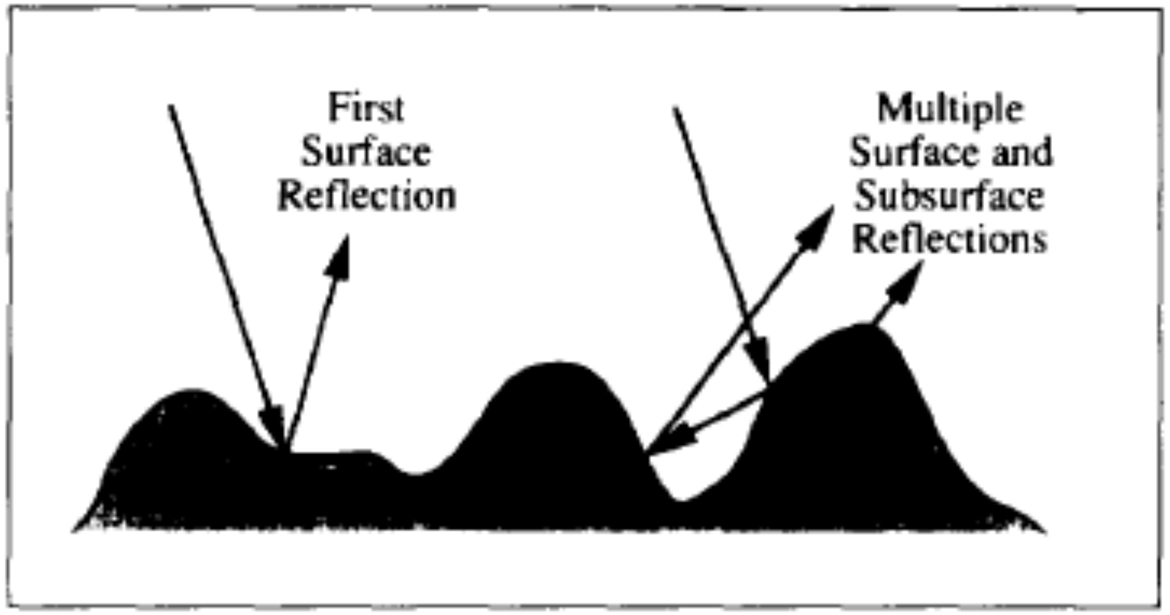
$$f(x, \omega_o, \omega_i) = \frac{dL(\omega_o)}{dE(\omega_i)}$$





BRDFs

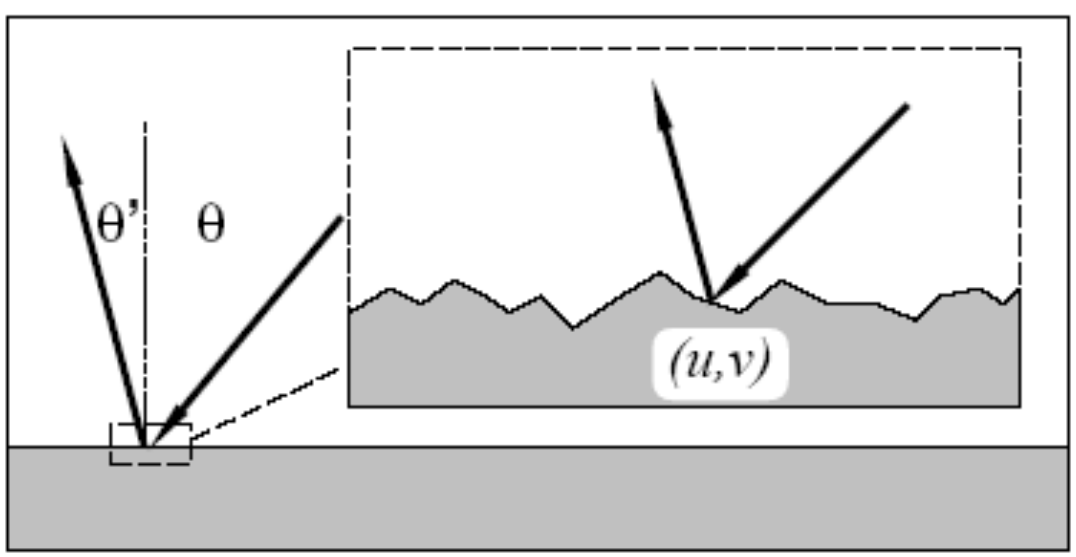
- Affected by the way light reflects on the surface
- Smoothness of surface:
  - Single reflection from a smooth surface - specular
  - Multiple reflections - diffuse
- Shadowing effect

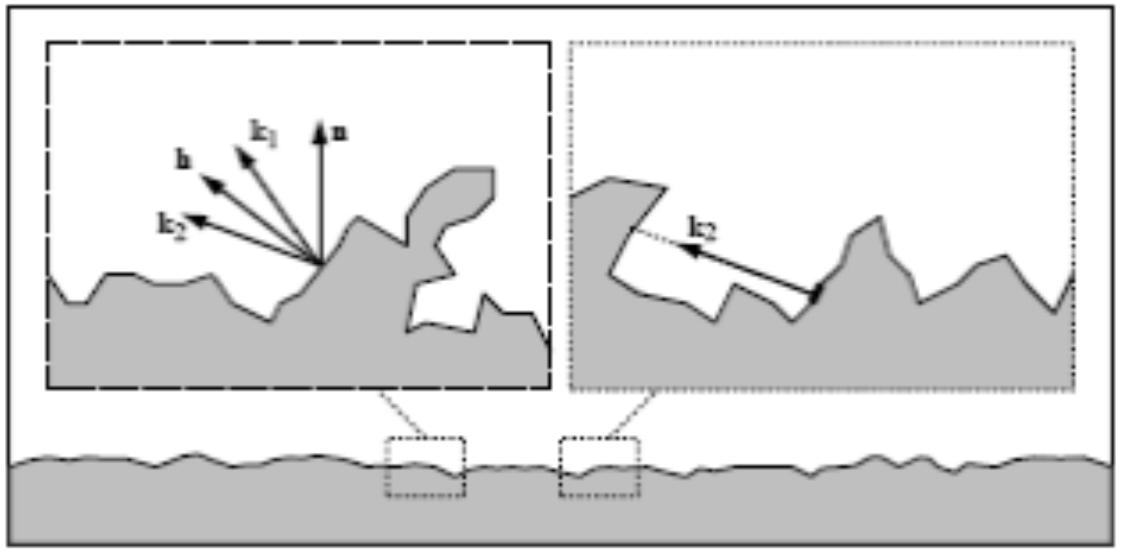


Microfacet theory

- [Torrance & Sparrow 1967]
  - Surface modeled by tiny mirrors
  - Value of BRDF at ω_i ω_o
    - # of mirrors oriented halfway between ω_o and ω_i
    where ω_i is the incoming direction, ω_o is the out going direction

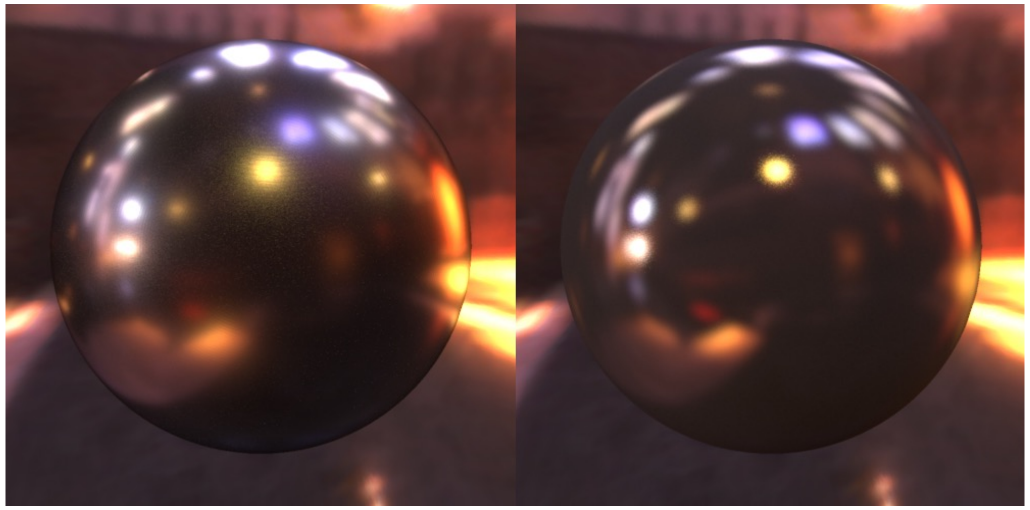
    - Modulated by Fresnel, shadowing/masking

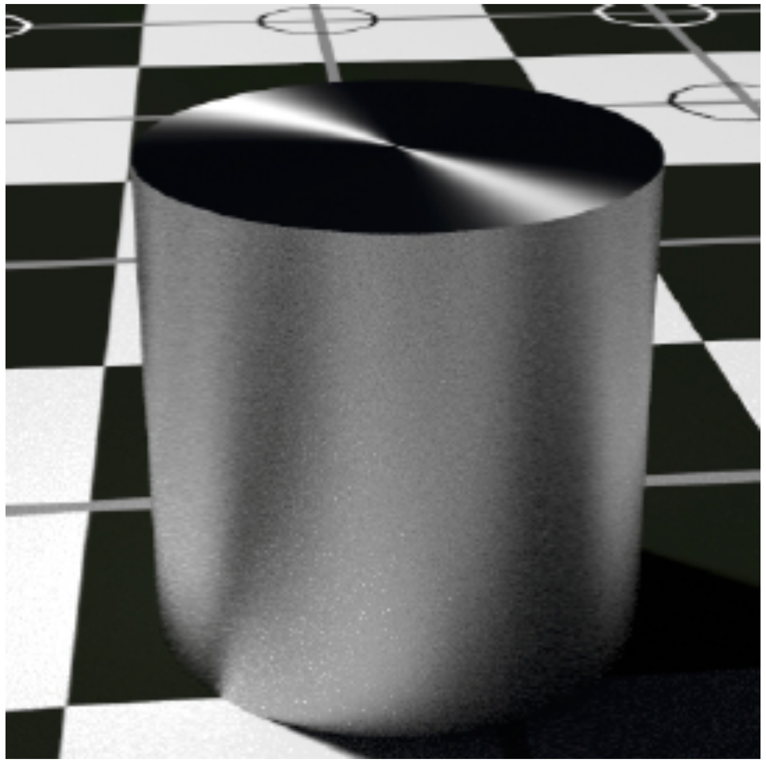




Isotropic and anisotropic BRDFs

- Isotropic:
  - Lighting depends only on angle with normal and relative angle around normal
- Anisotropic:
  - Brushed metal
  - Can't be modelled using diffuse and specular reflection





Examples

- Ward

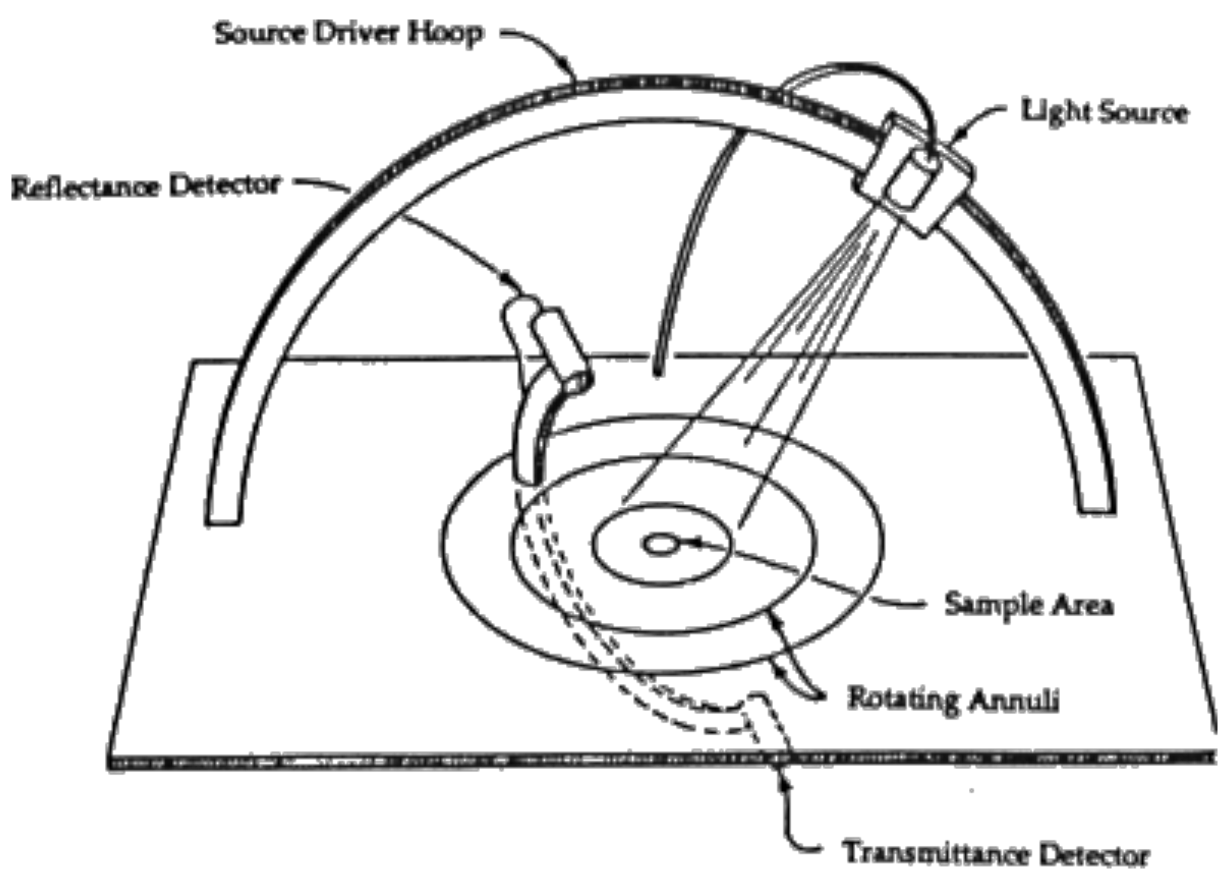
$$f(\omega_o, \omega_i) = \frac{\rho_s}{4\pi\alpha_x\alpha_y\sqrt{\cos\theta_i \cos\theta_o}} e^{\tan^2\theta_h \left(\frac{\cos^2\phi_h}{\alpha_x^2} + \frac{\sin^2\phi_h}{\alpha_y^2}\right)}$$

- Blinn-Phong:

$$f(\omega_o, \omega_i) = \frac{k_L}{\pi} + k_G \frac{8+s}{8\pi}(h \cdot n)^s$$
$$h = \frac{\omega_i + \omega_o}{2}$$

Capturing BRDFs

- Measured using a device called gonioreflectometer
  - Casting light from various directions onto the object, and capturing the light reflected back
- Problems:
  - Takes a long time and produces a huge amount of data (many GB)
  - Introduces errors

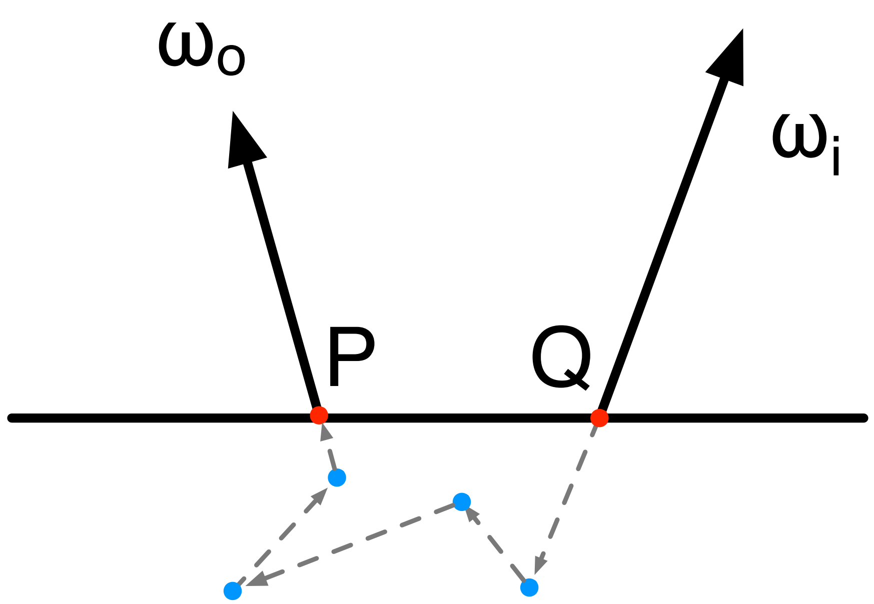


BSSRDF

- Bidirectional surface scattering reflection distribution function

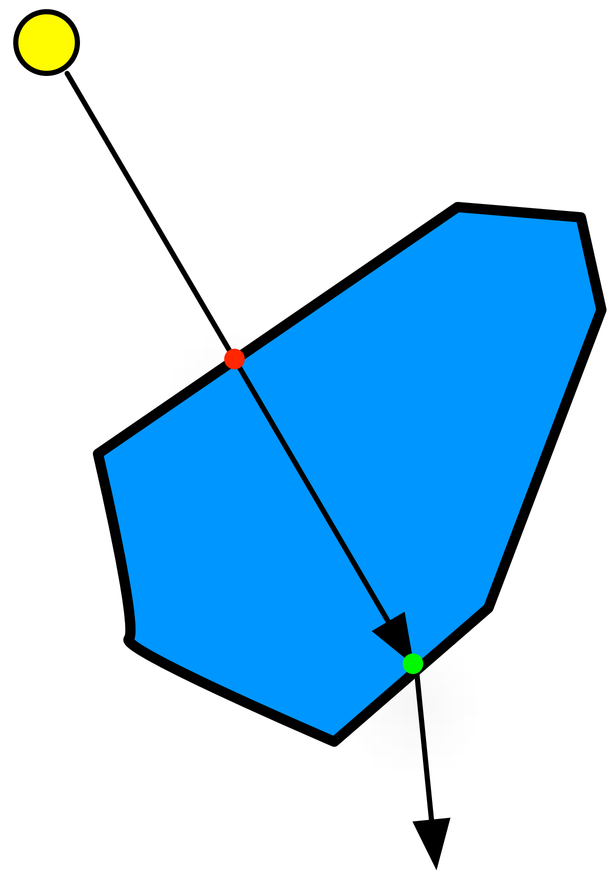
- Some surfaces exhibit subsurface scattering over a long range

$$f(P, Q, \omega_o, \omega_i) = \frac{dL(P, \omega_o)}{dE(Q, \omega_i)}$$



Rendering subsurface scattering

- For short distances, approximate with a BRDF
- Normal blurring
  - Use a bump map for specular reflection, but an unperturbed or blurred normal for diffuse shading
- Texture space diffusion
  - Render with diffuse lighting into a texture
  - Blur the texture and use for diffuse term in shading
- Depth map
  - Look up distance to surface in shadow map

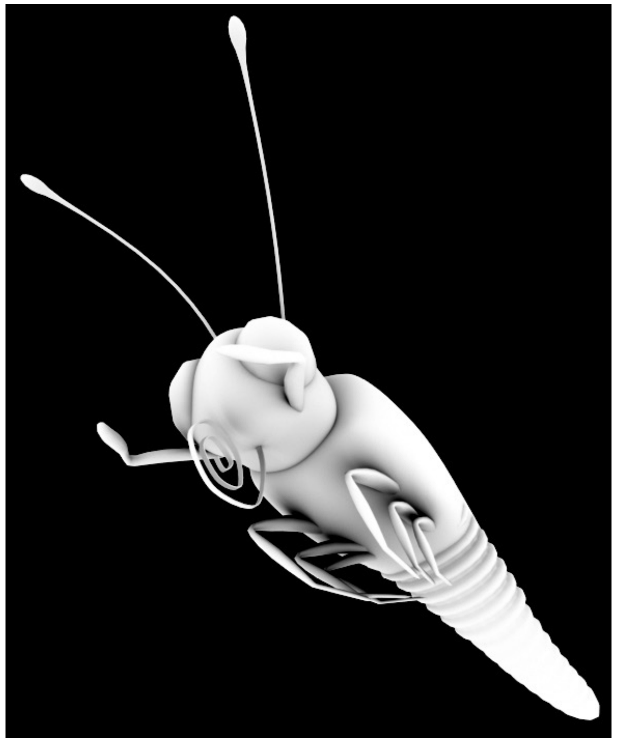


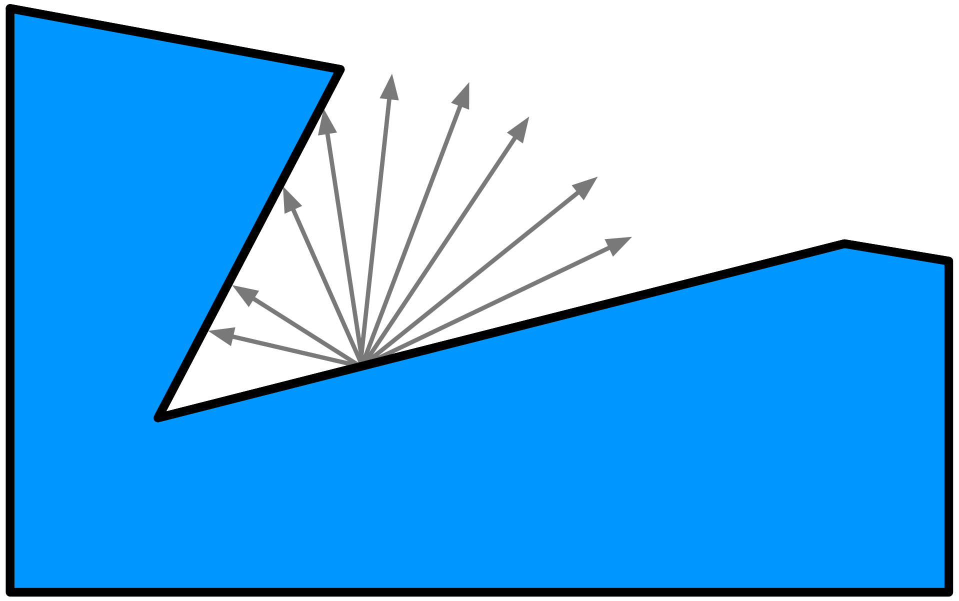
Ambient occlusion

- Shadowing of ambient light
- Incoming light is constant ambient lighting L from every direction, taking into account visibility function v(x,ω) at surface

$$E(x,n) = L_A \int_\Omega v(x,\omega_i)(n \cdot \omega_i) d\omega_i$$

$$k_A(x) = \frac{1}{\pi} \int_\Omega v(x,\omega)(n \cdot \omega_i) d\omega_i$$





Ambient occlusion

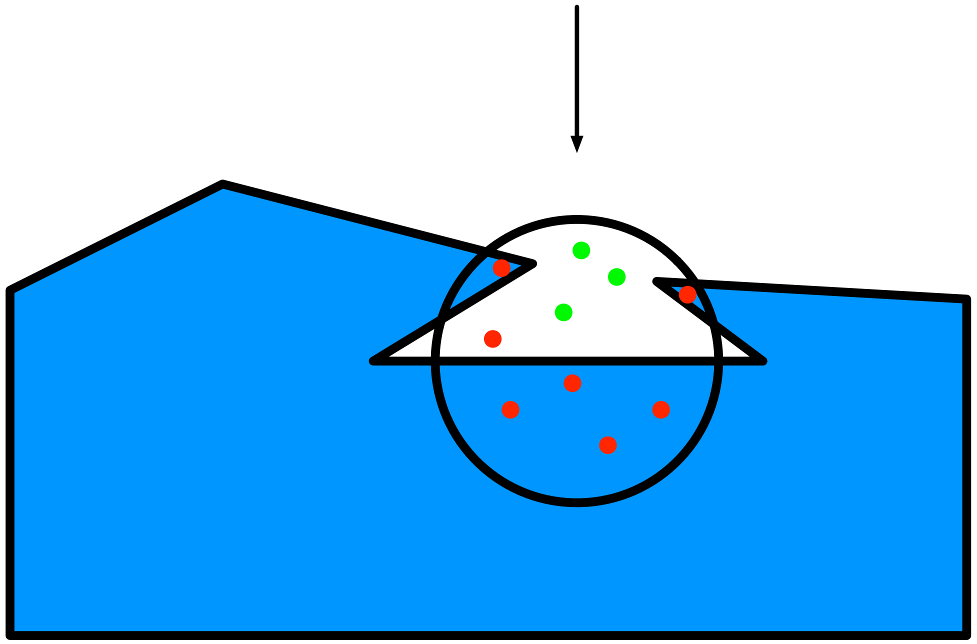
- This approach works well for discrete objects, what about closed rooms? Replace visibility function with a distance function.
- To combine with an environment map, we can apply normal bending:

$$n' = \frac{\int_\Omega v(\omega_i)\omega_i \cos\theta_i d\omega_i}{\|\int_\Omega v(\omega_i)\omega_i \cos\theta_i d\omega_i\|}$$

- Need to recalculate for non-static objects

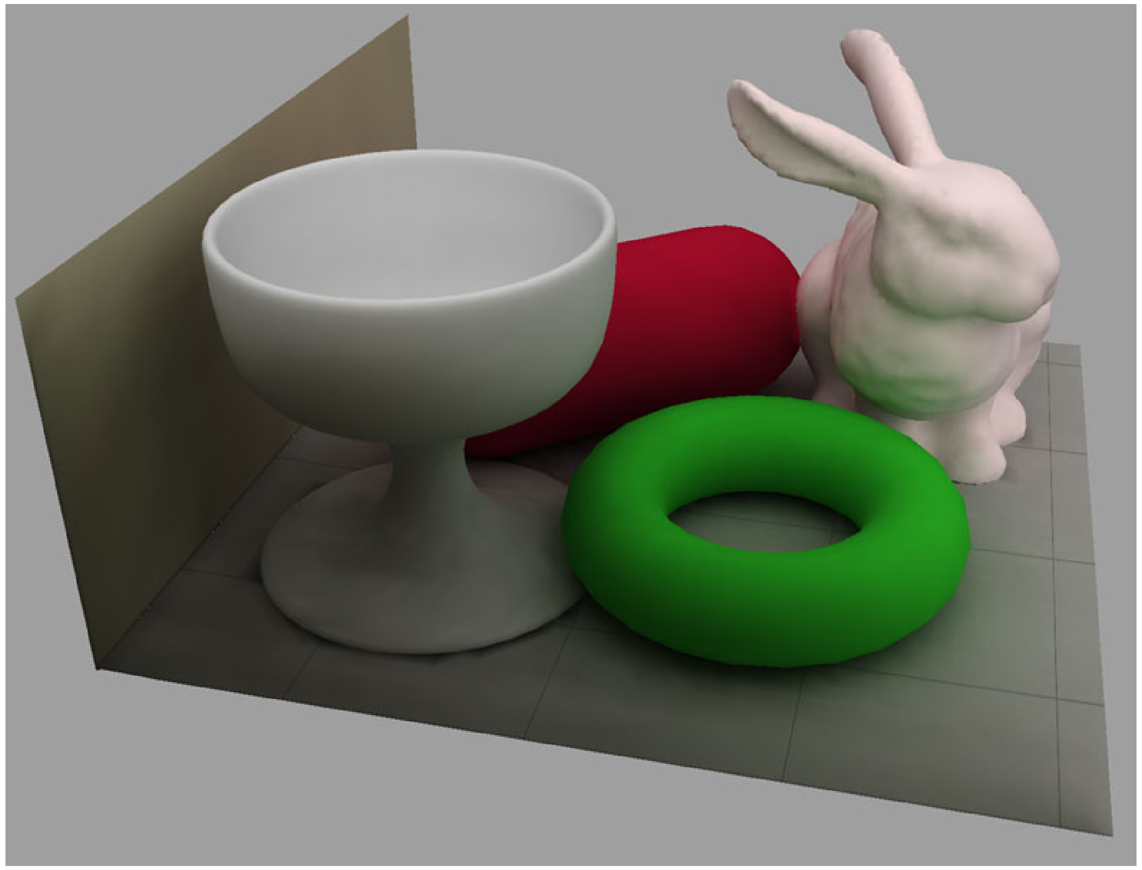
Screen space ambient occlusion

- Object space dynamic ambient occlusion is expensive
- Screen space ambient occlusion takes advantage of screen space depth buffer and surface normal information.
  - Sample points in a sphere around each pixel
  - Use fraction that are closer than Z-buffer depth value to estimate occlusion
  - Independent of object complexity

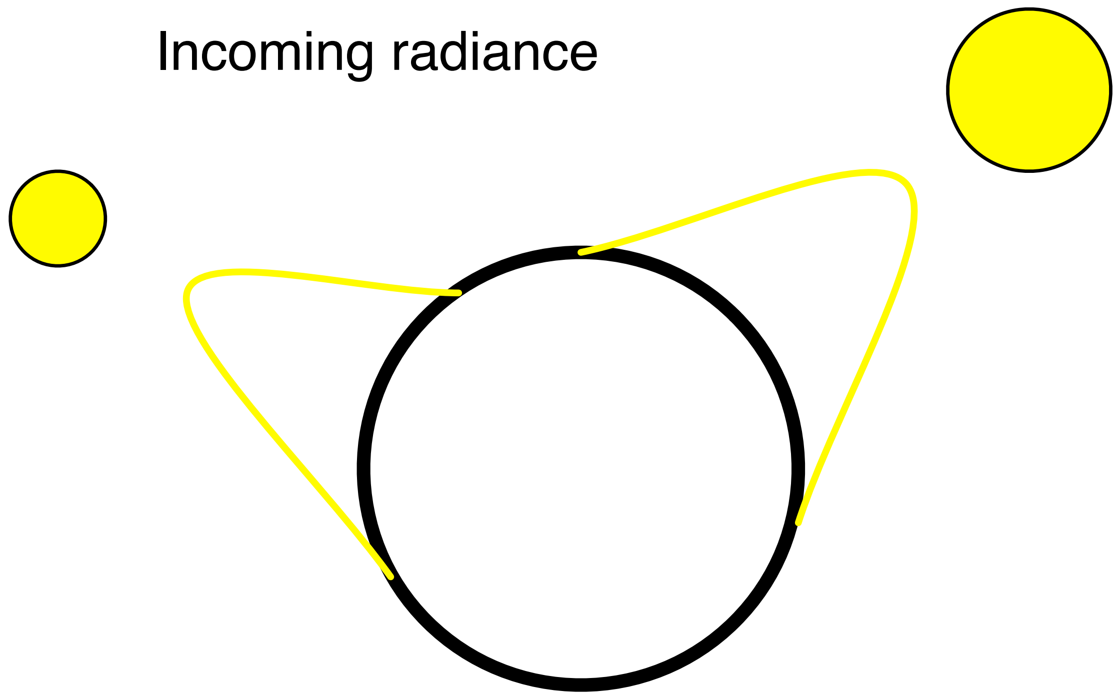


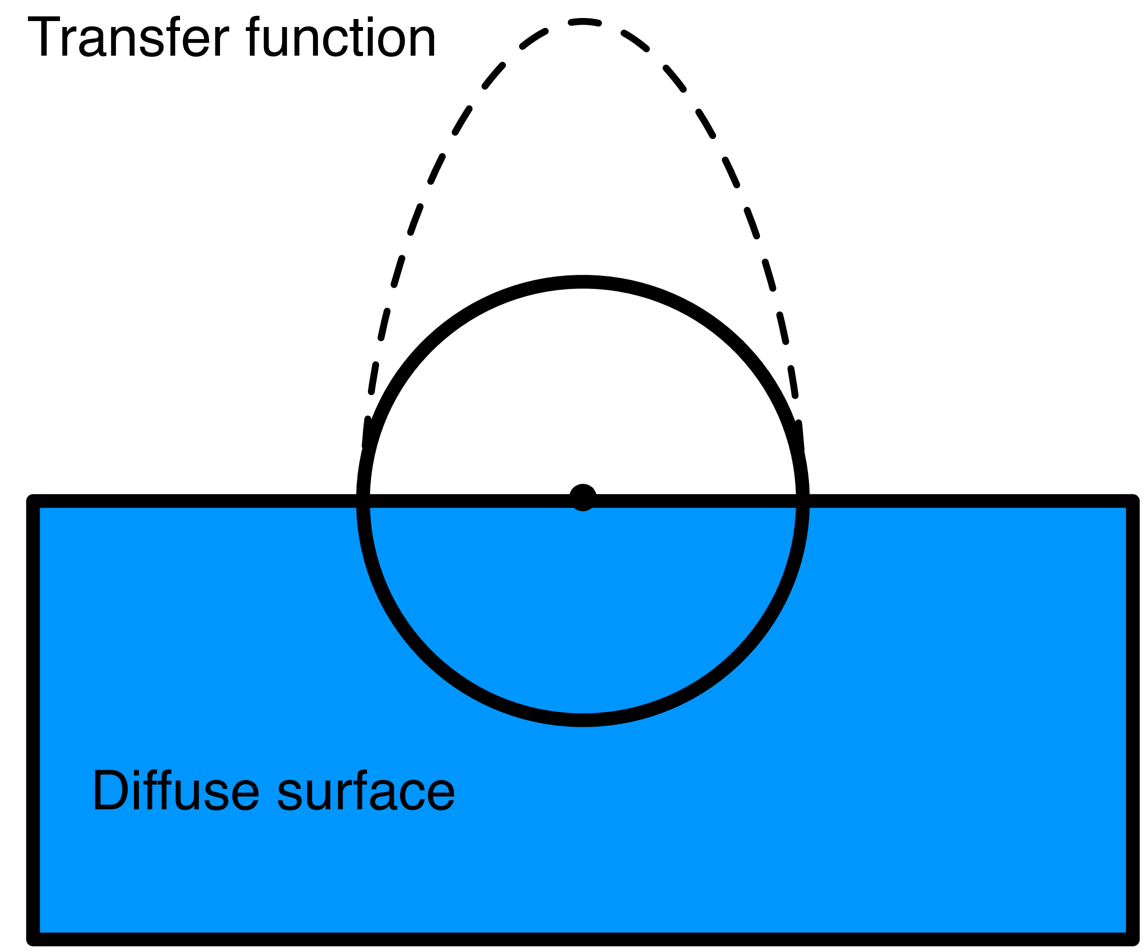
Spherical harmonic lighting

- Pre-computed radiance transfer (PRT) method
- Independent of view point, works with arbitrary lighting conditions

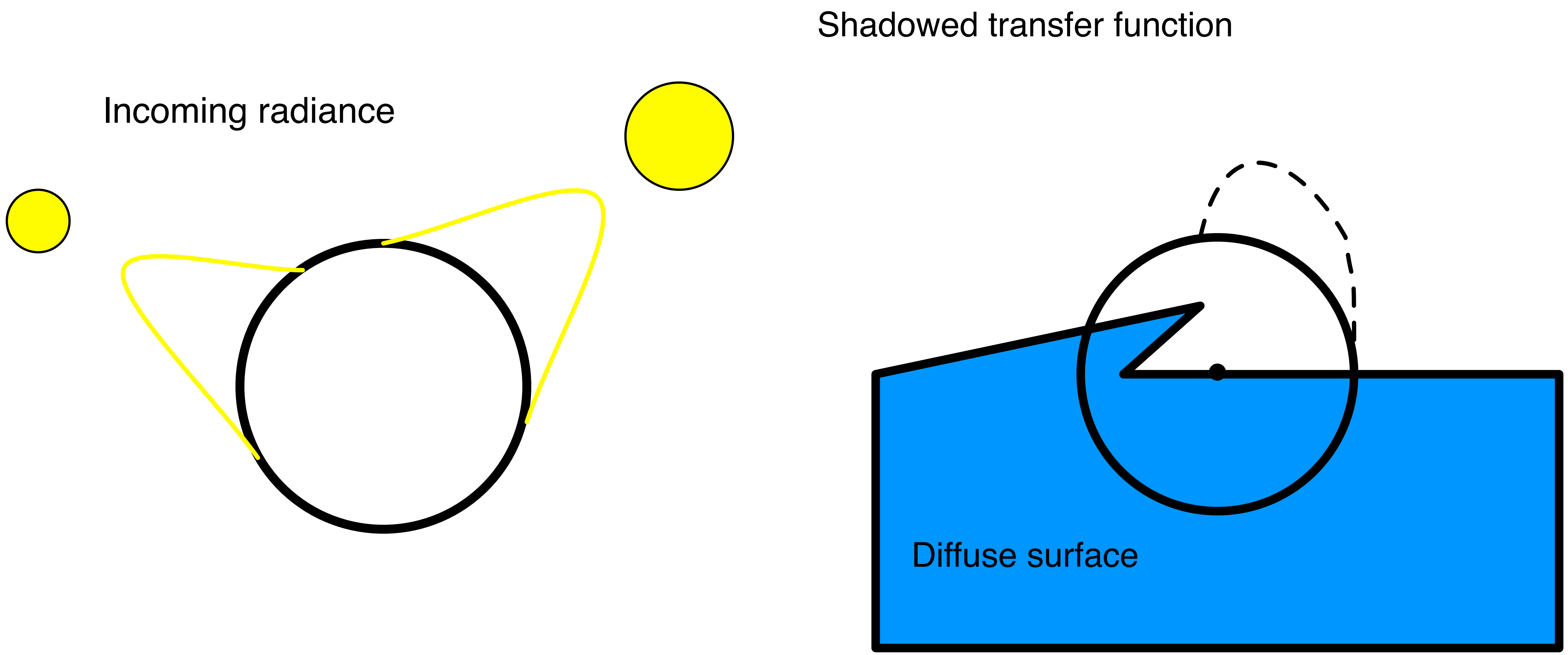


Overview

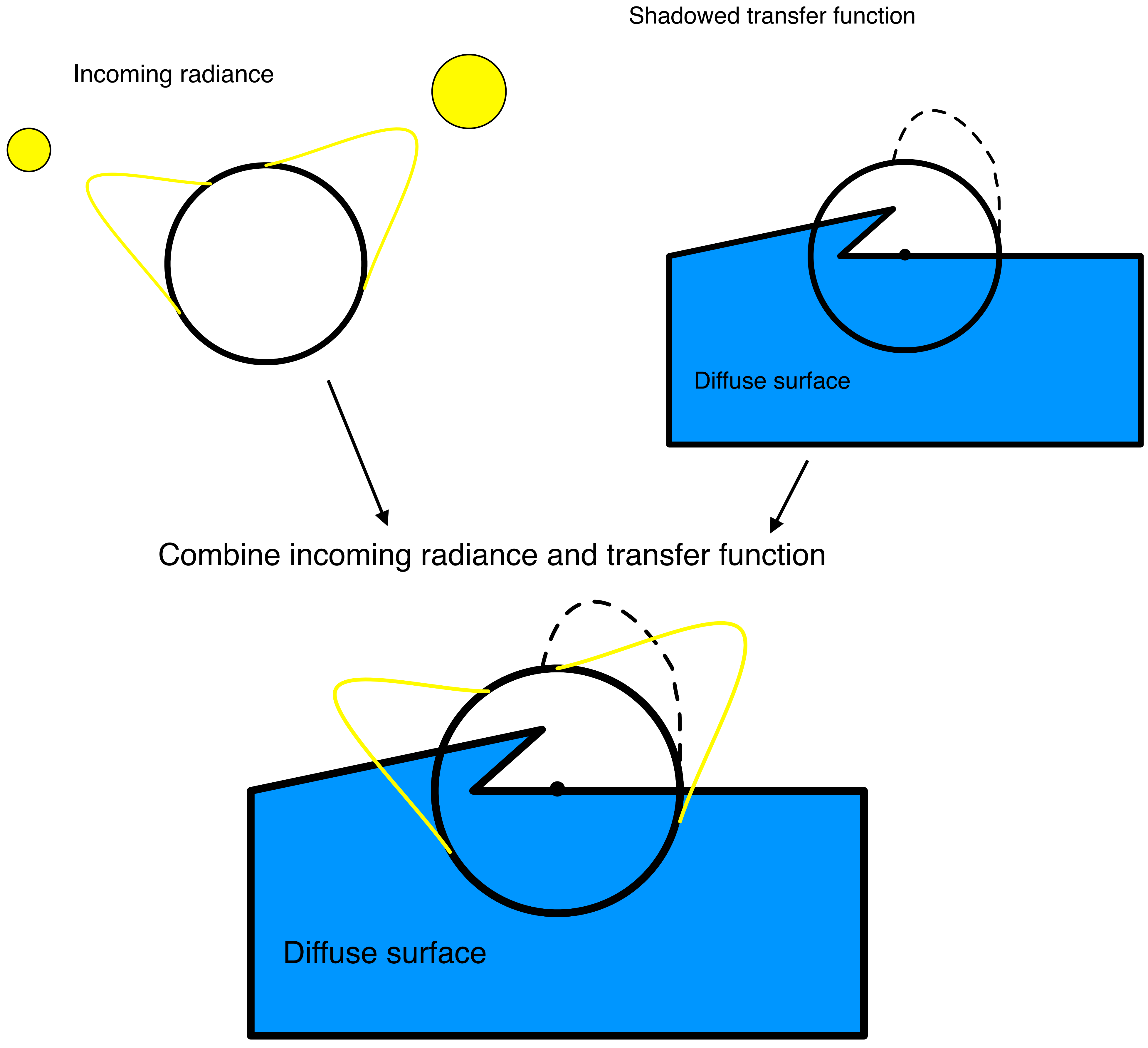




Overview

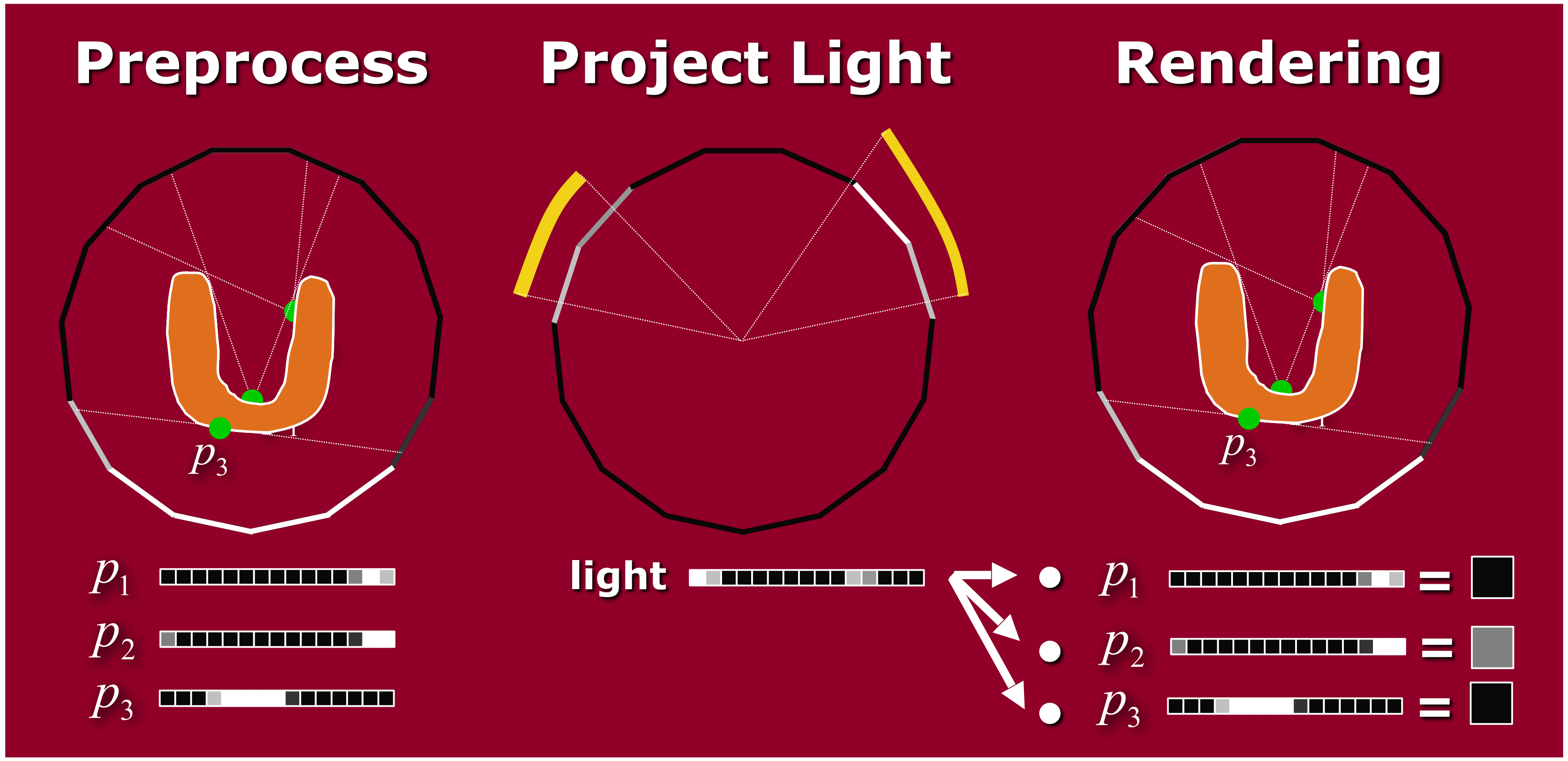

Incoming radiance
Shadowed transfer function
Diffuse surface

Overview


Shadowed transfer function
Incoming radiance
Diffuse surface
Combine incoming radiance and transfer function
Diffuse surface

Diffuse light transfer

- 2D example, incoming directions form a circle. Only considering shadowing:

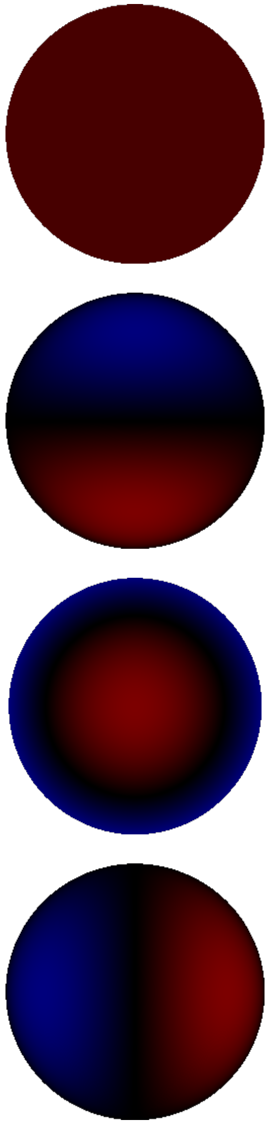


Spherical harmonics

- Used as a basis to represent functions defined over a sphere
- Properties:
  - Simple rotation and convolution

$$-l \leq m \leq l$$

$$h_l^m(\theta, \phi) = \begin{cases} \sqrt{2} K_l^m \cos(m\phi) P_l^m(\cos\theta), & m > 0 \\ \sqrt{2} K_l^m \sin(-m\phi) P_l^{-m}(\cos\theta), & m < 0 \\ K_l^0 P_l^0(\cos\theta), & m = 0 \end{cases}$$

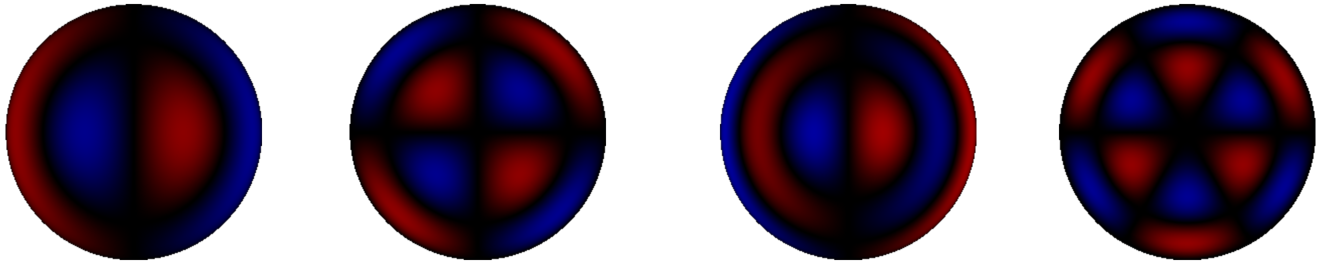


l=0 m=0

l=1 m=-1

l=1 m=0

l=1 m=1



l=2 m=1 l=3 m=-1 l=3 m=2 l=4 m=-2

Spherical harmonic lighting

- Diffuse lighting (constant for every outgoing direction) based on incoming light and transfer function

$$\int_\Omega L_i(\omega)t(\omega)d\omega$$

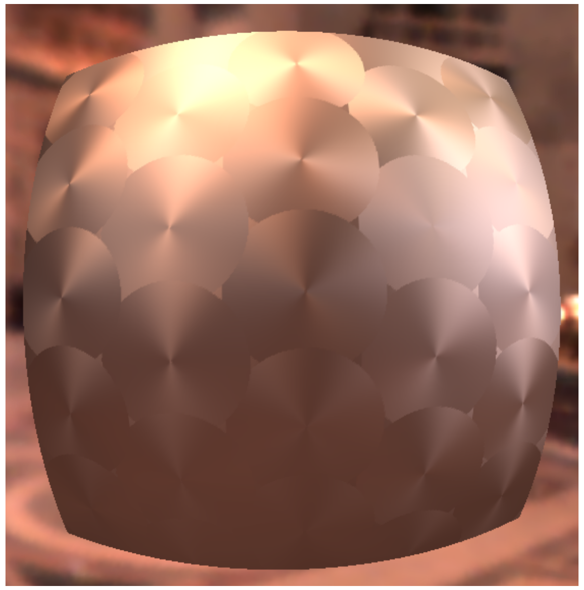
- Calculate integral by multiplying SH coefficients

$$\int_\Omega L_i(\omega)t(\omega)d\omega = \sum_j^\infty c_{L,j}c_{t,j}$$

- Approximation based on a finite number of coefficients

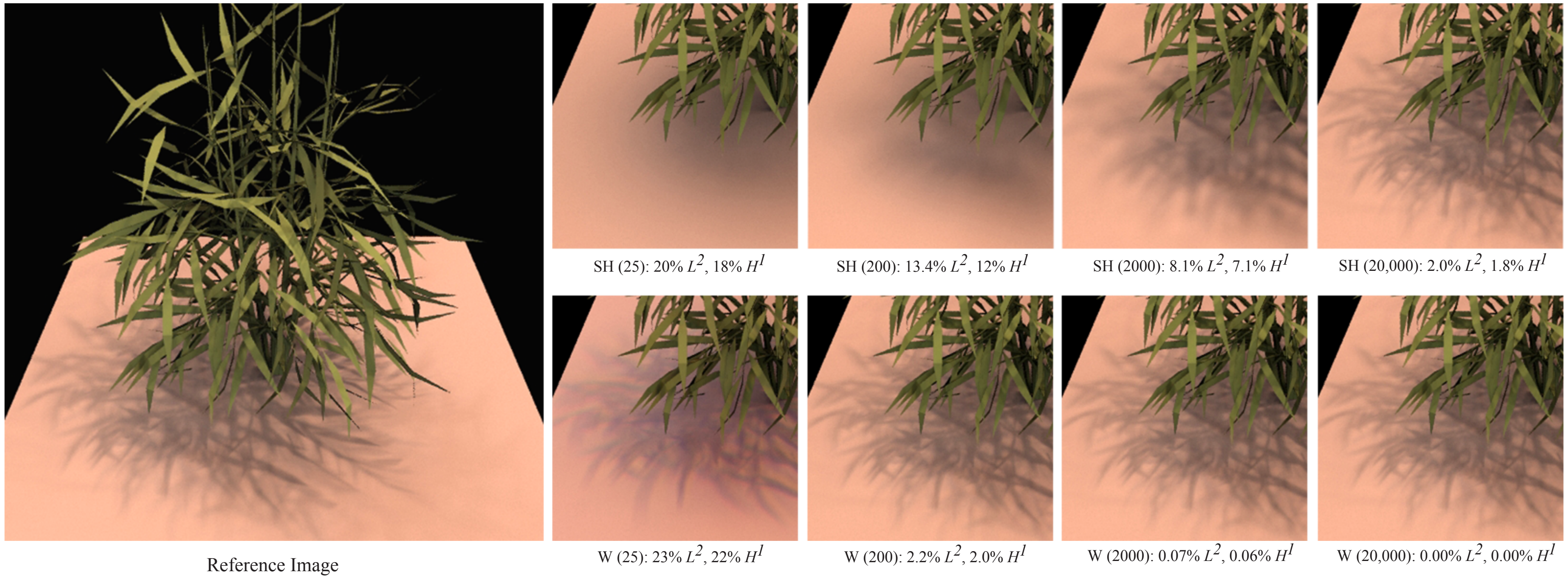
Examples

Shadows | 1 bounce interreflections | 2 bounce interreflections



Problems

- Only valid for static objects
- To represent sharp changes (e.g. shadows) or high frequency lights we need many spherical harmonic coefficients
  - An alternative approach - Haar wavelets

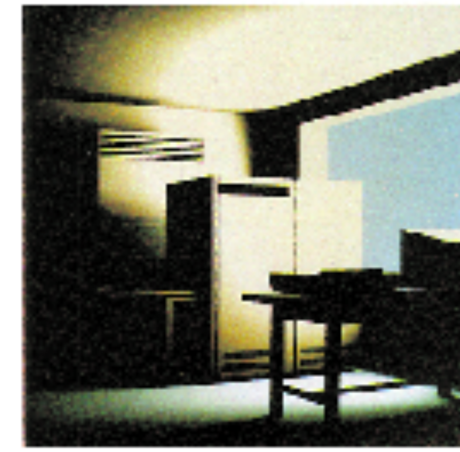


Overview

- BRDFs
- Ambient Occlusion
- Spherical Harmonic lighting
- **Radiosity**

Radiosity rendering

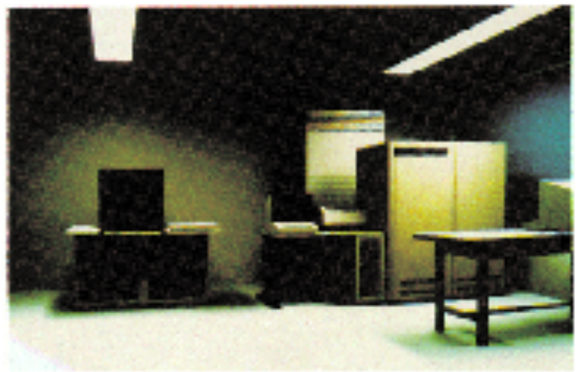
- Based on a method developed by researchers in heat transfer in 1950s
- Applied to computer graphics in the mid 1980s
- by Michael Cohen Tomoyuki Nishita



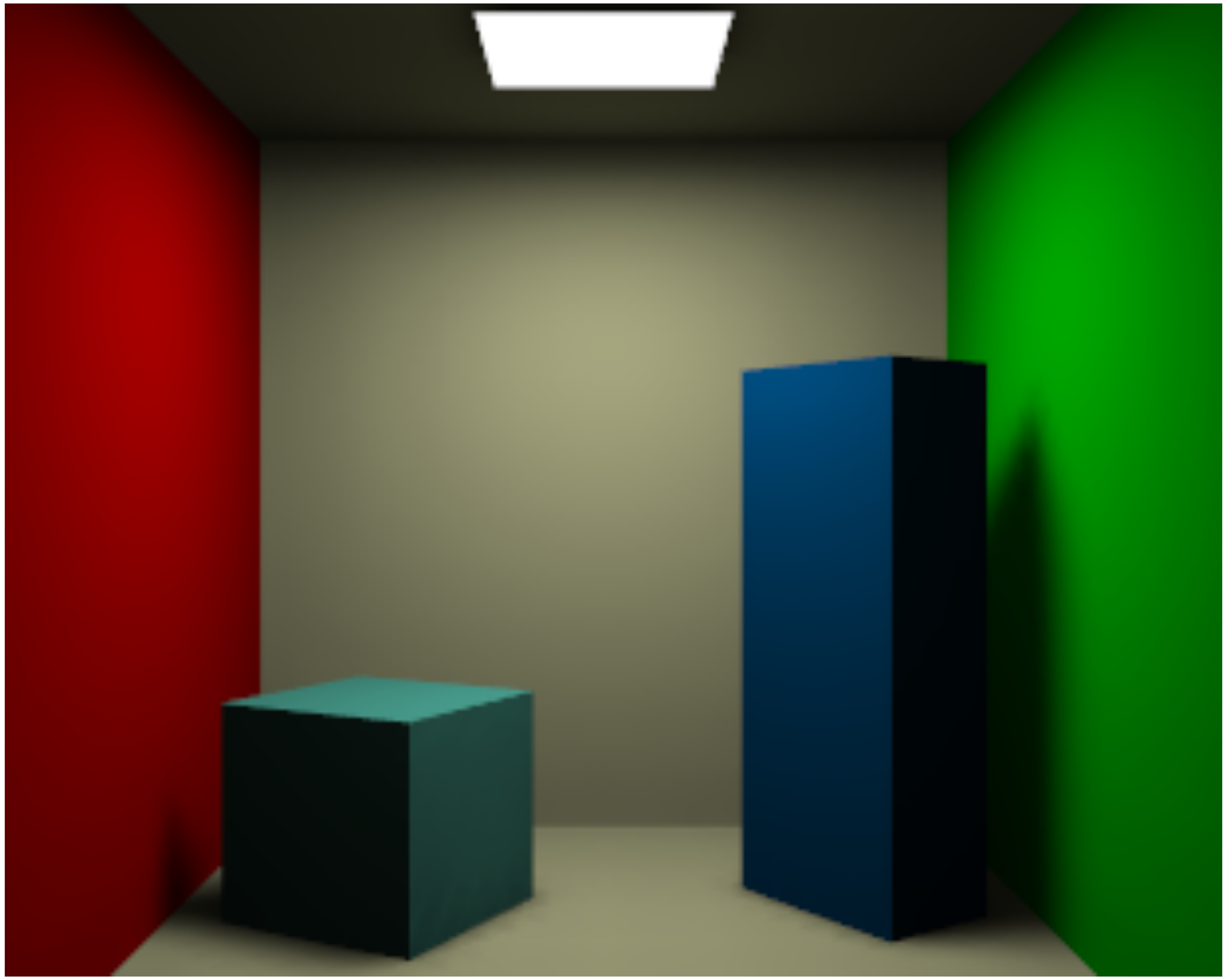
Radiosity rendering

- Can simulate inter-surface reflection
- Can produce nice ambient effects
- Can simulate effects such as
  - Soft shadows,
  - color bleeding
- Can only handle diffuse colour
- $\rightarrow$ need to be combined with ray-tracing to handle specular light

(a)

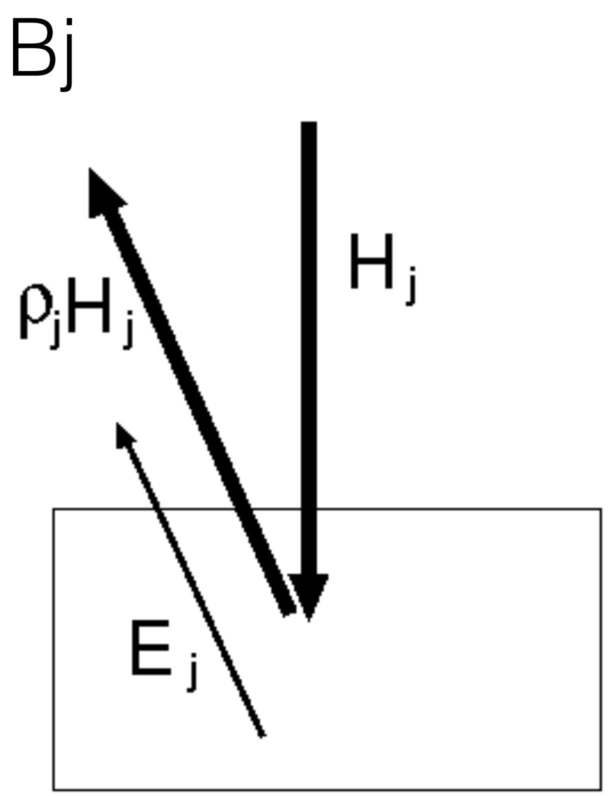


Colour bleeding and soft shadows



The Radiosity model

- At each surface in a model the amount of energy that is given off (Radiosity) is comprised of
  - the energy that the surface emits internally (E), plus
  - the amount of energy that is reflected off the surface (ρH)



$$B_j = \rho_j H_j + E_j$$

B_j is the radiosity of surface j,

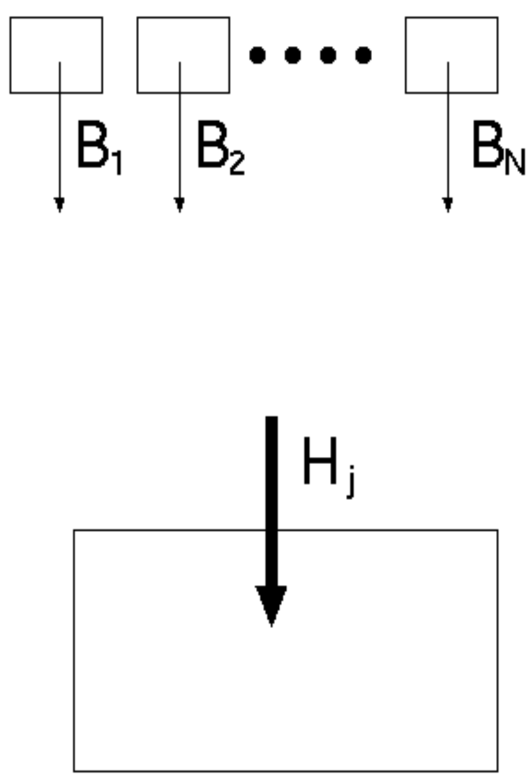
ρ_j is the reflectivity of surface j,

E_j is the energy emitted by surface j.

H_j is the energy incident on surface j

The Radiosity model

- The amount of incident light hitting the surface can be found by summing for all other surfaces the amount of energy that they contribute to this surface



$$H_j = \sum_{i=1}^{N} B_i F_{i,j}$$

$F_{i,j}$ Form factor

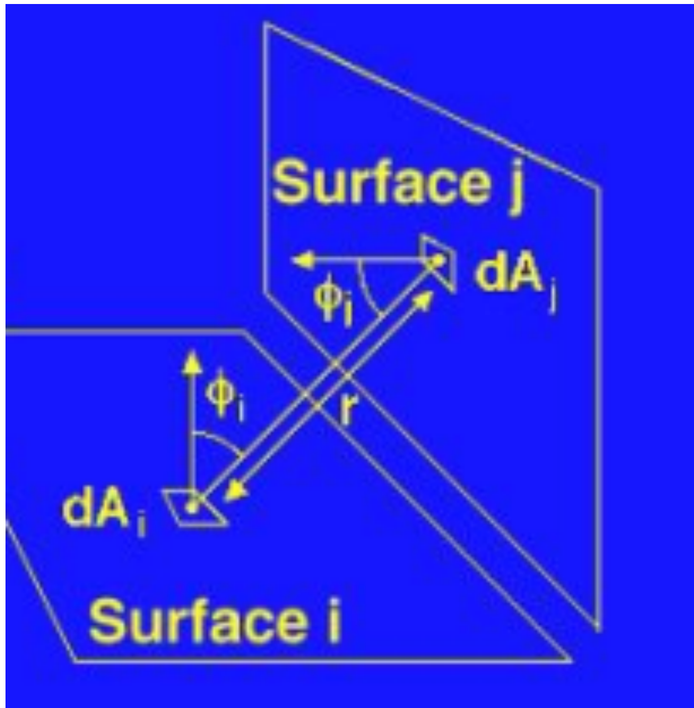
Form factors

- The fraction of energy that leaves surface i and lands on surface j
- Between differential areas, it is

$$\frac{\cos\phi_i \cos\phi_j}{\pi |r|^2} dA_i dA_j$$

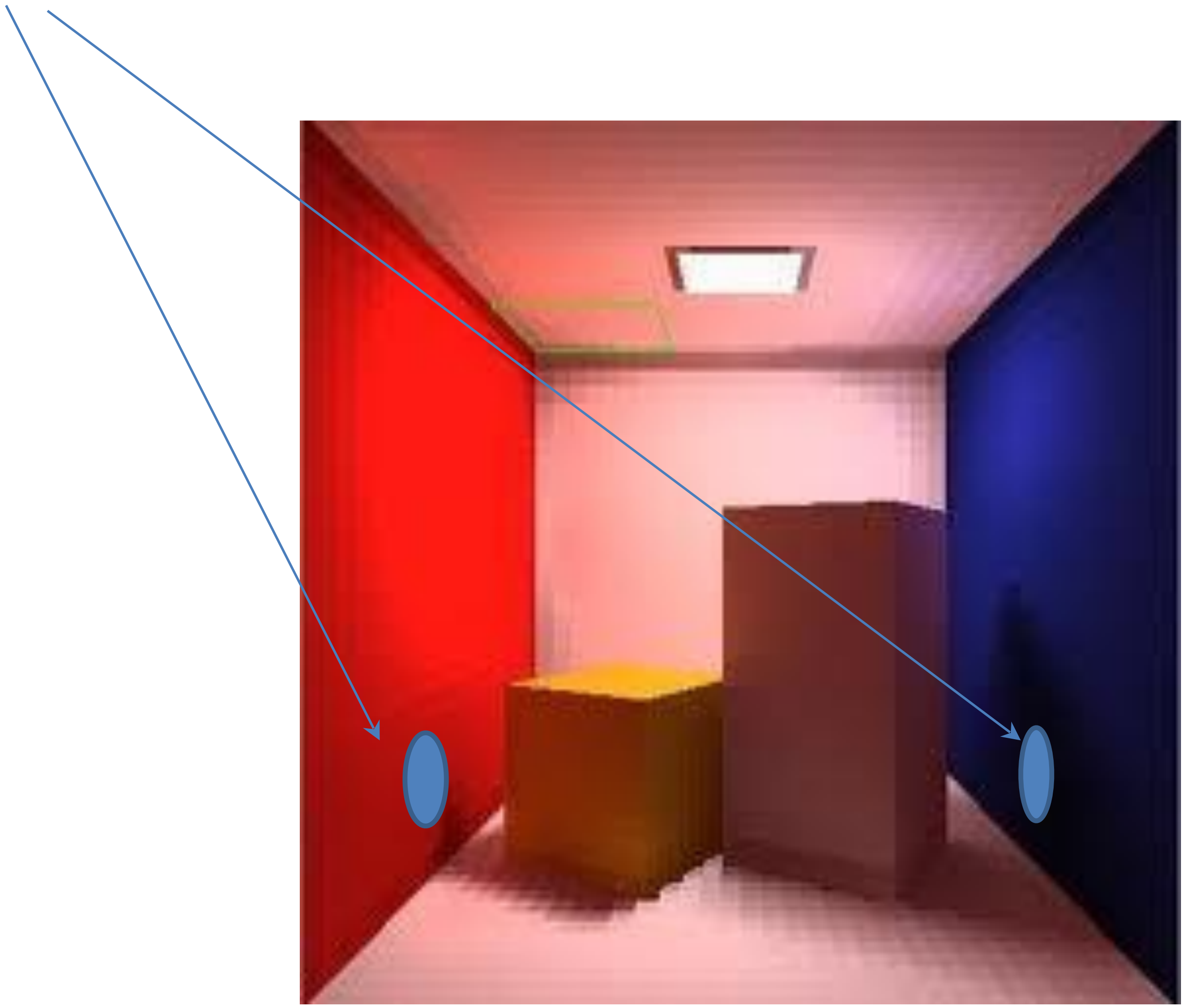
- The overall form factor between i and j is

$$F_{ij} = \sum_i \sum_j \frac{\cos\phi_i \cos\phi_j}{\pi |r|^2} dA_i dA_j$$



Form factors

- Also need to take into account occlusions
- The form factor for those faces which are hidden from each other must be zero



Radiosity matrix

- The radiosity equation now looks like this:

$$B_j = E_j + \rho_j \sum_{i=1}^{N} B_i F_{i,j}$$

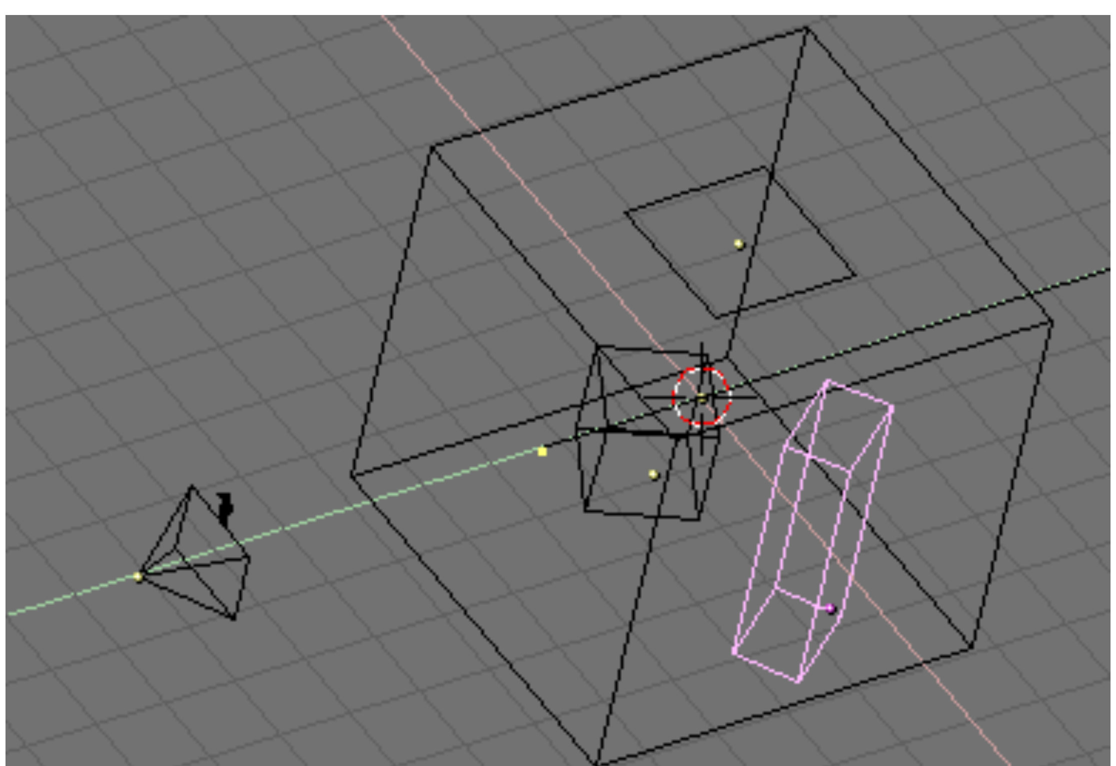
- The derived radiosity equations form a set of N linear equations in N unknowns. This leads nicely to a matrix solution:

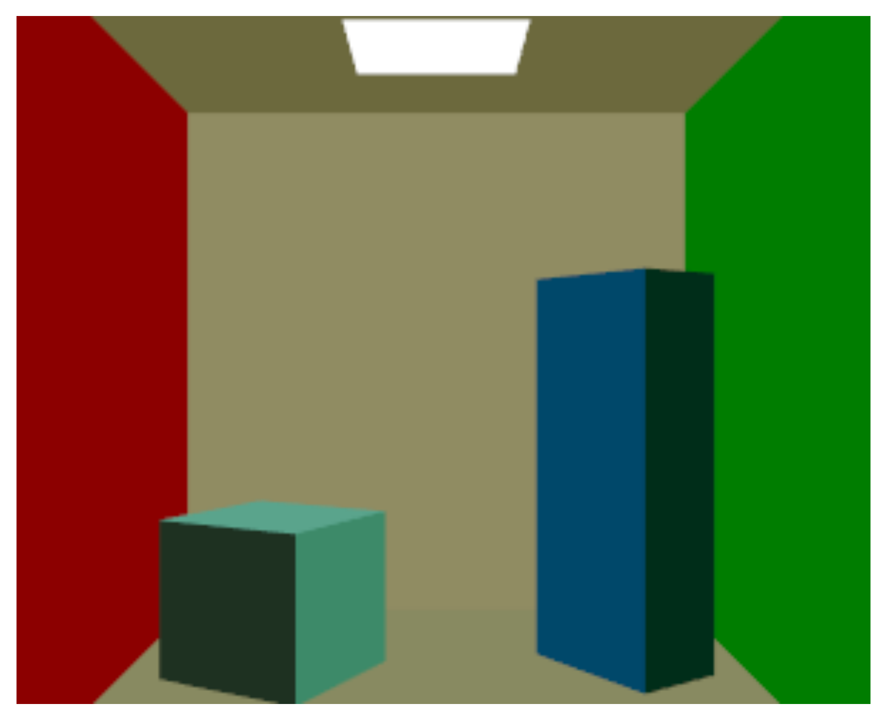
$$\begin{pmatrix} 1-\rho_1 F_{11} & -\rho_1 F_{12} & \dots & -\rho_1 F_{1N} \\ -\rho_2 F_{21} & 1-\rho_2 F_{22} & \dots & -\rho_2 F_{2N} \\ \vdots & \vdots & \dots & \vdots \\ -\rho_N F_{N1} & -\rho_N F_{N2} & \dots & 1-\rho_N F_{NN} \end{pmatrix} \begin{pmatrix} B_1 \\ B_2 \\ \vdots \\ B_N \end{pmatrix} = \begin{pmatrix} E_1 \\ E_2 \\ \vdots \\ E_N \end{pmatrix}$$

- Solve for B – Use methods like Gauss-Seidal

Radiosity steps

1. Generate Model (set up the scene)
2. Compute Form Factors and set the Radiosity Matrix
3. Solve the linear system
4. Render the scene





Radiosity steps

1. Generate Model (set up the scene)
2. Compute Form Factors and set the Radiosity Matrix
3. Solve the linear system
4. Render the scene

$$\begin{pmatrix} 1-\rho_1 F_{11} & -\rho_1 F_{12} & \ldots & -\rho_1 F_{1N} \\ -\rho_2 F_{21} & 1-\rho_2 F_{22} & \ldots & -\rho_2 F_{2N} \\ \vdots & \vdots & \ldots & \vdots \\ -\rho_N F_{N1} & -\rho_N F_{N2} & \ldots & 1-\rho_N F_{NN} \end{pmatrix} \begin{pmatrix} B_1 \\ B_2 \\ \vdots \\ B_N \end{pmatrix} = \begin{pmatrix} E_1 \\ E_2 \\ \vdots \\ E_N \end{pmatrix}$$

Radiosity steps

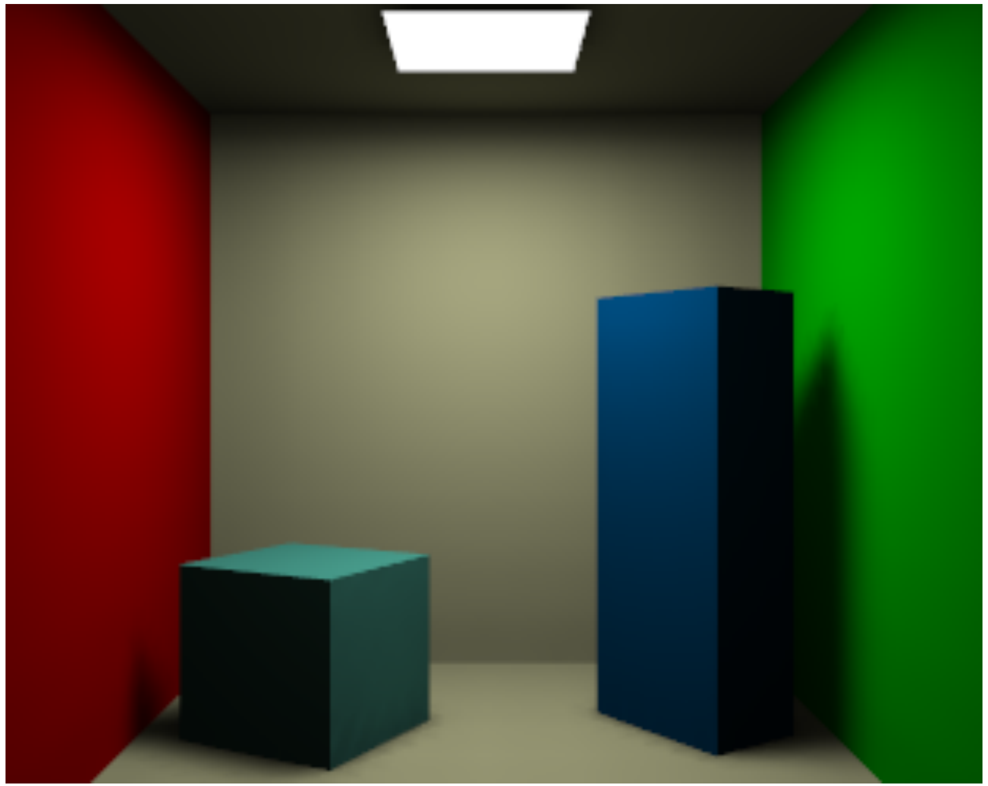
1. Generate Model (set up the scene)
2. Compute Form Factors and set the Radiosity Matrix
3. Solve the linear system
4. Render the scene

$$\begin{pmatrix} 1-\rho_1 F_{11} & -\rho_1 F_{12} & \dots & -\rho_1 F_{1N} \\ -\rho_2 F_{21} & 1-\rho_2 F_{22} & \dots & -\rho_2 F_{2N} \\ \vdots & \vdots & \dots & \vdots \\ -\rho_N F_{N1} & -\rho_N F_{N2} & \dots & 1-\rho_N F_{NN} \end{pmatrix} \begin{pmatrix} B_1 \\ B_2 \\ \vdots \\ B_N \end{pmatrix} = \begin{pmatrix} E_1 \\ E_2 \\ \vdots \\ E_N \end{pmatrix}$$

Radiosity steps

1. Generate Model (set up the scene)
2. Compute Form Factors and set the Radiosity Matrix
3. Solve the linear system
4. Render the scene

$$\begin{pmatrix} 1-\rho_1 F_{11} & -\rho_1 F_{12} & \dots & -\rho_1 F_{1N} \\ -\rho_2 F_{21} & 1-\rho_2 F_{22} & \dots & -\rho_2 F_{2N} \\ \vdots & \vdots & \dots & \vdots \\ -\rho_N F_{N1} & -\rho_N F_{N2} & \dots & 1-\rho_N F_{NN} \end{pmatrix} \begin{pmatrix} B_1 \\ B_2 \\ \vdots \\ B_N \end{pmatrix} = \begin{pmatrix} E_1 \\ E_2 \\ \vdots \\ E_N \end{pmatrix}$$



Radiosity steps

1. Generate Model
2. Compute Form Factors and set the Radiosity Matrix
3. Solve the linear system
4. Render the scene

$$\begin{pmatrix} 1-\rho_1 F_{11} & -\rho_1 F_{12} & \dots & -\rho_1 F_{1N} \\ -\rho_2 F_{21} & 1-\rho_2 F_{22} & \dots & -\rho_2 F_{2N} \\ \vdots & \vdots & \dots & \vdots \\ -\rho_N F_{N1} & -\rho_N F_{N2} & \dots & 1-\rho_N F_{NN} \end{pmatrix} \begin{pmatrix} B_1 \\ B_2 \\ \vdots \\ B_N \end{pmatrix} = \begin{pmatrix} E_1 \\ E_2 \\ \vdots \\ E_N \end{pmatrix}$$

- Where do we resume from if objects are moved?
- Where do we resume from if the lighting is changed?
- Where do we resume from if the reflectance parameters of the scene are modified?
- Where do we resume from if the view point changes?

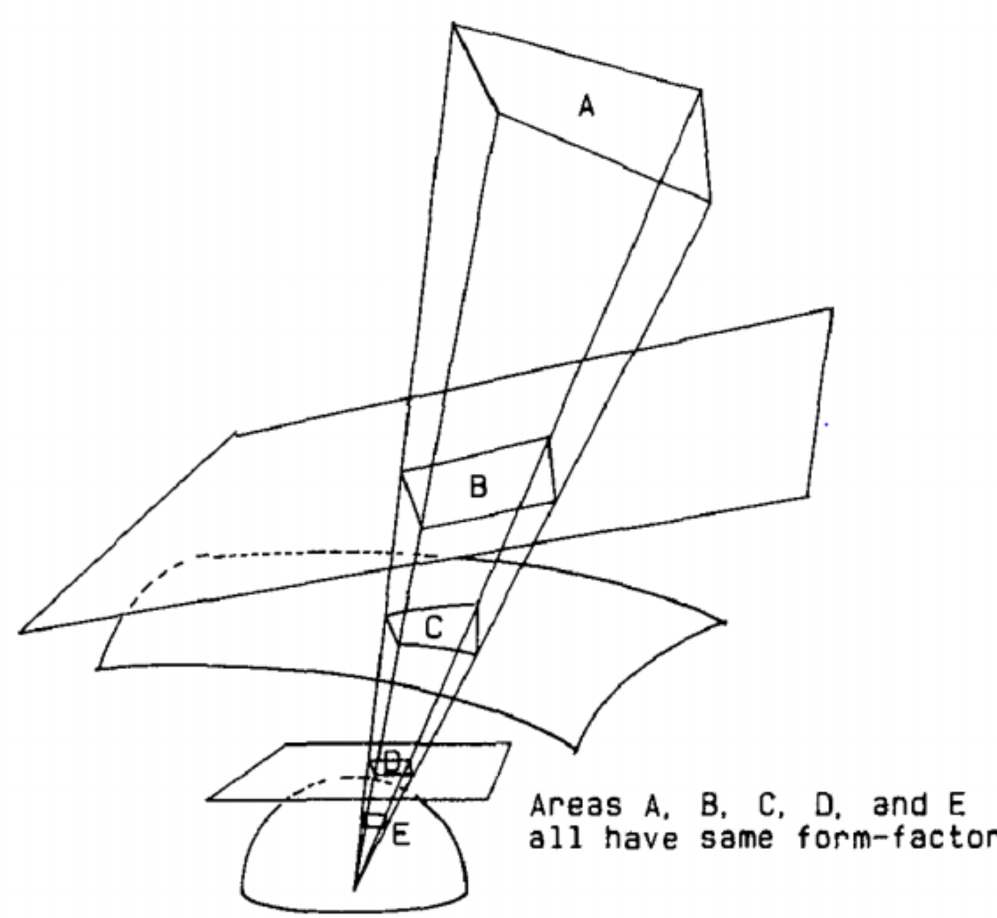
Radiosity features

- Very costly
- The faces must be subdivided into small patches to reduce the artefacts
- The computational cost for calculating the form factors is expensive
  - Quadratic in the number of patches
- Solving for Bi is also very costly
  - Cubic in the number of patches
- Cannot handle specular light

Hemicube method

by Michael Cohen'85

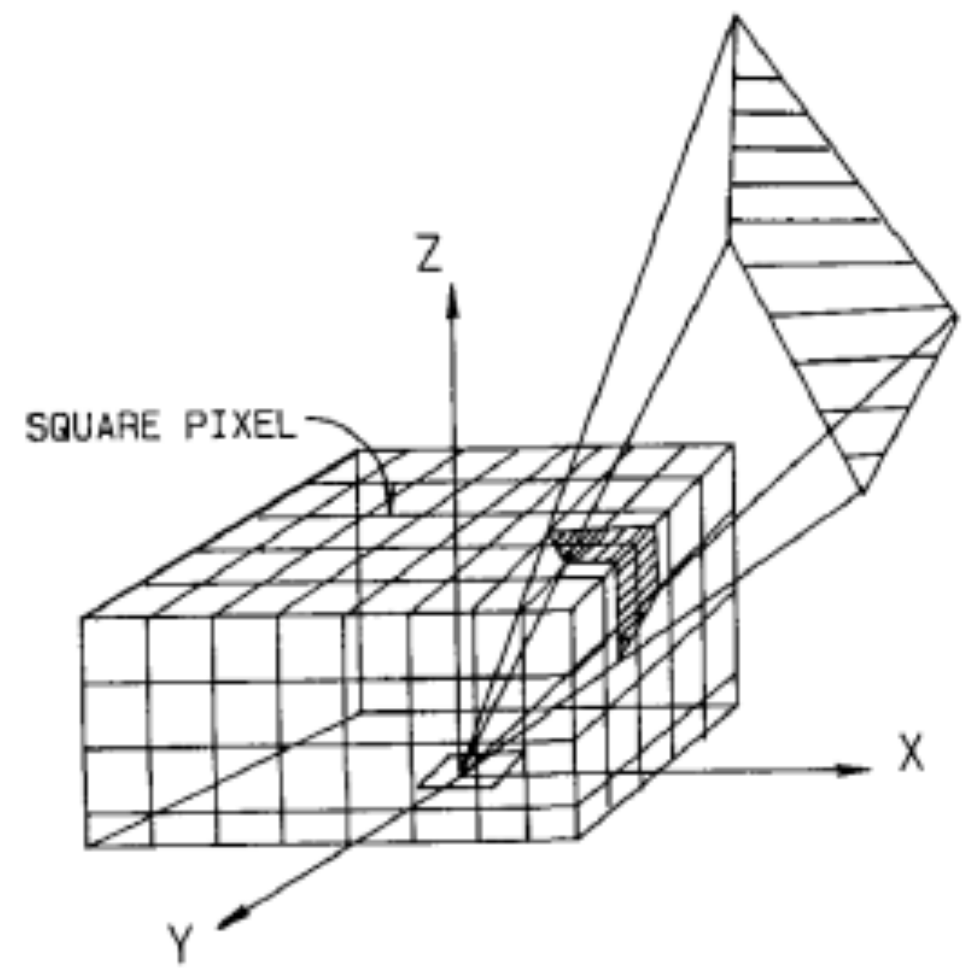
- Accelerates the computation of the form factor
- The form factor for the right four faces with respect to a small patch in the bottom is the same
- Then, we can project the patches onto a hemicube



AREAS WITH IDENTICAL FORM-FACTOR

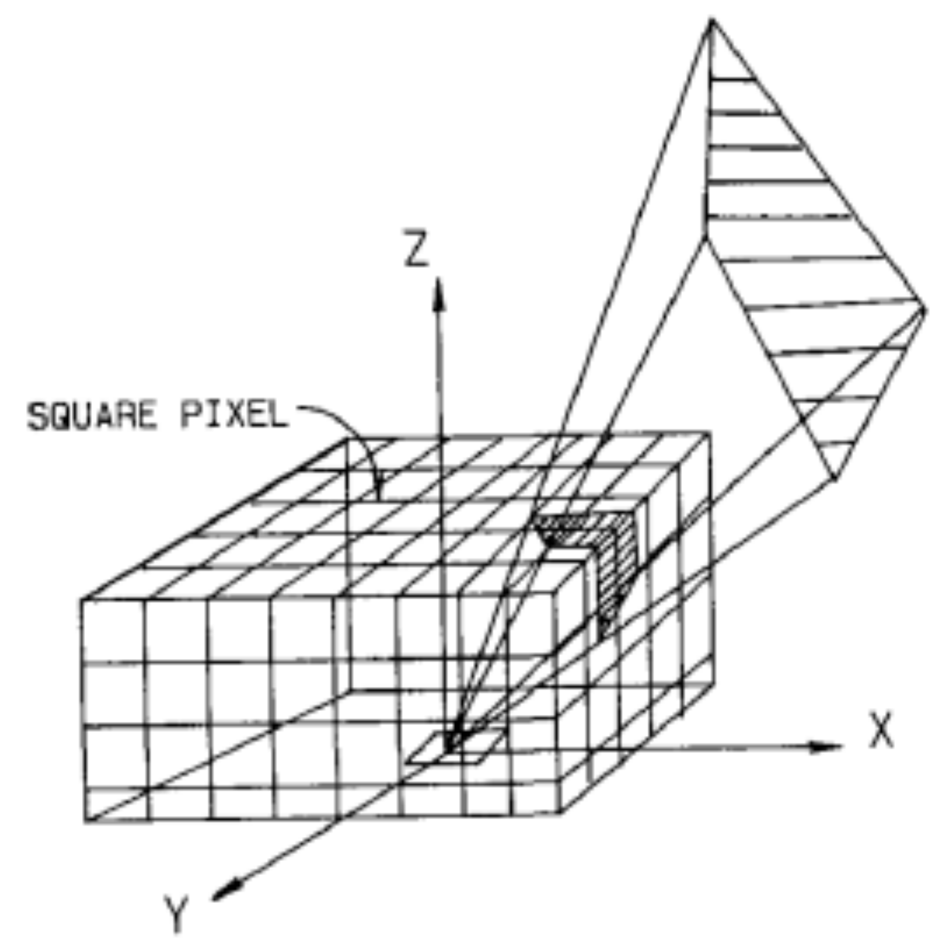
Hemicube method

- Prepare a hemicube around the patch i
- Project those polygons you want to compute the form factor with patch i onto the hemicube
- Then, compute the form factor between them



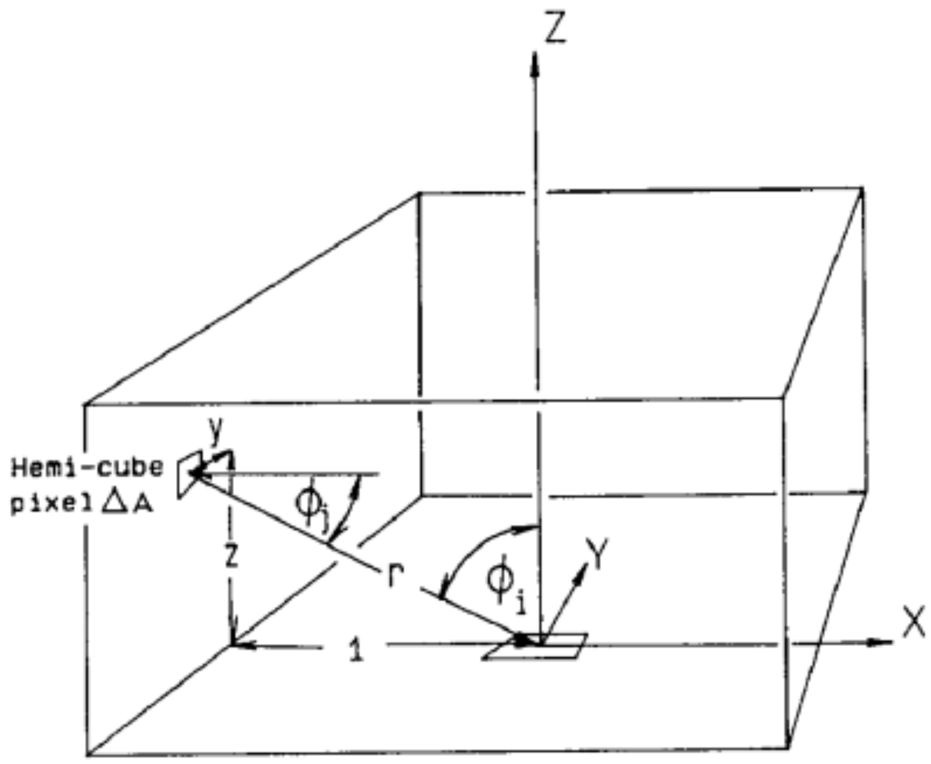
Hemicube method

- This can be done by perspective projection
- We can use the Z-buffer algorithm to find the closest polygon
  - Handling the occlusion
  - Setting the form factor of the pairs that occlude each other to zero



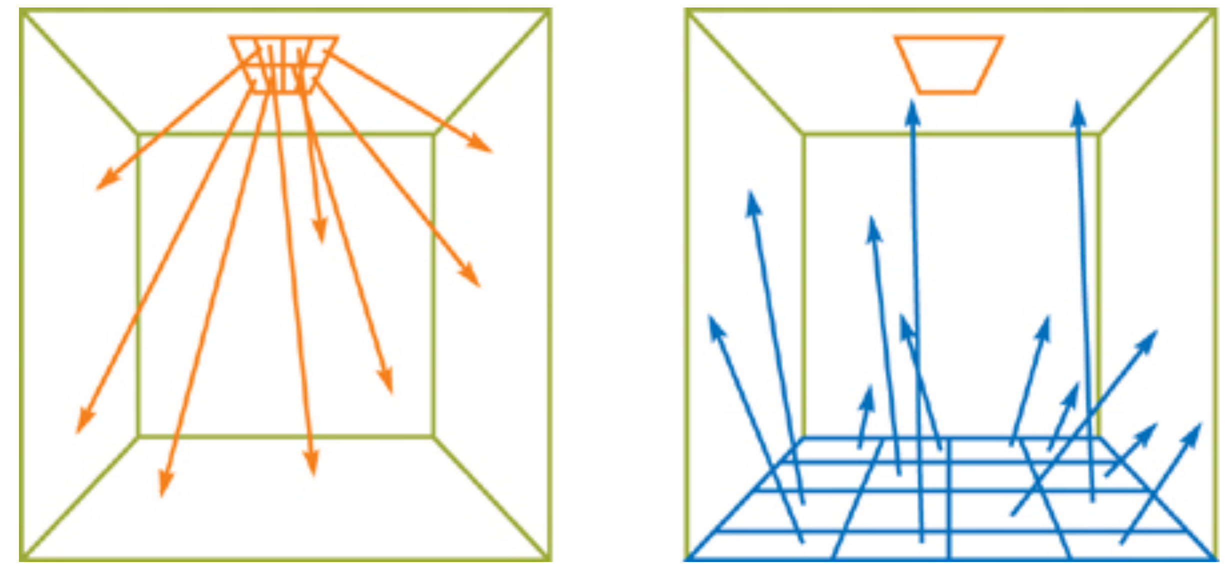
Hemicube method

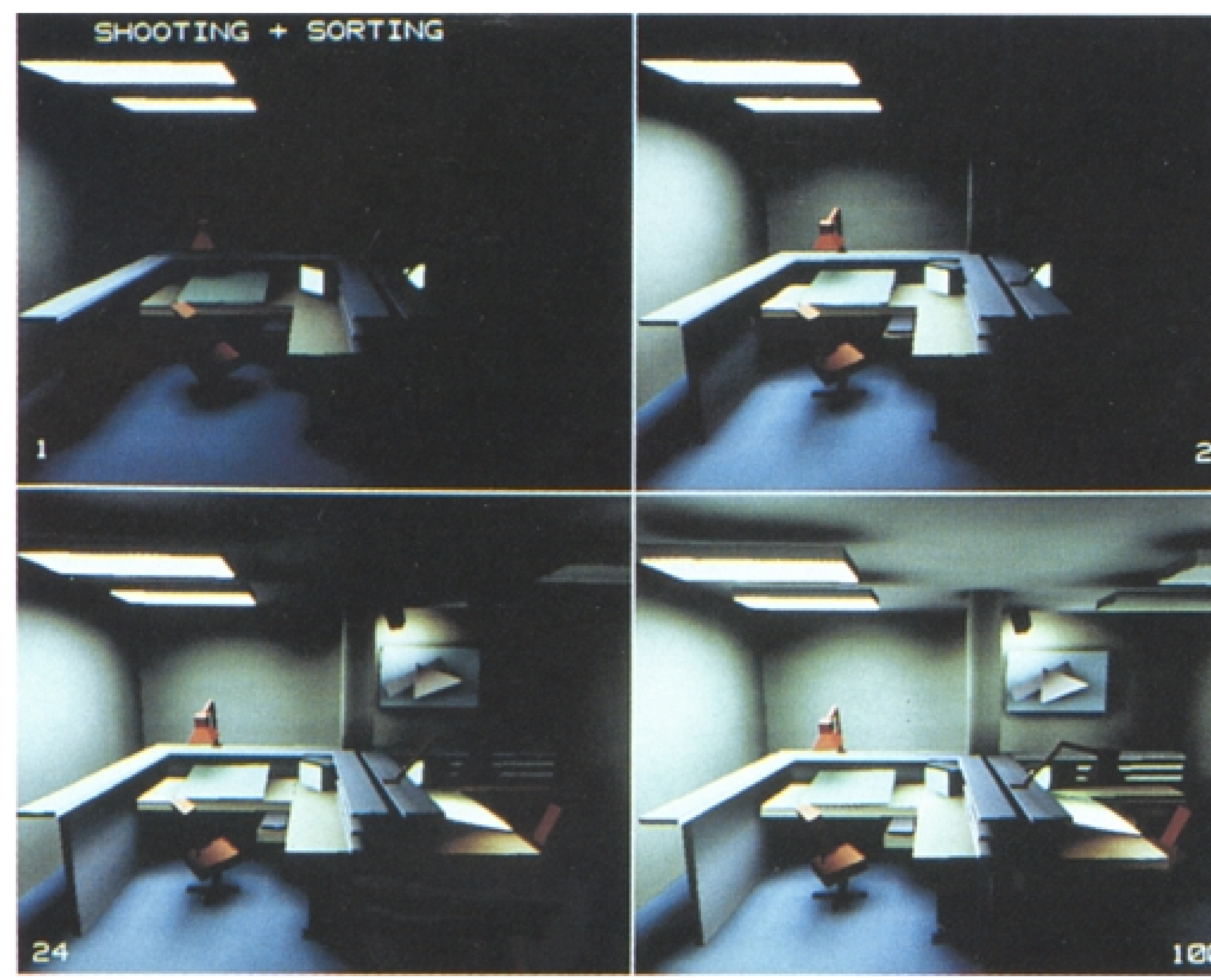
- The form factor between each pixel of the hemicube and the patch at the origin can be pre-computed and saved in a table
- Only 1/8 of all are needed, thanks to symmetry



Progressive refinement

- Find patch in scene with largest undistributed radiosity and distribute it into the scene
- Store radiosity shot into scene and mark as undistributed
- Iterate until converged
- No need to store radiosity matrix (form factors computed on the fly)





Summary

- BRDFs describe how a surface reflects light
- Ambient occlusion approximates shadowing of ambient lighting
- Spherical harmonic lighting allows us to efficiently render pre-calculated radiance transfer under arbitrary lighting conditions
- Radiosity can simulate diffuse inter-reflectance
- Form factor computation can be accelerated by hemi-cube.

References

- BRDFs: Shirley Chapter 20.1.6
- Ambient occlusion: Akenine-Möller Chapter 9.2
- Spherical harmonic lighting: Akenine-Möller Chapter 9.11
- Radiosity:
  - Foley Chapter 16.13
  - Extra: http://http.developer.nvidia.com/GPUGems2/gpugems2_chapter39.html

References - papers

- Spherical harmonics:
  - Precomputed Radiance Transfer for Real-Time Rendering. in Dynamic, Low-Frequency Lighting Environments. Peter-Pike Sloan. Jan Kautz. John Snyder, SIGGRAPH 2002
  - NG, R., RAMAMOORTHI, R., AND HANRAHAN, P. 2003. All-Frequency Shadows Using Non-Linear Wavelet Lighting Approximation. ACM Transactions on Graphics 22, 3, 376–381
  - Jan Kautz, Peter-Pike Sloan, Jaakko Lehtinen. "Precomputed Radiance Transfer: Theory and Practice", SIGGRAPH 2005 Courses
- Radiosity:
  - Cohen et al., The hemi-cube: a radiosity solution for complex environments, SIGGRAPH ‘85