Environment Sampling

Georgios Papaioannou - 2015

AMBIENT OCCLUSION

Ambient Occlusion

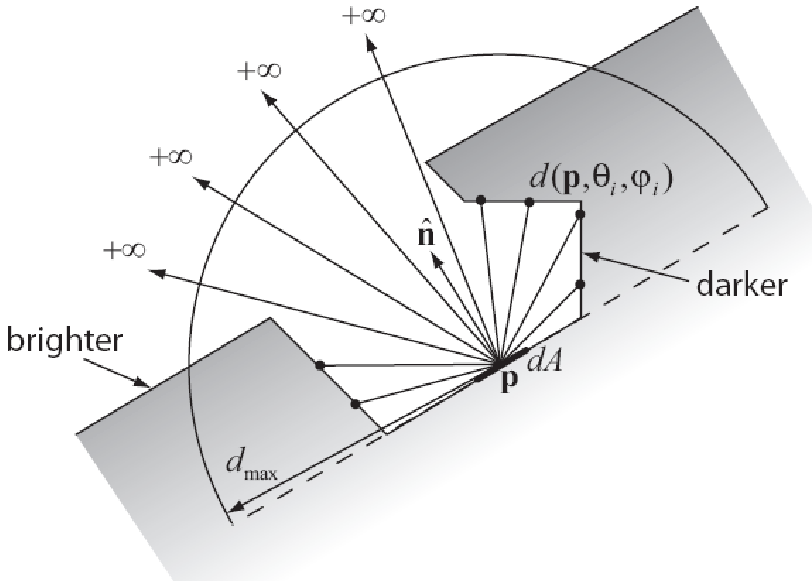
- Simplified shader for one-bounce case
- Radiance from other surfaces is regarded constant over all directions
- Energy reception from a distant environment treated as attenuation of light due to blocking
- Relates "openness" of a surface to brightness
  - No blocking: Full radiance received from all directions
  - Partial blocking: Near surfaces attenuate light
  - Full blocking: No light enters the surface.

Why Use Ambient Occlusion?

- A cheap way to simulate contribution of ambient (global) lighting
  - Though only convincing for outdoor scenes mostly
- Accentuates crevices → increases image contrast

Ambient Occlusion Estimation (1)

- Local or global illumination model?
- Hybrid!
  - Does not exchange light with other locations
  - Potentially search for occlusion up to a distance
  - Still requires visibility checks → intersections with other geometry



Ambient Occlusion Estimation (2)

- The value of occlusion shading can be easily determined if we set L_i in the reflectance equation to 1 and replace visibility with an attenuation score:

$$w(\mathbf{p}) = \frac{1}{\pi} \int_{\Omega} \mu\big(d(\mathbf{p}, \omega_i)\big) d\sigma_{\perp}(\omega_i)$$

- Where $d(\mathbf{p}, \omega_i)$ is the distance to the closest hit point within a radius d_{max} (or $+\infty$ if no hit occurred)
  - d_{max} can be set to ∞

Ambient Occlusion – Attenuation Function

- $\mu\big(d(\mathbf{p},\omega_i)\big)$ can be any intuitive function
- Simplest case:

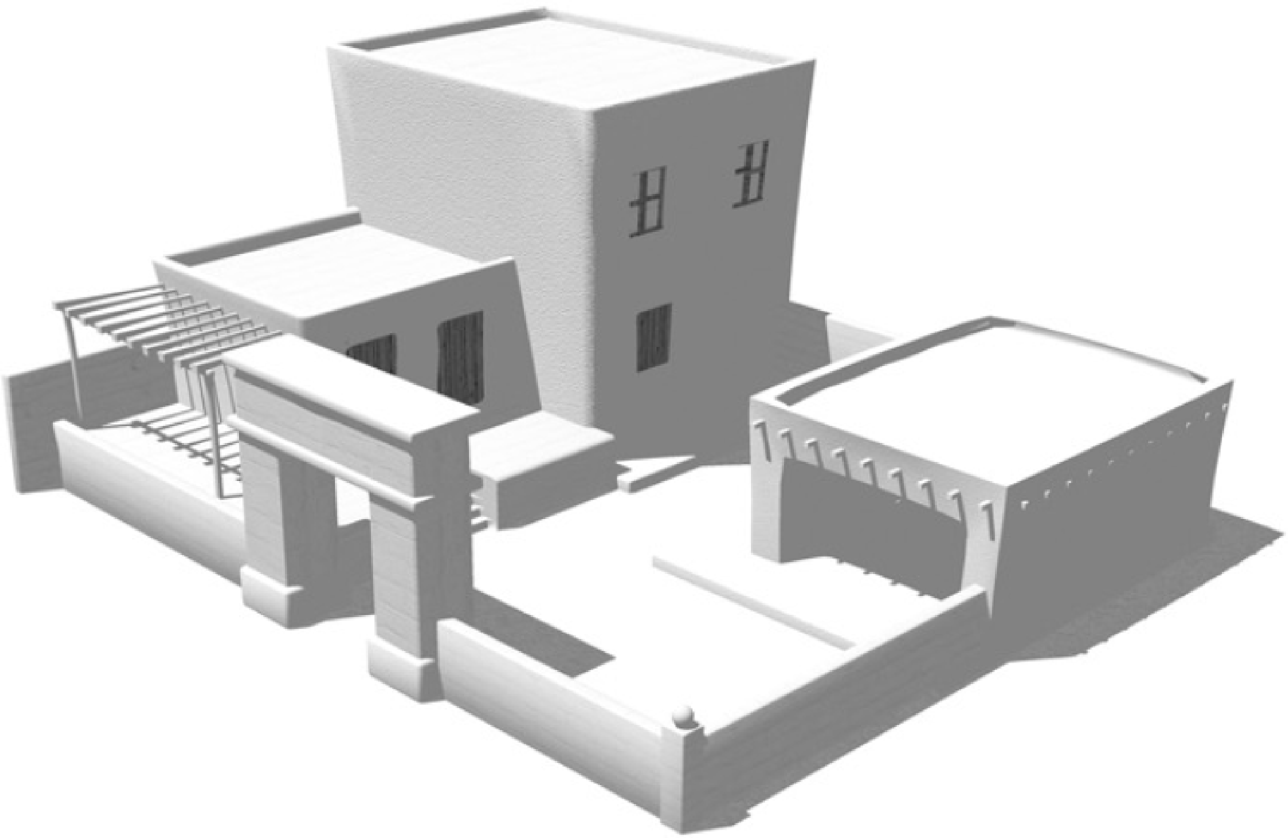
$$\mu\big(d(\mathbf{p},\omega_i)\big) = \begin{cases} 1, & no\ hit \\ 0, & otherwise \end{cases}$$

- But other forms can be used to limit the impact of distant occluders

A.O. : How is it Applied?

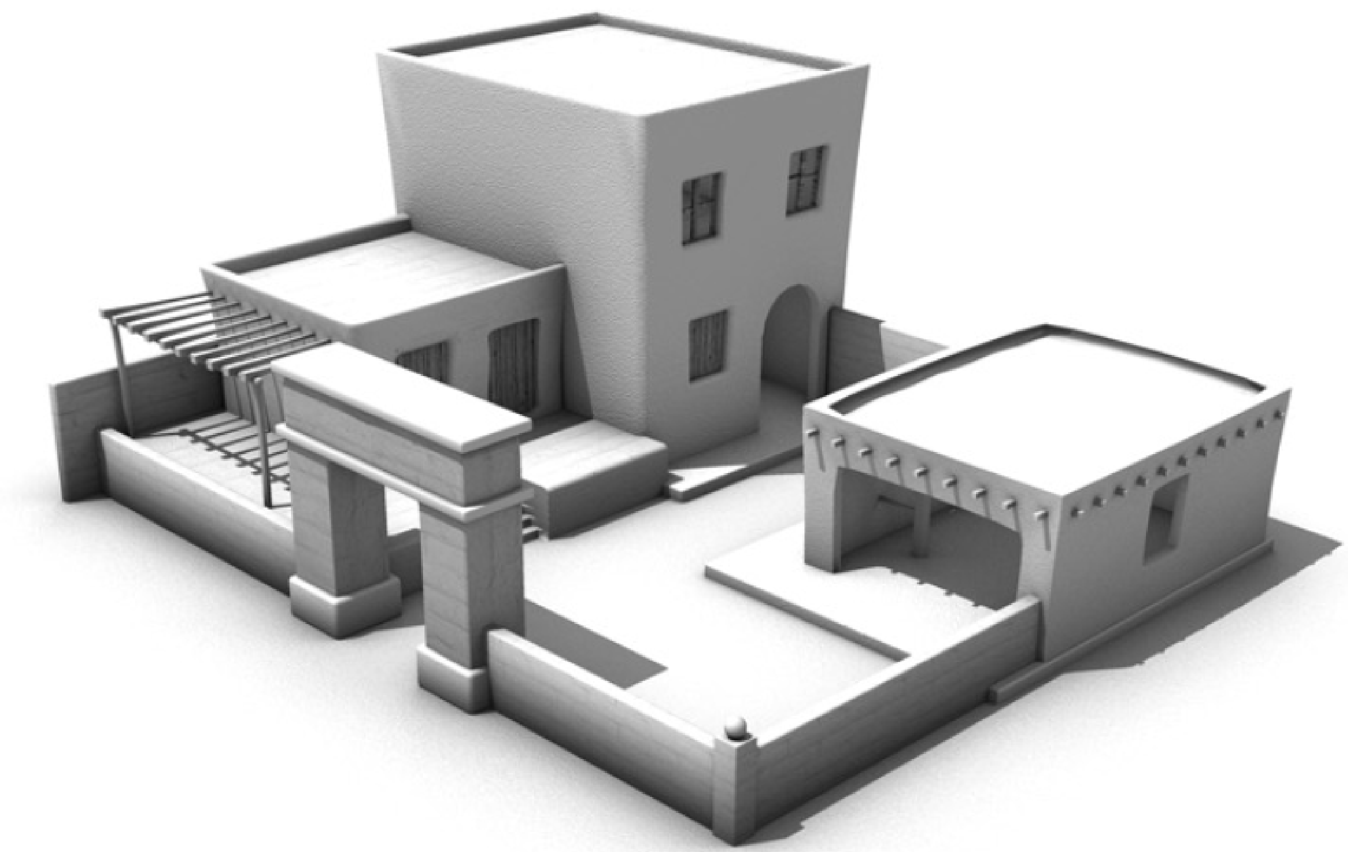
- We usually apply AO as a visibility function to attenuate ambient / sky color
- Some implementations also blend AO with diffuse or even specular lighting (not really correct…)

A.O. Example



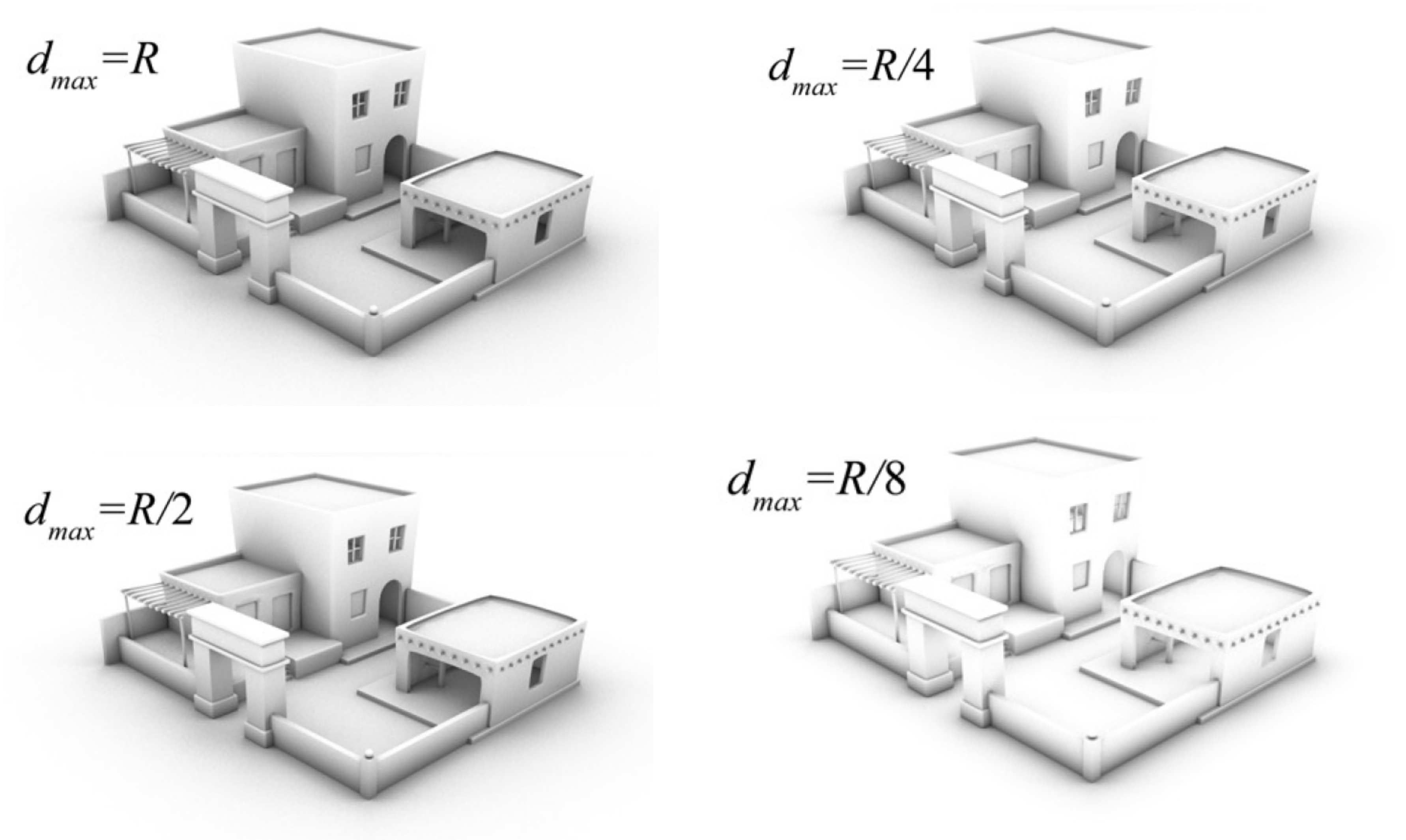
Scene rendered with constant ambient lighting

A.O. Example



Scene rendered with ambient occlusion ($d_{max}=R/8$)

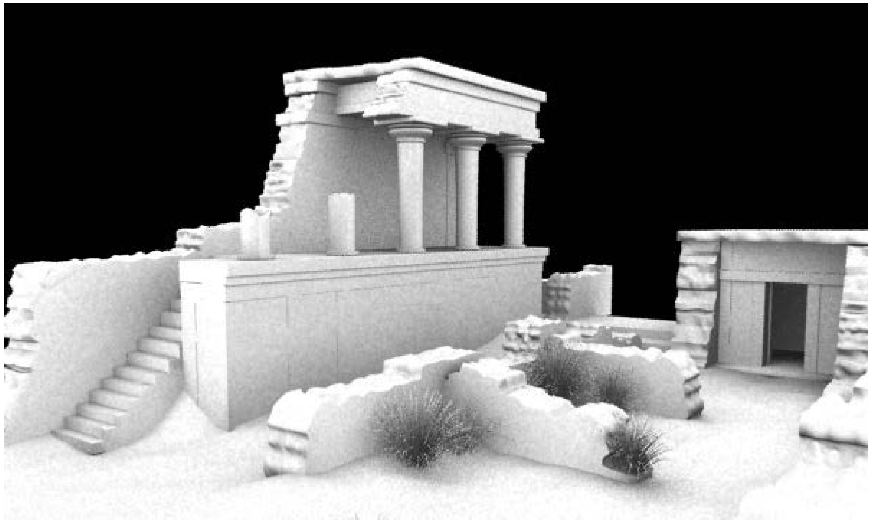
A.O. - Effect of maximum distance



Ambient Occlusion vs Uniform Light

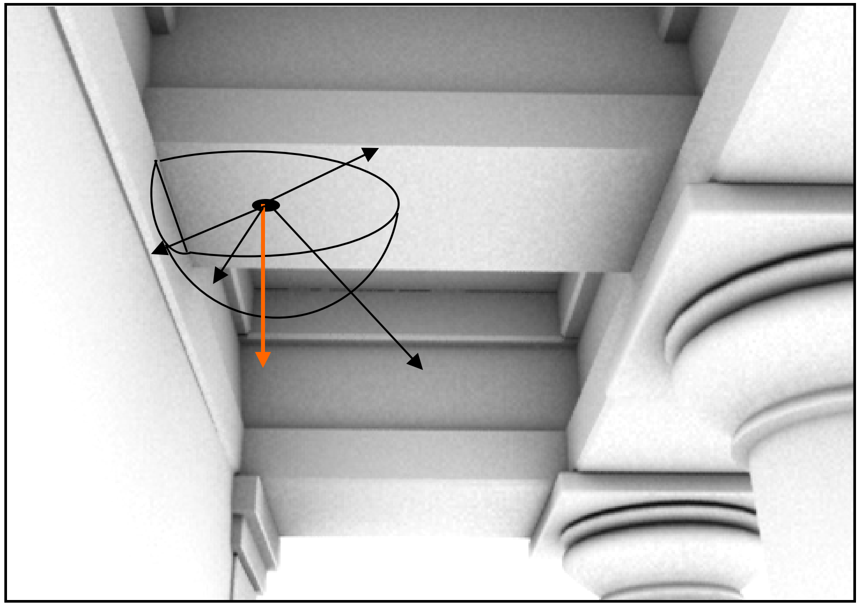
Hemispherical light

Ambient occlusion



Ambient Occlusion Calculation

- For every visible point $\mathbf{x}$:
  - Compute AO as Monte Carlo hemispherical integral. Sample the hemisphere with N rays:
    - Find closest intersection $\mathbf{y}$ with occluding geometry (the most expensive calculation)
    - Compute distance $d(\mathbf{x},\mathbf{y})$
    - Compute attenuation $\rho(d)$



- Also fast implementations for real-time graphics
  - Use screen-space information

Precalculated A.O.

- In real-time graphics, sometimes we can evaluate AO per vertex and store it as vertex color on meshes
- At runtime, we can then apply it on any shading for free
- Requires careful geometry tesselation during modeling to avoid problems:

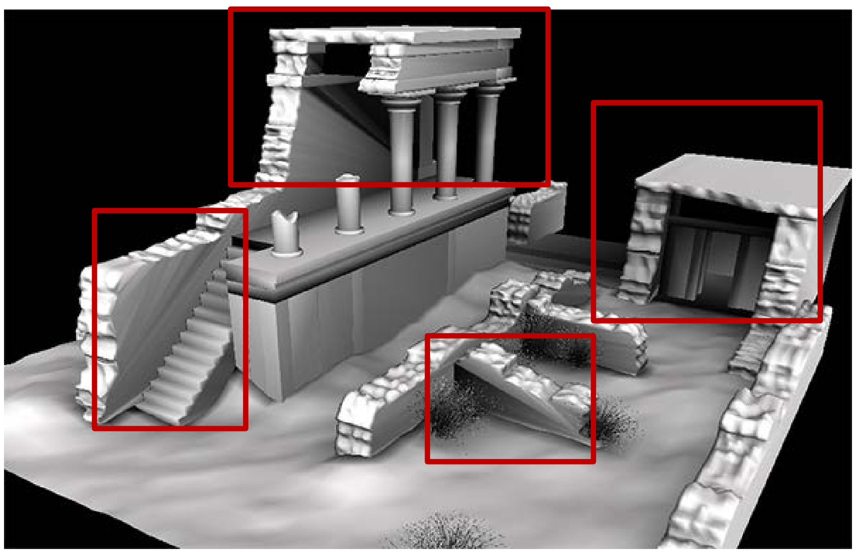


IMAGE-BASED LIGHTING

Image-based Lighting

- Very important in CG
- Helps a rendered image blend with a real surrounding
  - Mix synthesized and real imagery (films, games, AR)

Environment Maps (1)

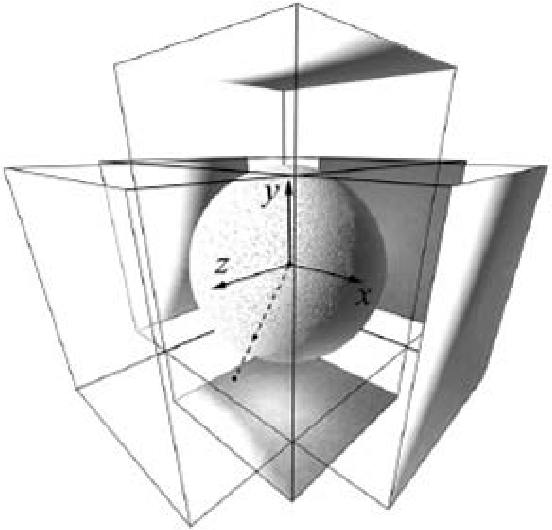
- An environment map is a representation of distant radiance parameterized w.r.t. an incoming direction ω_i
- Usually this information is discretely encoded on a set of images
- Other typical representations include spherical function coefficients

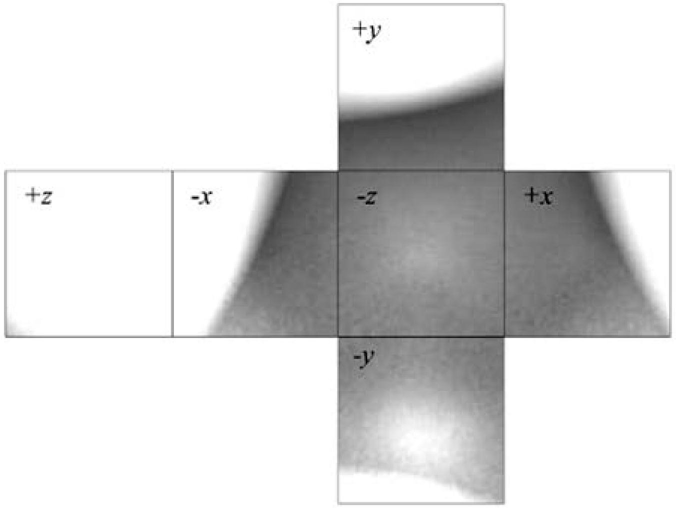
Environment Maps (2)

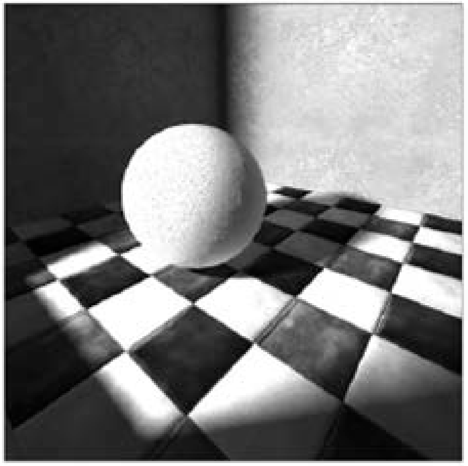
- Environment maps typically encode incoming illumination from the entire sphere around a point
- But can also be:
  - Hemispherical (e.g. sky lighting)
  - Cylindrical

Environment Maps (2)

- Mostly in real time graphics, it is convenient to store the spherical environment in cube maps:







Light Probes (1)

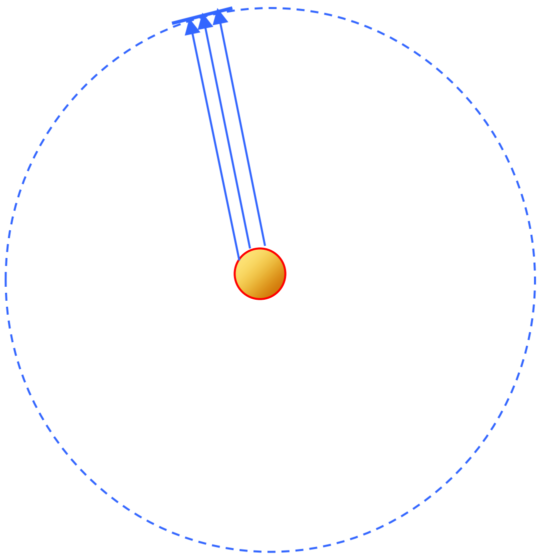
- Environment lighting images can be captured using physical light probes:
- Highly polished metallic spheres photographed to capture the real environment
  - Multiple exposures are typically taken to capture an HDR environment map

Light Probes (2)

- To properly map the environment:
  - with low distortion and
  - Elimination of the photographic equipment from the image
- Multiple photos of the probe are captured
- The results are merged into an (inverse) panorama

Using an Environment Map (1)

- The basic assumption about environment maps is that the environment is distant
- If assumed distant, incoming light is parameterized only by direction, as different points on the geometry will still index the same location on the environment map



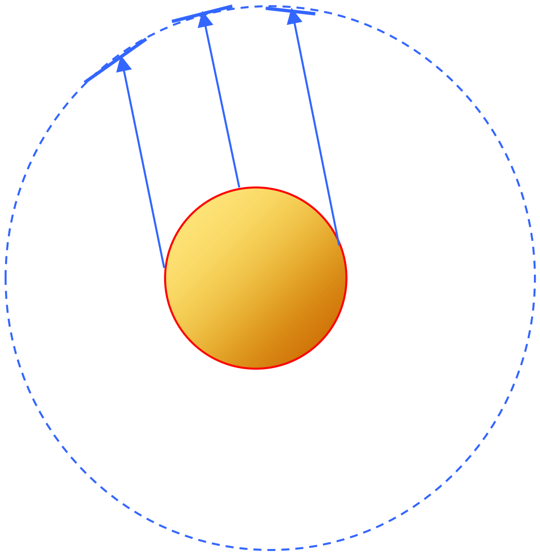
Using an Environment Map (2)

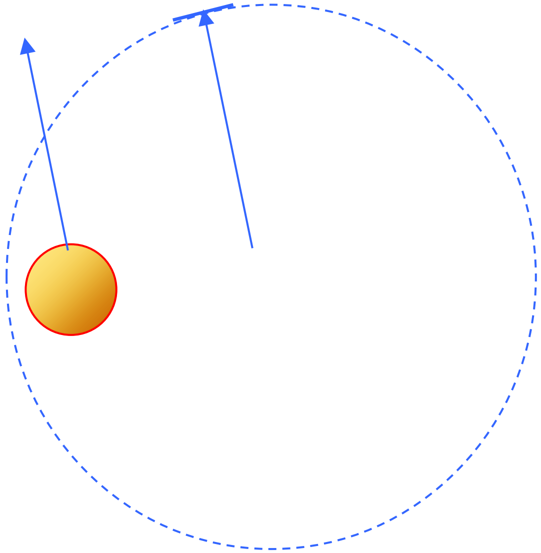
- Using as lighting the environment map on each point instead of using light sources:
  - Can provide a very natural look to artificial objects
  - Can blend the synthetic geometry with the captured environment
- This has been extensively used in movies

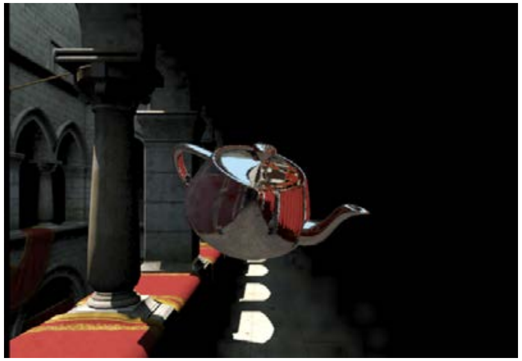
http://www.fxguide.com/featured/vfx-roll-call-for-the-avengers/

Using an Environment Map (3)

- When environment distances are comparable to the size of the synthetic objects, a single environment map cannot do the trick
- Env. maps are also only valid for a particular region near the capture point







Virtual Light Probes (1)

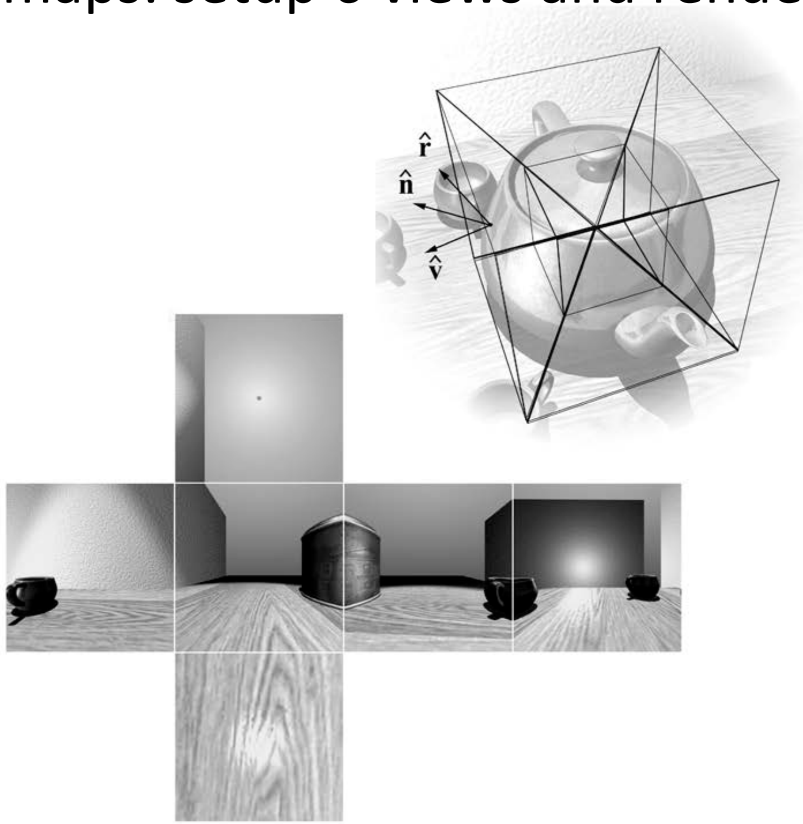
- In the previous example, the environment map was not captured from a real scene, but rather from a synthetic environment

Why do this?

- To significantly speed up indirect lighting calculations
- To apply indirect lighting to real-time rendering!
  - "Bake" incident light from a rendered environment
  - This lighting is the contribution of the env. Lighting to a surface
  - Can be combined with local shading from light sources

Virtual Light Probes (2)

- Generation:
  - Via cube maps: setup 6 views and render the scene

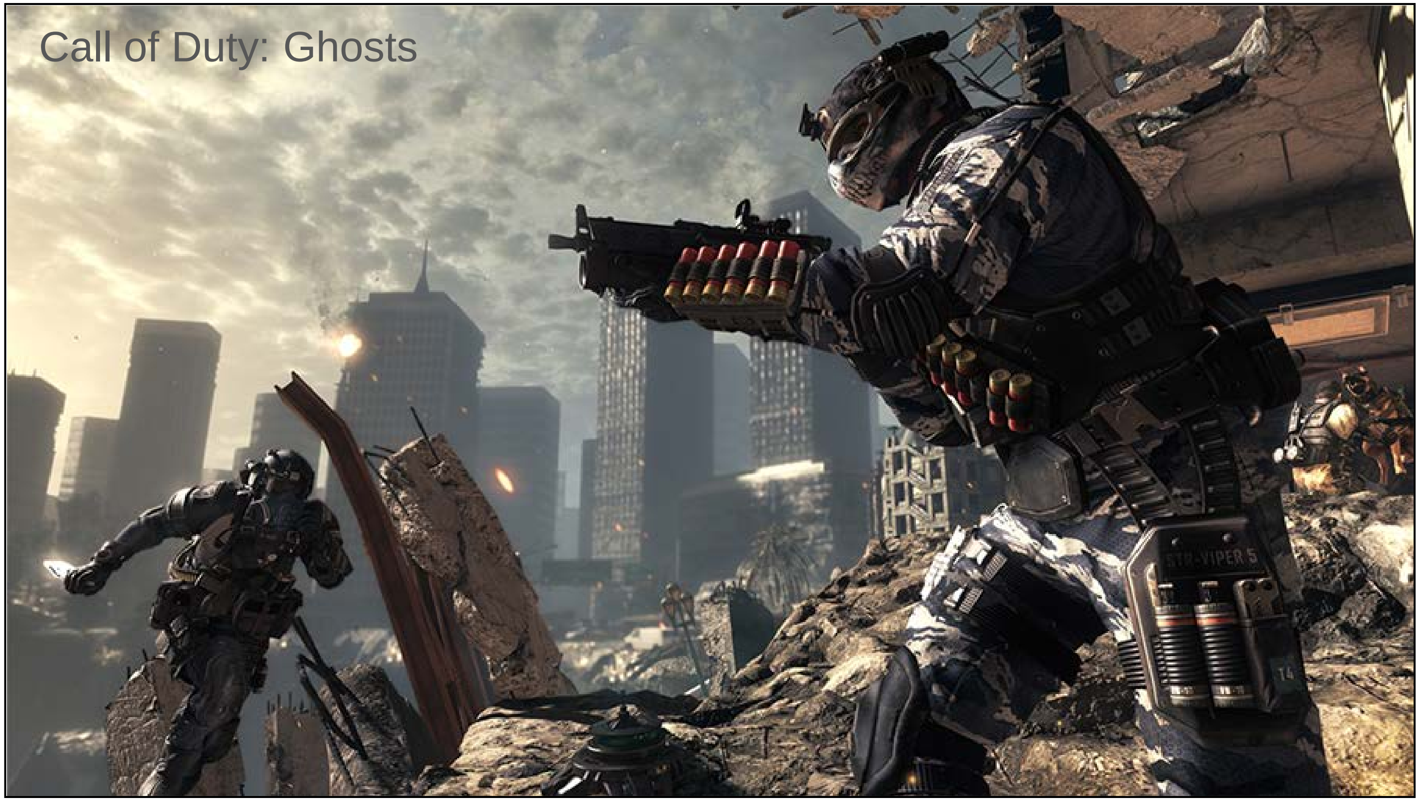


Virtual Light Probes (3)

- Generation:
  - Directly sample the geometry and store a compressed spherical representation (see RT GI slides)

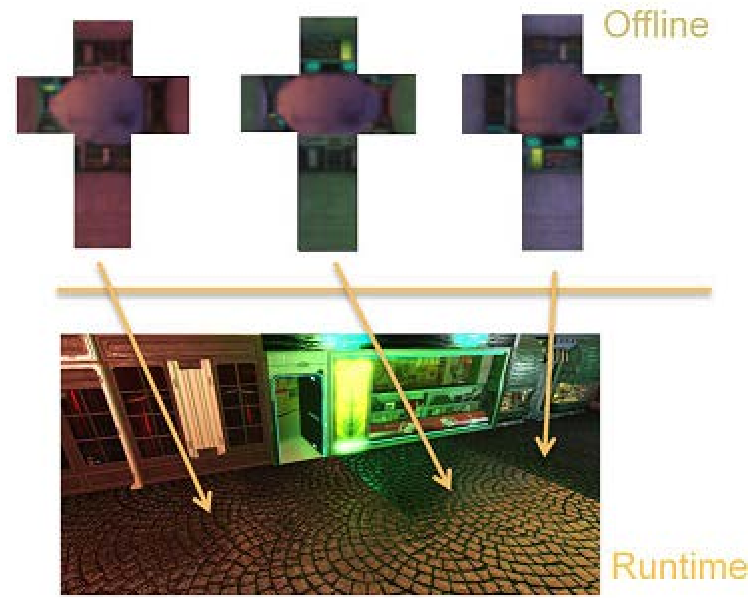
Environment Mapping in RT Applications

- Used for baking both rough indirect lighting and sky / ambient lighting



Multiple Light Probes

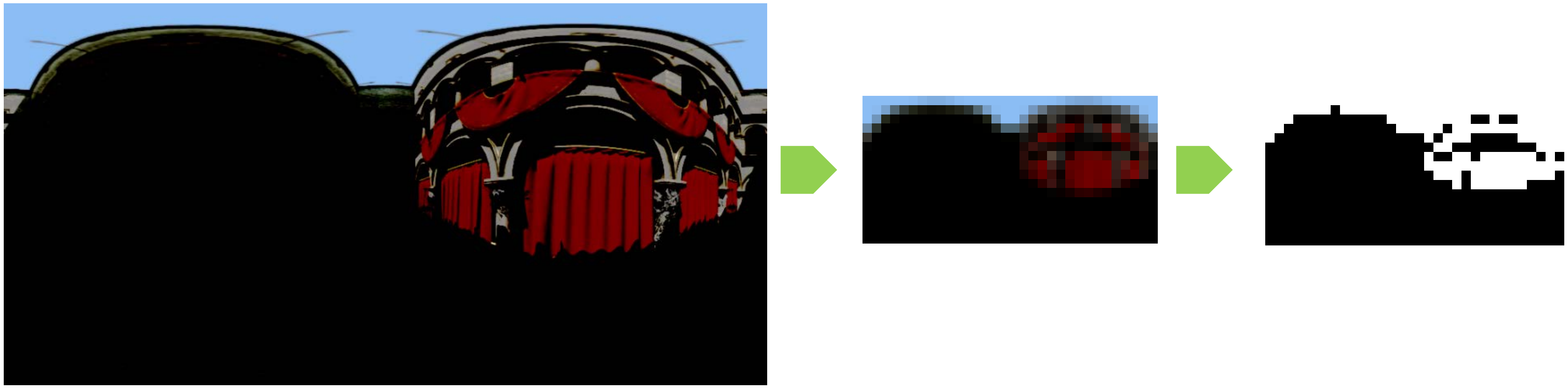
- To alleviate the invalidation of environment maps in different scene positions, multiple (virtual or physical) light maps can be generated from different locations
- At runtime, their contribution is interpolated



Importance Sampling Environment Maps (1)

- Environment maps cover the entire field of view around a point
  - At best, the hemisphere above the surface
- How do we sample the rendering equation integrand with only a few samples?
  - → importance sampling
- With env. maps, we do have the $L_e(\mathbf{p}, \omega_i)$!

Importance Sampling Environment Maps (2)



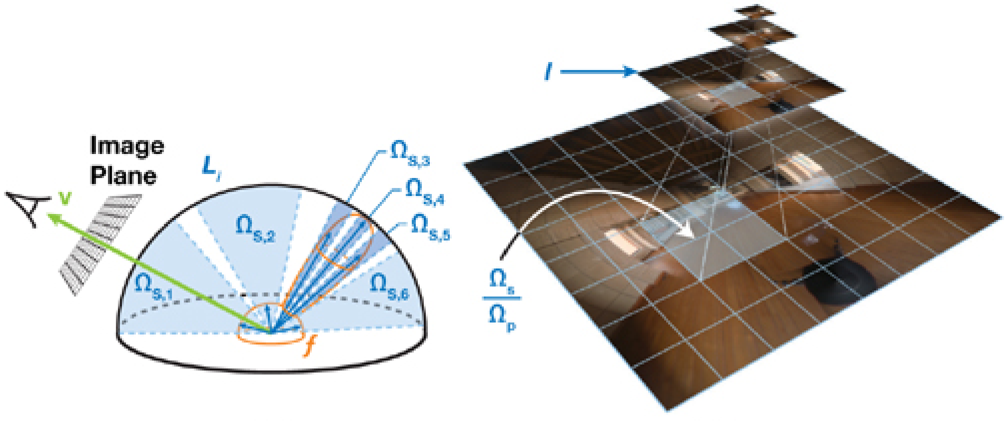
- So we are done: we mipmap and sample the map, after thresholding its values to obtain a sample mask
- No?

Importance Sampling Environment Maps (3)

- The lighting information is not enough!
- Remember the integrand also contains a visibility term!
- So we need to first evaluate the visibility function, then combine it with the env. map to obtain a distribution of good sampling locations
  – We need to find a way to approximately and quickly compute the visibility function…

Pre-Convolved Environment Maps

- To reduce the number of samples without introducing variance, another solution is to:
  - Prefilter the environment map (similar to mip-mapping)
  - During rendering, choose and blend environment mipmap levels according to the spread of the BSDF



(a) (b)

PRECOMPUTED RADIANCE TRANSFER

Orthonormal Basis Functions

- A basis function b_n is an element of a particular basis for a function space
- Every continuous function in the function space can be represented as a linear combination of basis functions:

$$f(x) = \sum_{n \in N} a_n b_n(x)$$

- Check similarity with vector spaces
- An orthonormal basis additionally satisfies the property:

$$\int b_i b_j = \delta(i-j) \qquad \forall i, j \in \mathrm{N}$$

Signal Projection on Orthonormal Bases

- The projection of an arbitrary continuous function on a set of basis functions results in the definition of the blending coefficients a_n
- It can be proven that for orthonormal function bases, the best least squares fitting of a function f over a predefined set of basis functions b_n results in:

$$a_n = \int f(x) b_n(x) \mathrm{d}x$$

- (Again, relate this with the dot product projection in orthonormal bases for vector spaces)

Signal Reconstruction

- The number of basis (blending) functions may be infinite or too large and therefore we must choose a finite subset of them that converges "reasonably" to the desired result

- The reconstructed function (signal) is derived from the linear combination of the (truncated series) of basis functions:

$$\tilde{f}(x) = \sum_{n=1}^{N} a_n b_n(x)$$

Spherical Harmonics (1)

- Spherical Harmonics define an orthonormal basis over the sphere **S**.
- A point s on the sphere is parameterized as:

$$s = (x, y, z) = (\sin\theta\cos\varphi, \sin\theta\cos\varphi, \cos\theta)$$

- They are harmonic functions and more specifically they constitute the angular part of the solution of the Laplace's equation on the unit sphere:

$$\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2} = 0$$

Spherical Harmonics (2)

- The (complex) basis functions are defined as:

$$Y_l^m(\theta,\varphi) = K_l^m e^{im\varphi} P_l^{|m|}(\cos\theta), l \in \mathbf{N}, -l \leq m \leq l$$

where P_l^m are the associated Legendre polynomials and K_l^m are the following normalization factors:

$$K_l^m = \sqrt{\frac{(2l+1)(l-|m|)!}{4\pi(l+|m|)!}}$$

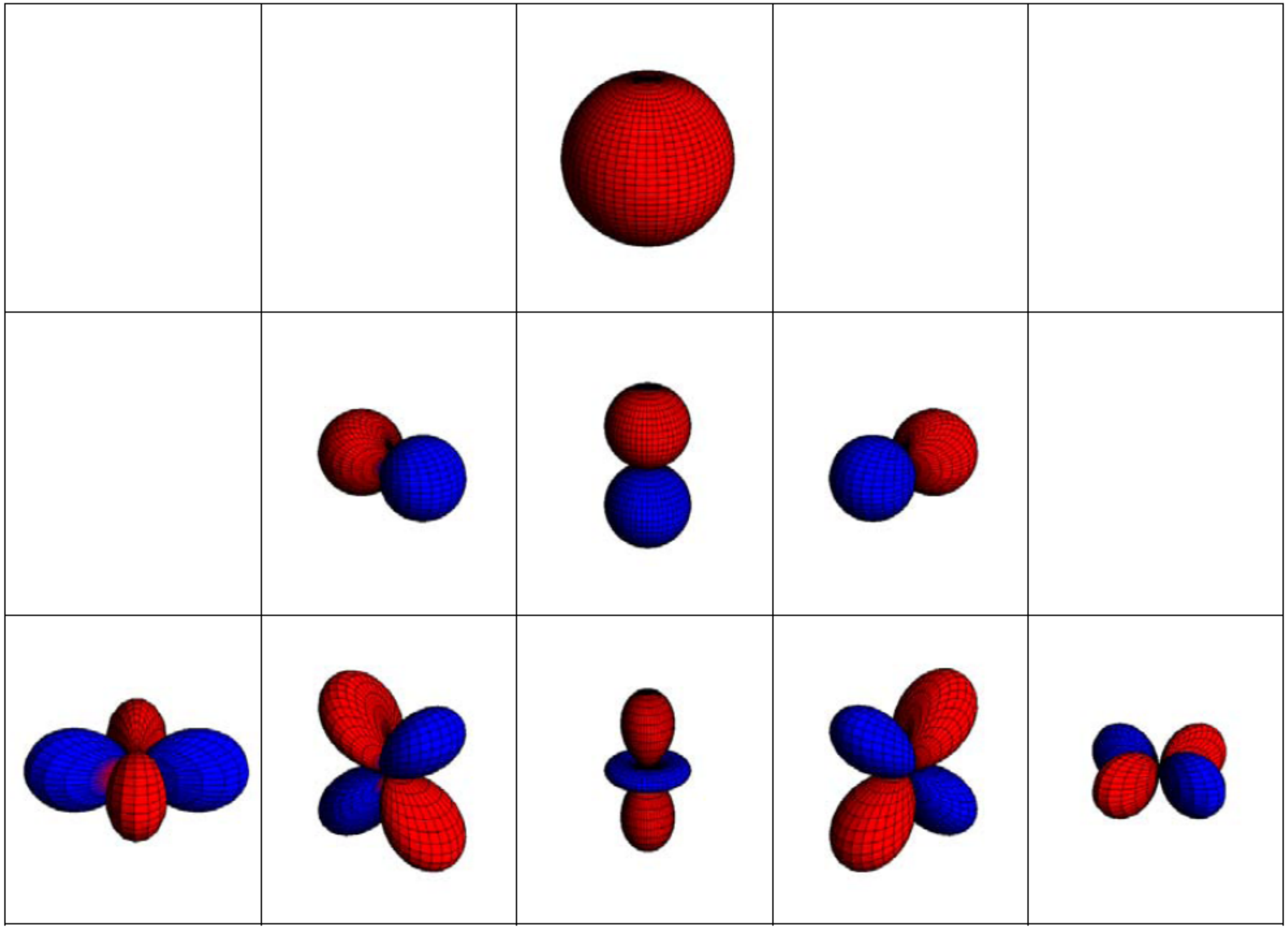
Spherical Harmonics (3)

- Real versions of the SH basis functions can be obtained from the transformation:

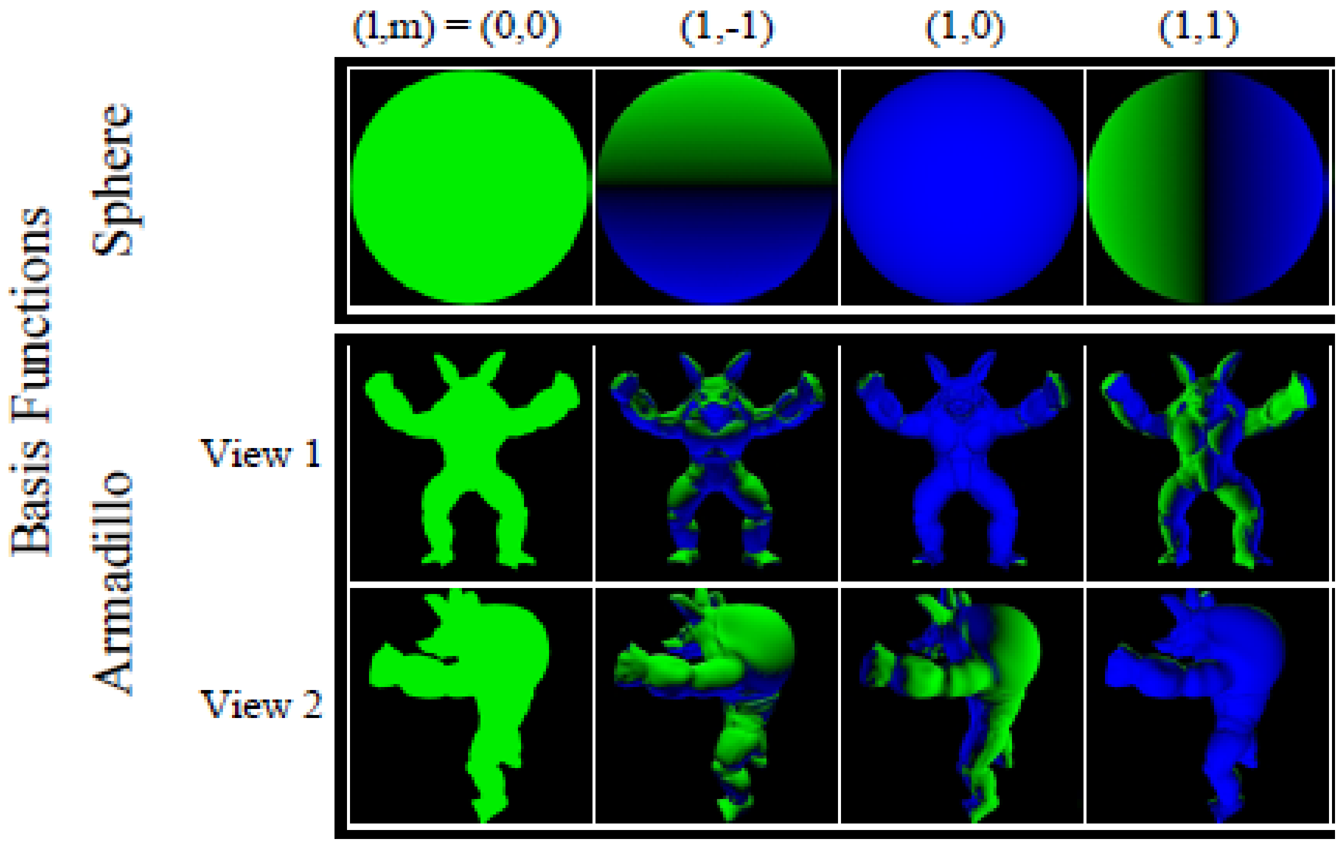
$$y_l^m = \begin{cases} \sqrt{2}\mathrm{Re}(Y_l^m) & m > 0 \\ \sqrt{2}\mathrm{Im}(Y_l^m) & m < 0 \\ Y_l^0 & m = 0 \end{cases} = \begin{cases} \sqrt{2}K_l^m \cos m\varphi \; P_l^m(\cos\theta) & m > 0 \\ \sqrt{2}K_l^m \sin|m|\varphi \; P_l^{|m|}(\cos\theta) & m < 0 \\ K_l^0 P_l^0(\cos\theta) & m = 0 \end{cases}$$

- l represents the band of the SH functions
- Each band has $2l+1$ SH basis functions
- Each band corresponds to an increasing angular frequency

Spherical Harmonics (4)



Spherical Harmonics (5)



Spherical Harmonics (6)

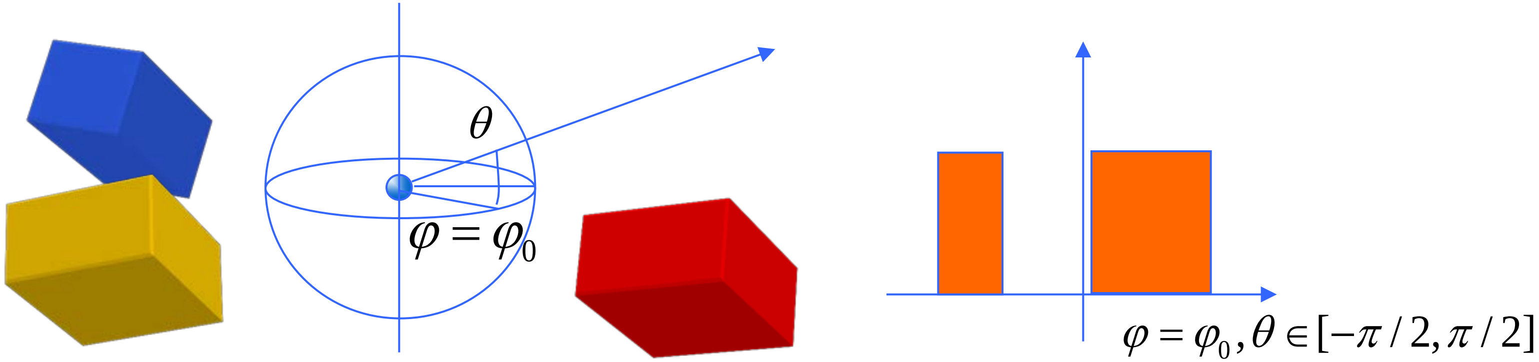
- Being an orthonormal set of basis functions:

$$f_l^m = \int f(s) y_l^m(s)\, ds$$

- The reconstruction of the signal can use up to any order of SH bands, truncating the infinite series of coefficients and respective basis functions
- Similarly, the encoded (projected) signal has to be band limited and encoded in a finite set of SH coefficients
- How many bands should we use?

Frequency Analysis of Radiance Field

- Similar to radiance, we can encode visibility as a 5D field:
  - What is the visibility (how open is the environment) at a point (x,y,z) in space in a direction (θ,ϕ)?
  - Encodes the ability of the specific point to receive light from an incident direction (θ,ϕ)



- What are the spectral characteristics of these fields?

Frequency Analysis of Illumination (1)

- Global illumination effects have distinctively different spectral characteristics
- As a principle:
  - Diffuse inter-reflections produce low frequency directional radiance
  - The same holds for most cases involving occlusion in diffuse light bounces
  - Direct illumination with occlusion (shadows) contains high frequencies in general (discontinuities)
  - Specular transmission usually contains high frequencies

Frequency Analysis of Illumination (2)

Encoding the Radiance/Visibility Field (1)

- Why?
  - Direct illumination is cheap to calculate at every point on the geometry
  - Indirect illumination is not
- Solution:
  - Precalculate on surfaces/cache points OR
  - Calculate at sparse locations at run time
- What:
  - Visibility AND/OR
  - Radiance field of indirect lighting

Encoding the Radiance/Visibility Field (2)

For real-time graphics:

- Calculating and storing the radiance/visibility field once or per frame:
  - Disassociates its utilization from the geometry
  - Enables the easy evaluation of GI in real-time graphics (direct rendering techniques)

Encoding Visibility (Distant Illumination) (1)

- From the rendering equation:

$$L_r(\phi_r,\theta_r) = L_e(\phi_r,\theta_r) + \int_{\Omega_i} L_i(\phi_i,\theta_i) f_r(\phi_r,\theta_r,\phi_i,\theta_i)\cos(\theta_i)d\omega_i$$

- If we assume only a “distant” environment emitting the radiance (e.g. sky, sun, distant light sources etc), then:

$$L_r(\phi_r,\theta_r) = \int_{\Omega_i} \underbrace{\boxed{L(\phi_i,\theta_i)}}_{\text{radiance}} \underbrace{\boxed{V(\phi_i,\theta_i) f_r(\phi_r,\theta_r,\phi_i,\theta_i)\cos\theta_i}}_{\text{transfer function}} d\omega_i$$

Encoding Visibility (Distant Illumination) (2)

- For diffuse surfaces this is simplified to:

$$L_r(\phi_r,\theta_r)=\frac{\rho}{\pi}\int_{\Omega_i} L(\phi_i,\theta_i)\overbrace{V(\phi_i,\theta_i)\cos\theta_i}^{T(\phi_i,\theta_i)}d\omega_i$$

- The hemisphere is aligned with the surface normal at every point
- The transfer function characterizes the specific point but for diffuse inter-reflection can be considered a slowly varying quantity (thus sparsely evaluated).

Encoding Visibility (Distant Illumination) (3)

- We can encode both the transfer function and the incident radiance using a set of basis functions
- Orthonormal bases (such as SH) are ideal as they provide the useful property:

$$\int \tilde{f}(s)\tilde{g}(s)ds = \sum_{i=1}^{k} f_k g_k$$

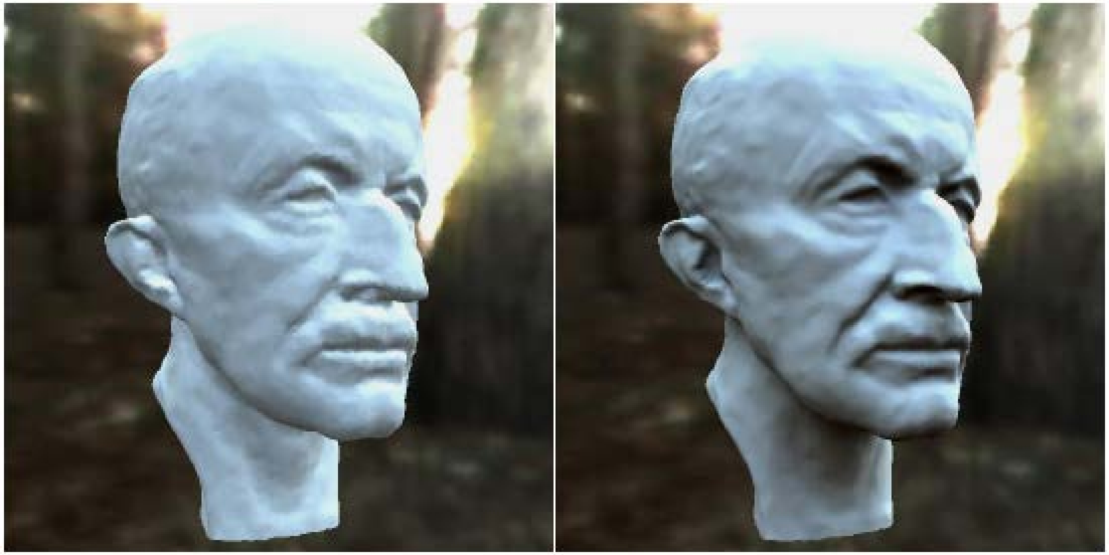
- i.e.: The integral of two band limited functions equals the dot product of their coefficients when projected to the orthonormal basis

Precomputed Radiance Transfer (1)

- The transfer (visibility over the hemisphere) function T can be precomputed and encoded in compact form
- When using Spherical Harmonics, 9 or 16 coefficients can effectively encode both T and L_i for diffuse light transfer
- The coefficients for T can be sparsely (pre-) evaluated, stored to and evaluated from:
  - A sparse lattice
  - A texture atlas

Precomputed Radiance Transfer (2)

$$L_r(\phi_r,\theta_r)=\frac{\rho}{\pi}\int_{\Omega_i} L(\phi_i,\theta_i)\cos\theta_i d\omega_i$$



$$L_r(\phi_r,\theta_r)=\frac{\rho}{\pi}\int_{\Omega_i} L(\phi_i,\theta_i)V(\phi_i,\theta_i)\cos\theta_i d\omega_i$$

Contributors

- Georgios Papaioannou