Spherical Harmonic Lighting: The Gritty Details

Robin Green

R&D Programmer

Sony Computer Entertainment America

What This Talk Is About

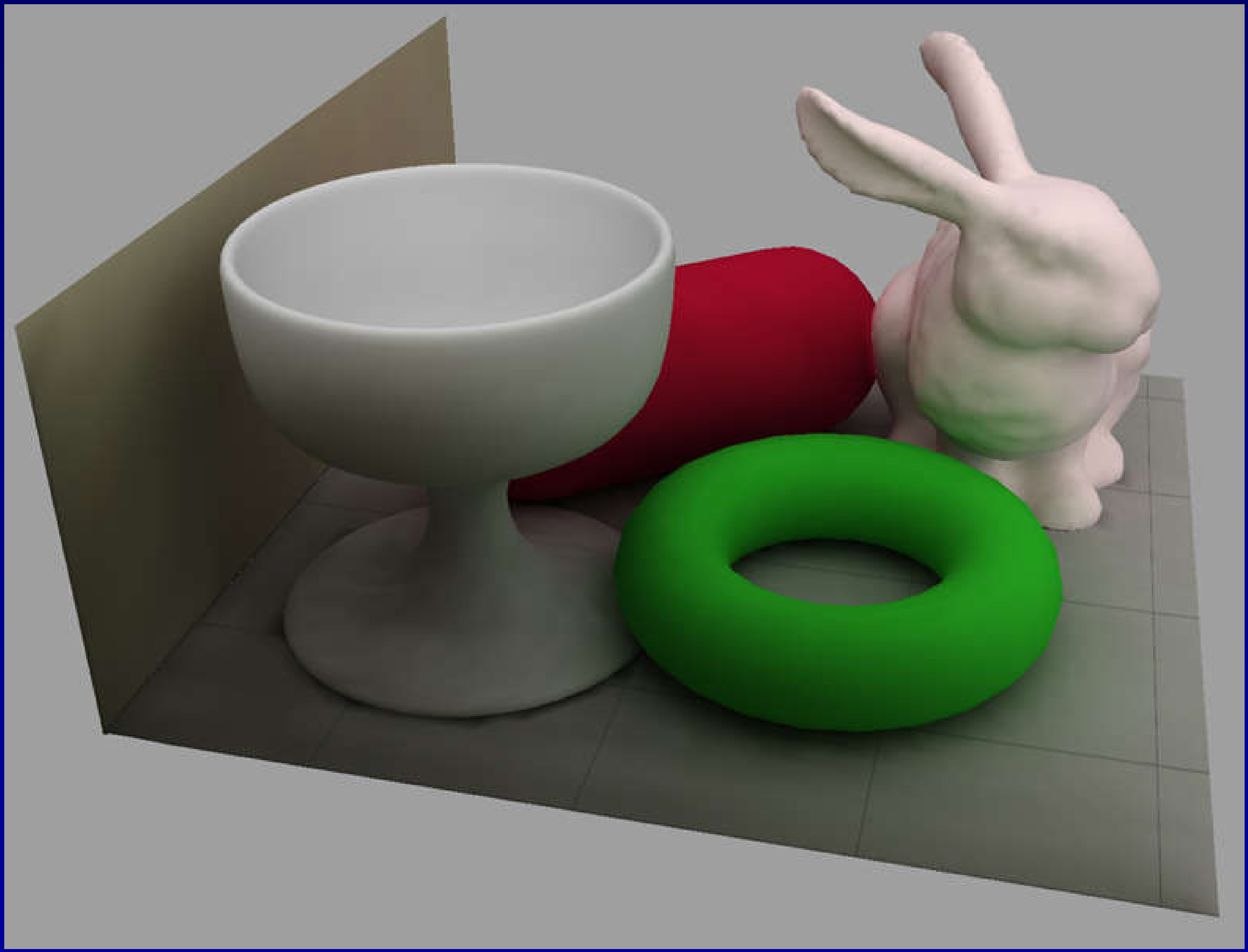
- **Advanced Lecture**
  - Explicit equations will be shown
- **This talk is one long HOWTO**
  - What SH Lighting is and how it works
  - Background to the difficult bits
  - How to write an SH Preprocessor
  - How to use SH Lighting in game
  - Example Source Code
  - Pretty Demonstrations

What is SH Lighting?

A efficient method for capturing and displaying Global Illumination solutions across the surface of an object

- Works on static models with dynamic lighting
- Extremely fast to render
- Independent of number or size of light sources
- High Dynamic Range lighting for free
- A drop-in replacement for diffuse lighting

Examples

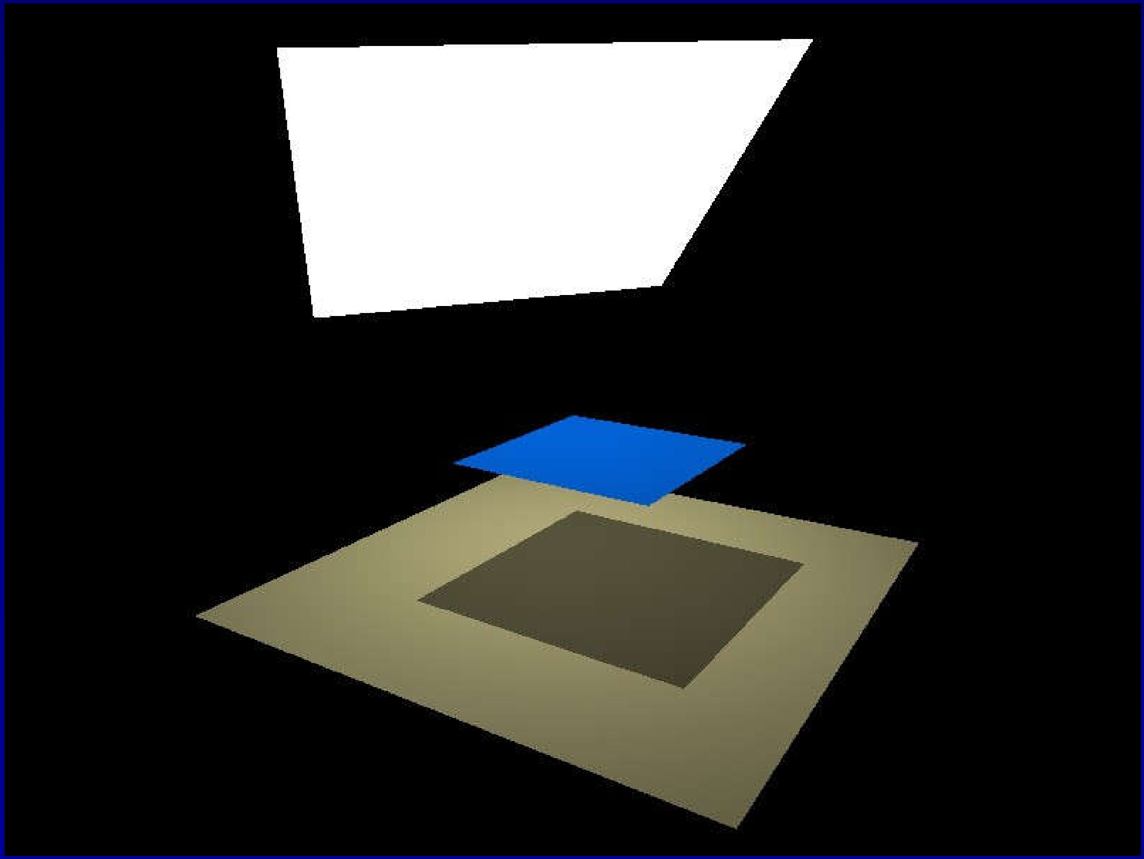


Examples

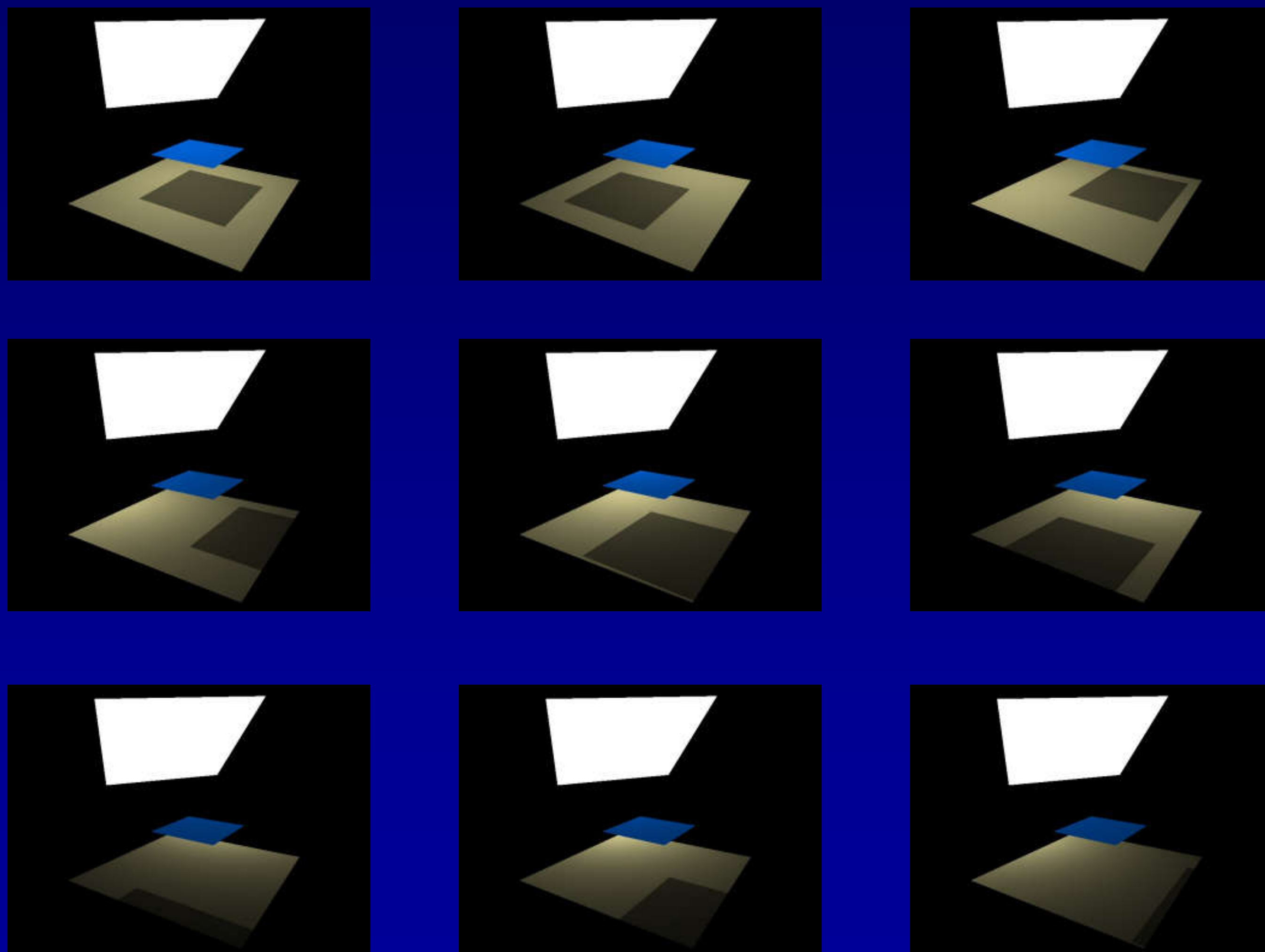
Examples

The Three Minute Pitch

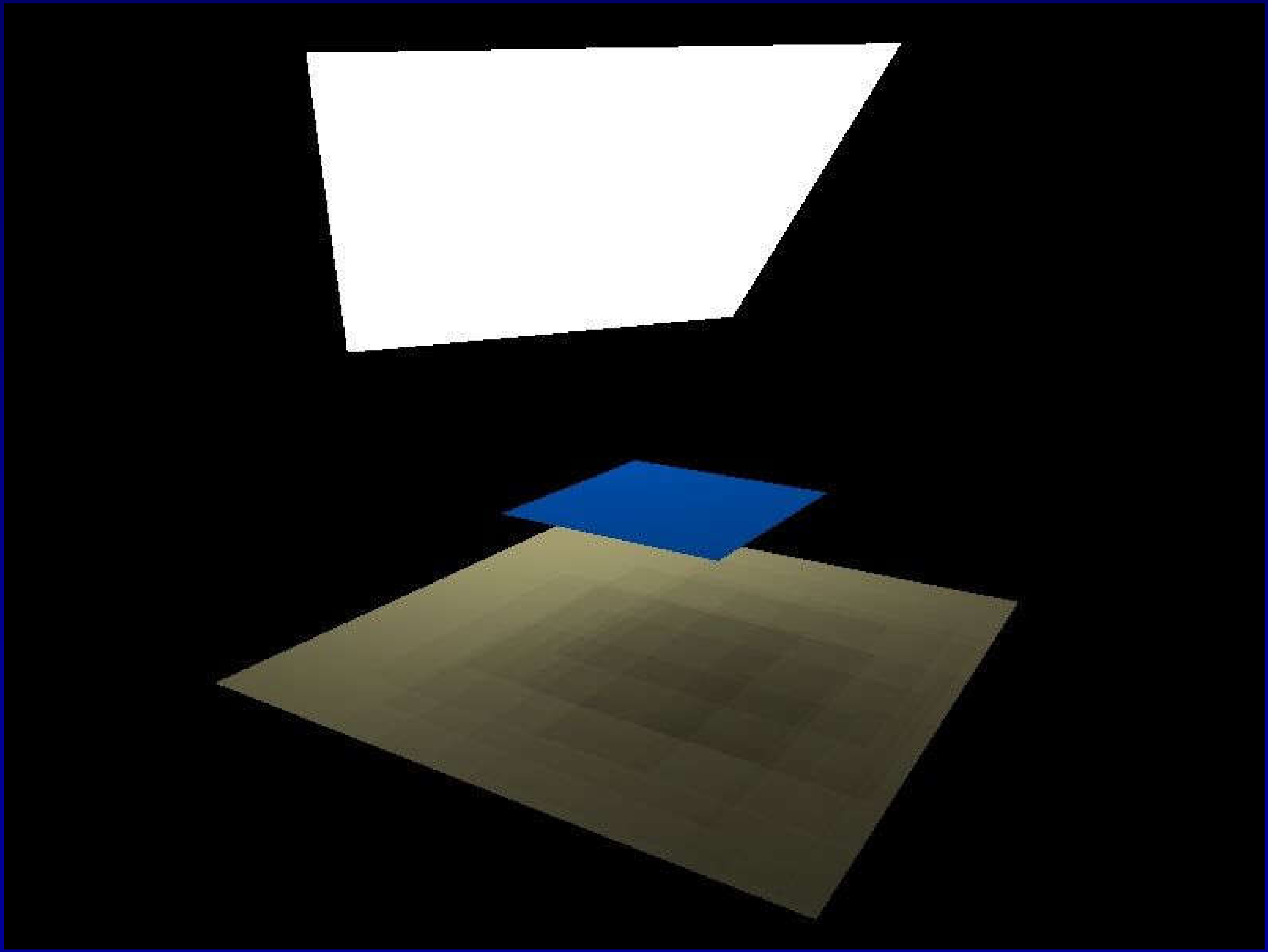
- Illuminating a scene with an area light sources should create soft shadows



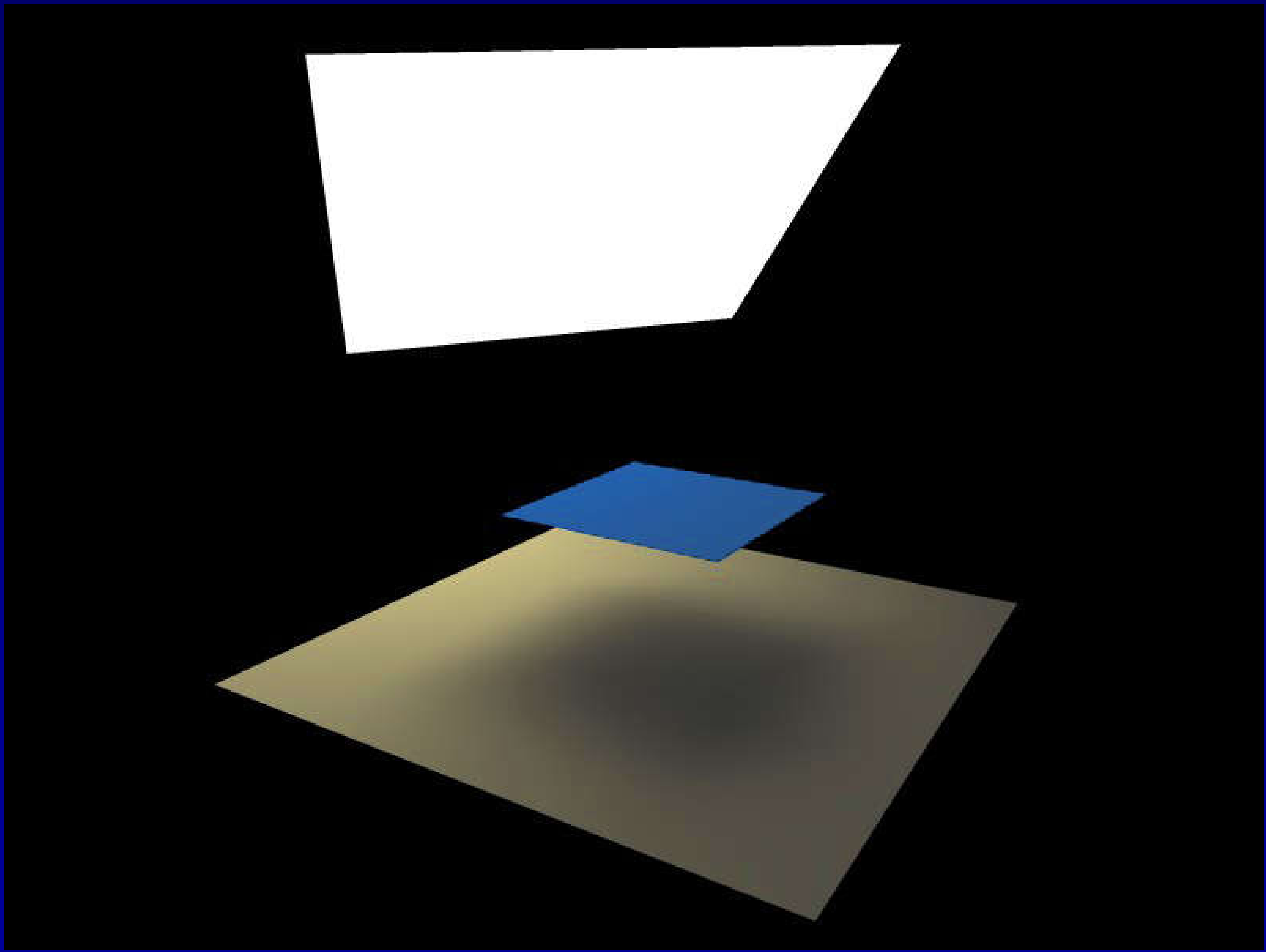
Area Light Sources



Area Light Sources



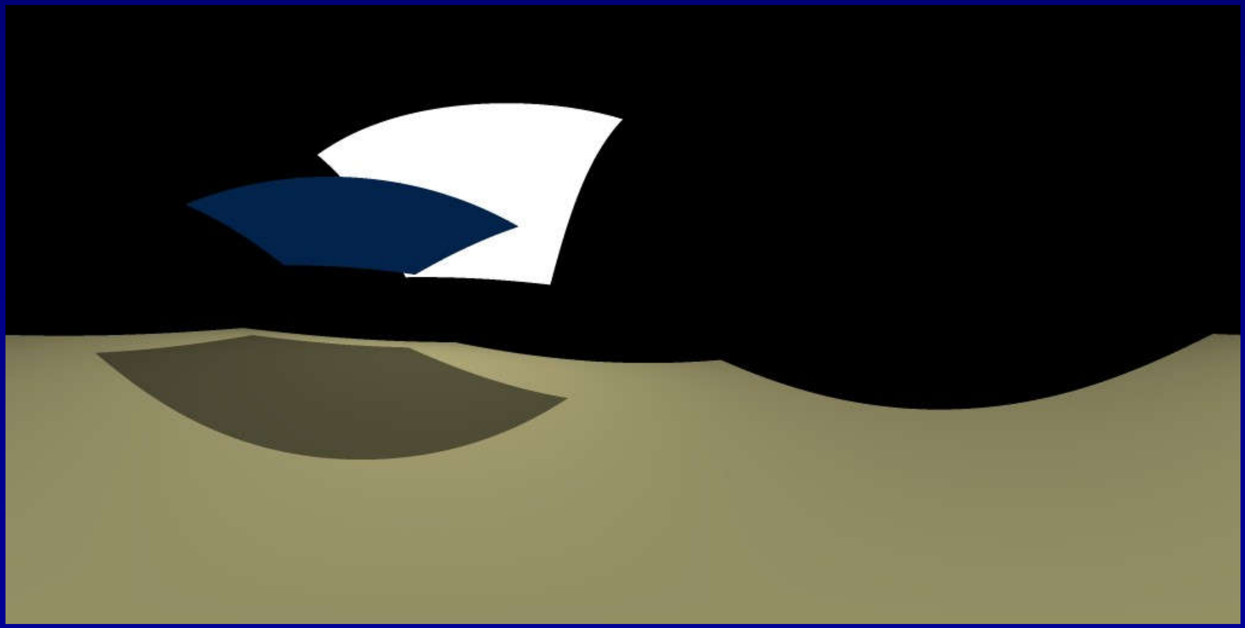
Area Light Sources



Another Viewpoint

Q: What can a point on the surface see?

A:



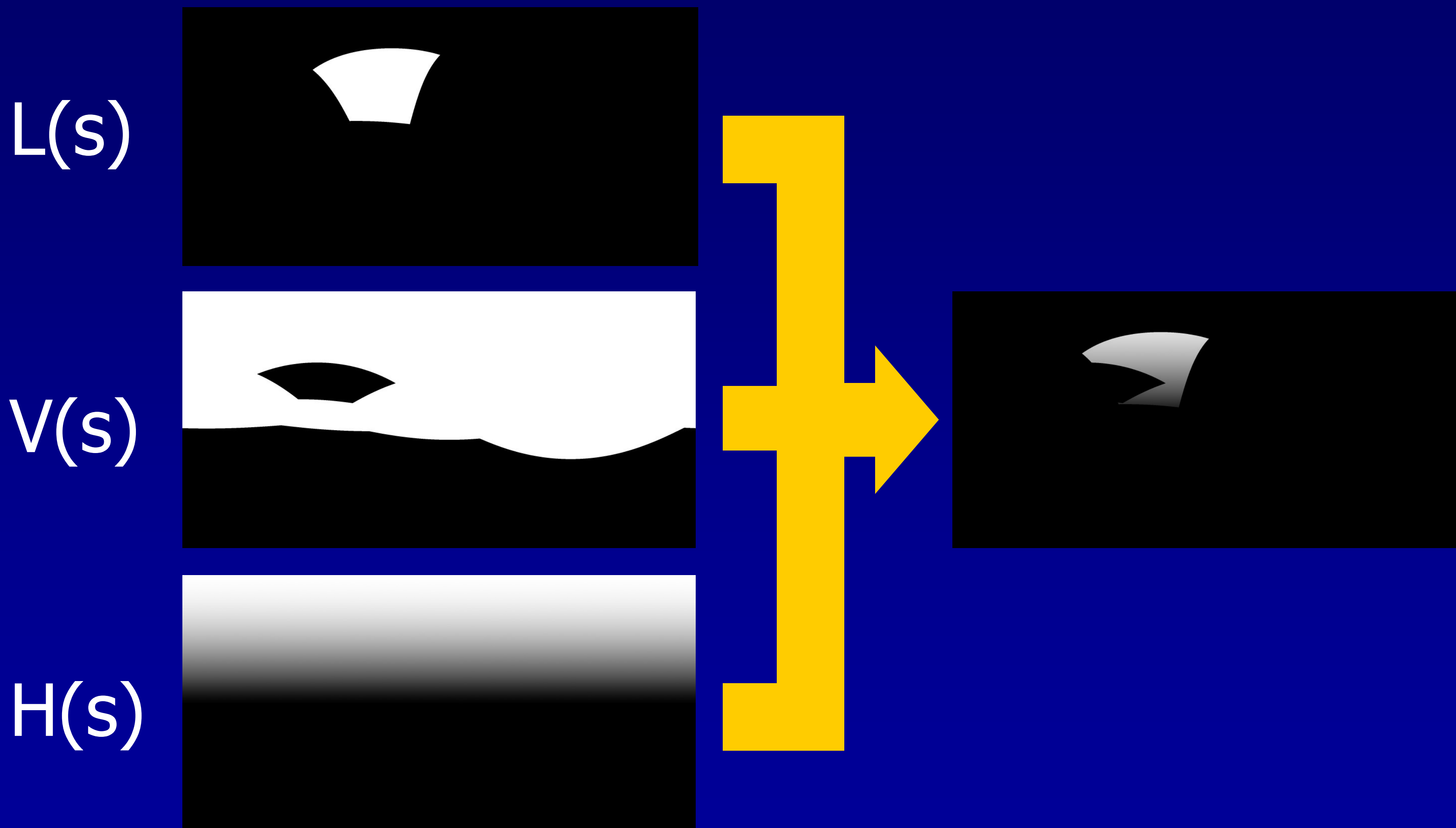
The Rendering Equation

- **The rendering equation tells us that:**

$$I_P = \int_S L_i(s) V_P(s) H_{\mathbf{N}_P}(s)\, ds$$

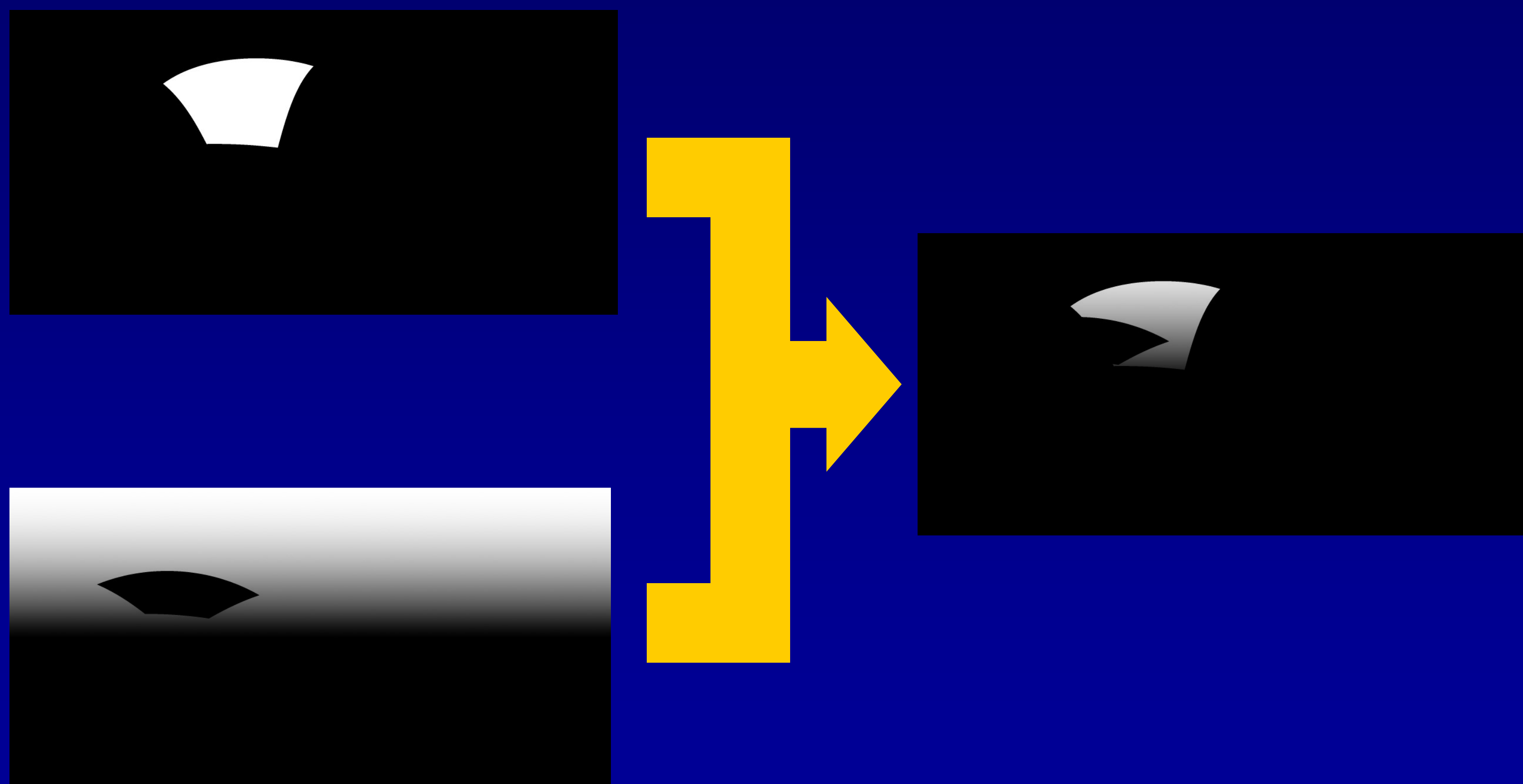
Reflected Light Intensity	=	Incoming Light Intensity	x	Visibility of Light Source	x	Hemisphere Cosine Term

Light as a Spherical Signal



The Transfer Function

L(s)



$$T(s) = V(s)*H(s)$$

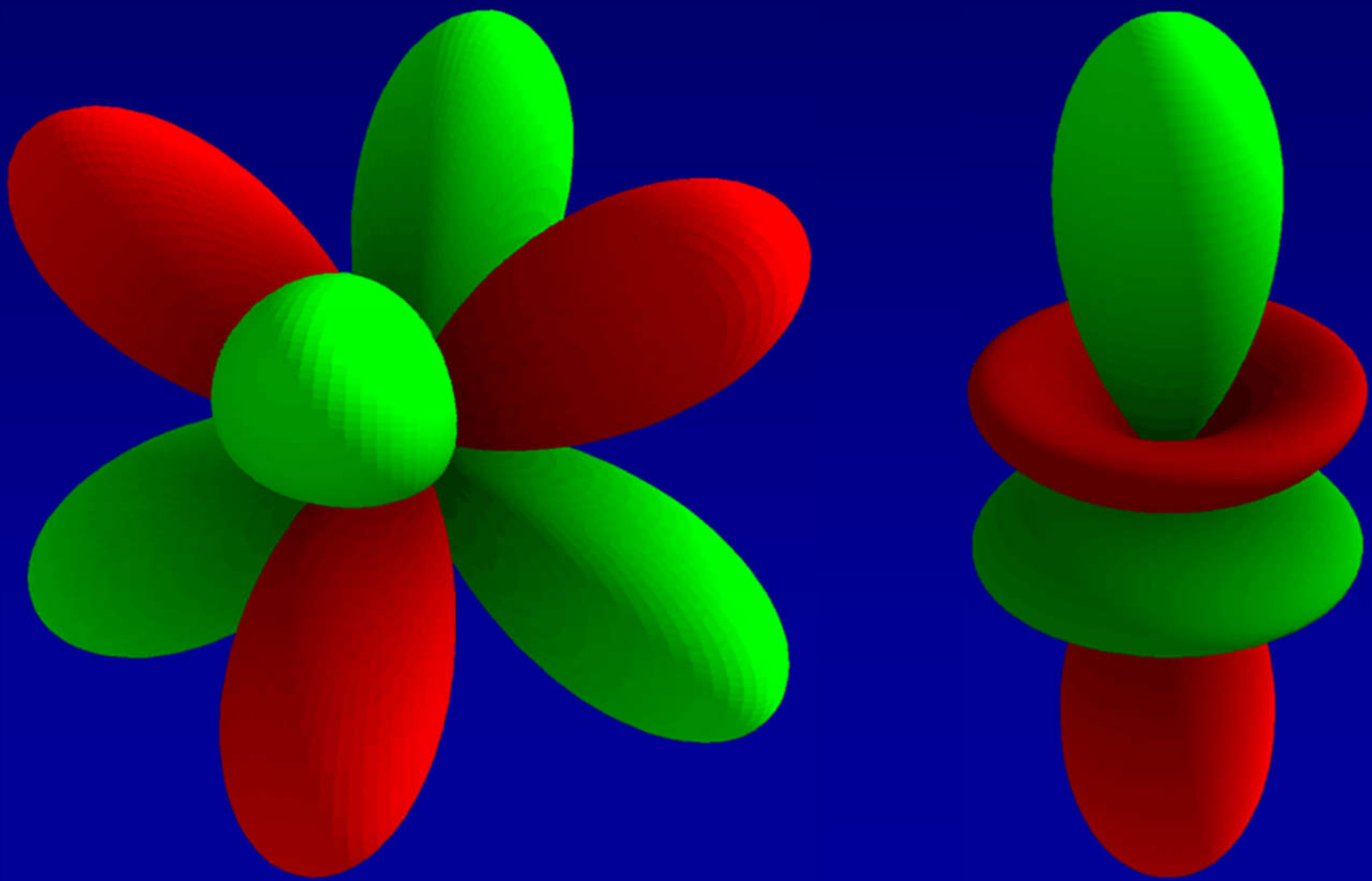
Encoding Spherical Functions

Two problems remain:

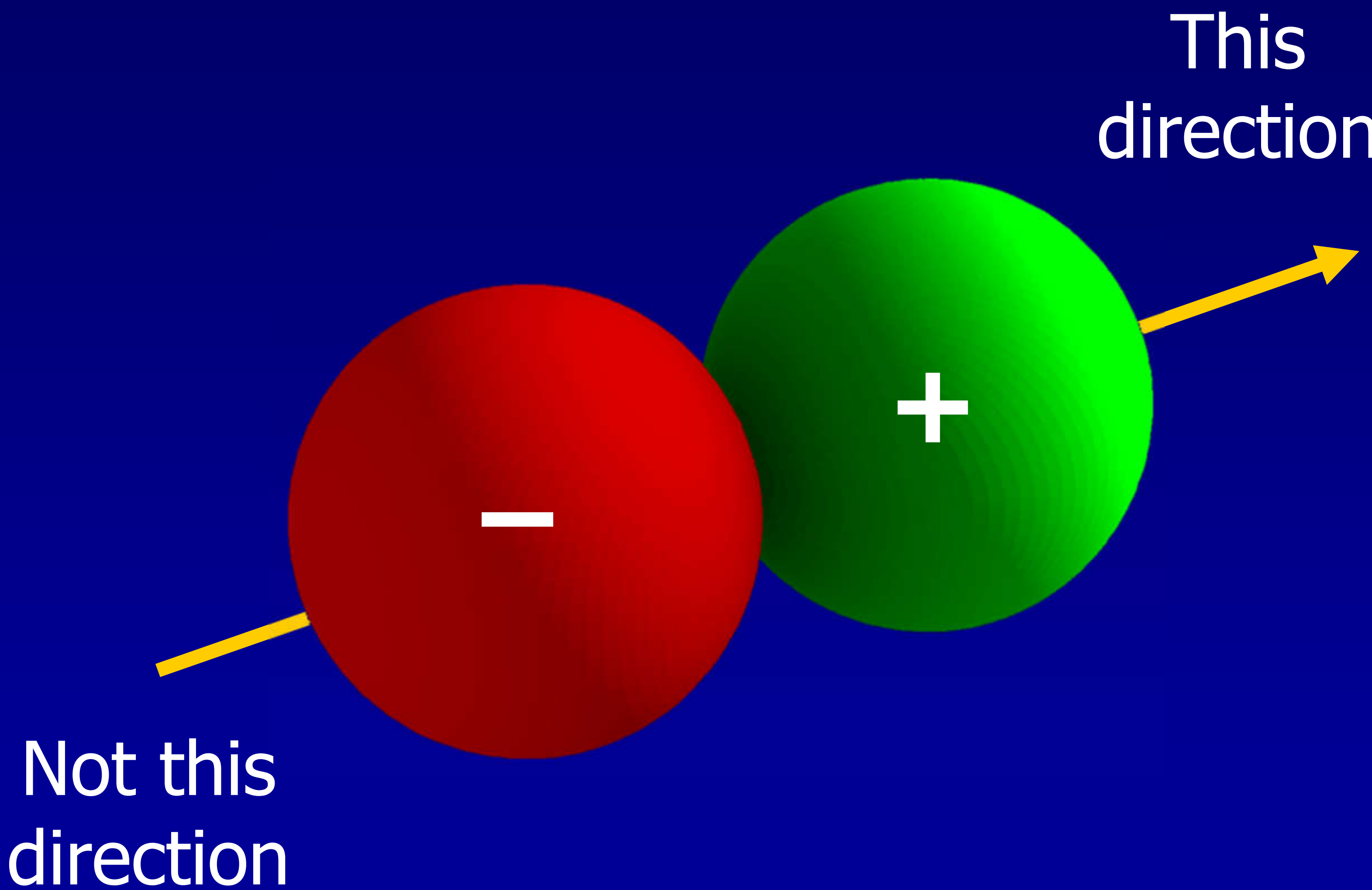
1. How to compactly represent functions over a sphere?

2. How to integrate over a sphere?

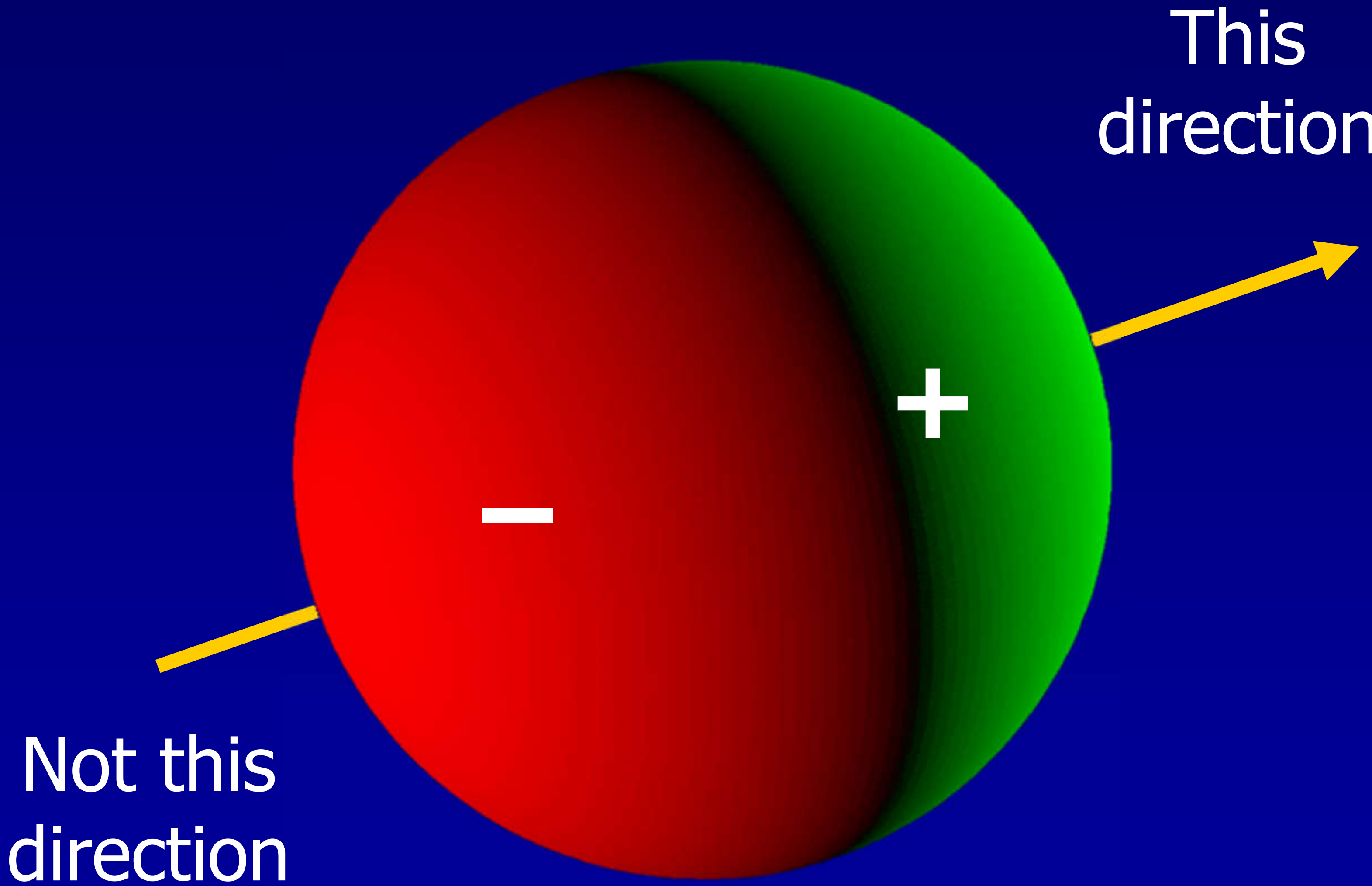
Real Spherical Harmonics



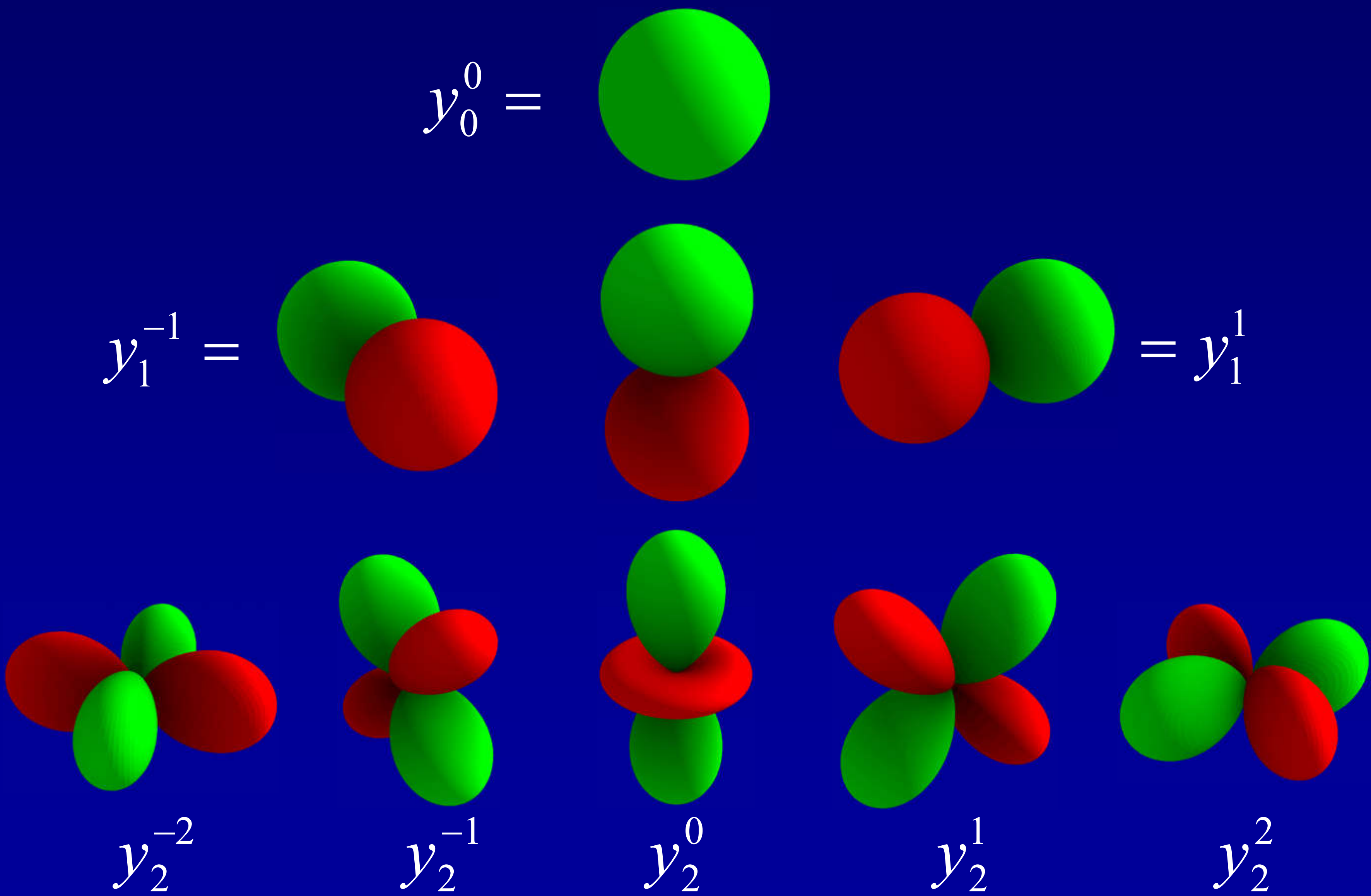
Reading SH Diagrams



Reading SH Diagrams



The SH Functions



SH Projection

- First we define a strict order for SH functions

$$i = l(l+1) + m$$

- Project a spherical function into a vector of SH coefficients

$$c_i = \int_S f(s) y_i(s) ds$$

SH Reconstruction

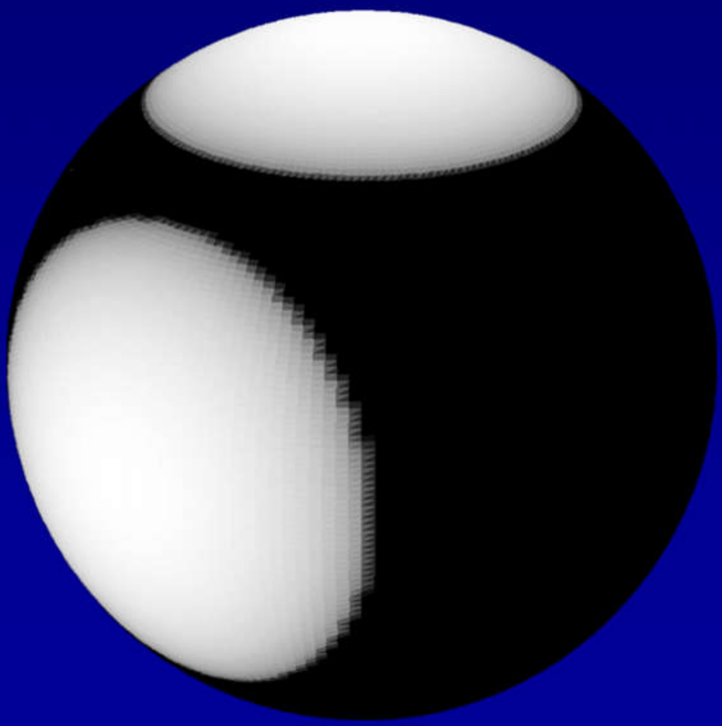
- To reconstruct the approximation to a function

$$\tilde{f}(s) = \sum_{i=0}^{N^2} c_i y_i(s)$$

- We truncate the infinite series of SH functions to give a **low frequency** approximation

An Example

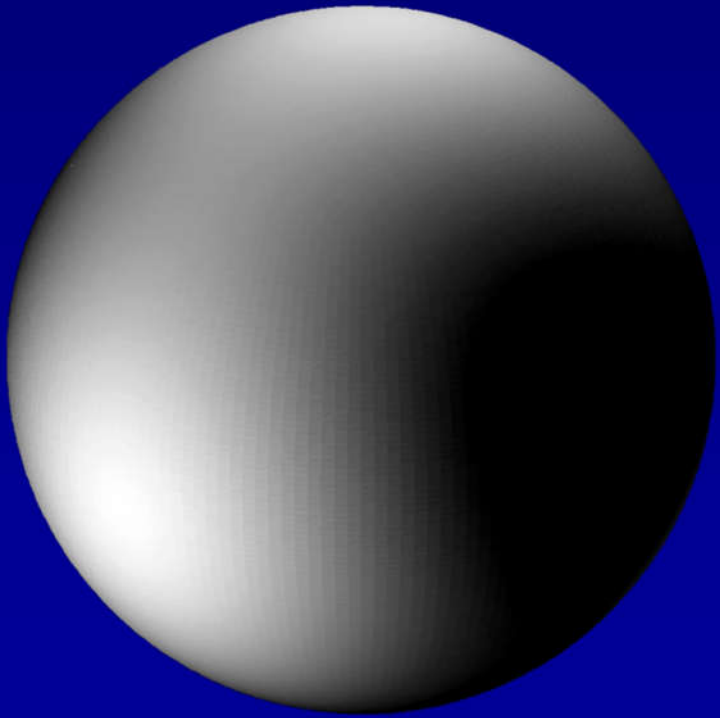
- **Take a function comprised of two area light sources**
  - SH project them into 4 bands = 16 coefficients

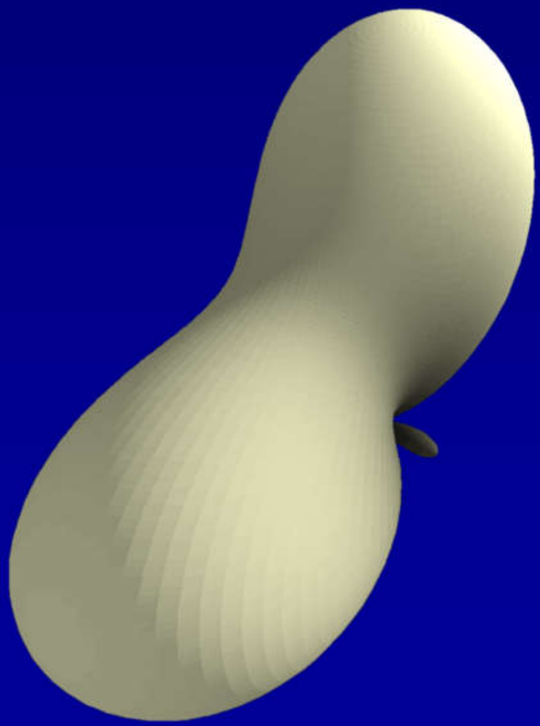


$$\begin{bmatrix} 1.329, \\ -0.679,\ 0.930,\ 0.908, \\ -0.940,\ 0,\ 0.417,\ 0,\ 0.278, \\ -0.642,\ 0.001,\ 0.317,\ 0.837, \\ -0.425,\ 0,\ -0.238 \end{bmatrix}$$

Low Frequency Light Source

- **We reconstruct the signal**
  - Using only these coefficients to find a low frequency approximation to the original light source





Here Is The Trick

- If both the light source and transfer function are expressed as vectors of SH coefficients, then the lighting integral

$$I_P = \int_S L(s)\, T_P(s)\, ds$$

can be calculated by a dot product between the coefficients

$$I_P = \mathbf{L} \cdot \mathbf{T}_P$$

What This Means

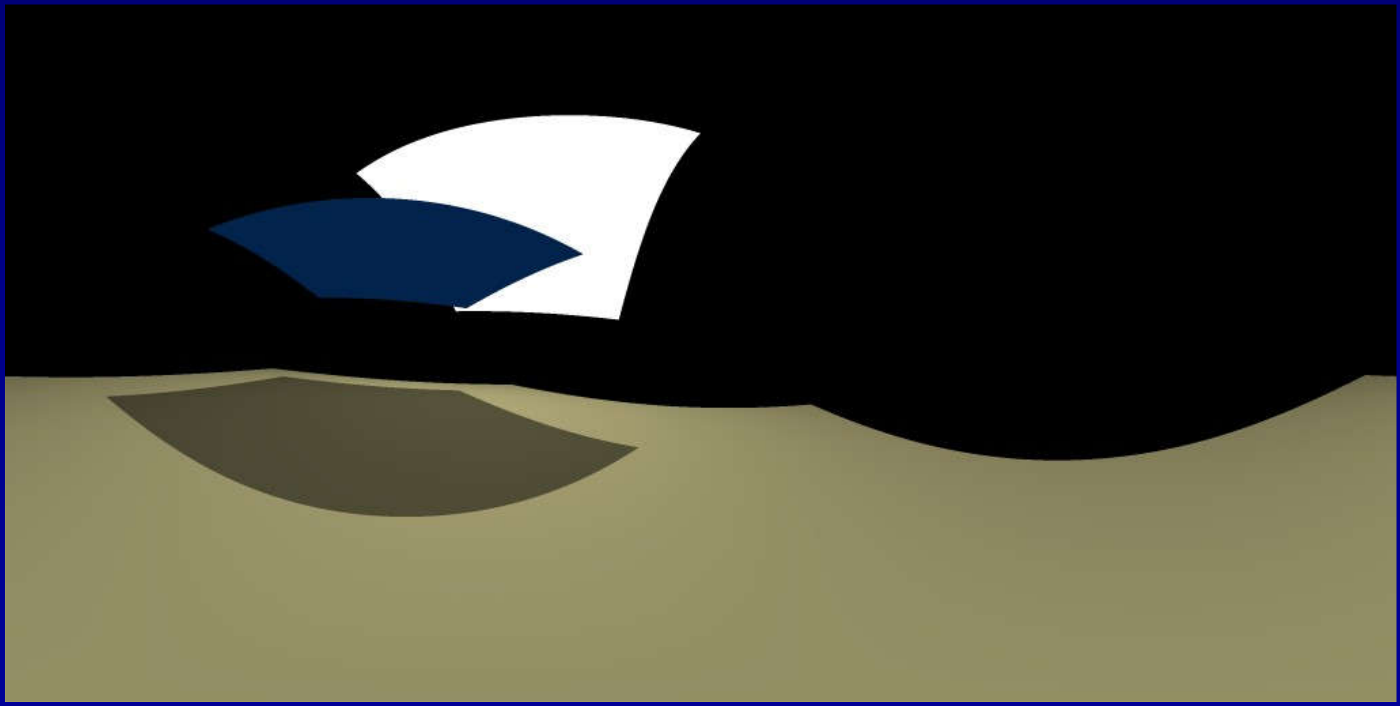
1. Lighting calculation is **independent** of the **number or size** of light sources

2. **Soft shadows are cheaper** than hard edge shadows

3. Transfer functions can be calculated as an offline **preprocess**

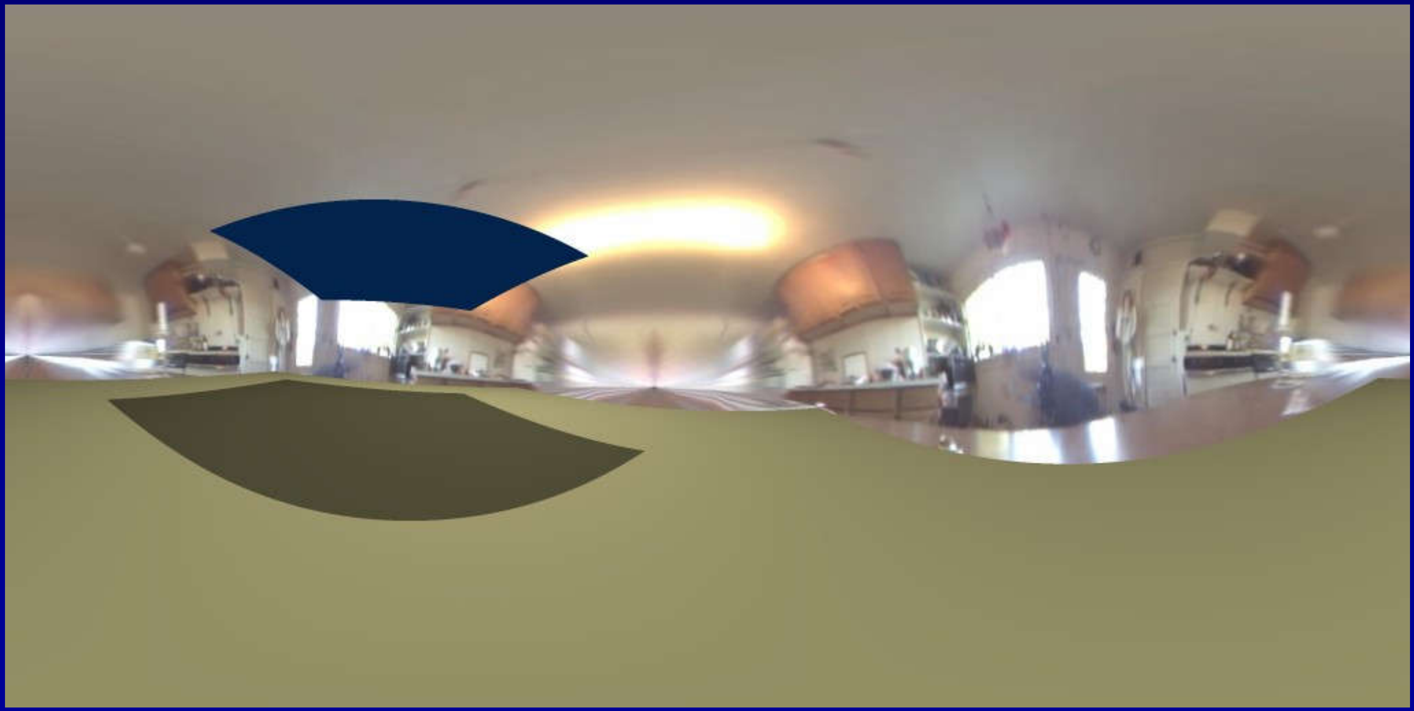
Extending the Lighting Model

- Look again at what the surface point can see



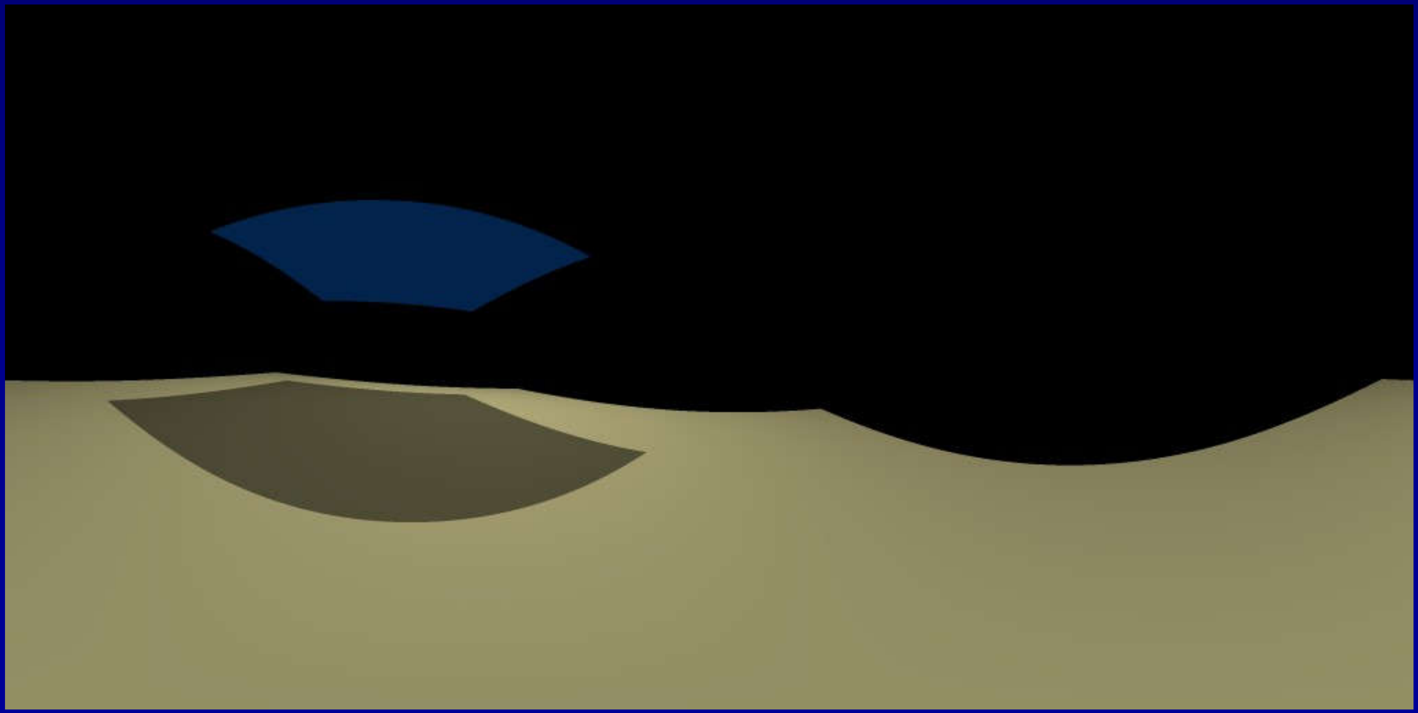
Complex Lighting Functions

- We can replace the lighting function with an HDR light probe for no extra cost



Secondary Illumination

- **The lit scene also illuminates the point**
  - This is how we capture diffuse-diffuse color bleeding



Diffuse Global Illumination

Good Points

- SH coefficients are the perfect way to **pass around energy** in Global Illumination solutions
- After direct illumination is calculated, self transfer requires **no additional raytracing**

Bad Points

- Assumes that all distant points have the **same illumination function** (e.g. no shadows over half the object)

Part 2: The Gritty Details

Background

- **Based on the Siggraph 2002 paper:**

  Precomputed Radiance Transfer for Real-Time Rendering in Dynamic Low-Frequency Lighting Environments, by Sloan, Kautz and Snyder.

- **This SH Paper assumes a lot of background from a first time reader:**
  - Orthogonal Polynomials
  - Monte Carlo Integration
  - The Rendering Equation
  - Use of symbolic math packages like Maple or Mathematica

- **This is what we will cover in the rest of the talk**

Real Spherical Harmonics

- **The Real Spherical Harmonics**
  - A system of signed, orthogonal functions over the sphere
  - Represented in spherical coordinates by the function

$$y_l^m(\theta, \varphi)$$

  - Where l is the band and m is the index within the band

Properties of SH Functions

- **SH Functions are fast to calculate using…**
  1. Spherical Coordinates using recurrence relations for P

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

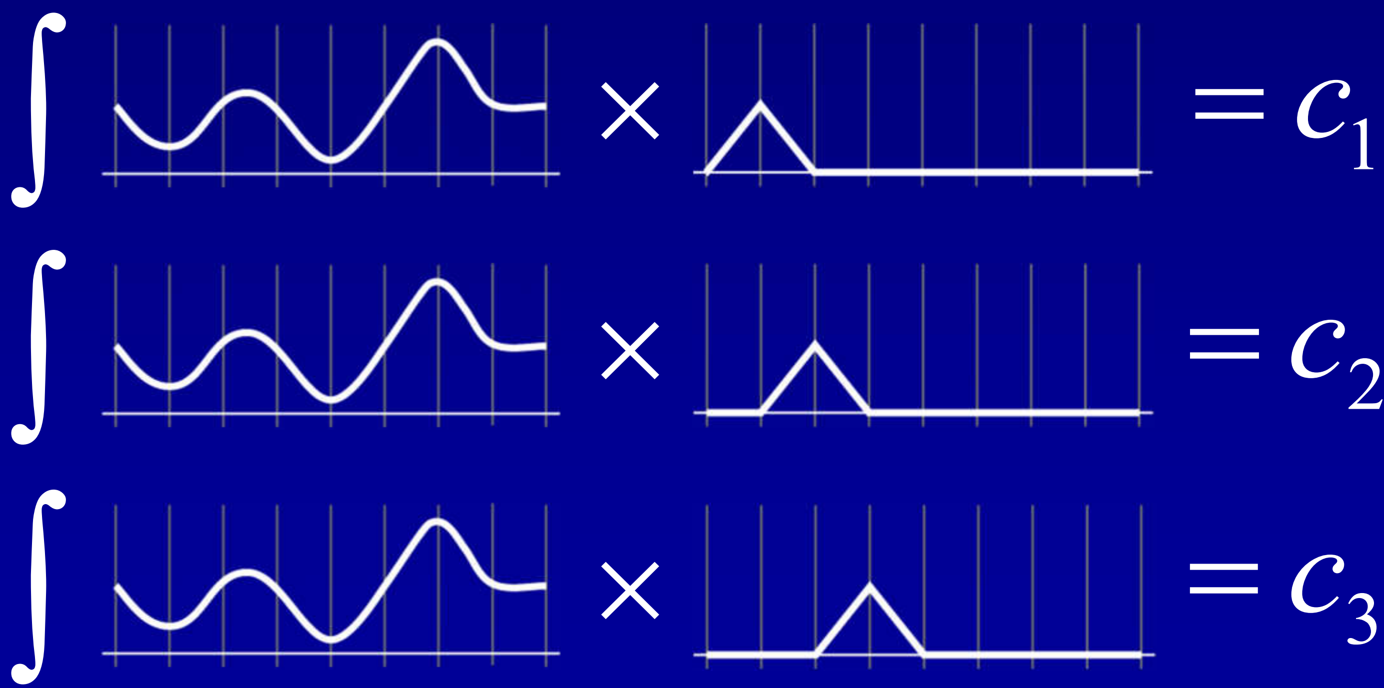
  2. Implicitly from Cartesian coordinates (x,y,z)

$$y_1^0=\frac{1}{2}\sqrt{\frac{3}{\pi}}z \qquad y_2^0=\frac{1}{4}\sqrt{\frac{5}{\pi}}\left(2z^2-x^2-y^2\right) \qquad y_2^2=\frac{1}{2}\sqrt{\frac{15}{\pi}}\left(x^2-y^2\right)$$

$$y_1^1=\frac{1}{2}\sqrt{\frac{3}{\pi}}x \qquad y_2^1=\frac{1}{2}\sqrt{\frac{15}{\pi}}zx \qquad \text{etc...}$$

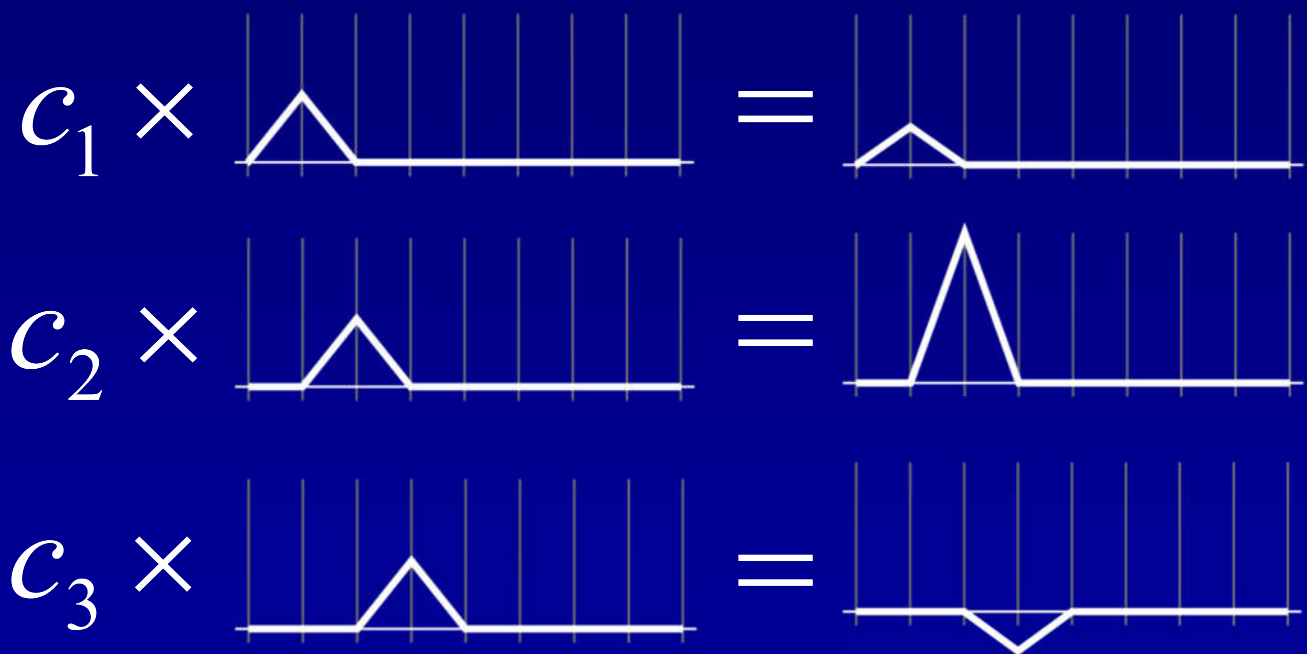
Properties of SH Functions

- **SH Functions are Basis Functions**
  - Basis Functions are pieces of signal that can be used to produce approximations to a function



Basis Functions

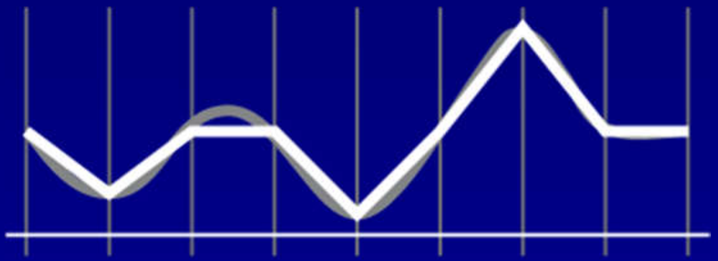
- We can then use these coefficients to reconstruct an approximation to the original signal



Basis Functions

- We can then use these coefficients to reconstruct an approximation to the original signal

$$\sum_{i=1}^{N} c_i B_i(x) =$$



- However, with SH lighting we mostly use operations that work directly on the coefficients themselves.

Orthogonal Basis Functions

- **SH functions are Orthogonal Basis Functions**
  - These are families of functions with special properties

$$\int B_i(x) B_j(x)\, dx = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

  - Intuitively, it's like functions don't overlap each other's footprint
    - A bit like the way a Fourier transform breaks a functions into component sine waves

Properties of SH Functions

- **SH Functions are Rotationally Invariant**
  - A rotated SH function is exactly the same as SH projecting the rotation of a function

$$Rot\left(SH\left(f\right)\right) = SH\left(Rot\left(f\right)\right)$$

  - This guarantees that under animation, lighting will not

    Crawl, Pulse,

    Flicker or Distort

  - Not a property that the Discrete Cosine Transform can claim

Monte Carlo Integration

- **How do we integrate over a sphere?**
  - We use Monte Carlo Integration, which is based on Probability theory
- **Probability refresher**
  - Take a random variable, say rolling a 6-sided die
  - The probability of rolling a specific value is 1/6
  - The Cumulative Density Function is the probability of rolling a value less than equal to x, e.g.

$$P(4) = P(1) + P(2) + P(3) + P(4)$$

$$= 4 \times \frac{1}{6} = \frac{2}{3}$$

Continuous Random Variables

- **Most values in this world are not discrete**
  - For example picking an angle between $(0..\pi)$
  - The probability of picking an exact value is zero
  - We can find the probability of finding a value within the range [a..b] by

$$P(a \le x \le b) = \int_a^b p(x)\,dx$$

  - The Probability Density Function $p(x)$ is defined as the derivative (or rate of change) of the CDF

Probability Density Functions

- **PDFs have some interesting properties**
  - All values in a PDF are positive or zero

$$P(a \leq x \leq b) \geq 0$$

  - All PDFs integrate to 1 over the entire range of inputs

$$\int_{-\infty}^{\infty} p(x)\, dx = 1$$

  - Random variables or functions are said to be distributed according to a given PDF

$$f(x) \sim p(x)$$

Expected Value of a Function

- Given a function $f(x) \sim p(x)$ the mean or expected value is calculated by

$$E(f) = \int_{-\infty}^{\infty} f(x) p(x)\, dx$$

- Another way of calculating the expected value is to sum a lot of point samples of $f(x)$ and divide by the number of samples

$$E(f) \approx \frac{1}{N} \sum_{i=1}^{N} f(x_i)$$

The Monte Carlo Estimator

- **We can combine these two results**
  - One of the sneakiest tricks in Engineering Mathematics

$$\int f(x)\,dx = \int \frac{f(x)}{p(x)}\,p(x)\,dx$$

$$\approx \frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)}{p(x_i)}$$

  - All we need to estimate the integral of an unknown function is to control how our **random samples are distributed**
    - weight the result by this probability

SH Projection

- **In order to SH Project a lighting function**
  - Evaluate the SH functions at random points on the sphere
  - Sum the product of the lighting function and the SH values

$$c_i \approx \frac{1}{N} \sum_{j=1}^{N} \frac{f(x_j) y_i(x_j)}{p(x_j)}$$

  - If we can guarantee the samples are uniformly distributed, we can move the weight outside of the sum

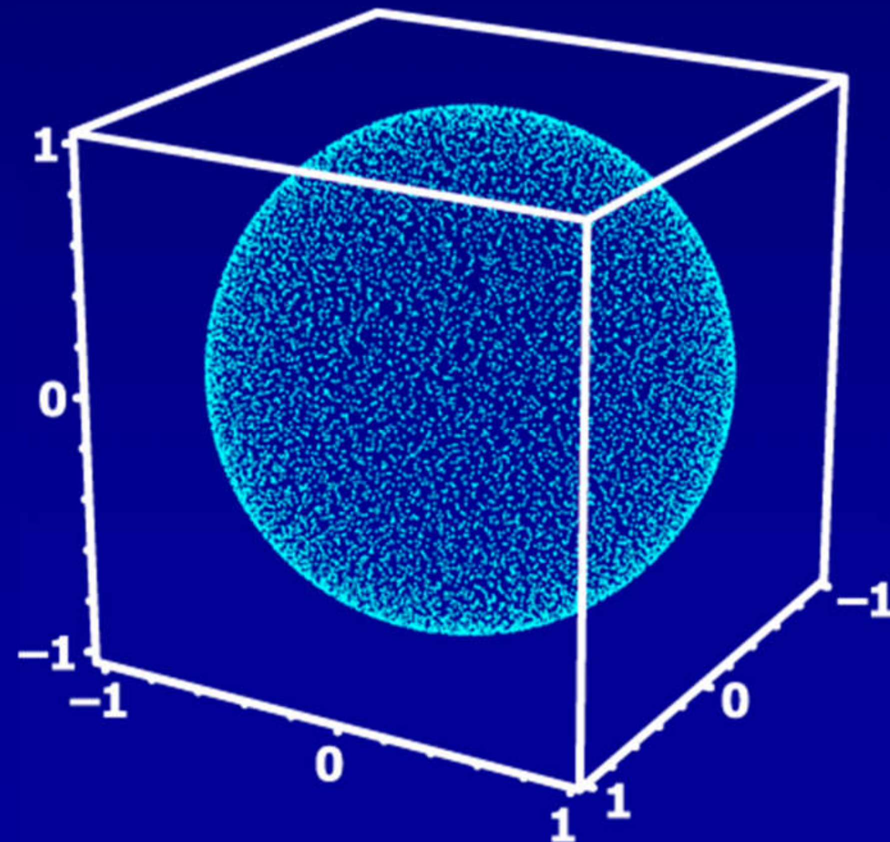
$$c_i \approx \frac{4\pi}{N} \sum_{j=1}^{N} f(x_j) y_i(x_j)$$

Generating Uniform Samples

- **To generate unbiased points over a sphere**
  - Map points from a unit square (x,y) into spherical coords

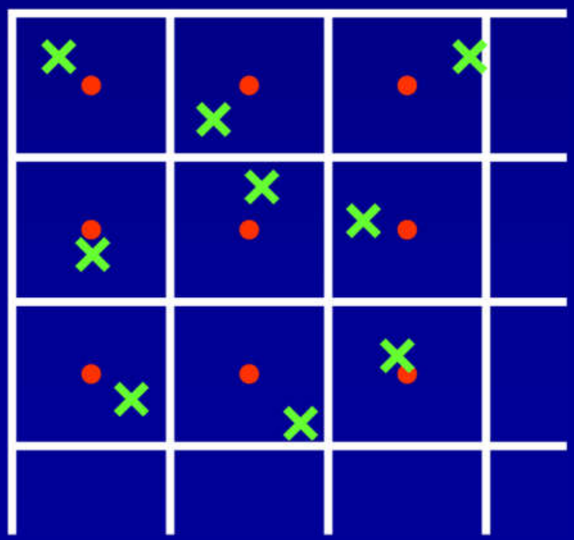
$$\theta = 2\arccos\left(\sqrt{1-x}\right)$$

$$\varphi = 2\pi y$$



Stratified Sampling

- **Monte Carlo integration suffers from noise**
  - Caused by high variance in our sampling scheme
    - The PDF is too different to the function we are integrating
  - To lower variance in our approximation we can use a jittered grid, sometimes called stratified sampling



The Rendering Equation

- **What are we trying to encode in SH coefficients?**
  - The simplified diffuse reflectance version of the Rendering Equation

$$L_{DS} = \frac{\rho}{\pi} \int_S L_i(s) V(s) H_{\mathbf{N}}(s) ds$$

Diffuse Shadowed light at point p

Surface reflectance (albedo) at point p

Incoming light intensity

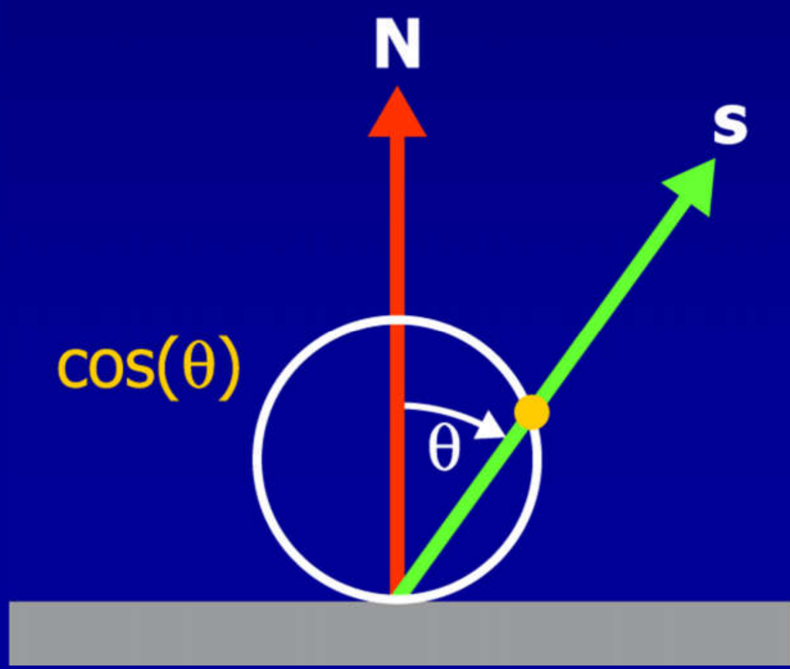
Visibility of incoming light

Hemisphere or Cosine Term

Cosine Term

- **Where does the cosine term come from?**
  - Essential to physically correct rendering
    - Comes from energy transfer formulation

$$H_{\mathbf{N}}(\mathbf{s}) = \max(\mathbf{N} \cdot \mathbf{s}, 0)$$



Transfer Function

- **Converting the rendering equation into two parts gives us the transfer function we are going to SH project**

$$T_P = V_P(\mathbf{s})\max(\mathbf{N}\cdot\mathbf{s},0)$$

  - The transfer function encodes the reflectance, the surface normal and the shadowing in one function
    - No need to store a surface normal per vertex

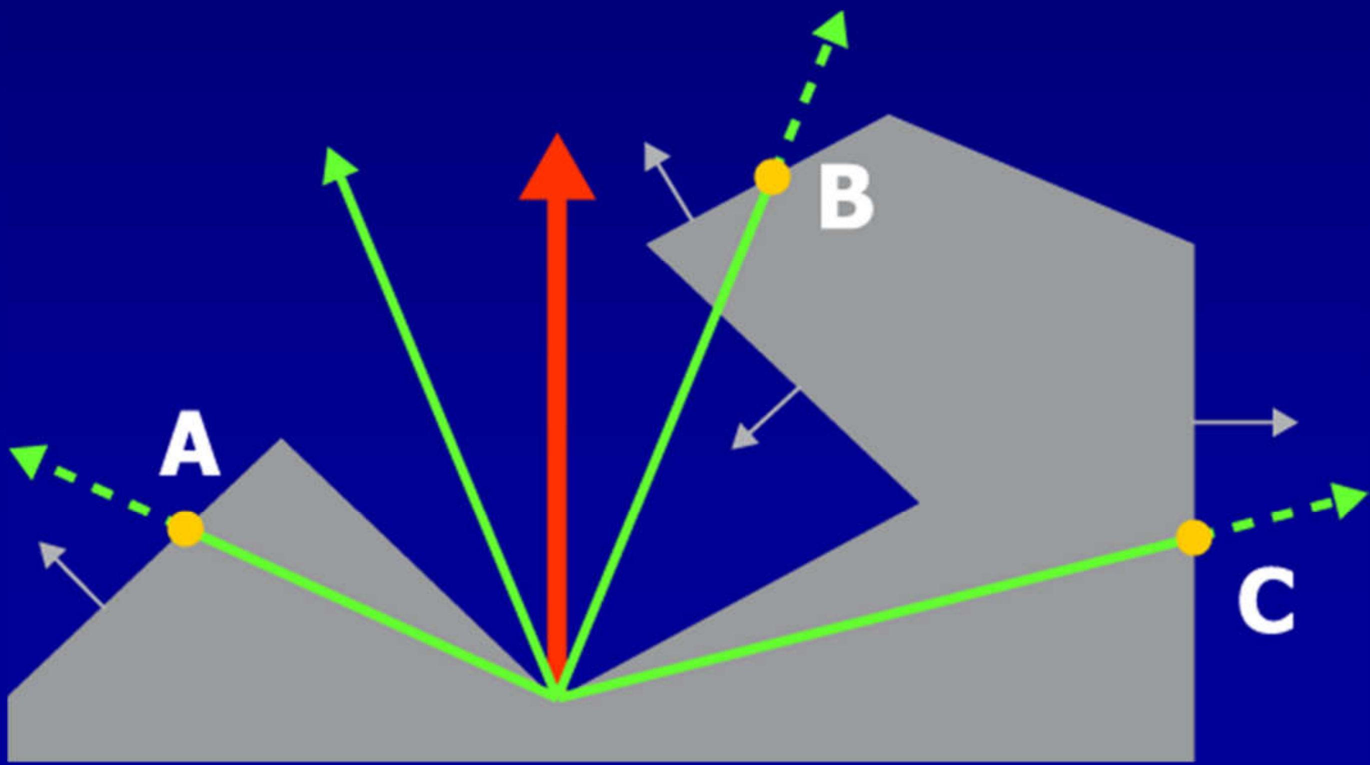
SH Preprocessing

- **Now we are ready to write an SH Preprocessor**
  - A simple raytracer that calculates SH coefficients for each vertex in our model

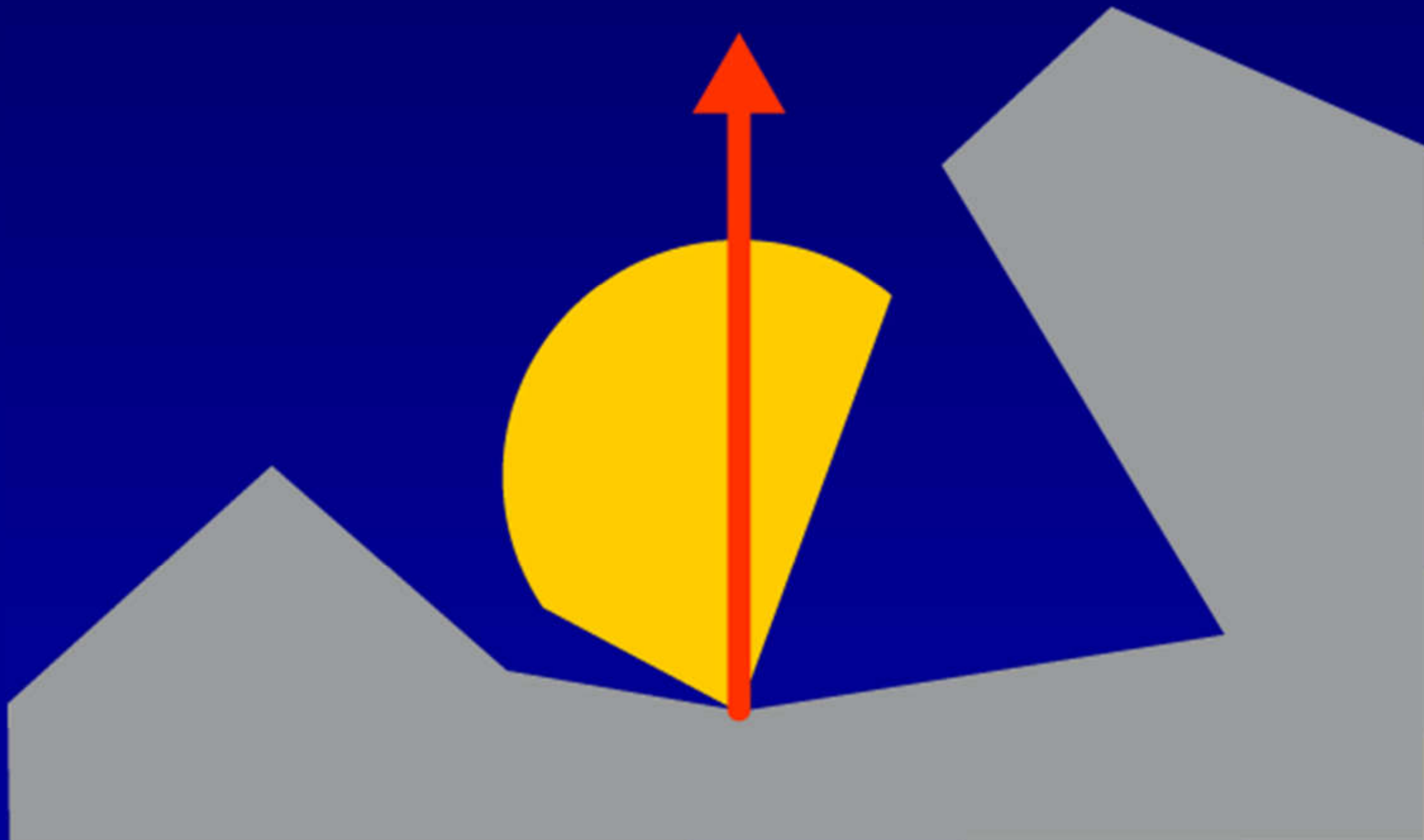
```
for(int i=0; i<n_samples; ++i) {
  double H = DotProduct(sample[i].vec, normal);
  if(H > 0.0) {
    if(!self_shadow(pos,sample[i].vec)) {
      for(int j=0; j<n_coeff; ++j) {
        value = H * sample[i].coeff[j];
        result[j] += albedo * value;
      }
    }
  }
}
const double factor = 4.0*PI / n_samples;
for(i=0; i<n_coeff; ++i)
   coeff[i] = result[i] * factor;
```

Raytracing Issues

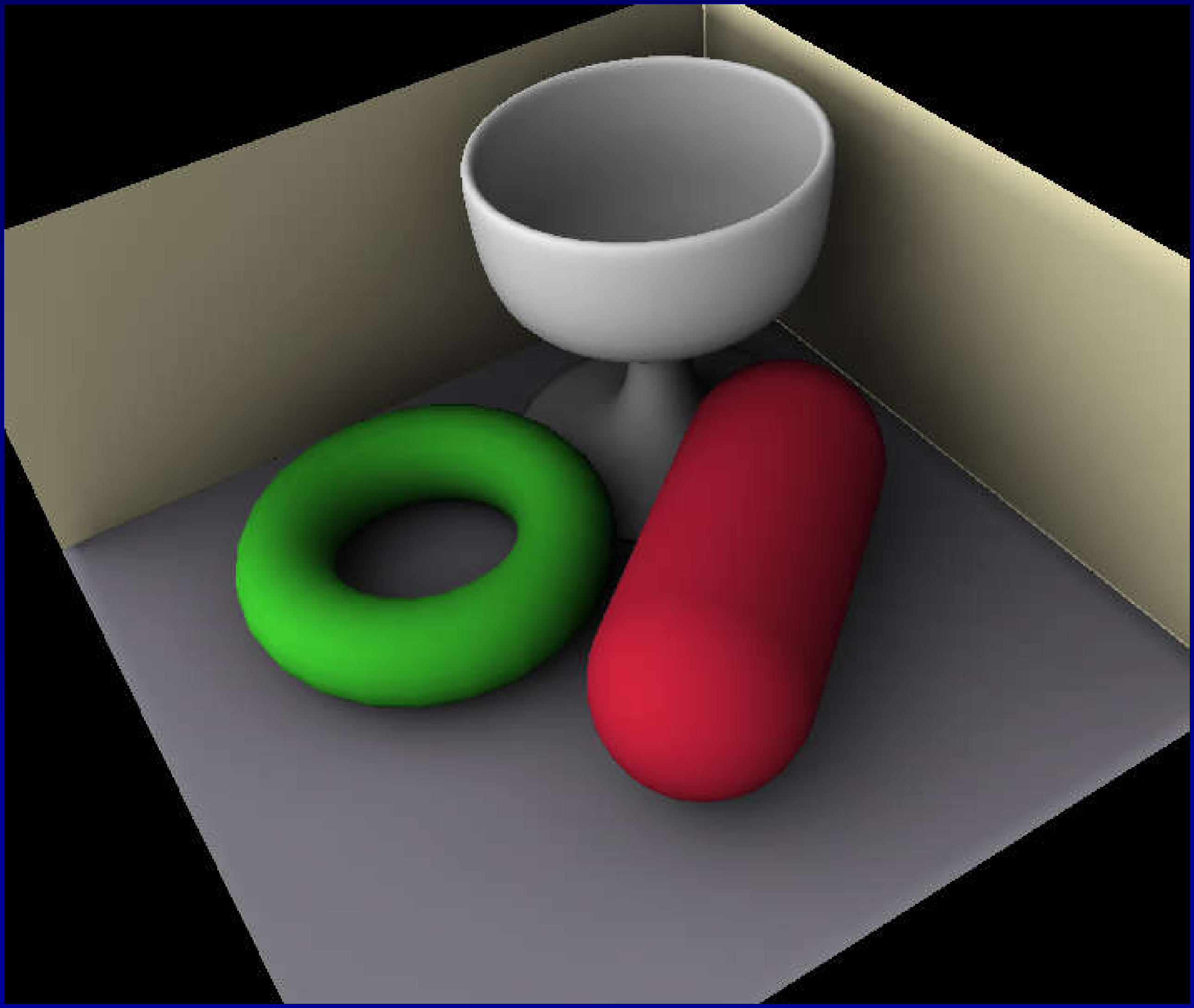
- **Shadow testing will often fire rays from inside a model**
  - Beware of single sided ray-triangle intersections
  - Prefer manifold models without holes



The Result



The Result



Self Transfer

- **We can do better**
  - The full rendering equation describes how lit surfaces also illuminate each other, leading to color bleeding

$$L_{DI} = L_{DS} + \frac{\rho}{\pi}\int_{S} \overline{L}_q(s)\left(1 - V(s)\right) H_{\mathbf{N}}(s)\,ds$$

Diffuse Inter-reflected light at point p

Diffuse Shadowed light

Surface Albedo

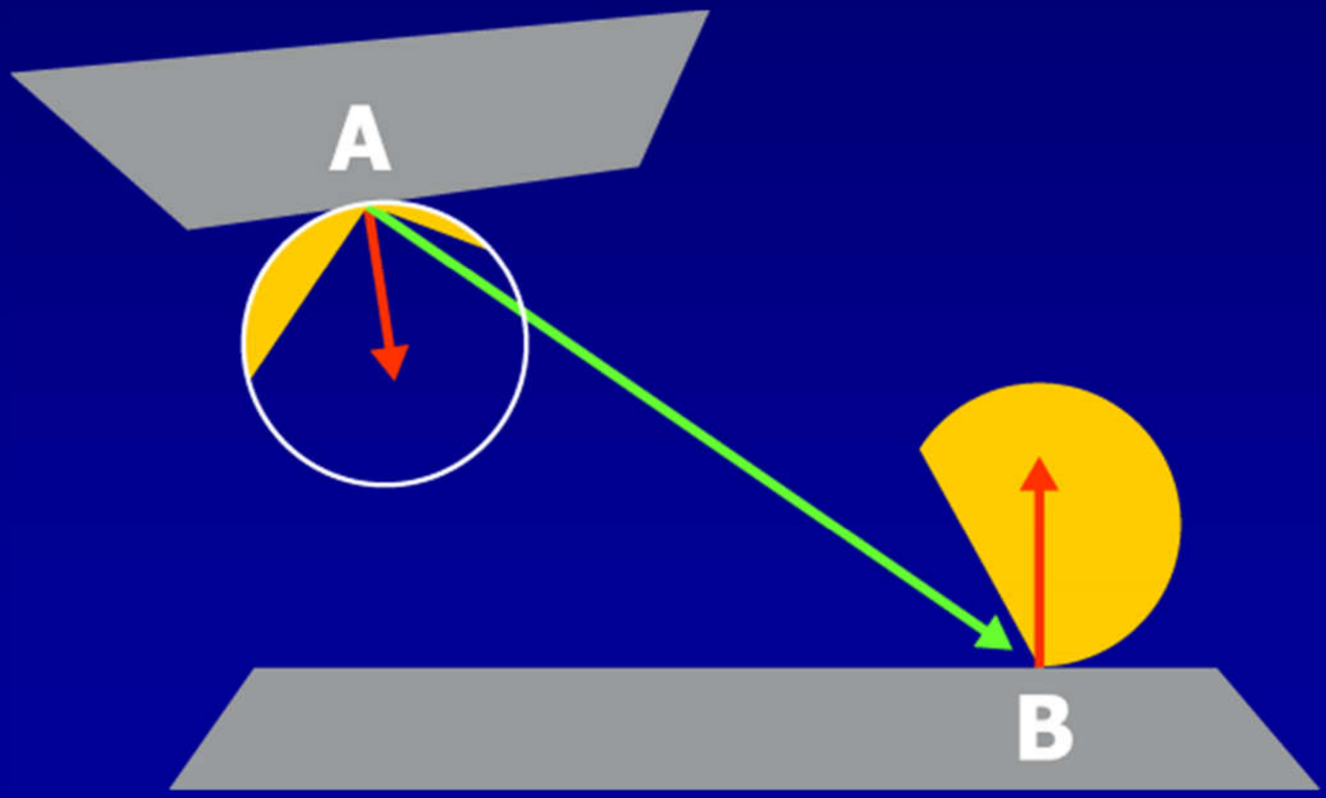
Light at point q from previous bounce

Visibility Term

Cosine Term

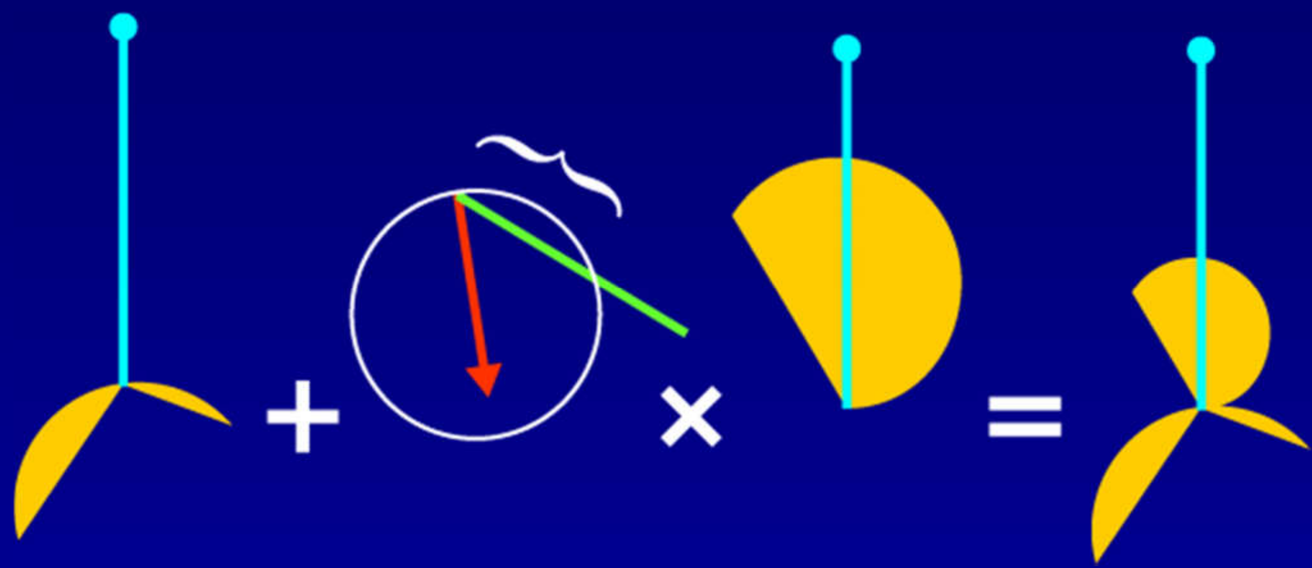
Self Transfer Diagram

- Point A receives illumination from point B proportional to the cosine term



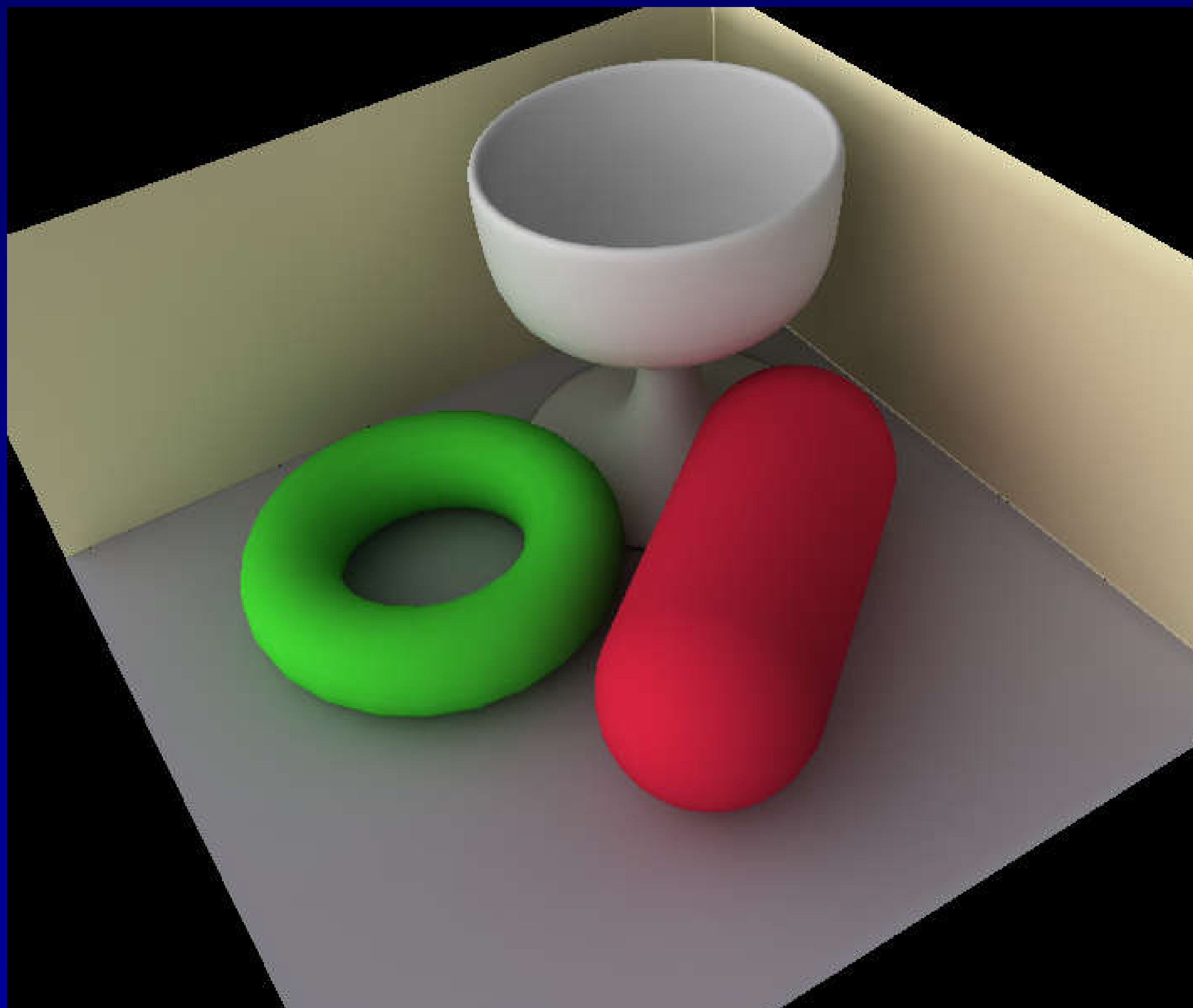
Self Transfer Equation

- **This can be expressed in graphically as**



- **Point A now receives light from above**
  - even though that direction is technically not visible to it
  - Using SH lighting, light transfer is just a series of multiply adds

The Result



SH Lighting at Runtime

- **Using SH Coefficients at runtime**
  - There are many ways to use the SH coefficients to reconstruct an image depending on what you want to achieve
    - Monochrome lights or Colored lights
    - Recolorable surfaces, fixed color or self transfer

```
for(int j=0; j<n_coeff; ++j) {
  vertex[i].red   += light[j] * vertex[i].sh_red[j];
  vertex[i].green += light[j] * vertex[i].sh_green[j];
  vertex[i].blue  += light[j] * vertex[i].sh_blue[j];
}
```

  - The accuracy for the lighting function
    - More coefficients encode higher frequency signals

SH Lighting As A Tool

- **Light is an additive function**
  - We can use SH lighting for part of a lighting calculation
    - Use SH Lighting for the sky sphere and add a point light and hard shadows to model the sun
    - Use SH Lighting for the diffuse part of surface reflectance

SH Rotation

- **An essential tool is to be able to rotate SH Functions**
  - Because SH functions are orthogonal, there is a simple linear matrix that will rotate SH coefficients directly

$$\mathbf{v}' = \mathbf{v}\mathbf{R}_{SH}$$

  - The laws of orthogonality tell us that the matrix is block diagonal sparse

$$R_{SH} = \begin{bmatrix}
\mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & 0 & 0 & 0 & 0 & 0 & \cdots \\
0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & 0 & 0 & 0 & 0 & 0 & \cdots \\
0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & 0 & 0 & 0 & 0 & 0 & \cdots \\
0 & 0 & 0 & 0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \cdots \\
0 & 0 & 0 & 0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \cdots \\
0 & 0 & 0 & 0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \cdots \\
0 & 0 & 0 & 0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \cdots \\
0 & 0 & 0 & 0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{bmatrix}$$

SH Rotation Matrix

- **This matrix is a pain to calculate in the general case**
  - But if we only need N bands, we can optimize the problem
- **If we break the rotation into ZYZ Euler angles**

$$\mathbf{R}_{SH}(\alpha,\beta,\gamma)=\mathbf{Z}_{\gamma}\mathbf{Y}_{\beta}\mathbf{Z}_{\alpha}$$

$$=\mathbf{Z}_{\gamma}\mathbf{X}_{-90}\mathbf{Z}_{\beta}\mathbf{X}_{90}\mathbf{Z}_{\alpha}$$

  - Two constant, sparse matrixes of ±90° about the X axis
  - Rotation about Z is very simple to calculate symbolically

$$\mathbf{M}_{ij}=\int_{S} y_i(s\mathbf{R})\,y_j(s)\,ds$$

SH Rotation about Z

$$\mathbf{Z}_\alpha = \left[\begin{array}{c:ccc:ccccc}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hdashline
0 & \cos(\alpha) & 0 & \sin(\alpha) & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -\sin(\alpha) & 0 & \cos(\alpha) & 0 & 0 & 0 & 0 & 0 \\ \hdashline
0 & 0 & 0 & 0 & \cos(2\alpha) & 0 & 0 & 0 & \sin(2a) \\
0 & 0 & 0 & 0 & 0 & \cos(\alpha) & 0 & \sin(\alpha) & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -\sin(\alpha) & 0 & \cos(\alpha) & 0 \\
0 & 0 & 0 & 0 & -\sin(2\alpha) & 0 & 0 & 0 & \cos(2\alpha)
\end{array}\right]$$

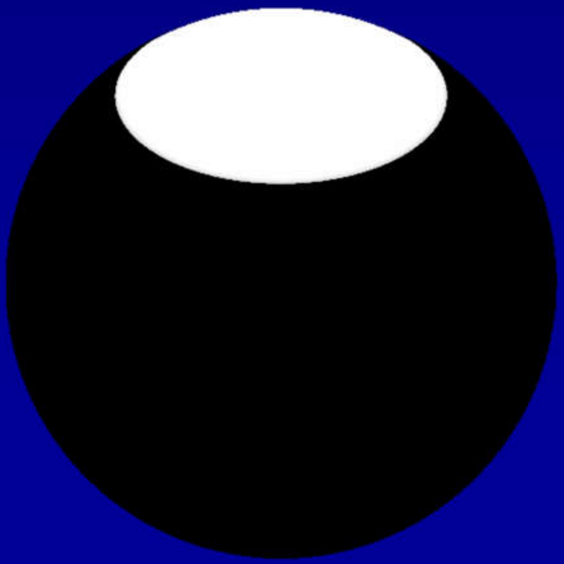
SH Rotation about X

$$\mathbf{X}_{90} = \left[\begin{array}{c:ccc:ccccc}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hdashline
0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
\hdashline
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & 0 & -\frac{\sqrt{3}}{2} \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\frac{\sqrt{3}}{2} & 0 & \frac{1}{2}
\end{array}\right]$$

SH Light Sources

- **Many ways of creating SH light sources**
  - Numerically from polar functions
  - Raytrace polygonal models or scenes
  - From HDR light probes or environment maps
  - Directly from analytical solutions
    - e.g. solution for a disk light source, angle t

$$d_t(\theta,\varphi)=\begin{cases}1 & (t-\theta)>0\\ 0 & otherwise\end{cases}$$



Analytical Disk Light Source

- Symbolically integrate this disk light function over a sphere in Maple or Mathematica to find the analytical expression

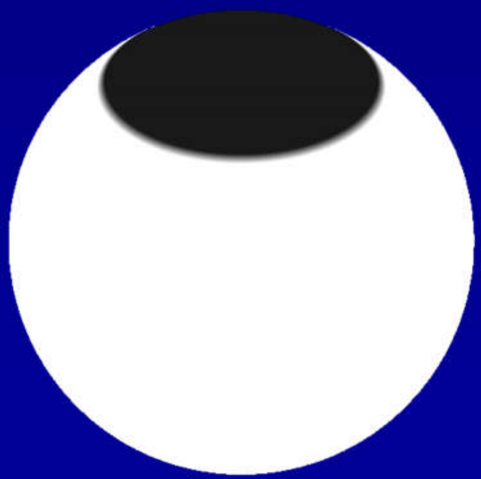
$$c_i = \int_{\varphi=0}^{2\pi} \int_{\theta=0}^{\pi} d_t(\theta,\varphi) y_i(\theta,\varphi) \sin\theta \, d\theta \, d\varphi$$

- Only 4 of the 25 coefficients in a 5 band disk light are non-zero
- Finally, use SH Rotation to position the light source

Proxy Shadows

- **A method of faking inter-object shadows**
  - If each object in a scene has it's own lighting function, we can use analytical "blockers" to subtract light from the direction of another object
  - Defining $b_t(s)$ as $1-d_t(s)$ allows us to construct a transfer matrix that masks SH coefficients

$$\mathbf{T}_{ij} = \int_S b_t(s) y_i(s) y_j(s)\, ds$$



Suggested by Alex Evans, Lionhead

Open Problems in SH Lighting

- **Faster SH Rotation methods**
  - Borrowing from research in Computation Chemistry
- **SH Lighting non-static objects**
  - As objects move relative to each other, the visibility function V(s) changes radically. How to encode this?
- **Exploiting the sparseness of SH vectors**
  - SH vectors often contain very few non-zero coefficients
- **Glossy Specular SH Lighting**
  - An elegant way to encode and use arbitrary BRDFs
  - Still too slow for the general case

Conclusion

- **SH Lighting is a new technique for lighting 3D models**
- **Brings area light sources and global illumination to real time games**
- **Works on any platform that can do Gouraud shading**
- **Can be used as a drop-in replacement for diffuse lighting on static scenes**
  - 2 band shadowing uses only 4 coefficients

For More Information

- **Document in Conference proceedings**
  - Visit the SCEA R&D website for a debugged version of the document

```
http://www.research.scea.com/
```

- **Notes and queries to**


```
robin_green@playstation.sony.com
```