The Rendering Equation

Rendering with Natural Light

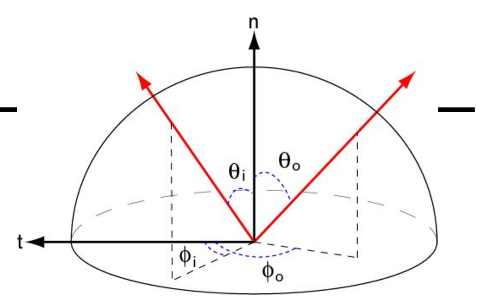
Paul Debevec et. al, SIGGRAPH 1998

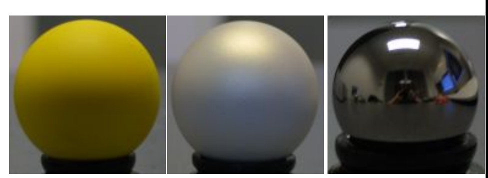
Image Based Lighting

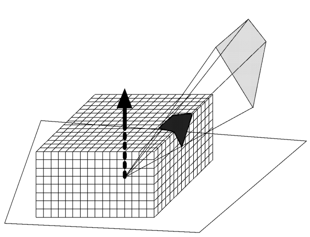
Paul Debevec et al, SIGGRAPH 2000

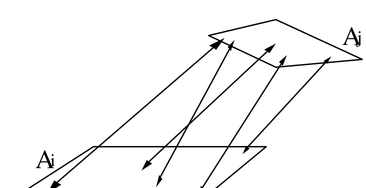
Last Time?

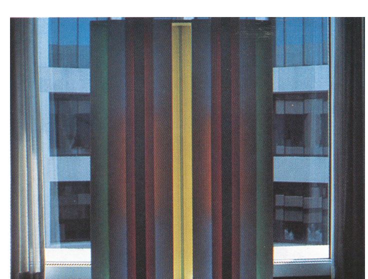
- Local Illumination
  - BRDF
  - Ideal Diffuse Reflectance
  - Ideal Specular Reflectance
  - The Phong Model
- Radiosity Equation/Matrix
- Calculating the Form Factors







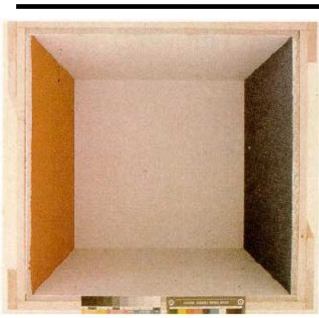




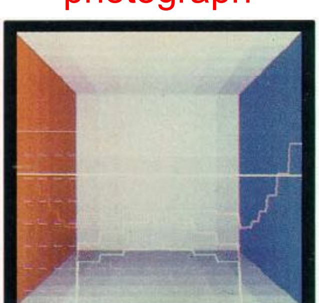
Today

- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors
  - Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Worksheet
- Papers for Next Time

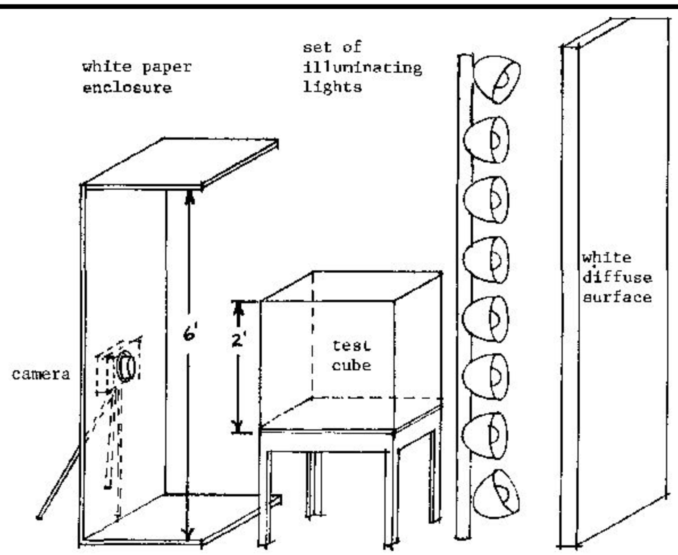
Reading for Today



photograph



simulation



Goral, Torrance, Greenberg & Battaile
Modeling the Interaction of Light Between Diffuse Surfaces SIGGRAPH '84

The Cornell Box

- Careful calibration and measurement allows for comparison between physical scene & simulation

photograph

simulation

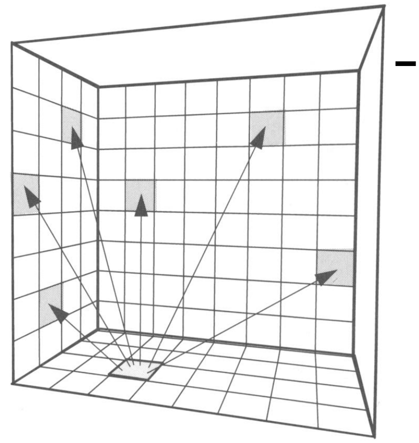
Light Measurement Laboratory
Cornell University, Program for Computer Graphics

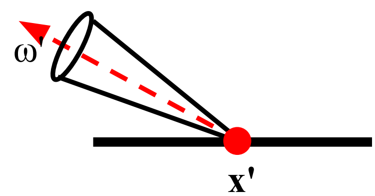
Today

- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors
  - Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Worksheet
- Papers for Next Time

Radiosity Overview

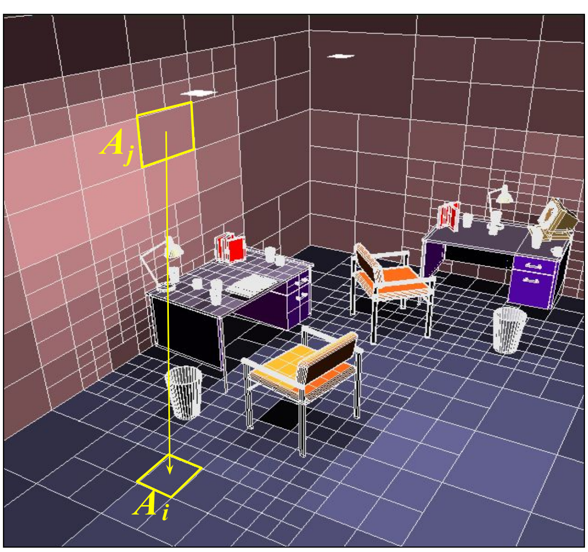
- Surfaces are assumed to be perfectly Lambertian (diffuse)
  - reflect incident light in all directions with equal intensity
- The scene is divided into a set of small areas, or patches.
- The radiosity, B_i, of patch *i* is the total rate of energy leaving a surface. The radiosity over a patch is constant.
- Units for radiosity: Watts / steradian * meter^2





Discrete Radiosity Equation

Discretize the scene into *n* patches, over which the radiosity is constant



light leaving patch *i*

material reflectivity

$$B_i = E_i + \rho_i \sum_{j=1}^{n} F_{ij} B_j$$

light emitted from patch *i*

form factor

The equation is recursive, but it can be solved iteratively

Radiosity in Matrix Form

$$B_i = E_i + \rho_i \sum_{j=1}^{n} F_{ij} B_j$$

n simultaneous equations with n unknown B_i values can be written in matrix form:

solve for B_i

$$\begin{bmatrix} 1-\rho_1 F_{11} & -\rho_1 F_{12} & \cdots & -\rho_1 F_{1n} \\ -\rho_2 F_{21} & 1-\rho_2 F_{22} & & \\ \vdots & & \ddots & \\ -\rho_n F_{n1} & \cdots & \cdots & 1-\rho_n F_{nn} \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{bmatrix}$$

A solution yields a single radiosity value B_i for each patch in the environment, a view-independent solution.

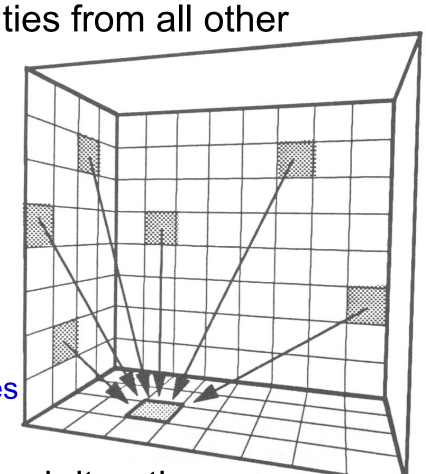
Solving the Radiosity Matrix

- Initialize all radiosity values to 0
- Each iteration, update the radiosity of each patch by *gathering* the contribution of radiosities from all other

$$\begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_i \\ \vdots \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_i \\ \vdots \\ E_n \end{bmatrix} + \begin{bmatrix} & & & \\ & & & \\ & & & \\ \rho_i F_{i1} & \rho_i F_{i2} & \cdots & \rho_i F_{in} \\ & & & \\ & & & \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_i \\ \vdots \\ B_n \end{bmatrix}$$

Radiosity values on iteration ***t+1***

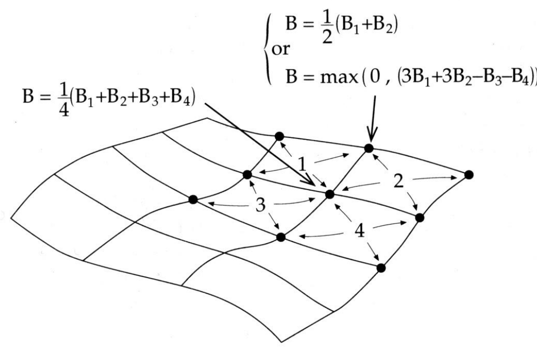
Radiosity values on iteration ***t***



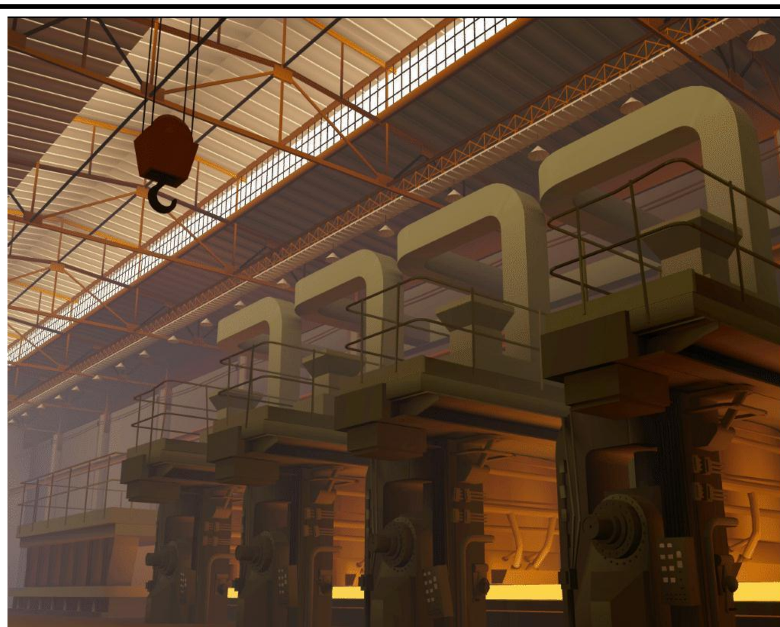
- Radiosity values only increase on each iteration
- This method is fundamentally a Gauss-Seidel relaxation

Interpolating Vertex Radiosities

- B_i radiosity values are constant over the extent of a patch.
- How are they mapped to the vertex radiosities (intensities) needed by the renderer?
  - Average the radiosities of patches that contribute to the vertex
  - Vertices on the edge of a surface are assigned values extrapolation



Questions?



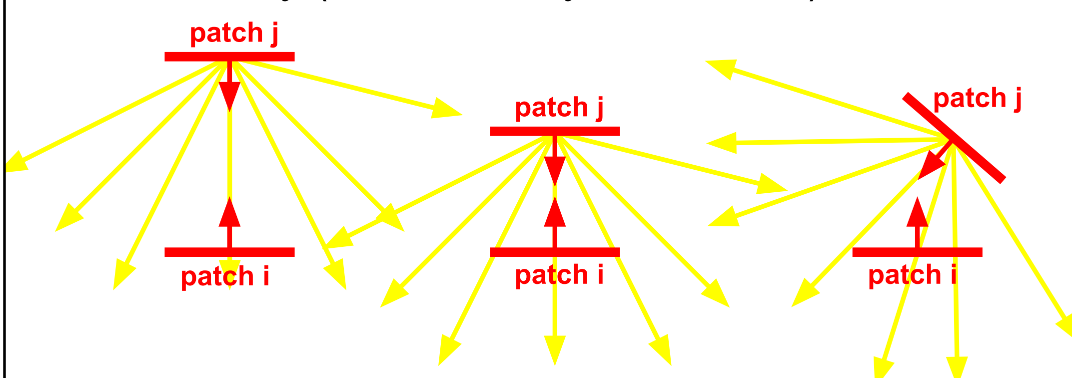
Factory simulation. 30,000 patches.
Program of Computer Graphics, Cornell University.

Today

- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors
  - Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Worksheet
- Papers for Next Time

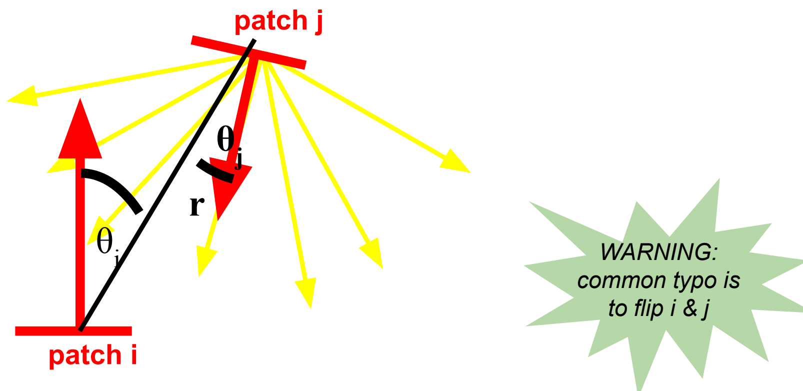
Calculating the Form Factor F_{ij}

- F_{ij} = fraction of light energy leaving patch j that arrives at patch i
- Takes account of both:
  - geometry (size, orientation & position)
  - visibility (are there any occluders?)



Calculating the Form Factor F_{ij}

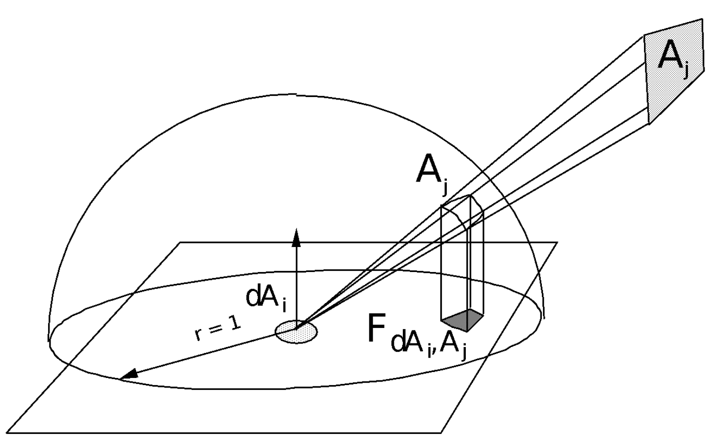
- F_{ij} = fraction of light energy leaving patch j that arrives at patch i



$$F_{ij} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{\cos \theta_i \cos \theta_j}{\pi r^2} V_{ij} \, dA_j \, dA_i$$

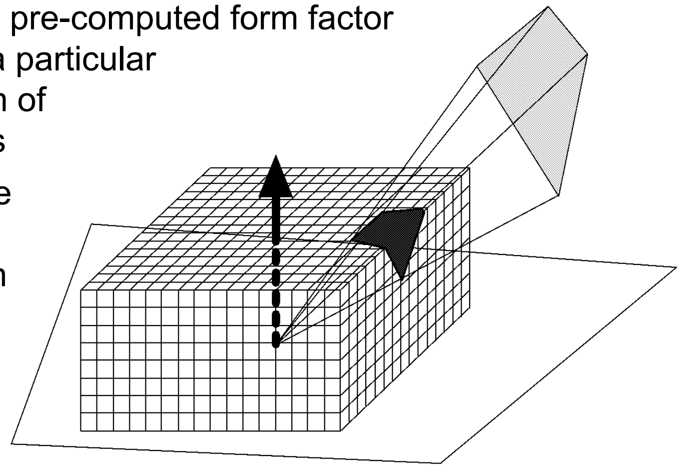
Form Factor Determination

The Nusselt analog: the form factor of a patch is equivalent to the fraction of the unit circle that is formed by taking the projection of the patch onto the hemisphere surface and projecting it down onto the circle.



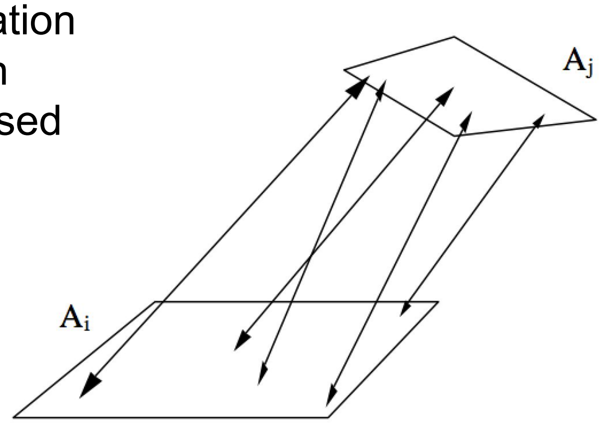
Hemicube Algorithm

- A hemicube is constructed around the center of each patch
- Faces of the hemicube are divided into "pixels"
- Each patch is projected (rasterized) onto the faces of the hemicube
- Each pixel stores its pre-computed form factor The form factor for a particular patch is just the sum of the pixels it overlaps
- Patch occlusions are handled similar to z-buffer rasterization



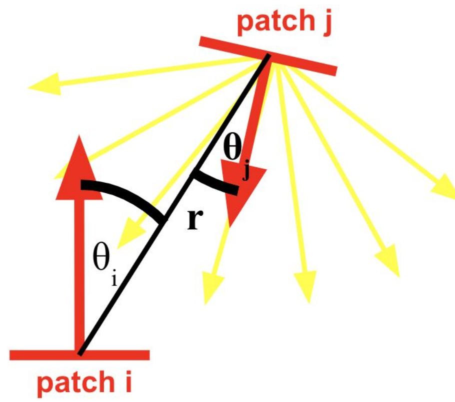
Form Factor from Ray Casting

- Cast *n* rays between the two patches
  - Compute visibility (what fraction of rays do not hit an occluder)
  - Integrate the point-to-point form factor
- Permits the computation of the patch-to-patch form factor, as opposed to point-to-patch



Calculating the Form Factor F_{ij}

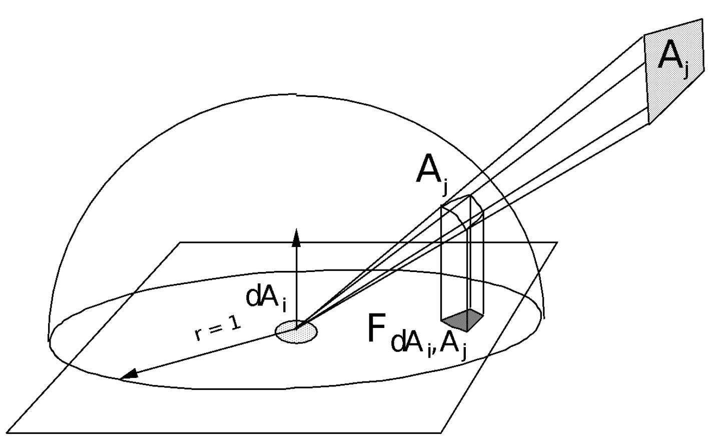
- F_{ij} = fraction of light energy leaving patch j that arrives at patch i



$$F_{ij} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{\cos\theta_i \cos\theta_j}{\pi r^2} V_{ij}\, dA_j\, dA_i$$

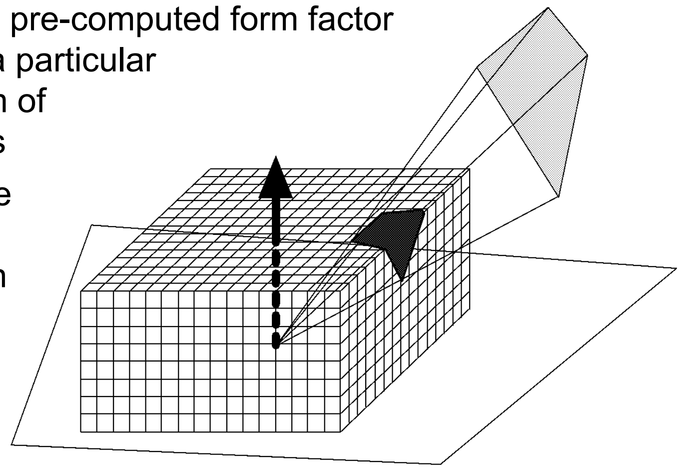
Form Factor Determination

The Nusselt analog: the form factor of a patch is equivalent to the fraction of the unit circle that is formed by taking the projection of the patch onto the hemisphere surface and projecting it down onto the circle.



Hemicube Algorithm

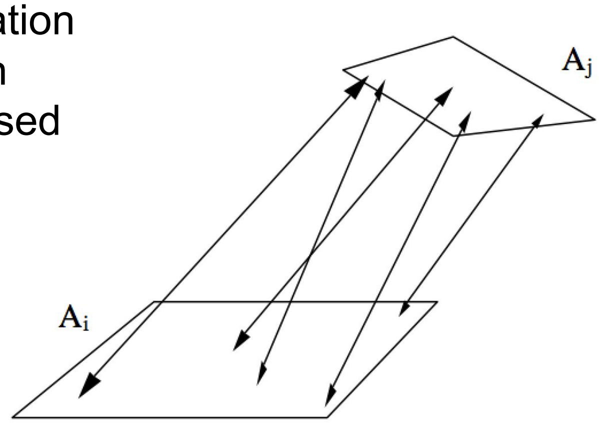
- A hemicube is constructed around the center of each patch
- Faces of the hemicube are divided into "pixels"
- Each patch is projected (rasterized) onto the faces of the hemicube
- Each pixel stores its pre-computed form factor The form factor for a particular patch is just the sum of the pixels it overlaps
- Patch occlusions are handled similar to z-buffer rasterization



Form Factor from Ray Casting

- Cast *n* rays between the two patches
  - Compute visibility (what fraction of rays do not hit an occluder)
  - Integrate the point-to-point form factor
- Permits the computation of the patch-to-patch form factor, as opposed to point-to-patch

Use this for HW3!



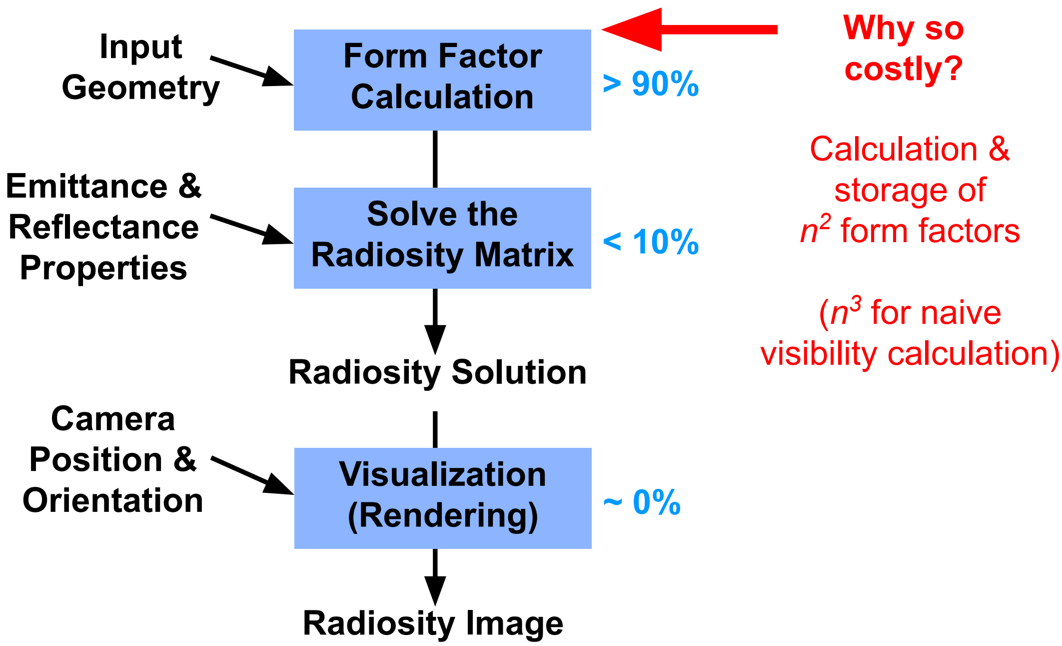
Questions?

Lightscape **http://www.lightscape.com**

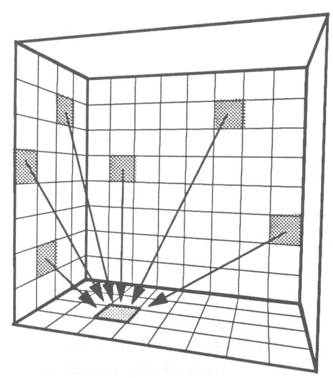
Today

- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors
  - Advanced Radiosity
    - Progressive Radiosity
    - Adaptive Subdivision
    - Discontinuity Meshing
    - Hierarchical Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Worksheet
- Papers for Next Time

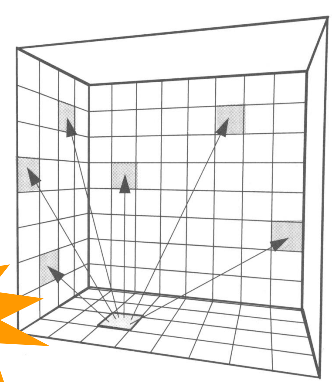
Stages in a Radiosity Solution



Progressive Refinement



- Goal: Provide frequent and timely updates to the user during computation
- Key Idea: Update the entire image at every iteration, rather than a single patch
- How? Instead of summing the light received by one patch, distribute the radiance of the patch with the most *undistributed radiance*.

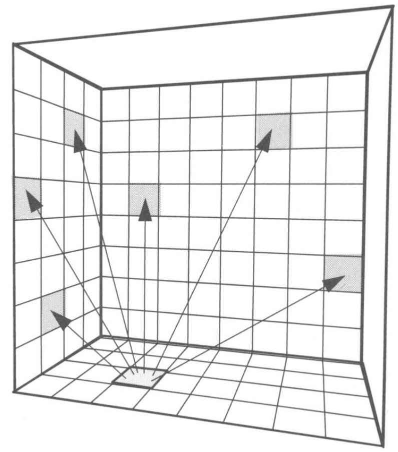


Use this for HW3!

Reordering the Solution for PR

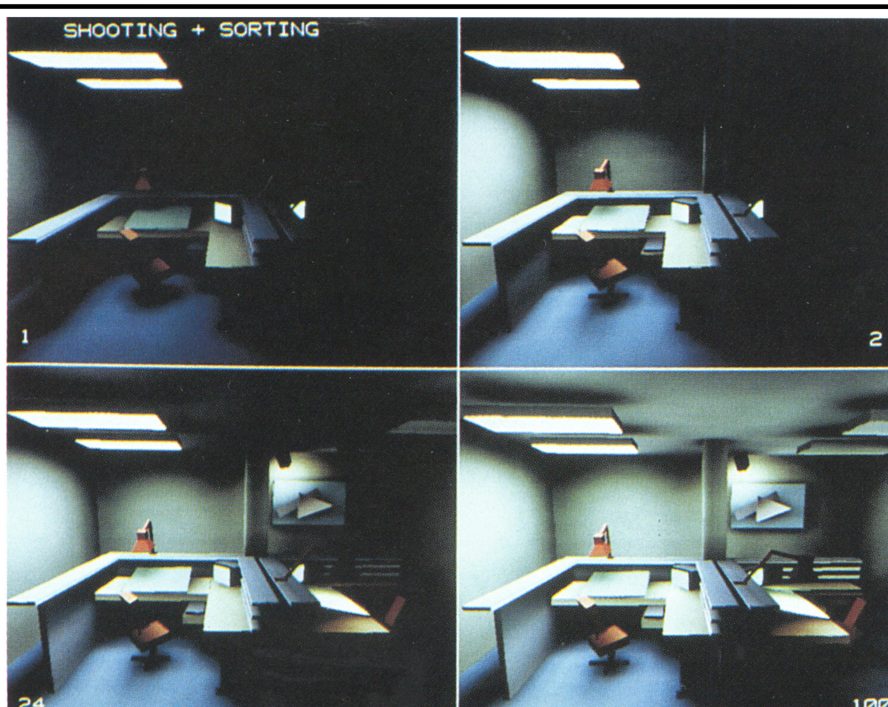
Shooting: the radiosity of all patches is updated for each iteration:

$$\begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ \vdots \\ B_n \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ \vdots \\ B_n \end{bmatrix} + \begin{bmatrix} \cdots & \rho_1 F_{1i} & \cdots \\ \cdots & \rho_2 F_{2i} & \\ & & \\ \cdots & \rho_n F_{ni} & \cdots \end{bmatrix} \begin{bmatrix} \\ \vdots \\ B_i \\ \vdots \\ \end{bmatrix}$$

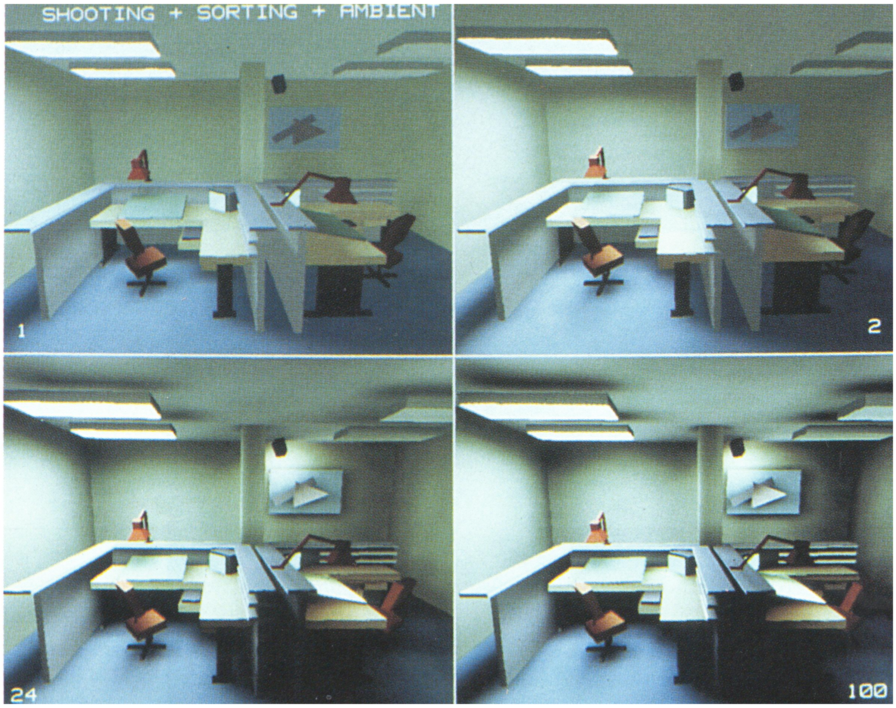


This method is fundamentally a Southwell relaxation

Progressive Refinement w/out Ambient Term



Progressive Refinement with Ambient Term



Questions?

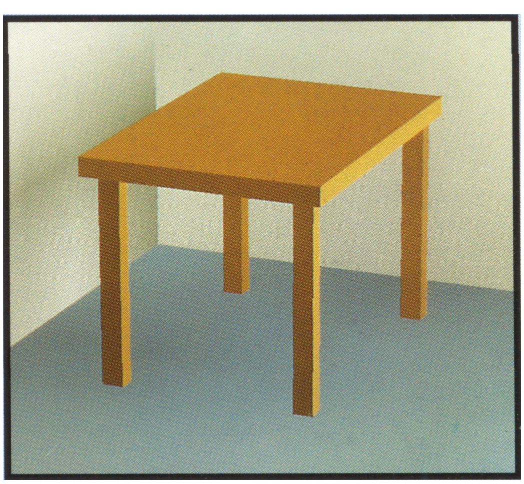
Lightscape **http://www.lightscape.com**

Today

- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors
  - Advanced Radiosity
    - Progressive Radiosity
    - Adaptive Subdivision
    - Discontinuity Meshing
    - Hierarchical Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Worksheet
- Papers for Next Time

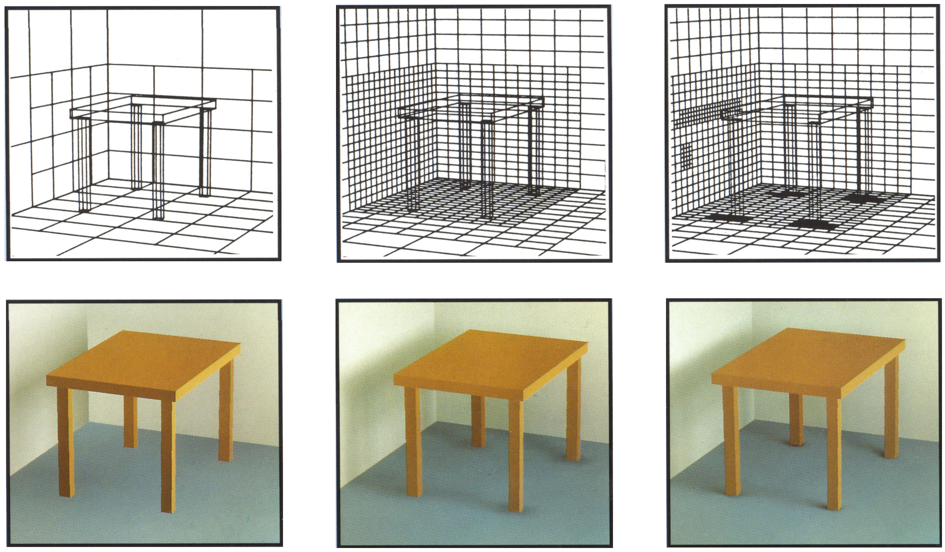
Increasing the Accuracy of the Solution

What's wrong with this picture?



- Image quality is a function of patch size
- Compute a solution on a uniform initial mesh, then refine the mesh in areas that exceed some error tolerance:
  - shadow boundaries
  - other areas with a high radiosity gradient

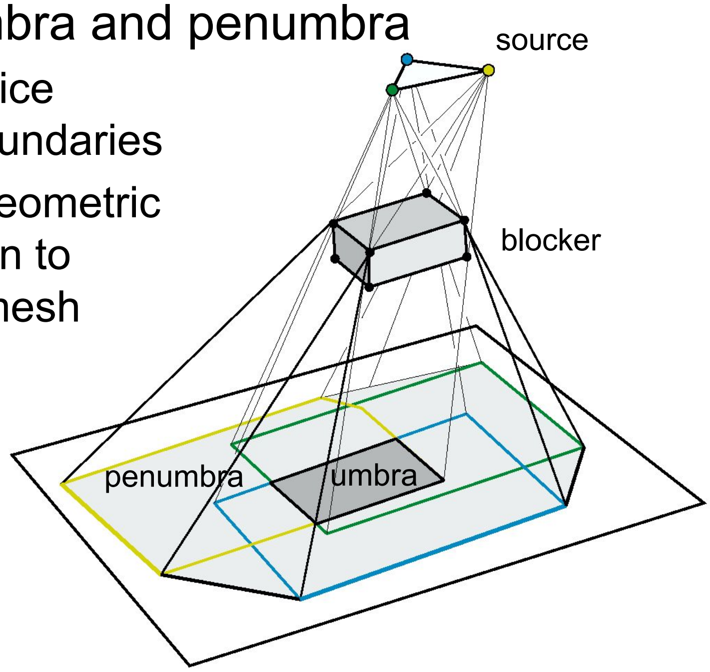
Adaptive Subdivision of Patches



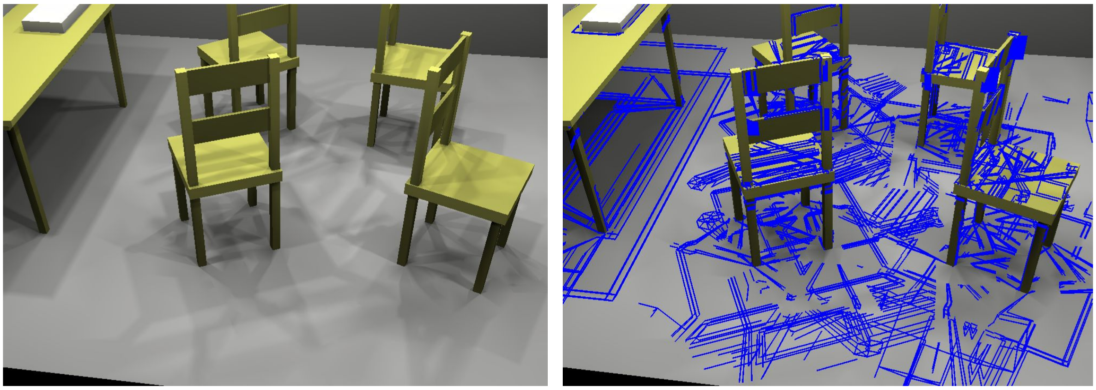
Coarse patch solution (145 patches) | Improved solution (1021 subpatches) | Adaptive subdivision (1306 subpatches)

Discontinuity Meshing

- Limits of umbra and penumbra
  - Captures nice shadow boundaries
  - Complex geometric computation to construct mesh

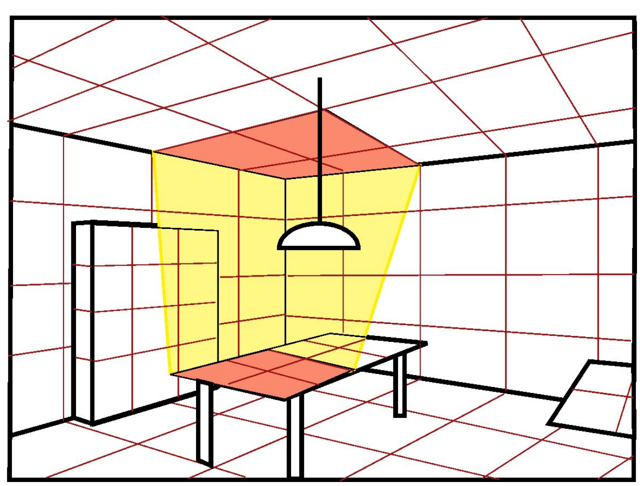


“Fast and Accurate Hierarchical Radiosity Using Global Visibility”
Durand, Drettakis, & Puech 1999



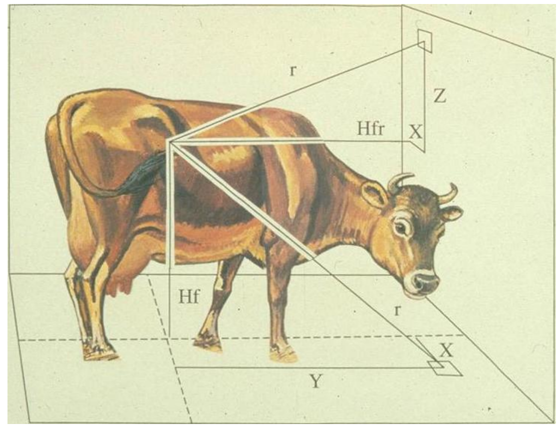
Hierarchical Radiosity

- Group elements when the light exchange is not important
  - Breaks the quadratic complexity
  - Control non trivial, memory cost



Practical Problems with Radiosity

- Meshing
  - memory
  - robustness
- Form factors
  - computation
- Diffuse limitation
  - extension to specular takes too much memory



Cow-cow form factor?

Questions?

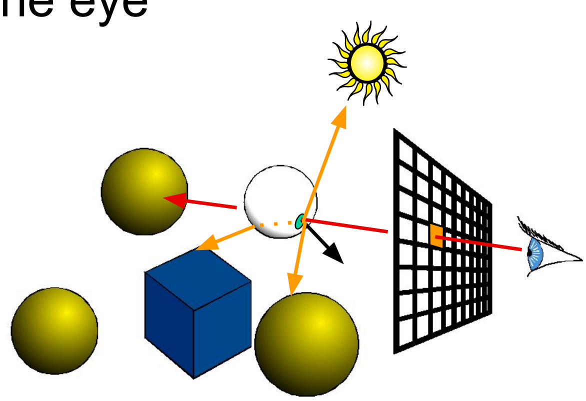
Lightscape **http://www.lightscape.com**

Today

- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors
  - Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Worksheet
- Papers for Next Time

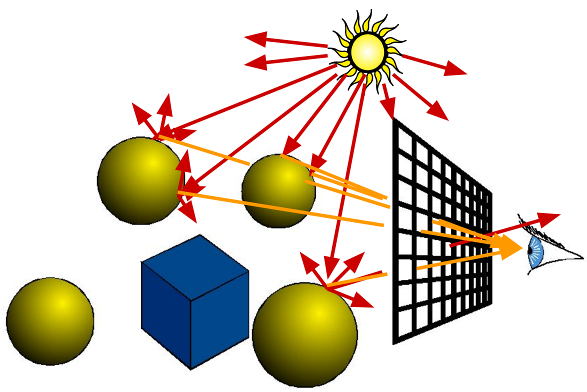
Does Ray Tracing Simulate Physics?

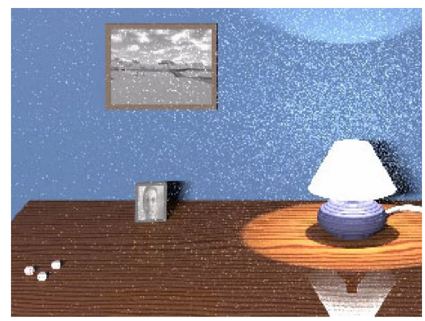
- No…. traditional ray tracing is also called "*backward" ray tracing*
- In reality, photons actually travel from the light to the eye



Forward Ray Tracing

- Start from the light source
  - But very, very low probability to reach the eye
- What can we do about it?
  - Always send a ray to the eye…. still not efficient

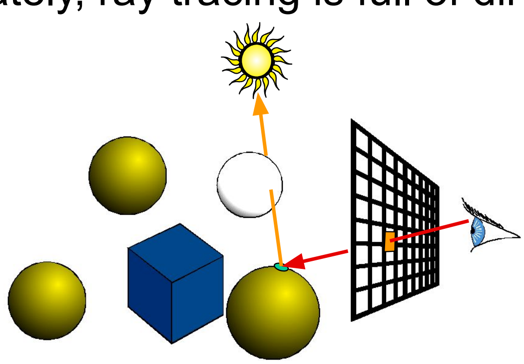




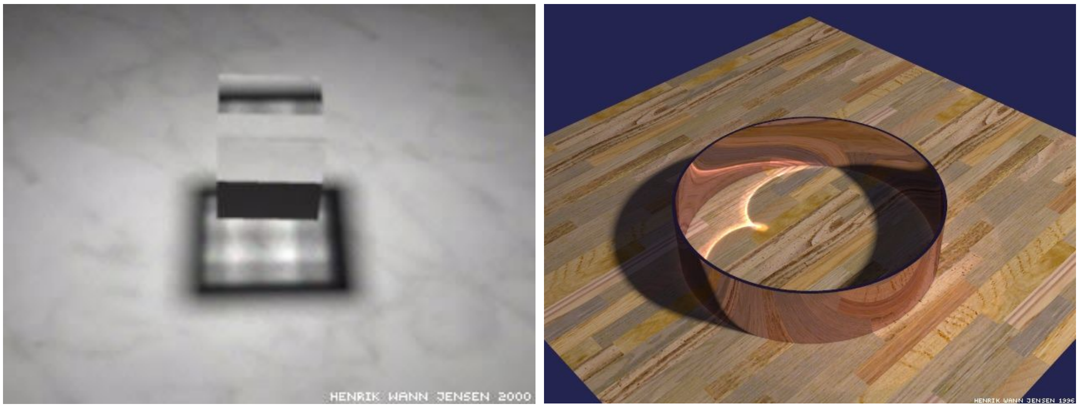
Henrik Wann Jensen

Transparent Shadows?

- What to do if the shadow ray sent to the light source intersects a transparent object?
  - Pretend it's opaque?
  - Multiply by transparency color?
    (ignores refraction & does not produce caustics)
- Unfortunately, ray tracing is full of dirty tricks



Is this Traditional Ray Tracing?

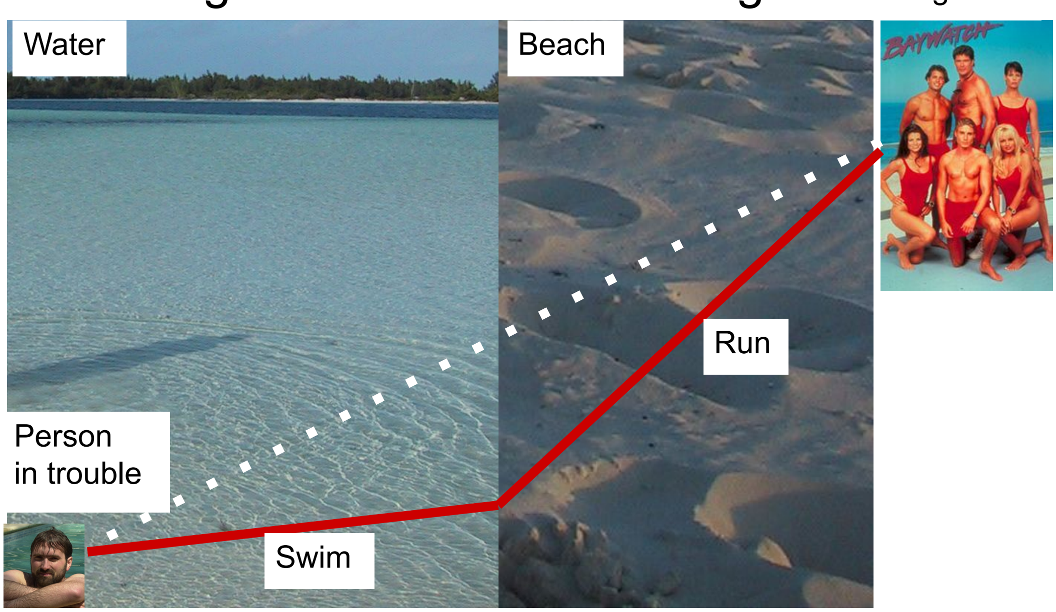


Images by Henrik Wann Jensen

No. Refraction and complex reflections for illumination are not handled properly in traditional (backward) ray tracing.

Refraction and the Lifeguard Problem

- Running is faster than swimming

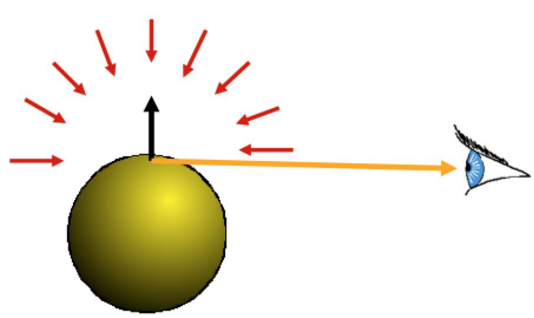


Today

- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors
  - Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Worksheet
- Papers for Next Time

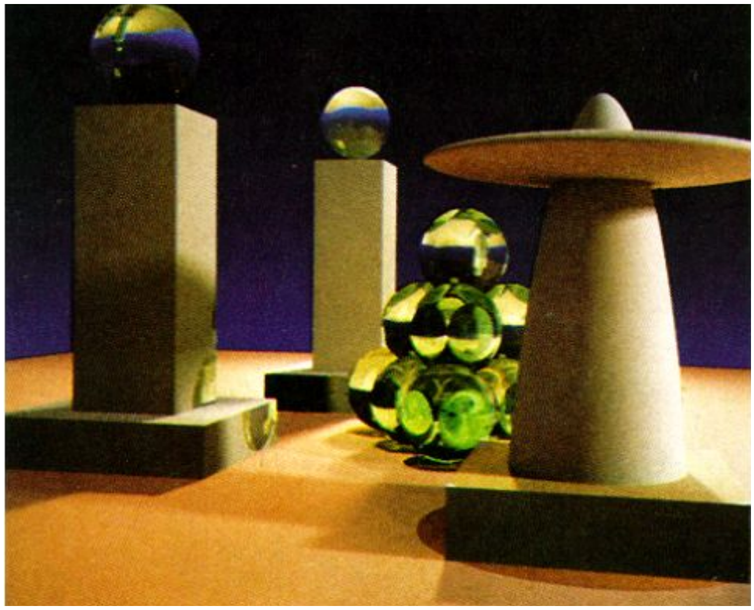
The Rendering Equation

- Clean mathematical framework for light-transport simulation
- At each point, outgoing light in one direction is the integral of incoming light in all directions multiplied by reflectance property

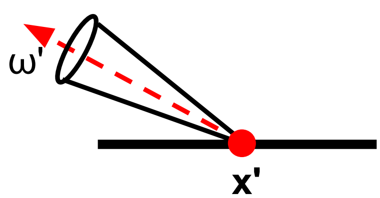


Reading for Next Time: *after break!*

- "The Rendering Equation", Kajiya, SIGGRAPH 1986



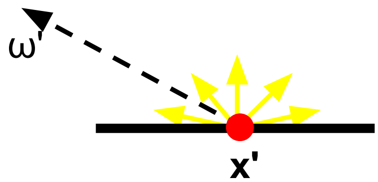
The Rendering Equation



$L(x',\omega') = E(x',\omega') + \int \rho_{x'}(\omega,\omega')L(x,\omega)G(x,x')V(x,x')\, dA$

L (x',ω') is the radiance from a point on a surface in a given direction ω'

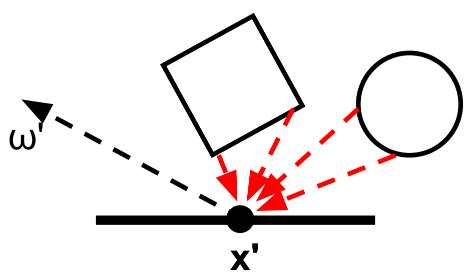
The Rendering Equation



$$L(x',\omega') = E(x',\omega') + \int \rho_{x'}(\omega,\omega')L(x,\omega)G(x,x')V(x,x')\, dA$$

E(x',ω') is the emitted radiance from a point: E is non-zero only if x' is emissive (a light *source*)

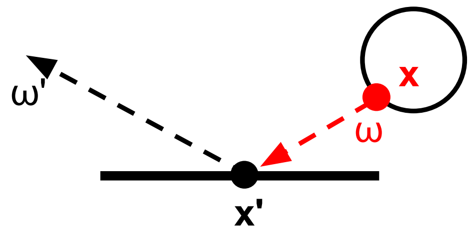
The Rendering Equation



$$L(x',\omega') = E(x',\omega') + \underbrace{\int \rho_{x'}(\omega,\omega')L(x,\omega)G(x,x')V(x,x')\, dA}$$

Sum the contribution from all of the other surfaces in the scene

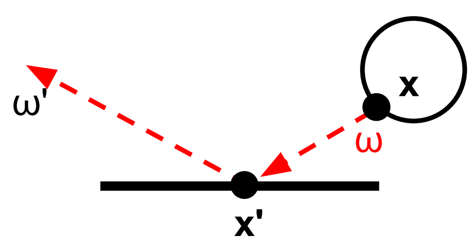
The Rendering Equation



$$L(x',\omega') = E(x',\omega') + \int \rho_{x'}(\omega,\omega') L(x,\omega) G(x,x') V(x,x')\, dA$$

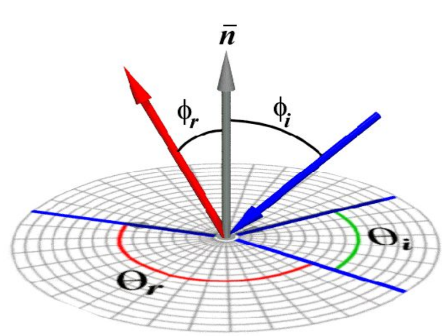
For each x, compute L(x, ω), the radiance at point x in the direction ω (from x to x')

The Rendering Equation

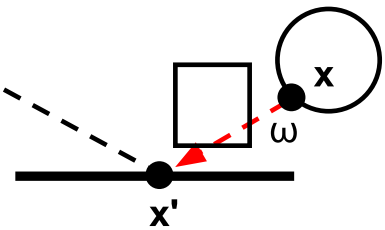


$$L(x',\omega') = E(x',\omega') + \int \rho_{x'}(\omega,\omega') L(x,\omega) G(x,x') V(x,x')\, dA$$

scale the contribution by $\rho_{x'}(\omega,\omega')$, the reflectivity (BRDF) of the surface at x'



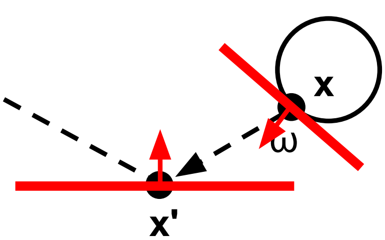
The Rendering Equation



$L(x',\omega') = E(x',\omega') + \int \rho_{x'}(\omega,\omega')L(x,\omega)G(x,x')V(x,x')\,dA$

For each x, compute V(x,x'),
the visibility between x and x':
1 when the surfaces are unobstructed
along the direction ω, 0 otherwise

The Rendering Equation

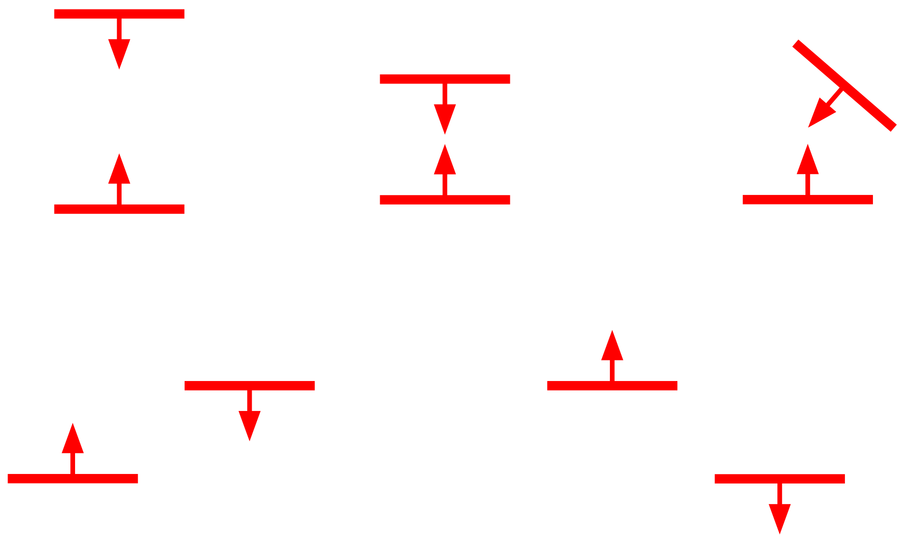


$L(x',\omega') = E(x',\omega') + \int \rho_{x'}(\omega,\omega')L(x,\omega)G(x,x')V(x,x')\,dA$

For each x, compute G(x, x'), which
describes the on the geometric relationship
between the two surfaces at x and x’

Intuition about G(x,x')?

- Which arrangement of two surfaces will yield the greatest transfer of light energy? Why?



Rendering Equation → Radiosity

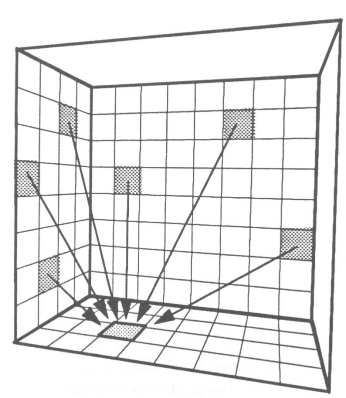
$$L(x',\omega') = E(x',\omega') + \int \rho_{x'}(\omega,\omega') L(x,\omega) G(x,x') V(x,x')\, dA$$

Radiosity assumption:
perfectly diffuse surfaces (not directional)

$$B_{x'} = E_{x'} + \rho_{x'} \int B_x \, G(x,x') V(x,x')$$

discretize

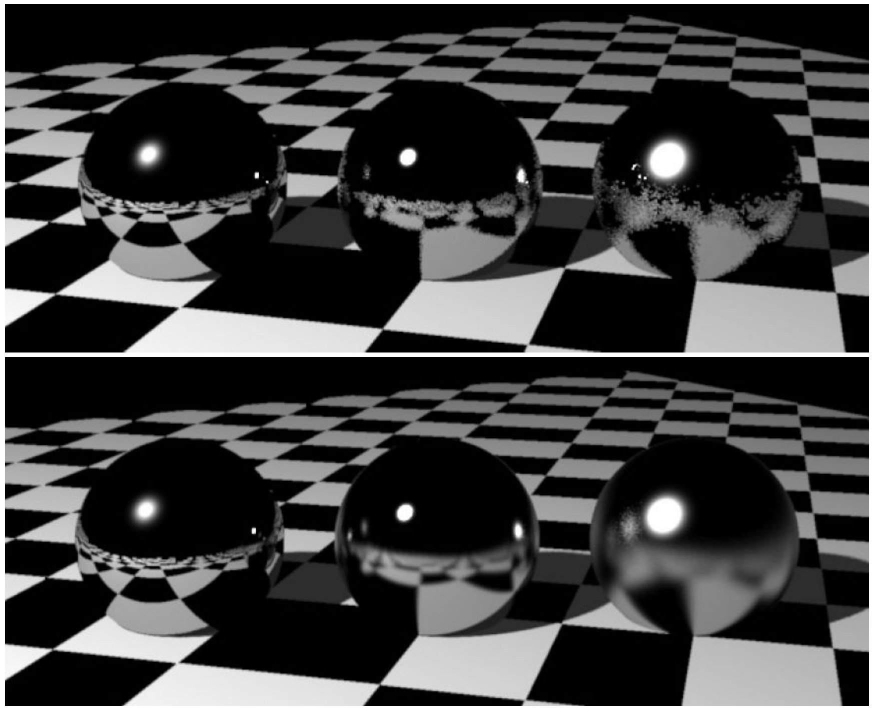
$$B_i = E_i + \rho_i \sum_{j=1}^{n} F_{ij} B_j$$



Questions?

1 glossy sample per pixel

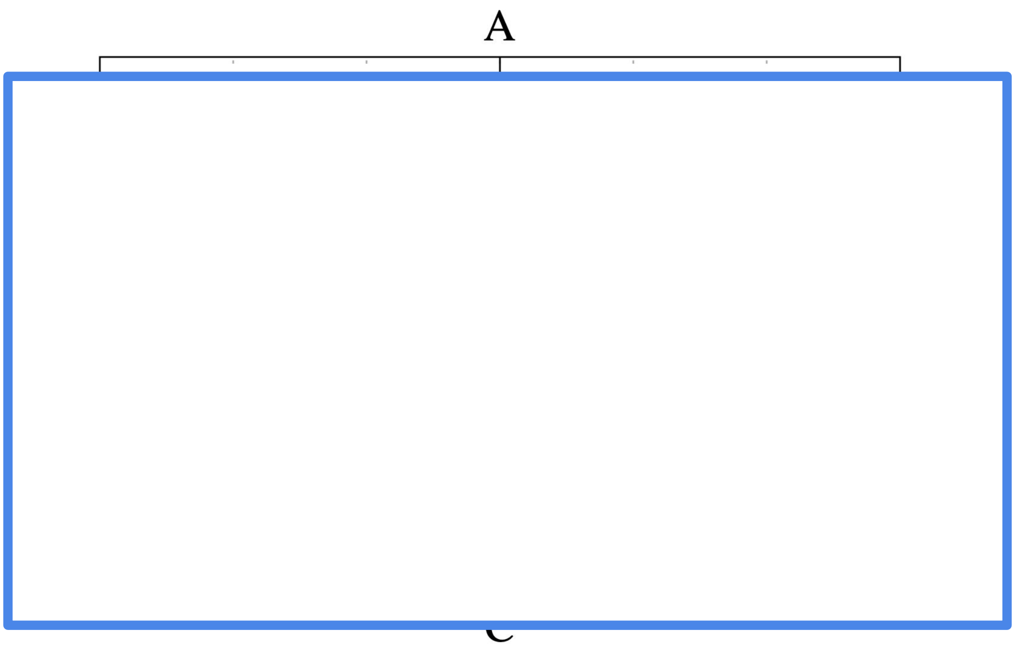
256 glossy samples per pixel



Today

- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors
  - Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Worksheet
- Papers for Next Time

Pop Worksheet!

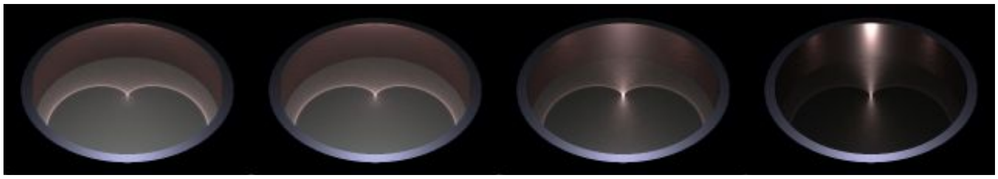


Today

- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors
  - Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Worksheet
- Papers for Next Time

Readings for Next Time:

- "Rendering Caustics on Non-Lambertian Surfaces", Henrik Wann Jensen, *Graphics Interface* 1996.



- "Global Illumination using Photon Maps", Henrik Wann Jensen, *Rendering Techniques* 1996.