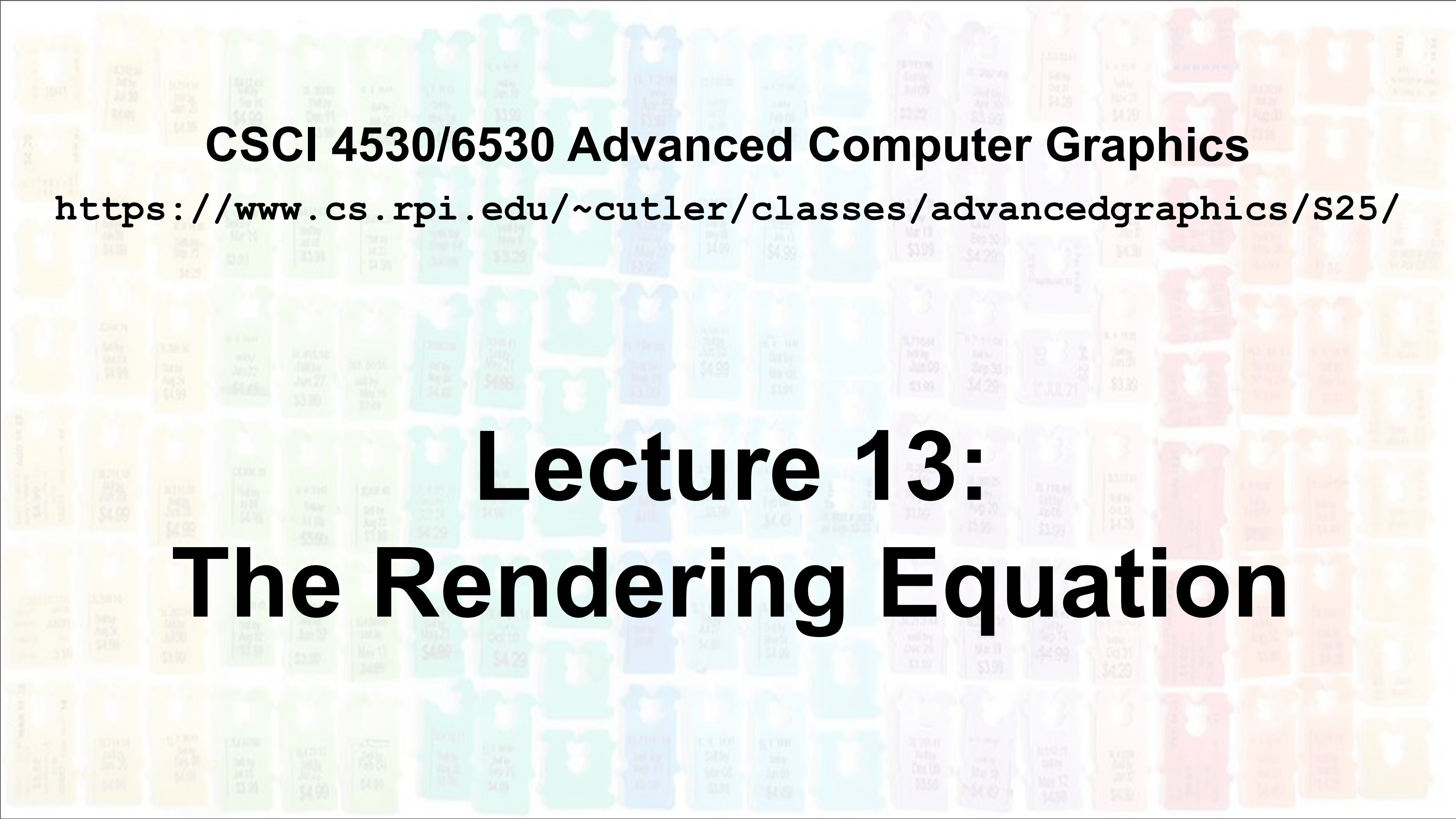


CSCI 4530/6530 Advanced Computer Graphics

https://www.cs.rpi.edu/~cutler/classes/advancedgraphics/S25/

Lecture 13: The Rendering Equation

Rendering with Natural Light

Paul Debevec et. al, SIGGRAPH 1998

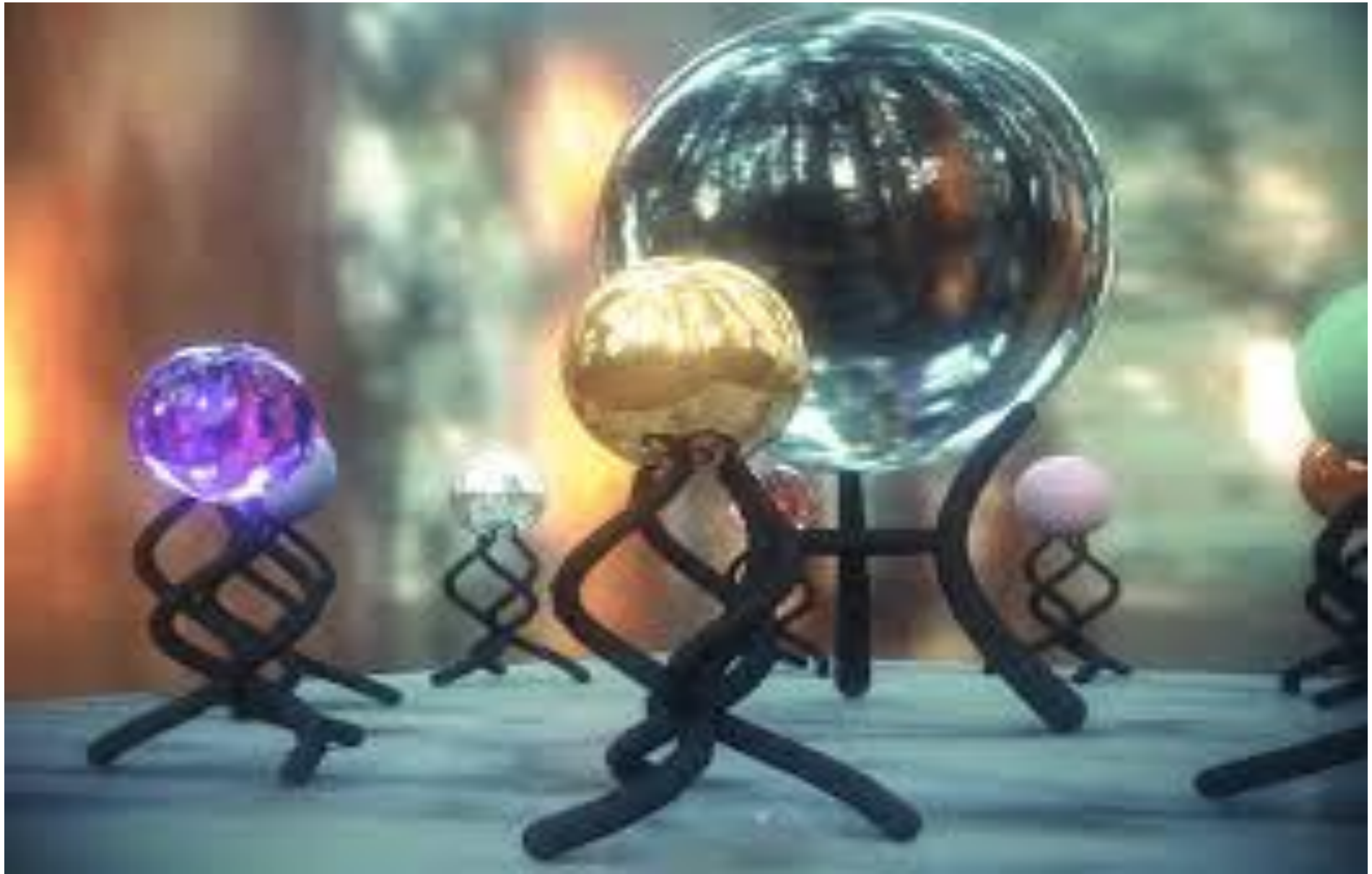


Image Based Lighting

Paul Debevec et al, SIGGRAPH 2000

Today

- HW3 & Final Project Brainstorming
- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors & Solving the Radiosity Matrix
  - Performance & Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Papers for Next Time
- Worksheet

Homework 3: Raytracing & Epsilon

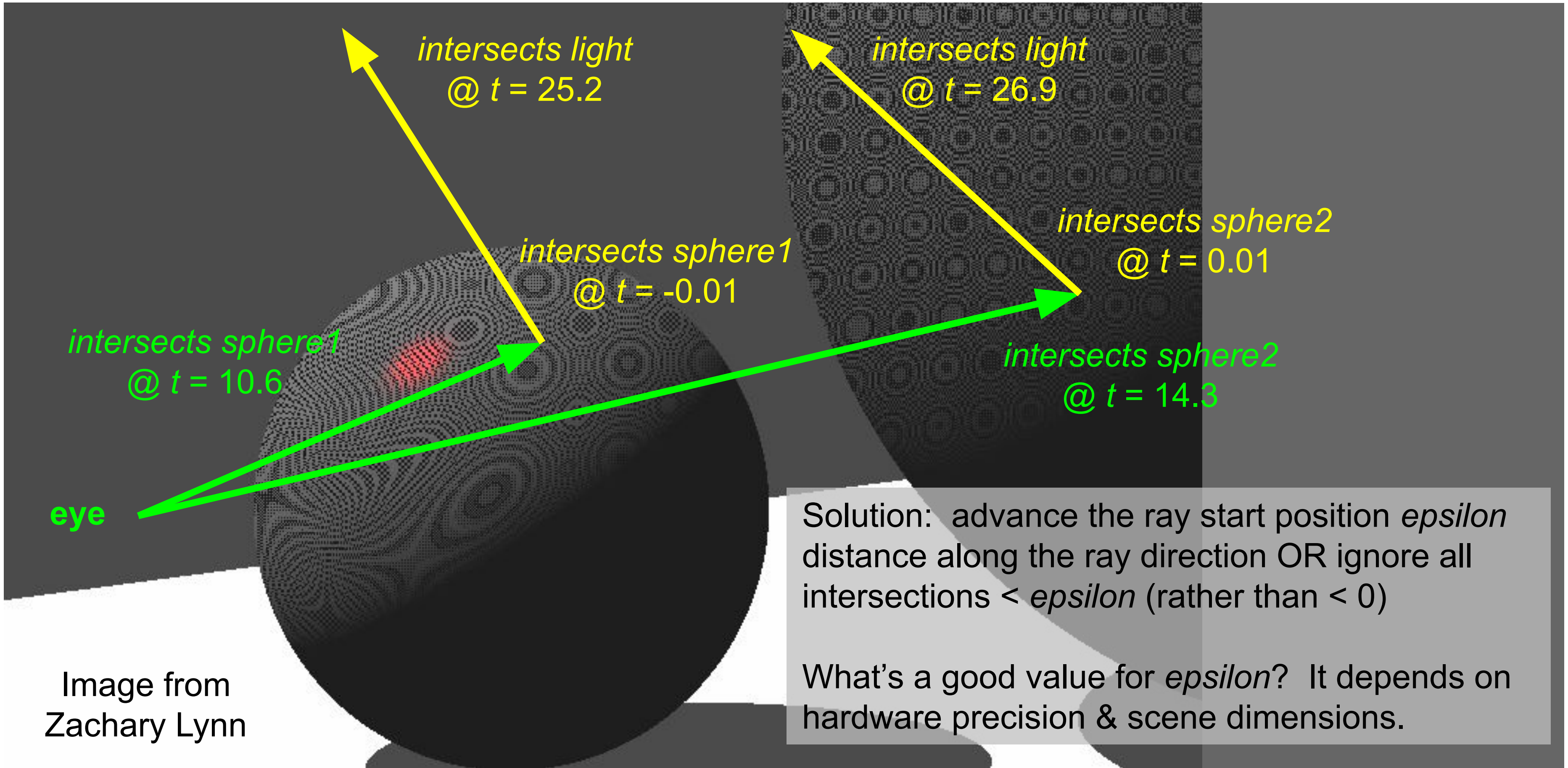


Image from Zachary Lynn

Solution: advance the ray start position *epsilon* distance along the ray direction OR ignore all intersections < *epsilon* (rather than < 0)

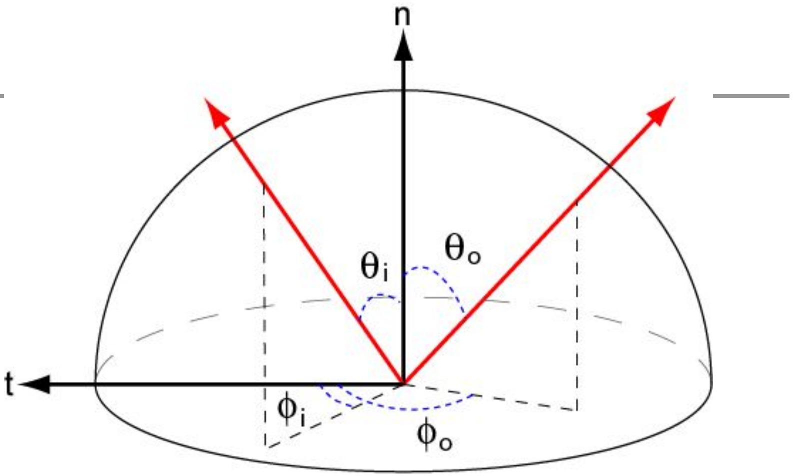
What's a good value for *epsilon*? It depends on hardware precision & scene dimensions.

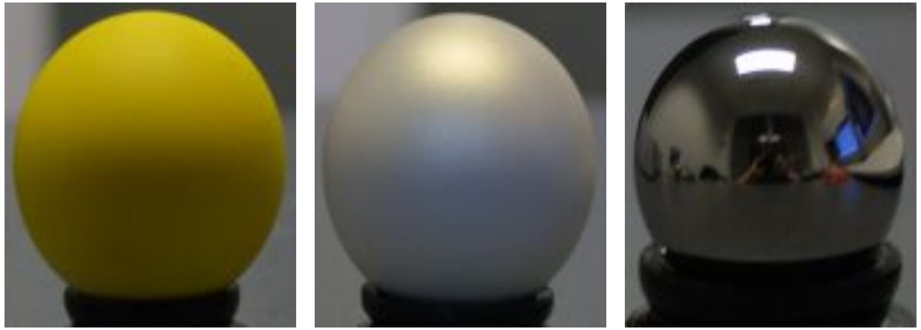
Final Project Brainstorming

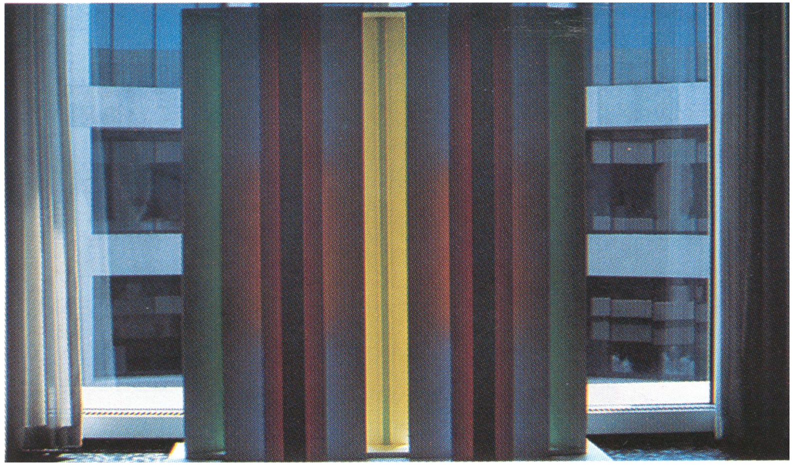
- Each student should post two different project ideas on the forum. For each idea:
  - Describe the idea, motivation, & an example potential result.
  - A significant/interesting technical implementation challenge.
- Have you already decided on one idea? Which one?
- Do you already have a partner? Who? (even if you have chosen an idea and/or a partner everyone must post 2 different ideas)
- Due Friday 2/28 @ 11:59pm
- Teams of 2 strongly recommended (individuals & teams >2 require instructor permission)
- Projects from prior terms are on the website

Last Time?

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



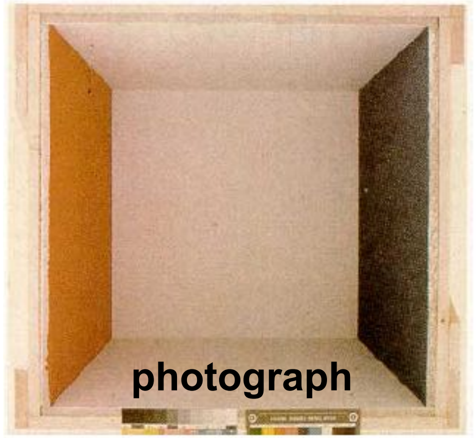


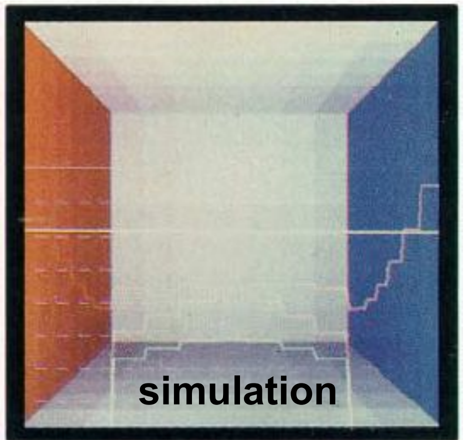


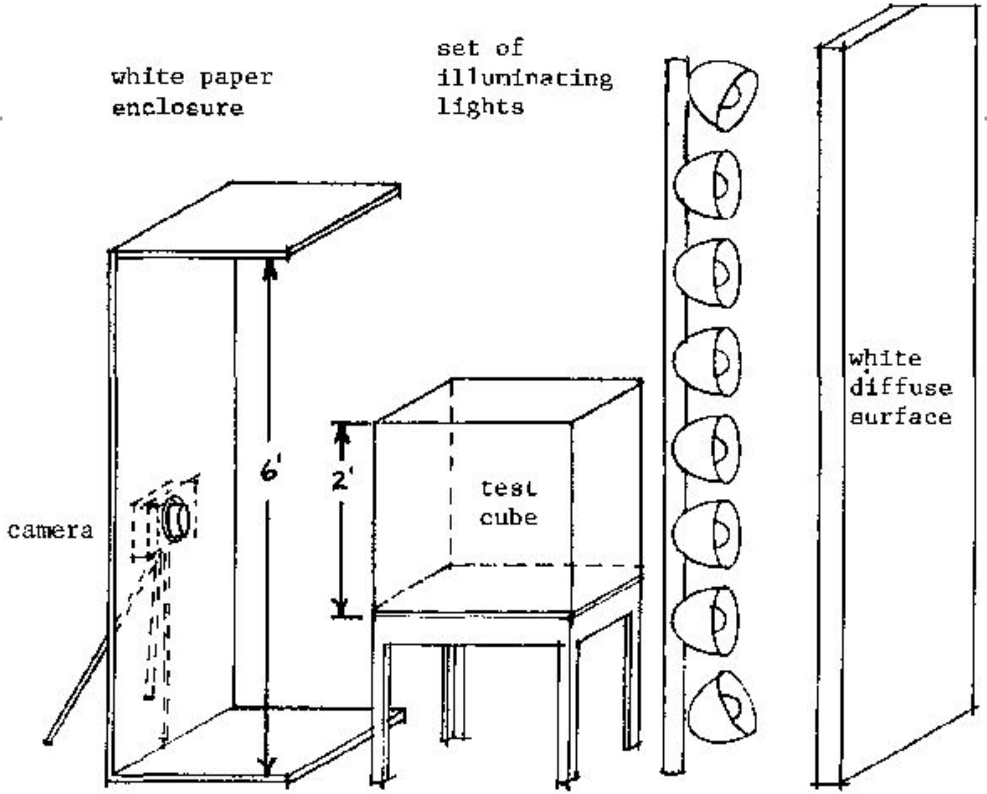
Today

- HW3 & Final Project Brainstorming
- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors & Solving the Radiosity Matrix
  - Performance & Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Papers for Next Time
- Worksheet

Reading for Today







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

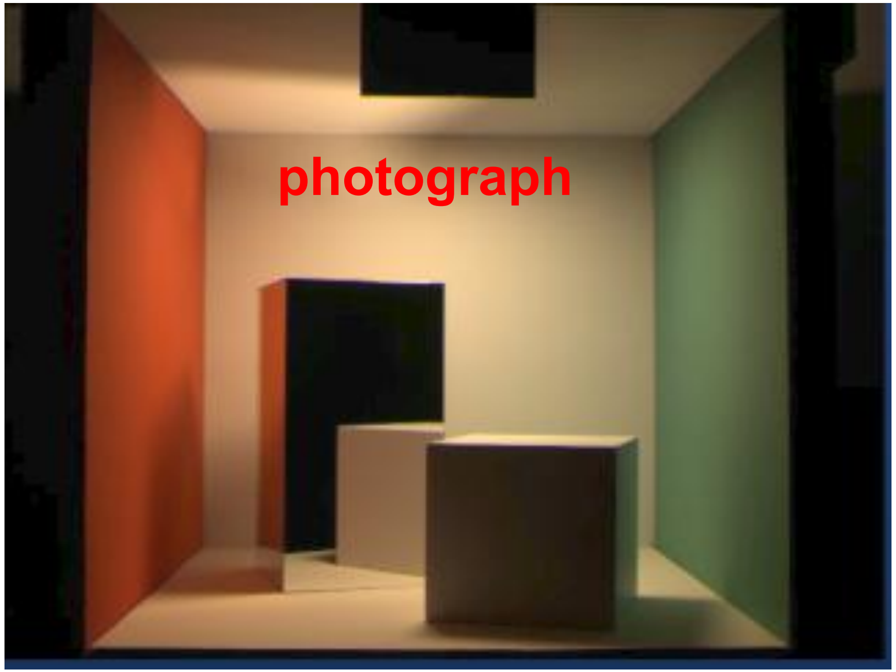
- Captures missing effects from ray tracing: Color bleeding
- Different choices/shortcuts than ray tracing
- Patches that receive and emit light
- Results are independent of viewer/camera position
- Comparison to real scene / photograph is good, “Cornell Box”!
- Add in specular highlights as secondary algorithm/post-process
- Precomputation to “bake” lighting (e.g., for interactive games)
  - In what scenes/applications is this useful?
- When do you need to recompute form factors?
  - Can some work be re-used after changes? In what scenarios?
- How do we handle non-diffuse light sources (e.g., open window)
- O(n^3) algorithm, does not include details on runtime, hardware used
- Maybe old papers (old font) should be reformatted for modern readers

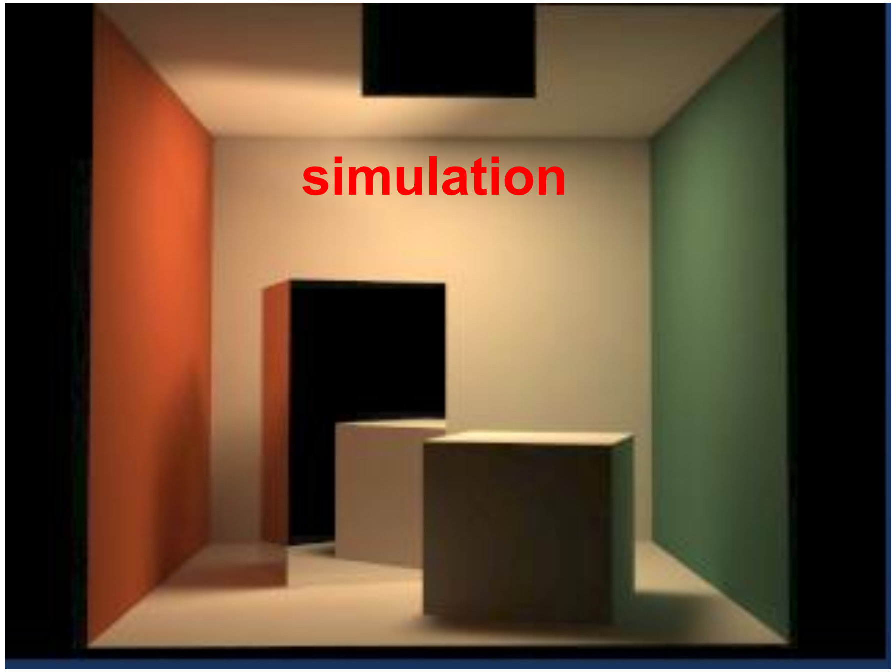
Today

- HW3 & Final Project Brainstorming
- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors & Solving the Radiosity Matrix
  - Performance & Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Papers for Next Time
- Worksheet

The Cornell Box

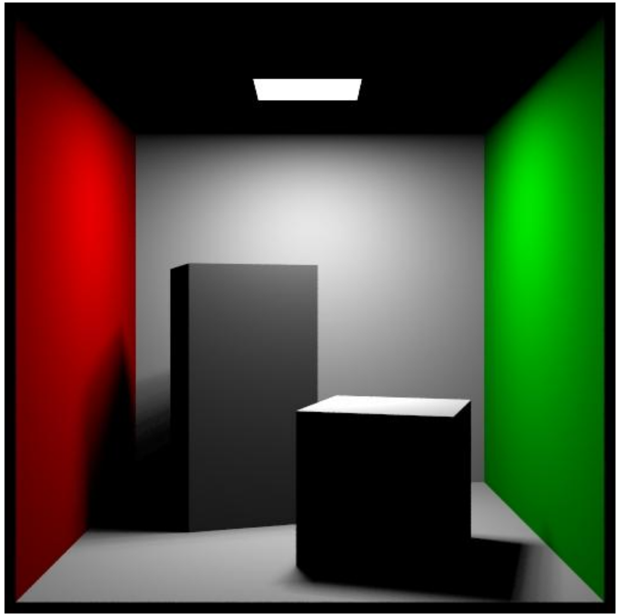
- Careful calibration and measurement allows for comparison between physical scene & simulation

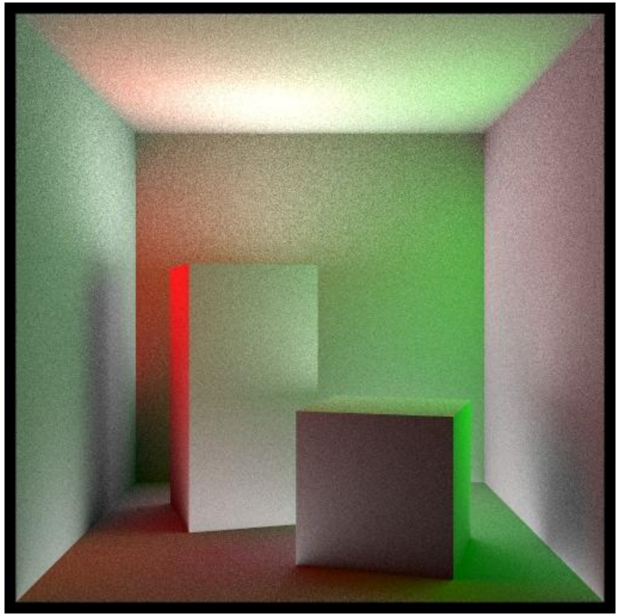


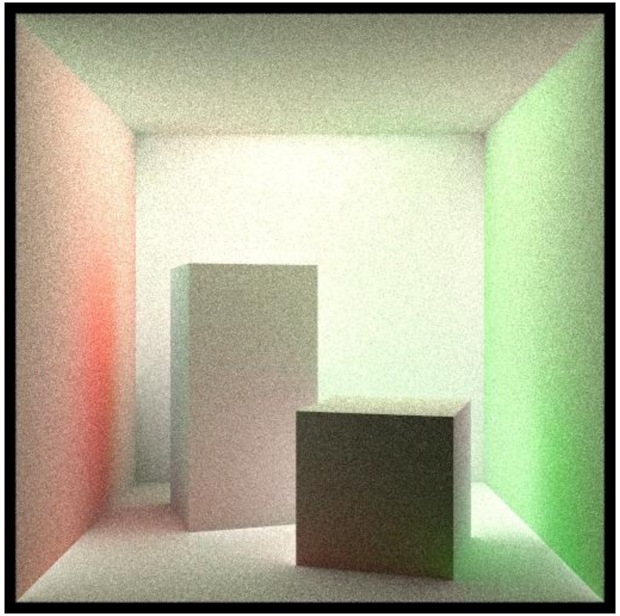


Light Measurement Laboratory Cornell University, Program for Computer Graphics

Visualizing Indirect Light / Inter-Reflections


direct illumination (0 bounces)


1 bounce

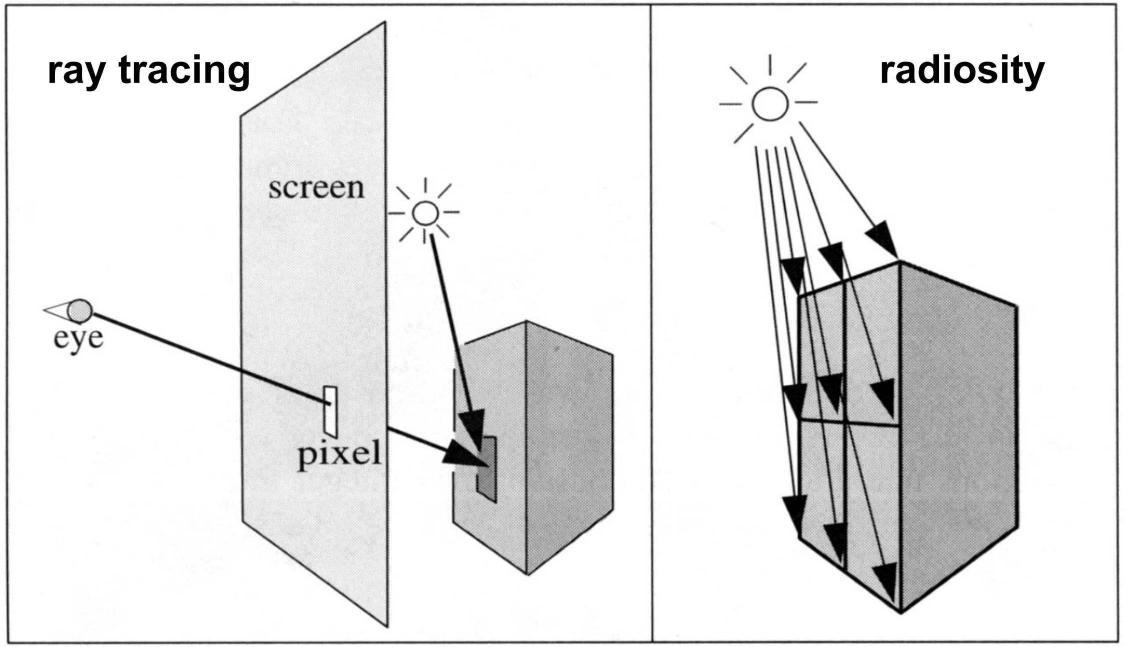

2 bounces

Note: image brightness not constant between images

images by Micheal Callahan
http://www.cs.utah.edu/~shirley/classes/cs684_98/students/callahan/bounce/

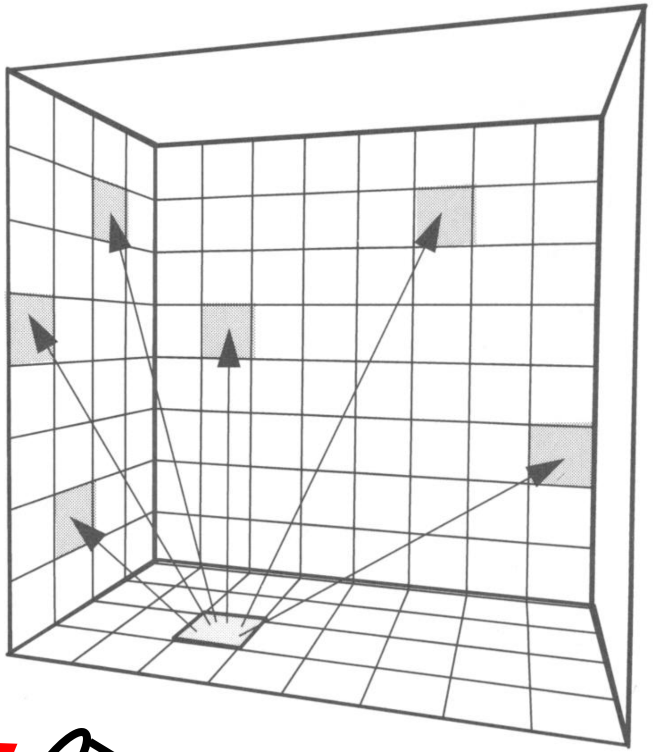
Radiosity vs. Ray Tracing

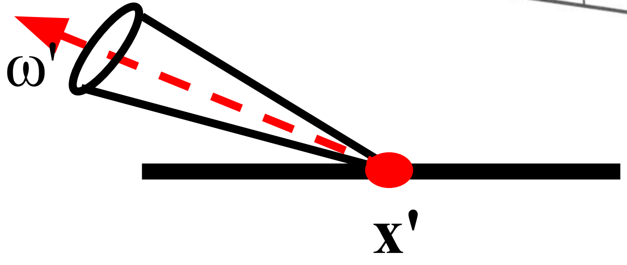
- Ray tracing is an *image-space* algorithm
  - If the camera is moved, we have to start over
- Radiosity is computed in *object-space*
  - View-independent (as long as we don't move the light)
  - Can pre-compute complex lighting to allow interactive walkthroughs



Radiosity Overview

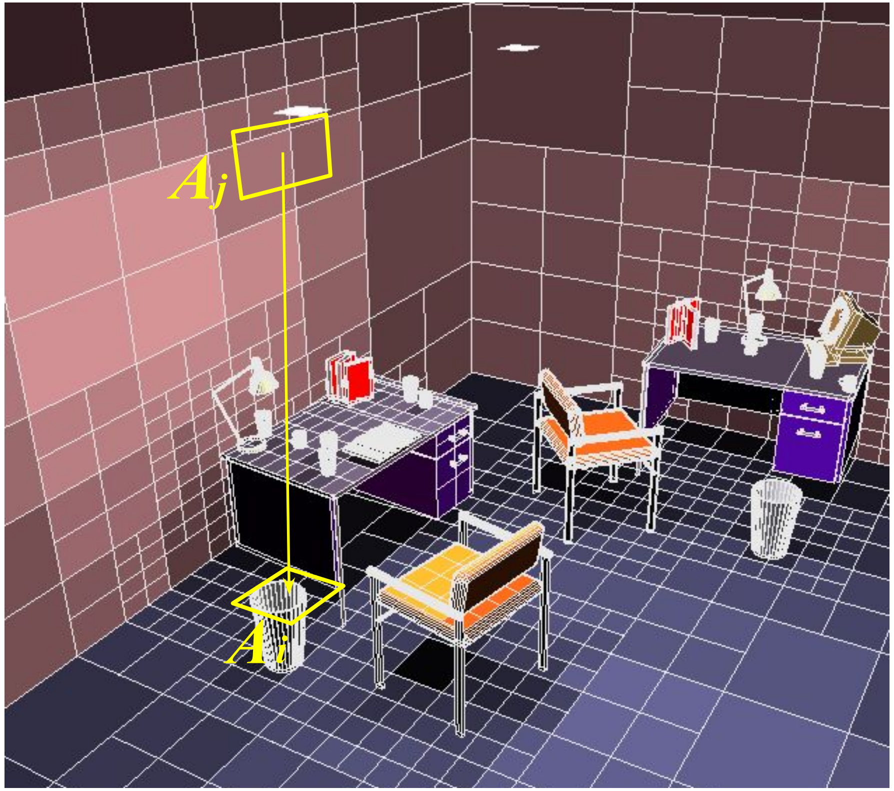
- Surfaces are assumed to be perfectly Lambertian (diffuse)
    - reflect incident light in all directions with equal intensity
- The scene is divided into a set of small areas, or patches
- The radiosity, $\mathbf{B}_i$, of patch *i* is the total rate of energy leaving a surface. The radiosity over 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 Equation in Matrix Form

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

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

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

Questions?

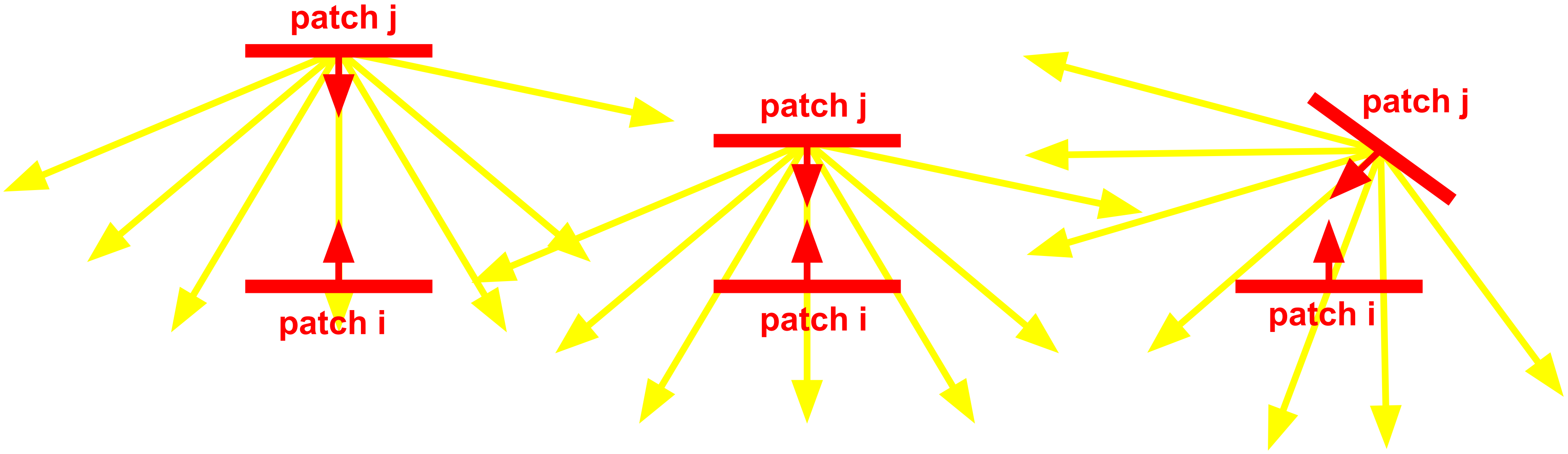
Lightscape *http://www.lightscape.com*

Today

- HW3 & Final Project Brainstorming
- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors & Solving the Radiosity Matrix
  - Performance & Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Papers for Next Time
- Worksheet

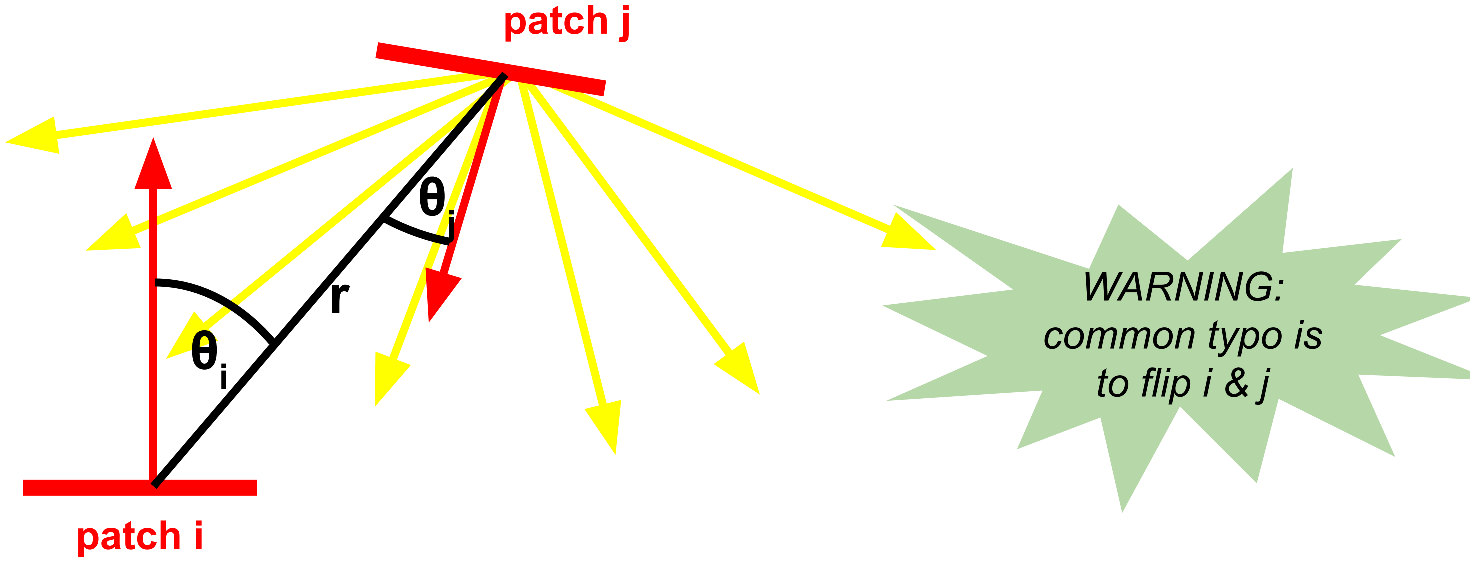
What is the Form Factor F_{ij} ?

- 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

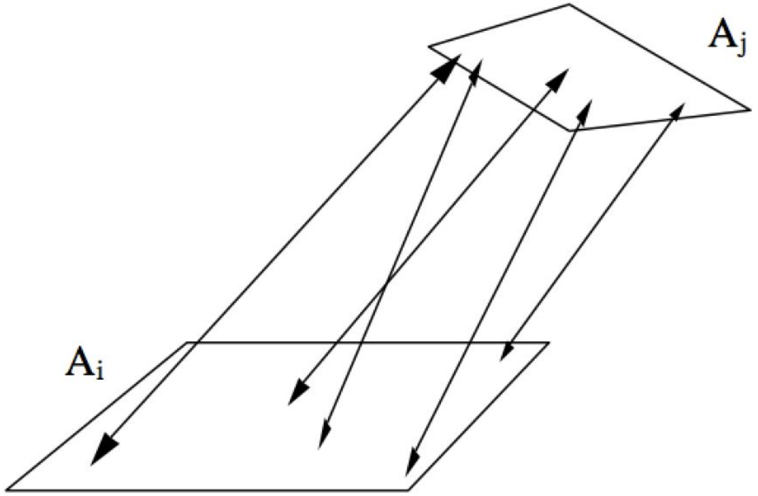


$$F_{ij} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{\cos\theta_i \cos\theta_j}{\pi r^2} \quad V_{ij}\, dA_j\, dA_i$$

Form Factor from Ray Casting

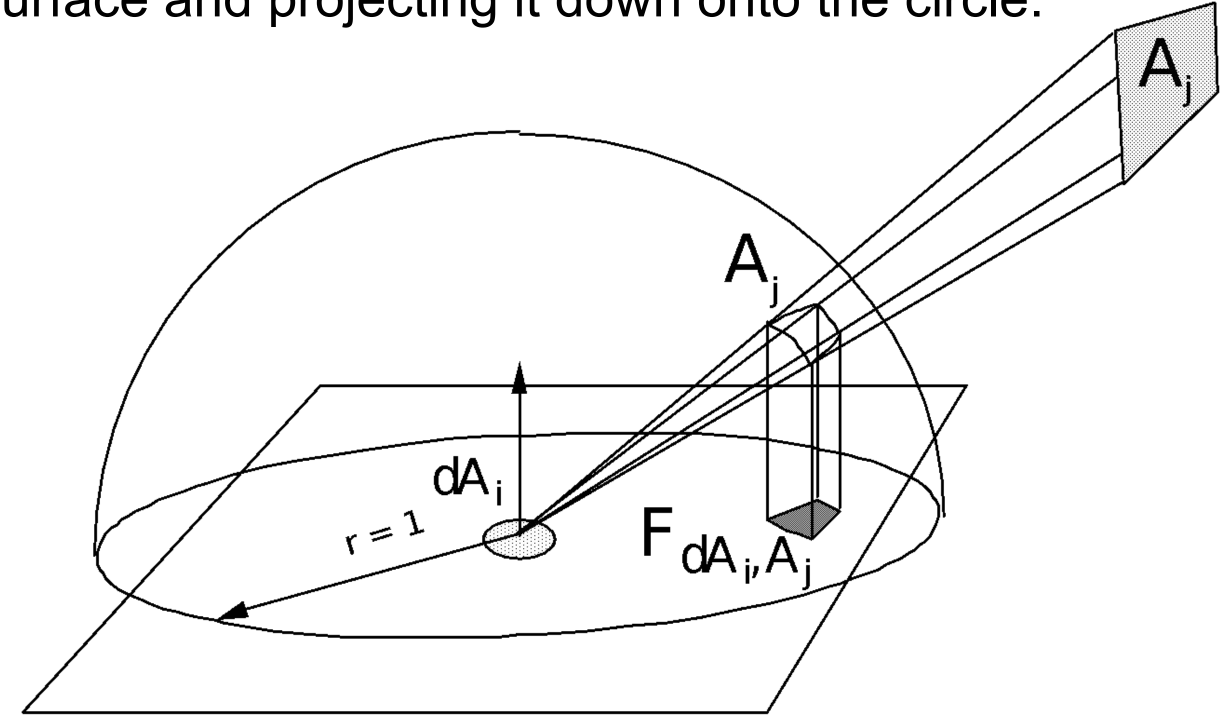
- Cast *n* rays between the two patches
  - Compute visibility (what fraction of rays do not hit an occluder)
  - Integrate the point-to-point form factor
- Permits the computation of the patch-to-patch form factor, as opposed to point-to-patch

Use this for HW3!



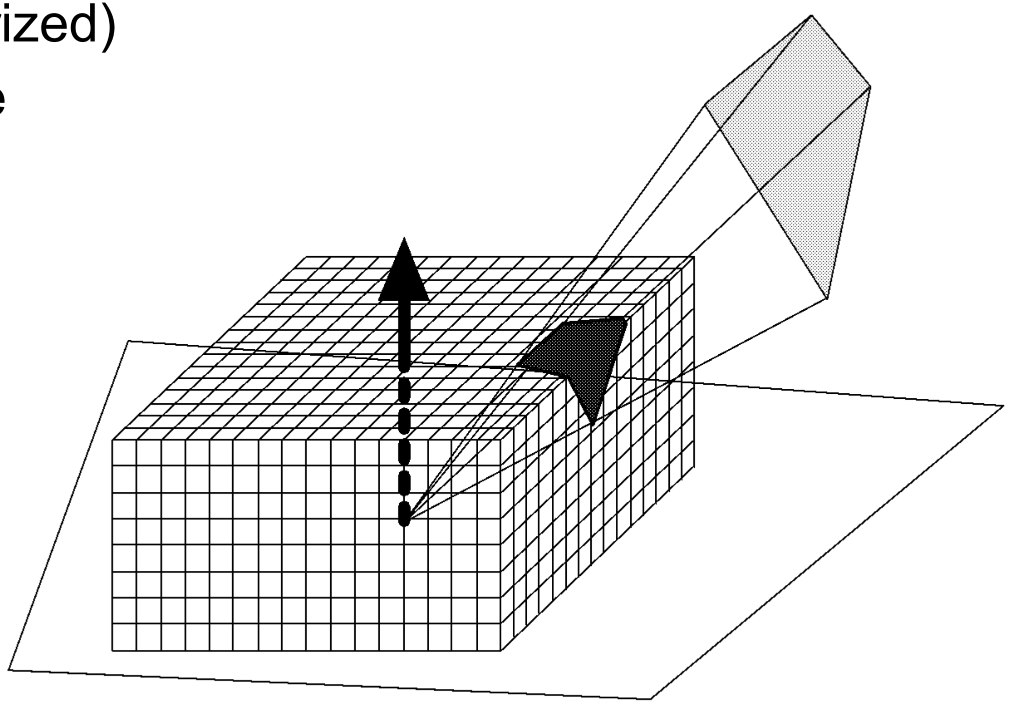
Form Factor Determination

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 *Leverage early graphics hardware…*

- 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



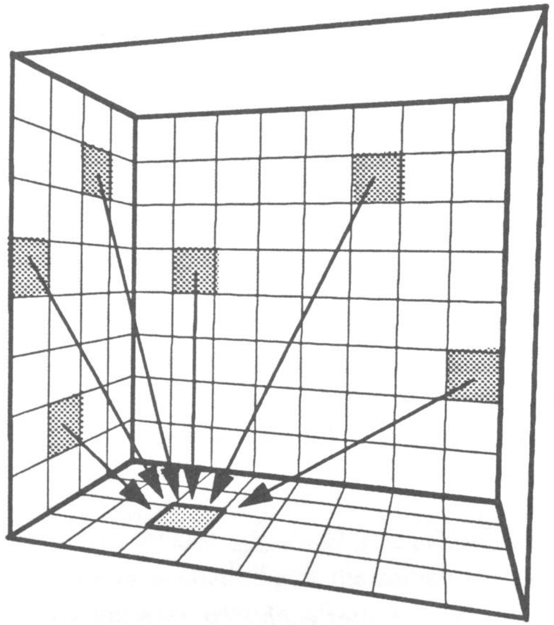
Solving the Radiosity Matrix

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

$$\begin{pmatrix} B_1 \\ B_1 \\ B_2 \\ \vdots \\ B_i \\ B_i \\ \vdots \\ B_n \\ B_n \end{pmatrix} = \begin{pmatrix} E_1 \\ E_2 \\ \vdots \\ E_i \\ \vdots \\ E_n \end{pmatrix} + \begin{pmatrix} \\ \rho_1 F_{i1} & \rho_2 F_{i2} & \cdots & \rho_n F_{in} \\ \\ \end{pmatrix} \begin{pmatrix} B_2 \\ \vdots \\ \\ \vdots \end{pmatrix}$$

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

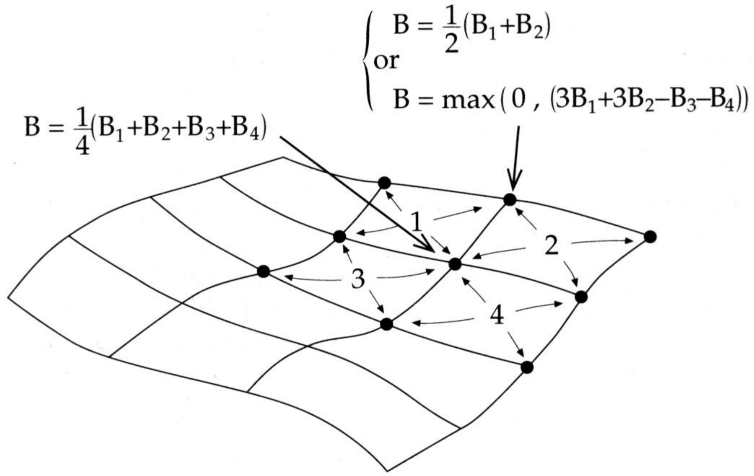
Radiosity values on iteration ***t***



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

Display Interpolated 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



$B = \frac{1}{2}(B_1+B_2)$
or
$B = \max(0, (3B_1+3B_2-B_3-B_4))$
$B = \frac{1}{4}(B_1+B_2+B_3+B_4)$
1
2
3
4

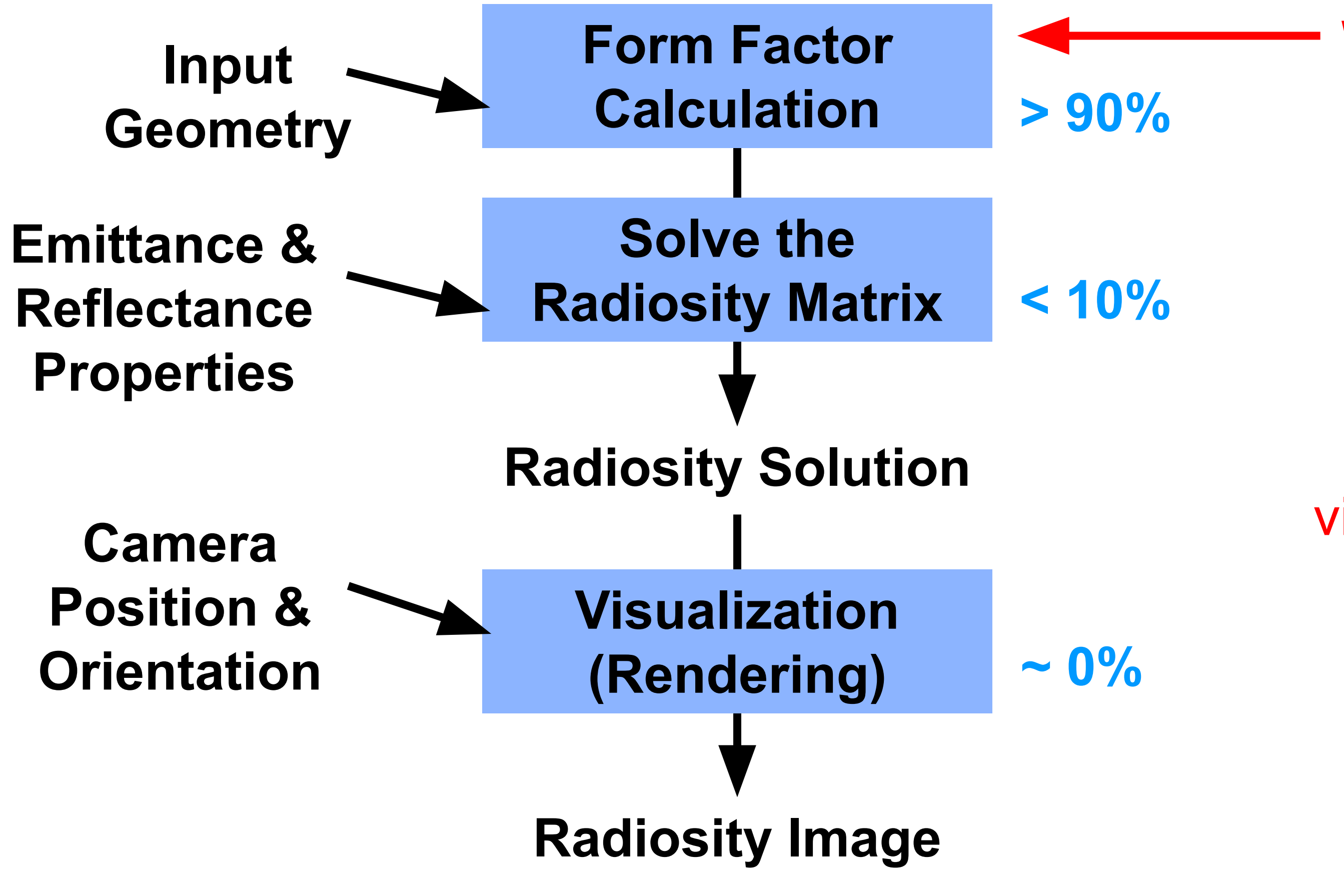
Questions?

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

Today

- HW3 & Final Project Brainstorming
- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors & Solving the Radiosity Matrix
  - Performance & Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Papers for Next Time
- Worksheet

Stages in a Radiosity Solution


Input Geometry
Form Factor Calculation
> 90%
Emittance & Reflectance Properties
Solve the Radiosity Matrix
< 10%
Radiosity Solution
Camera Position & Orientation
Visualization (Rendering)
~ 0%
Radiosity Image

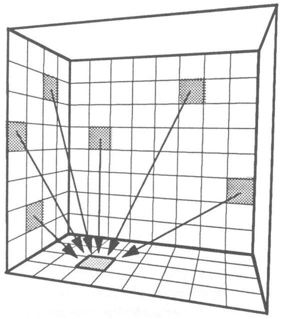
Why so costly?

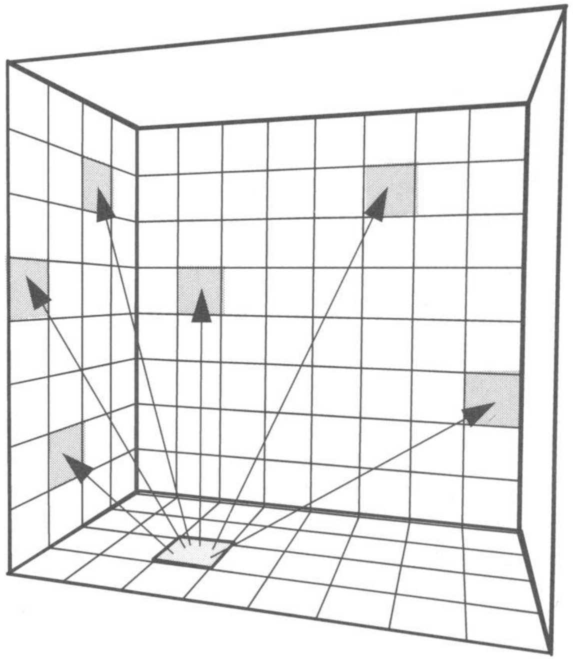
Calculation & storage of n^2 form factors

(n^3 for naive visibility calculation)

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…
  - Let's distribute the radiance of the patch with the most *undistributed radiance*.



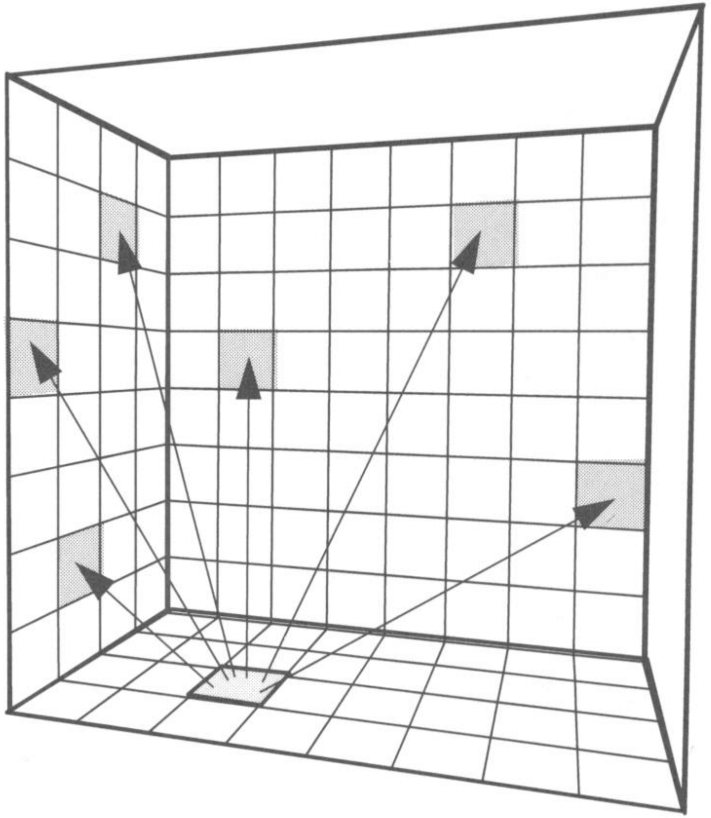


Use this for HW3!

Re-Ordered Solver for Progressive Refinement

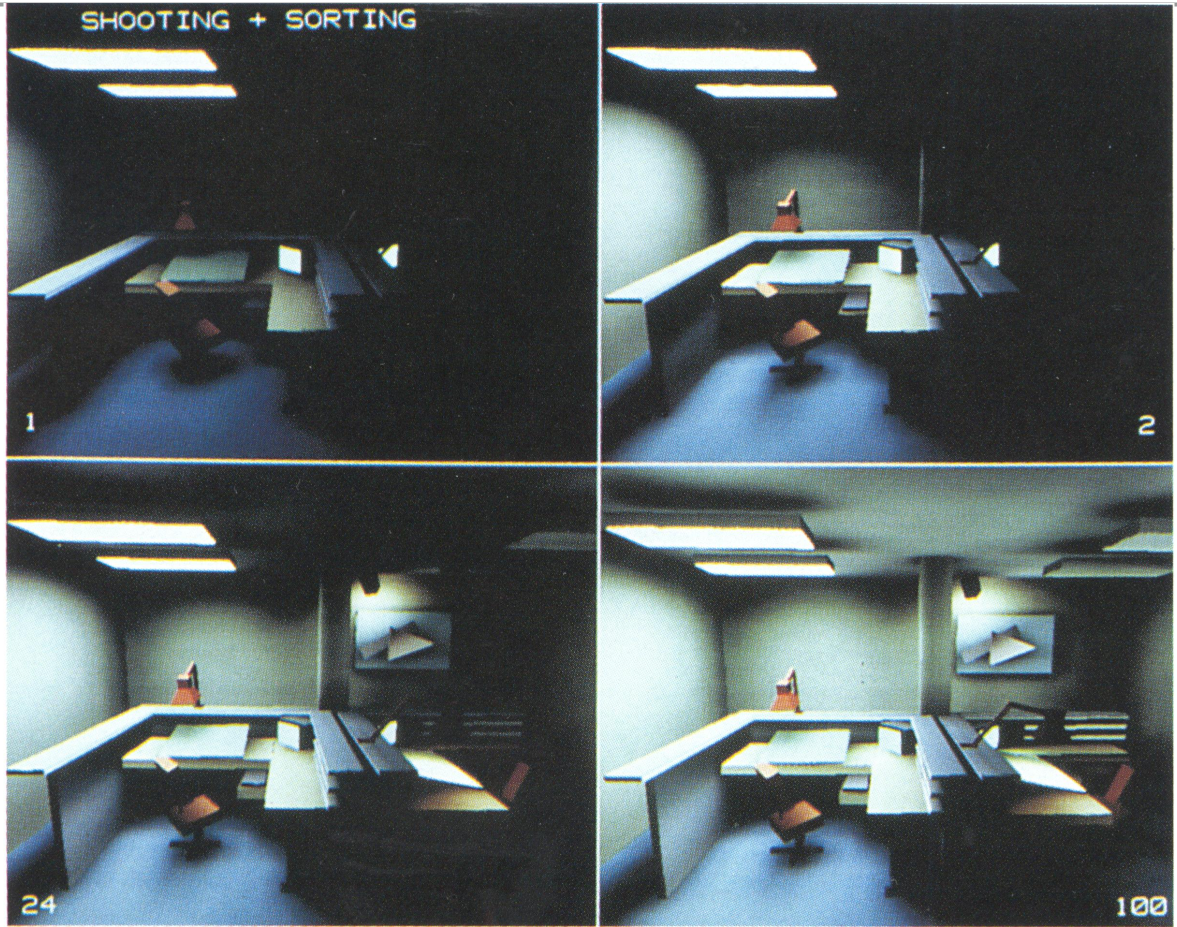
- *Shooting:* the radiosity of all patches is updated for each iteration:

$$\begin{pmatrix} B_1 \\ B_2 \\ \\ \cdots \\ \\ B_n \end{pmatrix} = \begin{pmatrix} E_1 \\ E_2 \\ \\ \cdots \\ \\ E_n \end{pmatrix} + \begin{pmatrix} \cdots & \rho_1 F_{1i} & \cdots \\ \cdots & \rho_2 F_{2i} & \cdots \\ & & \\ & & \\ & & \\ \cdots & \rho_n F_{ni} & \cdots \end{pmatrix} + \begin{pmatrix} \\ \cdots \\ B_i \\ \cdots \\ \\ \end{pmatrix}$$

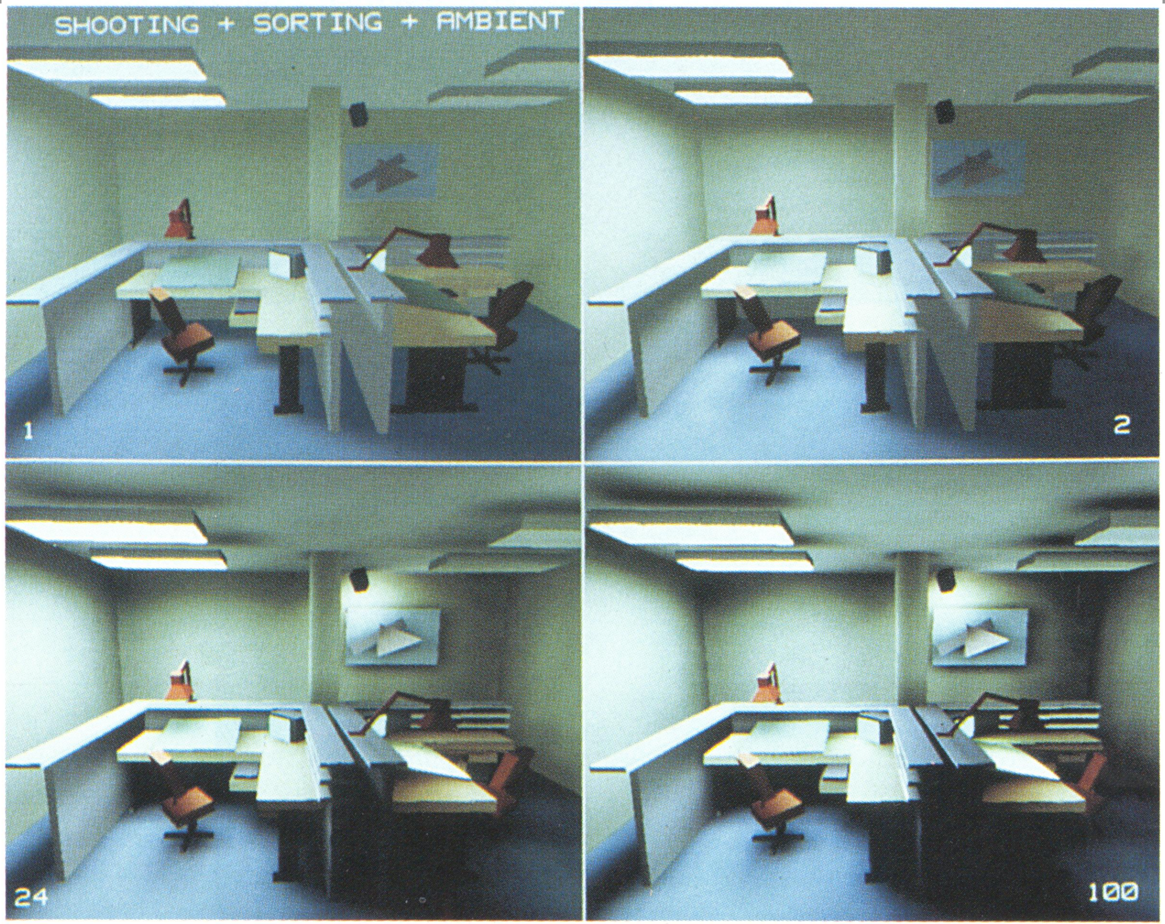


- This method is fundamentally a *Southwell Relaxation*

Progressive Refinement without Ambient Term



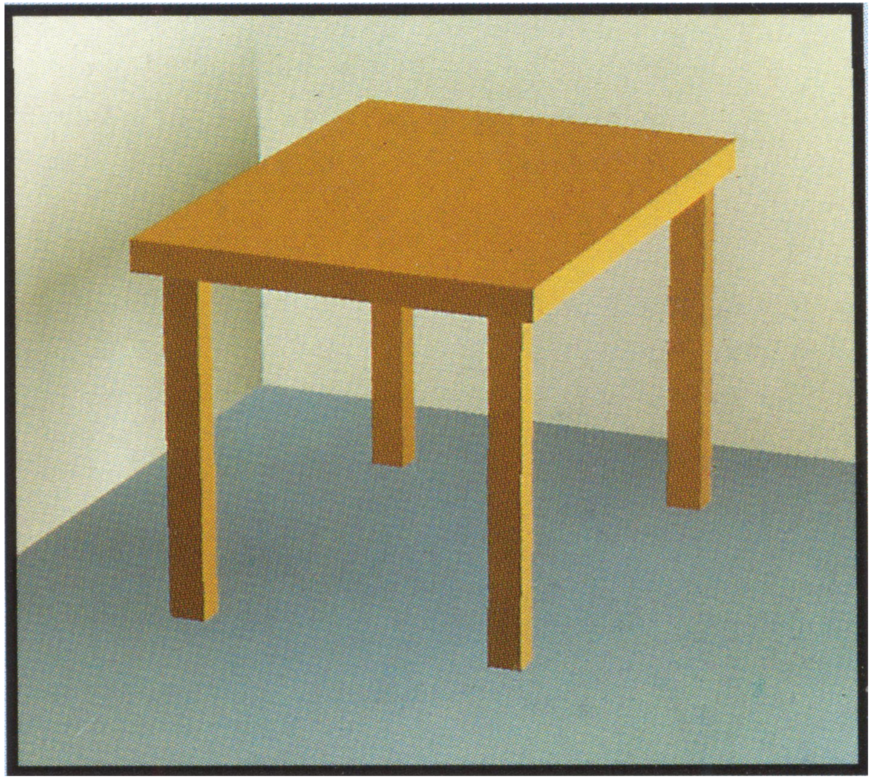
Progressive Refinement with Ambient Term



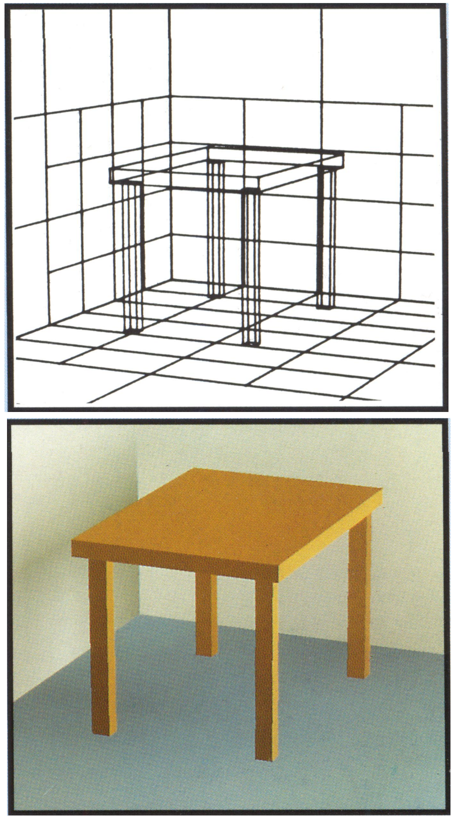
Questions? *Lightscape* *http://www.lightscape.com*

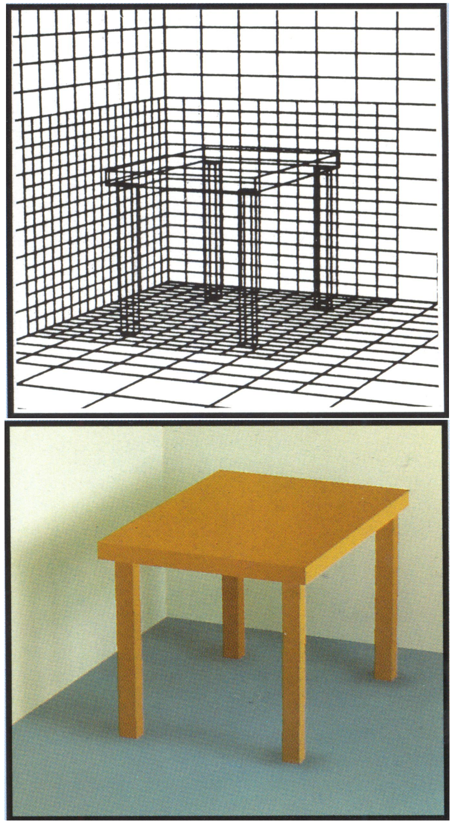
Increasing the Accuracy of the Solution

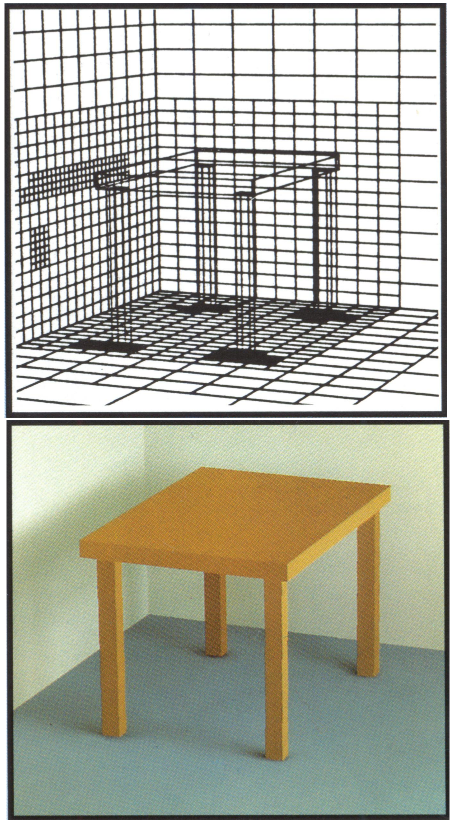
- **What's wrong with this picture?**
- Image quality is a function of patch size
- Compute a solution on a uniform initial mesh, then refine the mesh in areas that exceed some error tolerance:
  - shadow boundaries
  - other areas with a high radiosity gradient



Adaptive Subdivision of Patches

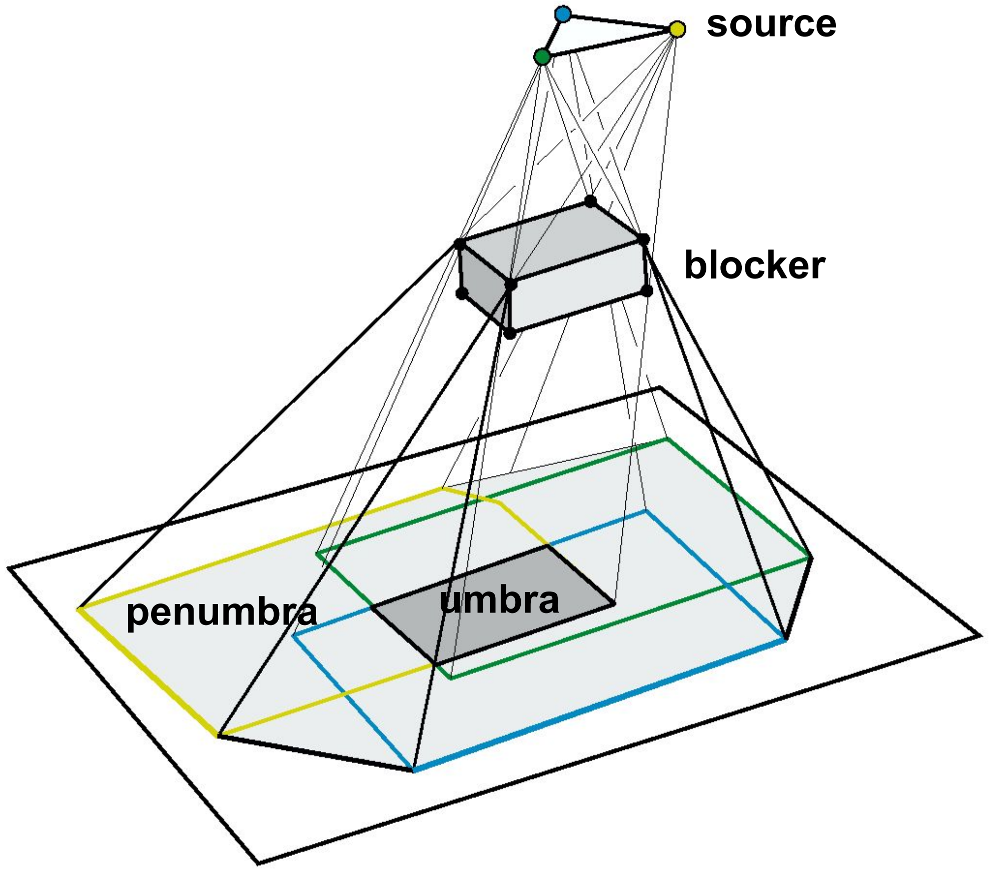

Coarse patch solution
(145 patches)


Improved solution
(1021 subpatches)

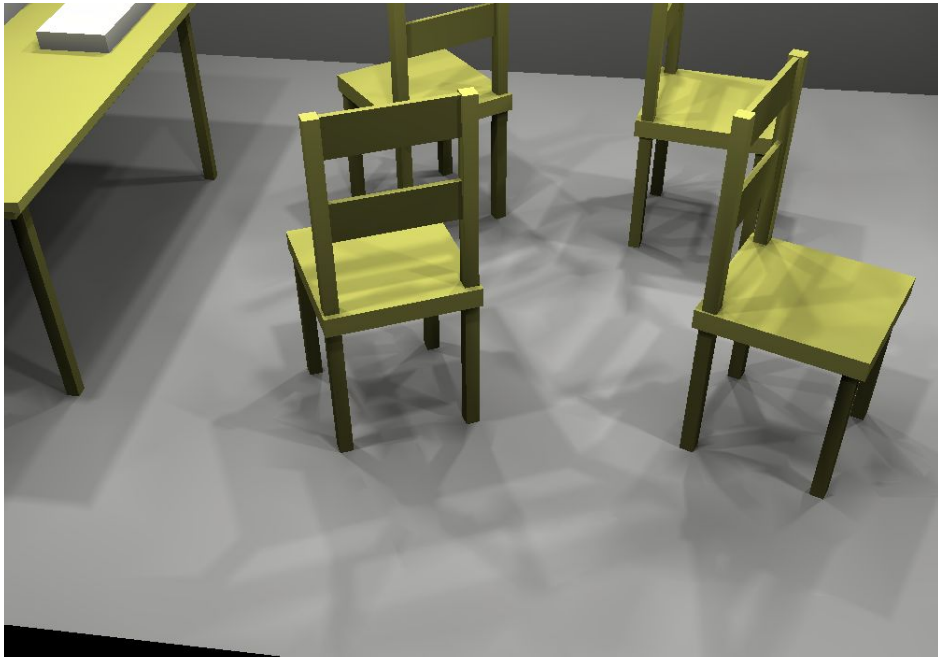

Adaptive subdivision
(1306 subpatches)

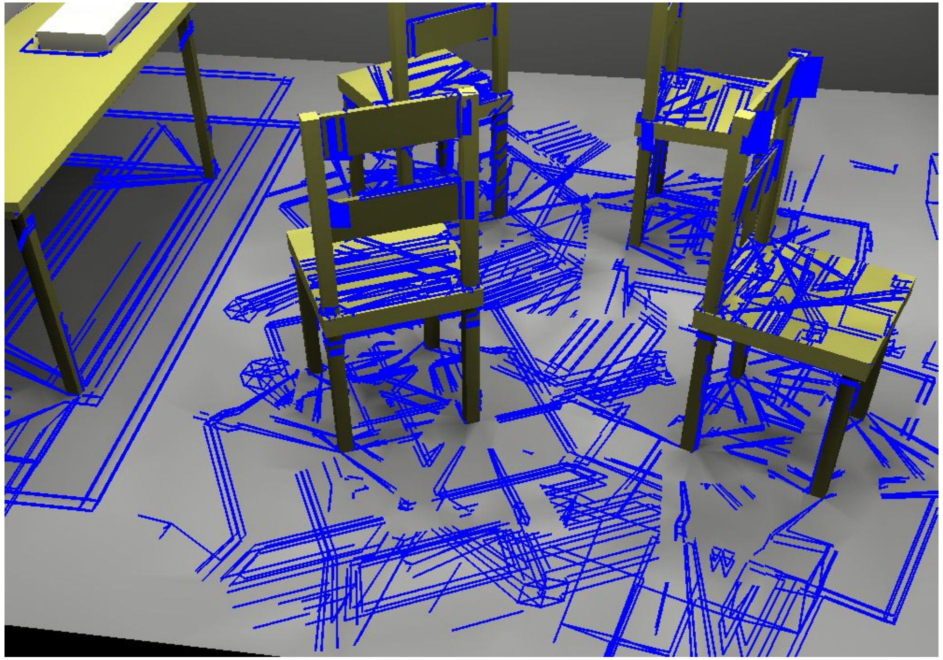
Discontinuity Meshing

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



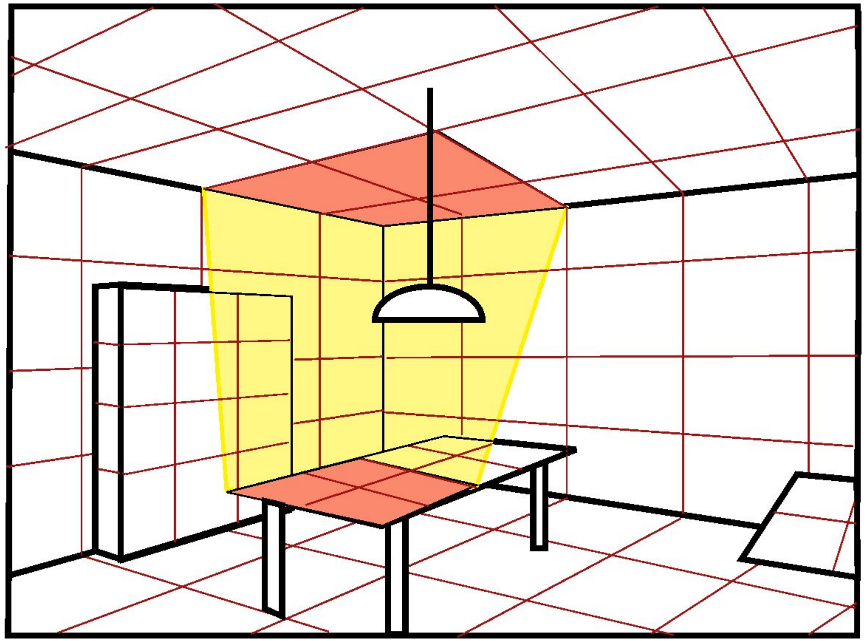
“Fast and Accurate Hierarchical Radiosity Using Global Visibility” Durand, Drettakis, & Puech 1999





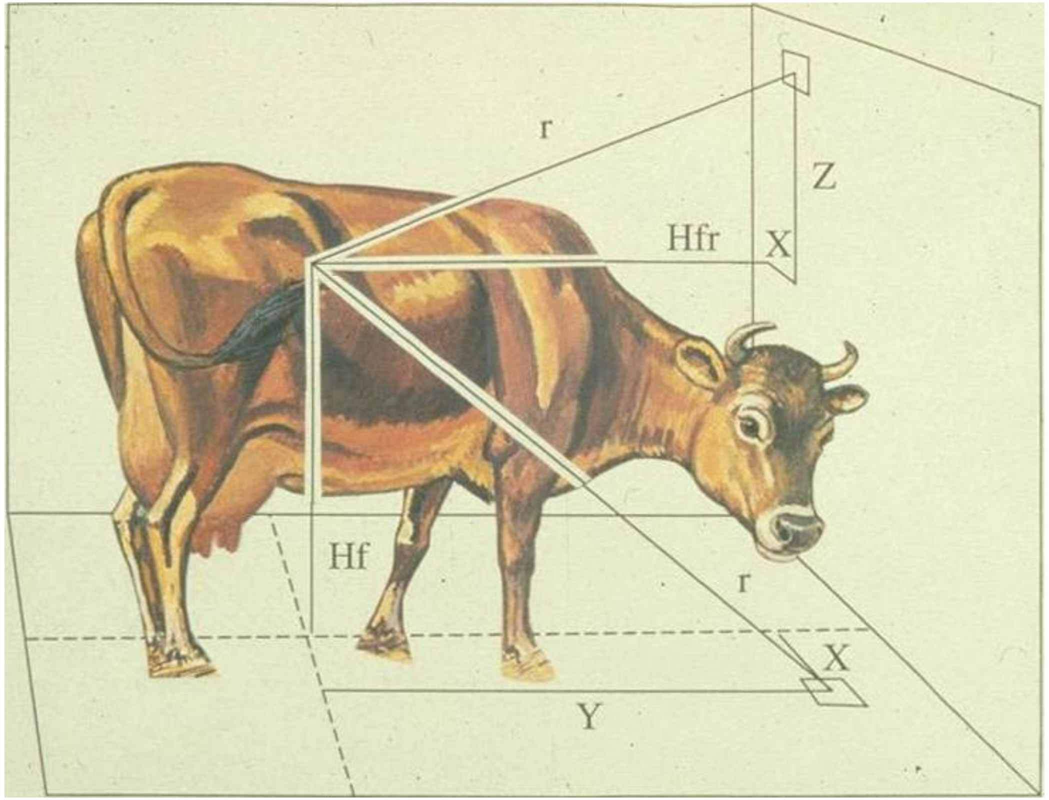
Hierarchical Radiosity

- 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?

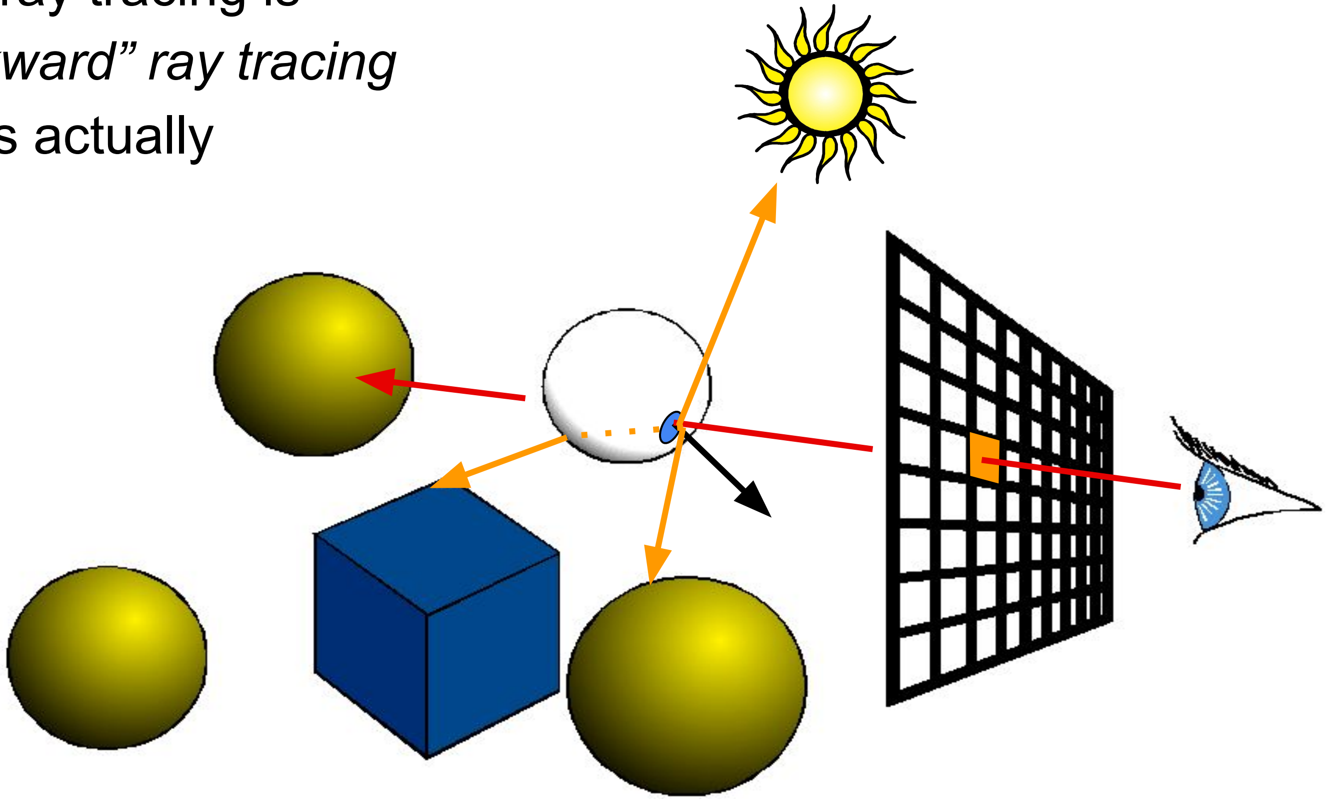
Questions? *Lightscape* *http://www.lightscape.com*

Today

- HW3 & Final Project Brainstorming
- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors & Solving the Radiosity Matrix
  - Performance & Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Papers for Next Time
- Worksheet

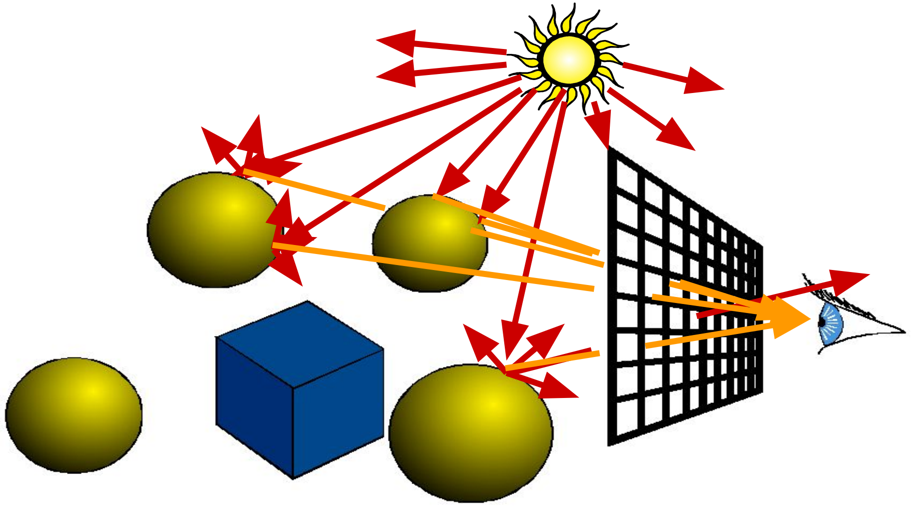
Does Ray Tracing Simulate Physics?

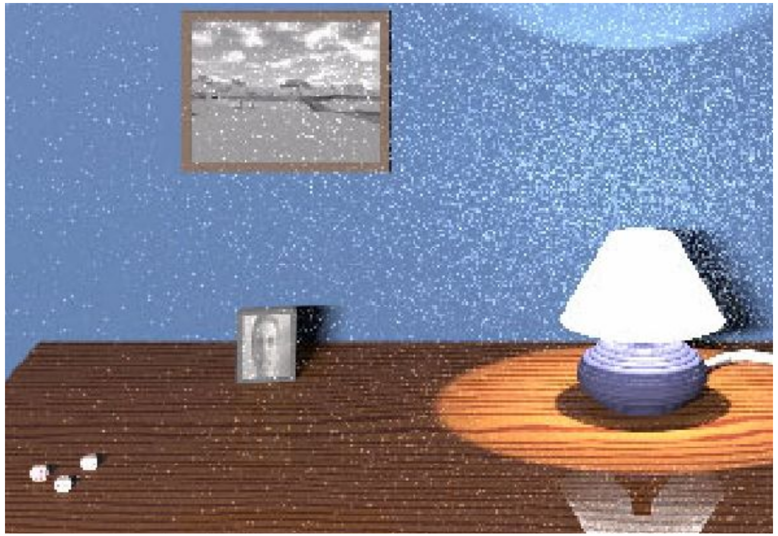
- No…. traditional ray tracing is also called “*backward” ray tracing*
- In reality, photons actually travel from the light to the eye



Forward Ray Tracing

- Start tracing from the light source
- Sample every direction uniformly
- Reflect/transmit rays based sampling the BRDF (material) probability distribution
- Very, very low probability the ray reaches camera/eye/image plane
- Tweak: Always send a final bounce ray to the eye (cheat the BRDF)
- Impractically slow & noisy!
- *More ideas to talk about later…*

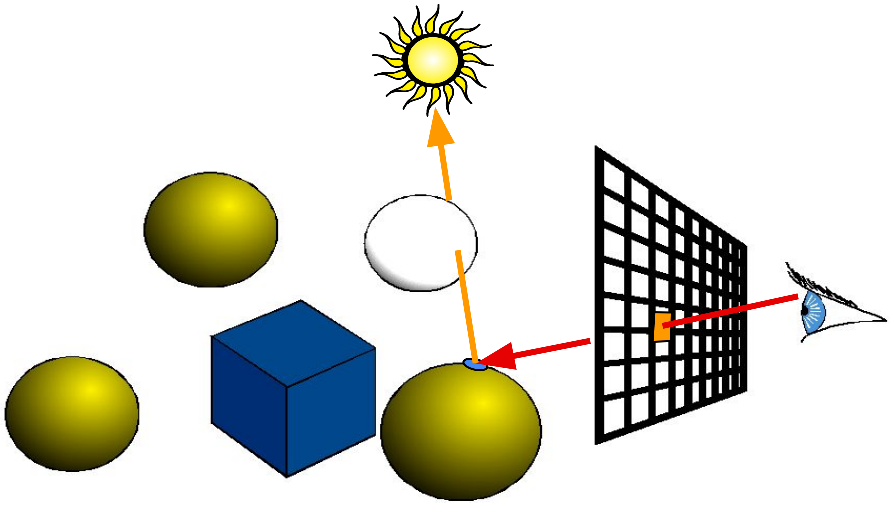


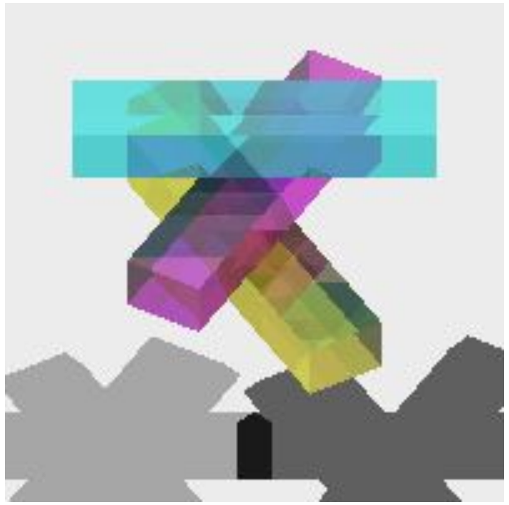


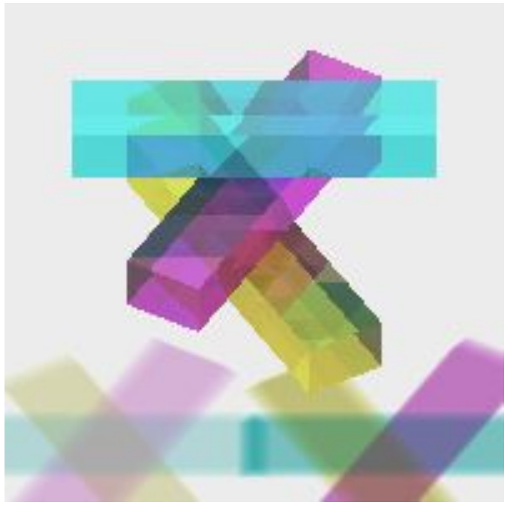
Henrik Wann Jensen

Transparent Shadows?

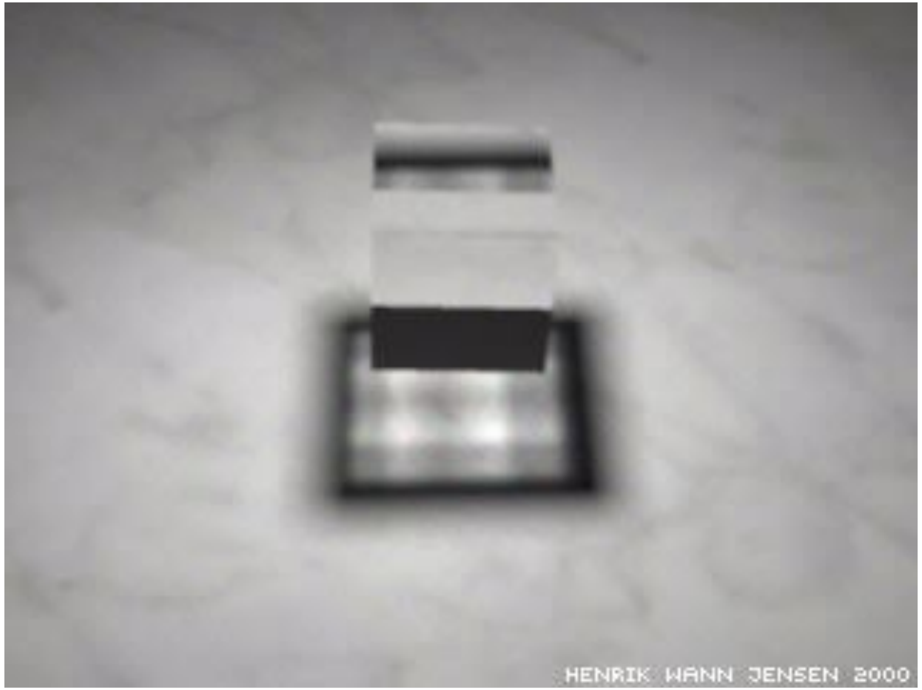
- What to do if the shadow ray sent to the light source intersects a transparent object?
  - Pretend it's opaque?
  - Multiply by transparency color?
    ignores refraction & does not produce caustics
- Unfortunately, ray tracing is full of dirty tricks

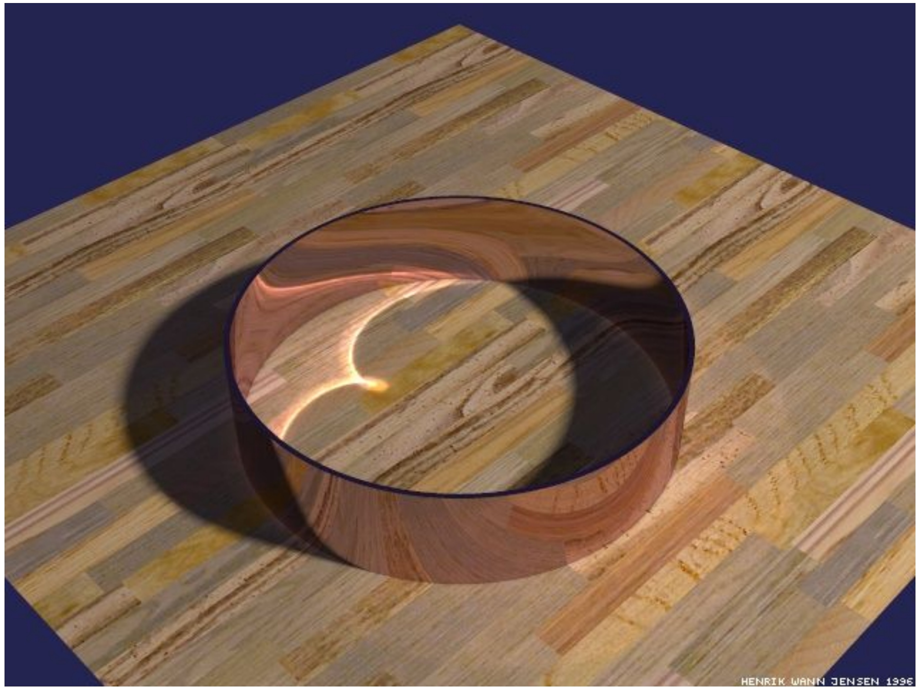






Is this Traditional Ray Tracing?



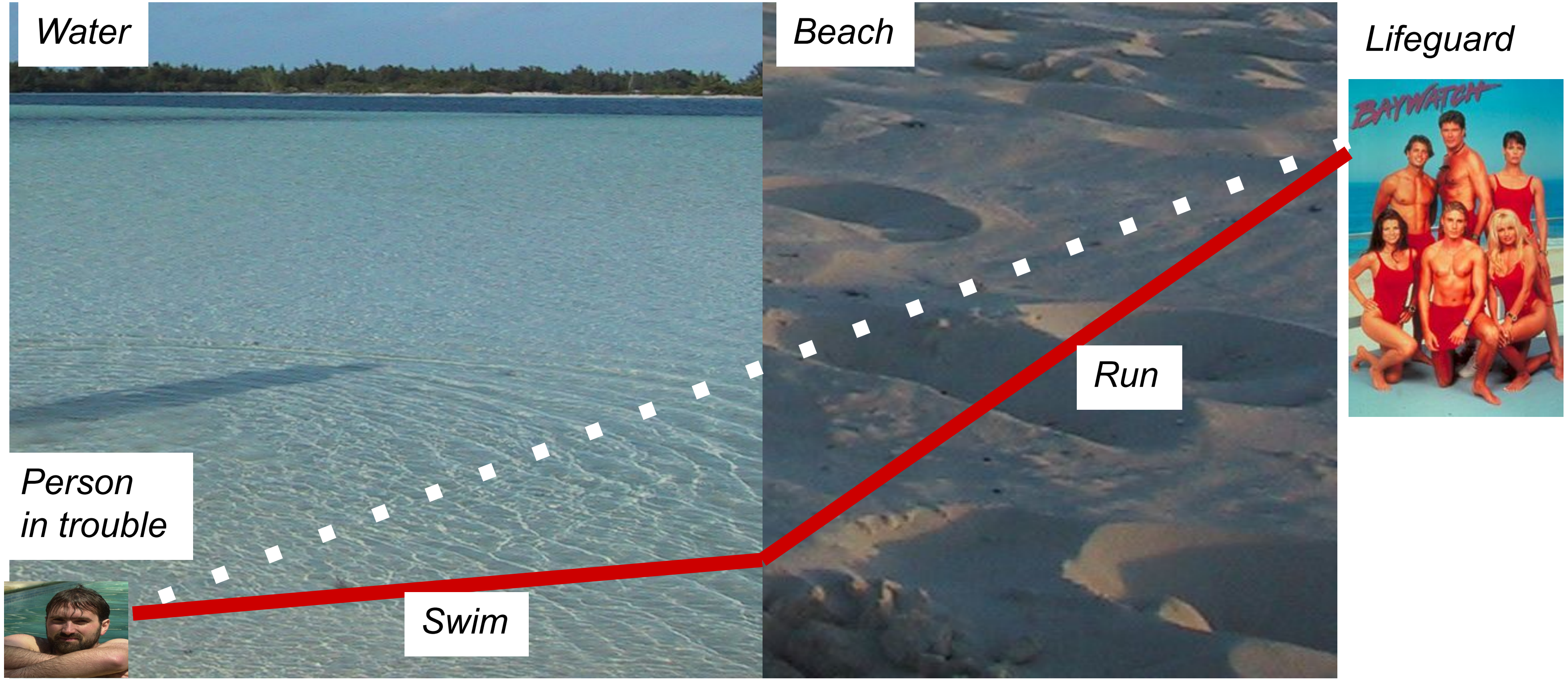


Images by Henrik Wann Jensen

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: What is the optimal rescue path?

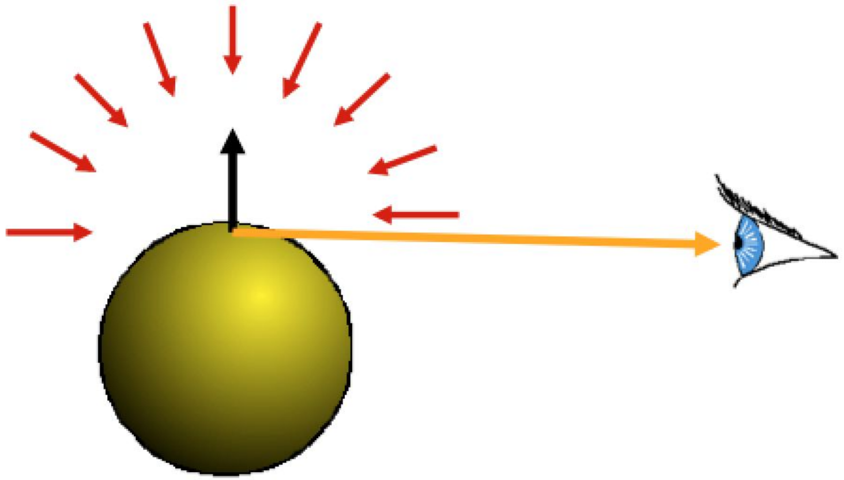


Today

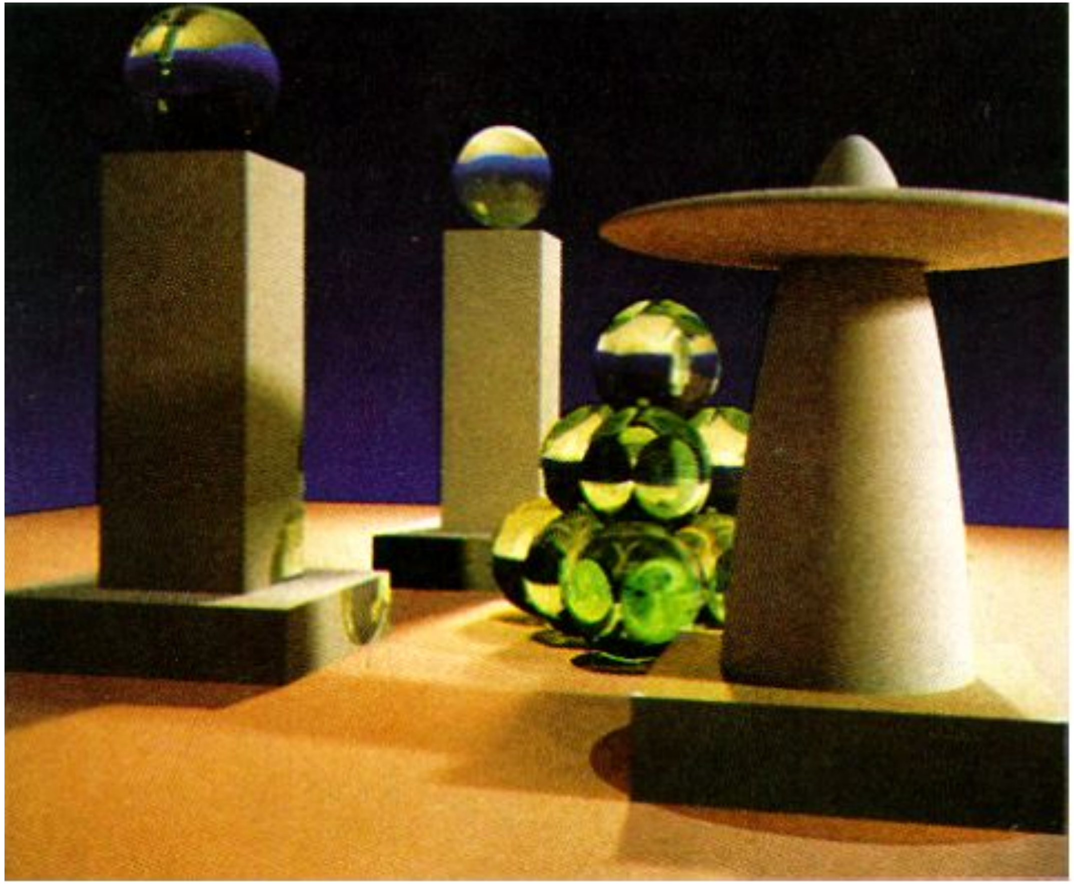
- HW3 & Final Project Brainstorming
- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors & Solving the Radiosity Matrix
  - Performance & Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Papers for Next Time
- Worksheet

The Rendering Equation

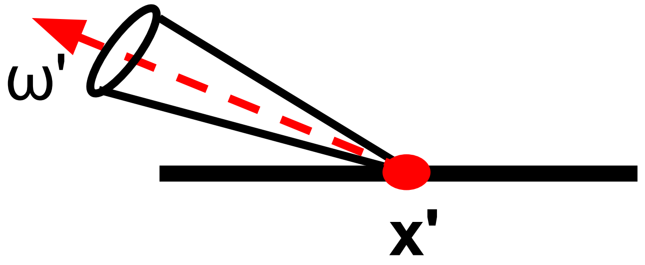
- Clean mathematical framework for light-transport simulation
- At each point, outgoing light in one direction
  is the integral of incoming light in all directions
  multiplied by reflectance property



“The Rendering Equation”, *Kajiya, SIGGRAPH 1986*



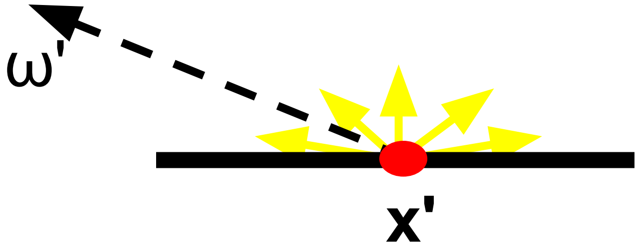
The Rendering Equation



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

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

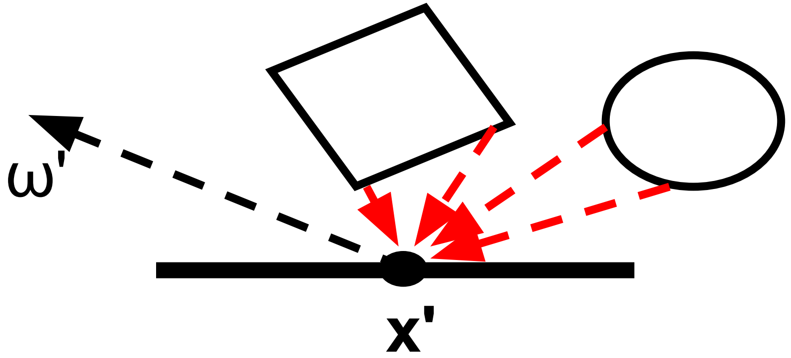
The Rendering Equation



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

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

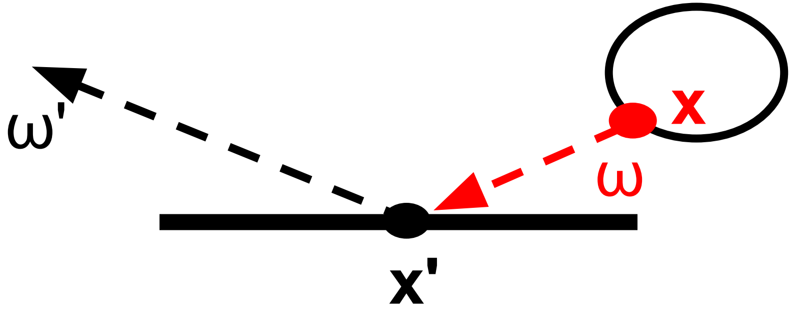
The Rendering Equation



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

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

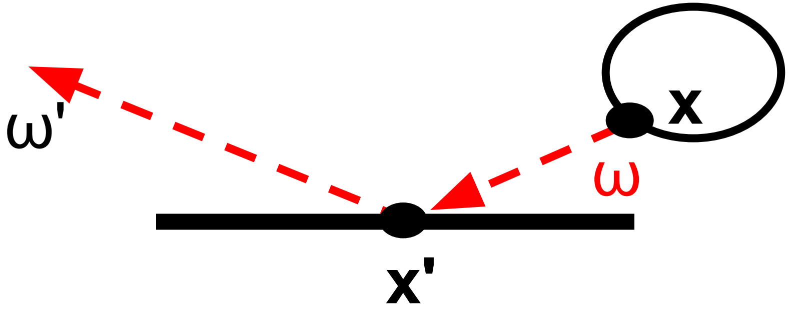
The Rendering Equation



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

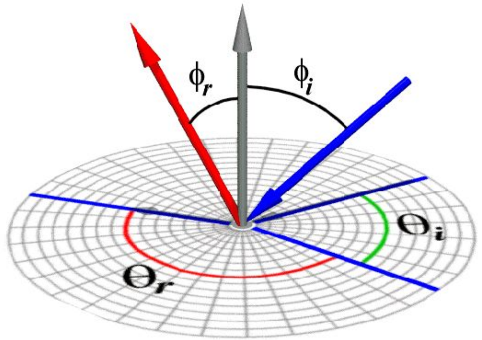
For each x, compute L(x, ω), the radiance at point x in the direction ω (from x to x')

The Rendering Equation

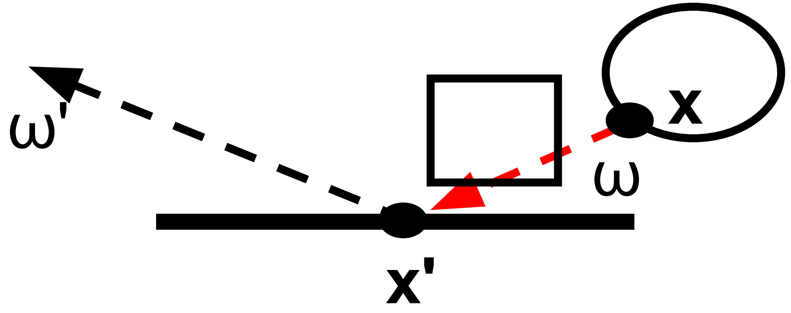


$$L(x',\omega') = E(x',\omega') + \int \rho_{x'}(\omega,\omega') L(x,\omega) G(x,x') V(x,x')\, dA$$

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



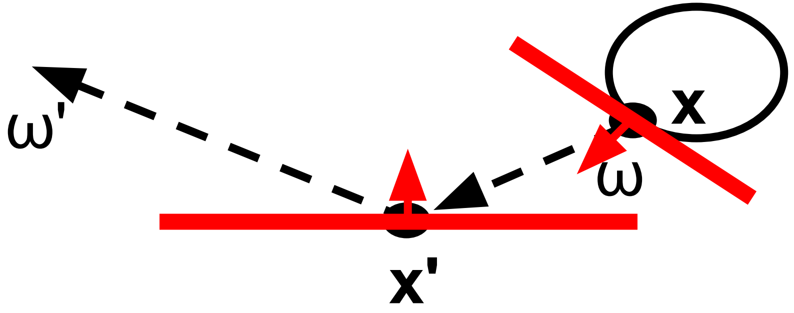
The Rendering Equation



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

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

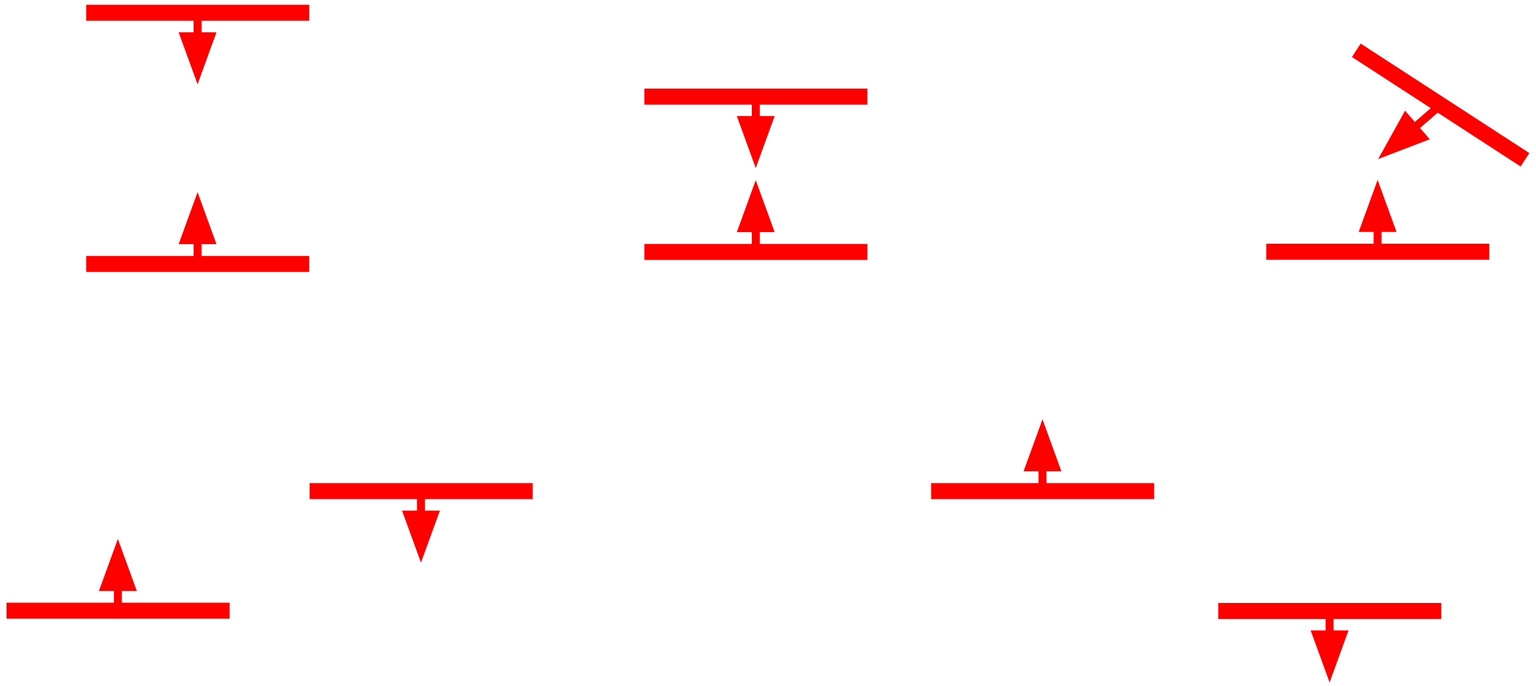


$$L(x',\omega') = E(x',\omega') + \int \rho_{x'}(\omega,\omega') L(x,\omega) G(x,x') V(x,x')\, dA$$

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

Intuition about Geometry Term G(x,x')?

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



Rendering Equation ➔ Radiosity

Different rendering techniques can be expressed as a subset/approximation of the full rendering equation.

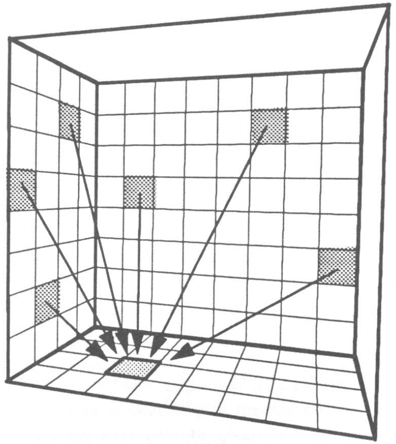
$$L(x',\omega') = E(x',\omega') + \int \rho_{x'}(\omega,\omega') L(x,\omega) G(x,x') V(x,x')\, dA$$

Radiosity assumption:
perfectly diffuse surfaces (not directional)

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

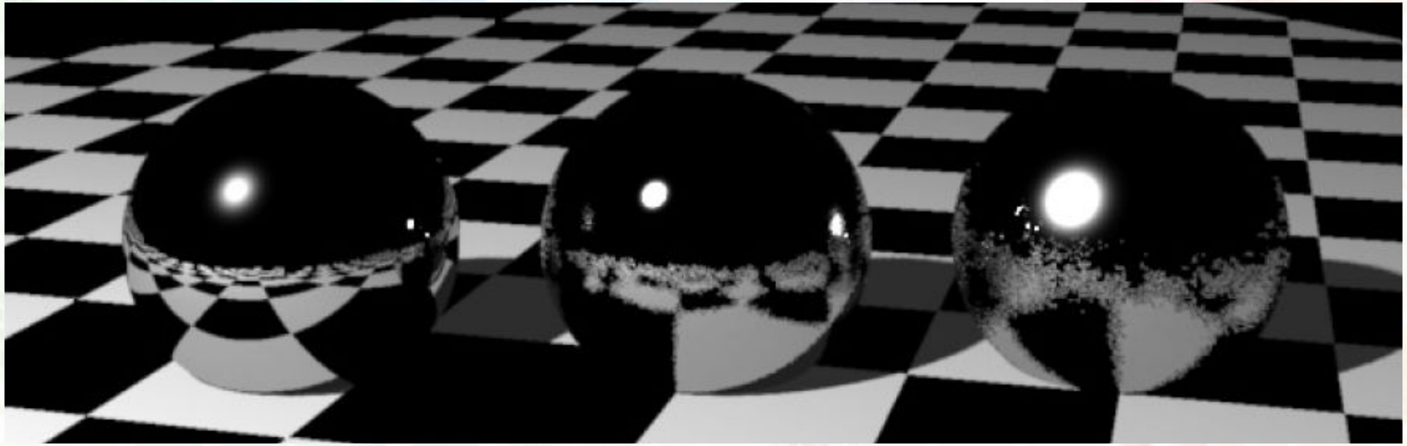
discretize

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

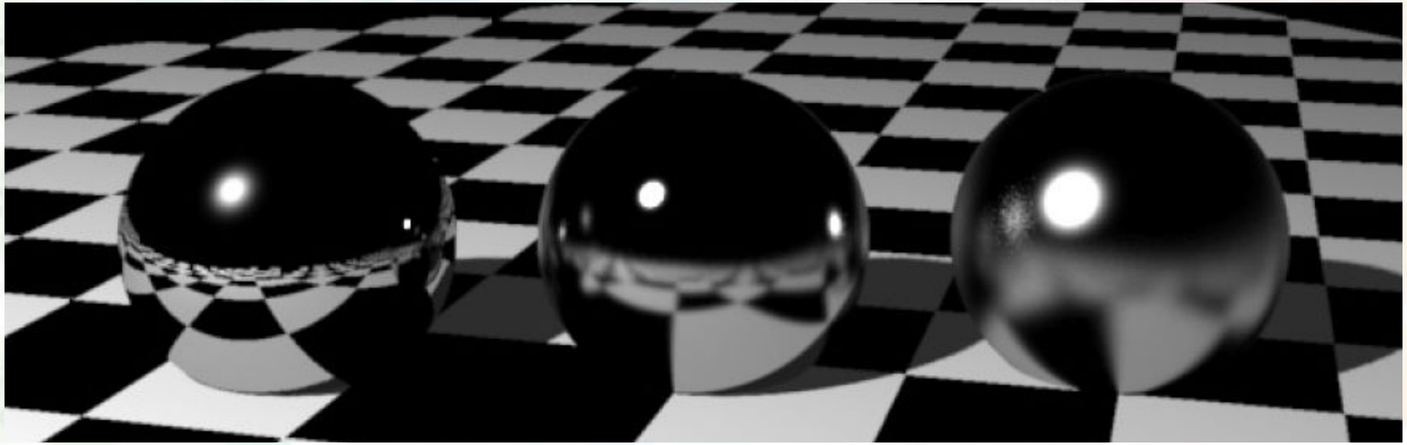


Questions?

1 glossy sample per pixel



256 glossy samples per pixel

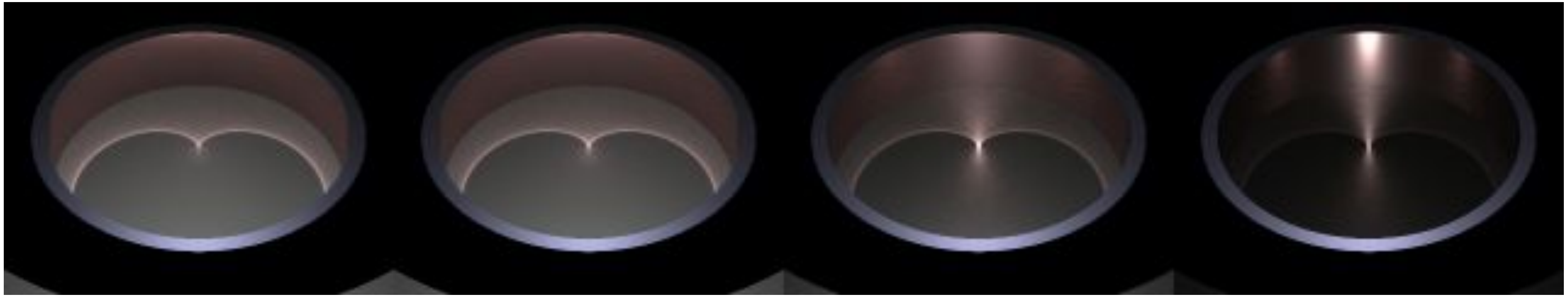


Today

- HW3 & Final Project Brainstorming
- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors & Solving the Radiosity Matrix
  - Performance & Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Papers for Next Time
- Worksheet

Readings for Next Time: *(pick one)*

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



Readings for Next Time: *(pick one)*

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

HENRIK WANN JENSEN 1996

HENRIK WANN JENSEN 1996

Today

- HW3 & Final Project Brainstorming
- Paper for Today
- Leftover from Last Time:
  - Radiosity Overview
  - Calculating the Form Factors & Solving the Radiosity Matrix
  - Performance & Advanced Radiosity
- Does Ray Tracing Simulate Physics?
- The Rendering Equation
- Papers for Next Time
- Worksheet

Worksheet: Radiosity Form Factors