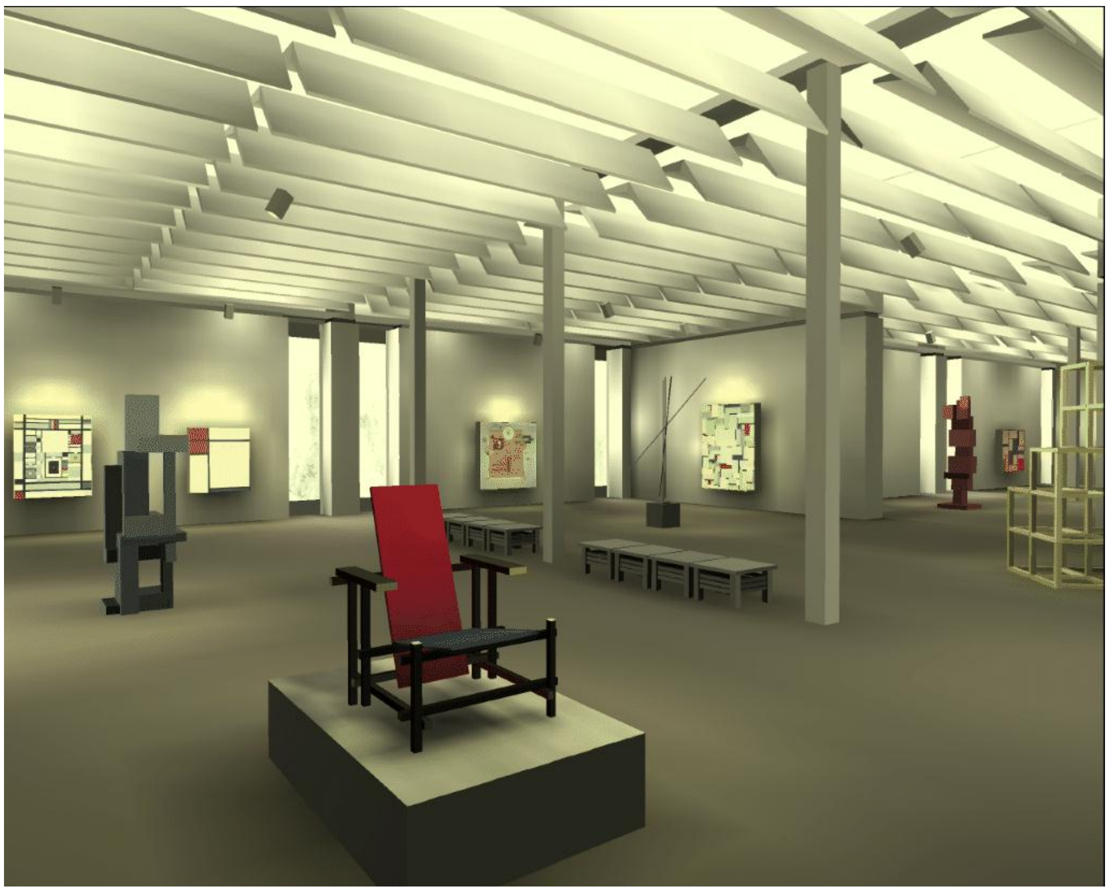
Global Illumination with Radiosity

Slides from
Fredo Durand and Barb Cutler, MIT
Pat Hanrahan, Stanford U.

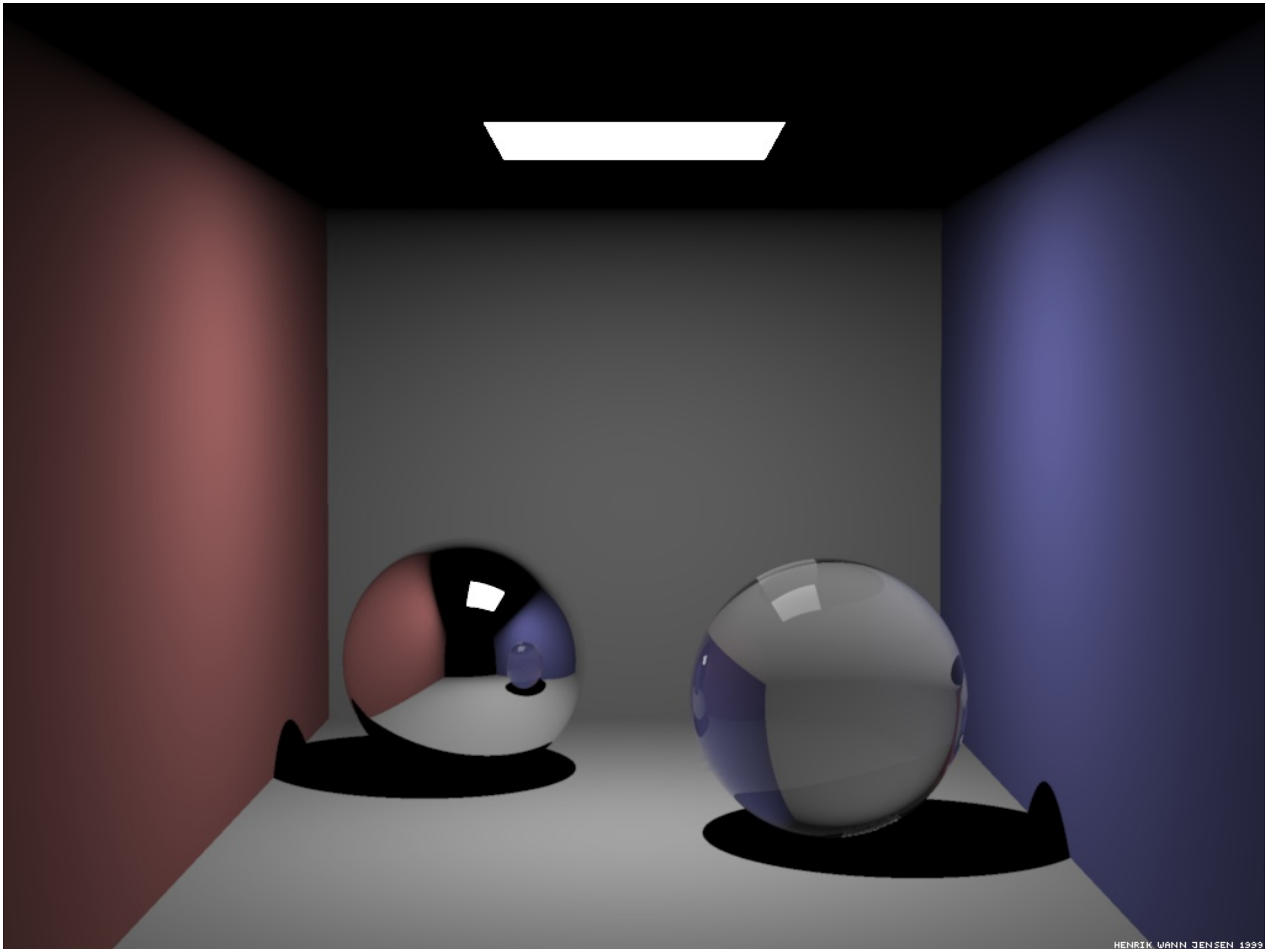
Why global illumination with radiosity?

- Simulate light inter-reflections (indirect lighting)
  - e.g. in a room much of the light is indirect

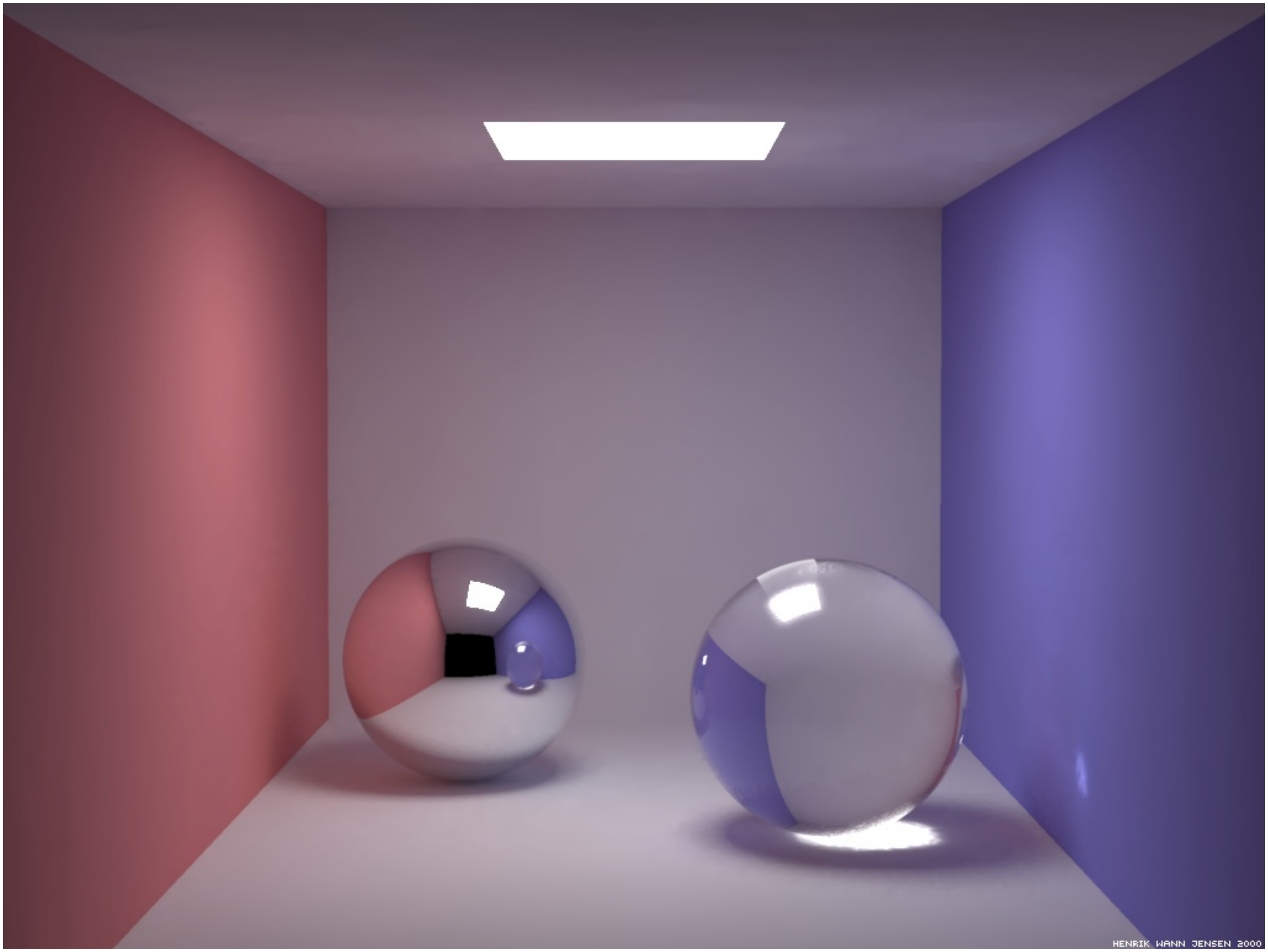


Museum simulation.
Program of Computer Graphics,
Cornell University.
50,000 patches.
Note indirect lighting from ceiling.

Direct illumination



Global Illumination



Radiosity Overview

- Classic radiosity = finite element method
- Assumptions
  - Diffuse reflectance
  - Usually polygonal surfaces
- Advantages
  - Soft shadows and indirect illumination
  - View-independent solutions
  - Precompute for a set of light sources
  - Useful for walkthroughs

Why Radiosity?

- Sculpture by John Ferren
- *Diffuse* panels

photograph:

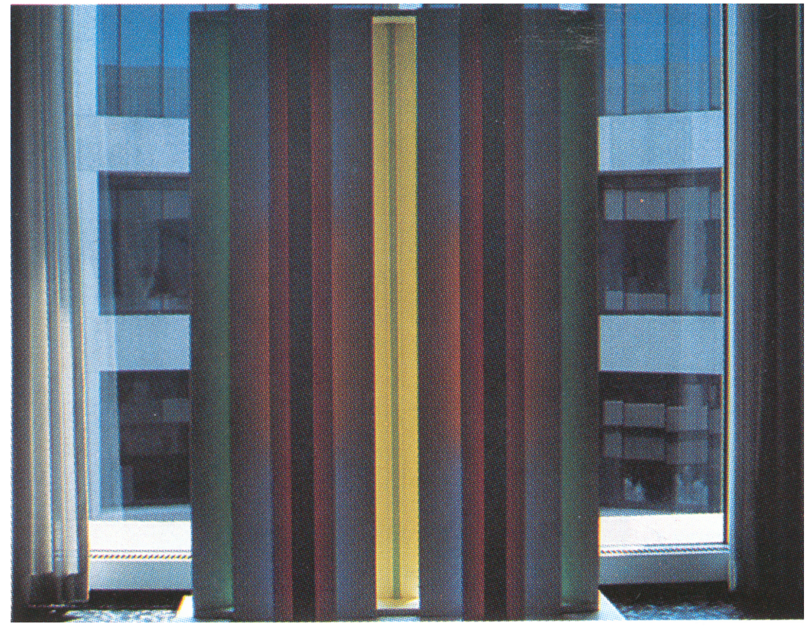
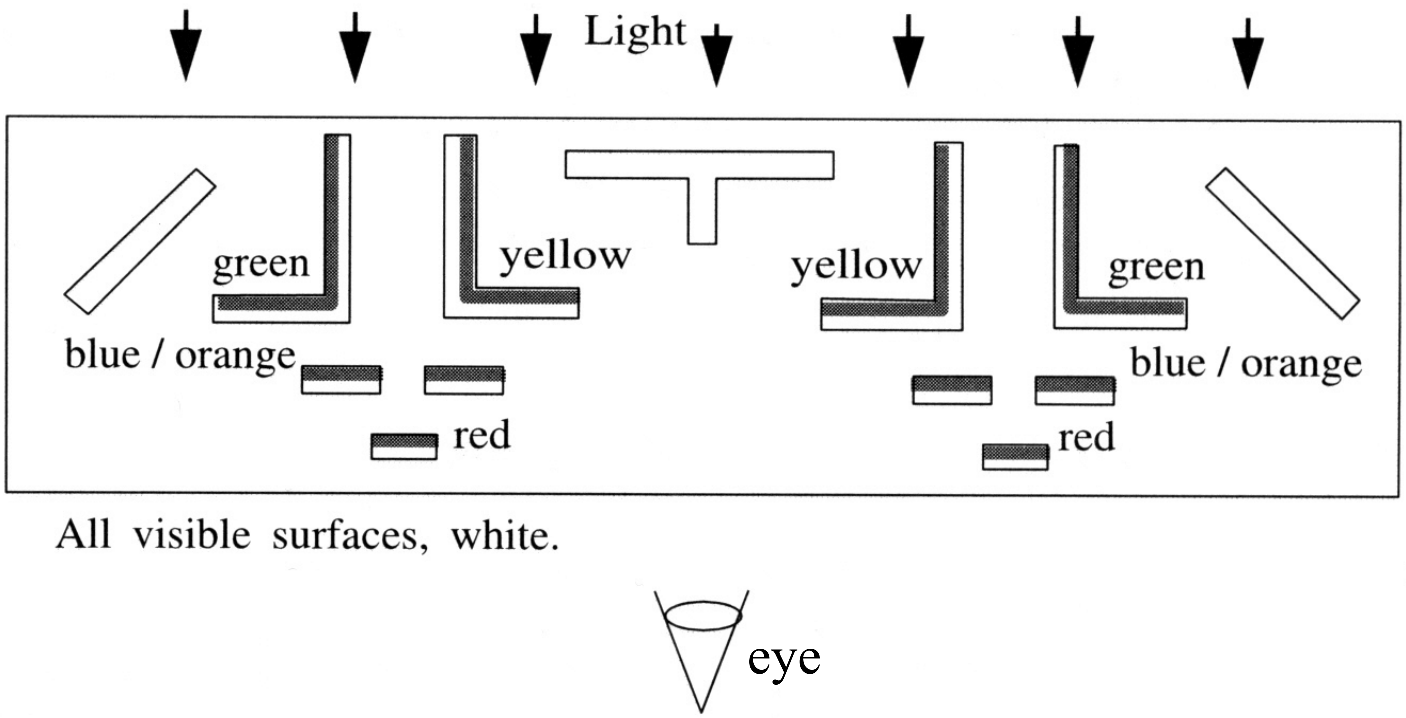
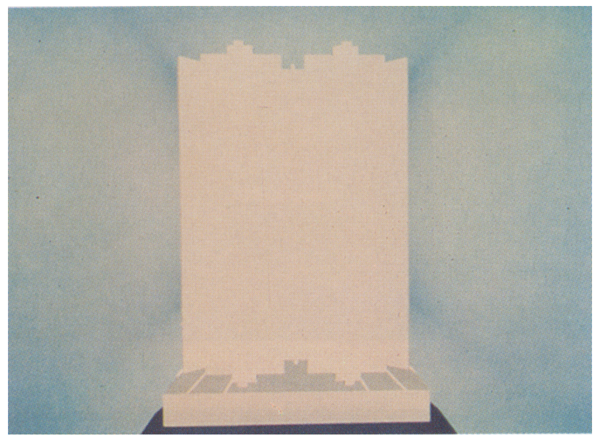


diagram
from above:



Radiosity vs. Ray Tracing

Original sculpture by John Ferren lit by daylight from behind.



Ray traced image. A standard ray tracer cannot simulate the interreflection of light between diffuse surfaces.

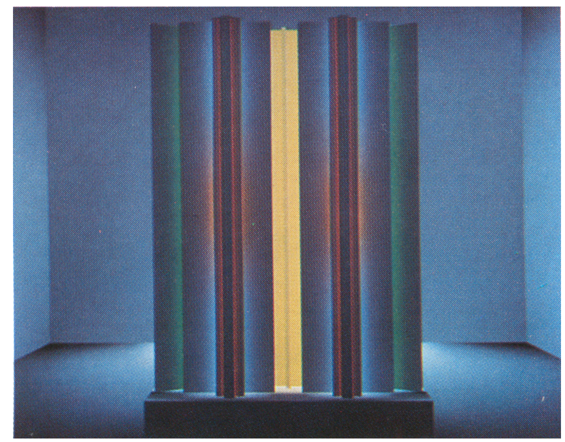
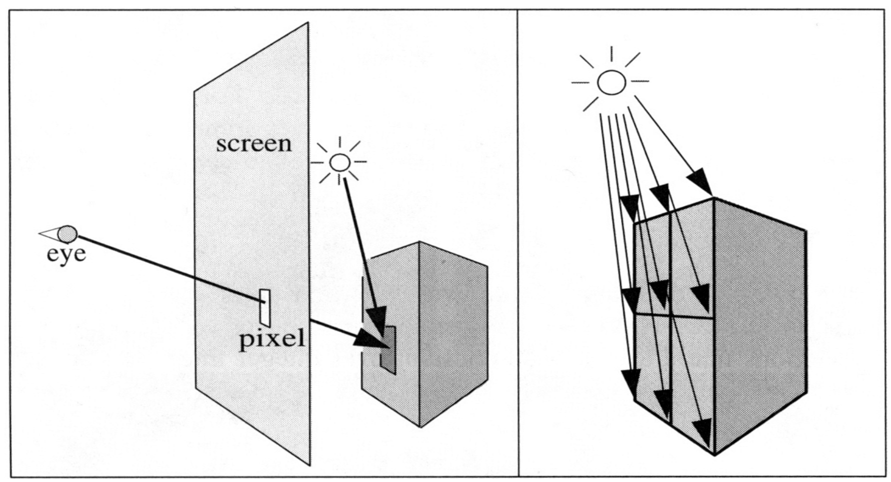


Image rendered with radiosity. note color bleeding effects.

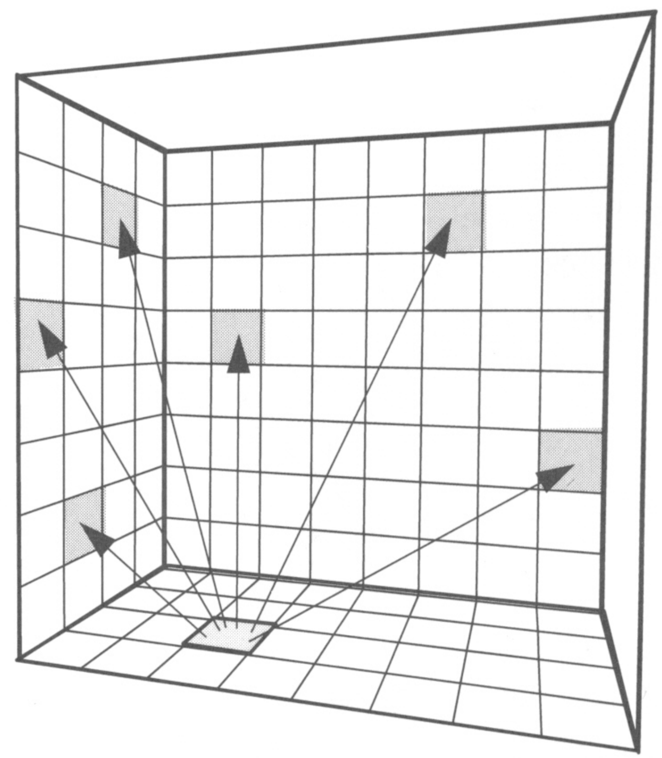
Radiosity vs. Ray Tracing

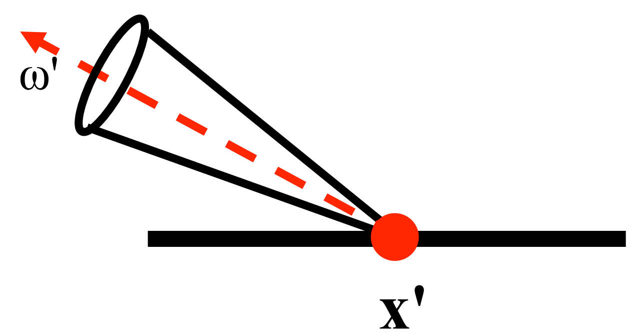
- Ray tracing is an *image-space* algorithm
  - If the camera is moved, we have to start over
- Radiosity is computed in *object-space*
  - View-independent (just don't move the light)
  - Can pre-compute complex lighting to allow interactive walkthroughs



Radiosity Overview

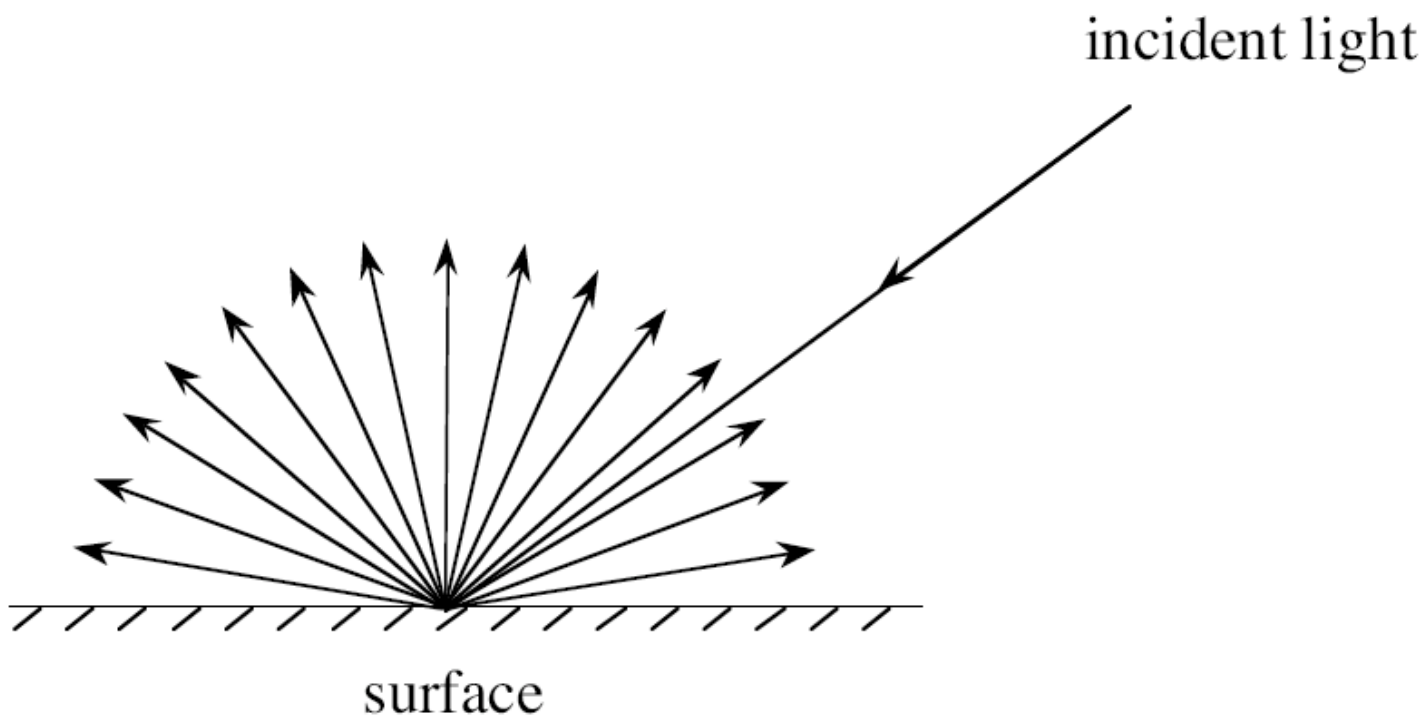
- Surfaces are assumed to be perfectly Lambertian (diffuse)
  - reflect incident light in all directions with equal intensity
- The scene is divided into a set of small areas, or patches.
- The radiosity, B_i, of patch *i* is the total rate of energy leaving a surface. The radiosity over a patch is constant.





Diffuse reflection

The incident light is scattered equally in all directions

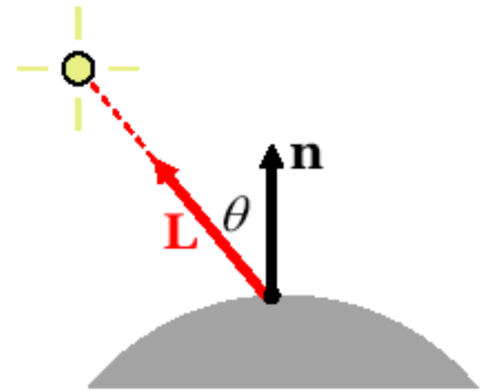


This is characteristic of dull, matt surfaces such as paper, bricks, carpet, etc.

Lambert's Law for Diffuse Reflection

Purely diffuse object

$$I = I_L k_d \cos\theta$$
$$= I_L k_d (\mathbf{n} \cdot \mathbf{L})$$



I : resulting intensity

I_L : light source intensity

k_d : (diffuse) surface reflectance coefficient

$$k_d \in [0,1]$$

θ : angle between normal & light direction

Continuous Radiosity Equation

reflectivity

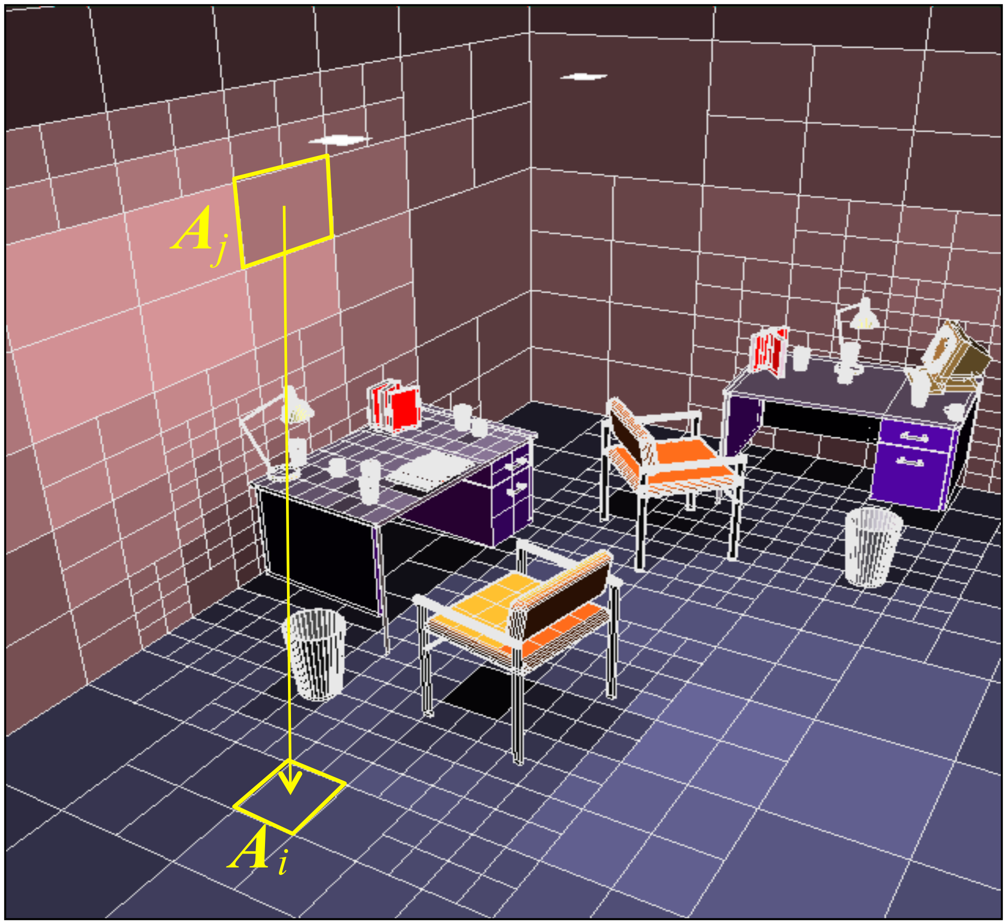
$$B_{x'} = E_{x'} + \rho_{x'} \int \underbrace{G(x,x')\ V(x,x')}_{\text{form factor}}\ B_x$$

G: geometry term
V: visibility term

No analytical solution,
even for simple configurations

Discrete Radiosity Equation

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



reflectivity

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

form factor

- discrete representation
- iterative solution
- costly geometric/visibility calculations

The Radiosity Matrix

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

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

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

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

Solve [F][B] = [E]

Direct methods: $O(n^3)$

- **Gaussian elimination**

  Goral, Torrance, Greenberg, Battaile, 1984

Iterative methods: $O(n^2)$

Energy conservation

$\rightarrow$ ***diagonally dominant*** $\rightarrow$ ***iteration converges***

- **Gauss-Seidel, Jacobi: Gathering**

  Nishita, Nakamae, 1985

  Cohen, Greenberg, 1985

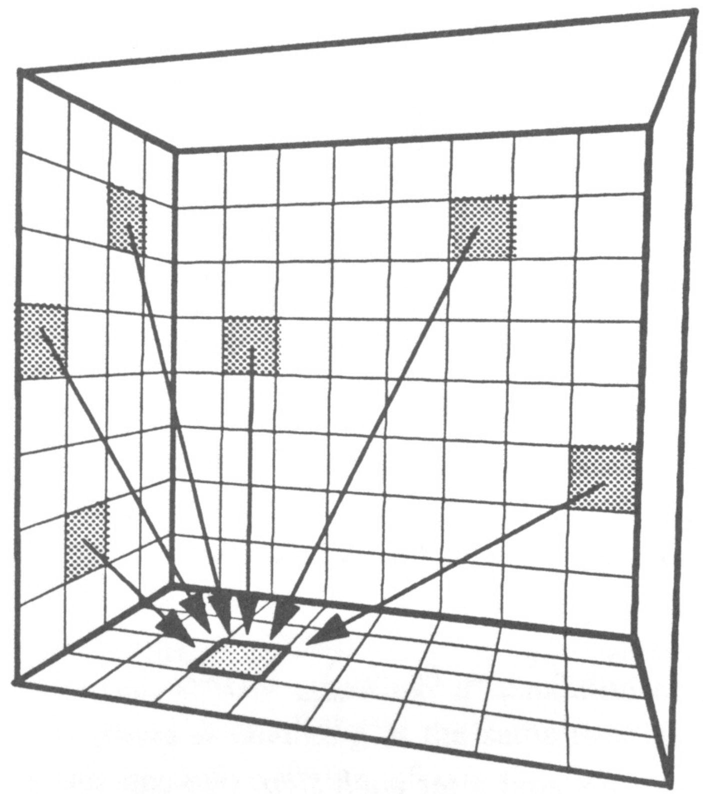
- **Southwell: Shooting**

  Cohen, Chen, Wallace, Greenberg, 1988

Solving the Radiosity Matrix

The radiosity of a single patch i is updated for each iteration by *gathering* radiosities from all other patches:

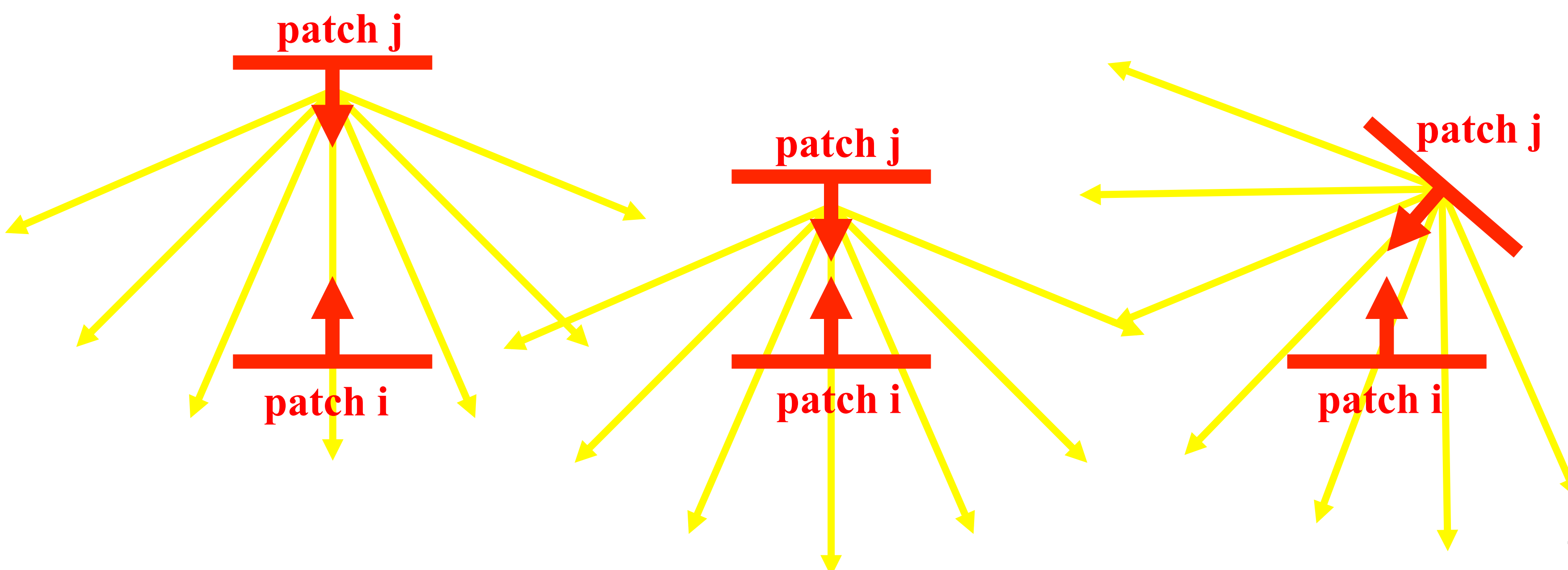
$$\begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_i \\ \vdots \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_i \\ \vdots \\ E_n \end{bmatrix} + \begin{bmatrix} & & & \\ & & & \\ & & & \\ \rho_i F_{i1} & \rho_i F_{i2} & \cdots & \rho_i F_{in} \\ & & & \\ & & & \end{bmatrix} \begin{bmatrix}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



This method is fundamentally a Gauss-Seidel relaxation

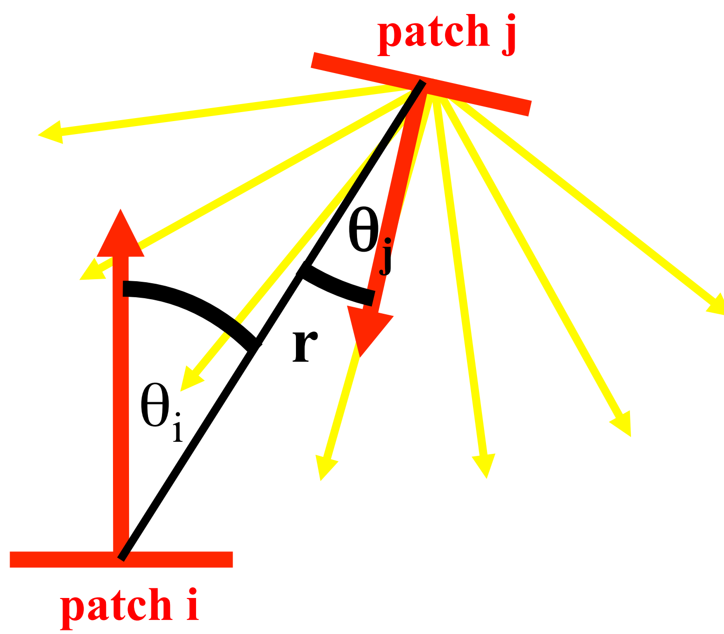
Calculating the Form Factor F_{ij}

- 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}

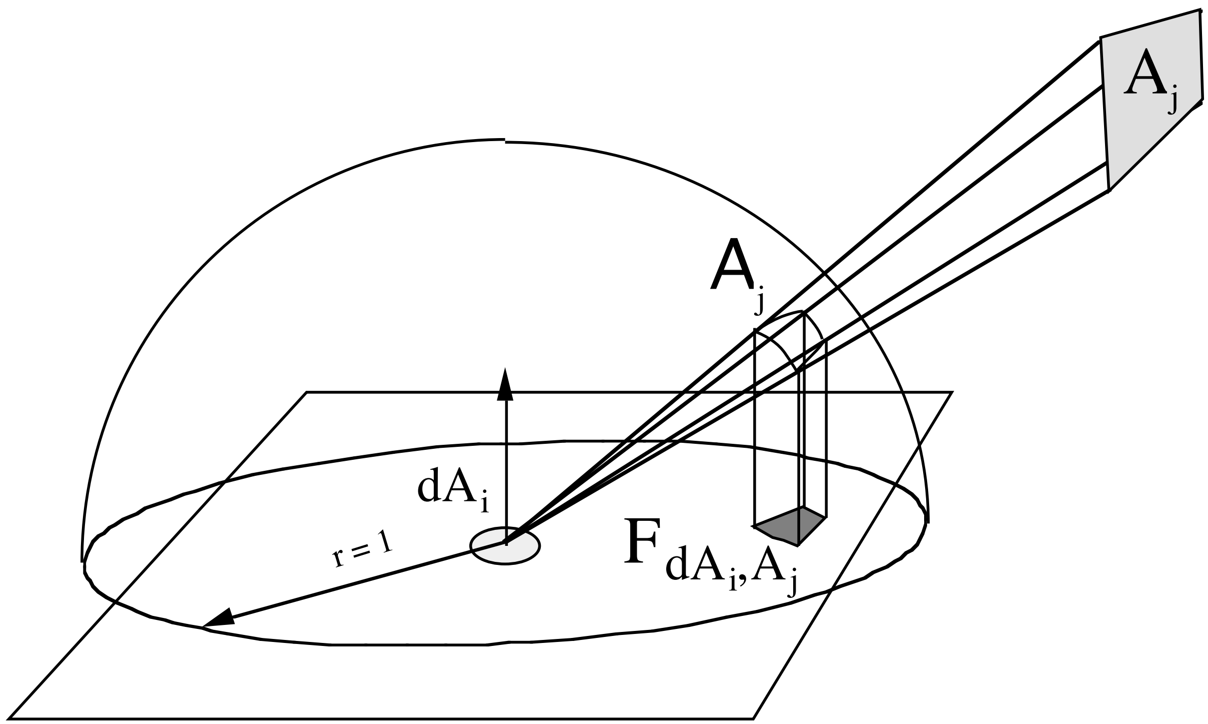
- F_{ij} = fraction of light energy leaving patch j that arrives at patch i



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

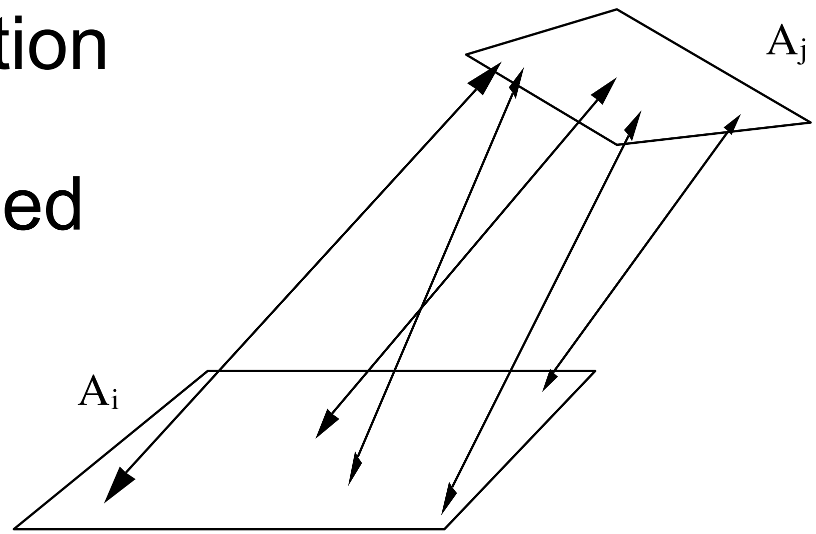
Form Factor Determination

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



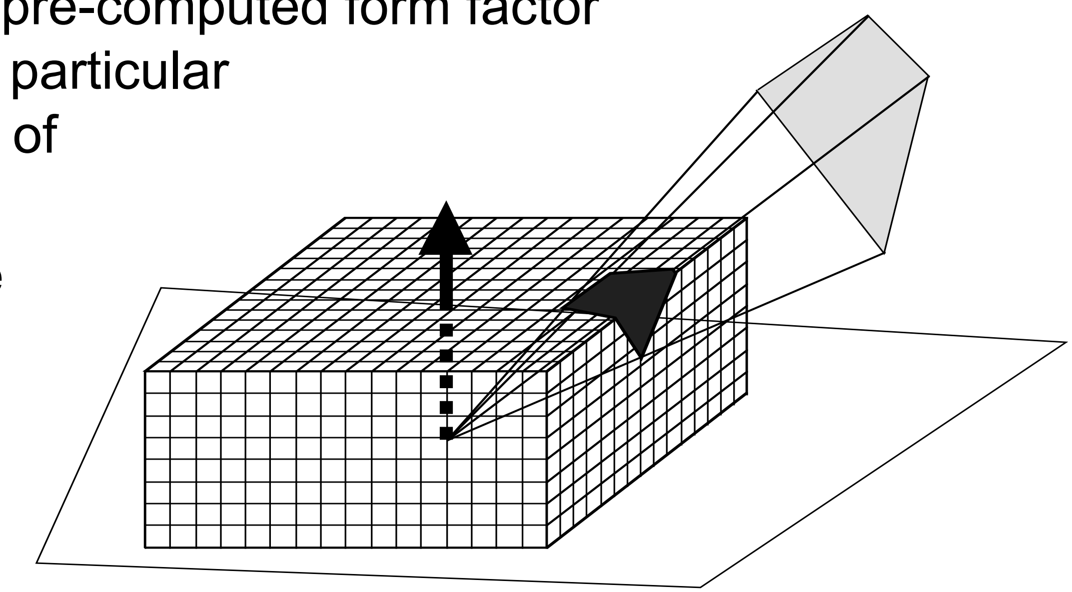
Form Factor from Ray Casting

- Cast *n* rays between the two patches
  - *n* is typically between 4 and 32
  - Compute visibility
  - Integrate the point-to-point form factor
- Permits the computation of the patch-to-patch form factor, as opposed to point-to-patch



Hemicube Algorithm

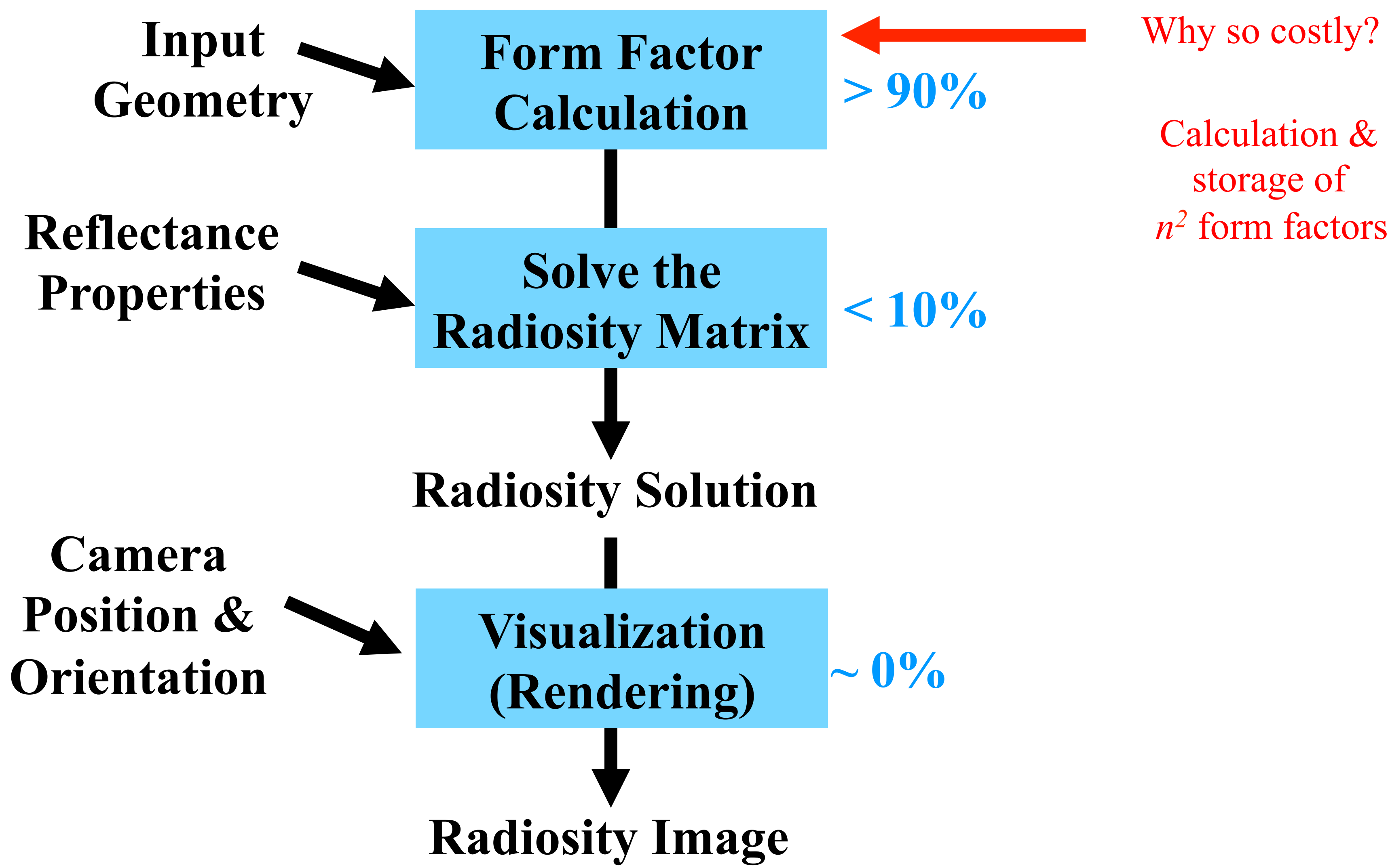
- A hemicube is constructed around the center of each patch
- Faces of the hemicube are divided into "pixels"
- Each patch is projected (rasterized) onto the faces of the hemicube
- Each pixel stores its pre-computed form factor The form factor for a particular patch is just the sum of the pixels it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



Hemicube Algorithm

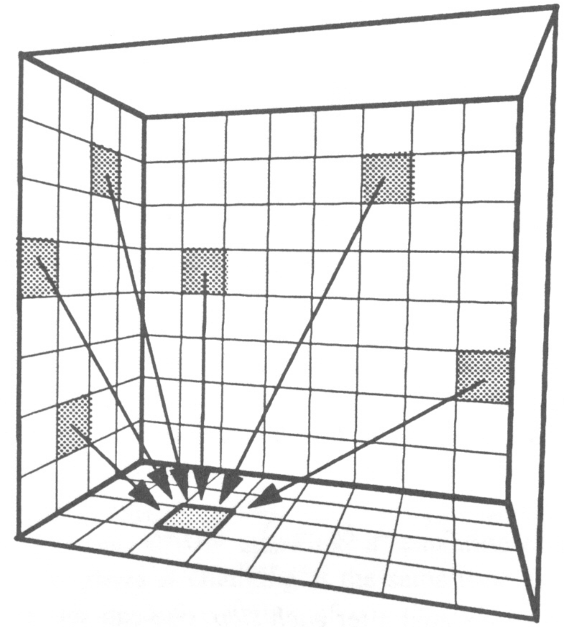
- Advantages
  - First practical method -> Patent!
  - Uses existing rendering systems; hardware
  - Computes row of form factors in O(n)
- Disadvantages
  - Aliasing errors due to sampling
  - Proximity errors
  - Visibility errors
  - Expensive to compute a single form factor

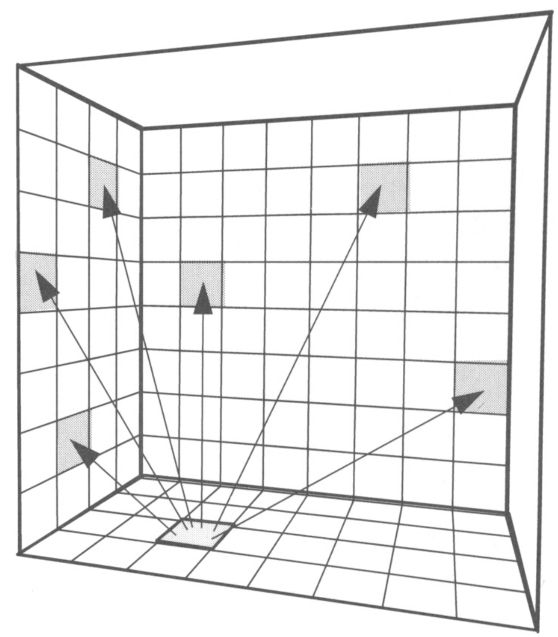
Stages in a Radiosity Solution



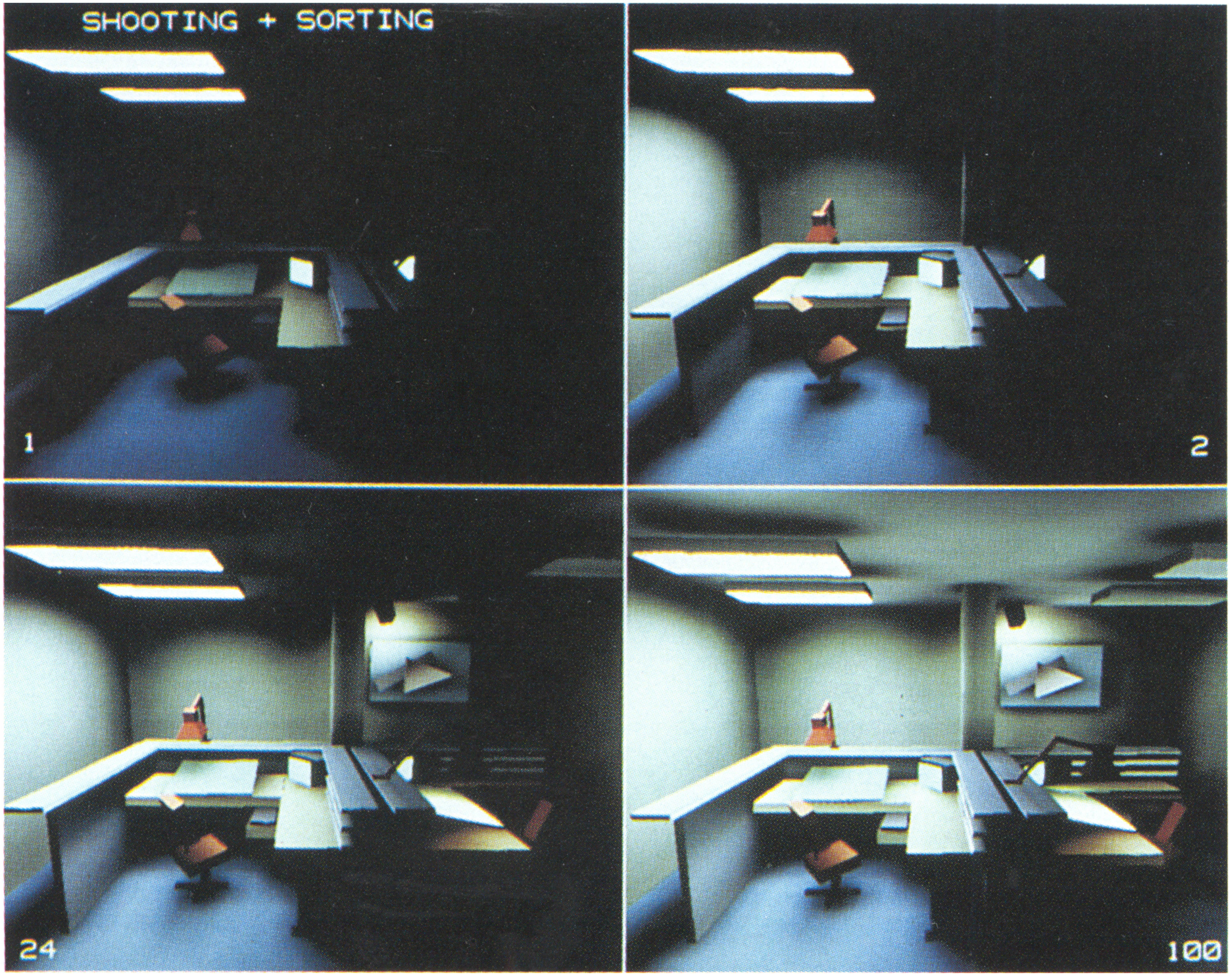
Progressive Refinement

- 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*.

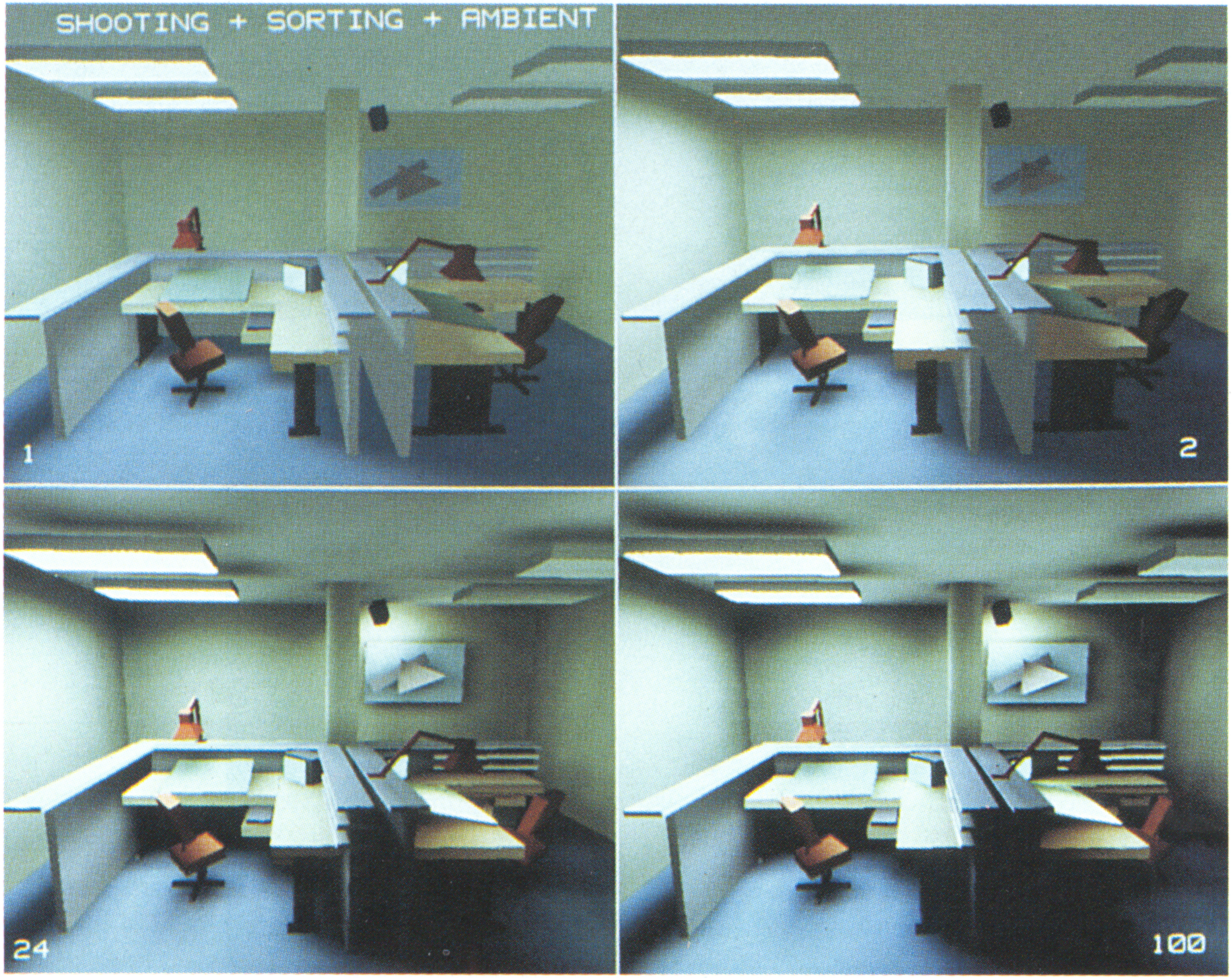




Progressive Refinement w/out Ambient Term



Progressive Refinement with Ambient Term

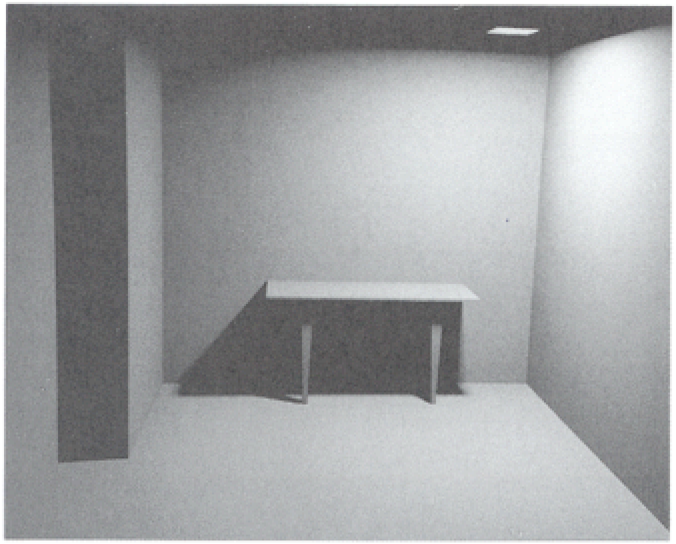


Results

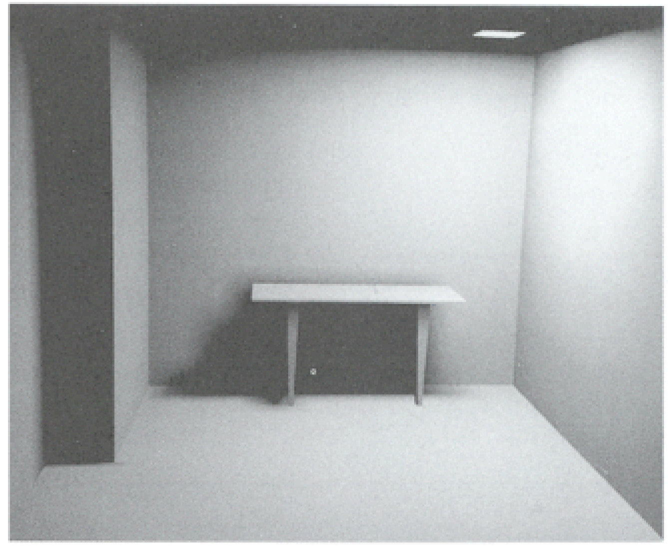
Factory simulation. Program of Computer Graphics, Cornell University.
30,000 patches.

Meshing for Radiosity

Accuracy



Reference Solution



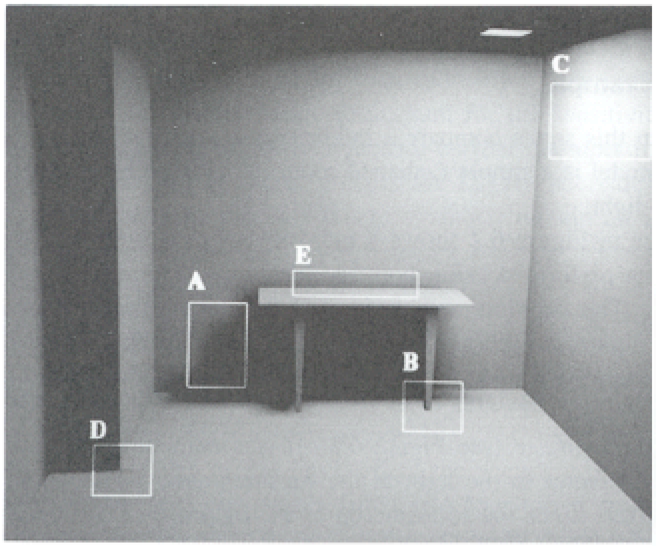
Uniform Mesh

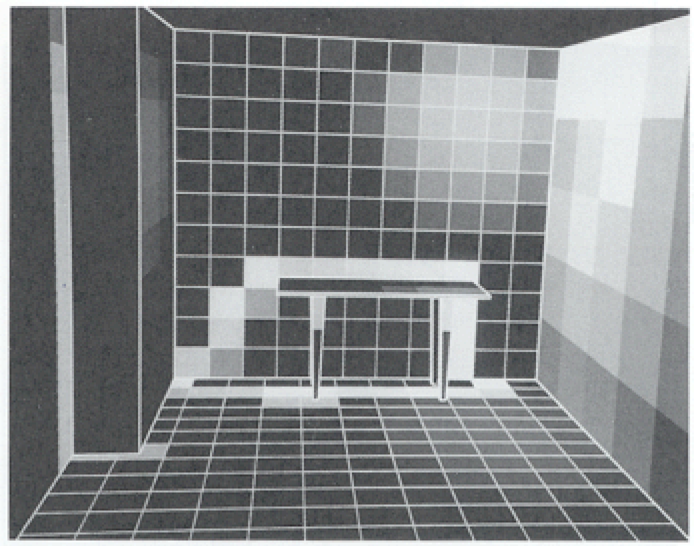
Table in room sequence from Cohen and Wallace

CS348B Lecture 17 Pat Hanrahan, Spring 2002

Slide 24 of 32

Artifacts





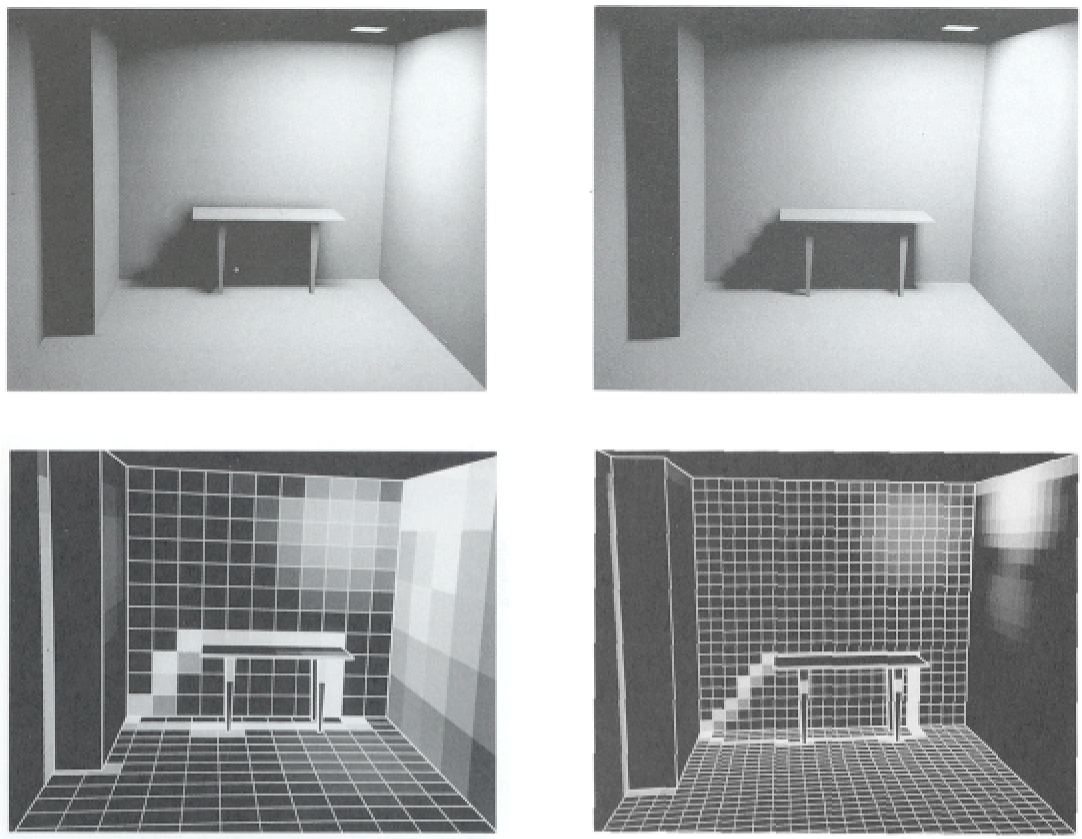
Error Image

A. Blocky shadows
B. Missing features
C. Mach bands
D. Inappropriate shading discontinuities
E. Unresolved discontinuities

CS348B Lecture 17 Pat Hanrahan, Spring 2002

Slide 25 of 32

Increasing Resolution

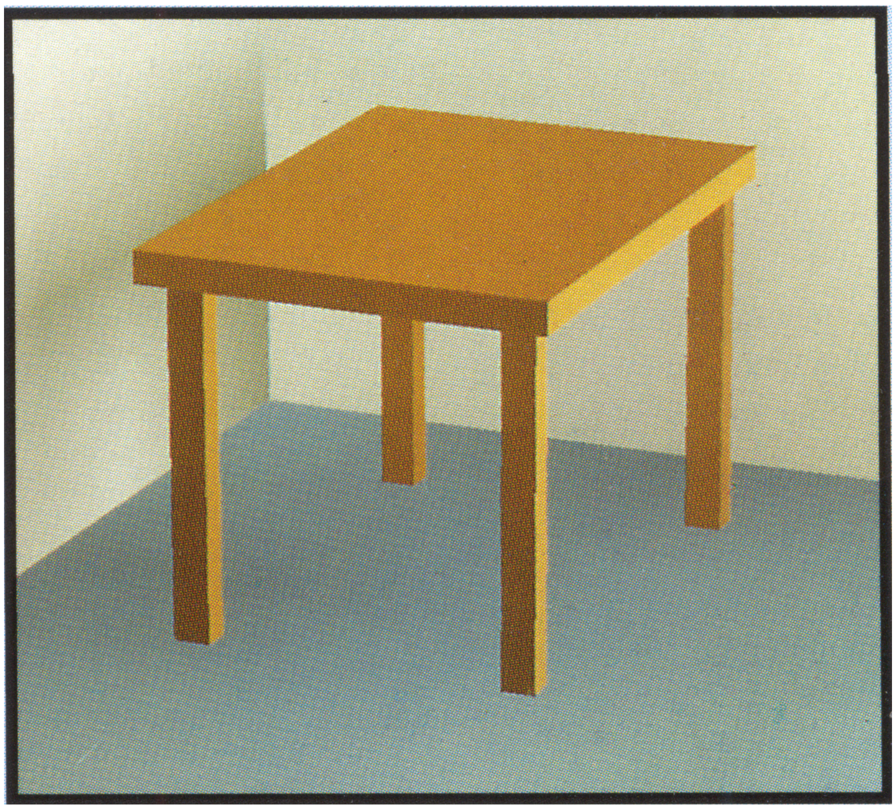


CS348B Lecture 17 Pat Hanrahan, Spring 2002

Slide 26 of 32

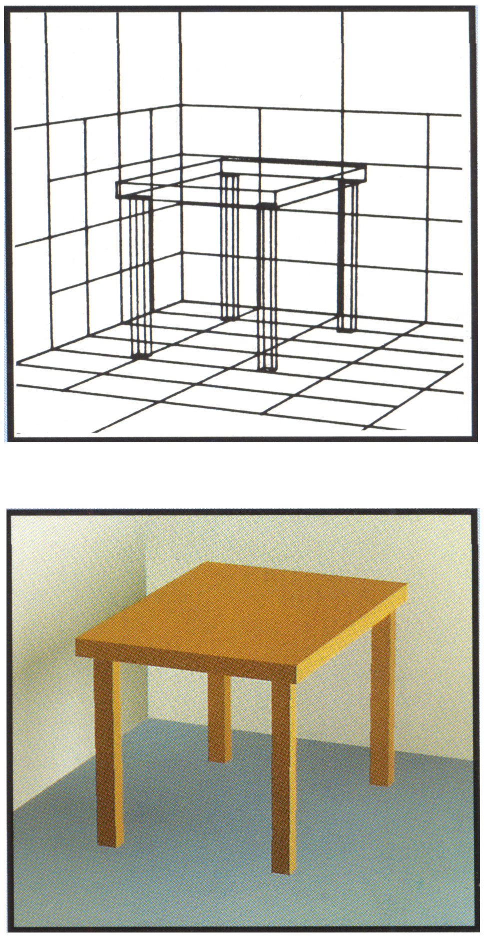
Increasing the Accuracy of the Solution

What's wrong with this picture?

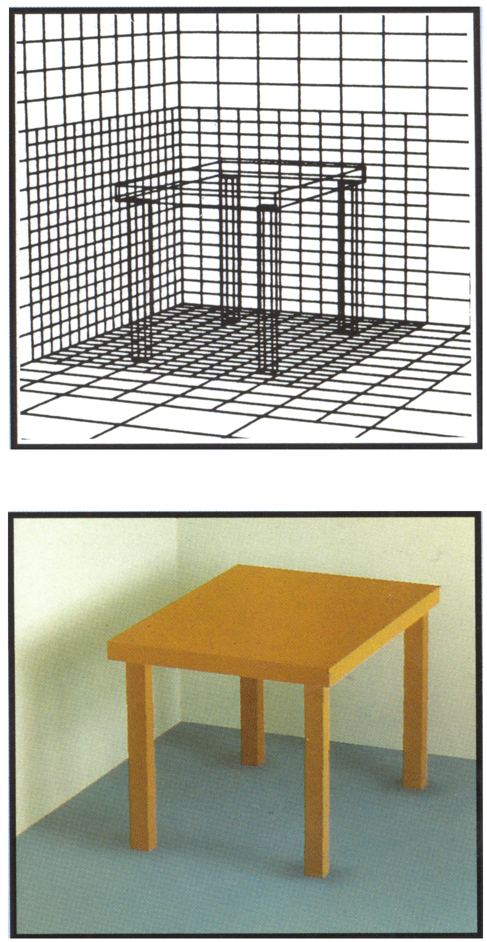


- The quality of the image is a function of the size of the patches.
- The patches should be *adaptively subdivided* near shadow boundaries, and other areas with a high radiosity gradient.
- Compute a solution on a uniform initial mesh, then refine the mesh in areas that exceed some error tolerance.

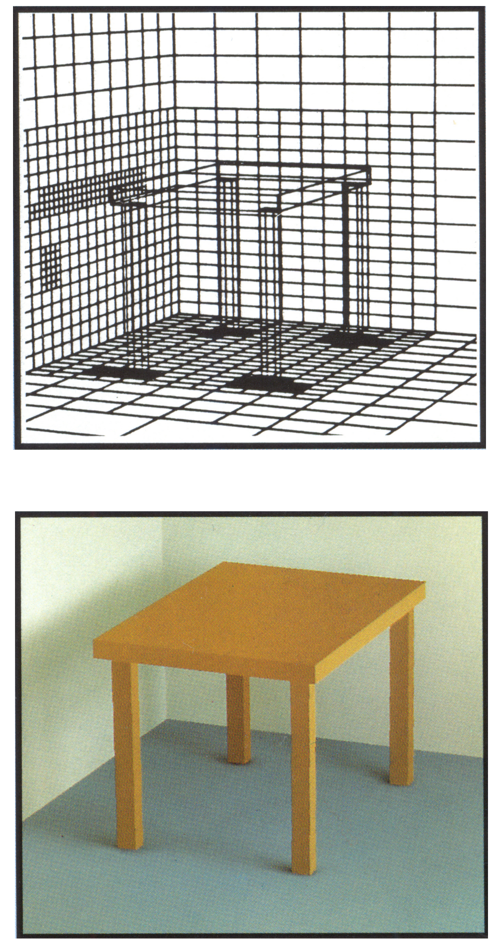
Adaptive Subdivision of Patches



Coarse patch solution
(145 patches)

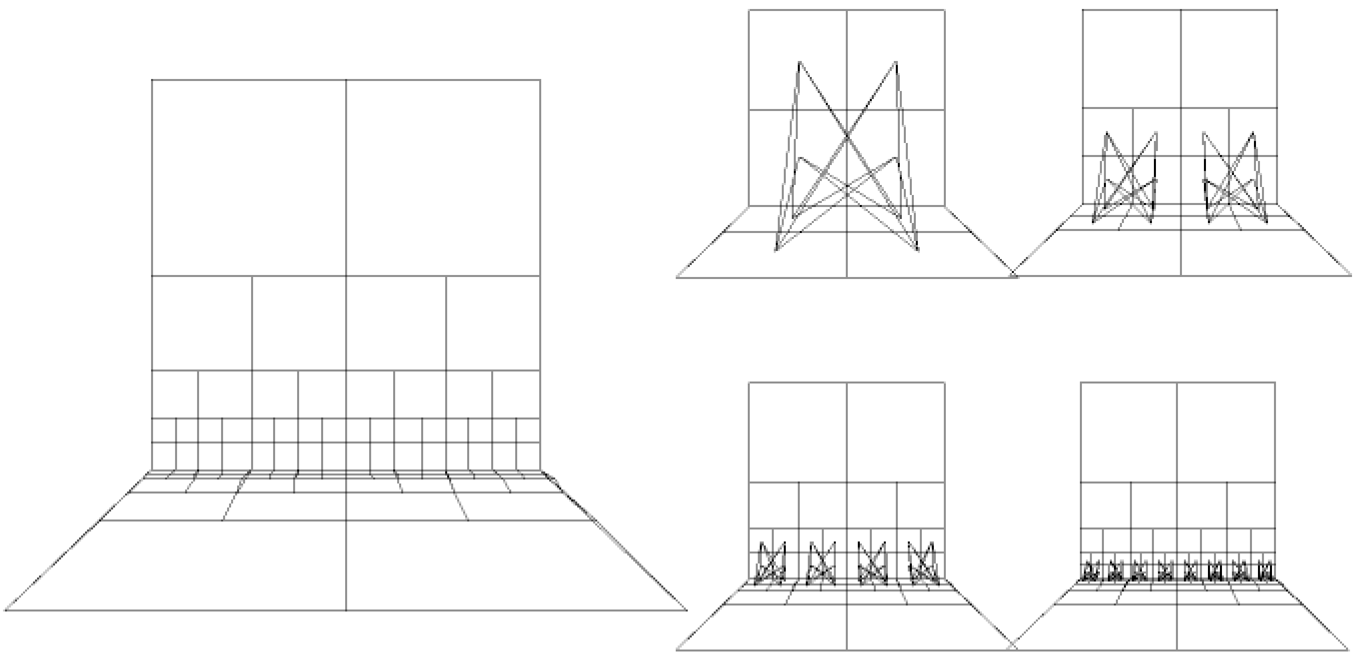


Improved solution
(1021 subpatches)



Adaptive subdivision
(1306 subpatches)

Hierarchical Radiosity

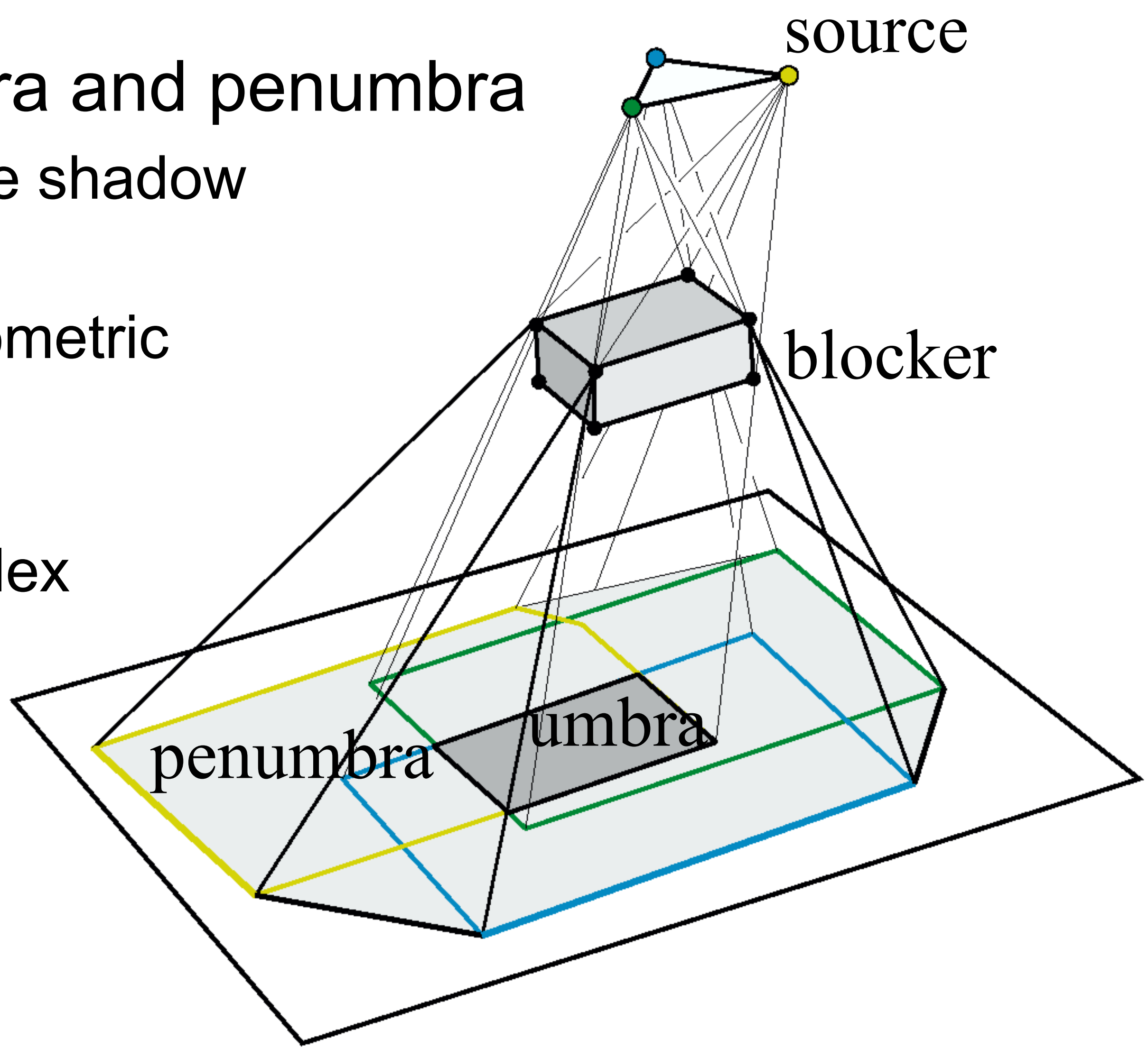


CS348B Lecture 17 Pat Hanrahan, Spring 2002

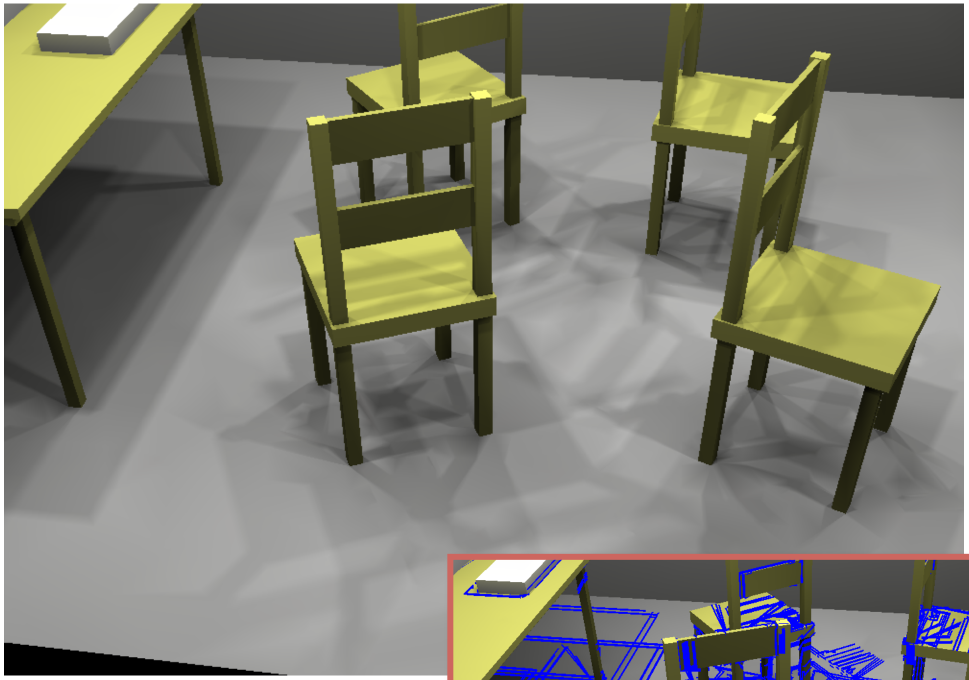
Slide 31 of 32

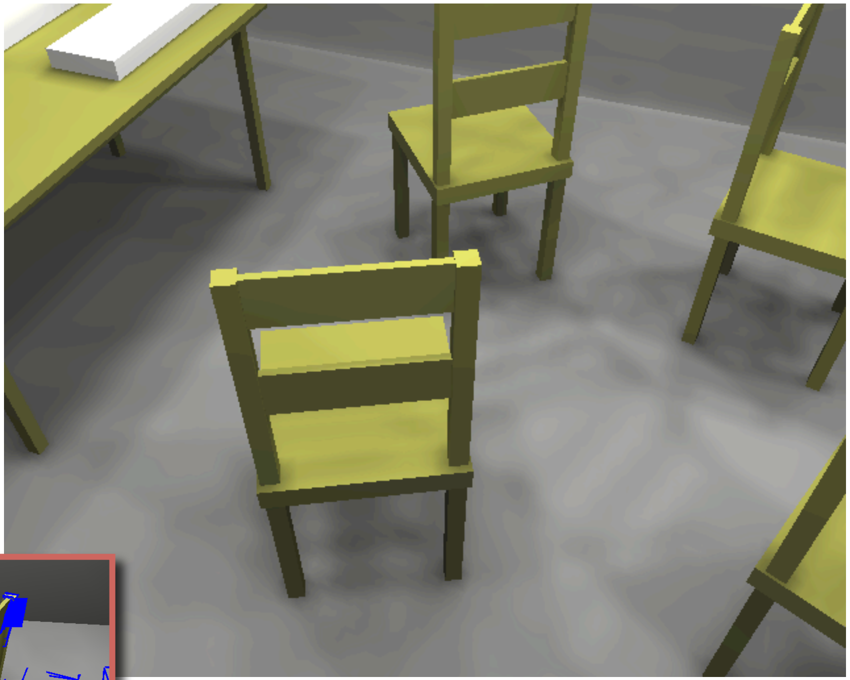
Discontinuity Meshing

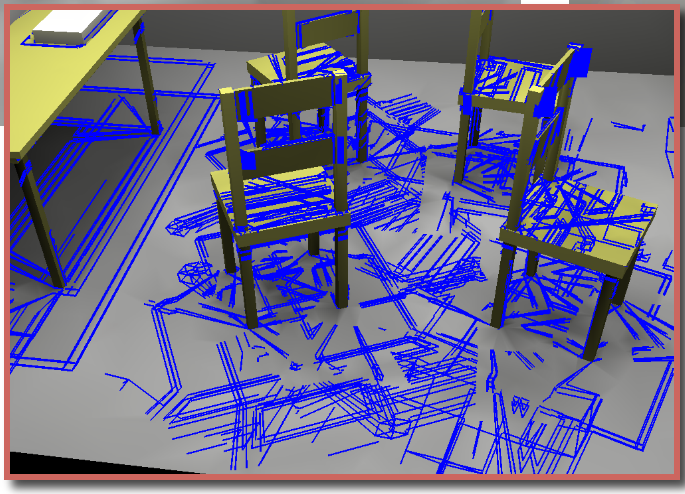
- Limits of umbra and penumbra
  - Captures nice shadow boundaries
  - Complex geometric computation
  - The mesh is getting complex



Discontinuity Meshing Comparison







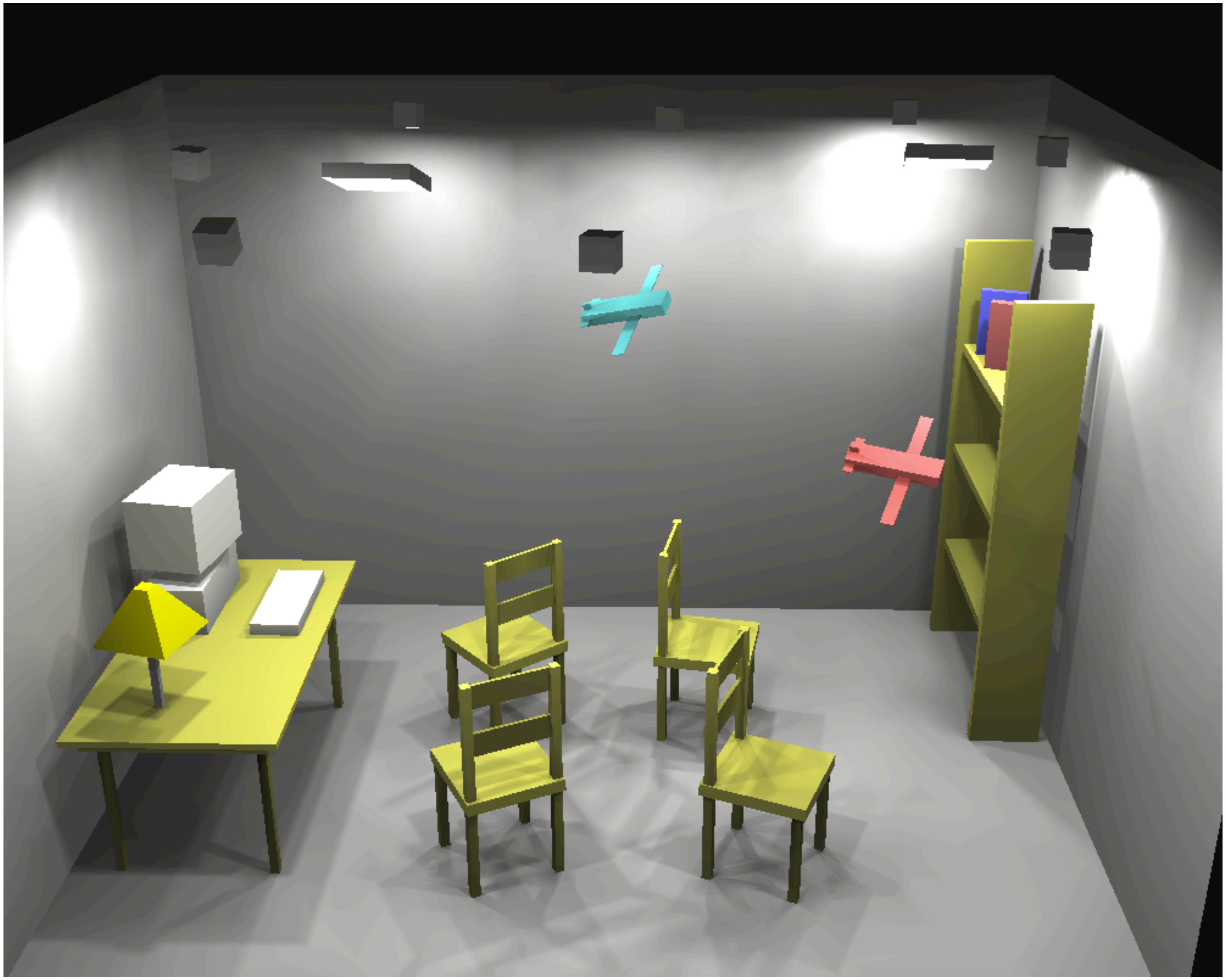
With visibility skeleton & discontinuity meshing

10 minutes 23 seconds

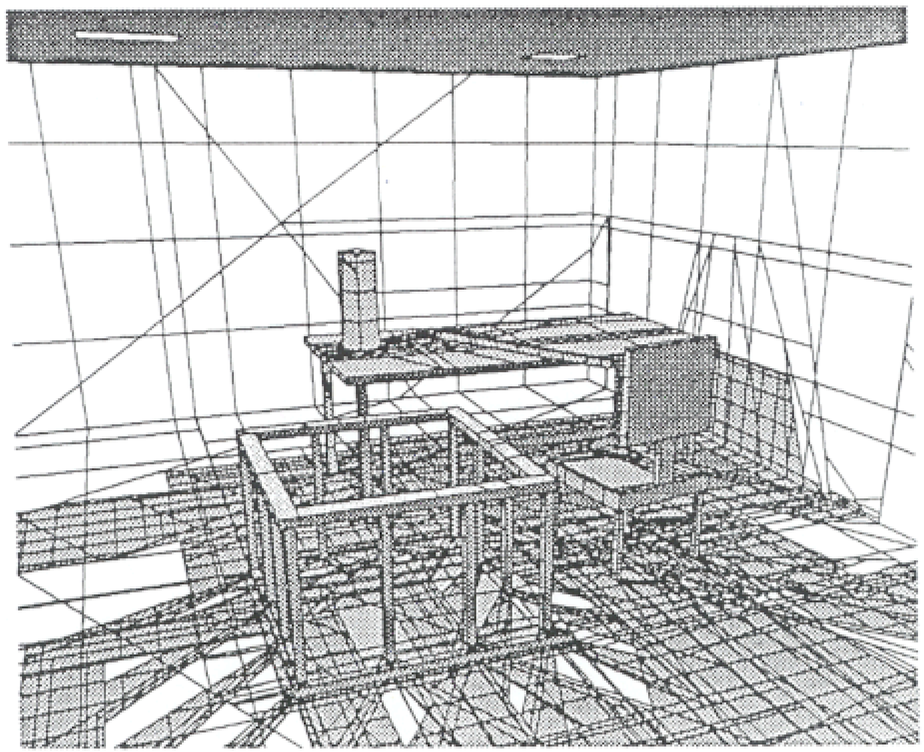
[Gibson 96]

1 hour 57 minutes

Discontinuity Meshing



Discontinuity Mesh



From Campbell et al.

CS348B Lecture 17 Pat Hanrahan, Spring 2002

Slide 29 of 32

Discontinuity Meshing

From Lischinski, Tampieri, Greenberg 1992

CS348B Lecture 17 Pat Hanrahan, Spring 2002

Slide 30 of 32

Results

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

Results

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

Results

Lightscape http://www.lightscape.com