Global Illumination Methods

CS535

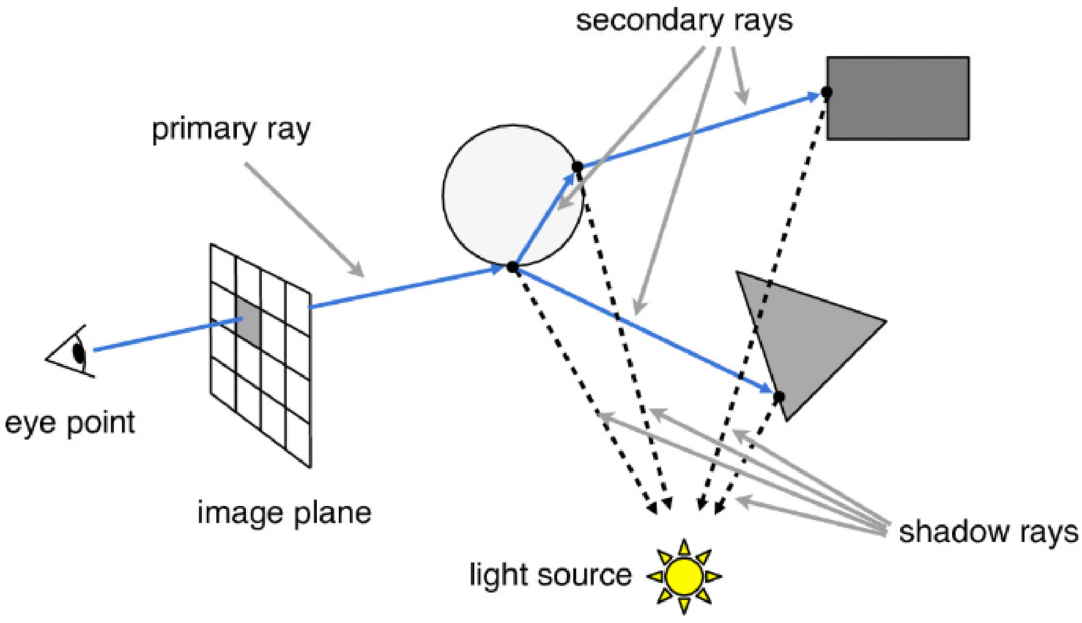
Daniel G. Aliaga
Department of Computer Science
Purdue University

Methodologies

- Ray Tracing
- Path Tracing
  - Single and Bidirectional
- Photon Mapping
- Radiosity

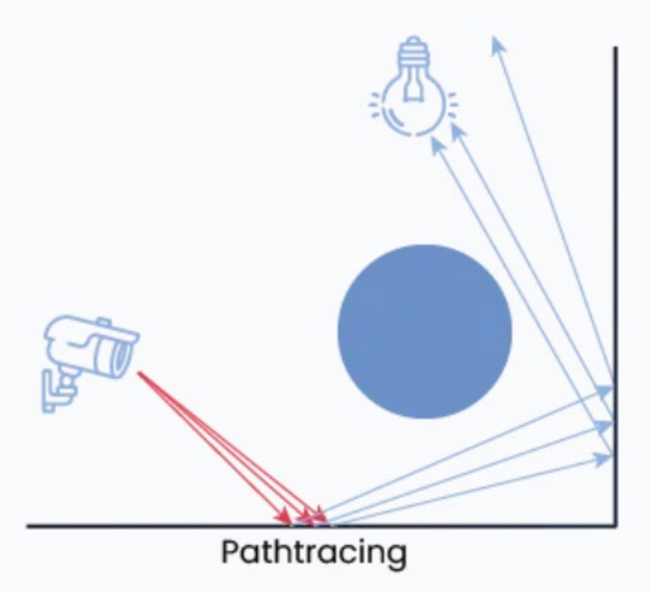
Ray Tracing

- From eye, shoot ray into scene, reflect/refract ray, shoot ray(s) to light, repeat
- *"From eye, a <u>forest</u> of rays is traced…"*



(Single Directional) Path Tracing

- From eye, shoot ray into scene, reflect/refract a single ray, repeat
- *"From eye, a <u>tree</u> of rays is traced…"*

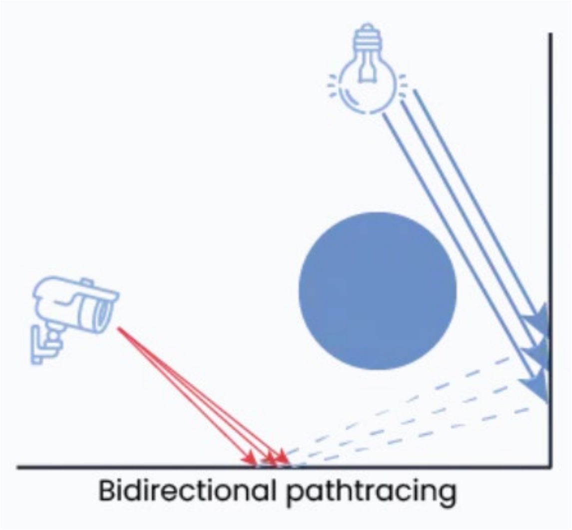


[Eclat Digital]

(Bidirectional) Path Tracing

- Path trace from eye and from light
  - At each eye path reflection/refraction point, connect to light path reflection/refraction point and determine occlusion, repeat
- *“From eye and light, a <u>tree</u> of rays is traced…”*

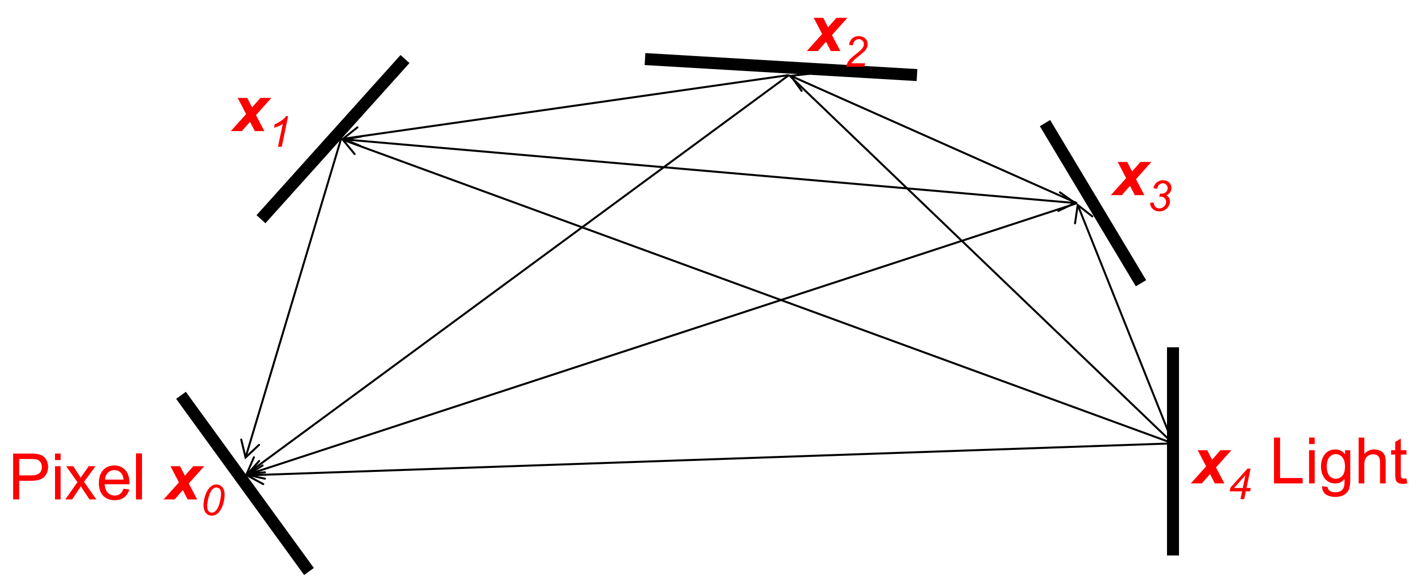
KEY: The likelihood of both paths meeting is low but that is why at each (or some) reflection/refraction points, they are attempted to be joined



[Eclat Digital]

(Bidirectional) Path Tracing

- Build a path by working from the eye and the light and join in the middle
- Don't just look at overall path, also weigh contributions from all sub-paths:



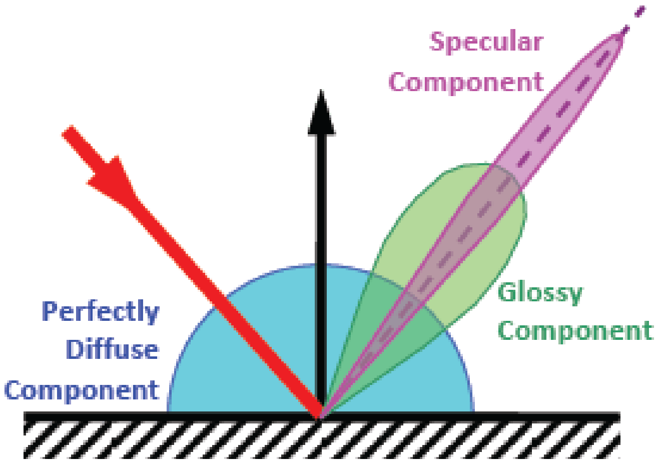
Metropolis Sampling (applied to path tracing)

- aka Metropolis Light Transport
- Metropolis algorithms generate a sequence of paths where each path can depend on the previous one
- For each sample:
  - Propose a new candidate depending on the previous sample
  - Choose to accept or reject according to a computed probability (if reject, re-use the old sample)

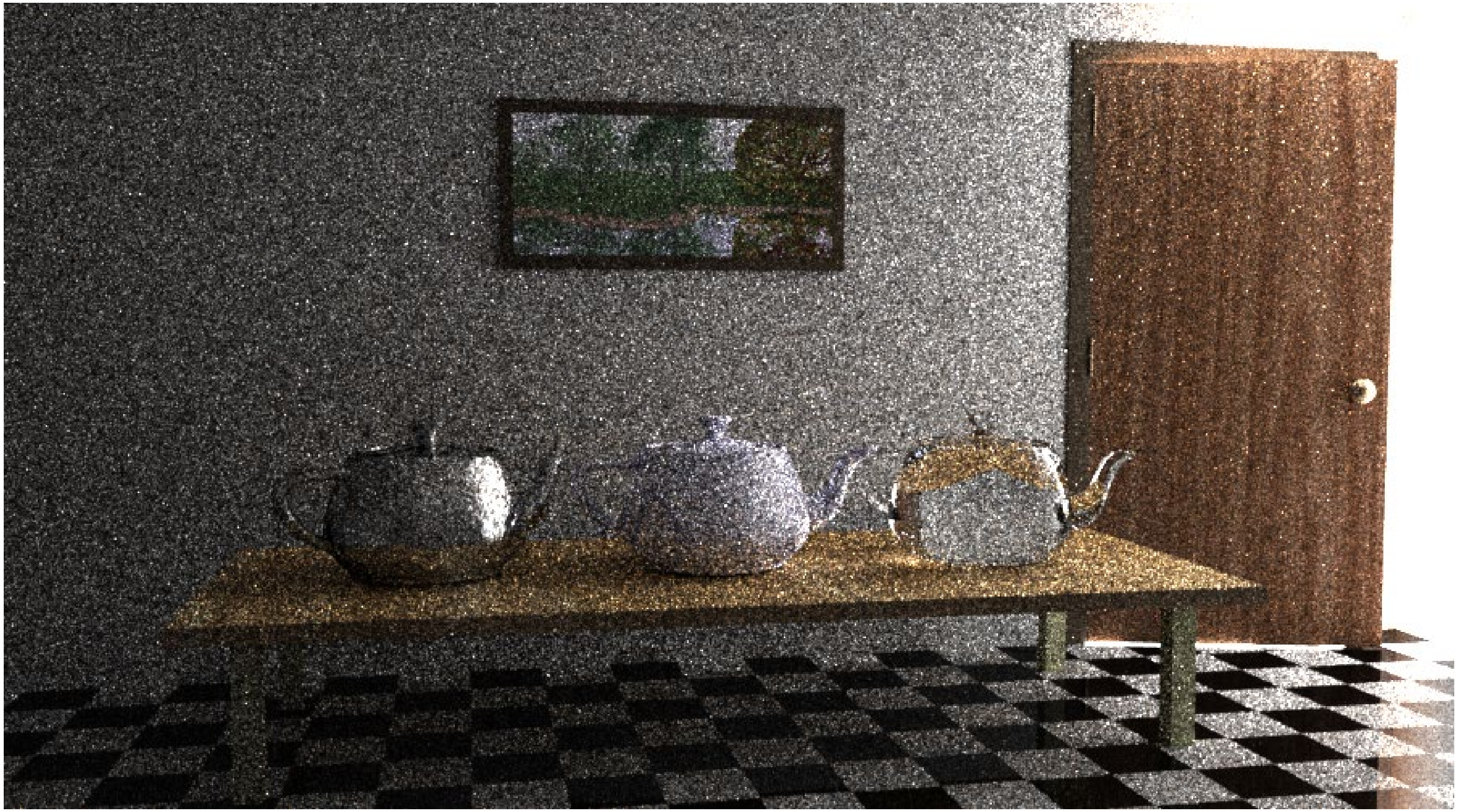
Metropolis Sampling (applied to path tracing)

- Recall BRDFs
  - Direct illumination BRDF models (diffuse, specular, Cook-Torrance, etc) or BRDFs from sampled material (e.g., use 10-15 coefficients and nonlinear basis functions)

- Use as probability distribution for path/ray sampling



Bidirectional Path Tracing



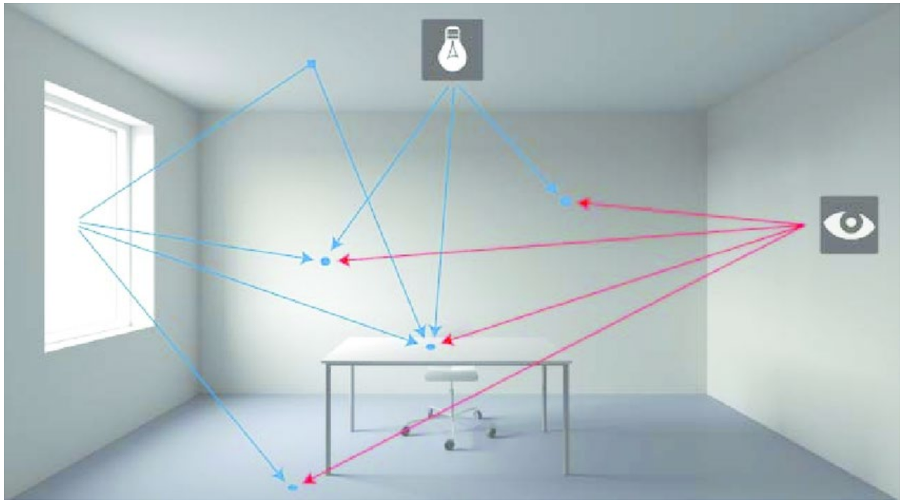
Work = “X”

Metropolis Light Transport

Work = “X”
(same work but smarter sampling)

Photon Mapping

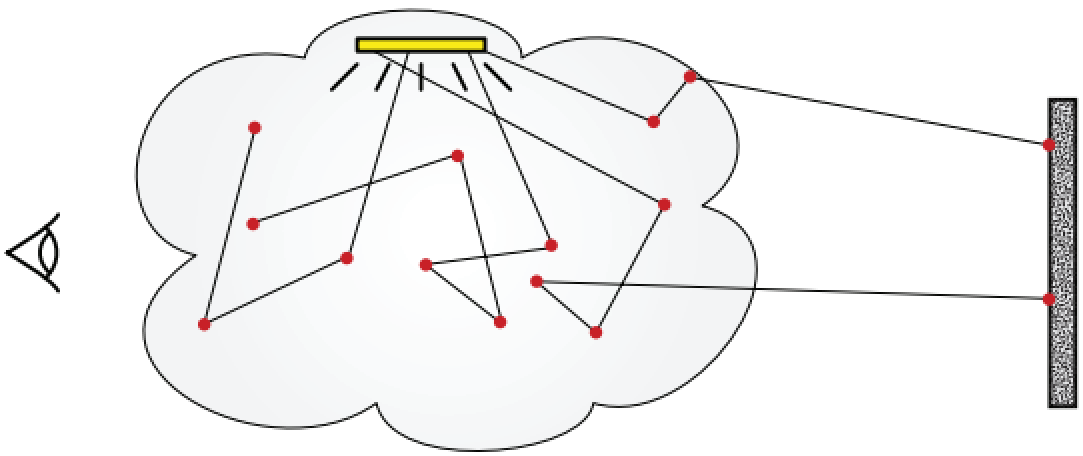
- Push light out from sources (instead of gathering it toward the eye)
  - Trace light rays
  - Trace eye rays and look for closest points having light and do an interpolation
  - Repeat…



[Jens Christoffersen]

Photon Mapping

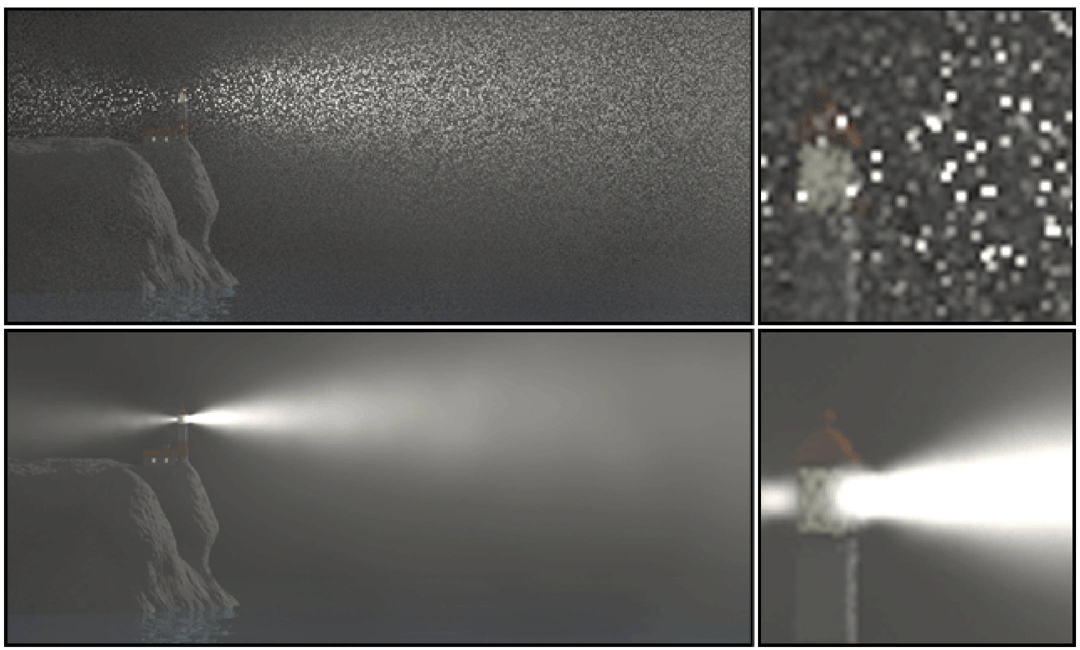
- Supports participating media
  - Light is stored not only on surfaces but within the medium as well



[Wojciech Jarosz]

Photon Mapping

- Supports participating media
  - Light is stored not only on surfaces but within the medium as well



[Jarosz et al.]

Radiosity

- Discretization approach to compute light transport, specific to diffuse scenes

- The main idea of the method is

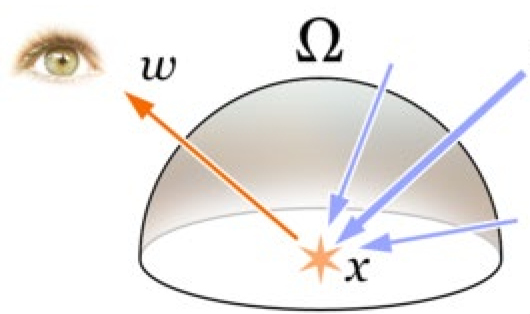
  to store illumination values on the surfaces of the objects, as the light is propagated starting at the light sources.

Radiosity

- Radiosity is inspired by ideas from heat transfer and is an application of a finite element method to solving the rendering equation for scenes with purely diffuse surfaces

$$L_o(\mathbf{x}, \omega, \lambda, t) = L_e(\mathbf{x}, \omega, \lambda, t) + \int_\Omega f_r(\mathbf{x}, \omega', \omega, \lambda, t) L_i(\mathbf{x}, \omega', \lambda, t)(-\omega' \cdot \mathbf{n}) d\omega'$$

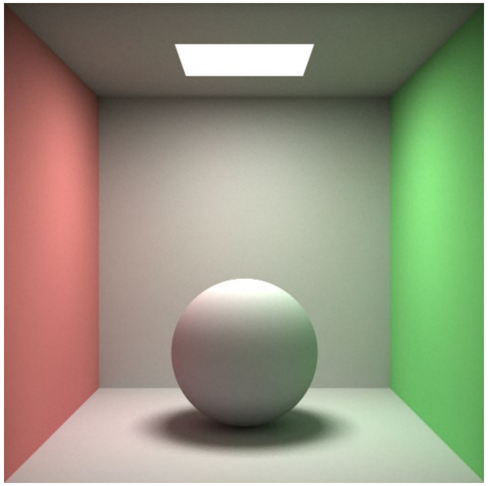
(rendering equation)



[Radiosity slides heavily based on Dr. Mario Costa Sousa, Dept. of of CS, U. Of Calgary]

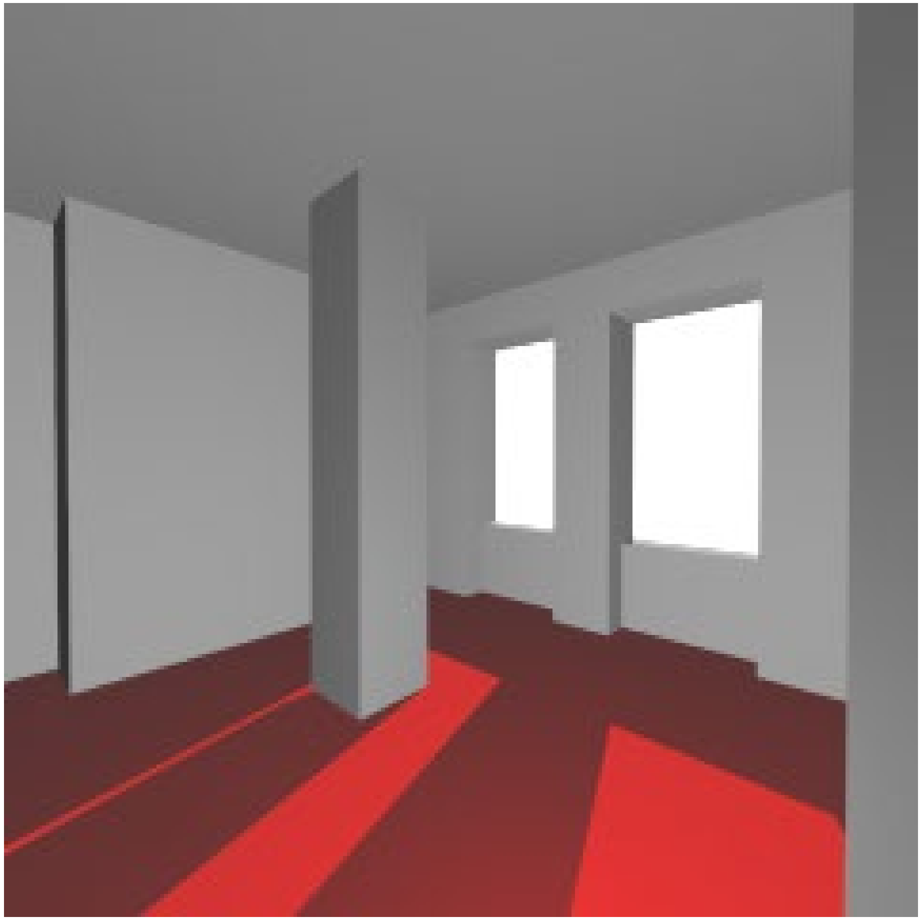
Radiosity

- Equation: $B_i \, dA_i = E_i \, dA_i + R_i \int_j B_j F_{ji} \, dA_j$




Ray Tracin


Radiosit



Radiosity Assumptions

- #1: surfaces are diffuse emitters and reflectors of energy, emitting and reflecting energy uniformly over their entire area.

- #2: an equilibrium solution can be reached; that all of the energy in an environment is accounted for, through absorption and reflection.

- #3: solution can-be/will-be viewpoint independent; i.e., the solution will be the same regardless of the viewpoint of the image.

The Radiosity Equation

- The "radiosity equation" describes the **amount of energy** which can be emitted from a surface
  - Is a sum of the energy inherent in the surface (e.g., a light source) plus energy which strikes the surface (e.g., from another surface)
- The energy which leaves a surface (surface "j") and strikes another surface (surface "i") is attenuated by two factors:
  - the **"form factor"** between surfaces "i" and "j“ (physical relationship)
  - the **reflectivity of surface "i“** (material property)

The Radiosity Equation

$$B_i = E_i + \rho_i \sum B_j F_{ij}$$

Radiosity of surface i

Emissivity of surface i

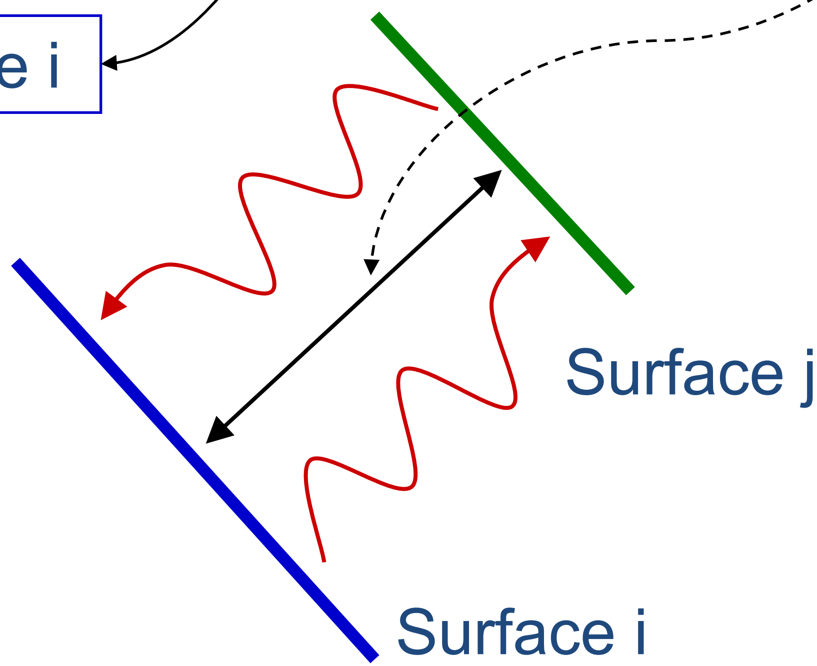
Reflectivity of surface i

will absorb a certain percentage of light energy which strikes the surface

Form Factor of surface j relative to surface i

accounts for the physical relationship between the two surfaces

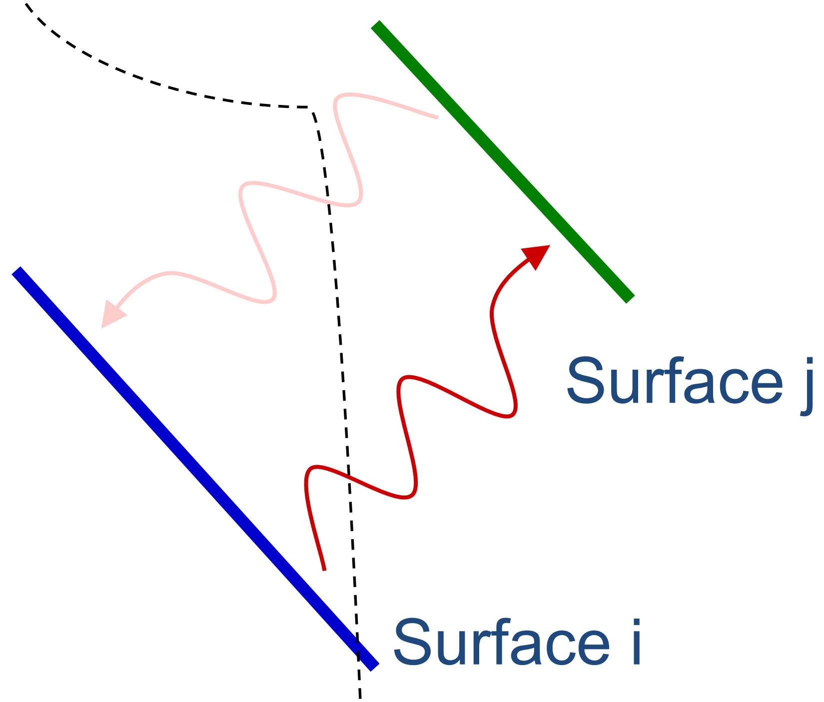
Radiosity of surface j



The Radiosity Equation

$$B_i = \boxed{E_i} + \rho_i \sum B_j F_{ij}$$

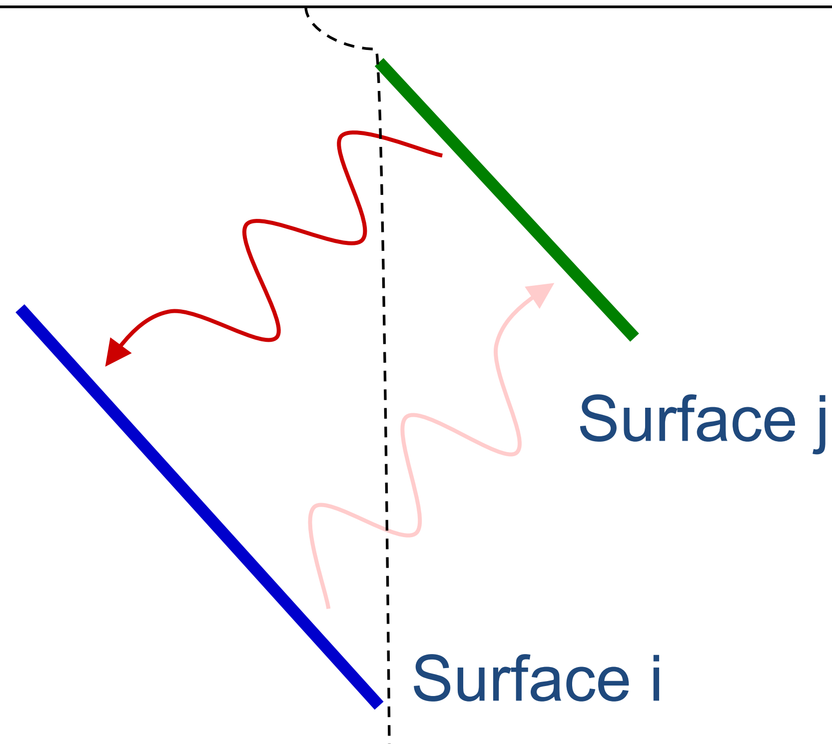
Energy emitted by surface i



The Radiosity Equation

$$B_i = E_i + \rho_i \sum B_j F_{ij}$$

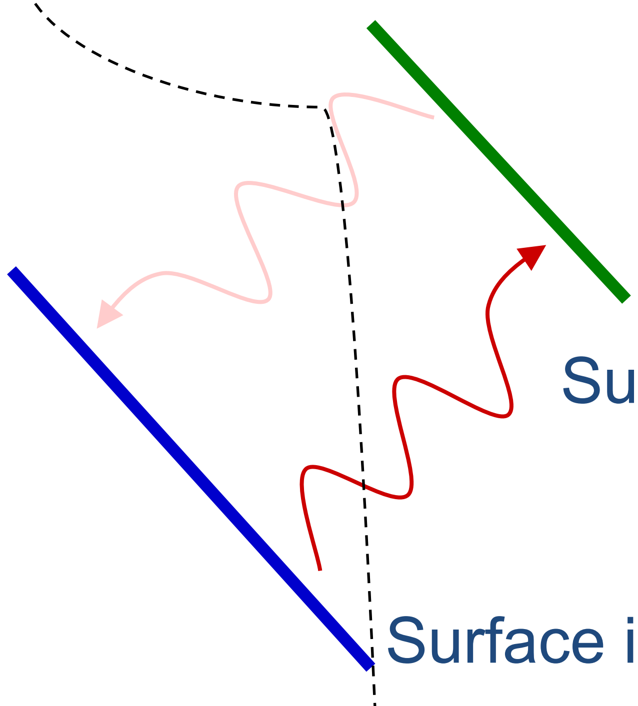
Energy reaching surface i from other surfaces



The Radiosity Equation

$$B_i = E_i + \rho_i \sum B_j F_{ij}$$

Energy reflected by surface i

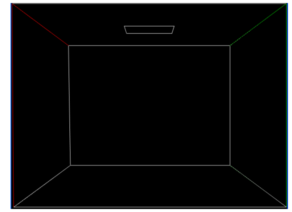
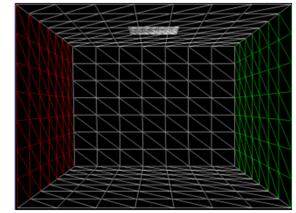


Surface j

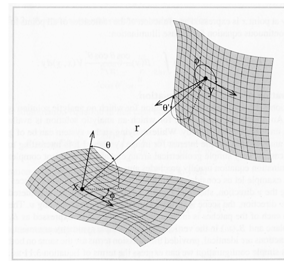
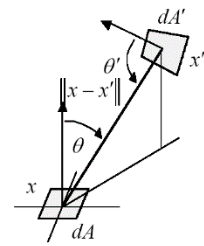
Surface i

Classic Radiosity Algorithm

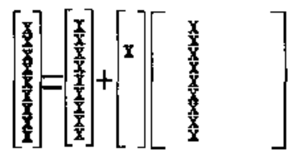
Mesh Surfaces into Elements

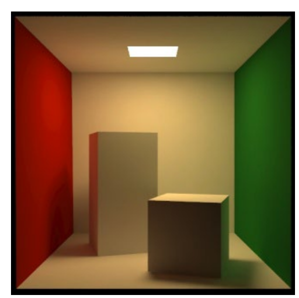
Compute Form Factors Between Elements

Solve Linear System for Radiosities

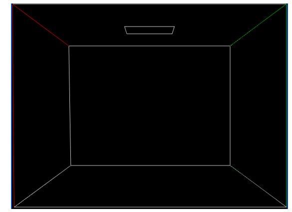


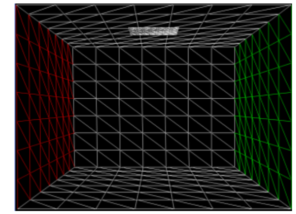
Reconstruct and Display Solution



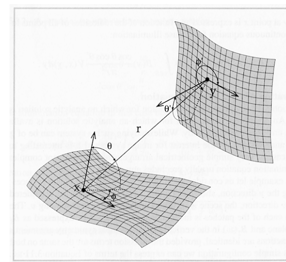
Classic Radiosity Algorithm

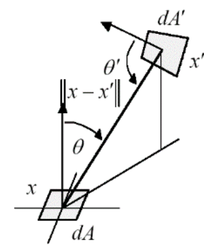
Mesh Surfaces into Elements



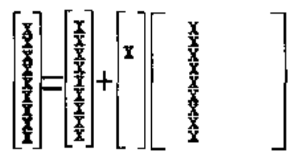


Compute Form Factors Between Elements

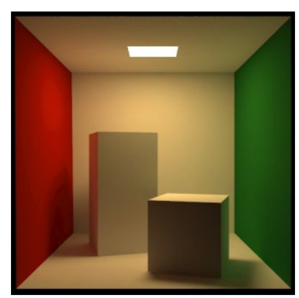




Solve Linear System for Radiosities



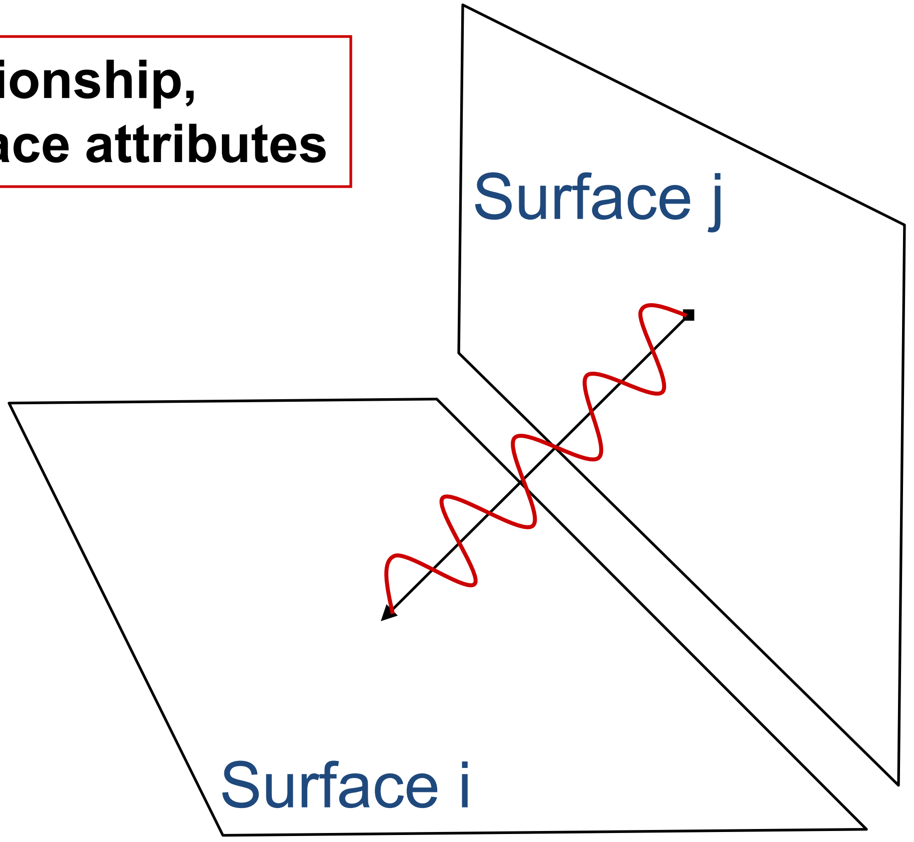
Reconstruct and Display Solution



The Form Factor:

The <u>fraction</u> of energy leaving one surface that reaches another surface

It is a purely geometric relationship, independent of viewpoint or surface attributes



Between differential areas, the form factor equals:

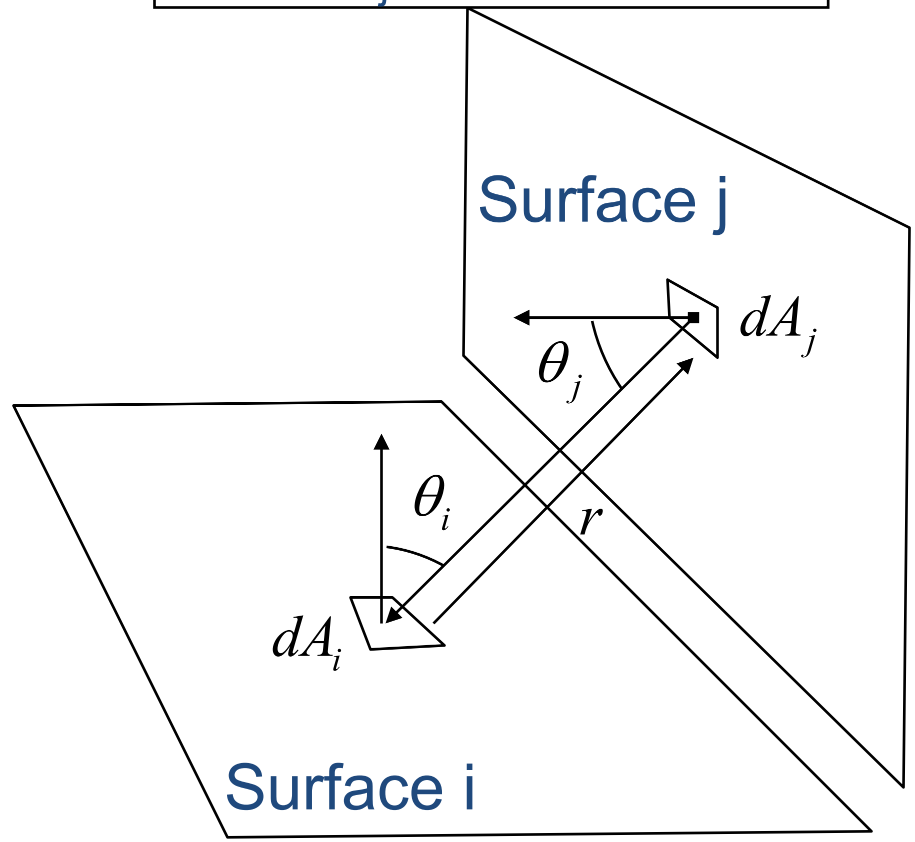
differential area of surface i, j

angle between $Normal_i$ and r

angle between $Normal_j$ and r

$$F_{dA_i dA_j} = \frac{\cos\theta_i \cos\theta_j}{\pi |r|^2}$$

vector from dA_i to dA_j

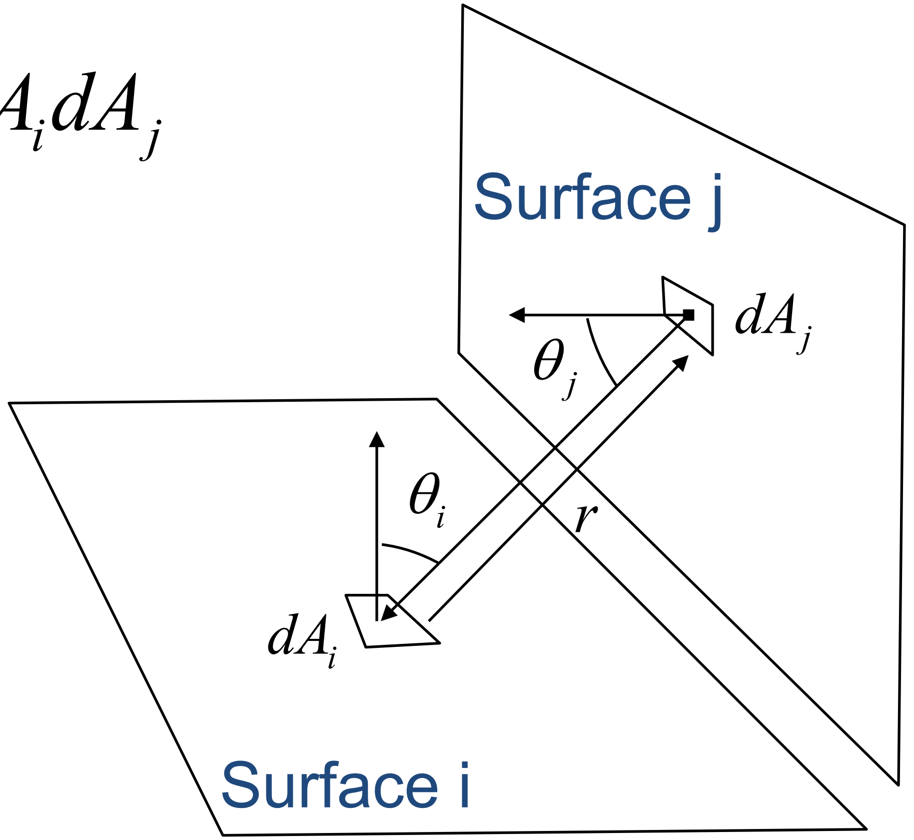


Between <u>differential areas</u>, the form factor equals:

$$F dA_i dA_j = \frac{\cos\theta_i \cos\theta_j}{\pi |r|^2}$$

The overall form factor between i and j is found by integrating

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



Form Factors in (More) Detail

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

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

where V_{ij} is the visibility (0 or 1)

Form Factors in (More) Detail

- Several ways to find form factors

- Hemicube was original method
  + Hardware acceleration
  + Gives F_{dAiAj} for all j in one pass
  - Aliasing

- Area sampling methods now preferred
  → Slower than hemicube but GPU-able
  → As accurate as desired since adaptive

Area Sampling

Subdivide A_j into small pieces dA_j

For all dA_j

cast ray *dAj-dAj* to determine V_{ij}

if visible

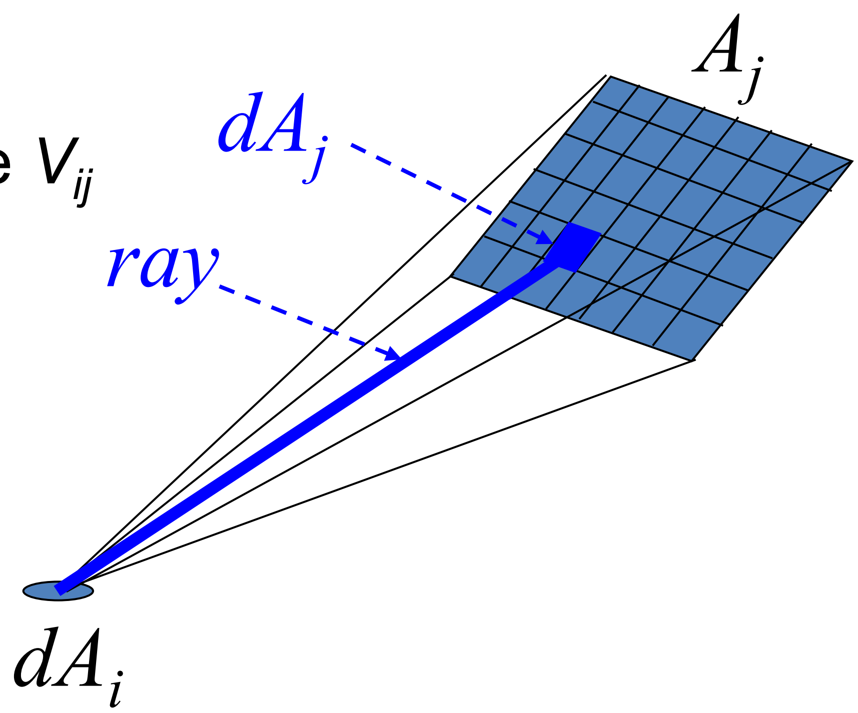
compute F_{dAidAj}

$$F_{dA_i dA_j} = \frac{\cos\theta_i \cos\theta_j}{\pi r^2} V_{ij} dA_j$$

sum up

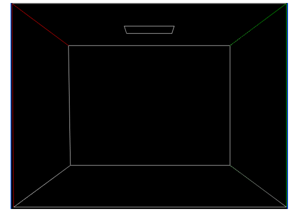
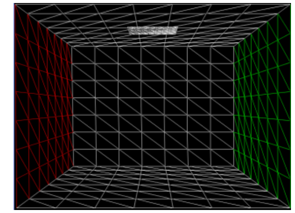
F_{dAiAj} += F_{dAidAj}

We have now F_{dAiAj}

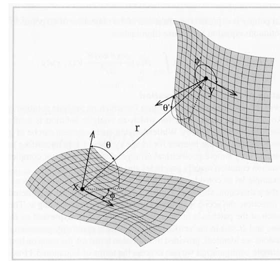
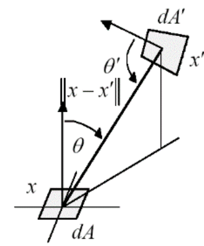


Classic Radiosity Algorithm

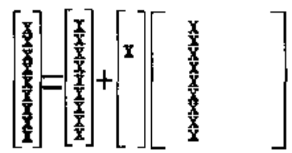
Mesh Surfaces into Elements

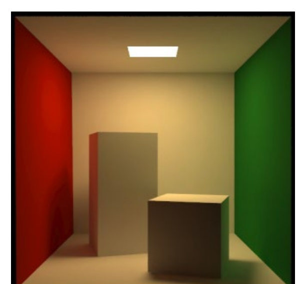
Compute Form Factors Between Elements

Solve Linear System for Radiosities



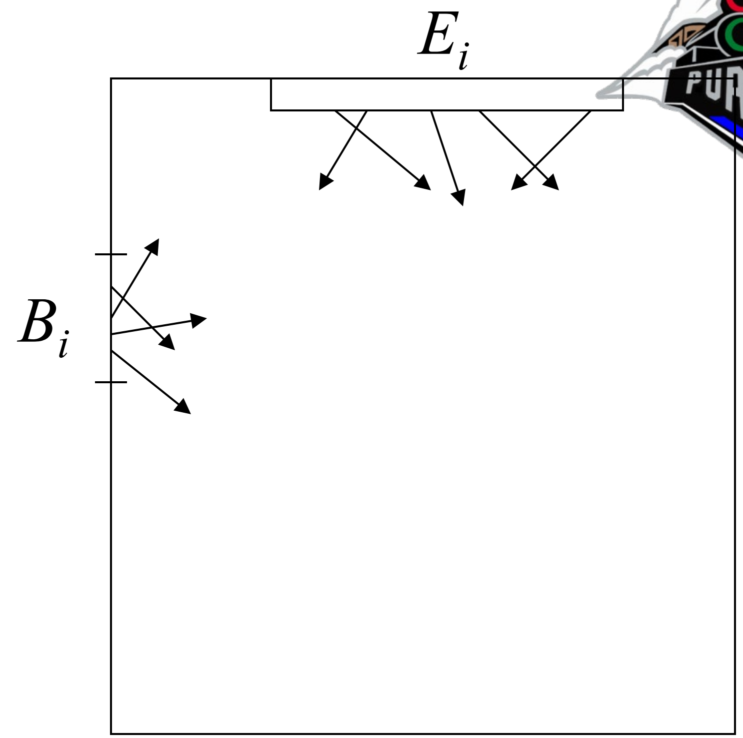
Reconstruct and Display Solution



Solving for radiosity solution

- The "Full Matrix" Radiosity Algorithm
- Gathering & Shooting
- Progressive Radiosity

Radiosity Matrix



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

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

$$\begin{bmatrix} 1-\rho_1 F_{11} & -\rho_1 F_{12} & \cdots & -\rho_1 F_{1n} \\ -\rho_2 F_{21} & 1-\rho_2 F_{22} & \cdots & -\rho_2 F_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -\rho_n F_{n1} & -\rho_n F_{n2} & \cdots & 1-\rho_n F_{nn} \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{bmatrix}$$

Radiosity Matrix

- The "full matrix" radiosity solution calculates the form factors between each pair of surfaces in the environment, then forms a series of simultaneous linear equations.

$$\begin{bmatrix} 1-\rho_1 F_{11} & -\rho_1 F_{12} & \cdots & -\rho_1 F_{1n} \\ -\rho_2 F_{21} & 1-\rho_2 F_{22} & \cdots & -\rho_2 F_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -\rho_n F_{n1} & -\rho_n F_{n2} & \cdots & 1-\rho_n F_{nn} \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{bmatrix}$$

- This matrix equation is solved for the "B" values, which can be used as the final intensity (or color) value of each surface.

Radiosity Matrix

- This method produces a complete solution, at the substantial cost of
  - first calculating form factors between each pair of surfaces
  - and then the solution of the matrix equation.
- This leads to substantial costs not only in computation time but in storage.

Solving for radiosity solution

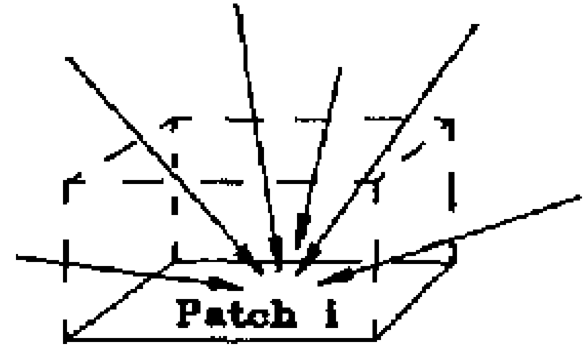
- The "Full Matrix" Radiosity Algorithm
- Gathering & Shooting
- Progressive Radiosity

Gathering

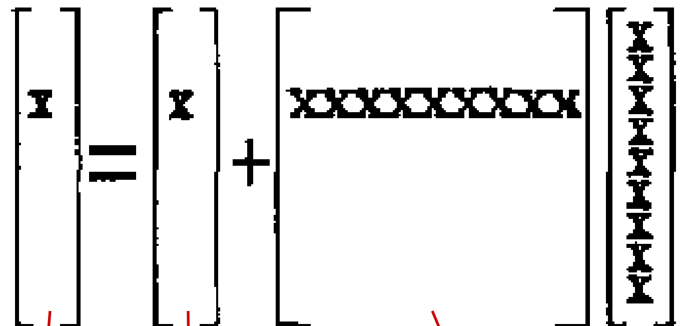
- In a sense, the light leaving patch i is determined by *gathering* in the light from the rest of the environment

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

$$B_i \ due\ to\ B_j = \rho_i B_j F_{ij}$$



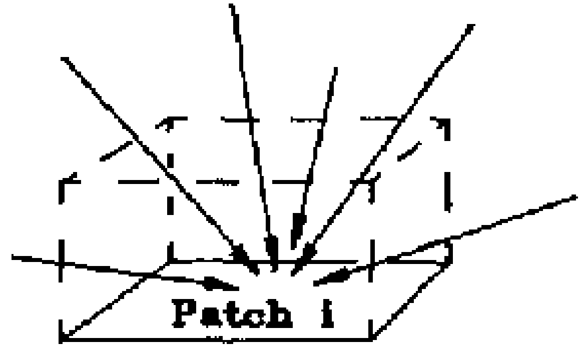
GATHERING



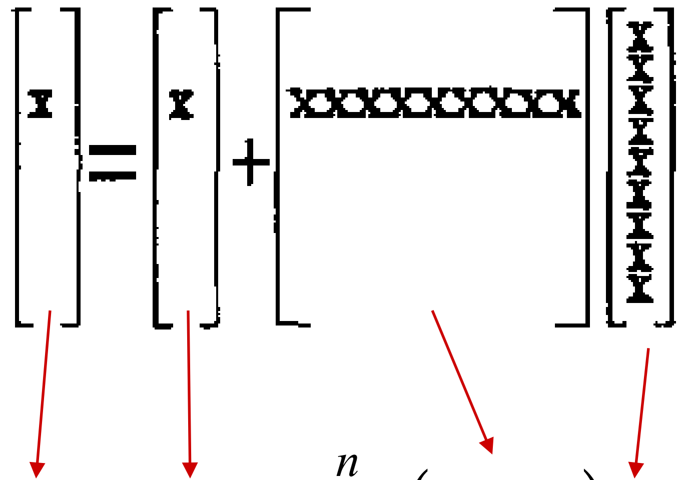
$$B_i = E_i + \sum_{j=1}^{n} \left(\rho_i F_{ij}\right) B_j$$

Gathering

- Gathering light through a hemi-cube allows one patch radiosity to be updated.

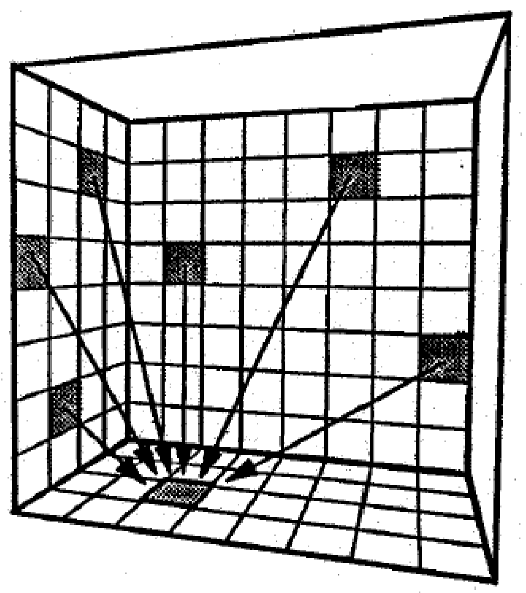


GATHERING



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

Gathering



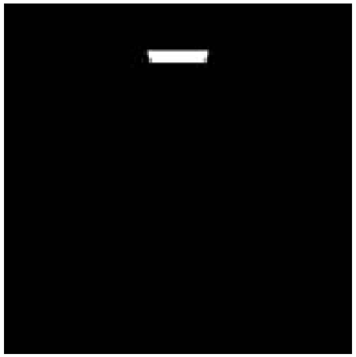
```
for(i=0; i<n; i++)
  B[i] = Be[i];

while( !converged ) {
  for(i=0; i<n; i++) {
    E[i] = 0;
    for(j=0; j<n; j++)
      E[i] += F[i][j]*B[j];
    B[i] = Be[i]+rho[i]*E[i];
  }
}
```

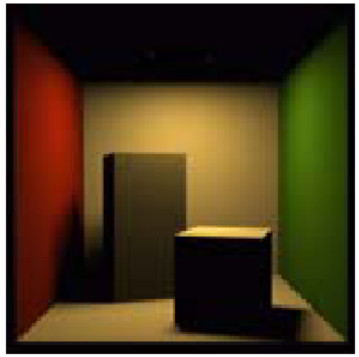
Row of F times B

Calculate one row of F and discard

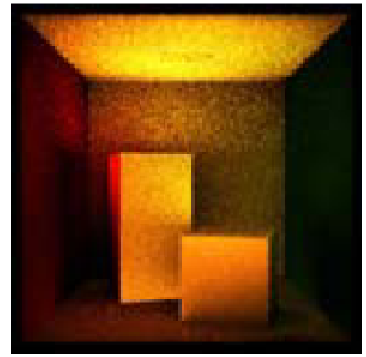
Successive Approximation



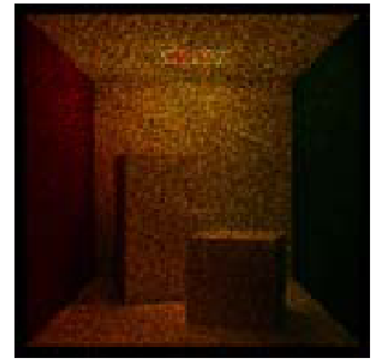
L_e



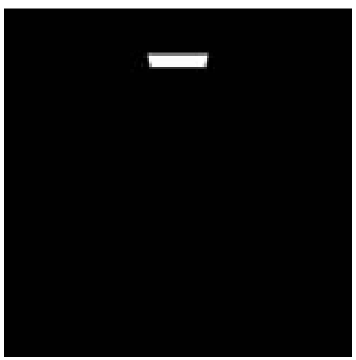
$K \circ L_e$



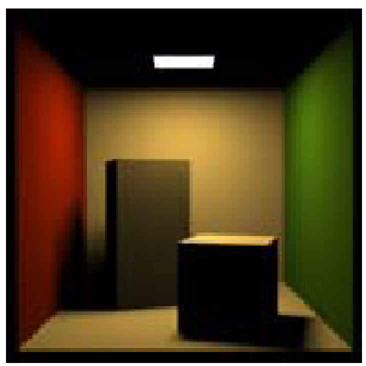
$K \circ K \circ L_e$



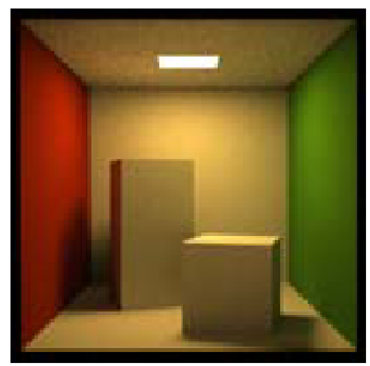
$K \circ K \circ K \circ L_e$



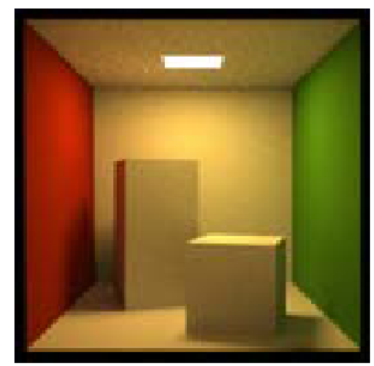
L_e



$L_e + K \circ L_e$

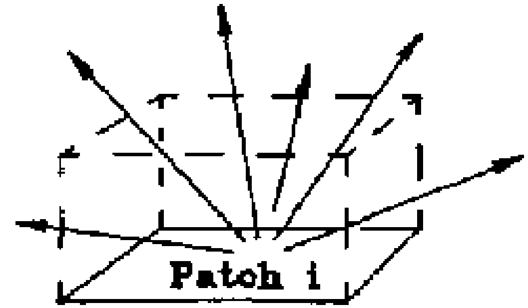


$L_e + \cdots K^2 \circ L_e$



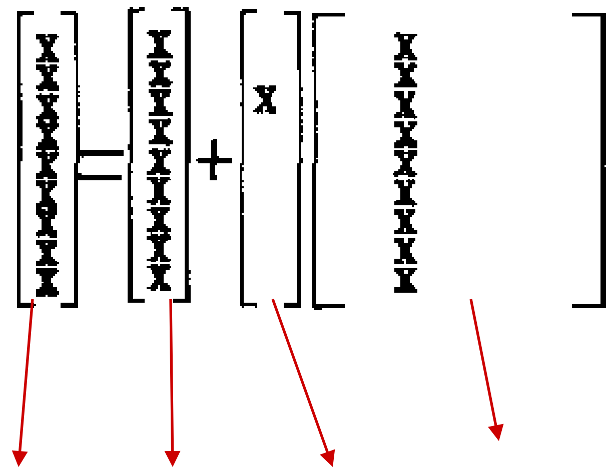
$L_e + \cdots K^3 \circ L_e$

Shooting



SHOOTING

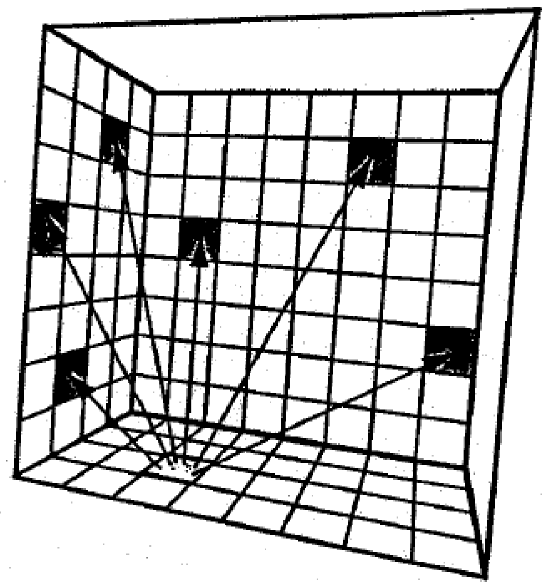
- Shooting light through a single hemi-cube allows the whole environment's radiosity values to be updated simultaneously.



For all *j* ⟹ $B_j = B_j + B_i\left(\rho_j E_{ji}\right)$

where $F_{ji} = \dfrac{F_{ij} A_i}{A_j}$

Shooting

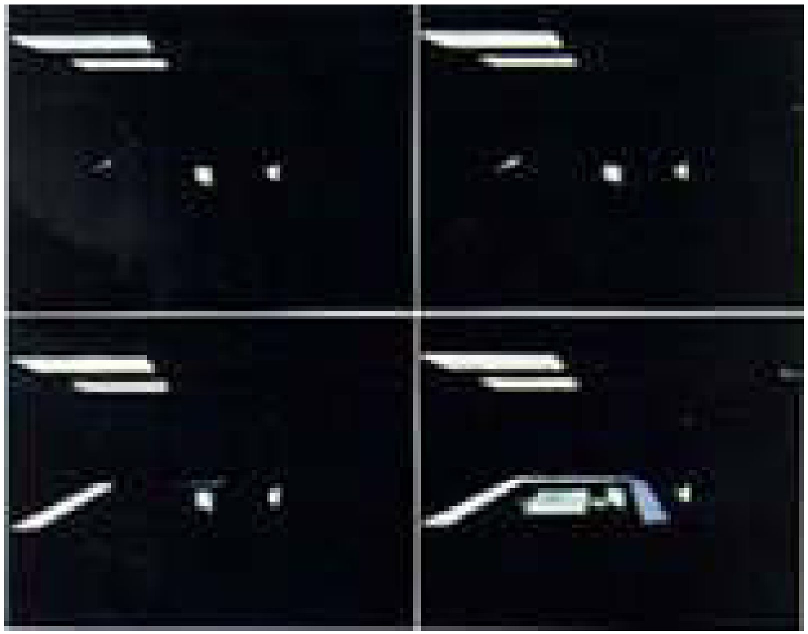


```
for(i=0; i<n; i++) {
  B[i] = dB[i] = Be[i];
  while( !converged ) {
    set i st dB[i] is the largest;
    for(j=0;j<n;j++)
      if(i!=j) {
        db =rho[j]*F[j][i]*dB[i];
        dB[j] += db;
        B[j] += db;
      }
    dB[i]=0;
  }
}
```

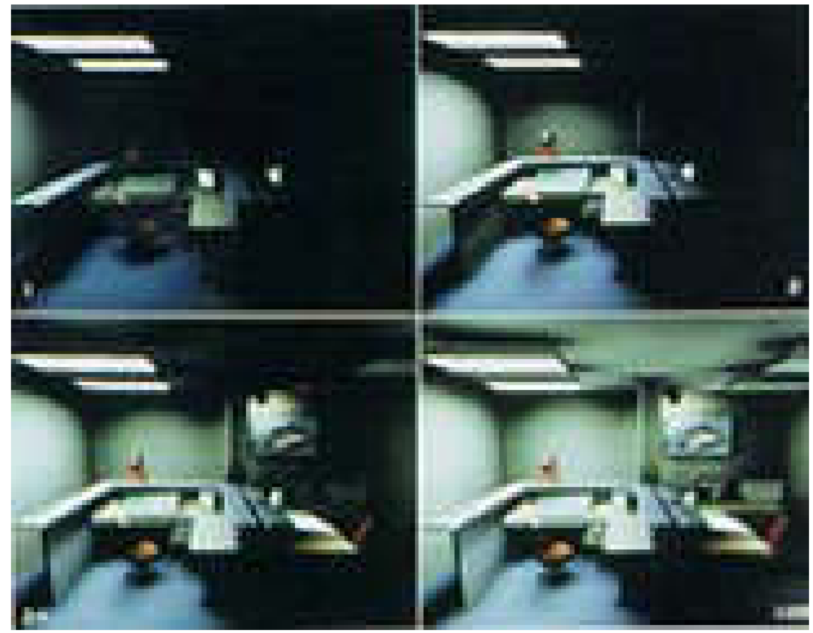
Brightness order

Column of F times B

Progressive Radiosity



(a)



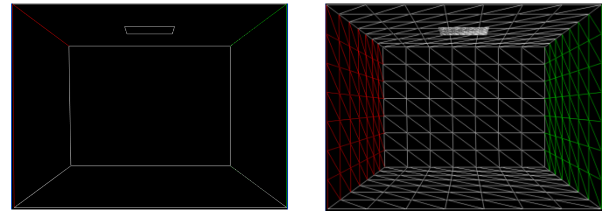
(b)

(a) Traditional Gauss-Seidel iteration of 1, 2, 24 and 100.
(b) Progressive Refinement (PR) iteration of 1, 2, 24 and 100.

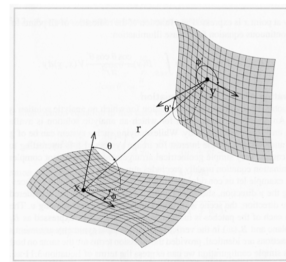
From Cohen, Chen, Wallace, Greenberg 1988

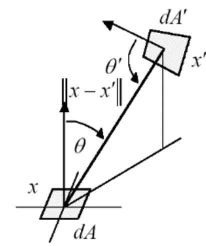
Classic Radiosity Algorithm

Mesh Surfaces into Elements

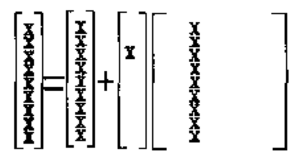


Compute Form Factors Between Elements

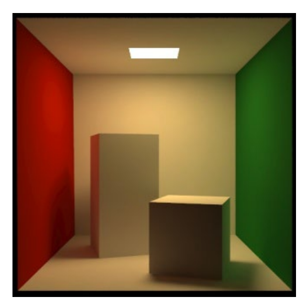




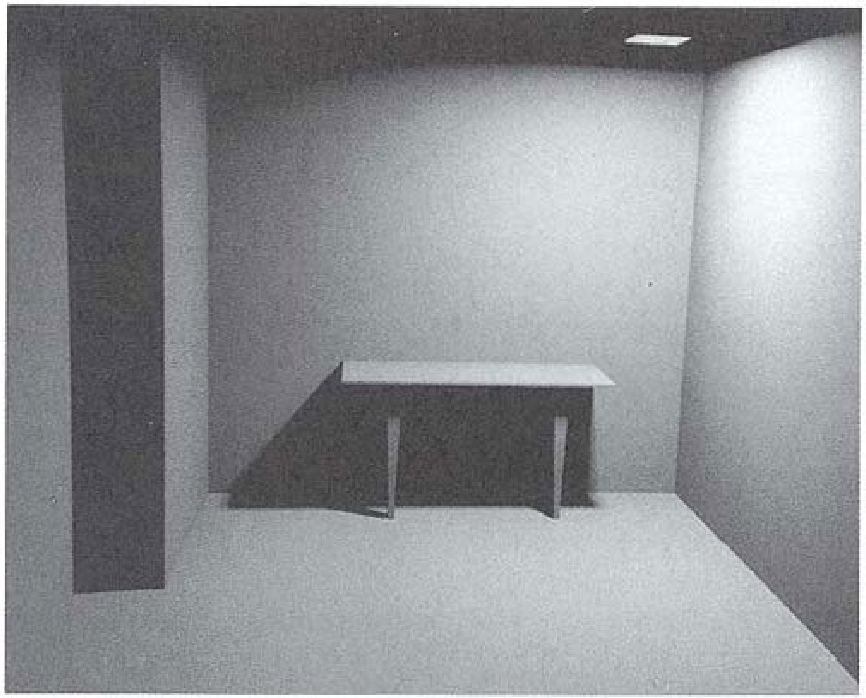
Solve Linear System for Radiosities

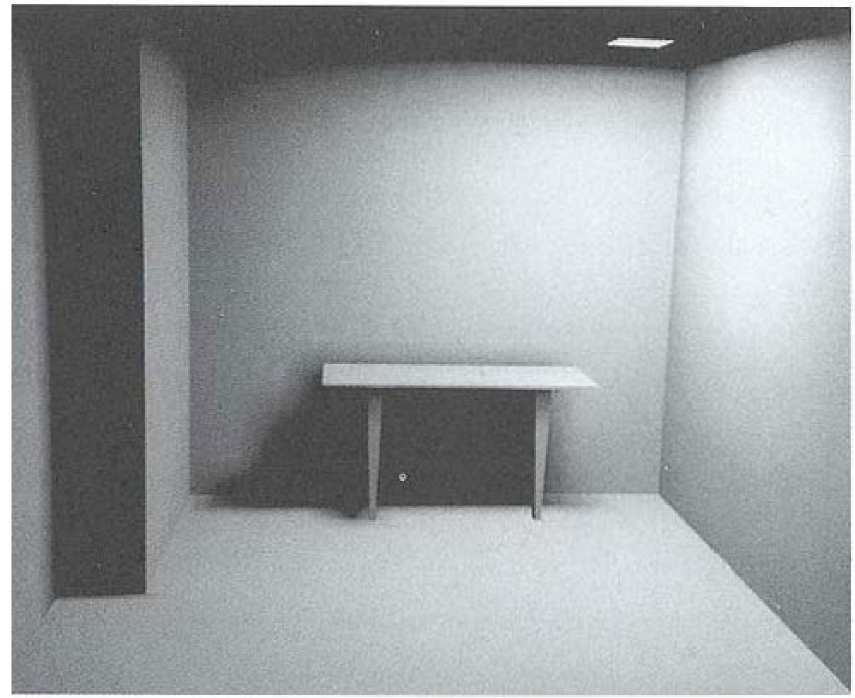


Reconstruct and Display Solution



Accuracy



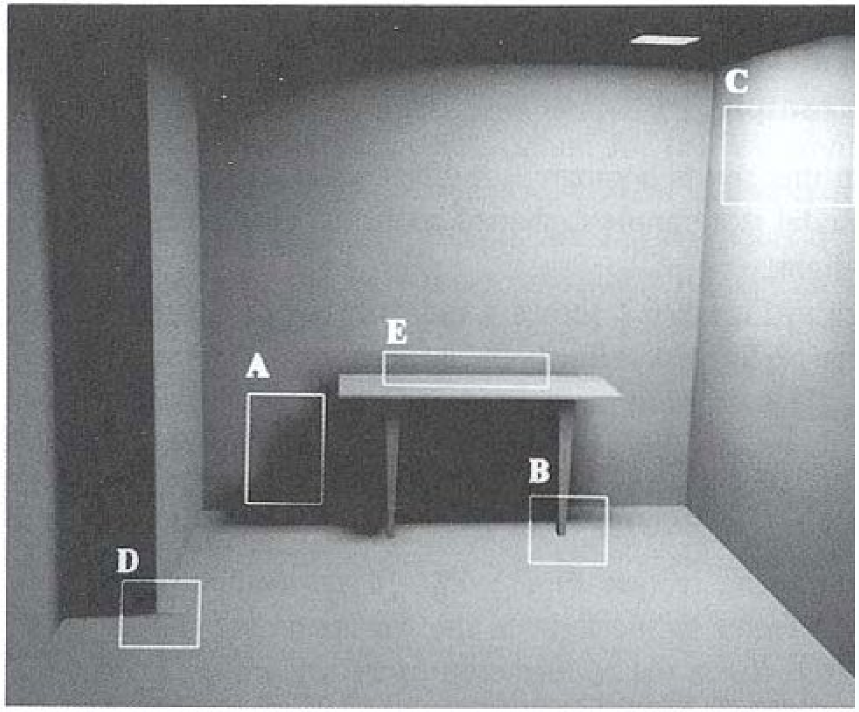


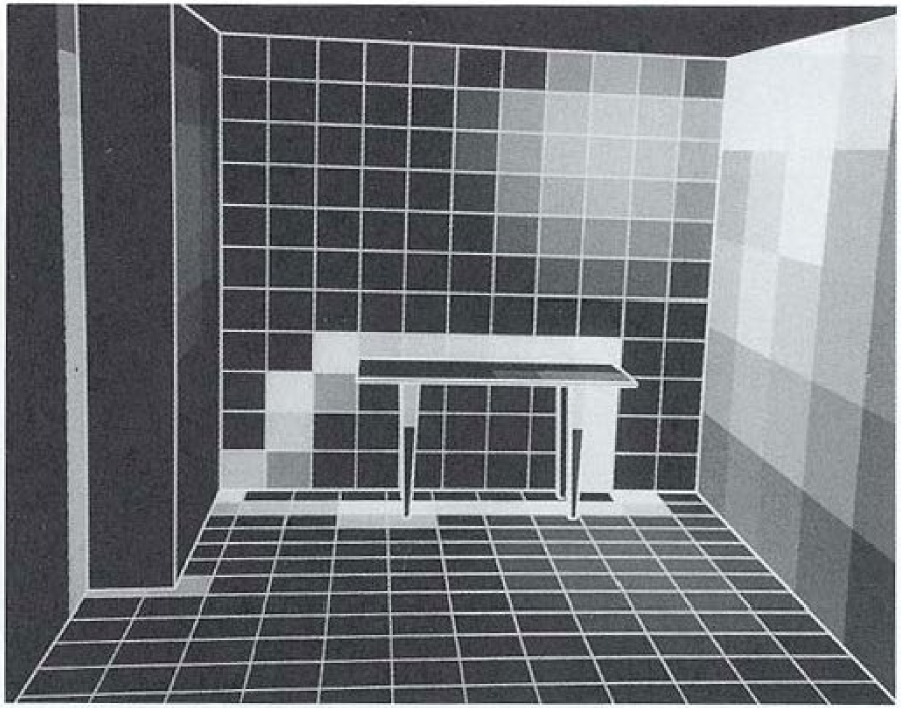
Reference Solution

Uniform Mesh

Table in room sequence from Cohen and Wallace

Artifacts – What can we do?

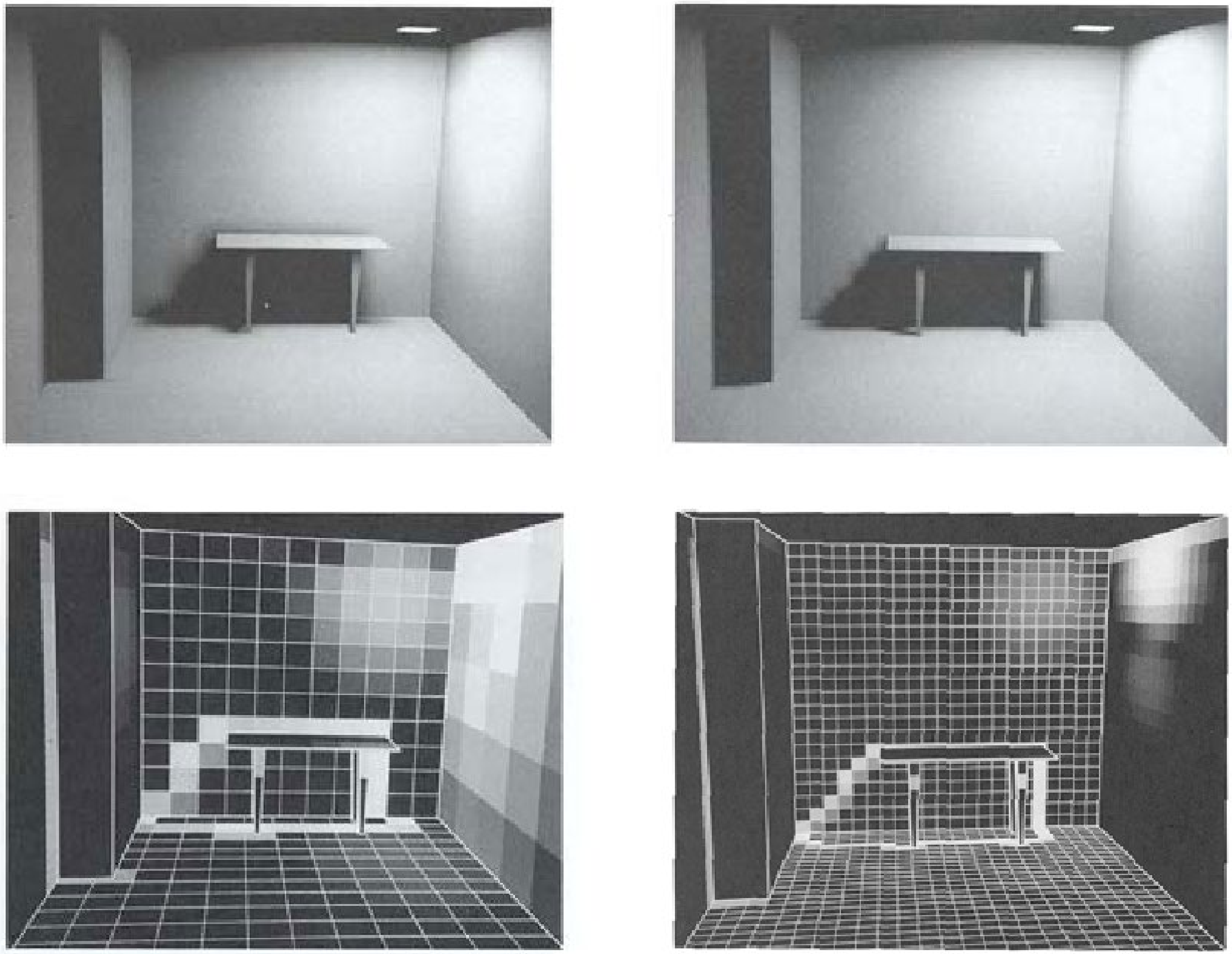




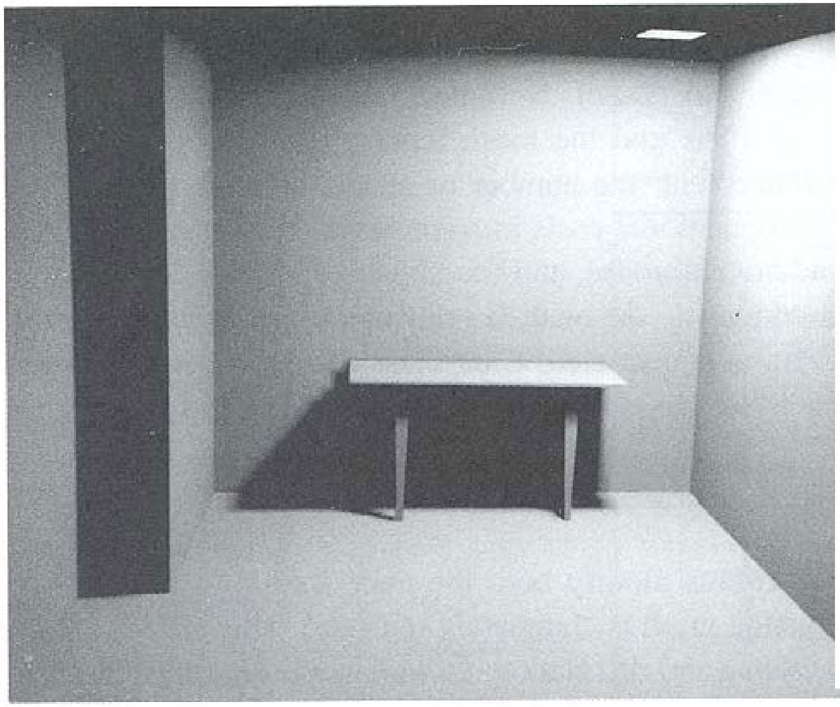
Error Image

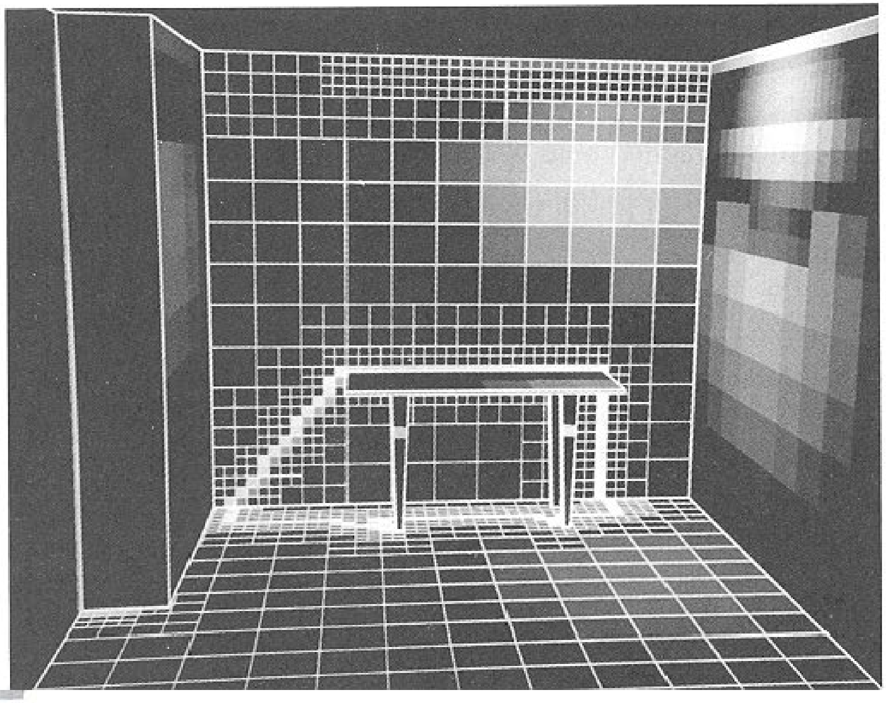
A. Blocky shadows
B. Missing features
C. Mach bands
D. Inappropriate shading discontinuities
E. Unresolved discontinuities

Option 1: Increasing Resolution

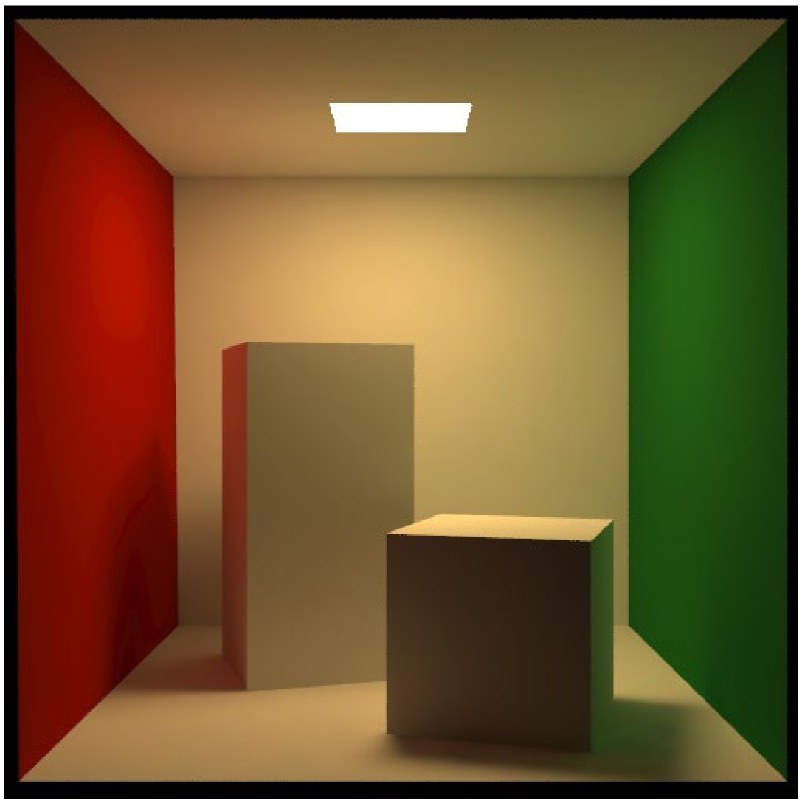


Option 2: Adaptive Meshing



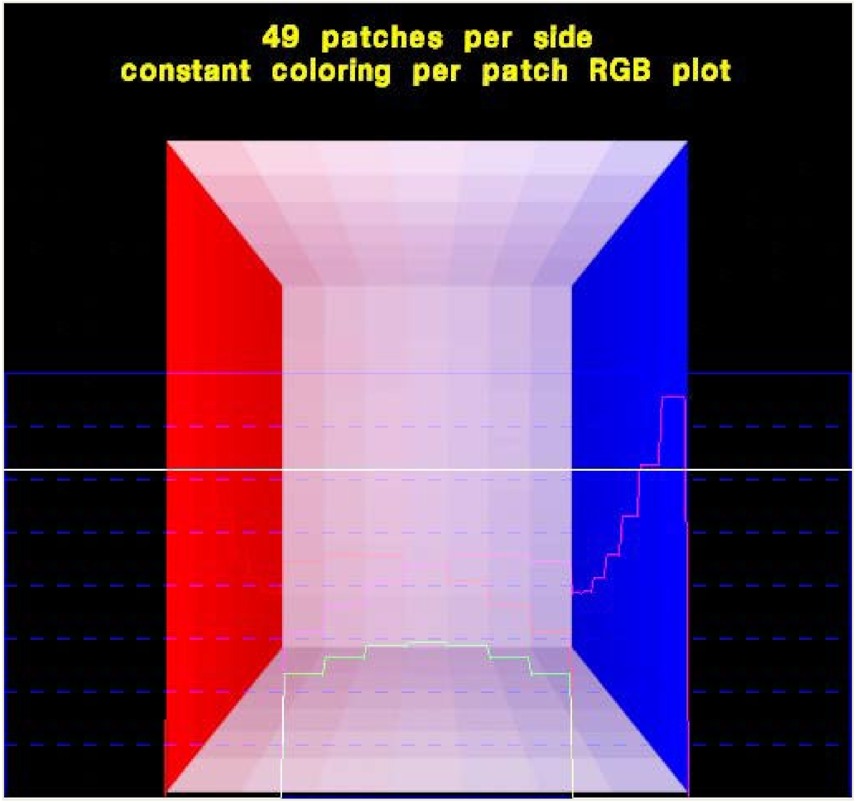


Example Radiosity Results



The Cornell Box

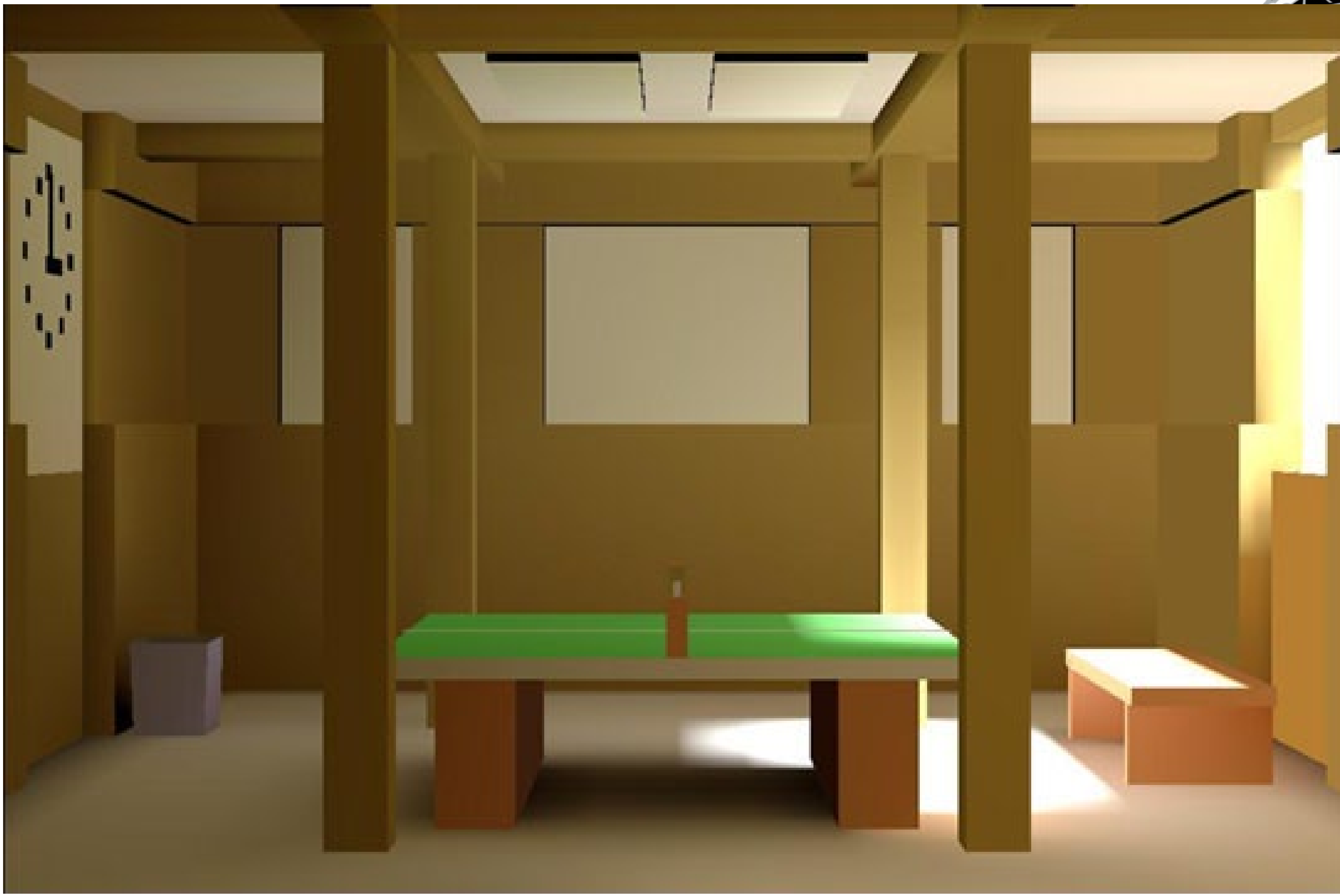
- This is the original Cornell box, as simulated by Cindy M. Goral, Kenneth E. Torrance, and Donald P. Greenberg for the 1984 paper *Modeling the interaction of Light Between Diffuse Surfaces*, Computer Graphics (SIGGRAPH '84 Proceedings), Vol. 18, No. 3, July 1984, pp. 213-222.
- Because form factors were computed analytically, no occluding objects were included inside the box.

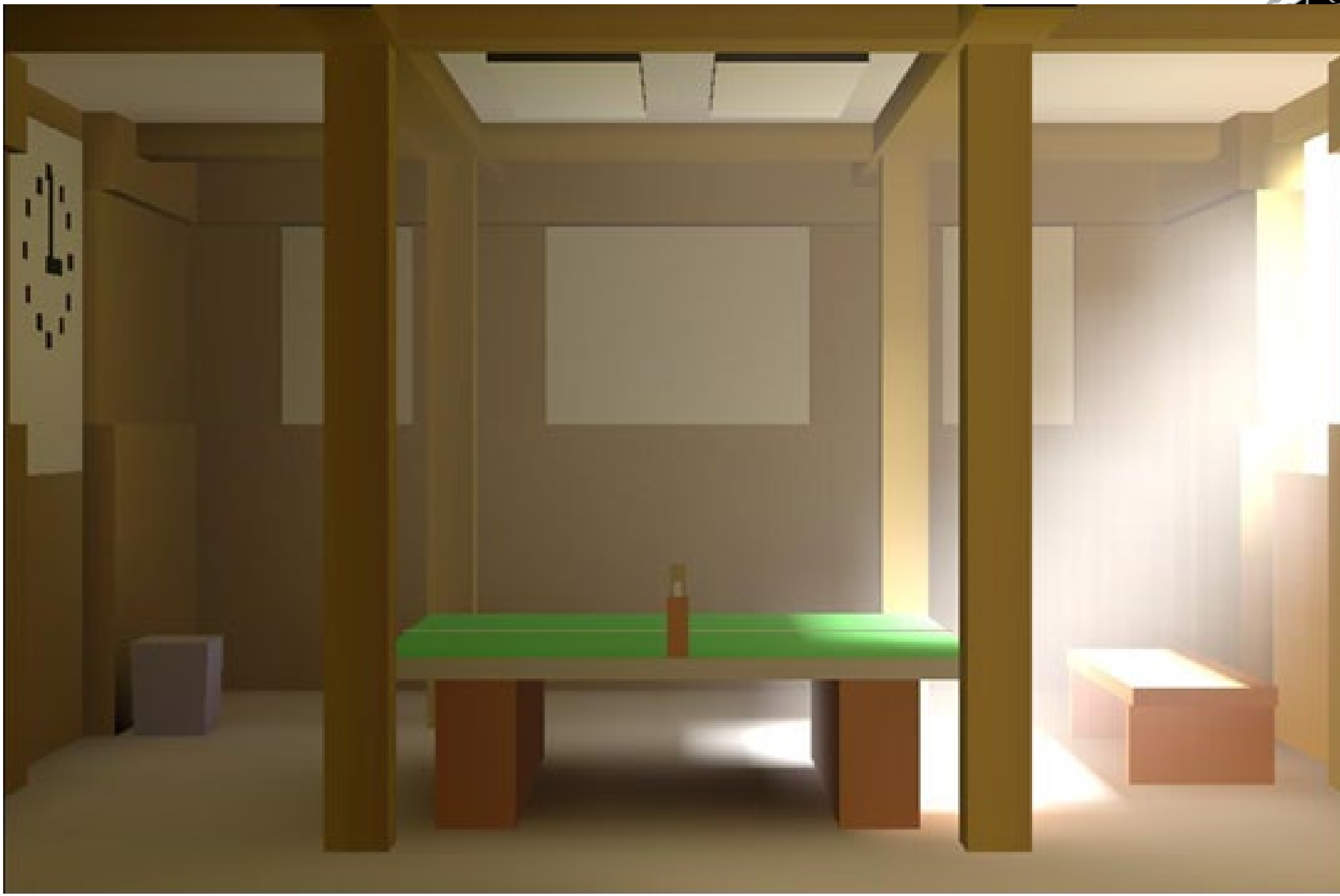


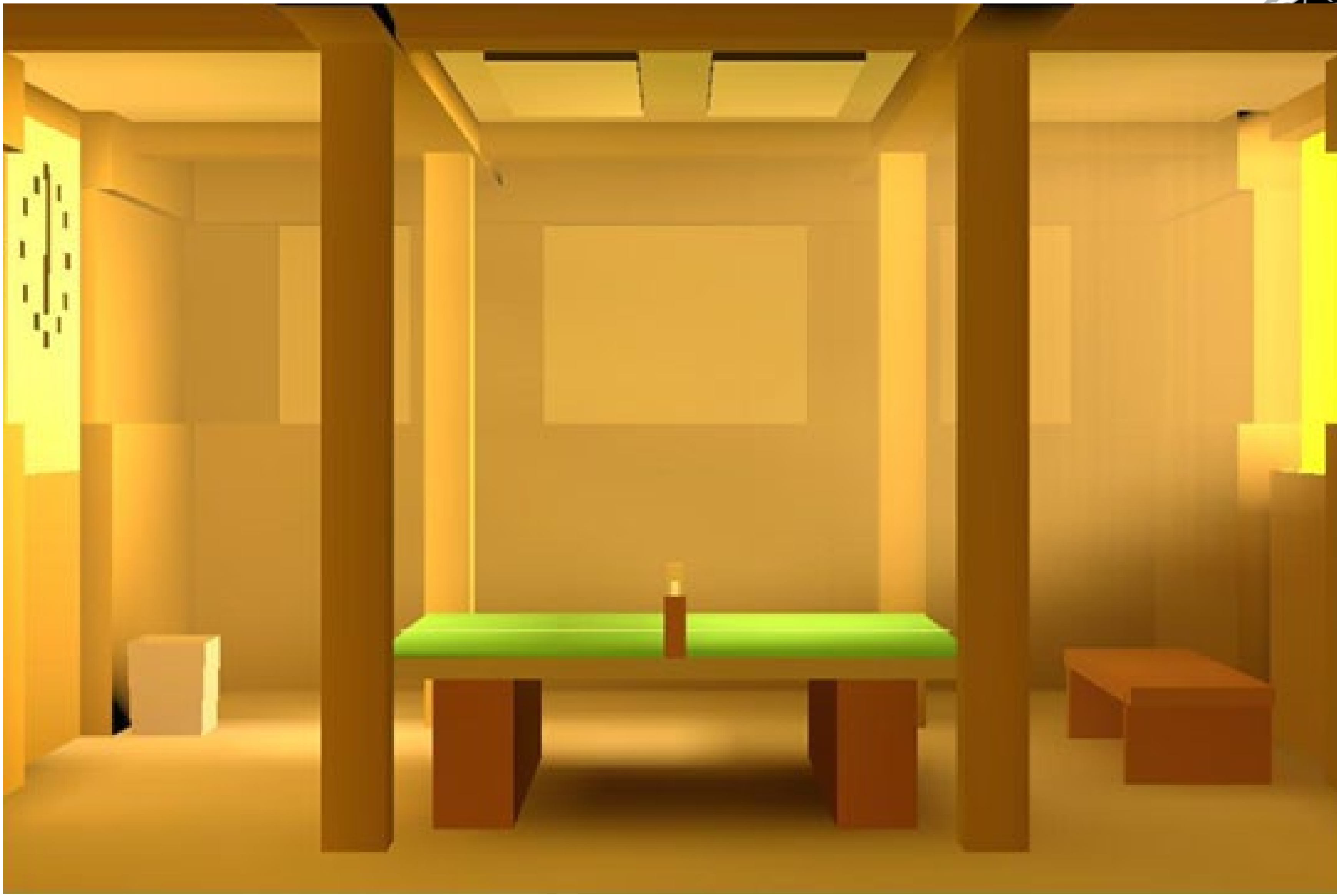
The Cornell Box

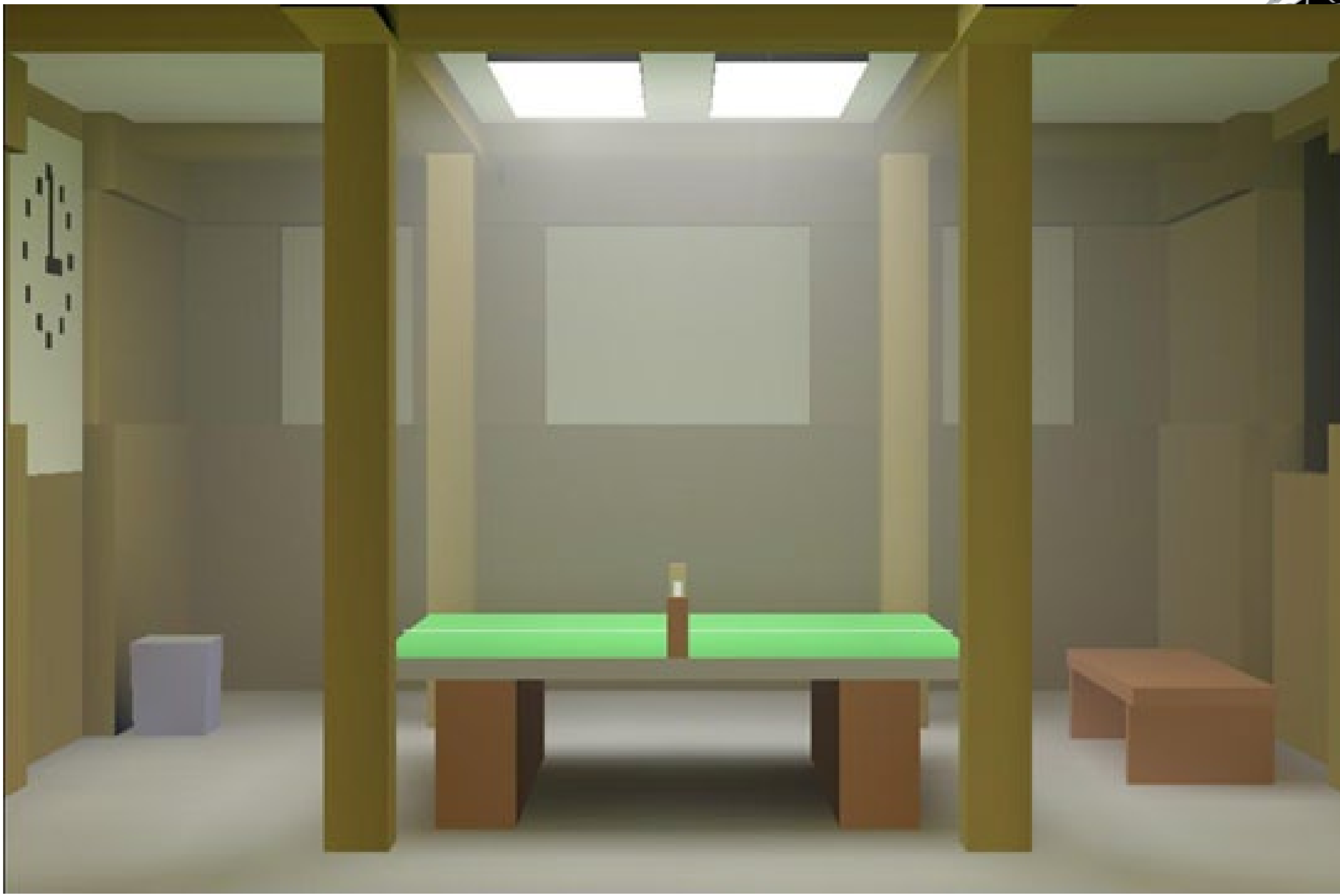
- This simulation of the Cornell box was done by Michael F. Cohen and Donald P. Greenberg for the 1985 paper *The Hemi-Cube, A Radiosity Solution for Complex Environments*, Vol. 19, No. 3, July 1985, pp. 31-40.

- The hemi-cube allowed form factors to be calculated using scan conversion algorithms (which were available in hardware), and made it possible to calculate shadows from occluding objects.









Discontinuity Meshing

Opera Lighting

- This scene from *La Boheme* demonstrates the use of focused lighting and angular projection of predistorted images for the background.
- It was rendered by Julie O'B. Dorsey, Francois X. Sillion, and Donald P. Greenberg for the 1991 paper *Design and Simulation of Opera Lighting and Projection Effects*.

Radiosity Factory

- These two images were rendered by Michael F. Cohen, Shenchang Eric Chen, John R. Wallace and Donald P. Greenberg for the 1988 paper *A Progressive Refinement Approach to Fast Radiosity Image Generation*.

- The factory model contains 30,000 patches, and was the most complex radiosity solution computed at that time.

- The radiosity solution took approximately 5 hours for 2,000 shots, and the image generation required 190 hours; each on a VAX8700.

Museum

- Most of the illumination that comes into this simulated museum arrives via the baffles on the ceiling.
- As the progressive radiosity solution executed, users could witness each of the baffles being illuminated from above, and then reflecting some of this light to the bottom of an adjacent baffle.
- A portion of this reflected light was eventually bounced down into the room.
- The image appeared on the proceedings cover of SIGGRAPH 1988.