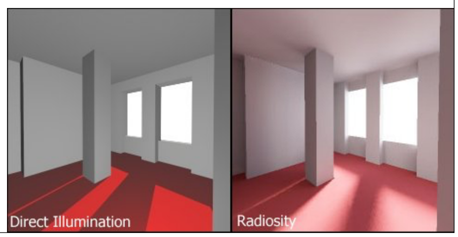
Radiosity

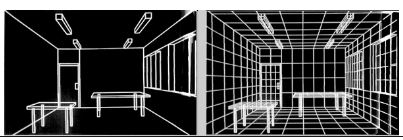
- ambient lighting is a hack
- want to model indirect lighting
- hard
- assume diffuse surfaces

radiosity = rate at which energy leaves surface



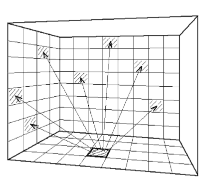
2-Pass Algorithm

- finite element method
- dice up surfaces into patches (n)
- first-pass: determine the radiosity of each patch
- second pass: render the model from the camera point of view



How?

$$B_i = E_i + \rho_i \sum_j B_j F_{j-i} \frac{A_j}{A_i}$$



B_i = radiosity (energy/unit time/unit area; w/m^2)

ρ_i = reflectance of patch i

E_i = rate at which energy leaves patch i

$F_{j\text{-}i}$ = form factor (fraction of energy leaving patch j and arriving at patch i)

- takes into account visibility

A_i = area of patch i

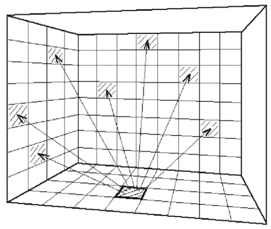
Goral et. al. 1984

Diffuse Environments

$$A_i F_{i-j} = A_j F_{j-i}$$

$$B_i = E_i + \rho_i \sum_j B_j F_{i-j}$$

$$B_i - \rho_i \sum_j B_j F_{i-j} = E_i$$



n simultaneous equations in n unknowns

Simultaneous Equations

$$\begin{bmatrix} 1-\rho_1 F_{1-1} & -\rho_1 F_{1-2} & \dots & -\rho_1 F_{1-n} \\ -\rho_2 F_{2-1} & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ -\rho_n F_{n-1} & \dots & \dots & 1-\rho_n F_{n-n} \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ \dots \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ \dots \\ E_n \end{bmatrix}$$

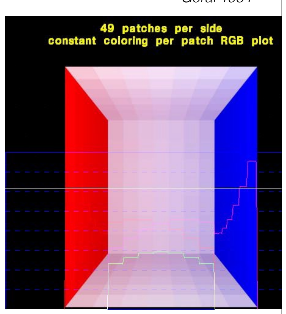
Solve for $B_1 \dots B_n$

Note: B_i and E_i are wavelength dependent

Solving Equations

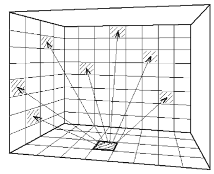
- solve matrix, n=1,000 … 1,000,000
- Gaussian Elim: $O(n^3)$ too slow
- Gauss-Sidel: $O(n^2)$ per iteration
  (each iteration is one bounce of light)

Goral 1984



note color bleeding

Steps Needed for Solution

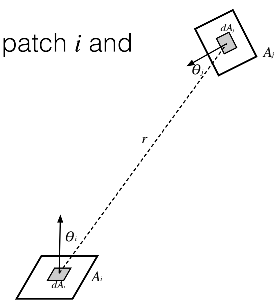


- solve for B_i:
  - compute form factors (need visibility)
  - solve $n \times n$ matrix for each frequency
    - can use iterative methods for matrix
  - compute radiosity per patch
- transfer radiosity to vertices
- linearly interpolate between vertices of each patch
  (Gouraud shading)

Form Factor

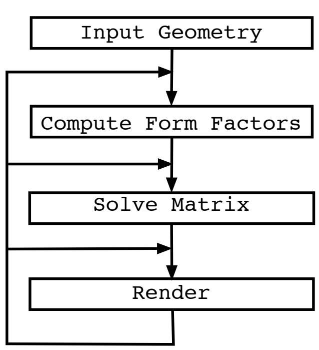
- $F_{i\text{-}j}$ = fraction of energy leaving patch i and arriving at patch j
- takes into account visibility

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



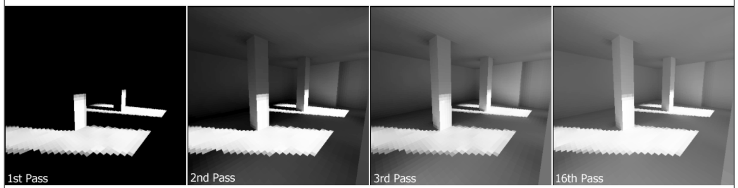
- H_{ij} is either 0 or 1, depending if dA_i sees dA_j

Program Flow



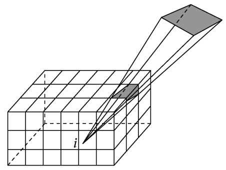
Gauss-Sidel Matrix Solution

- each iteration is one bounce of light
- 1st pass is direct light
- 2nd pass includes light after one bounce, …
- converges to solution

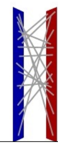


Computing Form Factor

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

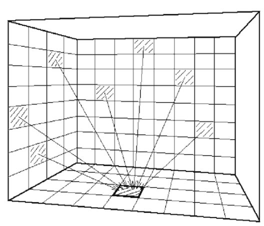


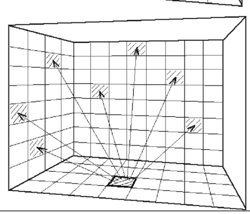
- need to compute visibility as well
- hemicube (use graphics hardware and table lookup)
- Monte Carlo



Speeding Up Radiosity

- Gauss-Sidel: gathering
- progressive refinement: shooting
- adaptive subdivision of patches
  (hierarchical radiosity)

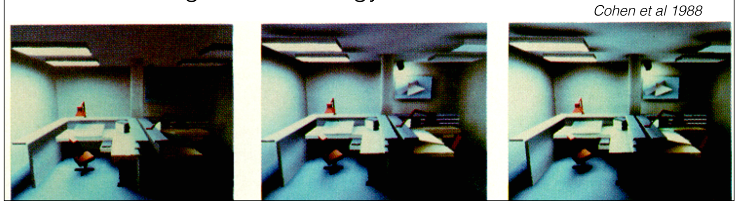




Progressive Refinement

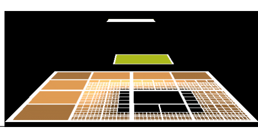
- don’t have to solve whole matrix
- “shoot” light energy from brightest emitters
- only compute form factors one row at a time
- iterate: patches with the most “unshot energy”
- remaining “unshot energy”-> ambient

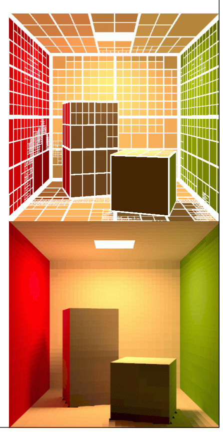
Cohen et al 1988



Hierarchical Radiosity

- start with coarse grid
- level of meshing dependent on closeness
- avoid computing form factors that represent small energy exchanges
- if form factor estimate > tolerance then subdivide
- if high radiosity gradient then subdivide

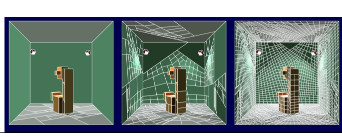




Hanrahan et al, 1991

Discontinuity Meshing

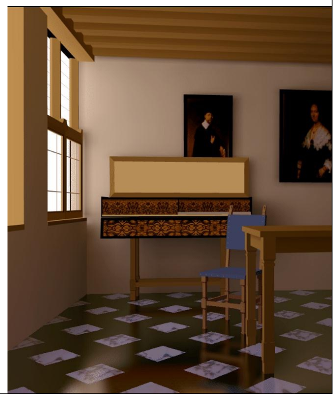
- squares not ideal patches
- shadow creep
- put mesh boundaries along sharp changes of intensity (eg. shadow boundaries)



Heckbert, 1991

Hybrid

- radiosity good for diffuse
- ray tracing good for specular
- radiosity first pass
- ray tracing second pass



Wallace et al 1987