The Radiosity Equation

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

Matrix form: $\boldsymbol{b} = \boldsymbol{e} + \boldsymbol{F}\boldsymbol{b}$

$$\boldsymbol{b}_i = B_i, \boldsymbol{e}_i = E_i, \boldsymbol{F}_{ij} = \rho_i F_{ij}$$

$$\boldsymbol{e} = (\boldsymbol{I} - \boldsymbol{F})\boldsymbol{b}$$

Simple Linear Eqn: $\boldsymbol{b} = (\boldsymbol{I} - \boldsymbol{F})^{-1}\boldsymbol{e}$

Simple Solution Strategy

- Provide the geometry of the scene
- Provide the surface properties (ρ_i, E_i)
- Construct $\boldsymbol{F}$
- Solve the system
- Use the computed radiosities as the intensities of the patches
- No lighting when rendering

Jacobi Relaxation

$$\sum_{j=1}^{N} M_{ij} B_j = E_i, \quad 1 \le i < N, \quad M_{ij} = \delta_{ij} - \rho_i F_{ij}$$

$$B_i = -\sum_{\substack{j=1 \\ j \ne i}}^{N} \frac{M_{ij}}{M_{ii}} B_j + \frac{E_i}{M_{ii}}, \quad 1 \le i < N$$

- Jacobi relaxtion: Start with a guess for B_i, then: $$B_i^{(m)} = -\sum_{\substack{j=1 \\ j \ne i}}^{N} \frac{M_{ij}}{M_{ii}} B_j^{(m-1)} + \frac{E_i}{M_{ii}}, \quad 1 \le i < N$$

Gauss-Seidel Relaxation

$$B_i^{(m)} = -\sum_{j=1}^{i-1} \frac{M_{ij}}{M_{ii}} B_j^{(m)} - \sum_{j=i+1}^{N} \frac{M_{ij}}{M_{ii}} B_j^{(m-1)} + \frac{E_i}{M_{ii}}, \quad 1 \le i < N$$

- Uses values from current iteration
- Allows updating in place
- Requires **strictly diagonally dominant:**

$$|M_{ii}| > \sum_{\substack{j=1 \\ j \ne i}}^{N} |M_{ij}|, \qquad 1 \le i < N$$

Displaying the Results

- Color is handled by discretizing wavelength and solving each channel separately
- Smooth shading:
  - Patch radiosities are mapped to vertex colors by averaging the radiosities of the patches incident upon the vertex
  - Per-vertex colors then used to Gourand shade

Value for Computation

- Most of the time is spent computing form factors - must solve N visibility problems
- However, same form factors for different illumination conditions
- Result is view independent - have radiosities for all patches. May be good or may be wasteful

Form Factor Computations

- Direct integration
- Conversion to contour integration
- Form factor algebra
- Monte Carlo integration
- Projection methods
  - Hemisphere
  - Hemicube

Direct Integration

- Only works for simplest cases

$$F_{x,disc} = \frac{u^2}{u^2 + v^2}$$

$$F_{x,rect} = \frac{1}{2\pi}\left[\frac{X}{\sqrt{1+X^2}}\tan^{-1}\left(\frac{Y}{\sqrt{1+X^2}}\right) + \frac{Y}{\sqrt{1+Y^2}}\tan^{-1}\left(\frac{X}{\sqrt{1+Y^2}}\right)\right] \qquad X = \frac{u}{w}, Y = \frac{v}{w}$$

$$F_{\perp} = \frac{1}{\pi X}\left\{X\tan^{-1}\left(\frac{1}{X}\right) + Y\tan^{-1}\left(\frac{1}{Y}\right) - \sqrt{X^2+Y^2}\tan^{-1}\left(\frac{1}{\sqrt{X^2+Y^2}}\right)\right\}$$
$$+ \frac{1}{4\pi X}\left\{\ln\left[\frac{(1+X^2)(1+Y^2)}{1+X^2+Y^2}\right] + X^2\ln\left[\frac{X^2(1+X^2+Y^2)}{(1+X^2)(X^2+Y^2)}\right] + Y^2\ln\left[\frac{Y^2(1+X^2+Y^2)}{(1+Y^2)(X^2+Y^2)}\right]\right\}$$

There is a formula for 2 isolated polygons, but it assumes they can see each other fully!

Contour Integral

- Use Stokes' theorem to convert the area integrals into contour integrals

$$F_{ij} = \frac{1}{2\pi A_i}\oint_{C_i}\oint_{C_j} \ln r\, d\vec{x} \cdot d\vec{y}$$

- But, still only works for unoccluded patches
- Care must be taken when $r \to 0$

Form Factor Algebra

- Use the properties of form factors and direct integration to compute new form factors
  - Additivity
  - Reciprocity
  - Point to disc form factors
  - Rectangle to rectangle form factors
- Difficult to apply automatically
  - Useful only in tiny scenes
- Unoccluded case only

Projection Methods

- For patches that are far apart compared to their areas, the inner integral in the form factor doesn't vary much
- So, compute point to patch form factors and weigh by area

$$F_{x,P} = \int_{y \in P} \frac{\cos\theta \cos\theta'}{\pi r^2} V(x,y)\,dy$$

Nusselt's Analogy

- Integrate over visible solid angle instead of visible patch area:

$$F_{x,P} = \frac{1}{\pi} \int_{\Omega_P} \cos\theta\, d\omega$$

$F_{x,P}$ is the fraction of the area of the unit disc in the base plane obtained by projecting the surface patch P onto the unit sphere centered at x and then orthogonally down onto the base plane.

The Hemicube

- We don't have to project onto a sphere
- We have algorithms, and even hardware, for projecting onto planar surfaces
- The hemicube consists of 5 such faces

$$\Delta F = \frac{\Delta A}{\pi(1 + x^2 + y^2)^2}, \text{ for the top face}$$

$$\Delta F = \frac{z\Delta A}{\pi(1 + z^2 + y^2)^2}, \text{ for a side face}$$

Hemicube, cont.

$$F_{x,P_j} = \sum_{q \in C(j)} \Delta F_q$$

- Pretend each face of the hemicube is a screen, and project the world onto it
- Code each polygon with a color, and count the pixels of each color to determine $C(j)$
- Quality depends on hemicube resolution and z-buffer depth

Overview

- Input geometry and surface properties
- Find form factors
- Solve the system
- Output geometry with per-face or per-vertex intensities