

Lecture 23: Radiosity

And he shall be as the light of the morning...

2 Samuel 23:4

1. Radiosity

Radiosity models the transfer of light between surfaces. Recursive ray tracing also models light bouncing off surfaces, but ray tracing makes several simplifying assumptions that give scenes a harsh, unnatural look. Ray tracing assumes that all light sources are point sources and that the ambient light is constant throughout the scene. These assumptions lead to stark images with sharp shadows. In contrast, radiosity models all surfaces as both emitters and reflectors. This approach softens the shadows and provides a more realistic model for ambient light.

Informally, radiosity is the rate at which light energy leaves a surface. There are two contributions to radiosity: emission and reflection. Hence

$$\text{radiosity} = \text{emitted energy} + \text{reflected energy}.$$

For the purpose of display, we shall identify intensity with radiosity.

Radiosity computations typically take much longer than recursive ray tracing because the model of light is much more complex. To simplify these computations, we shall model only diffuse reflections; we shall not attempt to model specular reflections with radiosity. Since radiosity replaces ambient and diffuse intensity, radiosity is view independent. Thus we can reuse the same computation for every viewpoint once we compute the radiosity of all the surfaces. View dependent calculations are required only to compute hidden surfaces.

2. The Radiosity Equations

We will begin with a very general integral equation called the *Rendering Equation* based on energy conservation. We will then repeatedly simplify and discretize this equation till we get a large system of linear equations -- the *Radiosity Equations* -- which we can solve numerically for the radiosity of each surface. Radiosity is identified with intensity, so once we have the radiosity for each surface we can render the scene.

2.1 The Rendering Equation. Energy conservation for light is equivalent to

$$\text{Total Illumination} = \text{Emitted Energy} + \text{Reflected Energy}.$$

We can rewrite these innocent looking words as an integral equation called the *Rendering Equation*.

Rendering Equation

$$I(x, x') = E(x, x') + \int_S \rho(x, x', x'') I(x', x'') dx'' \quad (2.1)$$

where

$I(x, x')$ is the total energy passing from x' to x .

$E(x, x')$ is the energy emitted directly from x' to x .

$\rho(x, x', x'')$ is the reflection coefficient -- the percentage of the energy transferred from x'' to x' that is passed on to x .

Essentially all of the computations in Computer Graphics that involve light are summarized in the Rendering Equation. Notice, in particular, that the Rendering Equation is precisely the set up for recursive ray tracing!

2.2 The Radiosity Equation -- Continuous Form. The continuous form of the Radiosity Equation is just the Rendering Equation restricted to diffuse reflections. Once again by conservation of energy:

Radiosity = Emitted Energy + Reflected Energy.

Now, however, since we are dealing only with diffuse reflections, we can be more specific about the form of the reflected energy. Restricting to diffuse reflections leads to the following integral equation for radiosity.

Radiosity Equation -- Continuous Form

$$B(x) = E(x) + \rho_d(x) \int_S B(y) \frac{\cos\theta \cos\theta'}{\pi r^2(x, y)} V(x, y) dy \quad (2.2)$$

where

$B(x)$ is the radiosity at the point x , which we identify with the intensity or energy reflecting off a surface in any direction -- that is, the total power leaving a surface/unit area/solid angle. This energy is uniform in all directions, since we are assuming that the scene has only diffuse reflectors.

$E(x)$ is the energy emitted directly from a point x in any direction. This energy is uniform in all directions, since we are assuming that the scene has only diffuse emitters.

$\rho_d(x)$ is the diffuse reflection coefficient -- the percentage of energy reflected in all directions from the surface at a point x . By definition, $0 \leq \rho_d(x) \leq 1$.

$V(x, y)$ is the visibility term:

$V(x, y) = 0$ if x is not visible from y .

$V(x, y) = 1$ if x is visible from y .

θ = angle between the surface normal (N) at x and the light ray (L) to y .

θ' = angle between surface normal (N') at y and the light ray (L) to x .

$r(x, y)$ = distance from x to y .

In the Radiosity Equation, the term

$$\int_S B(y) \frac{\cos\theta \cos\theta'}{\pi r^2(x,y)} V(x,y) dy = \text{energy reaching the point } x \text{ from all other points } y,$$

so

$$\rho_d(x) \int_S B(y) \frac{\cos\theta \cos\theta'}{\pi r^2(x,y)} V(x,y) dy = \text{total energy reflected from } x.$$

The factor $1/r^2(x,y)$ models an inverse square law, since the intensity of light varies inversely as the square of the distance. The factors $\cos\theta, \cos\theta'$ come from Lambert's Law (see Lecture 16, Section 4) and represent projections of the flux onto the emitting and reflecting surfaces (see Figure 1 and the accompanying discussion). The appearance of the factor π in the denominator will be explained shortly below.

To understand the cosine terms better, consider two small surface patches. Recall that the intensity (or radiosity) on any facet from any other facet is given by

$$I_{receptor} = \frac{\text{Light Deposited}}{\text{Unit Area}} = \frac{\text{Beam Cross Section}}{\text{Receptor Facet Area}} \times I_{source}$$

where

$$I_{source} = \frac{\text{Light Emitted}}{\text{Unit Area}} = \frac{\text{Beam Cross Section}}{\text{Source Facet Area}} \times I_{emitter}.$$

But from Figure 1,

$$\frac{\text{Beam Cross Section}}{\text{Facet Area}} = \cos(\theta), \cos(\theta').$$

so

$$I_{receptor} = \cos(\theta) \cos(\theta') I_{emitter}.$$

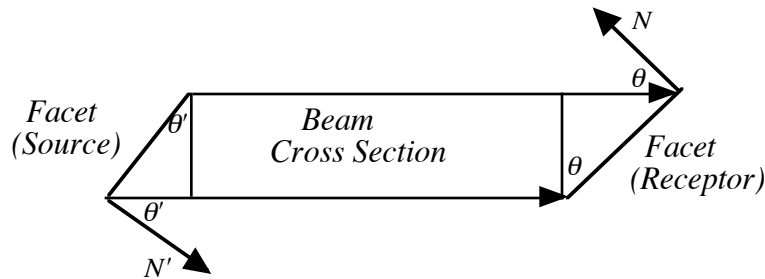


Figure 1: Lambert's Law. Intensity is given by the ratio of the beam cross section to the facet area, which, in turn, is equal to the cosine of the angle between the beam and the surface normal. Two cosines appear: one for the source and one for the receptor. Thus Lambert's Law accounts for the two cosines that appear in the Radiosity Equation.

The factor π in the denominator of the second term on the right hand side of Equation (2.2) arises for the following reason. Recall that in the Rendering Equation

$\rho(x', x, x'')$ = the percentage of the energy transferred from x'' to x that is passed on to a single point x' . (Note that here we have reversed the roles of x and x' .)

whereas in the Radiosity Equation

$\rho_d(x)$ = the percentage of energy reflected in all directions from the surface at a point x .

Thus we need to get from $\rho(x', x, x'')$ in the Rendering Equation to $\rho_d(x)$ in the Radiosity Equation. We shall now show that these two functions differ by a factor of π .

Directions can be represented by points on the unit sphere, and points on the unit sphere can be parametrized by two angles θ, ϕ . The angle θ represents the angle between the z -axis and the vector from the center of the sphere to the point (θ, ϕ) on the sphere, and the angle ϕ represents the amount of rotation around the equator from the x -axis to the great circle of constant longitude ϕ . Since $\rho_d(x)$ is the percentage of energy reflected in all direction whereas $\rho(x', x, x'')$ is the percentage of energy reflected in one fixed direction, it follows that $\rho_d(x)$ is the integral of $\rho(x', x, x'')$ over the unit hemisphere -- that is, over all the directions that make an acute angle with the normal vector at x , which we identify with the z -axis. A factor of $\cos(\theta)$ must appear in this integral for the same reason (projection) that this factor appears in Lambert's Law. Thus

$$\rho_d(x) = \int_H \rho(x', x, x'') \cos(\theta) dS,$$

where H is a unit hemisphere centered at x . Since we are dealing with diffuse reflection, the function $\rho(x', x, x'')$ is the same in all directions. Thus we can pull $\rho(x', x, x'')$ outside the integral, so

$$\rho_d(x) = \rho(x', x, x'') \int_H \cos(\theta) dS. \quad (2.3)$$

It remains then to compute $\int_H \cos(\theta) dS$.

To compute this integral, we use the parametrization (θ, ϕ) of the unit sphere provided by spherical coordinates -- that is, by setting

$$S(\theta, \phi) = (\sin(\theta) \cos(\phi), \sin(\theta) \sin(\phi), \cos(\theta)). \quad (2.4)$$

With this parametrization, a differential area (parallelogram) on the unit sphere is given by

$$dS = \left| \frac{\partial S(\theta, \phi)}{\partial \theta} d\theta \times \frac{\partial S(\theta, \phi)}{\partial \phi} d\phi \right| = \left| \frac{\partial S(\theta, \phi)}{\partial \theta} \times \frac{\partial S(\theta, \phi)}{\partial \phi} \right| d\theta d\phi.$$

But by Equation (2.4)

$$\frac{\partial S(\theta, \phi)}{\partial \theta} = (\cos(\theta) \cos(\phi), \cos(\theta) \sin(\phi), -\sin(\theta))$$

$$\frac{\partial S(\theta, \phi)}{\partial \phi} = (-\sin(\theta) \sin(\phi), \sin(\theta) \cos(\phi), 0),$$

so by direct computation

$$\left| \frac{\partial S(\theta, \phi)}{\partial \theta} \times \frac{\partial S(\theta, \phi)}{\partial \phi} \right| = \sin(\theta).$$

Therefore

$$\int_H \cos(\theta) dS = \int_0^{2\pi} \int_0^{\pi/2} \cos(\theta) \sin(\theta) d\theta d\phi = \int_0^{2\pi} \left. \frac{\sin^2(\theta)}{2} \right|_0^{\pi/2} d\phi = \frac{1}{2} \int_0^{2\pi} d\phi = \pi.$$

We conclude then from Equation (2.3) that

$$\rho_d(x) = \pi \rho(x', x, x'').$$

Thus when we replace $\rho(x', x, x'')$ in the Rendering Equation by $\rho_d(x)$ in the Radiosity Equation, we must divide by π . This accounts for the factor π in the denominator of the second term on the right hand side of the Radiosity Equation.

2.3 The Radiosity Equations -- Discrete Form. To find the radiosity at any point, we must solve the Radiosity Equation for $B(x)$ -- that is, we must calculate the integral on the right hand side of Equation (2.2). But this integral is not easy to compute. Worse yet $B(u)$ appears on both sides of the Radiosity Equation; thus we must know $B(u)$ to find $B(u)$!

The solution to both problems is to discretize the Radiosity Equation by breaking the surfaces in the scene into small patches. Since radiosity is approximately constant over a small patch, we shall be able to replace integrals by sums and continuous functions by discrete values. The integral equation then reduces to a large system of linear equations which we shall be able to solve by numerical methods.

To discretize the Radiosity Equation, we begin by breaking the surfaces S into small patches P_j , $j = 1, \dots, N$. Over a small patch P_j , the radiosity $B(y)$ is approximately a constant B_j , and the integral over S breaks up into a sum of integrals over the patches P_j . Therefore by Equation (2.2)

$$\begin{aligned} B(x) &= E(x) + \rho_d(x) \int_S B(y) \frac{\cos \theta \cos \theta'}{\pi r^2(x, y)} V(x, y) dy \\ &\approx E(x) + \rho_d(x) \sum_{j=1}^N B_j \int_{P_j} \frac{\cos \theta \cos \theta'}{\pi r^2(x, y)} V(x, y) dA_j. \end{aligned} \tag{2.5}$$

We still need to discretize $B(x)$ on the left hand side and $E(x)$ on the right hand side of Equation (2.5). We can approximate the radiosity and energy of single patch P_i by an area weighted average. Thus

$$\begin{aligned} B_i &\approx (1/A_i) \int_{P_i} B(x) dA_i \\ E_i &\approx (1/A_i) \int_{P_i} E(x) dA_i. \end{aligned}$$

Integrating Equation (2.5) over the patch P_i and dividing by the patch area A_i yields

$$B_i = E_i + \rho_i \sum_{j=1}^N B_j (1/A_i) \int_{P_i} \int_{P_j} \frac{\cos\theta \cos\theta'}{\pi r^2(x,y)} V(x,y) dA_j dA_i.$$

Notice that the double integral

$$F_{ij} = (1/A_i) \int_{P_i} \int_{P_j} \frac{\cos\theta \cos\theta'}{\pi r^2(x,y)} V(x,y) dA_j dA_i \quad (2.6)$$

is independent of the radiosity, and depends only on the geometry of the facets. The parameters F_{ij} are called *form factors*; we shall have a lot more to say about these form factors shortly.

We have now reduced the Radiosity Equation to the following discrete form.

Radiosity Equations -- Discrete Form

$$B_i = E_i + \rho_i \sum_{j=1}^N F_{ij} B_j \quad i = 1, \dots, N \quad (2.7)$$

where

B_i is the radiosity of patch P_i , which we identify with intensity.

E_i is the energy emitted from patch P_i (uniform in all directions, since by assumption we are dealing only with diffuse emitters).

F_{ij} are the form factors, which depend only on the geometry of the scene and are independent of the lighting.

ρ_i is the diffuse reflection coefficient for patch P_i -- $0 \leq \rho_i \leq 1$.

Notice, in particular, that

$$\sum_{j=1}^N F_{ij} B_j = \text{energy reaching patch } P_i \text{ from all the other patches}$$

$$\rho_i \sum_{j=1}^N F_{ij} B_j = \text{energy reflected from the patch } P_i.$$

The discrete form of the Radiosity Equations given in Equation (2.7) are the equations that we are going to solve for the radiosities B_i . These linear equations for B_i are typically the starting point for most discussions of radiosity. The energies E_i and the diffuse reflection coefficients ρ_i are usually specified by the user; the form factors F_{ij} depend on the geometry of the scene and must be computed. Therefore before we can solve the Radiosity Equations, we need to compute the form factors.

3. Form Factors

To gain a better understanding of the form factors, we are going to provide both a physical and a geometric interpretation for these constants. We begin with a physical interpretation.

Theorem 1: *The form factor F_{ij} is the fraction of the energy leaving patch P_i arriving at patch P_j .*

Proof: Let B_{ij} denote the radiosity transferred from P_i to P_j . From the Radiosity Equation,

$$B_{ij} = F_{ji}B_i.$$

Since radiosity is energy per unit area, to find the total energy transferred from P_i to P_j , we must multiply the radiosity transferred from P_i to P_j by the area of P_j . Let A_i be the area of patch P_i , and let A_j be the area of patch P_j . Then the total energy transferred from P_i to P_j is

$$A_jB_{ij} = A_jF_{ji}B_i.$$

But by Equation (2.6)

$$A_jF_{ji} = \int_{P_i} \int_{P_j} \frac{\cos\theta\cos\theta'}{\pi r^2(x,y)} V(x,y) dA_j dA_i = A_iF_{ij}.$$


Therefore

$$A_jB_{ij} = A_iB_iF_{ij},$$

so

$$F_{ij} = \frac{A_jB_{ij}}{A_iB_i}. \tag{3.1}$$

Since the radiosity B_i is the energy radiated in all directions from P_i per unit area, the denominator A_iB_i represents the total energy radiated from P_i . But we have already seen that the numerator A_jB_{ij} represents the total energy transferred from P_i to P_j . Therefore, it follows from Equation

(3.1) that the form factor F_{ij} is the fraction of the energy that leaves P_i and arrives at P_j . 

Corollary 1: *For each value of i , the form factors F_{ij} form a partition of unity. That is,*

$$\sum_j F_{ij} = 1. \tag{3.2}$$

Proof: This result follows from conservation of energy. By Theorem 1 the form factor F_{ij} is the fraction of the energy leaving patch P_i arriving at patch P_j . Since energy is conserved, the energy leaving patch P_i must arrive somewhere -- that is, at one of the patches P_j . Therefore

$$\sum_j F_{ij} = 1. \quad \img alt="hand icon" data-bbox="374 854 417 880"/>$$

To provide a geometric interpretation for the form factors, we must first discretize still further. Over a small patch P_i , we can treat the inner integral as roughly constant, so

$$F_{ij} = (1 / A_i) \int_{P_i} \int_{P_j} \frac{\cos\theta \cos\theta'}{\pi r^2(x,y)} V(x,y) dA_j dA_i \approx \int_{P_j} \frac{\cos\theta \cos\theta'}{\pi r^2(x,y)} V(x,y) dA_j. \quad (3.3)$$

We shall now investigate the meaning of the differential $\frac{\cos\theta \cos\theta'}{\pi r^2(x,y)} dA_j$.

The product $\frac{\cos\theta'}{r^2} dA_j$ is the projection of the differential area dA_j onto the unit hemisphere centered at the patch dA_i . Similarly, the product $(\cos\theta) \left(\frac{\cos\theta'}{r^2} dA_j \right)$ is the projection of the differential area $\frac{\cos\theta'}{r^2} dA_j$ onto the base of the hemisphere centered at dA_i , the plane perpendicular to the normal of dA_i (see Figure 2).

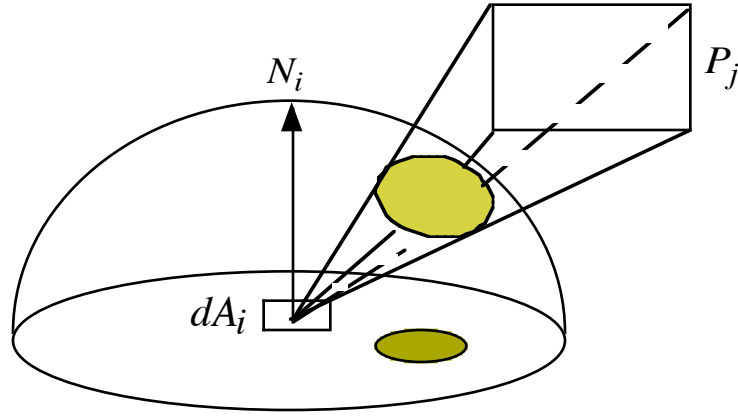


Figure 2: Projecting a patch P_j first onto the unit hemisphere centered at the patch dA_i and then onto the base of the hemisphere centered at dA_i .

In 3-dimensions the cosine terms in these projections are not so easy to visualize, so to get a feel for these cosine terms, let us look instead in 2-dimensions. To project a differential length dL onto a circle of radius r , where r is the distance between the patches, we multiply dL by $\cos(\theta')$ where θ' is the angle between the tangent to the circle of radius r and the tangent to the curve dL (see Figure 3a). Of course, the angle between the tangents is the same as the angle between the normals. If we think of dL as one patch and if the second patch is located at the center of the circle

of radius r , then the angle between the normals is the same as the angle θ' between the normal to dL and the vector between the patches. Furthermore, to project the circle of radius r onto the unit circle, we simply divide by r . Thus to project a differential length dL on one patch onto the unit circle centered at the other patch, we multiply dL by $\cos\theta' / r$. (In 3-dimensions, we must scale uniformly in two directions, so the scale factor r is replaced by r^2 .)

Now to project from the unit circle to the x -axis, we must multiply by $\cos(\theta)$, where θ is the angle between the tangent vectors to the patch at the origin and the patch along the unit circle. Since the x -axis represents the plane of the first patch, the angle θ is also the angle between the normal vector to the first patch and the vector between the two patches (see Figure 3b). Therefore, the product $\frac{\cos\theta\cos\theta'}{r}dL_j$ is the product of the projection of dL_j , first onto the unit circle and then

onto the x -axis. Essentially the same analysis holds in 3-dimensions. Notice by the way that the factor of π in the denominator of the integrand is simply the area of the unit circle at the base of the hemisphere. Hence $\frac{\cos\theta\cos\theta'}{r^2}dA_j$ is the projection of dA_j first onto the unit sphere centered at

dA_i and then onto the hemispherical plane of dA_i . Integrating $\frac{\cos\theta\cos\theta'}{r^2}dA_j$ over the patch P_j yields to the following results.

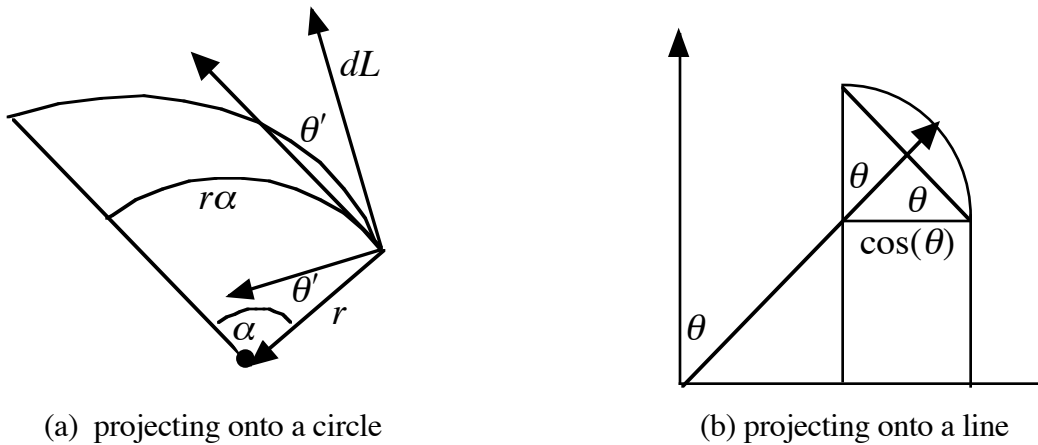


Figure 3: (a) Projecting a differential length onto a circle is equivalent to multiplying by $\cos(\theta')$, and (b) projecting a differential length onto a line is equivalent to multiplying by $\cos(\theta)$.

Theorem 2: The form factor F_{ij} is, up to division by π , the projection of the area of P_j first onto the unit hemisphere centered at dA_i and then onto the hemispherical plane of dA_i .

Corollary 2: Two surfaces $P_j, P_{j'}$ with the same projection onto the unit hemisphere centered at a small patch dA_i have the same form factor -- that is, $F_{ij} \approx F_{ij'}$.

Corollary 2 is the main result of all this analysis: two patches with equal projection on the unit hemisphere centered around a small patch dA_i will necessarily have the same form factor F_{ij} . We can use this insight to compute the form factors once and for all for some simple surface and then find the form factors for arbitrary surfaces by projecting onto the known surface.

3.1 Hemi-Cubes. A hemi-cube is the upper half of a cube with sides of length two, centered at a small patch P_i (see Figure 4). Here we are going to compute explicit formulas for the form factors for small patches on the surface of the hemi-cube. To find the form factors for arbitrary patches, we will project these patches onto the hemi-cube and use Corollary 2 from the previous section which says that patches with equal projections have equal form factors.

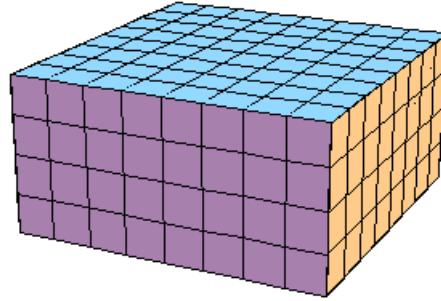


Figure 4: A hemi-cube. The lengths of the sides are twice the height of the hemi-cube.

To compute form factors for small patches on the surface of the hemi-cube, recall from Equation (3.3) that in general the form factor is given by

$$F_{di,j} = \int_{P_j} \frac{\cos\theta \cos\theta'}{\pi r^2(x,y)} V(x,y) dA_j.$$

For a very small patch with area ΔA_j , the integrand can be approximated by a constant, so

$$F_{di,dj} = \frac{\cos\theta \cos\theta'}{\pi r^2} \Delta A_j.$$

To find an explicit formula for the form factor $F_{di,dj}$, we need to find explicit formulas for $\cos\theta$, $\cos\theta'$, and r , where

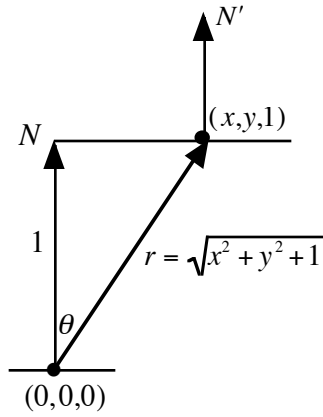
- θ is the angle between the normal N to the patch P_i and the vector from the center of P_i to the center of P_j ;
- θ' is the angle between the normal N' to the patch P_j and the vector from the center of P_i to the center of P_j ;
- r is the distance from the center of P_i to the center of P_j .

For the hemi-cube there are two kinds of patches to consider: patches on the top of the hemi-cube and patches along the sides of the hemi-cube. To compute the form factors for these patches, choose a coordinate system with the origin at the center of the hemi-cube and the z -axis aligned with the normal to the patch P_i at the center of the hemi-cube. For a small patch on the top face of the hemi-cube centered at the point $P = (x, y, 1)$, it follows from simple trigonometry (see Figure 5a) that

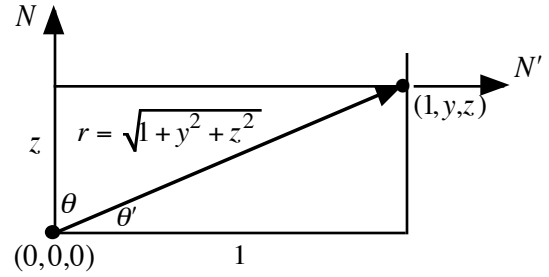
$$\cos\theta = \cos\theta' = 1/r.$$

Therefore for small patches with area ΔA_j on the top face of the hemi-cube centered at the point $P = (x, y, 1)$,

$$F_{di,dj} = \frac{\cos\theta\cos\theta'}{\pi r^2}\Delta A_j = \frac{\Delta A_j}{\pi r^4} = \frac{\Delta A_j}{\pi(x^2 + y^2 + 1)^2}.$$



(a) $\cos\theta = \cos\theta' = 1/r$



(b) $\cos\theta = z/r$ $\cos\theta' = 1/r$

Figure 5: Schematic views of the top and side faces of a hemi-cube. (a) For the top face of the hemi-cube, the vectors N, N' are parallel. Therefore by simple trigonometry, $\cos\theta = \cos\theta' = 1/r$. (b) For the side faces of the hemi-cube, the vectors N, N' are orthogonal. Therefore by simple trigonometry, $\cos\theta = z/r$ and $\cos\theta' = 1/r$.

Similarly, for a small patch on the side face of the hemi-cube parallel to the yz -plane centered at the point $P = (\pm 1, y, z)$, it again follows by simple trigonometry (see Figure 5b) that

$$\cos\theta = z/r \quad \cos\theta' = 1/r.$$

Therefore for small patches on side faces of the hemi-cube parallel to the yz -plane

$$F_{di,dj} = \frac{\cos\theta\cos\theta'}{\pi r^2}\Delta A_j = \frac{z\Delta A_j}{\pi r^4} = \frac{z\Delta A_j}{\pi(y^2 + z^2 + 1)^2}.$$

Finally, for a small patch on the side face of the hemi-cube parallel to the xz -plane centered at the point $P = (x, \pm 1, z)$, it follows by an analogous argument that

$$\cos\theta = z/r \quad \cos\theta' = 1/r$$

$$F_{di,dj} = \frac{\cos \theta \cos \theta'}{\pi r^2} \Delta A_j = \frac{z \Delta A_j}{\pi r^4} = \frac{z \Delta A_j}{\pi(x^2 + z^2 + 1)^2}.$$

Thus we have explicit formulas for the form factors for all the patches on the hemi-cube.

Once we have the form factors for the hemi-cube surrounding each patch, we can compute the form factors for the patches using the following algorithm.

Form Factor Algorithm

Compute the form factors for each cell of each hemi-cube, and store all of these form factors.

To find the form factor F_{ij} for a patch P_j relative to the patch P_i ,

For each face of the hemi-cube surrounding P_i :

- i. Clip the scene to the frustum determined by the center of P_i and the face of the hemi-cube. {See Lecture 14, Section 4.2 -- Pseudoperspective: frustum→box}
- ii. For each cell of the hemi-cube face:
 - Find the nearest polygon in the scene. {Apply a z -buffer algorithm -- see Lecture 22, Section 3.}
 - Label each cell with the closest polygon.
- iii. For each polygon P_j sum the form factors of the hemi-cube cells labeled j :

$$F_{di,j} = \sum_{q=j} F_q.$$

The calculation of form factors takes most of the time in the computation of radiosity. We can reduce this computation almost by half using the following identity.

Reciprocity Relationship

$$A_i F_{ij} = A_j F_{ji} \tag{3.4}$$

By the reciprocity relationship, once we calculate F_{ij} , we can compute F_{ji} with very little additional work. The reciprocity relationship holds because by Equation (2.6)

$$A_i F_{ij} = \int_{P_i} \int_{P_j} \frac{\cos \theta \cos \theta'}{\pi r^2} V(x, y) dA_j dA_i = A_j F_{ji}.$$

Note that we have already used this reciprocity relationship in the proof of Theorem 1. To use the reciprocity relationship to compute the form factors, we need to know the areas A_i, A_j . For planar polygons these areas are easy to compute. By Newell's Formula (Lecture 11, Section 4.2.2) if P_0, \dots, P_n are the vertices of a planar polygon P , then

$$Area(P) = (1/2) \sum_{j=1}^n |P_j \times P_{j+1}|.$$

4. The Radiosity Rendering Algorithm

We introduced radiosity in order to develop more realistic looking images using Computer Graphics. Now let us put together what we now know about the Radiosity Equations, form factors, hemi-cubes, and shading to develop a rendering algorithm based on radiosity.

Radiosity Rendering Algorithm

1. *Mesh the surfaces.*
Break each surface into small surface patches.
2. *Compute the form factors for each pair of surface patches.*
Use the hemi-cube algorithm.
3. *Solve the linear system (2.7) for the radiosities.*

$$B_i = E_i + \rho_i \sum_{j=1}^N F_{ij} B_j \quad i = 1, \dots, N.$$

(See below.)

4. *Compute the radiosity at the vertices of the patches.* (See below.)
5. *Pick a viewpoint.*
6. *Determine which surfaces are visible.*
Use any hidden surface algorithm.
7. *Apply Gouraud shading to the visible surfaces.*

The only steps that require further elaboration are step 3 and step 4. In step 3 we must solve a large system of linear equations. For large linear systems, standard techniques like Gaussian elimination are slow and unstable. We shall provide instead two alternative robust numerical methods for solving these equations, but we defer this discussion till the next section. In step 4 we need to find the radiosity for each vertex so that we can perform Gouraud shading in step 7 in order to eliminate discontinuities in intensity between adjacent patches.

For regular meshes consisting of rectangular patches, there are three kinds of vertices: interior, boundary, and corner (see Figure 6). For interior vertices, it is natural simply to set the intensity to the average of the radiosities at the four adjacent patches:

$$I_{interior} = \frac{B_1 + B_2 + B_3 + B_4}{4}.$$

For edges and corners there are two competing strategies: either set

$$I_{edge} = \frac{B_1 + B_2}{2}$$

$$I_{corner} = B_1,$$

or set

$$\frac{I_{edge} + I_{interior}}{2} = \frac{B_1 + B_2}{2}$$

$$\frac{I_{corner} + I_{interior}}{2} = B_1$$

so that

$$I_{edge} = B_1 + B_2 - I_{interior} = \frac{3B_1 + 3B_2 - B_3 - B_4}{4}$$

$$I_{corner} = 2B_1 - I_{interior} = \frac{7B_1 - B_2 - B_3 - B_4}{4}.$$

Both strategies yield reasonable results for Gouraud shading.

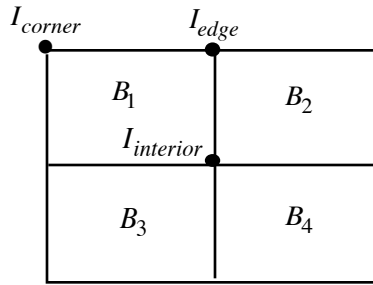


Figure 6: The intensities at the vertices depend on the radiosities of the adjacent patches. There are three kinds of vertices: interior vertices, edge vertices, and corner vertices. Each vertex type has a slightly different formula for intensity based on the radiosities of the adjacent patches.

5. Solving the Radiosity Equations

The Radiosity Equations are:

$$B_i = E_i + \rho_i \sum_{j=1}^N F_{ij} B_j \quad i = 1, \dots, N.$$

Bringing all the radiosities to the left hand side yields

$$\sum_{j=1}^N (\delta_{ij} - \rho_i F_{ij}) B_j = E_i \quad i = 1, \dots, N,$$

or equivalently

$$\begin{aligned} (1 - \rho_1 F_{11}) B_1 - \rho_1 F_{12} B_2 - \dots - \rho_1 F_{1n} B_n &= E_1 \\ -\rho_2 F_{21} B_1 + (1 - \rho_2 F_{22}) B_2 - \dots - \rho_2 F_{2n} B_n &= E_2 \\ \vdots & \\ -\rho_n F_{n1} B_1 - \rho_n F_{n2} B_2 - \dots + (1 - \rho_n F_{nn}) B_n &= E_n. \end{aligned}$$

We can rewrite these equations in matrix form as:

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

Our goal is to solve for the unknown radiosities $B = (B_1, \dots, B_n)^T$. We shall investigate two methods for solving these equations: gathering and shooting. Both methods take roughly the same total amount of time, but shooting facilitates faster viewing of intermediate solutions.

5.1 Gathering. To solve a large system of linear equations, we can employ standard relaxation techniques. Relaxation techniques are fixed point methods for linear equations. We studied these fixed point methods in Lecture 7, Section 3.2; we shall now briefly review these methods.

Suppose that

$$\sum_{j=1}^n M_{ij} B_j = E_i \quad i = 1, \dots, N. \quad (5.1)$$

Solving for B_i yields

$$B_i = \frac{E_i}{M_{ii}} - \sum_{j \neq i} \frac{M_{ij}}{M_{ii}} B_j. \quad i = 1, \dots, N.$$

In relaxation methods, we start with an initial guess

$$B^0 = \begin{pmatrix} B_1^0 \\ B_2^0 \\ \vdots \\ B_n^0 \end{pmatrix}.$$

Usually either

$$B^0 = \begin{pmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{pmatrix} \quad \text{or} \quad B^0 = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

but since we are using a fixed point method, typically we can choose any value for B^0 . We then apply a relaxation technique to compute the next guess B^p from the previous guess B^{p-1} . Under certain simple conditions (see below), these relaxation methods are guaranteed to converge to the solution of the linear system. The two most common relaxation methods are due to Jacobi and to Gauss-Seidel.

Jacobi Relaxation

$$B_i^p = \frac{E_i}{M_{ii}} - \sum_{j \neq i} \frac{M_{ij}}{M_{ii}} B_j^{p-1} \quad i = 1, \dots, N$$

Gauss-Seidel Relaxation

$$B_i^p = \frac{E_i}{M_{ii}} - \sum_{j=1}^{i-1} \frac{M_{ij}}{M_{ii}} B_j^p - \sum_{j=i+1}^N \frac{M_{ij}}{M_{ii}} B_j^{p-1} \quad i = 1, \dots, N$$

In Jacobi relaxation we use the values of the guess at the previous level to generate the values of the guess at the next level. In Gauss-Seidel relaxation, we use the values of the guess at the previous level along with the values of the guess already computed at the current level to compute the next value of the guess at the current level. Gauss-Seidel relaxation is a bit more complicated than Jacobi relaxation, but Gauss-Seidel relaxation typically converges faster to the solution of the linear system. (For additional details, see Lecture 7, Section 3.2.)

Convergence is guaranteed in both relaxation methods for any initial guess when the system is diagonally dominant. A system of equations such as (5.1) is *diagonally dominant* if

$$|M_{ii}| \geq \sum_{j \neq i} |M_{ij}|.$$

It follows easily from Corollary 1 that the Radiosity Equations -- Equations (2.7) -- are diagonally dominant (see Exercise 1), so we can use these relaxation methods to solve the Radiosity Equations.

Solving for the radiosity using relaxation methods is called *gathering* because we gather the radiosity from all the surfaces simultaneously. The disadvantage of gathering is that we must compute all the form factors to get an intermediate result. Thus we must solve for all the hemi-cubes radiosities before we can begin to render the scene. Typically solving for all the hemi-cube radiosities and computing all the form factors takes a long time, so if there is some error in the scene or in the code we will have to wait a long time to detect the mistake. Therefore we shall consider another technique called *shooting*, where we can render intermediate results progressively without waiting to compute all the form factors and all the radiosities for each hemi-cube.

5.2 Shooting -- Progressive Refinement. Each patch contributes to the radiosity of every other patch. To compute radiosity progressively, we fix one particular patch and compute the radiosity of every other patch due to the radiosity of the fixed patch. We can then display the scene and repeat the process by choosing another patch. In this way we get to see intermediate results quickly without waiting to finish the entire computation.

From the Radiosity Equations

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

so the radiosity B_i due to B_j is $\rho_i F_{ij} B_j$. To solve the Radiosity Equations by gathering, we need to compute all the form factors for every patch; thus we need one hemi-cube for each patch to initiate the computation.

But in shooting, we are interested initially only in the radiosity B_j due to one fixed radiosity B_i . The radiosity B_j due to B_i is $\rho_j F_{ji} B_i$. To compute the form factors F_{ji} directly for each j , we would still need to introduce one hemi-cube computation for each patch. But recall that by the reciprocity relationships (Equation (3.4))

$$A_i F_{ij} = A_j F_{ji};$$

therefore

$$B_j \text{ due to } B_i = \rho_j F_{ji} B_i = \rho_j F_{ij} B_i \frac{A_i}{A_j}.$$

Now to compute F_{ij} for each j , we need the form factors F_{ij} only for the single patch P_i in order to update all the patches P_j ! Thus we need to introduce only one hemi-cube at a time to update all the patch radiosities at once. This reduction can save a lot of time.

In shooting, we think of radiosity as accumulating on each patch until we shoot this radiosity out to all the other patches. Let ΔB_j denote the unshot radiosity of the patch P_j . Then from the Radiosity Equations it follows that after we shoot the accumulated radiosity ΔB_i from the patch P_i :

$$\begin{aligned} \Delta B_j &= (\Delta B_j)_{old} + \rho_j F_{ij} \Delta B_i \frac{A_i}{A_j} \\ B_j &= (B_j)_{old} + \rho_j F_{ij} \Delta B_i \frac{A_i}{A_j} \\ \Delta B_i &= 0. \end{aligned} \tag{5.2}$$

These equations lead to the following progressive refinement procedure.

Shooting Algorithm

1. *Initialize the radiosities.*

$$B^0 = \begin{pmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{pmatrix} = \Delta B^0.$$

2. Repeat until $\Delta B_j = 0$ for all j :
 - a. Select the patch P_i for which the total power $A_i \Delta B_i$ is maximal.
 - b. Compute the form factors F_{ij} , using the hemi-cube computation for the patch P_i .
 - c. Update B_j and ΔB_j for all the patches P_j , using Equation (5.2).
 - d. Display the scene using the current radiosity values B_j .

At the start of the shooting algorithm most of the radiosity is still unshot. Therefore initially the scene will appear quite dark. To get brighter pictures, we shall introduce an ambient correction term to account for the unshot radiosity. This correction term is for display only; the correction should not be added to the actual radiosities in the execution of the shooting algorithm.

To account for the ambient light due to unshot radiosity, we will first compute an average reflection coefficient and an average unshot radiosity. We will then add this average ambient reflection term to the radiosity of each patch.

We introduce an average diffuse reflection coefficient by taking an area weighted average of all the diffuse reflection coefficients:

$$\rho_{av} = \frac{\sum_{i=1}^N \rho_i A_i}{\sum_{i=1}^N A_i}.$$

We weight by area because larger patches will reflect more light. Next we compute the total amount of reflection R by summing all the recursive reflections for the light bouncing repeatedly off all the patches:

$$R = 1 + \rho_{av} + \rho_{av}^2 + \dots = \frac{1}{1 - \rho_{av}}.$$

The average unshot radiosity ΔB_{av} is the average of all the unshot radiosity weighted by area:

$$\Delta B_{av} = \frac{\sum_{i=1}^N \Delta B_i A_i}{\sum_{i=1}^N A_i}.$$

Finally, the ambient light A is the reflection of all the unshot light. Thus

$$A = R \Delta B_{av}.$$

For the purposes of display only, we add this ambient correction term to the radiosity of each patch, so that when we display the scene we set

$$B_i = B_i + \rho_i A.$$

Although shooting allows us to view intermediate stages of the scene without computing all the form factors, in the end shooting is no faster than gathering. Ultimately both gathering and shooting need to introduce the same number of hemi-cubes to find all the form factors for all the patches.

6. Summary

Radiosity is a classical energy transfer technique adapted to rendering surfaces in Computer Graphics. The advantages of radiosity over other standard rendering methods such as recursive ray tracing are that radiosity provides better photorealistic effects such as softer shadows and color bleeding. Radiosity computations are also view independent, since typically radiosity models only ambient light and diffuse reflection.

The main disadvantages of radiosity are that radiosity computations are expensive both in time and in space. Good accuracy demands lots of small patches to model curved surfaces, leading to lots of form factors and lots of hemi-cubes as well as a very large system of linear equations for the radiosities. In addition radiosity does not typically model specular reflections, since specular reflections are view dependent.

Enhancements to standard radiosity include a two pass algorithm for computing specular reflections, more realistic light sources, participating mediums such as atmospheric fog, adaptive mesh generation for computing more accurate form factors, and finite element methods or wavelets for calculating more accurate approximations to radiosity. The interested reader can find these subjects in the literature; we shall not pursue these topics here.

Below for easy reference we review the Rendering Equation along with the continuous and discrete forms of the Radiosity Equation and the form factors.

Rendering Equation

$$I(x, x') = E(x, x') + \int_S \rho(x, x', x'') I(x', x'') dx'' \quad (2.1)$$

where

$I(x, x')$ is the total energy passing from x' to x .

$E(x, x')$ is the energy emitted directly from x' to x .

$\rho(x, x', x'')$ is the reflection coefficient -- the percentage of the energy transferred from x'' to x' that is passed on to x .

Radiosity Equation -- Continuous Form

$$B(x) = E(x) + \rho_d(x) \int_S B(y) \frac{\cos\theta \cos\theta'}{\pi r^2(x, y)} V(x, y) dy \quad (2.2)$$

where

$B(x)$ is the radiosity at the point x , which we identify with the intensity or energy -- that is, the total power leaving a surface/unit area/solid angle.

$E(x)$ is the energy emitted directly from a point x . This energy is uniform in all

directions, since we are assuming that the scene has only diffuse emitters.

$\rho_d(x)$ is the diffuse reflection coefficient -- $0 \leq \rho_d(x) \leq 1$.

$V(x, y)$ is the visibility term:

$V(x, y) = 0$ if x is not visible from y .

$V(x, y) = 1$ if x is visible from y .

θ = angle between the surface normal (N) at x and the light ray (L) to y .

θ' = angle between surface normal (N') at y and the light ray (L) to x .

$r(x, y)$ = distance from x to y .

Radiosity Equations -- Discrete Form

$$B_i = E_i + \rho_i \sum_{j=1}^N F_{ij} B_j \quad i = 1, \dots, N \quad (2.7)$$

where

B_i is the radiosity of patch P_i , which we identify with intensity.

E_i is the energy emitted from patch P_i (uniform in all directions, since by assumption we are dealing only with diffuse emitters).

F_{ij} are the form factors, which depend only on the geometry of the scene and are independent of the lighting.

ρ_i is the diffuse reflection coefficient for patch P_i -- $0 \leq \rho_i \leq 1$.

Form Factors

$$F_{ij} = (1 / A_i) \int_{P_i} \int_{P_j} \frac{\cos \theta \cos \theta'}{\pi r^2(x, y)} V(x, y) dA_j dA_i \quad (2.6)$$

Exercises:

1. Prove that the Radiosity Equations are diagonally dominant.
2. Using Theorem 1, show that $F_{i, j \cup k} = F_{i, j} + F_{i, k}$
3. Let $M = (M_{ij})$, $B = (B_1, \dots, B_N)^T$, $E = (E_1, \dots, E_N)^T$, and let D be the diagonal matrix defined by

$$\begin{aligned} D_{ij} &= M_{ii} & i &= j \\ &= 0 & i &\neq j. \end{aligned}$$

Consider the system of linear equations:

$$\begin{aligned}
M_{11}B_1 + M_{12}B_2 + \cdots + M_{1n}B_n &= E_1 \\
M_{21}B_1 + M_{22}B_2 + \cdots + M_{2n}B_n &= E_2 \\
&\vdots & \vdots \\
M_{n1}B_1 + M_{n2}B_2 + \cdots + M_{nn}B_n &= E_n.
\end{aligned}
\quad \Leftrightarrow \quad M * B = E$$

a. Show that this system is equivalent to:

$$\begin{aligned}
M_{11}B_1 &= E_1 - M_{12}B_2 - \cdots - M_{1n}B_n \\
M_{22}B_2 &= E_2 - M_{21}B_1 - \cdots - M_{2n}B_n \\
&\vdots & \vdots & \vdots \\
M_{nn}B_n &= E_n - M_{n1}B_1 - \cdots
\end{aligned}
\quad \Leftrightarrow \quad D * B = E - (M - D) * B$$

b. Suppose that $M_{ii} \neq 0$ for $i = 1, \dots, N$. Show that the system in part a is equivalent to:

$$\begin{aligned}
B_1 &= \frac{E_1}{M_{11}} - \frac{M_{12}}{M_{11}}B_2 - \cdots - \frac{M_{1n}}{M_{11}}B_n \\
B_2 &= \frac{E_2}{M_{22}} - \frac{M_{21}}{M_{22}}B_1 - \cdots - \frac{M_{2n}}{M_{22}}B_n \\
&\vdots & \vdots & \vdots \\
B_n &= \frac{E_n}{M_{nn}} - \frac{M_{n1}}{M_{nn}}B_1 - \cdots
\end{aligned}
\quad \Leftrightarrow \quad B = D^{-1} * E - (D^{-1} * M - I) * B$$

c. Show that:

$$\begin{aligned}
D_{ij}^{-1} &= \frac{1}{M_{ii}} & i = j \\
&= 0 & i \neq j
\end{aligned}$$

$$D^{-1} * M - I = \begin{pmatrix} 0 & \frac{M_{12}}{M_{11}} & \cdots & \frac{M_{1n}}{M_{11}} \\ \frac{M_{21}}{M_{22}} & 0 & \cdots & \frac{M_{2n}}{M_{22}} \\ \vdots & \ddots & \ddots & \vdots \\ \frac{M_{n1}}{M_{nn}} & \cdots & \frac{M_{n,n-1}}{M_{nn}} & 0 \end{pmatrix}.$$

d. Let $T(B) = D^{-1} * E - (D^{-1} * M - I) * B$. Show that Jacobi relaxation is equivalent to iterating the transformation $T(B)$.

e. Conclude from part d that Jacobi relaxation is equivalent to finding a fixed point of the transformation $T(B)$.

4. Let $M = (M_{ij})$, $B = (B_1, \dots, B_N)^T$, $E = (E_1, \dots, E_N)^T$, and let L be the lower triangular matrix defined by:

$$\begin{aligned} L_{ij} &= M_{ij} & i \geq j \\ &= 0 & i < j. \end{aligned}$$

Consider the system of linear equations:

$$\begin{aligned} M_{11}B_1 + M_{12}B_2 + \dots + M_{1n}B_n &= E_1 \\ M_{21}B_1 + M_{22}B_2 + \dots + M_{2n}B_n &= E_2 \\ &\vdots & \vdots \\ M_{n1}B_1 + M_{n2}B_2 + \dots + M_{nn}B_n &= E_n. \end{aligned} \Leftrightarrow M * B = E$$

a. Show that this system is equivalent to:

$$\begin{aligned} M_{11}B_1 &= E_1 - M_{12}B_2 - \dots - M_{1n}B_n \\ M_{21}B_1 + M_{22}B_2 &= E_2 - M_{23}B_3 - \dots - M_{2n}B_n \\ &\vdots & \vdots \\ M_{n1}B_1 + \dots + M_{nn}B_n &= E_n \end{aligned} \Leftrightarrow L * B = E - (M - L) * B$$

b. Suppose that $M_{ii} \neq 0$ for $i = 1, \dots, N$. Show that the system in part *a* is equivalent to $B = L^{-1} * E - (L^{-1} * M - I) * B$.

c. Show that the matrix L^{-1} -- and hence the matrix $L^{-1} * M - I$ -- is easy to compute. In particular, show that L^{-1} is lower triangular and that each entry can be computed by solving one linear equation in one unknown.

d. Let $T(B) = L^{-1} * E - (L^{-1} * M - I) * B$. Show that Gauss-Seidel relaxation is equivalent to iterating the transformation $T(B)$.

e. Conclude from part *d* that Gauss-Seidel relaxation is equivalent to finding a fixed point of the transformation $T(B)$.

Programming Project:

1. Implement radiosity in your favorite programming language using your favorite API.
 - a. Use your implementation to render several different scenes.
 - Apply both gathering and shooting.
 - b. Compare the scenes rendered by radiosity with the same scenes rendered using recursive ray tracing.