GLOBAL ILLUMINATION

Many computer-aided design applications require a method to visualize the appearance of a final product without going to the expense of building a physical model. An architect may want to preview the appearance of several different lighting systems in a building being designed. A car designer may want to evaluate the visual effect of different types of paints on a particular car body. A safety engineer may want to evaluate whether illuminated exit signs will be visible in the event of a fire. In each case the user has a numerical description of an object, and needs to produce a realistic image of the object in use after it is built. Generating a realistic image from a numerical description requires a simulation of the global illumination of the scene.

Global illumination methods attempt to account for all the possible paths that light may take from light sources through the environment to the viewer of a scene. Accounting for the true behavior of light in an environment differentiates realistic synthesis from artistic renderings or diagrams of an environment. Aristic rendering relies on the artist's past experience to determine the colors and shades used to present the appearance of an object. Images rendered using global illumination simulations rely on the accuracy of the numerical descriptions and a model of light propagation to determine the colors and shades.

The numerical description includes the geometry of objects and their reflectance and transmittance. Figure 1 shows the process of forming an image. Viewpoint, view direction, image plane, and image resolution are specified, and the object visible through each pixel is determined. For an image to be a realistic portrayal of the scene, the color values of each pixel must be determined by the quantity and spectral distribution of the light that would arrive at the viewer of the real physical scene from the same direction. While the actual quantity and spectral distribution of light cannot be reproduced on the display device, a color metamer that will produce the same impression on the user can be computed for each pixel.

FUNDAMENTAL COMPONENTS OF GLOBAL ILLUMINATION

Simulating the global illumination of an object requires accounting for the direct illumination from light sources, the occlusion of direct illumination by other objects, indirect illumination from other objects, and the effects of attenuation and scattering of light by volumes of matter in the environment. Simulations of global illumination must account for all of these effects working together. For example, an object could

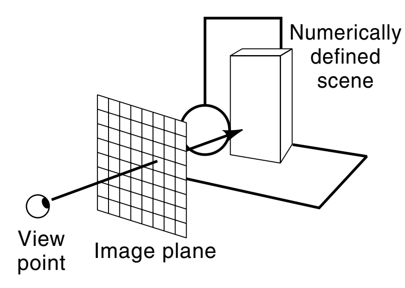

Figure 1. An image is formed by selecting a viewpoint, direction, and image resolution, and then determining the surface visible through each image pixel.

J. Webster (ed.), *Wiley Encyclopedia of Electrical and Electronics Engineering*. Copyright © 1999 John Wiley & Sons, Inc.

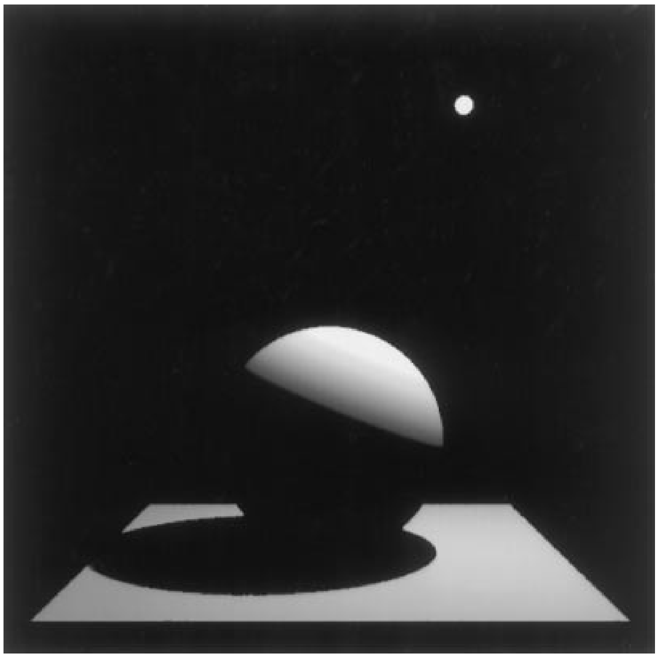

Figure 2. A small spherical light source illuminates a sphere sitting on a plane. The sphere casts a sharp shadow on the plane.

be illuminated by a complicated path starting at a light source, passing through a cloud of smoke, reflecting off a mirror, and then striking an object that is visible in the image.

Direct Illumination

Direct illumination requires an accurate numerical description of the initial source of light. The light source may be natural—the sun or sky, or manufactured—a lamp. The form of the numerical description required depends on the relative sizes and locations of the light sources and objects in the environment. If the source is small relative to the distance to the object it illuminates, it can be numerically modeled as a point. The distribution of the light emitted as a function of direction must be specified. If the object illuminated is also small compared to the distance to the source, as in the case of the sun illuminating a flower, the model may simply specify the direction and quantity of light.

A light source that is long in one dimension may be specified as a line source, or in two dimensions as an extended area source. In each case, directional distributions are needed as well. These may be obtained by measurement, or from manufacturers' descriptions for light fixtures.

The quantity of light reflected from a surface in the direction of the eye is computed using the geometry of the surface and its bidirectional reflectance distribution function (BRDF). The BRDF gives the quantity of reflected light in terms of the incident direction of light and the reflected direction, where directions are measured from the surface normal.

Shadows

Objects that do not have direct view of the light source are in shadow. Shadows are classified as *attached* and *cast.* An attached shadow occurs when the surface element is directed away from the light source. This depends only on the geometry of the object and the light source. A cast shadow occurs when a second object blocks the view of the source from the surface element. Figure 2 shows an example of attached and cast shadows. A small spherical light source shines on a large sphere sitting on a plane. The top of the sphere receives direct illumination. The underside of the sphere is in an attached shadow. There is a circular cast shadow on the plane at points where the plane's view of the light source is obstructed by the sphere.

Shadows are an essential cue for depth and location in an image. Any realistic image must estimate shadows. In scenes where the light source is adequately represented as a point, any location in the scene is either visible or invisible to the light source. All cast shadows are sharp. The shadowed area is referred to as the umbra.

There are fundamentally two approaches to computing which points are in shadow. One approach is to go to each point on each object, and check if there are any objects on the line between the point and the light source. This is most commonly done by ray casting. In ray casting, the intersection of the line to the source with the other objects in the environment is explicitly calculated to see if they block the source. Many efficient methods have been developed for ray casting, so that not every other object in the environment has to be tested as a blocker.

The other approach is to go to each light source and determine which surfaces are visible from the light. One example of a method to compute shadows from the light source is to compute shadow volumes, as described by Crow (1). Shadow volumes are semi-infinite volumes with the source as one vertex, and the sides of one of the surfaces defining the sides of the volume. Everything behind the surface within this volume is in shadow. A volume is formed for each surface in the environment to test whether it is casting shadows on other surfaces, unless it has already been found to be in the shadow volume of another surface.

Another example of finding shadows from the light source is to compute shadow maps, as introduced by Williams (2). In this approach, an image is computed with the light source as the viewpoint. The distance to each visible surface in this light source image is recorded. While the final image is being formed from the observer's viewpoint, shadows are determined at each visible point in the final image by determining whether the point is visible in the image from the light source.

In scenes with light sources that are not points, but lines or extended areas, shadows are not sharp. Figure 3 is the same as Fig. 2, with a larger light source. The cast shadow no longer has a sharp boundary, since some points on the plane have a partial view of the light source. Points in these regions form the penumbra. Shadows for extended sources can be computed by modifications of the techniques used for point sources. Shadow rays may be cast to many points on the source to estimate what part of the source is visible. Shadow volumes or shadow maps may be computed from many points on the source.

Interreflections

Often much of the light that illuminates an object does not come directly from a light source; instead, it arrives after being reflected or transmitted from other objects. Figure 4 shows some typical effects of interreflections. The scene in Fig. 4 is the same as in Fig. 2, except a wall has been added on the left side. The shadows that appeared in Fig. 1 are no

Figure 3. A large spherical light source illuminates the same sphere and plane as shown in Fig. 2. The shadow cast when the light source is larger is fuzzy, since some points on the plane have a partial view of the light source.

longer completely black. Light has been reflected from the wall into the shadowed areas.

The effect of interreflections and transmissions depends on the nature of the intermediate surfaces involved. Consider a white, diffuse (i.e., matte) surface. If light arrives at this surface after reflecting from a red surface, the incident light will be red, and the white surface itself will look slightly red. This effect is referred to as *color bleeding.* If the intermediate surface were white, the interreflection would simply increase the illumination of the surface. If the intermediate interreflection were from a specular (i.e., mirror-like) surface, the effect may be a caustic. For example, a curved mirror or a crystal sphere will cause the light to be focused into a small area, and the result is a bright spot (i.e., a caustic) on the target surface. The most time-consuming portion of global illumination solutions involves finding the most important paths that affect the illumination of a surface.

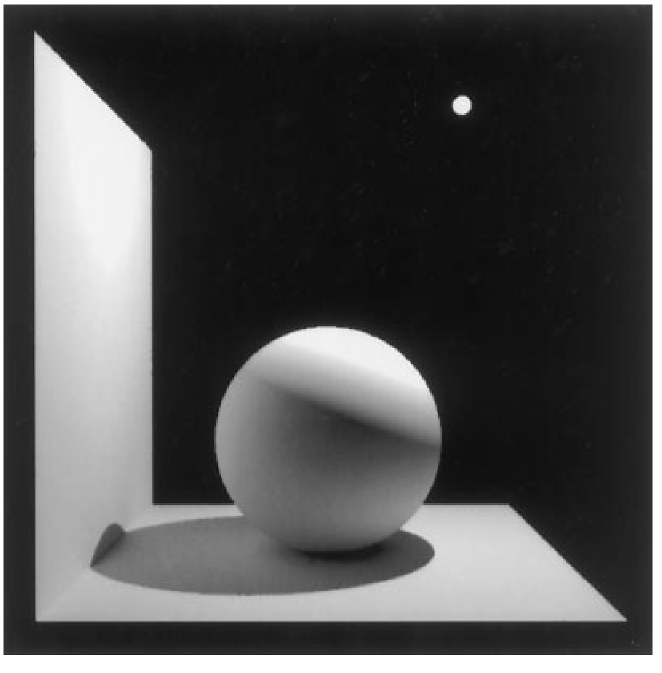

Figure 4. The same scene is shown as in Fig. 2, but a large wall has been added on the left. Light reflected from the wall illuminates shadowed areas on the sphere and plane.

One approach for computing the paths important to the illumination of objects in an image is ray tracing, developed by Whitted (3). In ray tracing, rays are followed from the eye, through the pixel and into the scene, and finally to the light sources. For scenes dominated by specular surfaces, for which light is reflected in just one direction, this is very efficient. It is also efficient in the sense that it only considers paths that will have an effect on the final image. In the ray tracing approach, shadows are computed by casting a ray from each object in the image to each light source.

Another approach to finding the important paths through the environment is to use finite element methods. These are often referred to as radiosity methods in computer graphics, and were introduced by Goral et al. (4) and Nishita and Nakamae (5). In finite element methods, simultaneous equations are formed describing the amount of light exchanged between each pair of surfaces. Finite element approaches are efficient when environments are dominated by diffuse rather than specular surfaces. A result is computed for the entire environment, rather than for one image. This is useful for applications in which an observer wants to navigate interactively through an environment, rather than to look at one still image of it. In finite element methods, shadows are accounted for in the calculation of the coefficients of exchange between each surface and the light sources. The geometric portion of these coefficients is referred to as the form factor, and accounts for the mutual visibility of the surfaces. These factors are computed by any of a variety of methods, including variations of ray casting, shadow volumes, and shadow maps.

Volumes of Media

Volumes of media, rather than just solid surfaces, also affect the paths of light in an environment. Examples of volumes of media include smoke, fog, and dust. Volumes of media that affect, or participate in the light transport in a scene are sometimes referred to as *participating media.* Figure 5 shows a simple scene without any participating media. The scene is illuminated by daylight entering a window at the right. Figure 6 shows the same scene filled with a volume of a participating medium. A bright area is visible where the medium scatters the entering daylight into the direction of the image viewpoint. The visibility of objects in the room is slightly reduced by the presence of the medium.

Volumes of media may reduce the quantity of light traveling along a path either by absorbing the light, or by scattering it out of the path. This reduction of the quantity of light causes the medium to cast full or partial shadows. Volumes of media may also increase the quantity of light traveling in a particular direction by emitting light (i.e., volumetric light

Figure 5. A scene with no participating media. The scene is illuminated by daylight coming in a window at the right.

sources such a glowing gas) or by scattering light in the direction of the path.

The effects of volumes are computed by extensions of the methods used for surfaces. Ray tracing methods can be used to follow the paths of light through volumes, increasing or decreasing light values along the path to account for absorption, scattering, and emission. Finite element methods can be used by discretizing volumes into small subvolumes, and computing the coefficient of exchange between each pair of subvolumes and each surface-volume pair, as well as between all the pairs of surfaces.

Both ray tracing and finite element approaches in their most basic forms are very inefficient for completely computing the global illumination for a scene. Many variations of each approach have been developed, as well as both hybrid ray tracing and finite element methods.

MATHEMATICAL FORMULATION AND SOLUTION METHODS

The equation governing the light transport required for global illumination is referred to as the *rendering equation* in com-

Figure 6. The same scene as shown in Fig. 5, but filled with a participating medium. The bright area is visible as the result of the medium scattering incident daylight in the direction of the viewer.

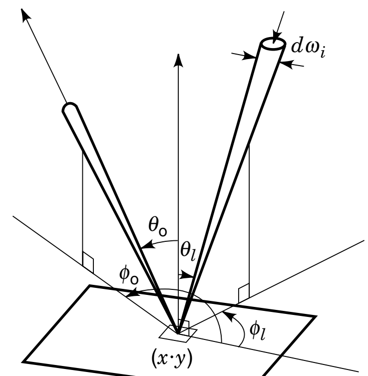

Figure 7. The rendering equation is expressed in terms of the light incident on a surface from a direction (θ_i, ϕ_i), and leaving the surface in direction (θ_o, ϕ_o.)

puter graphics. It is expressed in terms of radiance, the energy per unit time, unit solid angle and unit projected area in a direction of travel. The rendering equation gives the spectral radiance L_o leaving a surface location (x, y) that is visible in the image, in the direction to the viewer (θ_o, ϕ_o) at a wavelength λ. The angles (θ_o, ϕ_o) are measured with respect to the surface normal, as shown in Fig. 7.

The quantities that are defined for a surface when an environment is numerically modeled are the radiance it emits, $L_e(\lambda, x, y, \theta_o, \phi_o)$, and its BRDF $f_r(\lambda, x, y, \theta_i, \phi_i, \theta_o, \phi_o)$. The emitted radiance is nonzero only for light sources. The BRDF represents the light reflected in direction (θ_o, ϕ_o) as a result of light incident from (θ_i, ϕ_i). The BRDF is not a ratio of energies, but rather a distribution function that gives the reflected radiance divided by the incident energy flux density (energy per unit time and area.) Using these quantities, the rendering equation is:

$$\begin{aligned} L_o(\lambda, x, y, \theta_o, \phi_o) = {} & L_e(\lambda, x, y, \theta_o, \phi_o) \\ & + \int_{\Omega_i} f_r(\lambda, x, y, \theta_i, \phi_i, \theta_o, \phi_o) \\ & L_i(\lambda, x, y, \theta_i, \phi_i) \cos\theta_i \, d\omega \end{aligned} \tag{1}$$

The equation states that the radiance leaving the surface is equal to the emitted radiance plus the reflected radiance. The integration on the right hand side is over the entire hemisphere of incident angles Ω_i above the surface, to account for all light than can strike the surface and be reflected into the direction of interest.

In an image, the quantity of light coming through each pixel is sought, so the average value of L_o is computed for the area A_p around the center of each pixel p:

$$L_p(\lambda) = \frac{1}{A_p} \int_{A_p} L_o(\lambda, x, y, \theta_o, \phi_o) \, dA$$

Arbitrary spectral distributions can not be displayed on a video monitor or on the printed page, so the radiance is integrated to find the three primary color components X, Y, and Z as defined by the Commission Internationale de l'Éclairage (CIE). For example, the X component is computed by convolving the spectral radiance distribution $L_p(\lambda)$ with the CIE defined function $x(\lambda)$ for a standard observer:

$$Xp = \int L_{\rm p}(\lambda)x(\lambda)\,d\lambda$$

The final image needs to be expressed in terms of the red, green, and blue (RGB) primaries of the display device. The values are computed using the appropriate linear transformation from XYZ to RGB for the particular spectral distributions of the display pixels. The range of values computed for RGB will be dictated by the range of radiances in the scene being modeled. The values need to be scaled to the range of the display device (typically to a range of 0. to 1. or 0 to 255) The final mapping of RGB values to the display values is known as tone mapping.

Techniques for computing global illumination consist of strategies for computing simplified discrete approximations of the rendering equation. Historically, many methods for rendering realistic images were developed without reference to Eq. (1). However, as demonstrated by Kajiya (6), all methods that attempt to simulate some aspect of global illumination can be derived from the rendering equation.

Ray Tracing

A simple ray tracing approximation to the rendering equation is the result of assuming that all interreflections are either due to specular reflections, diffuse reflection from the light source, or reflection of a constant ambient radiance. It is also assumed that any light sources are isotropic point sources that emit an energy flux density that does not change with distance from the light source. Assuming that surfaces can only reflect (not emit) light, the right-hand side of the Eq. (1) becomes the sum of three simple terms:

$$L_{\rm o}(\theta_{\rm o},\phi_{\rm o}) = k_{\rm s}L_{\rm sp}(\theta_{\rm sp},\phi_{\rm sp}) + k_d\cos\theta_{\rm so}L_{\rm e,so} + k_{\rm a}L_{\rm a} \tag{2}$$

where k_s, k_d and k_a are respectively the specular, diffuse, and ambient reflectance coefficients. Each reflectance coefficient ranges in value between zero and one. The three coefficients are a simplified expression of the BRDF. The direction (θ_{sp}, ϕ_{sp}) is the direction of specular reflection, and is equal to (θ_o, $\phi_o + \pi$). The angle θ_{so} is the angle between the surface normal and a ray cast to the light source. The explicit dependence on wavelength λ and location (x, y) in each term has been omitted for convenience. Generally, the details of the spectral distribution are disregarded, and the ray tracing approximation is expressed in terms of RGB for typical monitor values. The integral over the area around each pixel is often approximated by taking some small number of samples for each pixel and averaging them.

In Eq. (2), L_a is just a preassigned constant for the environment. Only a scalar product needs to be computed for the term k_aL_a. The term $k_d\cos\theta_{so}L_{e,so}$ requires that a ray be cast to the light source. If there is an object along the path, the term is zero. If there is a clear path to the source, the cosine of the angle is computed and multiplied by the light source radiance. The term for a single light source can be replaced by a sum over many point light sources, with a ray cast at each source. The term $k_sL_{sp}(\theta_{sp}, \phi_{sp})$ is nonzero only for specular, shiny surfaces. A ray is cast from the object in the specular direction, and the next object hit is found. Eq. (2) is applied recursively to find the value of L_o for that object, and that radiance is used as L_{sp}.

In an environment in which all objects are shiny, there is no end to the recursive application of the equation, and ray paths of infinite length would be followed. However the reflectance coefficients k are all less than or equal to one, so each successive ray followed in the path accounts for a smaller and smaller contribution to the value of L_p being calculated on the image. Typically, the ray paths are cut off after some fixed number of interreflections, or when the contribution of the nth reflection is less than some fixed percentage of the light value computed so far.

Equation (2) is easily extended to account for transmitting materials that refract light, such as glass, by adding a term $k_tL_t(\theta_t, \theta_t)$ where k_t is the transmission coefficient and $L_t(\theta_t, \phi_t)$ is the radiance from the refracted direction given by Snell's Law. For a transmitting surface, rather than just following a ray in the specular direction, a ray path is also followed in the refracted direction.

Simple ray tracing has the disadvantage of not computing much of the light transported in the scene. The effect of all diffusely reflected light must be provided in the ambient term, and this term does not vary through the environment. Another disadvantage is that specular reflections are purely mirror-like. Materials like brushed metals cannot be approximated. Another disadvantage is that the fall off of light energy with distance squared that is accounted for by the solid angle term in Eq. (1) is omitted in simple ray tracing.

Distribution Ray Tracing. Distribution ray tracing is a modification of simple ray tracing that accounts for effects of the distributed nature of many of the variables in lighting. Specular reflection may not be in a single direction only, but may be distributed within a cone of directions, giving reflections in a surface a fuzzy appearance. Light sources aren't points, but are distributed in space, resulting in shadows with penumbras. Originally, the method was introduced as distributed ray tracing by Cook et al. (7), but it is now referred to as distribution ray tracing to distinguish it from parallel algorithms that distribute ray tracing calculations over many processors.

For distribution ray tracing Eq. (1) is approximated:

$$\begin{aligned} L_{\rm o}(\theta_{\rm o},\phi_{\rm o}) = {} & \frac{1}{\Omega_{\rm cone}}\int_{\rm cone} k_{\rm s}(\theta_{\rm sp},\phi_{\rm sp})L_{\rm sp}(\theta_{\rm sp},\phi_{\rm sp})\,d\omega \\ & + k_{\rm d}\int_{A_{\rm source}} \frac{L_{\rm e,so}\cos\theta_{\rm so}\cos\theta_{\rm fs}}{r_{\rm so}^2}\,dA + k_{\rm a}L_{\rm a} \end{aligned} \tag{3}$$

where the first integral is over a cone of directions subtending a solid angle Ω_{cone} around the direction of specular reflection, and the second integral is over the area of the light source. The coefficient k_s is allowed to have nonzero values over a range of directions, rather than being a delta function in the mirror direction. The distance from the surface to a point on the light source is r_{so}. The inclusion of the term $1/r_{so}^2$ in Eq. (3) accounts for the fall off of energy flux density with distance squared that was missing in the simple ray tracing method. The angle θ_{fs} is the angle between the normal of the light source surface and the ray cast toward the source. Including $\cos\theta_{fs}$ accounts for the decrease in light received when a source is viewed obliquely. The integrals are evaluated by

Monte Carlo integration. The integrals are replaced by sums:

$$L_o(\theta_o, \phi_o) = \frac{1}{N}\sum_{n=1}^{N} k_s(\theta_{sp,n}, \phi_{sp,n}) L_{sp,n} + \frac{A_{so}}{M}\sum_{m=1}^{M} \frac{k_d L_{e,so,m} \cos\theta_{so,m} \cos\theta_{fs,m}}{r^2_{so,m}}$$

where the summations are over N and M trials respectively. For the first summation, directions in the solid angle Ω_{cone} are sampled randomly to compute the appropriate values of k_s and L_{sp}. For the second summation, points on the area light source are sampled to compute $\cos\theta_{so}$, $\cos\theta_{fs}$, r_{so} and the visibility of the source. The distribution ray tracing method can be used to simulate many other effects. The calculations of the integral over the spectrum to compute RGB can be performed by Monte Carlo integration. Motion blur can be computed by integrating the value of L over a time window.

Because distribution ray tracing uses Monte Carlo integration, the resulting images may look "noisy." When an insufficient number of samples are used, there is a significant error in the computed value. A group of pixels that should have nearly the same value, because the object visible through those pixels has nearly uniform illumination, may have a different amount of error at each pixel. The result is noise in the image, with a spatial frequency equal to the spacing of the pixels in the image.

Basic probability theory gives an estimate of the expected deviation after N trials. Letting the individual sample values be L_n and the average of these samples after N trials be $\overline{L}$, the expected deviation L_{dev} in the estimate will be:

$$L_{dev} = \sqrt{\frac{\sum_{j=1}^{N}(L_j - \overline{L})}{N-1}}$$

which demonstrates that the noise in the image will decrease linearly as the square root of the number of samples increases.

With adequate sampling, distribution ray tracing can provide a much better approximation to the rendering equation than simple ray tracing. However, distribution ray tracing requires much longer to compute an image, and it still fails to account for all diffuse interreflections.

Monte Carlo Path Tracing. A complete solution to the rendering equation can be obtained by extending the idea of distribution ray tracing to Monte Carlo path tracing. In naive Monte Carlo path tracing, the Eq. (1) is approximated by replacing the integral with a summation:

$$L_o(\theta_o, \phi_o) = L_e(\theta_o, \phi_o) + \frac{\pi^2}{Q}\sum_{q=1}^{Q} f_r(\theta_{i,q}, \phi_{i,q}, \theta_o, \phi_o) L_i(\theta_{i,q}, \phi_{i,q}) \cos\theta_{i,q} \sin\theta_{i,q} \tag{4}$$

where the samples in the summation are taken in random directions in the incident hemisphere. Each sample in the summation is calculated by casting a ray in the direction (θ_i, ϕ_i) and estimating L_i. If a light source is hit, L_i is known. If a nonlight source is hit, L_i is evaluated by applying Eq. (4) recursively.

The naive form of Monte Carlo path tracing results in very large sample deviations. Excessively large numbers of samples (in the thousands) may be needed to produce a noise-free image for some scenes. Typically, a nonlinear cumulative distribution function is formed for selecting the direction ($\theta_{i,q}$) to reduce the sampling where $\cos\theta_i \sin\theta_i$ has relatively small values. Another common technique to reduce the deviation is to rewrite the single summation as two summations—one over all light sources and one over the incident hemisphere, excluding light directly from light sources. At each step in the ray path, separate estimates are made of the direct and indirect illumination contributions to L.

As in simple ray tracing, paths can become quite long. One strategy is to stop paths at some predefined length. This consistently underestimates L. Another strategy is to use a stochastic method to determine whether to continue the path. With this strategy a reflectance coefficient k ranging from zero to one is computed for each surface by integrating the BRDF over the hemisphere. In a given trial, a uniformly distributed number between zero and one is chosen. If this number is less than k, a ray is followed, and the sample value is $f_r(\theta_{i,q}, \phi_{i,q}, \theta_{o,q}, \phi_{o,q})L(\theta_{i,q}, \phi_{i,q})\cos\theta_{i,q} \sin\theta_{i,q}/k$. If the random number chosen is greater than k, the value of the sample is zero, and no further rays are followed. With this technique, long paths are likely to be followed in high-reflectance environments, and short paths in low-reflectance environments.

Many modifications have been developed to reduce the noise inherent in Monte Carlo path tracing. One widely used modification is Ward's *Radiance* method (8). Radiance uses a semi-stochastic method that limits the number of directions sampled and the length of paths followed. Radiance stores values of irradiance (i.e., the incident illumination before it is multiplied by the BRDF) as they are computed along paths for use in estimating radiances in subsequent paths.

Backward Ray Tracing. All of the ray tracing methods that start with the eye have difficulty computing caustics—bright spots that are the result of one or more specular reflections from the light source to a diffuse surface that is visible in the image. In backward ray tracing, rays are followed from light sources to specular objects to the first diffuse surface encountered along the path. The quantity of light represented by that path is recorded along with the location of the end of the path. Many such paths are recorded and simply displaying them as bright spots would produce a noisy image. A reconstruction filter is used to find a spatial average of the incident light energy per unit area on the portion of the surface struck by the caustic paths. This average incident illumination is then used to compute smooth regions of caustic illumination.

Backward ray tracing actually follows the natural path of light from the light source to the eye. It is referred to as "backward" ray tracing, however, since most ray tracing in computer graphics starts at the eye.

Finite Element or Radiosity Solutions

An alternative to ray tracing for solving the rendering equation is to use finite element approaches. Typically, finite element methods in global illumination are referred to as radiosity methods. Radiosity methods were originally developed in the fields of heat transfer and illumination engineering to compute the transfer of energy by radiation (e.g., see chapter

8 of Ref. 9.) Unlike simple ray tracing, in which all interreflections are assumed to be mirror-like, in the basic radiosity method, all interreflections are assumed to be ideal diffuse (i.e., Lambertian). The radiosity of a surface is the energy leaving the surface per unit area and time. For an ideal diffuse surface, the radiance leaving the surface is the same in all directions, and is equal to the radiosity of the surface, divided by π. The BRDF of an ideal diffuse surface is independent of direction, and is equal to ρ/π, where ρ is the reflectance of the surface, that is, the ratio of reflected and incident energy flux densities.

The radiance changes relatively slowly as a function of position on diffuse surfaces, except where there are shadow boundaries or sudden changes in reflectance. In the basic radiosity method, the radiance is assumed to be constant for discrete surfaces. Surfaces used to represent the scene are discretized into meshes of smaller surface elements for this assumption to hold. In the final image, surface radiances are interpolated so that the mesh is not visible.

For the radiosity method, Eq. (1) is approximated by:

$$L_n = L_{e,n} + \rho_n \sum_{\text{surfaces}} L_m F_{nm} \tag{5}$$

where L_n is the radiance of the surface n, $L_{e,n}$ is the emitted radiance, and ρ_n is the reflectance. The summation is over all other surfaces in the environment m. L_m is the radiance of each other surface m, and F_{nm} is the form factor between n and m. The form factor F_{nm} is the fraction of energy leaving surface n that arrives on surface m, and is given by:

$$F_{nm} = (1/A_n) \int_{A_n} \int_{A_m} \frac{VIS_{nm} \cos\theta_n \cos\theta_m \, dA_m \, dA_n}{\pi r_{nm}^2} \tag{6}$$

where A_n and A_m are the areas of the two surfaces, θ_n and θ_m are the angles between the line between points on surface n and surface m and the surface normals, and r_{nm} is the distance between the two surfaces. VIS_{nm} is equal to 1 where n and m are visible to one another, and 0 otherwise. Figure 8 shows the geometry of the form factor.

It is counterintuitive that the factor F_{nm} appears in Eq. (5), rather than F_{mn}. The reversal in the subscripts is a consequence of the reciprocity property of form factors:

$$A_n F_{nm} = A_m F_{mn}$$

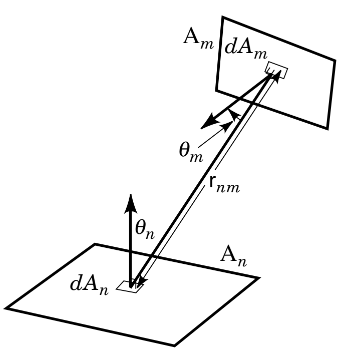

Figure 8. The form factor between two surfaces depends on the angles between the surface normals and the line of sight between the two surfaces.

For an ideal diffuse surface with radiance L, the energy leaving the surface per unit area and time is πL. The energy leaving the surface per unit time then is πLA. The energy per unit time leaving a surface m arriving at a surface n is $\pi L_m A_m F_{mn}$. An equation for the energy per unit time leaving surface n then is:

$$\pi L_n A_n = \pi L_{e,n} A_n + \rho_n \sum \pi L_m A_m F_{mn} \tag{7}$$

The left-hand side is the energy per unit time leaving surface n, and the right-hand side is the energy per unit time that is emitted plus the energy per unit time that is reflected. Dividing Eq. (7) by πA_n and applying the reciprocity property gives Eq. (5).

For environments that are well modeled as ideal diffuse, the major computational tasks in the radiosity method are meshing the surfaces, computing the form factors, and solving the set of simultaneous equations, one for each surface of the form of Eq. (5).

There are typically two steps in meshing. First, there is an initial meshing before the solution of the simultaneous equations begins. Second, there is an adaptive meshing during the solution. Major features that must be captured by appropriate meshing are shadow boundaries. Simple initial calculations, using techniques such as shadow volumes, may be done to estimate where shadow boundaries will occur, and the surfaces are more finely meshed in these areas. More sophisticated techniques for detecting where jumps in illumination will occur are referred to as discontinuity meshing.

The final model will be viewed by interpolating between surface values to give smooth radiance distributions where there are no illumination discontinuities. If the mesh is not fine enough, high-order discontinuities in the interpolation can be visible as Mach bands. To avoid these discontinuities, the meshing is refined adaptively during the solution where there are surface-to-surface variations that exceed some predefined threshold.

A wide variety of methods have been developed for computing form factors. For small surfaces that are far apart, the terms inside the integral are relatively constant. For these cases, where r_{nm}^2 is very large compared to the areas of the two surfaces the form factor is simply approximated as:

$$F_{nm} = \frac{VIS_{nm} \cos\theta_n \cos\theta_m A_m}{\pi r_{nm}^2} \tag{8}$$

In this instance, the major computational work is determining the visibility of n to m, usually by casting a ray or rays.

For surfaces that are closer together, Eq. (6) can be approximated by sampling many pairs of points on n and m. These samples can be made regularly by subdividing n and m into smaller pieces so that Eq. (8) holds. Or, the integral in Eq. (6) can be approximated stochastically by evaluating the integrand at random pairs of points.

Another approach to form factor calculations is a class of methods based on Nusselt's analogy (chapter 7 of Ref. 9.) Methods that use this approach assume that the form factor from n to m is approximately equal to the fraction of a unit circle centered on the center of surface n that is covered by the projection of m on to the hemisphere above n, and then projected to the plane of n. The hemisphere above n can be discretized into small sections q, for which the form factor

F_{nq} is known. The form factor from n to any other surface is just equal to the sum of the factors F_{nq} for the surfaces q through which m is visible to n. Rays may be cast through each hemispherical section q to determine visibility. A variation of the Nusselt analogy approach is the hemicube. In the hemicube algorithm, the sphere is replaced by half a cube, and the visibility calculations are performed by using graphics hardware to project surfaces on each side of the hemicube.

The structure of the simultaneous equations for radiosity allow iterative solutions such as Gauss–Seidel. However, viewing the results of early iterations of a traditional iterative solution does not give much insight into the appearance of the final scene. Modeling a scene is an iterative operation itself, and a good early estimate of the illumination is needed. A variation of iterative equation solving known as progressive refinement is often used instead. In a progressive refinement solution, radiances are updated by "shooting." Light is shot from high radiance surfaces (such as light sources) first. When the current highest radiance surface n that has not "shot" some portion of its radiance ΔL_n is identified, the radiance of all the other patches m are updated with:

$$\Delta L_m = \rho_m \Delta L_n F_{nm} \frac{A_n}{A_m}$$

Typically, the form factors in a progressive refinement solution are not precomputed. Each time a surface shoots, the form factors from that surface are recalculated. The calculation of factors as needed reduces the storage needed from $O(N^2)$ to $O(N)$ where N is the total number of patches into which the surfaces have been discretized. Factors from surfaces that do not contribute significantly to interreflections are never computed.

Hierarchical Methods. Even with progressive refinement methods, radiosity solutions that account for every pair of very small surface mesh elements are computationally expensive. Hierarchical methods, introduced by Hanrahan et al. (10) avoid much of this expense by adjusting the level of meshing used based on the distance between the surfaces in the current calculation. While the exchange of light between surface n is being computed to a surface m that is close by, it views surface m as being finely meshed. When the exchange is being computed between surface n and a surface p that is far way, surface p may not be subdivided at all. The exchange of light is computed at the appropriate surface subdivision level.

In hierarchical methods, the discretized mesh for each surface is represented as a tree. In each node in the tree, the surface is discretized more finely than in its parent node. The leaf nodes in the tree contain the smallest mesh elements that represent the surface. Light exchange is computed by first considering each pair of surfaces at the top level of their hierarchies. If the approximate form factor between these two surfaces is less than a predetermined threshold, a link is formed between the two surfaces. If the approximate form factor exceeds the threshold, the surfaces are compared at the next finer level in the hierarchy. This process is repeated recursively until the two surfaces are linked at the appropriate level. Surfaces for which the form factor is zero (because they do not view one another) are not linked.

When the radiosity solution is computed using the hierarchical representation, each surface interacts with other surfaces at the appropriate level using the links. Instead of each small surface element interacting with every other small surface element, most interactions occur at relatively high levels in the hierarchy. Far fewer form factors are computed. The hierarchical representation can be used to update radiances in either an iterative Gauss–Seidel, or in a progressive refinement solution. To maintain a correct representation, each time a radiance is updated at some level in the tree hierarchy, the updated value is pulled up the tree to the root node, and pushed down the tree to the leaf nodes.

Radiosity Extensions. Radiosity methods have been extended in a number of different ways. An important limitation of the basic method is the limitation to ideal diffuse reflection. One extension to include mirror-like surfaces computes additional form factors that account for the exchange of energy between surfaces that are visible to each other via mirror reflections. A more general extension of radiosity for arbitrary surfaces is to model the BRDF and radiance of each surface as a sum of spherical harmonics.

Another limitation of the basic method is the assumption of spatially constant radiance on each surface element, which requires high levels of meshing to avoid artifacts in the final interpolation. The radiosity method can be reformulated as a general finite element method with higher order (rather than constant) basis functions representing the variation of radiance across each surface. A wide variety of basis functions has been found useful in different cases, including wavelets. The difficulty in both directional and higher order radiosity is that viewing the results requires nonlinear interpolation at display time. Since current graphics hardware displays view independently colored vertices with linear Gouraud shading, the advantages of hardware speedup for interactive navigation of a scene can't be used with higher-order methods.

Hybrid Methods

Since ray tracing and radiosity methods both have advantages, many hybrid ray tracing/radiosity method have been developed. Most of these are multipass methods. In multipass methods, the radiances are computed in many steps, with different types of light transfer computed in each step.

A simple two-pass method can be used for environments with Lambertian and mirror-like surfaces. In the first pass, form factors and extended form factors are used in a radiosity solution to account for reflections between diffuse surfaces, and for diffuse surfaces with one mirror-like reflection between them. In the second pass, ray tracing is used to render the final picture. The radiance calculated by the radiosity solution is used in place of the light source and ambient contributions in Eq. (2), and mirror-like reflections are followed as in basic ray tracing.

A variation of the two-pass method is to use the radiosity method and distribution ray tracing. A first radiosity pass is computed, but the radiance for each patch is adjusted by subtracting out the light reflected directly from light sources. In the second pass, distribution ray tracing is used to compute specular and near specular reflections, and reflections directly from area light sources. The radiance from the adjusted radiosity solution is used in place of the ambient term.

An example of a multipass method uses radiosity, Monte Carlo path tracing, and backward ray tracing for caustics. In the first pass, a radiance is computed for each surface using the radiosity method. In the second pass, an image is formed using Monte Carlo path tracing with the modification that when a path hits a second ideal diffuse surface in succession, the radiance from the radiosity solution is used rather than following more rays. In the third pass, backwards ray tracing is used to find bright caustics. These are added on to the radiances computed in the Monte Carlo path tracing step. By excluding any Monte Carlo paths that followed a path of all specular surfaces and then hit a light source, the double counting of light is avoided.

Extensions to Volumes of Media

When there are volumes of media present, the rendering equation becomes an integrodifferential equation, first described in the context of graphics image formation by Kajiya and Von Herzen (11). The equation is expressed as the differential change in radiance ∂L as it passes through a differential distance in the volume ∂s:

$$\begin{aligned}\frac{\partial L}{\partial s} &= a(s)L_{\mathrm{e}}(s) - [a(s)+\sigma(s)]L(s) \\ &\quad + \frac{\sigma(s)}{4\pi}\int_{4\pi} L_i(s,\theta_{\mathrm{i}},\phi_{\mathrm{i}})P(s,\theta_{\mathrm{i}},\phi_{\mathrm{i}})\,d\omega\end{aligned} \tag{9}$$

Here $L(s)$ is the radiance along a path s in the direction s, $L_{\mathrm{e}}(s)$ is the radiance emitted, $a(s)$ is the fraction of light absorbed per unit length, and $\sigma(s)$ the fraction scattered per unit length. The function $P(s, \theta_{\mathrm{i}}, \phi_{\mathrm{i}})$ is the scattering phase function. $P(s, \theta_{\mathrm{i}}, \phi_{\mathrm{i}})$ is the ratio of the radiance incident from direction $(\theta_{\mathrm{i}}, \phi_{\mathrm{i}})$ that is scattered into a direction of the path, to the radiance that would be scattered into the path by an isotropic medium (a medium that scatters the same amount of light in all directions.) The left-hand side of Eq. (9) is the change in the radiance per unit length traveled in the medium. On the right-hand side are the three terms that account for this change—the increase due to emission, the decrease due to absorption and scattering out of the path, and the increase due to scattering into the path. The dependence of a, σ and P on the location s represents the spatial variations in the density and composition of the medium.

Equation (9) can be integrated to find the following formal solution:

$$\begin{aligned}L(s) &= L(0)\tau(s) + \int_0^s J(s^*)\tau(s-s^*)[a(s^*)+\omega(s^*)]\;ds^* \\ J(s) &= \frac{a(s)}{[a(s)+\omega(s)]}L_e(s) + \frac{\sigma(s)}{4\pi[\omega(s)+a(s)]}\int_{4\pi} \\ &\quad L_{\mathrm{i}}(s,\theta_{\mathrm{i}},\phi_{\mathrm{i}})P(s,\theta_{\mathrm{i}},\phi_{\mathrm{i}})\,d\omega \\ \tau(s) &= \exp\left(-\int_0^s [a(s^*)+\sigma(s^*)]\,ds^*\right)\end{aligned} \tag{10}$$

where $J(s)$ is the "source" radiance at a point in the medium, and $\tau(s)$ is the transmittance of the path from 0 to s. The value $L(0)$ is the radiance of the opaque surface that is visible at the beginning of the path. The integral from 0 to s in Eq. (10) is a path integral that accounts for all of the increase along the path due to emission and scattering.

A common ray tracing approximation for Eq. (10) assumes a spatially uniform "linear fog":

$$L(s) = L(0)\frac{T-s}{T} + L_{\mathrm{a}}\frac{s}{T}$$

where T is a specified thickness of the medium that totally obscures anything behind it, and L_a is a constant ambient term that approximates the source radiance. The linear function of s is used to approximate the transmittance for computational efficiency.

A more advanced ray tracing method is a two-pass method that estimates scattered radiance at discrete points within the medium in the first pass. The radiance may be estimated as the result of a single scatter from the light source for volumes with a low-scattering albedo [i.e., a small value for $\sigma/(a + \sigma)$]. The radiance for media with a high-scattering albedo can be found by approximating Eq. (9) with a perturbation expansion of the albedo and by representing the radiance with spherical harmonics to form a set of first-order partial differential equations for radiance (see Ref. 11.) Once the radiance is known within the medium, the radiance along a path can be computed by performing the path integral in Eq. (10). This method works well for media such as clouds, that are isolated from other objects in the scene. It does not take into account though all of the possible interreflections between surfaces and volumes in the scene.

A complete solution to the rendering equation in the form of Eq. (9) can be found with a variation of Monte Carlo path tracing. Equation (10) can be rewritten as:

$$L(s) = L(0)\tau(s) + [1-\tau(s)]\int_0^s J(s)\frac{\tau(s-s^*)}{[1-\tau(s)]}[a(s^*)+\sigma(s^*)]\,ds^* \tag{11}$$

Each estimate of L begins by choosing a random number s' between 0 and s, where s is the distance to the closest visible surface. The value of $\tau(s)$ is approximated by $\exp\{-[a(s') + \sigma(s')]s\}$. A second random number is selected between 0 and 1. If the number is less than 1, $L(s)$ will be approximated by the surface term $L(0)$. $L(0)$ is approximated by applying Eq. (11) recursively. If the number is greater than one, L is estimated as the second term in Eq. (11). A random number is chosen to determine a point s'' for evaluating the integrand of the path integral. The value of $J(s'')$ is approximated as L_{e} plus an estimate of the scattered light formed by selecting a random direction in the sphere of points around s''. As with Monte Carlo path tracing, the method can be modified to sample light sources separately, and different path ending strategies can be used.

Finite element methods can also be used to solve Eq. (9) The equivalent of the assumption of ideal diffuse reflection for surfaces is isotropic scattering for volumes. Rather than being the total energy leaving a volume per unit area and time, the radiosity of a volume is only the energy leaving by emission or scattering. Light that passes straight through the volume is not included in the volume radiosity. Volume radiosity then is just π times the source radiance J in the volume. The radiosity equations for a scene including volumes of media are:

$$4(\sigma_n + a_n)J_nV_n = 4a_nL_{e,n}V_n + \frac{\sigma_n}{(\sigma_n + a_n)} \left(\sum_{\text{surfaces}} L_j\overline{S_jV_n} + \sum_{\text{volumes}} J_k\overline{V_kV_n}\right)$$

$$L_wA_w = E_wA_w + \rho_w \left(\sum_{\text{surfaces}} L_j\overline{S_jS_w} + \sum_{\text{volumes}} J_k\overline{V_kS_w}\right)$$

where $\overline{S_jS_w}$, $\overline{S_jV_n}$ and $\overline{V_kV_n}$ are the surface-to-surface, surface-to-volume, and volume-to-volume form factors, similar in form to the form factors in the basic radiosity method. The surface-to-surface factors $\overline{S_jS_w}$ differ from the form factors F_{jw} in the original radiosity method in that they account for the attenuation of light by any volume of media that lies between the two surfaces, and they are multiplied by the area of A_j and so have units of area.

All of the volume sources radiances and surface radiances are found by solving a set of simultaneous equations, where there is one equation for each volume and one for each surface. The radiance for each pixel is computed using Eq. (10), using the values of L and J from the radiosity solution.

As with basic radiosity, volume radiosity methods have been extended for directional scattering distributions, and with hierarchical approaches.

ADVANCED TOPICS

Computing global illumination efficiently is still an active area of research. Research topics include more efficient radiosity and ray tracing techniques, techniques for rendering interactively, new scene representations, and techniques to exploit the properties of human perception.

Even with hierarchical radiosity, the computational complexity of radiosity is still of the order number of surfaces squared. Clustering methods attempt to extend hierarchical methods to hierarchies of objects, rather than just to represent an individual surface mesh as a hierarchy. Rather than computing surface-to-surface interaction, interreflections are computed cluster to cluster, where a cluster may contain a large number of surfaces. One approach is to model clusters of surfaces as volumes of participating media. Another approach is to model clusters as points of light with directional radiance distributions when viewed at a distance. A difficult issue with clustering is appropriately pushing and pulling the light through the hierarchy. Unlike a flat surface, in which all of the light received by a surface is distributed to the children of that surface, a child surface in a cluster may not receive energy from a particular direction because it is shadowed by another surface within the cluster.

Another approach to reducing the complexity of radiosity is to replace the entire interreflection calculation with a Monte Carlo backward ray tracing, i.e., using Monte Carlo path tracing from the light sources, and following and recording the results from all paths (not just specular paths.) After the path tracing is complete, reconstruction filters are used to estimate the radiance distribution across each surface. Although the interreflections are computed by a kind of ray tracing, the final result can be viewed as a radiosity result, since the scene can be navigated by displaying Gouraud-shaded (or texture-mapped) polygons with precomputed radiances.

An advantage of radiosity methods has been that environments with precomputed radiances can be navigated interactively. However, recomputing radiances when the geometry is altered interactively is still a challenge. Approaches to recomputing the global illumination include modifying the progressive refinement method to shoot "negative" light to undo the effects of the object that has moved in its original position. Light is then reshot selectively to add in the effects of interreflection from the object in its new position. Methods have also been explored that exploit the links developed in the hierarchical radiosity calculation to keep track of what radiances need to be updated when an object is moved.

In ray tracing approaches, bidirectional methods are being developed. These methods attempt to combine the advantages of from-the-eye Monte Carlo path tracing with the advantages of backward ray tracing. Methods for finding the direction to trace parts of the paths most efficiently are being investigated, including simulated annealing to generate the light paths used to compute the image.

Traditionally, ray tracing has been used only for still images or for animations with prespecified geometries and view paths. Scenes were only represented as geometry, and ray tracing does not compute radiances on points on the geometry, but on the image plane. However, new ways of representing scenes are being developed in the area of image-based rendering. In image-based rendering, new views of an environment are generated by interpolating between images rather than by reprojecting geometries onto the image plane. To perform this interpolation, additional information is stored at each pixel. In range images, the additional information stored at each pixel is the depth or distance from the observer at each pixel. In light fields or lumigraphs, a directional radiance distribution, rather than a single radiance, is stored for each pixel. These new image representations are opportunities for designing ray tracing methods for navigating environments in which the appearance of objects is not independent of view.

An outstanding challenge for both ray tracing and radiosity is the insertion of numerically defined objects into imagery of existing physical scenes with consistent illumination. Some progress has already been made by combining algorithms from computer vision for extracting object geometries, properties, and lighting information with global illumination algorithms.

Most current algorithms compute the value of radiance per pixel. That radiance subsequently has to be scaled to be in the range of the final display device. A typical physical scene may have radiances a factor of a hundred or more higher than the highest radiance displayable by a video monitor. Often linear scaling is used. However, not only is the absolute monitor radiance limited, the displayable contrast is also limited, often with a ratio of 30 to one between the brightest and dimmest areas of the display. Nonlinear scalings are needed to maintain the impression of the 1000 to one or more contrast ratios visible in the real world. Finding appropriate tone mapping operators to perform these scalings is an active area of research. Furthermore, since the range of radiances computed by global illumination are going to be greatly compressed in the final display, methods to minimize the calculation of radiances to an accuracy that will appear on the final display are being investigated.

BIBLIOGRAPHY

1. F. C. Crow, Shadow algorithms for computer graphics. In J. C. Beatty and K. S. Booth (eds.), *Tutorial: Computer Graphics,* Silver Spring, MD: IEEE Comput. Soc. Press, 1982.
2. L. Williams, Casting curved shadows on curved surfaces. *Proc. SIGGRAPH '78.* In *Computer Graphics,* **12** (3): 270–274, 1978.
3. T. Whitted, An improved illumination model for shaded display. *Com. ACM,* **23** (6): 343–349, 1980.
4. C. M. Goral et al., Modeling the interaction of light between diffuse surfaces. *Proc. SIGGRAPH '84.* In *Computer Graphics,* **18** (3): 213–222, 1984.
5. T. Nishita and E. Nakamae, Continuous tone representation of three-dimensional objects taking account of shadows and interreflection. *Proc. SIGGRAPH '85.* In *Computer Graphics,* **19** (3): 23–30, 1985.
6. J. T. Kajiya, The rendering equation. *Proc. SIGGRAPH '86.* In *Computer Graphics,* **20** (4): 143–150, 1986.
7. R. L. Cook, T. Porter, and L. Carpenter, Distributed ray tracing. *Proc. SIGGRAPH '84.* In *Computer Graphics,* **18** (3): 137–145, 1984.
8. G. J. Ward, The radiance lighting simulation and rendering system. *Proc. SIGGRAPH '94.* In *Computer Graphics,* Proc., Annual Conf. Series, 459–472, 1994.
9. R. Siegel and J. R. Howell, *Thermal Radiation Heat Transfer* Washington, DC: Hemisphere, 1981.
10. P. Hanrahan, D. Salzman, and L. Aupperle, A rapid hierarchical radiosity algorithm. *Proc. SIGGRAPH '91.* In *Computer Graphics,* **25** (4): 197–206, 1991.
11. J. T. Kajiya and B. P. Von Herzen, Ray tracing volume densities. *Proc SIGGRAPH '84.* In *Computer Graphics,* **18** (3): 165–174.

Reading List

I. Ashdown, *Radiosity: A Programmer's Perspective,* New York: Wiley, 1994. A guide to radiosity solution methods that includes a lot of $C++$ code examples.

M. F. Cohen and J. R. Wallace, *Radiosity and Realistic Image Syntheses,* Boston: Academic Press, Professional, 1993. A treatment of radiosity solutions that includes many extensions to the basic solution method.

A. S. Glassner, *Principles of Digital Image Synthesis,* San Francisco: Morgan Kaufmann, 1995. An exhaustive two volume work that describes all aspects of generating realistic images including global illumination.

F. X. Sillion and C. Puech, *Radiosity and Global Illumination,* San Francisco: Morgan Kaufmann, 1994. Includes discussion of both radiosity and Monte Carlo methods for computing global illumination.

G. W. Larson and R. Shakespeare, *Rendering with Radiance,* San Francisco: Morgan Kaufmann, 1998. A complete description of rendering with accurate global illumination using the Radiance software system. Examples of practical applications such as architectural design are included.

HOLLY RUSHMEIER
IBM T. J. Watson Research Center

GLOW DISCHARGE DEPOSITION. See PLASMA DEPOSITION.