Extensions of the Linear and Area Lighting Models

Kevin P. Picott
Alias Research

Extending traditional lighting models, these techniques can help you realistically render the lighting effects achieved from linear (1D) and area (2D) light sources.

Lighting models have always been a major part of creating realistic computer graphics imagery. Kajiya's rendering equation details the accurate calculation of a scene's illumination.[1] In practice, we can't calculate this lighting completely, so we have to make some approximations. The accuracy of the lighting model depends on which approximations we apply.

Traditionally, we make certain simplifying assumptions so that lighting calculations are easier to perform. These include the following:

- Point light—The light has zero size and a single position in space.
- Directional light—The light is infinitely far away so that all light rays from it are parallel.
- Spot light—The light has zero size, a single position in space, and a cone of influence with a fixed angular spread.

While these assumptions simplify our lighting calculations, they also tend to detract from the scene's realism. Recently, researchers have proposed more complicated lighting models that eliminate one or more of these assumptions. The most accurate of these is Greenberg and Cohen's radiosity model.[2] Unfortunately, an implementation of this model doesn't fit well into a system that employs traditional shading models. The radiosity model requires special coding that must be written from scratch, but the model I developed can be inserted into any traditional ray tracer.

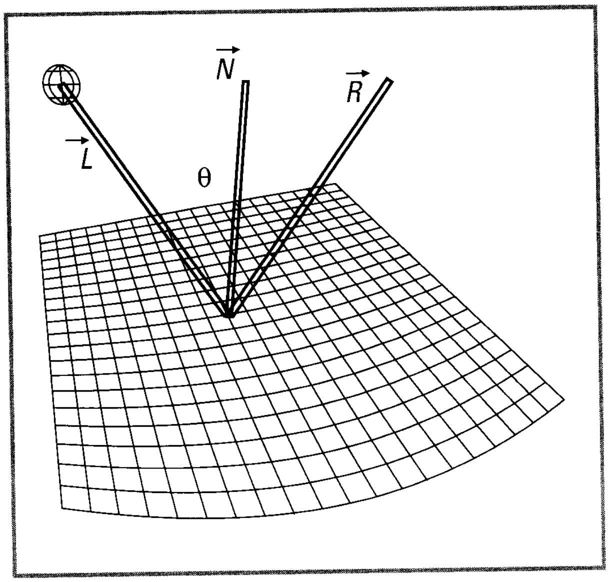


Figure 1. The standard diffuse-reflection model.

Working to achieve the accuracy of the radiosity model and integrate its strengths into the traditional shading model, I remove one of the more prominent assumptions from the point-light model—that the light has zero size. First, I expand my technique into one dimension, then two.

Linear lights

The first light we wish to tackle is one-dimensional, analogous to a fluorescent tube. This light has a finite length, but no width or height. We approximate the linear light with an infinite series of point lights, lined up along the direction of the "tube."

In this discussion I refer to the standard diffuse lighting geometry (see Figure 1). $\vec{N}$ is the surface normal, $\vec{L}$ is the vector from the surface to the light source, θ is the angle between $\vec{N}$ and $\vec{L}$, and $\vec{R}$ is the reflection of $\vec{L}$ around $\vec{N}$.

Diffuse intensity by integration

To calculate diffuse intensity I_d we use the simple Lambert shading model:

$$I_d = I_p k_d \frac{\cos\theta}{|\vec{L}|^n} \tag{1}$$

k_d is the diffuse reflection coefficient of the surface. I_p is the intensity value of the point light source. The constant n is a decay coefficient, usually 1 or 2.

The easiest way to calculate cos θ is by using the dot product of the two vectors. Assuming that $\vec{N}$ is normalized, rewrite Equation 1 as

$$I_d = I_p k_d \frac{(\vec{L} \cdot \vec{N})}{|\vec{L}|^{n+1}} \tag{2}$$

To extend this formula to the linear model we must integrate over all possible values of $\vec{L}$. We write $\vec{L}$ as a parametric function of t, with a starting point of $\vec{L}_0$ and a direction of $\vec{L}_d$.

$$\vec{L} = \vec{L}_0 + t\,\vec{L}_d, \quad t \in [0,1] \tag{3}$$

Now we can write out the full diffuse equation as an integral over t:

$$I_d = I_p k_d \int_0^1 \frac{((\vec{L}_0 + t\vec{L}_d) \cdot \vec{N})}{|(\vec{L}_0 + t\vec{L}_d)|^{n+1}}\, dt \tag{4}$$

At this point we can make a further simplifying assumption concerning n. A shading model following physical laws would require $n = 2$, since light obeys an inverse square law. Empirical studies indicate that other values of n give better results in certain circumstances. Typically, we use values of n between 0 and 3, so we can restrict the integration to these values only. These integrations are straightforward, although somewhat long. For the interested reader, the mechanics of the integrations appear in the sidebar titled "Linear-light diffuse integration calculations."

Specular intensity methods

Now that we have a formula for diffuse lighting, we need a formula for the specular component to implement a more interesting lighting model, such as the Phong model.[3] Recall that the specular illumination is given by

$$I_s = I_p k_s \cos^n \alpha \tag{6}$$

In this equation I_s is the specular illumination, I_p is the intensity value of the point light source, k_s is the specular reflection constant of the surface, α is the angle between the sight vector $\vec{E}$ and the reflected light ray $\vec{R}$, and n is the reflection value of the surface ("shininess"). In this equation n is allowed to be any real number, with typical values between 0 and 200. A quick run through a symbolic math package should convince you that this equation will not be as nice to integrate as the diffuse equation. This leaves us with two options. We can either numerically integrate the equation, using one of several known methods, or we can approximate the equation.

Since we'll apply the equation to every single vertex that uses our lighting model, numerical integration is too slow for practical use. Two other possible approaches also failed: Creating Chebyshev polynomials was too expensive to compute, and approximating $\int \cos^n \alpha$ with $(\int \cos \alpha)^n$ gave us a dropoff that appeared too severe and unrealistic.

The method yielding the best results uses our original assumption that the linear light is approximated by an infinite series of point lights. We further approximate the specular contribution of the light to the contribution made by the point

Linear-light diffuse integration calculations

To start off, we simplify Equation 4 in this article by extracting the polynomial coefficients of t as follows:

$$A = \vec{L}_0 \cdot \vec{N} \quad B = \vec{L}_d \cdot \vec{N} \quad C = \vec{L}_0 \cdot \vec{L}_0 \quad D = 2\vec{L}_d \cdot \vec{L}_0 \quad E = \vec{L}_d \cdot \vec{L}_d$$

Notice that with these equations, we only need to calculate the quantities C, D, and E once per light source. You can think of these equations as derived parameters of the light itself (since they are not independent of the actual parameters L_d and L_0). Only A and B need be recalculated for each point on the surface.

We can now write Equation 4 as a rational polynomial integral in t:

$$I_d = I_p k_d \int_0^1 \frac{A + Bt}{\sqrt{(C + Dt + Et^2)^{n+1}}}\, dt$$

Now we can use a symbolic mathematics package (like Mathematica) to solve for each value of n. For $n = 0$, we calculate

$$I_d = \frac{I_p k_d}{2E^{3/2}} \left\{ (2\boldsymbol{B}\sqrt{E}(\sqrt{C+D+E} - \sqrt{C}) + (\boldsymbol{BD} - \boldsymbol{2AE})\left(\log\left(\sqrt{C} + \frac{D}{2\sqrt{E}}\right) - \log\left(\frac{D}{2\sqrt{E}} + \sqrt{E} + \sqrt{C+D+E}\right)\right) \right\}$$

For $n = 1$, we calculate

$$I_d = I_p k_d \left\{ \frac{\boldsymbol{B}}{2E}(\log(C) - \log(C + D + E)) + (\boldsymbol{BD} - \boldsymbol{2AE}) \frac{\operatorname{atan}\left(\frac{D}{\sqrt{4CE - D^2}}\right) - \operatorname{atan}\left(\frac{D + 2E}{\sqrt{4CE - D^2}}\right)}{E\sqrt{4CE - D^2}} \right\}$$

For $n = 2$, we calculate

$$I_d = I_p k_d \left\{ \frac{2D\sqrt{C}(\boldsymbol{A} - \boldsymbol{B}) + 4\boldsymbol{A}E\sqrt{C} + 4\boldsymbol{B}C(\sqrt{C+D+E} - \sqrt{C}) - 2\boldsymbol{A}D\sqrt{C+D+E}}{\sqrt{C+D+E}(4C^{3/2}E - D^2\sqrt{C})} \right\}$$

And for $n = 3$, we calculate

$$I_d = I_p k_d \left\{ \frac{2\boldsymbol{B}C - \boldsymbol{A}D}{C(4CE - D^2)} + \frac{\boldsymbol{A}D - \boldsymbol{B}D + 2\boldsymbol{A}E - 2\boldsymbol{B}C}{(C + D + E)(4CE - D^2)} - \frac{(4\boldsymbol{A}E - 2\boldsymbol{B}D)\left(\operatorname{atan}\left(\frac{D}{\sqrt{4CE - D^2}}\right) - \operatorname{atan}\left(\frac{D + 2E}{\sqrt{4CE - D^2}}\right)\right)}{(4CE - D^2)^{3/2}} \right\}$$

To minimize calculation time, we can perform further optimization of expressions when coding the actual equations. The above equations show in boldface the values that change on each surface. We can compute the remainder of the values, including the expensive log(), atan(), and sqrt(), once per light in a precomputation phase.

along the line that has the largest specular value for a given light ray. We want to find the single point among the infinite series that makes the largest contribution to the integral. We assume the rest of the points are negligible. Although results obtained with low n values are satisfactory, this assumption works best with higher n values because the contribution from neighboring points on the light becomes negligible. (You can fine tune the approximation by sampling a neighborhood around the optimal point whose size depends on the value of n, then sum the values of that sampling. A single sample taken at the optimal point is a good approximation, but a better approximation is a set of samples taken in a neighborhood of the optimal point. The neighborhood's size decreases with increasing values of n.)

Only one step remains: Find out which point on the line will make the largest contribution to the specular illumination. In Equation 6, the only variable is α, therefore the largest contribution will be made when $\cos\alpha$ is as high as possible. (Note that we always restrict α to be positive. That is, α is always in the range 0 to 180, as opposed to the negative angles of 0 to –180.) Glancing at our handy cosine curve, we see that for positive angles the cosine is maximized when the angle is minimized. This means that the point we are looking for occurs where the

Linear-light specular calculations

We start by expanding and rewriting Equation 7 in the article as a rational polynomial in t.

$$\alpha = \frac{\vec{S}\cdot\vec{L}_0 + t\vec{S}\cdot\vec{L}_d}{|((\vec{L}_0 + t\vec{L}_d)\cdot(\vec{L}_0 + t\vec{L}_d))|\,|\vec{S}|}$$

$$= \frac{1}{|\vec{S}|}\,\frac{\vec{S}\cdot\vec{L}_0 + t\vec{S}\cdot\vec{L}_d}{\sqrt{\vec{L}_0\cdot\vec{L}_0 + t(2\vec{L}_0\cdot\vec{L}_d) + t^2(\vec{L}_d\cdot\vec{L}_d)}}$$

To keep the equations simple, we substitute representative constants for the parameters of t:

$$A = \vec{L}_d\cdot\vec{L}_d \quad B = 2\vec{L}_0\cdot\vec{L}_d \quad C = \vec{L}_0\cdot\vec{L}_0 \quad D = \vec{S}\cdot\vec{L}_0 \quad E = \vec{S}\cdot\vec{L}_d$$

Once again, in this set of variables we find that only D and E need be recalculated for each point on the surface. We can calculate values A, B, and C once as parameters of the light source.

Differentiating with respect to t will enable us to find the extreme points of α:

$$\frac{d\alpha}{dt} = \frac{1}{|\vec{S}|}\frac{d}{dt}\frac{D + Et}{\sqrt{At^2 + Bt + C}}$$

$$= \frac{1}{|\vec{S}|}\left(\frac{E}{\sqrt{C + Bt + At^2}} - \frac{(B + 2At)(D + Et)}{(C + Bt + At^2)^{3/2}}\right)$$

Now solve for the $\frac{d\alpha}{dt} = 0$ (min/max points).

(We can cancel the term $C + Bt + At^2$ because this is the length of the light. Special case handling can take over when this is 0.)

$$\frac{E}{\sqrt{C + Bt + At^2}} = \frac{(B + 2At)(D + Et)}{(C + Bt + At^2)^{3/2}}$$

$$2E(C + Bt + At^2) = (B + 2At)(D + Et)$$

$$2EC + 2EBt + 2EAt^2 = BD + BEt + 2ADt + 2AEt^2$$

$$t = \frac{(BD - 2CE)}{(BE - 2AD)}$$

To get the final formula, we can back-substitute the values we had for A, B, C, D, and E:

$$t = \frac{((\mathbf{2\vec{L}_0\cdot\vec{L}_d})(\vec{S}\cdot\vec{L}_0) - 2(\mathbf{\vec{L}_0\cdot\vec{L}_0})(\vec{S}\cdot\vec{L}_d))}{((\mathbf{2\vec{L}_0\cdot\vec{L}_d})(\vec{S}\cdot\vec{L}_d) - 2(\mathbf{\vec{L}_d\cdot\vec{L}_d})(\vec{S}\cdot\vec{L}_0))}$$

Once again, we only need to calculate the values that appear in boldface once per light source at setup time. We can use these values as constants for every surface point illuminated by this light.

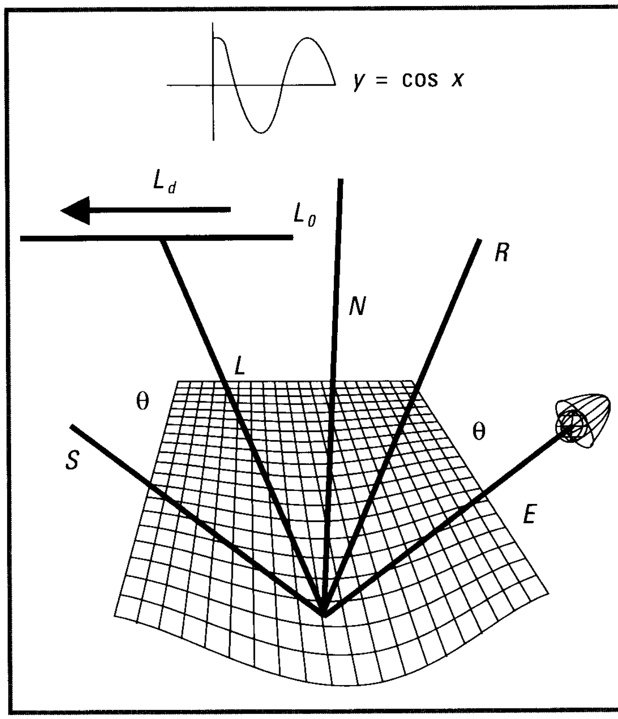


Figure 2. The linear-light specular approximation.

angle between the sight vector $\vec{E}$ and the reflected light vector $\vec{R}(\theta)$ is the smallest.

Notice that if we find the smallest angle between the light vector and the reflected sight vector, we are also finding the minimal angle θ, since the two angles are equal (see Figure 2). We can now formulate the angle θ as a function of t using a dot product:

$$\theta = \frac{\vec{S}\cdot(\vec{L}_0 + t\vec{L}_d)}{|(\vec{L}_0 + t\vec{L}_d)|\,|\vec{S}|} \tag{7}$$

Minimizing θ, we come up with a complicated but easily calculated formula for the specular intensity. (For details on the calculation of θ see the "Linear-light specular calculations" sidebar.)

$$t = \frac{((2\vec{L}_0\cdot\vec{L}_d)(\vec{S}\cdot\vec{L}_0) - 2(\vec{L}_0\cdot\vec{L}_0)(\vec{S}\cdot\vec{L}_d))}{((2\vec{L}_0\cdot\vec{L}_d)(\vec{S}\cdot\vec{L}_d) - 2(\vec{L}_d\cdot\vec{L}_d)(\vec{S}\cdot\vec{L}_0))} \tag{8}$$

This leaves a linear equation that is easily solved after checking for zero in the denominator. In this case, a zero denominator means that the light is parallel to the reflected ray. This parallelism implies zero specularity contribution. In Figures 3 and 4, we can see that the simple additions of this technique have already begun to make the image look more realistic.

Area lights

Now let's tackle 2D light. This light is analogous to a light set into the ceiling. It has a finite length and width but no height, just as an inset light would not shine from the sides. We assume that we can approximate the area light by an infinite series of point lights arranged in a rectangular array. In the specific calculations we perform, this is identical to an infinite series of linear lights arranged in a line perpendicular to their orientation. For greater generality we will use the point model; this way we can define arbitrary polygons instead of only rectangles. If we proceeded with the model of an infinite series of linear lights laid side by side, we would be restricting our model to parallelograms. But by using an array of points, we can easily extend the model to any bounded (polygonal) area.

Diffuse intensity by contour integration

In an appendix on continuous tone representation, Nishita and Nakamae present the luminance calculation for a general polygonal light source[4] (see Figure 5). The basis for this equation is that you can calculate the illumination generated by a polygonal light source as the integral of the contributions across the whole polygon. Using an application of Greene's theorem, we reduce this double integral to a contour integral, where the contours are just the edges of the polygon and hence discretely summable.

The area light is defined by its boundary vectors ($Q_{l+1} - Q_l$, for each edge of the light source). P is the point on the surface being illuminated. β_l is the angle between $\overrightarrow{PQ_l}$ and $\overrightarrow{PQ_{l+1}}$. δ_l is the angle between the tangent plane of the surface at P and the plane defined by the three points P, Q_l, and Q_{l+1}. m is the number of edges on the area light (4 in the case of a rectangle).

$$I_d = \frac{I_p}{2} k_d \sum_{l=1}^{m} \beta_l \cos \delta_l \quad \textbf{(9)}$$

Nishita and Nakamae extended this equation to include calculations of umbra and penumbra illumination. They clipped the light source against the silhouette of all objects as viewed from the surface point and summed over the clipped pieces. This technique involves taking the viewpoint of the surface point and dividing the light source into all of the continuous polygon pieces that are visible (see Figure 6, areas A, B, and C). This is valid since the invisible portions of the light source contribute no light to the surface. We can view the remaining portions of the light source as a group of area lights under which the arguments presented earlier hold.

$$I_d = \frac{I_p}{2} k_d \sum_{\text{pieces } A,B,C} \; \sum_{l=1}^{m} \beta_l \cos \delta_l \quad \textbf{(10)}$$

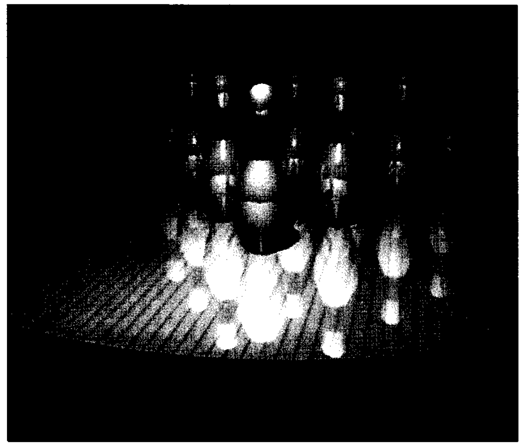


Figure 3. A bowling lane rendered with spotlights. Note the sharp shadows and cutoff area of the light. The crisp lighting is reminiscent of a studio setup, lacking other objects in the environment.

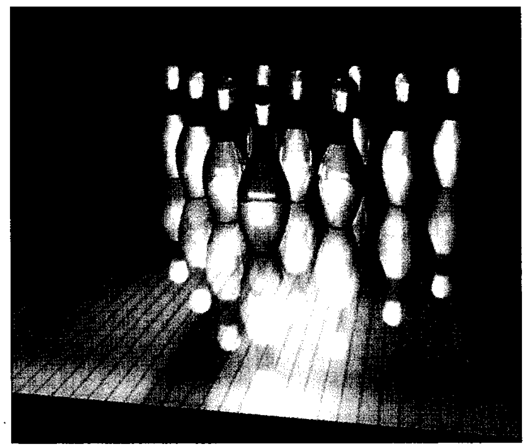


Figure 4. The bowling lane rendered with linear lights. Note the softness of the shadows and the way the light blends more into the scene.

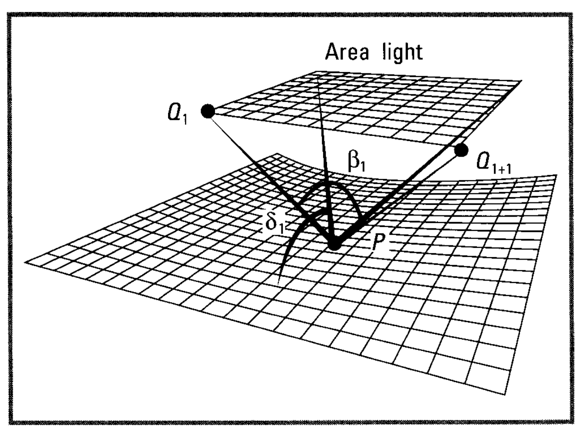


Figure 5. Contour integration geometry.

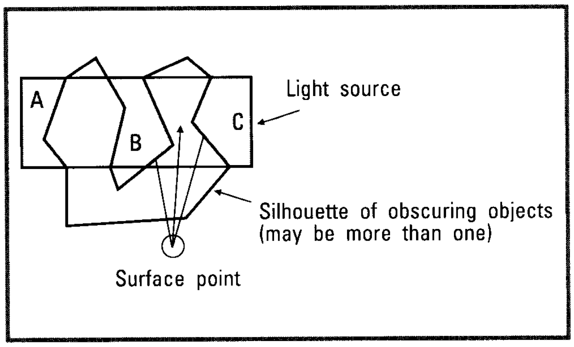


Figure 6. Clipping of the visible light source to produce the penumbra effect.

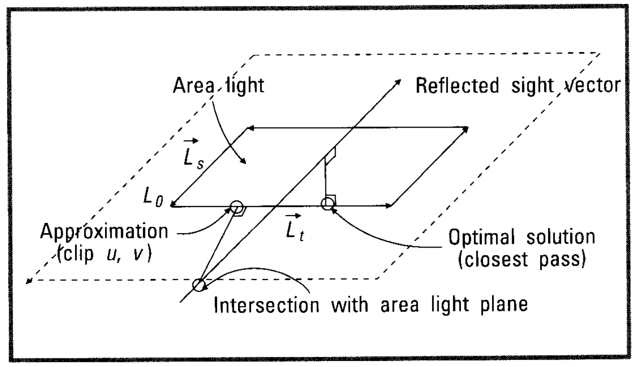


Figure 8. The area-light specularity approximation.

These equations are easy to calculate and take little computational power, yet they yield fairly realistic images (see Figure 7).

Specular intensity methods

Since our specular approximation for linear lights was successful, let's apply the same reasoning to an area light. We find the point on the light source where the angle between the reflected sight vector and the vector from the surface point to the light source is minimal (see Figure 8). In this discussion I use a few short mathematical derivations, the full details of which appear in the "Area-light specular mathematics" sidebar.

We define the light source by an origin point $\vec{L}_0$ and two direction vectors $\vec{L}_u$ and $\vec{L}_v$ indicating the two axes that form the rectangular light. This definition ensures that the light is planar, while also giving us the flexibility of forming parallelograms. (As it turns out, parallelograms fold in nicely to both the diffuse and the specular algorithms, but nonplanar lights do not. Planar N-gons, where $N > 4$, come for free in the diffuse algorithm, but they require a bit of extra work in the specular algorithms. I don't present that extra work here.)

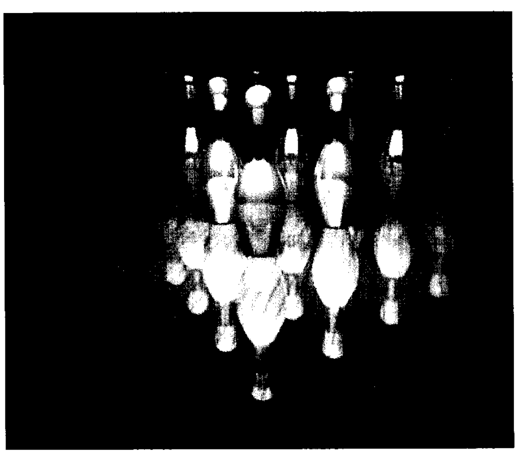


Figure 7. The bowling lane rendered with area lights. The shadows are almost nonexistent because the light is directly overhead. The pins do not self-shadow since some of the light reaches all parts of them.

First, as Bowyer and Woodwark explained,[5] we calculate the equation of the plane in which the light source lies from the three known points $\vec{L}_0$, $\vec{L}_0 + \vec{L}_u$, and $\vec{L}_0 + \vec{L}_v$. We represent this equation as

$$Ax + By + Cz + D = 0 \qquad \textbf{(11)}$$

Once we have the equation of the plane, we can perform a line-plane intersection to find out where the reflected sight vector crosses the plane of the light source. Then we test the intersection point to see if it lies inside the light source. This gives a trivial acceptance test. The intersection is relatively straightforward, but the check for being inside the light source requires a little explanation.

If our intersection point is (x, y, z), we can convert this to (u, v) coordinates in the parametric space defined by an origin at $\vec{L}_0$ and axes $\vec{L}_u$ and $\vec{L}_v$. After doing this, checking for the point being inside the light source is reduced to checking that the u and v parameters of the transformed intersection point fall in the range $[0, 1]$. This works for nonorthogonal axes since we are working solely in the u, v parametric space. For N-gons, where $N > 4$, we require an N-axis clipping algorithm in place of this simple range check.

We set up a transformation matrix to simplify calculations. In this case, the transformation goes from 3-space (x, y, z) to 2-space (u, v), so the matrix will be 3 rows by 2 columns. We can define this matrix fully by noting that $\vec{L}_0$ goes to $(0, 0)$, $\vec{L}_v$ goes to $(0, 1)$, and $\vec{L}_u$ goes to $(1, 0)$. This gives us a linear system of 6 equations with 6 unknowns, which we can solve with Gaussian elimination to give the transformation matrix $\overline{T}$.

Now we apply the transformation to the intersection point:

$$(uv) = (xyz)\overline{T} \qquad \textbf{(12)}$$

Area-light specular mathematics

Plane equation from three points

The three points we can use to calculate the equation of the plane that the light source lies in are

$$J = \vec{L}_0 \quad K = \vec{L}_0 + \vec{L}_v \quad L = \vec{L}_0 + \vec{L}_u$$

Using the notation that x_J means the x component of the point J, we write out the implicit form of the planar equation:

$$\begin{vmatrix} x - x_J & y - y_J & z - z_J \\ x_K - x_J & y_K - y_J & z_K - z_J \\ x_L - x_J & y_L - y_J & z_L - z_J \end{vmatrix} = 0$$

Solving for this determinant and putting it into the planar equation form $Ax + By + Cz + D = 0$ we come up with

$$\begin{aligned} A &= (y_K - y_J)(z_L - z_J) - (z_K - z_J)(y_L - y_J) \\ B &= (z_K - z_J)(x_L - x_J) - (x_K - x_J)(z_L - z_J) \\ C &= (x_K - x_J)(y_L - y_J) - (y_K - y_J)(x_L - x_J) \\ D &= -(x_K((y_K - y_J)(z_L - z_J) - (z_K - z_J)(y_L - y_J)) + \\ &\quad y_K((z_K - z_J)(x_L - x_J) - (x_K - x_J)(z_L - z_J)) + \\ &\quad z_K((x_K - x_J)(y_L - y_J) - (y_K - y_J)(x_L - x_J))) \end{aligned}$$

Line-plane intersection

Calculating this intersection is relatively straightforward. We start with the plane equation in standard form and the line equation in parametric form, where $x = x(t)$, $y = y(t)$, and $z = z(t)$. All we have to do is substitute the parametric forms of x, y, and z into the plane equation and solve for t. In our case the functions $x(t)$, $y(t)$, $z(t)$ can be expressed in vector form as $P + t\vec{S}$, where P is the surface point and $\vec{S}$ is the eyepoint reflection vector.

$$A(P + t\vec{S})_x + B(P + t\vec{S})_y + C(P + t\vec{S})_z + D = 0$$

$$t = -\frac{AP_x + BP_y + CP_z + D}{A\vec{S}_x + B\vec{S}_y + C\vec{S}_z}$$

Transformation between planes

The problem here is to transform a 2D space defined by an origin point and two arbitrary axis vectors into a parametric space with origin (0, 0) and axes (0, 1) and (1, 0). All we require is that this transformation is affine (it must preserve relative distances). Recall that what we want out of this space are the u, v coordinates of a point on the light-source plane relative to its position, requiring that the range $u, v \in [0,1]$ define exactly the transformed rectangular light source.

To solve for this transformation we set up a matrix $\bar{T}$ that will transform each point into its corresponding u-v values. $\bar{T}$ will be 3×2, consisting of 6 entries. Since we have three points to transform and each transforms to a pair of numbers, we get a total of 6 equations and 6 unknowns:

$$\bar{T} = \begin{pmatrix} A & D \\ B & E \\ C & F \end{pmatrix}$$

$$\begin{array}{lll} \vec{L}_0\bar{T} = (0, 0) & \vec{L}_u\bar{T} = (1, 0) & \vec{L}_v\bar{T} = (0, 1) \\ \vec{L}_0 \cdot [ABC] = 0 & \vec{L}_0 \cdot [DEF] = 0 & L_u \cdot [ABC] = 1 \\ L_u \cdot [DEF] = 0 & L_v \cdot [ABC] = 0 & L_v \cdot [DEF] = 1 \end{array}$$

Our 6 unknowns are A, B, C, D, E, and F. The values $\vec{L}_0, \vec{L}_u$, and $\vec{L}_v$ are constant. Given these 6 equations and 6 unknowns, we can solve using traditional matrix elimination techniques (the details of which are omitted for brevity) to yield the final values:

$$\begin{aligned} W = {} & L_{u_z}\vec{L}_{0_y}\vec{L}_{v_x} - \vec{L}_{u_y}\vec{L}_{0_z}\vec{L}_{v_x} - \vec{L}_{u_z}\vec{L}_{0_x}\vec{L}_{v_y} + \\ & \vec{L}_{u_x}\vec{L}_{0_z}\vec{L}_{v_y} + \vec{L}_{u_y}\vec{L}_{0_x}\vec{L}_{v_z} - \vec{L}_{u_x}\vec{L}_{0_y}\vec{L}_{v_z} \end{aligned}$$

$$\begin{array}{ll} A = (\vec{L}_{u_z}\vec{L}_{0_y} - \vec{L}_{u_y}\vec{L}_{0_z})/W & B = (\vec{L}_{u_x}\vec{L}_{0_z} - \vec{L}_{u_z}\vec{L}_{0_x})/W \\ C = (\vec{L}_{u_y}\vec{L}_{0_x} - \vec{L}_{u_x}\vec{L}_{0_y})/W & D = (\vec{L}_{0_z}\vec{L}_{v_y} - \vec{L}_{0_y}\vec{L}_{v_z})/W \\ E = (\vec{L}_{0_x}\vec{L}_{v_z} - \vec{L}_{0_z}\vec{L}_{v_x})/W & F = (\vec{L}_{0_y}\vec{L}_{v_x} - \vec{L}_{0_x}\vec{L}_{v_y})/W \end{array}$$

There are now three possibilities for (u, v):

1. Both in range: We're done. The intersection point is on the light itself.
2. One of u, v out of range: We now have a good idea of where the closest point is. It lies along the axis that the out-of-range parameter has violated. For example, if u is in range but v is less than 0, the closest point will lie along the u axis, where u is between 0 and 1. (In real-space terms, it will lie along the vector $\vec{L}_0 + t\vec{L}_u$ for $0 \le t \le 1$). It will not necessarily lie at $[u, 0]$, since this implies that the reflected vector is perpendicular to the light source axis, which we cannot assume. To find the closest hit, we can project the reflection vector onto the area light plane and intersect the projected line with the vector $\vec{L}_0 + t\vec{L}_u$.
3. Both out of range: This will require a bit more work. Since we don't know specifically which axis was violated first, we cannot tell which of the two possible axes the closest point lies along. The best thing to do here is to calculate the two closest points along each axis using the above method, then manually compare the resulting angles to see which is smaller.

In Figure 7 you can see a definite improvement in the image's realism. I achieved these effects with the area-light calculations instead of the point-light approximations.

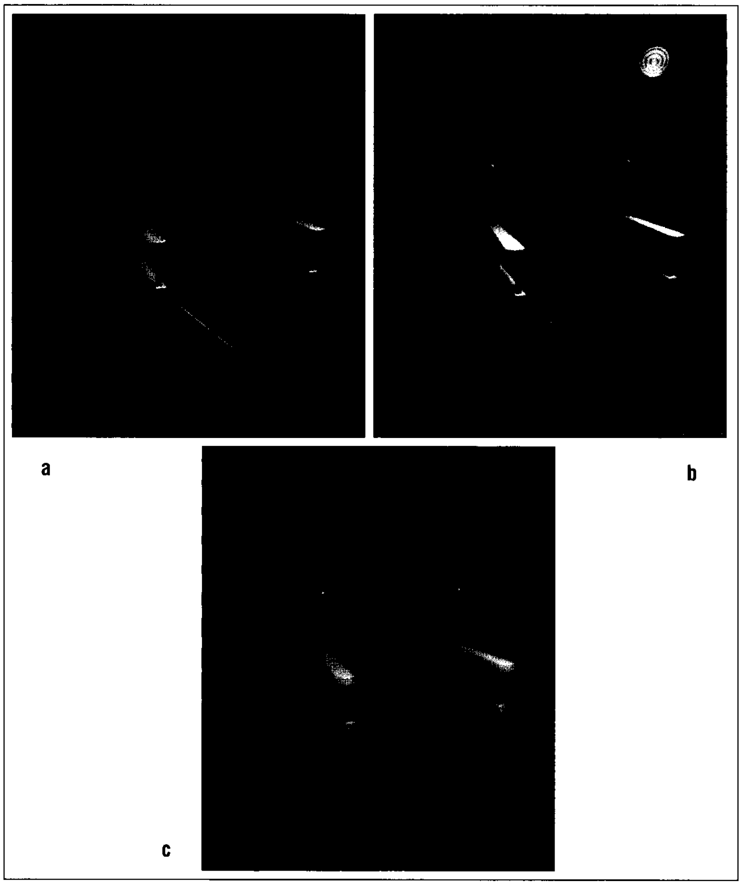


Figure 9. A more complex rendering of a studio: (a) spotlight rendering, (b) linear-light rendering, and (c) area-light rendering.

Future work

To solve for specular illumination, we can extend Nishita and Nakamae's[4] use of Green's theorem to solve the diffuse area illumination. A further extension would be examining the mathematical techniques used to come up with these solutions and applying similar techniques to the calculations for linear lighting. I hope this yields simpler formulas.

Even more interesting is the possibility of generalizing the contour integration formula to allow for 0- and 1-dimensional polygons. Currently, 0 dimensions would not work because the angle β_l is identically 0 and thus would cancel out any lighting contribution. The formula might work for one dimension, but I haven't tested it yet.

With the notable exception of inset ceiling lights, real lights have three dimensions, not two. Since Green's theorem yielded such a nice result for 2D lights, it seems only natural that Gauss's theorem (the 3D equivalent of Green's) would give an equally nice result for 3D lights. This is definitely an avenue for future investigation.

Conclusions

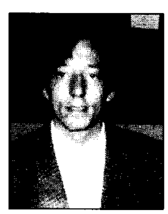
As shown in the carefully lit images of Figure 9, the extension of the lighting model to 1D and 2D lights is a very powerful tool in realistic surface rendering. With a little bit of intuition and calculus, we can implement these lighting models at a cost not much higher than traditional lighting techniques. Empirical approximations, where appropriate, also contribute to keeping the cost of lighting calculations low. ❑

Acknowledgments

I acknowledge the readers of this article, Jim Craighead, Andrew Pearce, Bob Leblanc, and Richard Sargent, all of Alias Research, without whom I would not have been able to organize this information so clearly. I also thank Gary Mundell, also of Alias, for the color plates of the studio, which dramatically highlight the practical use of these techniques.

References

1. J.T. Kajiya, "The Rendering Equation," *Computer Graphics* (Proc. Siggraph), Vol. 20, No. 3, July 1986, pp. 143-150.
2. D.P. Greenberg and M.F. Cohen, "A Radiosity Solution for Complex Environments," *Computer Graphics* (Proc. Siggraph), Vol. 19, No. 3, July 1985, pp. 31-40.
3. B. Phong, "Illumination for Computer-Generated Pictures," *Comm. ACM*, Vol. 18, No. 6, June 1975, pp. 311-317.
4. T. Nishita and E. Nakamae, "Continuous Tone Representation of 3D Objects Taking Account of Shadows and Interreflection," *Computer Graphics* (Proc. Siggraph), Vol. 19, No. 3, July 1985, pp. 23-30.
5. A. Bowyer and J. Woodwark, *A Programmer's Geometry*, Butterworths, London, 1983.



Kevin Picott has been employed for the past two years as a graphics software designer in the animation and modeling groups at Alias Research in Toronto, Ontario. His current research interests are animation, surface deformation, and software engineering.

Picott holds a BMath in Computer Science and an MMath in Computer Graphics from the University of Waterloo. He is a member of ACM, Siggraph, and IEEE.

Picott can be reached at Alias Research, 110 Richmond St., E. Toronto, Ontario M5C 1P1. His e-mail address is kpicott@alias.com.