CHAPTER 6

Daylight Simulation

by John Mardaljevic

The primary goal of daylighting analysis is the reliable evaluation of the potential of a design to provide useful levels of natural illumination. This chapter will explain how *Radiance* can be used to predict the daylighting performance of an architectural design.

It is expected that you will already be familiar with the fundamentals of the *Radiance* system, and that you have some knowledge of the way the command-line interface operates. The diffuse indirect calculation is particularly important for daylighting analysis, so a good understanding of the key features of this method is desirable. If you are specifically interested in daylighting but are new to *Radiance*, this chapter, together with the general introduction, could serve as a starting point for investigating the system.

Daylighting analysis can take many forms. A comprehensive survey of all the ways in which *Radiance* can be used to address these issues would require a book in itself. To limit the discussion to a single chapter, some compromises have to made. Rather than give cursory mention to a multiplicity of techniques, we will describe a set of key procedures in detail. These are presented in the form of case study examples. Some of the examples are straightforward descriptions of how to get from A to B. Others are expanded to demonstrate, for instance, the correspondence

between analytical solutions and *Radiance* predictions, or accuracy criteria and efficiency. If you already know some daylighting, you may wish to skip the first few case studies.

The chapter begins with an overview of daylight monitoring, with little or no mention of *Radiance*. Next, there is a general discussion about evaluation techniques and how, in broad terms, these influence the *Radiance* modeling and simulation. The bulk of the chapter is taken up with case study examples.

The important *Radiance* programs for this chapter, that is, those for which you will learn how to make informed choices for critical parameter values, are **rtrace, rpict, mkillum, gensky,** and the script **dayfact**. It is expected that you have already formed, from the general introduction, some appreciation of the function and use of the rtrace, rpict, and mkillum programs. We will use a handful of other *Radiance* programs, such as **oconv** and **rcalc**, as a matter of course.

6.1 Daylight: Monitoring, Sky Models, and Daylight Indoors

The source of all daylight is the sun. Scattering of sunlight in the atmosphere by air, water vapor, dust, and so on gives the sky the appearance of a self-luminous source of light. Here we are concerned only with daylight modeling for architectural purposes, so both the sky and the sun will be treated as light sources distant from the local scene. The brightness of the sun, or a point on the sky, will not be modified by scattering or absorption. In other words, the effects of participating media phenomena such as smog or haze on daylight will not be considered.[15]

The illumination produced by the sky depends on its luminance. Sky luminance varies according to a series of meteorological, seasonal, and geometric parameters that are difficult to specify. Characterizing the sun and sky for lighting simulation is equivalent to light source photometry for electric luminaires. Geometrically, the sky is simple to describe: the sky always has the same "shape" and "position." The brightness pattern of the sky, however, can be quite difficult to characterize for all but heavily overcast conditions. When clouds are present, the sky brightness distribution can change dramatically over very short time scales. For these reasons, it has been necessary to devise ideal sky brightness patterns known as *sky models.* These are used for the majority of daylight simulation applications. Sky models are used to generate sky brightness patterns from basic daylight quantities.

15. You are encouraged to investigate these effects at your leisure once you have grasped the requisite techniques.

6.1.1 Measuring Daylight

Continuous monitoring of the sky brightness began in earnest in the 1950s. There are now many locations in the industrialized world where 10 or more years of daylight data have been recorded and archived. The degree of monitoring varies from the most basic stations, which record integrated quantities averaged over time, to those that measure a comprehensive range of daylight metrics including the actual sky brightness distribution. They can be divided into classes as follows.

Basic

The longest time-series data from which daylight availability can be elucidated are the climatic or weather tapes [PO83]. These usually contain hourly integrated values of global and diffuse irradiance (Figure 6.1). Irradiance is a measure of the total energy flux (watts/meter^2) incident on a surface. The visible part of the radiant energy, the illuminance (lumens/meter^2), is calculated using a luminous efficacy

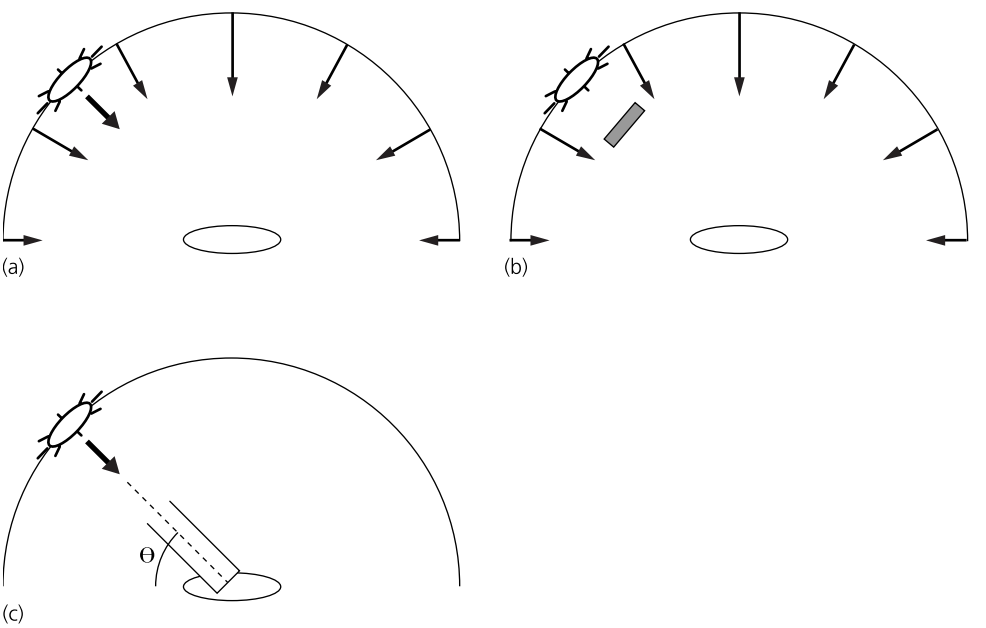

Figure 6.1 Basic daylight components: (a) global horizontal (sky and sun), (b) diffuse horizontal (sky only), and (c) direct normal (sun only).

model [Lit88]. Luminous efficacy, K, can be thought of simply as the ratio of illuminance to irradiance with units of lumens per watt:

$$K = \frac{683 \int S_\lambda v_\lambda d\lambda}{\int S_\lambda d\lambda} \qquad (6.1)$$

where S_λ is the spectral radiant flux in watts per unit wavelength interval at wavelength λ and V_λ is the relative spectral response of the eye at that wavelength. See Supplemental Information in Chapter 10 for a plot of this function. Equivalently, the ratio of luminance to radiance, which gives the same value, may be used.

This ratio is not constant and will vary with solar altitude, cloud cover, and sky turbidity. Furthermore, under the same sky conditions, the luminous efficacy for direct-beam radiation will be different from that for the diffuse component.

Intermediate

Monitoring of the visible component of irradiation, the illuminance, is nowadays more common. An intermediate-level monitoring station will measure global and diffuse illuminance together with the corresponding irradiance values. More comprehensive monitoring would include measurements of the direct components of solar illuminance and solar irradiance. These direct solar components are measured normal to the direction of the sun (Figure 6.1), so the instruments that record these quantities are mounted on sun-tracking motorized drives.

In addition, some stations record the illuminance incident on vertical surfaces facing north, south, east, and west. Here, the four vertical photocells are screened from ground-reflected radiation and the illuminance recorded is that due to the sky only. Although the four vertical values can provide some indication of the azimuthal asymmetry in the brightness distribution, these are still integrated quantities.

Advanced

The finest level of detail is provided by stations that also measure the actual sky luminance distribution using a sky-scanning device. The number of measurements taken during each scan varies according to the instrument used. These data provide the measurements necessary to validate sky models. Measured sky brightness distributions may also be used directly in the lighting simulation [Mar95].

In addition to lighting quantities, many stations also record dry bulb temperature and relative humidity.

Basic daylight quantities provide the input to sky model generator programs. Global horizontal, diffuse horizontal, and direct normal are related as follows:

$$I_{gh} = I_{dh} + I_{dn} \sin\theta \tag{6.2}$$

where I_{gh} is the global horizontal irradiance, I_{dh} is the diffuse horizontal irradiance, I_{dn} is the direct normal irradiance, and θ is the sun altitude. The same relation holds for illuminance quantities.

6.1.2 Sky Models

The simplest sky model of them all is the Uniform Luminance Model, which describes a sky of constant brightness. It was intended to represent a heavily overcast sky. It has long been appreciated, however, that a densely overcast sky exhibits a relative gradation from darker horizon to brighter zenith; this was recorded as long ago as 1901. The Uniform Luminance Sky is therefore a poor representation of any actually occurring meteorological conditions and is generally not used for illuminance modeling.

The CIE Standard Overcast Sky, originally known as the Moon and Spencer Sky, was devised to better approximate the luminance distribution observed for overcast skies. Adopted as a standard by the CIE in 1955, this description is the one most frequently used for illuminance modeling. Normalized to the zenith luminance, it has the form

$$L_\zeta = \frac{L_\zeta(1 + 2\cos\zeta)}{3} \tag{6.3}$$

where L_ζ is the luminance at an angle ζ from the zenith and L_ζ is the zenith luminance. Comparisons with measured data have demonstrated the validity of the CIE Standard Overcast Sky model as a representation of dull sky conditions [KV93].

To describe the brightness distribution for clear sky conditions requires a considerably more complex mathematical representation. The complexity arises from a number of observed effects that are accounted for in the model. Among these are a bright circumsolar region, a sky luminance minimum that is at some point above the horizon, and a brightening of the sky near the horizon. The scales of these effects are related to the solar position and the relative magnitudes of the illumina-

tion produced by the sun and sky. Like the CIE overcast standard, the CIE clear sky model is normalized to zenith luminance and the sky luminance distribution is given by [CIE73]

$$L = L_{\zeta}\frac{(0.91 + 10e^{-3\theta} + 0.45\cos^2\theta)(1 - e^{(-0.32/\sin\gamma)})}{(0.91 + 10e^{-3(\pi/2-\gamma_s)} + 0.45\sin^2\gamma_s)(1 - e^{-0.32})} \qquad (6.4)$$

where γ is the sky point altitude, γ_s is the solar altitude, and θ is the angle between the sun and the sky point. Note that the spectral distribution of skylight—its color—is not predicted by any of these models.

The overcast and clear CIE models are representations of extreme sky types—densely overcast or completely clear. Intermediate skies—that is, thin/moderate cloud cover and/or hazy atmospheric conditions—are more likely occurrences than totally clear or overcast skies for many geographical locations. Sky models generate continuous sky luminance distribution patterns. The discontinuous aspects of skylight—instantaneous cloud patterns—are not addressed.

6.1.3 Daylight Indoors—The Components of Illuminance

It helps to characterize the daylight entering a space by its origin—sun or sky—and the path by which it has arrived—directly from the source or by reflection (Figure 6.2). These categories will be particularly useful later on, when we relate light exchanges by reflection to ambient parameter settings.

6.2 Evaluation Techniques and Accuracy

Daylight simulation for interior spaces can be divided into two modes of evaluation:

- Quantitative, or numerical
- Qualitative, or visual

Quantitative data are usually presented in the form of line graphs, surface plots, or false-color maps, for example, of the distribution of illuminance across a plane. We use images to give an impression of what the finished building will look like, usually from several different viewpoints and under different lighting conditions. These modes are complementary rather than exclusive, and indeed often overlap. You may find that, even for purely numerical work, a few well-chosen images will facilitate the process of obtaining accurate predictions.

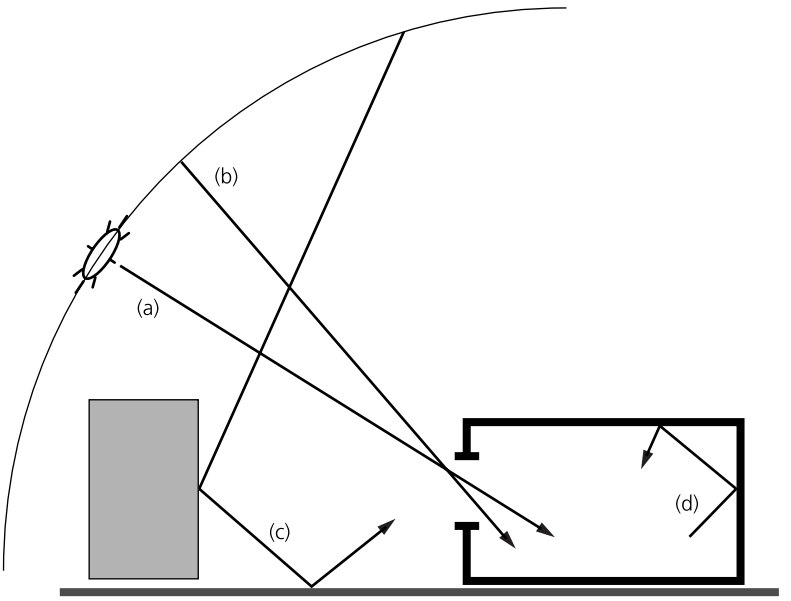

Figure 6.2 Components of daylight: (a) direct sun, (b) direct sky, (c) externally reflected, and (d) internally reflected.

For want of better criteria, we will distinguish between scenes that use the standard CIE overcast sky for illumination and the rest, which use any type of sky with sun. Overcast skies tend to be used for numerical work, which is aimed toward obtaining unambiguous quantities such as the daylight factor. Sunny sky conditions are particular to each sky, and the analysis under these circumstances will be more complex and less general than, say, a daylight factor evaluation. A few of the more common forms of analysis are described below for each of the two categories of illumination.

1. Standard CIE overcast sky conditions (daylight factor prediction)
   - Analysis of an architectural design to ensure compliance with, say, a statutory minimum daylight provision
   - Comparative evaluation of design options
   - Prediction of the daylight factor reduction caused by introducing new external obstructions to the local environment, such as a proposed nearby building
   - Visual impression of the scene accompanied by a false-color image of daylight factor (or illuminance values)

2. Skies with sun
   - Visual impression at certain times of day/year
   - Solar penetration/shading studies, such as a "movie" sequence of images
   - Effect of advanced glazing materials, such as a "movie" sequence and/or illuminance plots
   - Glare evaluation, such as locating sources of glare in an image and predicting indices for visual comfort probability

These are just some of the possibilities. The daylight factor approach is a standard technique and warrants detailed description.

6.2.1 The Daylight Factor Approach

The daylight factor at any point is the ratio of the interior illuminance at that point to the global horizontal illuminance under CIE standard overcast sky conditions. The daylight factor (DF) is normally expressed as a percentage:

$$DF = \frac{E_{in}}{E_{out}} \cdot 100 \qquad (6.5)$$

The interior illuminance is usually evaluated at workplane height (.Figure 6.3). Direct sunlight is, of course, excluded from the calculation. Because the overcast skies will generally be the dullest, the daylight factor method should be considered a "worst case" evaluation, primarily suited to calculating minimum values. Because the sky luminance does not vary with azimuth, the orientation of the scene about the *z*-axis has no effect on DF.

The conventional method to evaluate daylight factors, still very much in use, is from illuminance measurements taken inside scale models under artificial sky conditions. Unlike thermal, acoustic, or structural models, physical models for lighting do not require any scaling corrections. While a detailed physical model may indeed provide reliable results, such models can be very expensive to construct, especially if several design variants are to be evaluated. Increasingly, architects and design consultants are looking to computer simulation to offer an alternative solution approach.

Daylight factors are usually evaluated for uncluttered spaces. Since we are not interested in visual impression, the scene description usually accounts for only the important structural features of the space, and furniture and so on is not included.

Illuminance (and DF) are quantities that we derive from the irradiance predicted by the rtrace program. Often you will see that the irradiance values from the standard output of rtrace are converted directly to illuminance (or DF). Wherever in

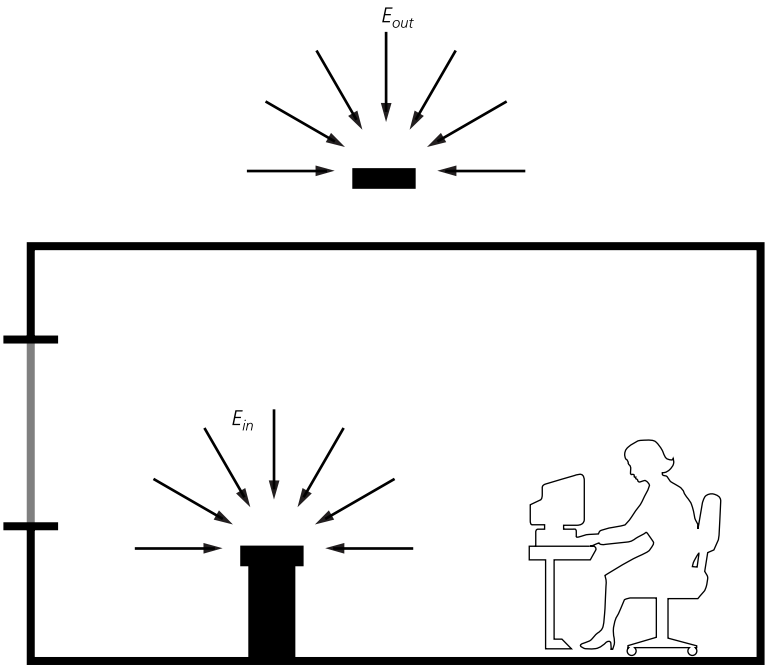

Figure 6.3 Internal and external horizontal illuminance.

the text we refer to illuminance (or DF) prediction, we shall use the term to mean irradiance prediction followed by conversion to the appropriate units. The following section describes, in general terms, how the mode of analysis influences the setting of key *Radiance* parameters.

6.2.2 Pictures, Numbers, and Accuracy

For a conventional office scene constructed with typical materials, an accurate (±10%) illuminance prediction usually requires four or more ambient bounces [Mar95]. We will see later that some of the other ambient parameters can be set to fairly low-resolution values without compromising too much the accuracy of the illuminance calculation. As most users will already have discovered, however, coarse ambient parameter settings can give fast renderings but usually produce blotchy images.

So why is it that parameters that might result in blotchy images can nevertheless give accurate illuminance predictions? The answer becomes apparent when we consider the relative complexity of DF (illuminance) prediction and image generation.

A screen-size image will comprise approximately one million pixels. Empty scenes look fairly boring, so we usually include tables, chairs, and so on, to make it look more like a real room. Depending on the view point, the image is likely to include several items of furniture. The more cluttered the scene from the view point, the harder the interreflection calculation has to work. This means more frequent sampling if we wish to avoid blotches, with the resulting computational overhead. Contrast this with an uncluttered space for DF evaluation. For an accurate prediction, it is essential that the first level of hemispherical sampling produce a good estimate of the irradiance gradient. DFs are usually evaluated at a relatively small number of points, say 50 to 500, across a plane. Furthermore, it is much easier to estimate irradiance gradients across one plane than across the hundreds of surfaces we are likely to see in the image (Figure 6.4). Because the first estimate is so important for DF calculations, we usually set a high value for **-ad**, but relax the parameters that determine the density of the ambient calculation. This allows us to use a high value for **-ab** without the simulations becoming unmanageable.

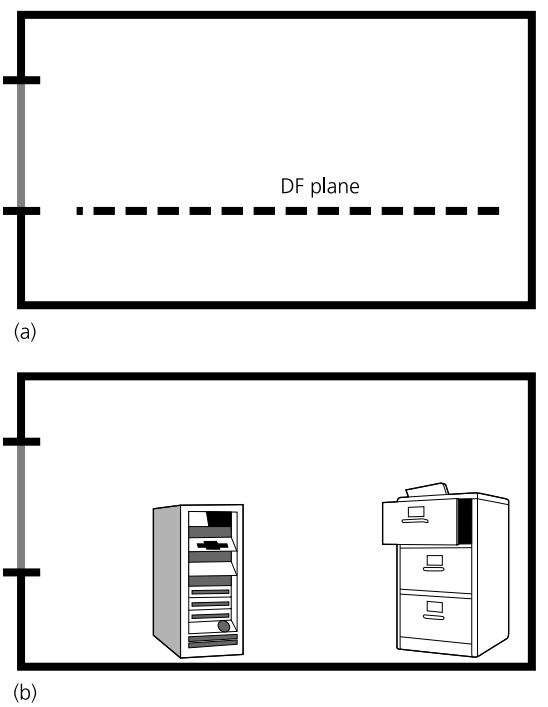

Figure 6.4 Illuminance calculation (a) can be used to calculate daylight factors. Image generation (b) can be used to render images with detail corresponding to need.

As we turn our attention now to image generation, the experienced user will already be aware that the cost of computing images rises with each successive bounce. Do images need more than one or two ambient bounces? First, we should decide what information we want our image to contain. Do we really want highly accurate (±10%) luminance values for every pixel in the image? One or two ambient bounces may give us pretty accurate luminance values, say, within 25%, for the majority of pixels in the scene, for example most of the wall, floor, and ceiling. But do we really want to crank up the number of ambient bounces to five or more just to add a little bit of luminance to each pixel, or possibly shade in what may be a tiny part of the scene? Given that sooner or later we will want to solve real-world problems, within real-world time constraints, the answer for the majority of us will be no.[16] The ambient calculation is one of the keystone features of *Radiance* and, used carefully, it can impart a tremendous impression of realism to a synthetic image. Note that it is the *directionality* of the ambient shading that lends this realism, for example the brightening of surfaces near a sun patch. This can be largely achieved with just one or two ambient bounces (possibly applied in conjunction with a mkillum-generated window). With increasing ambient bounces, the higher-level reflections tend toward a homogeneous and isotropic field of diffuse radiation. These higher-level reflections add little that can be noticed on a monitor to the pixel luminance already achieved with, say, -ab = 2. For image generation, the higher-order reflections are therefore best approximated by the careful setting of a constant ambient value (**-av**). How to choose a value for the -av parameter will be demonstrated in the case study examples. Absolute accuracy is required for illuminance prediction, and the constant ambient value is usually set to zero for these calculations.

For image generation, the conversion of a window to a light source using the mkillum utility can significantly speed up the production of smooth renderings. The technique works well as long as the total number of secondary light sources is kept reasonably small. For illuminance calculations, however, where -ab > 2 is usually essential, the preprocessing of windows to secondary light sources is generally not recommended. Similarly, for those rare occasions when images need to be rendered using a high value for -ab, it may be best to avoid using secondary sources and rely on the ambient calculation.

16. There will be exceptions; remember that these are recommendations, not rules.

6.2.3 Color Specification

How we specify the colors of the objects in our scene is another consideration. Color will influence the photometric results owing to interactions between surfaces. Visually, we perceive this in renderings as "color bleed," whereby a surface takes on some of the hue of other, usually more strongly colored, surfaces. This can be a significant effect, not just for the surface materials, but also for the sky and the sun if they are given a nongray radiance. If the RGB color values of materials are known from spectrophotometer measurements, these should be used in the simulations. If this information is not available, then for purely quantitative work, you are urged to specify gray reflection, transmission, and emission properties for *all* the materials and sources. For visual impression, however, if color data are not available, you will have to make a few good guesses. The setting of spectral radiance values for colored skies will be addressed in Section 6.7.4, Sky Spectral Radiance Values.

Having covered some of the basics, we will now demonstrate, using a series of case study examples, how to apply *Radiance* to the solution of realistic daylighting problems.

6.3 Case Study I: Creating the Luminous Environment

The sky and sun are, on an architectural scale, considered to be very distant from the local scene. In other words, the unobstructed view of the sky will be identical for all observers placed anywhere in the scene. The sky is therefore specified as a source solid angle rather than a dome of actual extent. From our local "flat Earth," the sky appears to be a luminous hemisphere. Thus, we model it as a source whose angle is 180 degrees, and we aim the center of the source directly upward, that is, toward the zenith.

Here we introduce a basic calculation technique fundamental to daylight prediction. The following example demonstrates the use of the rtrace program to determine the horizontal irradiance resulting from an unobstructed uniform sky.

6.3.1 Example: Uniform Sky

The scene file, which we will call *sky_uni.rad*, describes our entire scene, which is simply a hemispherical sky of unit radiance:

```
# uniform brightness sky (B=1)
void glow sky_glow
0
```

```
0
4 1 1 1 0
sky_glow source sky
0
0
4 0 0 1 180
```

By giving each of the spectral channels the same radiance (i.e., 1), we are defining a colorless, or "gray," sky. From this scene file, generate an octree, say

```
% oconv sky_uni.rad > sky_uni.oct
```

Now execute the rtrace program to determine the horizontal irradiance due to the uniform sky. A typical command might look like this:

```
% echo "0 0 0 0 0 1" | rtrace -h -I+ -w -ab 1 sky_uni.oct
```

which writes to the standard output the simulated spectral (RGB) irradiance values:

```
3.141593e+00  3.141593e+00  3.141593e+00
```

Because the Boolean irradiance switch is set to "on" (i.e., -I+), rtrace interprets the standard input as the measurement position (0 0 0) and orientation (0 0 1). In other words, rtrace will evaluate the irradiance at point 0 0 0 for a surface (an imaginary one) whose surface normal points upward (0 0 1). The output, therefore, is a triad of predicted values for spectral (RGB) horizontal irradiance. To convert the spectral irradiance triad to irradiance, use the following formula:[17]

$$I = 0.265 I_R + 0.670 I_G + 0.065 I_B \qquad (6.6)$$

Because the sum of the multiplying factors is 1, the achromatic irradiance equals 3.141593, which is of course the value for π. We will now compare this with an analytically derived result. For any hemisphere of radiance $B(\theta, \phi)$ the horizontal irradiance is given by

$$I = \int_0^{2\pi} \int_0^{\pi/2} B(\theta, \phi) \sin\theta \cos\theta \, d\theta \, d\phi \qquad (6.7)$$

where for a uniform sky, $B(\theta, \phi) = B$, and Equation 6.7 simplifies to

17. The coefficients should match those specified in *src/common/color.h*.

$$I = B\int_0^{2\pi}\int_0^{\pi/2} \sin\theta\cos\theta\, d\theta\, d\phi \qquad (6.8)$$

$$= \pi B$$

which, for a sky of unit radiance, gives $I = \pi$. This value for irradiance is what the rtrace simulation predicted. Because the sky was of uniform brightness, all the samples return the same radiance, and we therefore get an exact answer. For any nonuniform sky, however, the prediction will never exactly match an analytically derived result. We see this in the next example. We shouldn't worry, though, because Monte Carlo–based algorithms were never intended to give exact solutions, but they can give very accurate ones.

6.3.2 Example: CIE Overcast Sky

A more realistic example applies the same rtrace technique to a CIE standard overcast sky. Inserting the CIE overcast sky brightness distribution function (Equation 6.3) into Equation 6.7, and evaluating, gives

$$I = \int_0^{2\pi}\int_0^{\pi/2} B_z\left(\frac{1+2\sin\theta}{3}\right)\sin\theta\cos\theta\, d\theta\, d\phi \qquad (6.9)$$

$$= \frac{7}{9}\pi B_z$$

where B_z is the zenith radiance. As with the uniform sky, the analytical result is exact. However, before we can repeat the above test with rtrace, we need to be able to create skies that have nonuniform brightness distributions. To do this, we select a predefined brightness function that corresponds to the CIE overcast description, then use this to vary the brightness of the glow material. This is achieved by using the gensky program, which can generate descriptions for several sky types. We will first look at how gensky can produce CIE overcast skies. To do this, we use the **-c** option to designate the type of sky we want, but we will also use the **-b** option so we can specify a zenith radiance for the sky. (The sun angles need to be declared

also, but these will not be used by gensky for the CIE overcast, so any values can be supplied). The command

```
% gensky -ang 45 0 -c -b 1
```

writes the following to the standard output:

```
# gensky -ang 45 0 -c -b 1
# Ground ambient level: 0.8

void brightfunc skyfunc
2 skybr skybright.cal
0
3 2 1.00e+00 1.56e-01
```

The comment lines echo the gensky command and recommend a *ground ambient level*. We will discuss the significance of this value later; for the moment, we will restrict ourselves to the meaning of the rest of the output. The last line of the gensky output has three (real) arguments. These are the number `2`, indicating the type of sky, the zenith radiance (`1.00e+00`), and the ground radiance (`1.56e-01`). The zenith radiance is what we expect, since we specified this as an input argument to gensky. The significance of the ground radiance we leave for later, because our simple scene, for now, will comprise only the sky.

The output from the gensky program provides a brightness function (`skyfunc`) that we can apply as a modifier to the glow material. The easiest way to include the modifier is to execute the gensky command in the description file. The contents of the file *sky_ovc.rad* would then be as follows:

```
# CIE overcast sky (Bz = 1)

!gensky -ang 45 0 -c -b 1

skyfunc glow sky_glow
0
0
4 1 1 1 0
sky_glow source sky
0
0
4 0 0 1 180
```

The RGB radiance that the sky now assumes is `skyfunc` *multiplied* by the RGB radiance specified for glow, which here is unity for each of the channels because we want a gray (overcast) sky.

Now we create the octree for this scene, just as before:

```
% oconv sky_ovc.rad > sky_ovc.oct
```

and then calculate the horizontal irradiance using rtrace (pipe the output through rcalc to obtain the achromatic irradiance directly):

```
% rtrace -w -h -I+ -ab 1 sky_ovc.oct < samp.inp | rcalc -e \
 '$1=$1*0.265+$2*0.670+$3*0.065'
```

which produces the value

```
2.434001
```

The exact theoretical value for irradiance from the CIE overcast sky is $7_{\pi}B_z/9 = 2.443451$, since $B_z = 1$. Our predicted value is in good agreement with this. Note also that rather than being supplied through the pipe by the echo command, the coordinates are now read from the file *samp.inp*.

6.3.3 Example: CIE Overcast Sky Defined by Its Horizontal Illuminance

The preceding example showed how to generate a brightness distribution based on the standard CIE overcast sky model. The absolute brightness of the sky, however, was normalized for the purposes of illustration. Furthermore, the input and output were in units of radiance or irradiance. Before we can tackle real-world problems, we need to be able to relate the more usual daylighting quantities of luminance and illuminance to the radiance and irradiance inputs required by gensky. Recall that although the *Radiance* system calculates in units of radiance/irradiance, we will use a constant value for the factor to convert these to luminance/illuminance, or vice versa.

Daylighting practitioners commonly describe a sky in terms of the diffuse horizontal illuminance that is produced by that sky. Recall that the CIE overcast model does not include the sun, so here the global horizontal illuminance will be the same as the diffuse horizontal illuminance. The CIE overcast sky can therefore be fully characterized by the horizontal illuminance, usually given in lux. A realistic horizontal illuminance for a (brightish) overcast sky is 10,000 lux. This is a convenient figure to work with; for example, a daylight factor of 5% corresponds to an illuminance of 500 lux. The gensky program gives us two ways in which we can generate a 10,000-lux CIE overcast sky. We can specify either the zenith radiance (**-b** option) or the horizontal (diffuse) irradiance (**-B** option). The second option is perhaps the more direct, and we shall use that for the next rtrace example.

Luminous Efficacy

This conversion factor is the *Radiance* system's own value for luminous efficacy and is fixed at K_R = 179 lumens/watt (lm/w). This should not be confused with the more usual daylighting value, which can be anywhere between 50 and 150 lm/w depending on the type of sky or light considered.

First, we need to modify the gensky command to produce a 10,000-lux sky. The irradiance that corresponds to this illuminance is 10,000/179 = 55.866 w/m^2. The line giving the gensky command should now look like this:

```
!gensky -ang 45 0 -c -B 55.866
```

The rest of the file remains as before. Let's now double-check that this sky is indeed what we specified. Run oconv as before, then execute a slightly modified rtrace command:

```
% rtrace -w -h -I+ -ab 1 sky_uni.oct < samp.inp | rcalc -e   \
 '$1=($1*0.265+$2*0.670+$3*0.065)*179'
```

The calculation returns the value

```
9977.17002
```

which is pretty close to our starting value of 10,000 lux, in fact within 0.3%. Notice that the irradiance output is now multiplied by 179 to convert it to illuminance (lux). So far, the only ambient parameter that we've set for the simulation has been -ab; all the other parameters will use the default settings. Since this scene comprises only a glow source, the parameters that relate directly to the density of the irradiance gradient calculation (i.e., **-aa** and **-ar**) will have no effect. Before we go on to more complex (i.e., realistic scenes), we will first have a look at the sky we have generated. To view the sky, start the **rview** program:

```
% rview -vta -vp 0 0 0 -vd 0 0 1 -vu 0 1 0 -vh 180 -vv 180 sky_ovc.oct
```

to give an angular fish-eye view of the entire sky. The view point will be useful later on, so save it in a file called *ang180.vf* using the rview command. A false-color image of the sky will show more clearly the CIE overcast sky luminance distribution:

```
% rpict -vf ang180.vf sky_ovc.oct   \
   | falsecolor -s 4000 -l cd/m^2 > ovc_lum.pic
```

The luminance scale in the **falsecolor -s** option was set too close to the approximate zenith luminance of the sky, found either from Equation 6.9 or by using the `trace` command in rview. The default label `nits` has been changed to the more familiar cd/m^2, which means the same thing. The false-color image shows what we expect to see from Equation 6.3: a brightness distribution depending only on altitude where the zenith luminance is three times that of the horizon.

6.3.4 The Ground "Glow": An "Upside-Down" Sky

Although it might seem too self-evident to point out, we should remind ourselves that at the horizon the sky "meets" the ground. An actual ground plane of finite extent, say, a disc of radius *r*, will always fall short of an "infinite" horizon. For any given view toward the horizon, we can make the gap (a black void) between the edge of the ground and the sky appear smaller by using a larger *r*. However, we can never make them meet. Furthermore, there are good reasons not to introduce an actual ground plane of inordinately large size: the resolution of an ambient calculation will be dependent on the maximum dimension of the scene.

To get around this problem, we use an upside-down sky to represent a luminous ground. To do this, we apply the `skyfunc` modifier to a 180-degree glow source, where the direction vector is pointing downward. To include a glowing ground in our scene, add the following lines to the file *sky_ovc.rad*:

```
skyfunc glow ground_glow
0
0
4 1 1 1 0

ground_glow source ground
0
0
4 0 0 -1 180
```

The glowing ground behaves differently from a glowing sky. Although the same modifier is used for both, *Radiance* can distinguish between the two by testing the *z* component of any ray's direction vector. Above the horizon, the sky-model brightness distribution is applied, but below the horizon, a constant brightness value is used.[18] Note that as with the sky, the ground brightness is achromatic. The

18. In fact, a sharp-cutoff mixing function ensures a continuous transition from ground to sky. This operates only about the horizon, leaving most of the sky independent of the ground's brightness and vice versa.

radiance value that will be used for the ground brightness was determined by the gensky program. It is based on two factors: the sky's (diffuse) horizontal irradiance and the "average ground reflectivity." The horizontal irradiance is either supplied as an argument to gensky or evaluated from the zenith radiance. The "average ground reflectivity" may also be supplied as a gensky argument (`-g refl`); otherwise, a default value of 0.2 is used (as will be the case for us). The value 0.2 (or 20%) is a typical value for ground plane reflectance. We can check the gensky-supplied value for ground radiance very easily using Equation 6.8, since the ground is in effect a luminous "hemisphere" of constant brightness. Execute the gensky command as it appears in the *scene* file:

```
% gensky -ang 45 0 -c -B 55.866
```

Recall that the last number of the gensky output for the CIE overcast sky is the ground radiance, which here is shown to be 3.56e+00 w/m^2. The illuminance from a hemisphere source of this brightness is $\pi(3.56 \times 179) = 2001.9$ lux, which is 20% (or 0.2) of the horizontal illuminance due to the sky. We shouldn't worry too much about using an "upside-down" sky for the ground, but we should be aware of the practicalities. Although the ground radiance is based on the sky's horizontal irradiance, putting something between the sky and the ground will not affect the brightness of either (Figure 6.5). In other words, no matter how built-up the model becomes, with nearby tall structures and so on, the ground radiance (where it is visible) will be the same as for an empty scene. By the same token, a single building is an obstruction. Therefore, all scenes should include a local ground plane that participates in the interreflection calculation. This will ensure that the ground plane brightness is a function of both the sky brightness and the local environment.

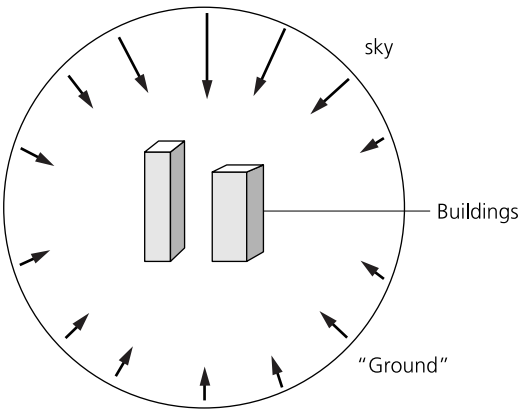

Figure 6.5 The luminous "envelope" describes luminance as a function of incident direction.

6.3.5 Summary

The scene we have constructed thus far is a seamless luminous envelope. The brightness of this envelope is based on a combination of a mathematical sky model and a ground plane reflectance model. We can specify the absolute brightness of this environment using physically meaningful quantities. Environments of this type will contain the rooms, office spaces, and so on, for which we wish to predict daylight quantities.

6.4 Case Study II: Predicting Internal Illuminances

In this example, we demonstrate how to predict DF levels for a simple scene. We show how to automate the execution of the rtrace program and how this can be used to test for convergence in the ambient calculation. The section concludes with an introduction to the dayfact script.

6.4.1 A Simple Space

The room we will use is 3 meters wide, 9 meters deep, and 2.7 meters high. These dimensions are typical of a deep-plan office module. The long dimension is aligned north-south; the room has a single south-facing window of width 2.6 meters and height 1.5 meters. The south wall is 0.2 meter thick and the window is set in the middle of this wall, so there are internal and external windowsills of depth 0.1 meter. The plan view of the room is shown in Figure 6.6. The room description is maintained in three scene files:

- *room.rad*—walls, floor, ceiling geometry
- *mat_gray.rad*—material description for walls, floor, ceiling geometry
- *window.rad*—window geometry and material description

6.4.2 Computing Daylight Factor Values

A typical analysis might begin by determining the daylight factor along the midpoint of the room. The file *samp1d.inp* contains the coordinates of the positions at which the DFs will be evaluated. Executing the rtrace command from a shell script is a convenient way to automate systematic explorations of parameter settings. The following script shows how to automate the DF calculation and test the sensitivity of the prediction to the number of ambient bounces. For this test, we cover the range `-ab 1` to `-ab 5`.

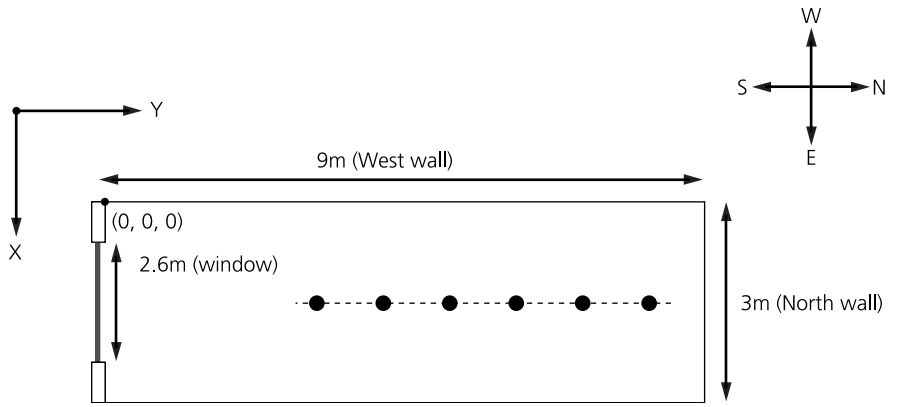

Figure 6.6 Plan view of room.

```
#!/bin/csh -f

# loop through ab
foreach ab (1 2 3 4 5)

echo "Ambient bounces" $ab

# Calculate DF

   rtrace -w -h -I+ -ab $ab -aa 0.2 -ad 512 \
     -as 0 -ar 128 scene.oct \
     < samp1.inp | rcalc -e\
     '$1=($1*0.265+$2*0.670+$3*0.065)*179/10000*100'

end
```

For all other parameter settings, the current rtrace defaults will, of course, be applied.[19] The predictions follow a characteristic pattern as shown in Figure 6.7: close to the window, the predictions for the range of -ab are relatively similar (17% to 20% at 0.5 meter). Farther away from the window, where interreflection becomes more important, they agree less (0.24% to 1.26% at 5 meters). We expect the predictions for -ab 5 to be greater than those for -ab 1, but sampling variance may mask that. We also expect the illuminance, and therefore the DF, to gradually

19. Some of these are declared in the script to allow comparison later on. Default values occasionally change when a new version of *Radiance* is released.

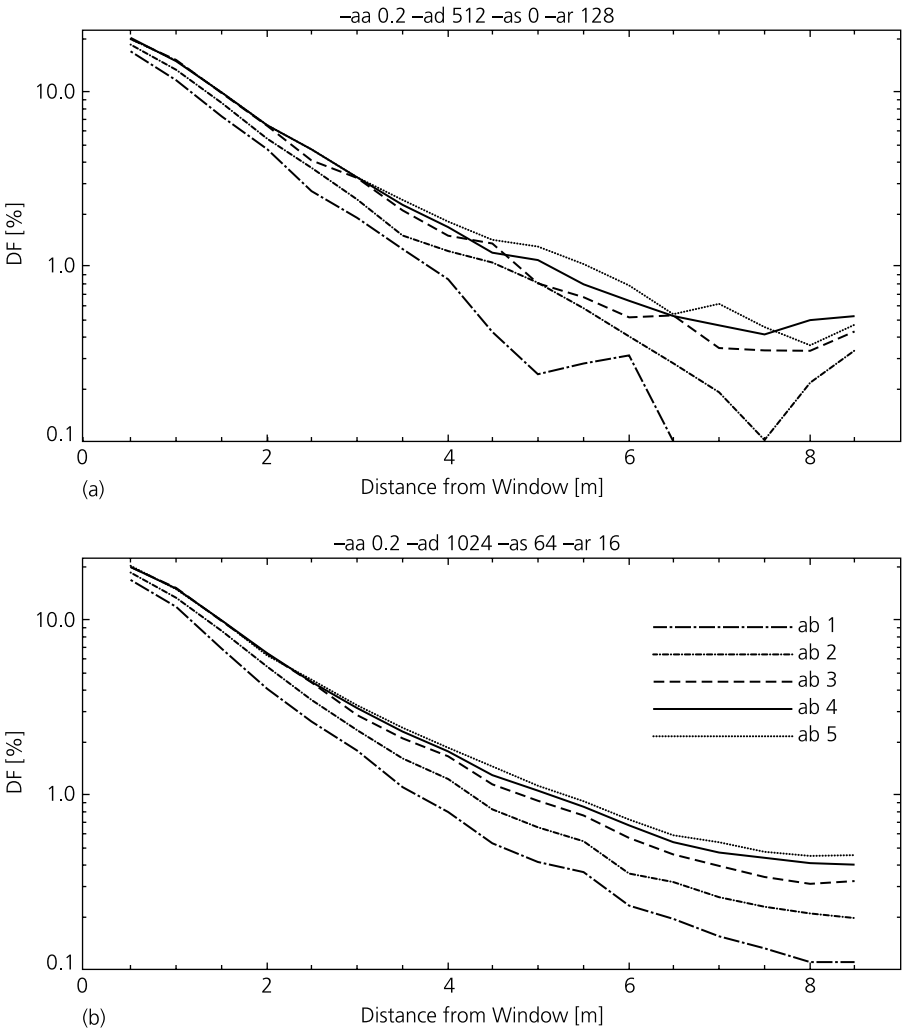

Figure 6.7 Daylight factor plots showing the effects of the -ab parameter. The top graph (a) uses fewer samples over the hemisphere, -ad 512 -as 0, than the bottom graph (b) which uses -ad 1024 -as 64.

decrease away from the window. The DF curves in Figure 6.7(a) nevertheless confound our expectations: the predictions are simply not good enough to show a consistent pattern in the data. This is especially noticeable at the rear of the room, where the curves are very jagged.

You may be relieved to learn that we don't *always* have to work through a series of -ab simulations before we can discover that one or more of the other ambient parameter settings was too coarse. We can, for many situations, use the `-ab 1` as a diagnostic to help us make better choices for some of the other settings. Recall that for `-ab 1`, the illuminance predicted will be that due to the portion of sky that is directly visible from the point of calculation, that is, the direct sky component. This component is usually the major contributor to the total illuminance at that point. If we get the direct sky component (`-ab 1`) wrong, our predictions for the total illuminance (-ab > 1) will be also poor. For this space, we know that some sky should be visible from all the points for which we want to predict the DF. Examination of the data for `-ab 1` reveals that for several points at the back of the room, the DF was predicted to be zero. This tells us that too few rays were spawned to guarantee adequate sampling of the window from all points in the DF plane. To remedy this, we should set -ad to a higher value, say 1024. We can further improve our estimates at `-ab 1` by enabling the ambient supersampling option (**-as**) in the rtrace calculation. The value we set for -as is the number of extra rays that will be used to sample areas in the divided hemisphere that appear to have high variance. In other words, for this scene, additional rays will be used to sample around the window—assuming, of course, that the ambient division sampling picked up the window in the first instance.

We now repeat the DF predictions with `-ad 1024` and `-as 64`. The ambient accuracy is the same as before, but the ambient resolution has been relaxed to `-ar 16`. These DF predictions look much better as shown in Figure 6.7(b). The curves are fairly smooth and the rank order is the same at all points along the DF plane. Which of these predictions, if any, are correct? Before we can answer this, we need to distinguish between *absolute* accuracy and *useful* accuracy. For daylighting purposes, it is important to obtain reliable predictions of the DF distribution in the critical range 10% to 0.5%. The recommended minimum DF for full daylighting is 5%, and the 1% value is generally considered to be a minimum below which the provision of daylight can be considered negligible. Thus, we need to be fairly certain of the DF down to the 1% level. There is little practical use in resolving the 0.1% DF boundary, or in distinguishing between the 0.02% and 0.05% levels. With this in mind, there is little to choose between the `-ab 4` and `-ab 5` curves. Would it be worthwhile predicting the DFs for -ab greater than 5? For this case, no. We can see from the curves that the difference between successive DF predictions for higher -ab gets smaller each time. Remember, the predictions will never be

exact, so the DF curves for scenes like this will never be perfectly smooth. The basic tenets for setting the ambient parameters are

1. Set -ad high enough to capture the visible luminous features at the first bounce.
2. Give sufficient ambient bounces to redistribute the light.
3. Set the remaining ambient parameters to sufficiently high resolution to deliver *acceptably* smooth results.

6.4.3 The Dayfact Script

The dayfact script is a user-friendly interface to the illuminance prediction capabilities of rtrace. The script essentially performs the same rtrace illuminance calculation shown above, but in addition it can create contour plots of

- Workplane illuminance
- Workplane daylight factors
- Potential savings resulting from daylight illumination based on a given lighting design level

The script works out the points in the DF plane based on user-supplied values for the plane origin and dimensions. It also determines the global horizontal illuminance directly from the gensky arguments. Try the script out using one of the ambient parameter combinations from the preceding example.

Dayfact is a handy utility to have, but because it hides some of the workings of rtrace, we do not recommend that you use it to investigate convergence and so on. Application-specific shell scripts are far better suited to exploring these aspects of the ambient calculation. The contour-level defaults built into dayfact may not be ideal for everyone and cannot be overridden. Users who do want *Radiance* contour images are urged to use the falsecolor script. Taking a dayfact-produced illuminance picture as input, falsecolor offers a great deal of user control over contour levels, color mapping, and so on. See the falsecolor manual page for details. Alternatively, you can import the illuminance prediction data into a proprietary software package that can produce contour, surface plots, and so on. The next example shows how additional objects, ground plane, and so on affect the ambient calculation, and shows how to account for them correctly.

6.5 Case Study III: Introducing Complexity

In this section, we add a ground plane and a nearby building to our simple scene. We model the ground plane as a disc of, say, radius 20 meters, centered on the origin. The diffuse reflectance for the disc material is the same as the ground plane reflectance used in the gensky command (0.2, or 20%). We can guess that the effect of the ground plane will be to slightly lower the DFs calculated in the preceding example, because, as we mentioned earlier in the chapter, we are replacing (locally) a ground glow of constant radiance with a material whose brightness now depends on the geometry and reflectance of nearby objects as well as the sky (Figure 6.8). In the vicinity of the room, the calculated ground plane radiance will be less than the ground glow radiance because the room obscures some of the ground plane's view of the sky. Rtrace now has to evaluate the ground plane brightness during the simulation; we should therefore consider the additional cost to the ambient calculation. This is best explained using a simplified ray diagram to represent the ambient bounces (Figure 6.9). The ground component of internal illuminance is, in effect, "one bounce further away" with a ground plane than it is with a ground glow. The same will be true for nearby buildings that obscure the sky (glow)—the building facade brightness will have to be evaluated as part of the ambient calculation. To complete the modifications to the scene, we now add an external obstruction: a nearby building. We represent this using a box 9 meters square and 12 meters tall, which has a diffuse reflectance of 30%. The box is positioned so that

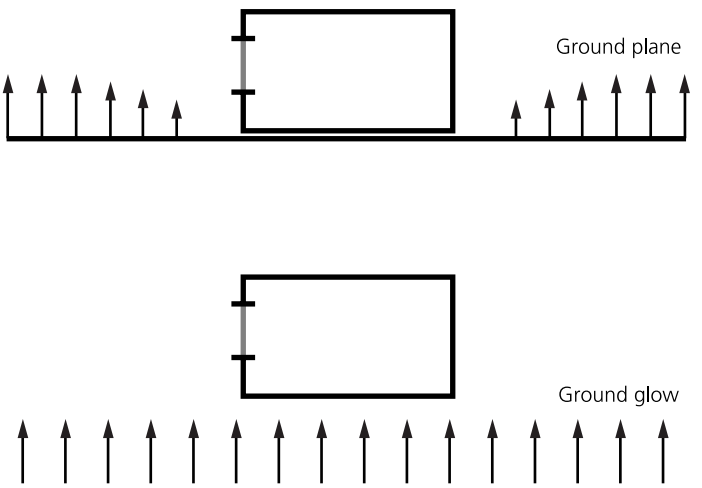

Figure 6.8 Ground plane versus ground glow.

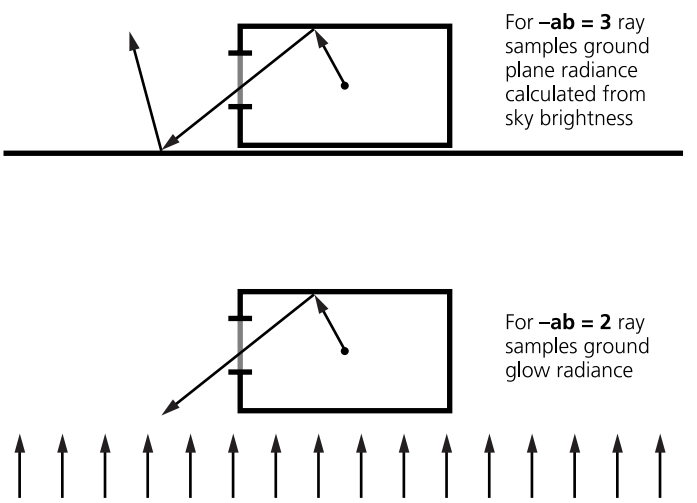

Figure 6.9 Ambient bounces and the ground plane.

it faces the room window and obscures much of the view of the sky from inside the room. The DF predictions are repeated as before, only now we increase the maximum -ab to 7.

The results for two ambient accuracy settings are shown in Figure 6.10. The DF curves in Figure 6.10(a) are surely unsound: the `-ab 1` curve shows an *increase* in DF from 0.5 to 1 meter, and for higher -ab the DF at the rear of the room is *greater* than for the unobstructed case. Before we despair, let us examine the predictions obtained using the higher ambient accuracy in Figure 6.10(b). The DF curves now begin to make sense. Why the dramatic difference? This example was contrived to create the circumstances under which the irradiance interpolation algorithm would, for certain parameter combinations, perform relatively poorly. To appreciate why this has happened, we need to recognize that irradiance interpolation can occur across the points supplied to rtrace in the same way that it can across the surfaces (i.e., pixels) computed by rpict. In other words, hemispherical sampling (at the first level) will not necessarily be initiated from every point in the DF plane supplied to rtrace.

To understand the possible outcomes, we need to examine in more detail the way the simulation progresses. Hemispherical sampling at the first level will always be initiated from the first point supplied to rtrace provided that -ab ≥ 1. From the rays spawned at the first point, the ambient calculation will predict the way the indirect irradiance is changing about that point—this is the indirect irradiance gradient. The calculation also evaluates an estimation of error associated with the prediction

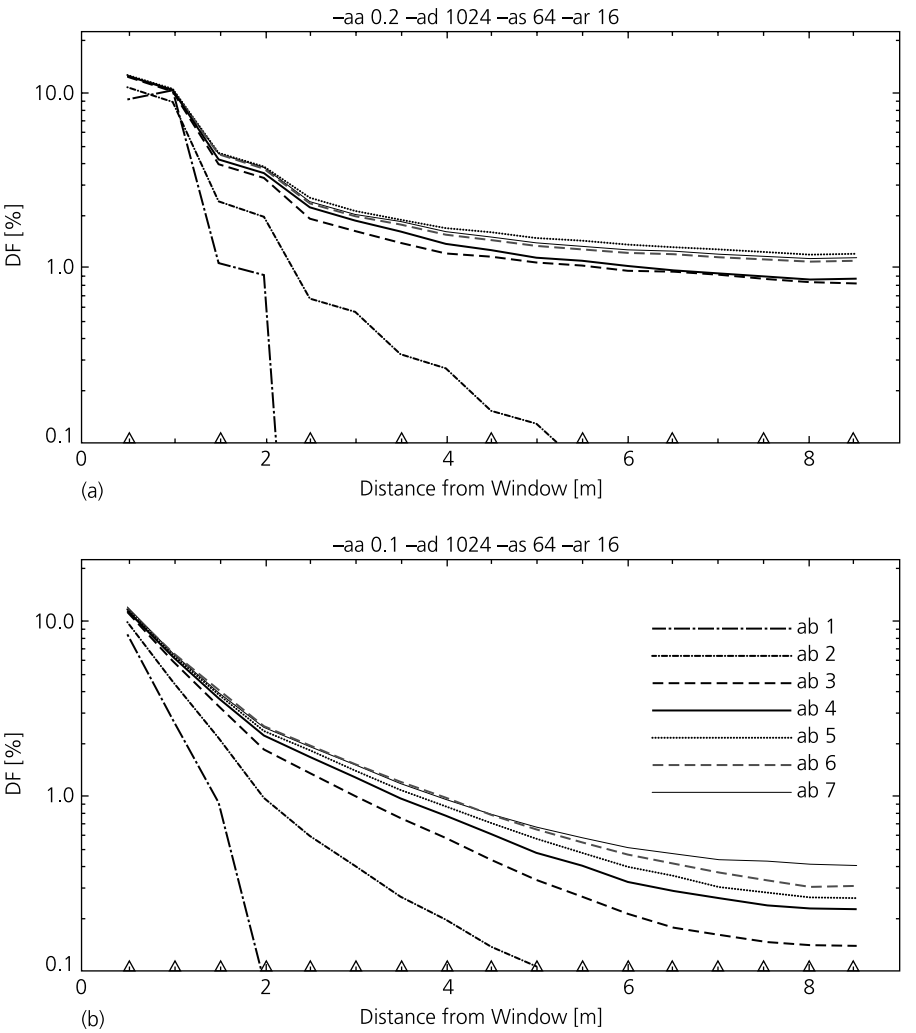

Figure 6.10 Daylight factor curves with ground plane and obstruction. The top graph (a) shows the -aa 0.2 setting, which results in an inappropriate interpolation. The bottom graph (b) shows better results with the -aa 0.1 setting.

for the irradiance gradient. These quantities, together with the ambient accuracy parameter, are used to determine a "radius of validity" for the gradient estimate. If the next point supplied to rtrace is within this radius, the indirect irradiance is evaluated from the gradient estimate and not from further hemispherical sampling. In

other words, the value is obtained by a form of interpolation rather than by actual sampling. This is a somewhat abridged description of the way the ambient calculation operates; see Chapter 12 for a detailed explanation.

Factors that influence the scale over which interpolation may occur are

- Ambient accuracy (-aa)
- Ambient resolution (-ar)
- Maximum scene dimension

The minimum possible spacing between hemispherical sampling points is the maximum scene dimension multiplied by the ambient accuracy divided by the ambient resolution. We can confirm that the bad results for `-aa 0.2` arose from interpolation by plotting on the abscissa of both graphs the points in the DF plane from which hemispherical sampling was initiated (Δ markers). For `-aa 0.1`, sampling was initiated from all the points supplied to rtrace; for `-aa 0.2`, it was from every other point. Note that a doubling of the value for ambient resolution (i.e., from 16 to 32) would *not* necessarily have effected the same cure. This is because the -ar parameter acts as a limiting device. If you are already running up against the -ar limit, increasing the setting will result in a higher density of sampling. If the limit has not been reached, then increasing -ar should have no effect.

It should now be apparent why the ground plane size should be chosen with care. This is usually the largest surface in any scene, and its size will directly affect the sampling density for any given -aa and -ar. As a rule of thumb, the ground plane should be at least twice the maximum extent (horizontal or vertical) of the scene contents.[20]

We urge you to develop this exploration of the ambient calculation one or two stages further. Add or change, one at a time, features of the scene and investigate the effect that this has on the convergence characteristics of the ambient calculation. Try to anticipate the effect of changes in scene composition and/or parameter combination. The *Radiance* ambient calculation may appear difficult to control the first few times. However, by carrying out a handful of exploratory tests, you will begin to develop an almost intuitive sense of how to manage the simulation to good effect.

20. The dimensions of a scene can be obtained using the getbbox program.

6.6 DF Prediction: Tricks of the Trade

Here are a few hints on how to accelerate the modeling and evaluation process.

6.6.1 Appropriate Complexity

For illuminance (DF) prediction, it is not normally necessary to model nearby external obstructions in fine detail. Most building facades can be modeled using a single material whose reflectance is an area-weighted average of the reflectances of the major facade elements. It may be necessary to pay attention to surface finish, especially when the adjacent building is clad in mirrored glazing.

Where visual realism is not intended, the scale of modeling complexity should generally be commensurate with the scale of the effect of the modeled structures on internal light levels. A good example of putting this principle into practice might be a DF analysis for an office module in an atrium building (Figure 6.11). Nesting of a moderately detailed scene description in a simpler structure should not compromise the accuracy of the DF predictions, but it can produce significant savings in modeling time.

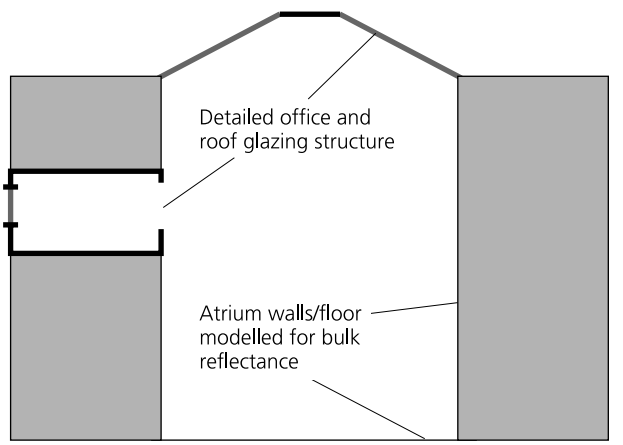

Figure 6.11 Nesting of a detailed office module in a coarsely modeled atrium building.

6.6.2 Views from the DF Plane

It often helps to visualize the scene from one or more view points along the DF plane. Choose a point in the DF plane, say, near the window, and generate a view looking directly upward—use the interactive previewer rview. Set the view type to hemispherical (**h**) and the view angles to 180 degrees. As the image resolution gradually improves, you will see a hemispherical projection view of the sky through the window. Set -av to some value to reveal the other surfaces. This makes it easier to understand the image, but what we are really interested in is the view of the sky. Compare the views with and without the external obstruction (Figure 6.12). The impact of the nearby building on internal light levels can be roughly estimated just from these images. Since the building obscures about half the view of the sky, the DF values will be approximately halved. This is a worst-case guess—it will, of course, depend on the facade reflectance. Examining a scene in this way will help you to appreciate the luminous environment "from a light meter's point of view."

6.6.3 The Ambient Exclude/Include Options

It is possible to limit the number of surfaces that participate directly in the indirect irradiance calculation. By limiting the scope of the ambient calculation, we can make significant savings in simulation time. This is achieved by telling rtrace not to include certain named material modifiers in the indirect calculation. Instead, the named materials will receive the constant ambient-value approximation. There is a complementary option called ambient include. With this option, only the named materials participate in the indirect calculation; the rest receive constant ambient-

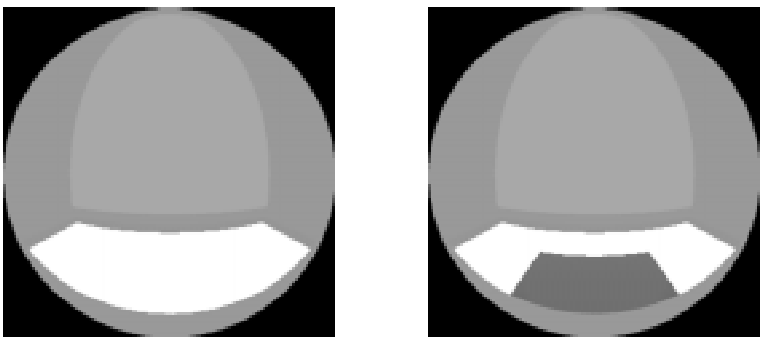

Figure 6.12 Two views from the daylight factor plane: unobstructed view (a), and view with nearby building (b).

value approximation. We should take care to exclude only those materials that play no major part in the illumination of the space. The rtrace manual page explains how the options are enabled.

6.7 Case Study IV: Creating Skies with Sun

There are two *Radiance* sky generator programs. The "official" program, which is part of the standard *Radiance* release, is called gensky; it offers a selection of sky model types based on CIE standards. The other program is called **gendaylit**; it is one of the many *Radiance* extension programs, that is, it is not part of the standard release but is freely available to all users. We will discuss this program briefly near the end of this section.

6.7.1 Gensky

In addition to the standard CIE overcast model, the *Radiance* sky generator program can produce sun descriptions and sky brightness distributions that correspond to either the CIE clear or intermediate skies. The gensky program has several modes of operation, and unless you are careful, you can end up using a sky generated from default geographical parameters that are not appropriate to the intended location. Try the command

```
% gensky -defaults
```

to check what the current defaults are. Furthermore, if you do not explicitly specify parameters in the gensky command that are related to absolute sky and sun brightness, these quantities will be evaluated using standard functions. These quantities also may not be entirely suitable for your location.

The only way to be certain of the sky and sun brightness is to supply them as gensky arguments. The sky brightness can be specified in terms of either the zenith radiance (**-b** option) or the horizontal diffuse irradiance (**-B** option). The sun brightness is either given directly (**-r** option) or evaluated from the horizontal direct irradiance (**-R** option). Most users will want to generate sun and skies based on either measured or yardstick values for global horizontal and diffuse horizontal illuminance. For example, say we want to generate a sun and intermediate sky description from these measured quantities: a global horizontal illuminance of 66,110 lux and a diffuse horizontal illuminance of 41,881 lux. The sun position was recorded as altitude 49.6 degrees and azimuth 222.5 degrees. The altitude is the angle in degrees above the horizon and the azimuth is measured as degrees east

of north. Note that this azimuth convention is different from the one used in *Radiance,* which is degrees west of south, so we need to subtract 180 degrees from the measured azimuth value. From the illuminance quantities, we need to deduce the correct gensky arguments for the -B and -R options—they are the easiest to figure out from what we have.

$$\text{horizontal diffuse irradiance} = \frac{\text{horizontal diffuse illuminance}}{\text{luminous efficacy}} \qquad \textit{(6.10)}$$

$$233.97 = \frac{41881}{179}$$

and

$$\text{horizontal direct irradiance} = \frac{\text{hor. global ill.} - \text{hor. diffuse ill.}}{\text{luminous efficacy}}$$

$$135.35 = \frac{(66110 + 41881)}{179}$$

Thus, our gensky command, executed in a scene file, would look like this:

```
# Intermediate sky with sun
# Igh=66,110 lux, Idh=41,881 lux.

!gensky -ang 49.6 42.5 +i -B 233.97 -R 135.35

skyfunc glow sky_glow
0
0
4 0.986 0.986 1.205 0

sky_glow source sky
0
0
4 0 0 1 180
```

Remember that the material and surface specifications for the sky should follow the gensky command. This sky has a small blue excess specified for the glow material (see below). You may wish to generate a sun position based on an actual time of day, in which case the site latitude, longitude, and standard meridian need to be known. The following example demonstrates how to set these values. See also the gensky manual pages for the full list of options.

6.7.2 Time of Day Image Sequence

The progression of the solar beam in a space can be shown by images generated for different times of day. The creation of these can be automated by treating the gensky (or gendaylit) time parameters as shell variables. Here we show how to generate a dawn-to-dusk sequence of images. The location is Athens; the date is July 1. The geographical coordinates of Athens are 37.97 degrees N and 23.5 degrees E, but the site meridian on which local time is based is at longitude 30.0 degrees E, that is, two hours ahead of the time at the Greenwich meridian. The gensky command for an intermediate sky at noon on this day is

```
% gensky 7 1 12 +i -a 37.97 -o -23.50 -m -30
```

Note that negative angles are used for degrees east of Greenwich (or south of the equator). Experienced shell programmers all have their own styles and are likely to do things slightly differently. The example below illustrates just one of the many ways to automate scene and picture file creation.

```
#!/bin/csh -f
#
# Set month, day and geographical coordinates
#
set mon   = 07
set month = July
set day   = 01
set coord = (-a 37.97 -o -23.50 -m -30)

set ab =    2
set ad = 512
set as = 128
set ar = 64
set aa = 0.3

foreach hr (05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20)

set skypar = ($mon $day $hr +i $coord)
set gambv = `gensky $skypar | rcalc -i '# Ground ambient\ level: ${ga}' -e '$1=ga'`
if ($gambv == 0) goto SKYDARK
set inamb = `rcalc -n -e '$1='"$gambv"'/2'`
set inamb = ($inamb $inamb $inamb)

set ambpar = (-ab $ab -ad $ad -as $as -ar $ar -aa $aa -av $inamb)
```

```
oconv -i scene.oct '\!gensky '"$skypar" sky.rad > hr.oct

rpict -vf view.vf $ambpar \
     -x 1024 -y 1024 hr.oct \
    | pfilt -1 -e 0.06 -x /3 -y /3 \
    | pcompos - 0 0 '\!psign '"$month"' '"$day"' '"$hr"'h00' 0 0 > $month$day$hr.pic

rm hr.oct

SKYDARK:

end
```

We do not intend this book to be a treatise on shell programming, so we will describe this script purely in functional terms—what it does, rather than why we do it in this way. First, we define shell variables for the month (number and name), the day, and the geographical coordinates. We then define most of the ambient parameter shell variables. The `foreach` line starts the loop; here we cycle through all the hours listed in the parentheses. Next, we group all the gensky parameters into one shell variable: `skypar`. The four lines that follow are used to set a shell variable for the constant ambient value. The value itself is based on the ground ambient value, which is extracted from the gensky output; that is why we execute gensky here. This scene was very open, so the constant ambient value was set to half the ground ambient value: a rough estimate, but adequate for this task. Included here is a test for night (that is, zero-brightness) skies. Next, just to be neat, we group all the ambient parameters to one shell variable. Then we make the scene octree. There is no need to recreate the entire octree when we are changing only the sun and sky. So to maximize efficiency, we use the include option of oconv to specify a previously created scene octree. This octree contains everything but the sun and sky. You will notice that the gensky command is executed inline with oconv. The file *sky.rad* contains the material and source descriptions for the sky and ground glow materials. Remember that this always follows the gensky command or output. The rendering command looks a little daunting, but it is really quite straightforward. For each pass of this command,

1. A picture is generated; maximum dimension is 1024 pixels.
2. The picture is filtered down to one third the original size and the exposure is set.
3. A picture label based on the settings of the shell variables for month, day, and hour is created.
4. The label is added to the filtered image.

This could be achieved in four separate steps, each producing its own output, three of which would be discarded. By using the UNIX pipe, however, we avoid the intermediary output, creating only what we want to keep.

On completion, we are left with a sequence of images showing the illumination of the scene at various times of day. These could be combined into a single picture, or even used as the basis for an animation. The script could easily be changed to cycle through other parameters, say, month, day, building orientation, and so on.

On the CD-ROM, we have included an example animation sequence showing a daylit interior throughout the hours of a day. The exposure of the images was computed to correspond to human visual response using the new pcond program.

6.7.3 Gendaylit

Another *Radiance* sky generator program, gendaylit, (written by Jean-Jacques Delavnay) produces a description based on the Perez All-Weather model [PSM93]. With this model, the generator program determines the sky conditions (overcast, intermediate, clear, and so on) based on the input parameters. You are therefore spared having to choose a particular sky type. For this reason, it is perhaps the best sky model to use with a time series of measured illuminance data, for instance, for an automated set of simulations. The gendaylit program source code is included on the CD-ROM; its use is described in the accompanying manual page.

6.7.4 Sky Spectral Radiance Values

Spectral radiance values for nongray skies should be calculated so that they do not affect the overall sky luminosity. To ensure that this is the case, the following condition should hold:

$$1 = 0.275L_R + 0.670L_G + 0.065L_B \qquad \textit{(6.11)}$$

where L_R, L_G, and L_B are the RGB radiance values for the sky glow material. The same should be true for the ground as well.

6.8 Rendering Scenes Illuminated by Sunny Skies

So far, the emphasis has been on illuminance prediction and how to obtain highly accurate values. A lighting designer will have no problem interpreting these data, but this is only part of the story. The majority of people can only really appreciate an architectural design once they have seen the finished building. If you want to

know in advance what it will look like, you need to visualize it somehow. The capabilities of the *Radiance* system make it particularly well suited to the rendering of architectural scenes under daylight illumination.

Recall that when we render a scene, we are not striving for absolute accuracy in the prediction of luminance for every pixel in the image. In fact, the accuracy criteria we employ for judging images include many subjective elements. With this in mind, we demonstrate in this section a few different approaches to image synthesis. You will by now be aware that it is impossible to recommend a single set of rendering parameters that can guarantee an efficient solution for every conceivable design type. It should, however, be possible to anticipate from the actual design and lighting conditions the best approach to solving the problem.

6.8.1 A Note about the Rad Program

This chapter is really intended for those users who will eventually want to carry out exacting quantitative work and/or produce high-quality renderings of daylight-illuminated scenes. For either of these tasks, it helps to gain a detailed understanding of how key features of the *Radiance* system work. A more direct route to producing renderings, however, is to use the **rad** program. This "executive control" program will automatically determine many of the parameter values based on a few intuitive variable settings. Rad will also construct a "rendering pipeline" for you. This could include fairly complex operations, such as a mkillum preprocess of windows. The rad program, therefore, screens you from many of the intricacies of the rendering process; it has greatly improved the overall usability of the *Radiance* system. Try out the rad program and see if suits your needs—its use is described in Part I, Tutorials (Chapters 1 through 3). Sooner or later, though, and particularly for research applications, you will want to exercise complete control over all aspects of the simulation. The sections that follow will show how this can be achieved.

6.8.2 The Simple Space Lit by a Sunny Sky

Recreate the simple room scene octree using the intermediate sky description. Include the ground plane but leave out the external obstruction. Use the rview interactive renderer to view the scene from somewhere at the back of the room, looking toward the window at about eye-level height. All that you will see at first is the sky through the window and the sun patch on the floor/wall. Initiate the interreflection calculation by setting the number of ambient bounces to 1. Restart the image with the command `new`.[21]

---

21. Note that further increases in the -ab value from within rview will not show up in the onscreen rendering (even after issuing a `new` command) because the cached values will be reused and they were computed with only a single bounce.

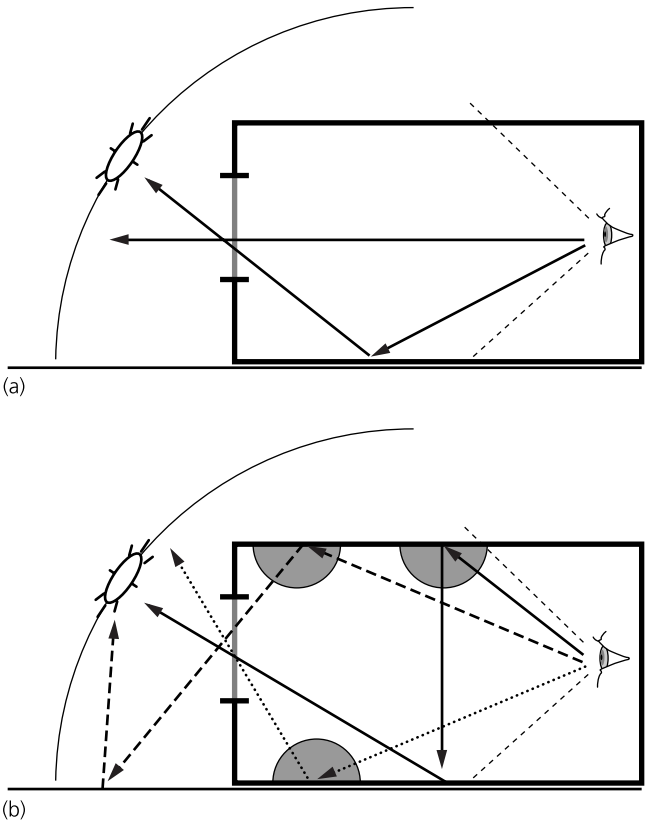

Figure 6.13 Possible light transfers for ambient bounces equal to 0 (a), and 1 (b).

You now begin to see more of the room, but it will appear blotchy because the default ambient parameter settings for rview are fairly coarse. At this stage in the chapter, we should be able to anticipate the pattern of light transfer in this scene for ambient bounce settings of 0 and 1. A pair of simplified ray diagrams illustrate some of the light transfers we can expect (Figure 6.13). With the ambient calculation switched off, we see the sky (glow) through the window and whatever sun patches are directly visible from the view point. With the interreflection calculation switched on, several other routes to the eye (that is, the camera) become possible

via hemispherical sampling. Three of these are illustrated in the second diagram of the figure. Each one shows how a distinct component of internal illumination might be evaluated during the simulation. The point in the ray path where hemispherical sampling was initiated is marked by a shaded semicircle. The illumination components and the source origin are

1. The ceiling illuminated by the sun patch inside the room (solid rays)
2. The ceiling illuminated by the sun patch outside on the ground plane (dotted rays)
3. The floor inside the room illuminated by the sky glow (dashed rays)

It is important to appreciate the element of chance at work whenever hemispherical sampling is used. If the number of initial sampling rays (-ad) were set too small, the calculation might, for example, "miss" the sun patch even though it was "visible" from the point at which the rays were spawned. By the same token, an unrepresentative chance "hit" of a small sun patch by one of the sampling rays can produce a gross overestimate for indirect irradiance. In a rendering, the artifacts associated with ambient undersampling are all too apparent—bright and dark blotches. To avoid this, we need to set a sufficiently high value for the number of initial sampling rays.

Hemispherical sampling is generally too expensive to initiate at every surface visible from the eyepoint (that is, from every pixel). The calculation needs good indirect irradiance estimates from sampling at a limited number of locations. We then rely on the irradiance interpolation algorithm to estimate the in-between, or missing, values. To generate a fairly smooth rendering for the sunlit space, accounting for the first level of interreflection, we would need to set moderately high resolution values for the ambient parameters. To approximate the effect of the higher-level reflections, we should set a value for the -av parameter. In a later section (Visualizing a Highly Detailed Atrium Scene), we show how to obtain a good estimate for this parameter using rview. A rough guess, however, would be something in the range of 1/50 to 1/200 of the ground ambient value (obtained by executing the gensky command).[22] You may decide that `-ab 1` is insufficient to model the major light transfers, and that `-ab 2` is needed. In fact, this is almost certainly the case, because by using only one ambient bounce, we fail to account for the externally reflected component of *sky light*. This is likely to result in significant underestimation of the ceiling luminance near the window, since this part of the room has a good "view" of the (external) ground plane.

22. This range in percentage terms, 2% to 0.5%, corresponds approximately to the daylight factor about the middle of the room.

6.8.3 The Mkillum Approach

We can somewhat reduce the element of chance in our calculations for important light transfers by treating the window opening in a special way. The *Radiance* system allows you to select known sources of light (windows, skylights, and so on) and precompute light output distributions for them. They are then moved from the indirect (stochastic) calculation to the direct (deterministic) calculation. The program we use for this task is called mkillum. To illustrate the effectiveness of this approach, consider hemispherical sampling spawned at the rear wall of the room and also at the window plane. At the rear wall, the window subtends a solid angle that accounts for about 5% of the hemispherical "view" normal to the wall surface. Therefore, only about 5 in every 100 rays spawned from this point will directly sample the luminous environment through the window—even though we know the window to be the only "source" of illumination. The same sampling strategy at the window plane, however, will cause about half the rays to sample the sky and the remainder to sample the ground. This is how mkillum works; you direct the program to determine a light output distribution for the window based on the sampling of incident radiation and the glazing transmission properties. In any subsequent calculation or rendering, the glazing elements are treated as "secondary light sources." Note that mkillum can account only for the diffuse component of light that passes through the glazing; the direct and specular components are unaffected.

Mkillum parameters can be specified in the scene description file, but on first encountering the technique, you may prefer to control all aspects of the calculation from the command line. In this case, you must keep the window description, materials, and surfaces in a separate file. To create a scene octree with the modified window description usually requires three stages:

1. Prepare scene octree in the normal way.
2. Use mkillum to compute the light output distribution of named glazing elements, usually one or more polygons. On completion, the program will have created new window description(s) using a special light source material called *illum*. In addition, there will be data files, one for each illum surface, that contain the material's light output distribution.
3. Recreate the scene octree, replacing the original window description with the modified light source window.

The commands might be as follows:

```
% oconv room.rad window.rad sky.rad out.rad > scene.oct
% mkillum [rtrace options] scene.oct < window.rad > mkiwin.rad
% oconv room.rad mkiwin.rad sky.rad out.rad > mkiscene.oct
```

What rtrace settings you use will depend on which light transfers you think need to be modeled, and on the complexity of the external scene. A series of simplified ray diagrams[23] (Figure 6.14) shows what ab settings will account for these components of diffuse radiation incident at the window:

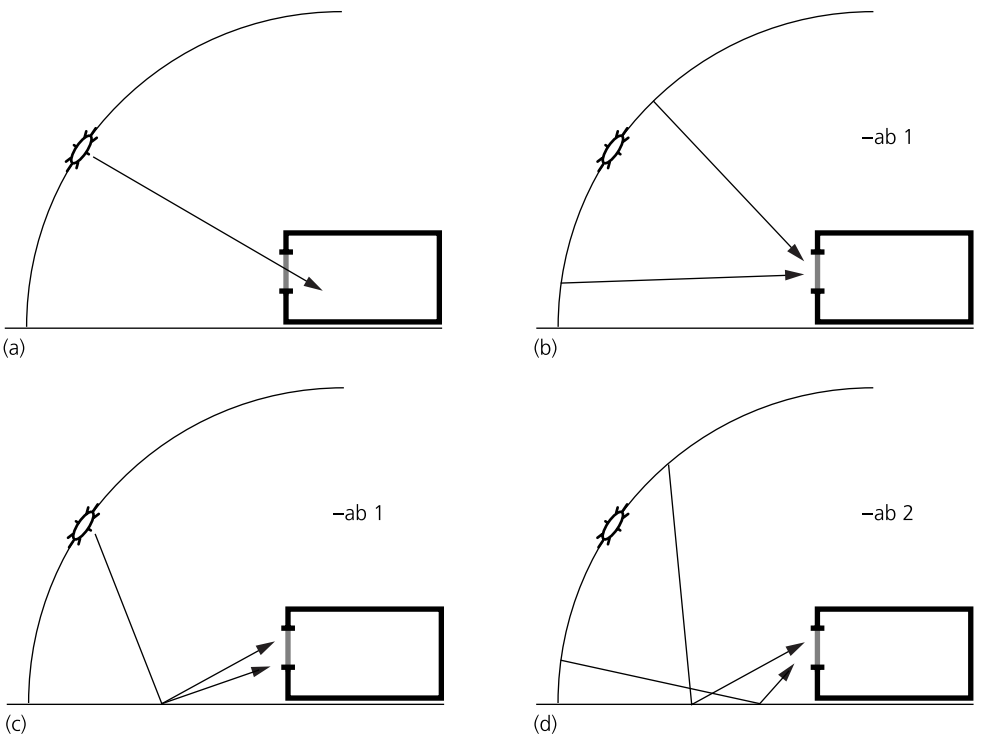

Figure 6.14 The direct solar component (a) is not accounted for by mkillum because it is part of the *Radiance* direct calculation. The direct sky component (b) is accounted for by mkillum, as is the indirect solar component (c), and the indirect sky component (d).

23. Here we ignore the fact that *Radiance* actually traces rays backward from the eye, and instead adopt the more intuitive convention that rays emanate from luminous sources.

- The diffuse component of light from the glow sky (b)
- The diffuse component of the first-order reflection of solar radiation from outside surfaces, for example the ground plane (c)
- The diffuse component of the first-order reflection of sky radiation from outside surfaces, for example the ground plane (d)

For the majority of scenes, setting –ab 2 is usually sufficient to account for most of the diffuse light transfer paths to a window

Ordinary *Radiance* light sources are opaque; if this were the case with the illum window, we would not be able to see through it. To avoid this, the illum sources have a dual nature. When treated in the direct component calculation, they behave like ordinary light sources, but when viewed directly, they revert to the original material description.

The mkillum approach requires a certain amount of user expertise to be implemented effectively for all but the simplest of cases. We therefore hope that if you are interested, you will take some time to familiarize yourself with the technique. The "Drafting Office" example in the *obj/virtual* subdirectory is a good place to start. The scene, devised by Greg Ward Larson, demonstrates fairly advanced use of the mkillum approach.

6.9 Visualizing a Highly Detailed Atrium Scene

Every design will present its own set of problems. With an ambitious project, even the experienced *Radiance* user is likely to chance upon one or more unforeseen difficulties. While these are undeniably frustrating at times, the possibility of discovering new techniques with *Radiance* usually serves to inspire the user—discovery is, after all, part of the fun. The visualization and analysis of a design known as the Foggo Atrium was one such project.

The IESD Center at De Montfort University, UK, was invited to participate in a case-studies design project for low-energy urban offices. The proposed design, by the architectural firm Peter Foggo Associates, was for a building that avoided air conditioning and made maximum use of daylight. The floor plan of the five-story building was fairly deep: 16.5 meters and 15 meters (upper two stories). The design would incorporate a linear atrium to provide core illumination. The lighting analysis brief called for both daylight factor prediction and visualization of the scene. The daylight factors were required to assess the effectiveness of external facade shading devices, and of the atrium as a provider of illumination. The images, on the other hand, were conceived to create a strong visual impression of what the design might look like.

A synthetic image of the atrium, Plate 16, shows the degree of complexity that was achieved for this model. The entire *Radiance* scene description was created from the command line. This task was not as horrendous as it might first appear. Once a basic office module had been worked up, it was easy to generate much of the structure using the repeated-transformation option in **xform**. In fact, the scene description consists of hierarchies of repeated transforms at various scales—for example, ceiling lights, a single office module, a row of office modules, and so on. For the daylight factors, however, a fully detailed office module was nested in a simple atrium model using the technique described in the section called DF Prediction: Tricks of the Trade.

6.9.1 Ambient Calculation Parameter Values

Having created the scene description, how do we go about selecting values for the ambient parameters? First, we need to decide what light transfers are needed to produce the major illumination components for the rendering we have in mind. This will depend to some degree on how we choose to illuminate the model, and on the view parameters. For open scenes, it is invariably the case that some direct solar illumination greatly enhances the impact of the rendering. Overcast-sky illumination looks dull and dreary in renderings and in real life. So we opt for a sunny sky description, in this case a CIE clear sky with sun. From what we know of the model geometry and orientation, we can decide on a viewpoint and make a good guess at the solar altitude and azimuth positions. A visual check with rview will tell us whether or not we have chosen well.

This atrium has numerous facade windows and many roof glazing elements. With so many potential sources of light, it would be very inefficient to calculate their contribution in the deterministic domain. Preprocessing of glazing elements to secondary light sources is therefore not advised for this type of building. Consequently, we will rely exclusively on the ambient calculation to model the interreflection.

The following sections show, step by step, how to make informed choices for ambient parameter values *before you begin any batch rendering*. Trial and error can be an instructive process. However, when, as here, the number of possibilities is nearly infinite, we need to drastically reduce the options before we do any exploring.

Setting -ab

Having settled on a view point and a sun position, we then set the ambient calculation parameters. The most important of these is, of course, the number of ambient bounces. We could go for a low-cost rendering and set `-ab 0`, but the final result, we know, would not be very convincing. At one ambient bounce, the sky and sun patch become potential sources of indirect illumination. At two ambient bounces, we have the potential to calculate indirect illumination for surfaces that have no direct line of sight to either sky or sun patch. This should be sufficient to give most of the surfaces that we can see a *calculated* diffuse irradiance. We approximate the effect of subsequent ambient bounces with a constant ambient value.

Setting -av and -aw

The constant ambient value option serves two functions. The first is to participate in the interreflection calculation, where it approximates the contribution of the higher-order reflections (see Chapter 12 for a description of the way this approximation is calculated). The other function is as sole provider of indirect illumination to surfaces excluded from the ambient calculation (see the Ambient Exclude/Include Options section, below). It usually pays to spend a moment or two to determine a "good" value for this parameter. With simple models, a value can sometimes be arrived at by analytical means. For the majority of scenes, however, it is more likely that you will need to base the estimate on calculated values. Here, we demonstrate how rview can be used to make a reasonable estimate for a constant ambient value. Where in the scene should we determine this value? The average radiance in the middle of the office floor at level 2 will be very different from the average radiance at the top of the elevator shaft. We decide by anticipating where in the scene the ambient calculation will expend the greatest effort. This is most likely to be for the office ceilings, many of which are visible from our viewpoint. Consequently, a "good" ambient value for the office spaces is what we should determine. This can be achieved in the following way:

1. Start the previewer rview with the irradiance option (**-i**) enabled, `-ab 1`, and maybe -ad set to higher than the default.
2. Wait a while for some detail to appear, then select a region in shade to refine (`frame` option). In this case, a bit of the ceiling at level 2 would be suitable.
3. After some further refinement, pick out and display the irradiance evaluated at a surface on the ceiling (use the `trace` option). We call this value I.
4. Recall that a uniform radiance that produces an irradiance, I, is simply I/π. (See Equation 6.8.)

Try this value (I/π) with the ambient bounces set to zero. Does it give similar indirect illumination for the same surface? If yes, this is the value to use.

The ambient weight parameter -aw, if enabled (i.e., -aw > 0), will modify the default ambient value in a moving average as new indirect irradiances are computed. This may produce more accurate renderings for scenes where the luminance extremes, and therefore the indirect contributions, are not too great. However, this is rarely the case for renderings with daylight, and it is usually safest to disable this option, setting `-aw 0`.

Setting -ad and -as

Having decided on values for -ab and -av, how do we go about setting the remaining ambient parameters? The sun patches on the floor and structure of the atrium will be significant sources of indirect illumination. To capture these potential sources, we should use a relatively large number of ambient divisions, in this case `-ad 1024`. Ambient supersampling should therefore be set to about one half or one quarter of this value.

Setting -aa and -ar

Our view of the atrium will reveal an enormous amount of fine-scale detail, for example the numerous ceiling lights and acoustic baffles. None of these objects is seen really close up, but we still want to calculate values for them rather than use a constant ambient approximation. Otherwise, we would not see, in the shading, the local illumination effect of the sun patch. Exact shading for each and every surface, however, is not really necessary; moderate irradiance interpolation errors over the scale size of a ceiling fixture should not be too conspicuous in the final image. Thus, a moderately accurate value should suffice. For this rendering, `-aa 0.3` was used.

Having settled on a value for -aa, we can base the ambient resolution on a minimum separation for indirect irradiance values in the cache. In other words, for distances less than this minimum, the calculation will always resort to interpolation, rather than initiate more sampling, regardless of the error estimate associated with that interpolation. This prevents the calculation from expending massive effort resolving irradiance gradients over negligible scales. Strictly speaking, this distance gives the scale at which the interpolation accuracy *begins* to deteriorate from the -aa setting. How do we decide on a magnitude for this scale? It often helps to evaluate this scale for a range of -ar and then to choose the value that gives the best

compromise between speed and accuracy. The scene dimension, D_{max}, is found from the scene octree using the -d option of **getinfo**. For this atrium, it was 99.2 meters. The minimum separation for cached irradiances, S_{min}, is given by

$$S_{min} = \frac{D_{max} \times \text{-aa}}{\text{-ar}} \qquad (6.12)$$

For `-aa 0.3`, the S_{min} for a range of -ar are given in Table 6.1. The third column gives the approximate relative cost of the calculation based on a minimum ambient resolution of 32. From these values, we can make a reasonably informed choice for -ar and anticipate the trade-off between accuracy and speed. For the minimum -ar listed, the potential exists for poor irradiance interpolation over scales of about 1 meter. These could be quite conspicuous from our view point, whereas (potentially) inaccurate shading over scales smaller than about 0.25 meter is far less likely to impair image quality. Higher resolution is of course possible, but at some cost. With this in mind, an ambient resolution of 128 seems a reasonable compromise.

S_{min} [m]	-ar	Relative cost
0.93	32	1
0.47	64	4
0.23	128	16
0.12	256	64

Table 6.1 Minimum separation and relative computational cost for a range of -ar settings.

Ambient Exclude/Include Options

Having set the parameters that control the computation of indirect irradiance, we should decide whether we want to exclude any materials from this calculation. Excluded materials will use the ambient value approximation directly, rather than a calculated indirect irradiance. Depending on the scene, we can make significant savings in rendering time by applying this option. How do we decide what to leave out? Exclusion criteria could be any of the following:

- Surfaces not visible from our view point (and unimportant in terms of light transfer)
- High-detail areas (the -ar parameter may already impose a partial restriction here)
- Surfaces that have a small diffuse reflectance (say, less than 5%)
- Surfaces that will appear very small in the final image
- "Sticks"—surfaces that will appear as thin lines in the final image

Some of the surfaces of the Foggo Atrium model that did not participate in the rendering ambient calculation (and the reasons for their exclusion) were

1. External facade detail including light shelf surfaces (not visible)
2. Window frames (sticks)
3. Windowsills (small)
4. Atrium roof vent slats (detail and small)
5. Atrium roof glazing bars (sticks)
6. Black handrail supports (low diffuse reflectance)

As we can see from the final image, Plate 16, these exclusions hardly detract from the quality of the rendering.

Note: It is easier to apply this technique if you segregate the materials into include and exclude types when you first construct the scene. In CAD terms, it helps to build up the model, layer by layer, with these requirements in mind.

Ambient File Use and the "Overture" Calculation

For a daylight rendering, the lion's share of the computation is invariably taken up by the ambient calculation. It makes sense, therefore, to save the cached indirect irradiance values to a file so they can be reused for later renderings. With a well-populated ambient file, it can be surprising how little time additional renderings take to complete, especially when there is significant overlap between views. There are rules that have to be observed when reusing ambient files. The most important of these rules is that you must always set the same combination of ambient parameters for every rendering that uses the ambient file. There is a special exception to this (see below). Also, the ambient exclude (or include) list should not change after the ambient file has been created.

Interpolation accuracy can be improved if the "presentation" (i.e., large) image is rendered using an already partially populated ambient file. The creation of the initial ambient file is known as an "overture" calculation. The ambient parameters values for the "overture" calculation should be those we have made the case for above. We use the same view parameters that are intended for the "presentation" image, but we generate the ambient file for a small picture size, no larger than, say, 64 by 64 pixels. We then reuse the ambient file to render a larger "presentation" image. The overall cost of the rendering will not be much greater than that of a one-pass approach, but the results can be significantly better.

Having created the ambient file with the "overture" calculation, you can, with caution, relax *some* of the ambient parameters for the larger renderings. The parameter revisions could be one or both of the following:

- Reduce -ad and -as by about 50%
- Slightly increase -aa (i.e., by 0.05 or 0.10)

The other ambient parameter settings should not be changed. If you do decide to change any of the -ad, -as, or -aa settings after the "overture" calculation, you should be aware that the modifications will not be reflected in the header of the ambient file. Thus, you need to track both the picture and the ambient file headers to obtain a complete record of the parameter settings for an image.

6.9.2 Batch Rendering

The ambient parameter values are set and we are ready to make the first rendering. Starting with the "overture" calculation, we generate a small image and save the ambient file. The "presentation" image we have in mind is a rendering at approximately the resolution of the monitor display: about 1000 pixels square. We rarely show images at the resolution at which they were rendered; alias artifacts always look unpleasant and greatly detract from the impression of realism. The highest quality is achieved by creating the rendering at two or three times the eventual size, then scaling it down using the **pfilt** program. We could go directly from the "overture" calculation to an (unfiltered) presentation image about 3000 pixels square. This is quite a leap and may take some time to render. In this case, we might prefer to reassure ourselves with an intermediate-sized image, say, 500 pixels square. This should provide sufficient detail for us to appraise the effectiveness of the ambient calculation. For certain scenes with multiple ambient bounces, you may find that it is the "overture" calculation that takes the longest, and that subsequent renderings, regardless of size, are completed relatively quickly. In this case, don't be too concerned if the "overture" calculation seems to be taking a long time to generate a small image.

Rendering time can be like kitchen cupboard space—it doesn't matter what you *need*, you always fill up what's *available*. It makes sense, therefore, to batch-render a series of images, say, overnight or over the weekend. Automate the rendering from shell scripts and keep track of the progress by setting the -e and -t options of rpict.

A Critical Appraisal of the Atrium Rendering

The viewpoint and lighting were chosen to create a striking impression rather than to show a typical view. The low view point was deliberately chosen to reveal specular reflections from the "terrazzo" floor and the nearby water feature. This effect is perhaps too exaggerated, and the floor itself has something of the appearance of calm water in a murky pool. It is in fact the uniformity of the floor that is the problem, rather than the specular component. If the floor had been divided up into slabs or tiles, and these given slightly different material properties, the final result would be much more convincing. If each tile had a small random component applied to its surface normal, giving us a slightly uneven floor, the rendering would be better still. These issues are related to material properties and to the way the model was constructed; what about the contribution of the indirect calculation?

In terms of overall impression, the diffuse shading looks pretty good. The indirect illumination effect of the sun patch is readily apparent, and the shading on the underside of the walkways between the elevator shafts is particularly realistic. At a finer level of detail, even individual ceiling fixtures don't look too bad, though there does appear to be some erroneously bright shading at the very smallest scales. Errors of this proportion were anticipated when we set the -aa and -ar parameters. On larger scales, we can see no evidence of light blotches, so our -ad and -as parameters were adequate for this scene.

6.9.3 Summary

From the limitless number of conceivable ambient parameter combinations, we have arrived at a set of values that we hope will either give acceptable results immediately or require only minor amendment. For each parameter, we have shown how the choice is influenced by the building design, the illumination, and the view point. The same approach could be applied to many architectural rendering problems.

However thoughtful our selection of ambient parameter values, we are unlikely to hit on the ideal combination that delivers the best compromise between speed and accuracy. Even if we stumbled across this magic combination, how would we know? Unless we tried out zillions of other combinations, we never would. Thus, we shouldn't worry about this too much. It is important, though, to have good ballpark values to begin with. Thereafter, we should be able to anticipate the effect, to a greater or lesser degree, of any subsequent parameter modifications. After all, our goal is to provide workable solutions to real-world problems.

6.10 Conclusion

Accurate simulation of the quantity and distribution of daylight in an architectural space is now a realistic prospect. The *Radiance* system can be used to predict illumination levels and visual appearance under daylight conditions for virtually any building design. In this chapter, we have looked at just some of the ways in which *Radiance* can be applied to solving daylight problems. We hope that daylight designers will find the techniques of value and use them to solve their own lighting problems. More important, we hope that the majority will be inspired to take a closer look at the system and the possibilities it offers.

Author's Biography

John Mardaljevic is a Research Fellow at the Institute of Energy and Sustainable Development (IESD), De Montfort University, Leicester, UK. He received his B.Sc. (1982) in physics and his M.Phil. (1988) in astrophysics, both from the University of Leicester. In 1990, he took up a research assistant post with De Montfort University. His first work there was on a project to assess dynamic thermal simulation programs for passive solar design. In 1991, he began to look into daylighting design tools for complex spaces, in particular atrium buildings. The *Radiance* system seemed particularly well suited to coping with modern atria: complex designs with a large number of specular or semispecular reflecting surfaces. As data from the International Daylight Measurement Program became available, the emphasis of Mardaljevic's work shifted toward validation and novel approaches to illuminance prediction—specifically, a comparison of sky model performance based on internal illuminance predictions and the formulation, implementation, and validation of the daylight coefficient approach for the *Radiance* system. This technique offers the potential for an efficient evaluation of the internal illuminance due to any sky condition by reusing precomputed illuminance values from a discretized sky. These studies form the basis of a recently completed Ph.D.

In addition to pure research, Mardaljevic has used *Radiance* to create renderings and to provide design advice for various architectural projects. To date, these have included atria (daylight factor and visualization), artificially lit offices, shading analysis (a preprocess for thermal simulation programs), and the evaluation of the visibility of a large-scale video display screen against daylight-produced glare.

From 1993 until his return to De Montfort in 1996, Mardaljevic worked as a research assistant at the University of Aberdeen, Scotland. Based at the Marine Laboratory, Aberdeen, he worked on oceanographic and ecosystem modeling projects. He has published papers on astrophysics, marine science, and illumination modeling. He is married and has a daughter.