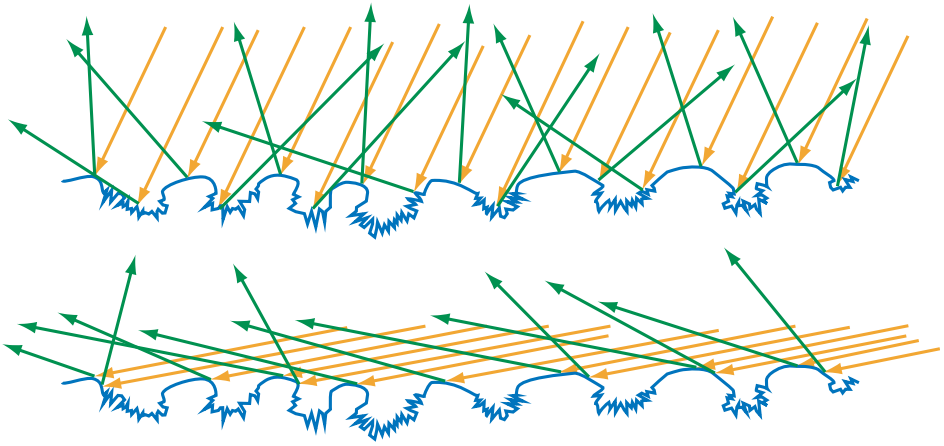

Figure 9.28. The microgeometry shown has a strong correlation between height and surface normal, where the raised areas are smooth and lower areas are rough. In the top image, the surface is illuminated from an angle close to the macroscopic surface normal. At this angle, the rough pits are accessible to many of the incoming light rays, and so many rays are scattered in different directions. In the bottom image, the surface is illuminated from a glancing angle. Shadowing blocks most of the pits, so few light rays hit them, and most rays are reflected from the smooth parts of the surface. In this case, the apparent roughness depends strongly on the angle of illumination.

length. Point your tube toward a brightly lit window or computer screen. With your view angle nearly parallel to the paper you will see a sharp reflection of the window or screen in the paper. The angle has to be extremely close to 90° to see the effect.

Light that was occluded by microscale surface detail does not disappear. It is reflected, possibly onto other microgeometry. Light may undergo multiple bounces in this way before it reaches the eye. Such interreflections are shown on the right side of Figure 9.27. Since the light is being attenuated by the Fresnel reflectance at each bounce, interreflections tend to be subtle in dielectrics. In metals, multiple-bounce reflection is the source of any visible diffuse reflection, since metals lack subsurface scattering. Multiple-bounce reflections from colored metals are more deeply colored than the primary reflection, since they are the result of light interacting with the surface multiple times.

So far we have discussed the effects of microgeometry on specular reflectance, i.e., the surface reflectance. In certain cases, microscale surface detail can affect subsurface reflectance as well. If the microgeometry irregularities are larger than the subsurface scattering distances, then shadowing and masking can cause a *retroreflection* effect, where light is preferentially reflected back toward the incoming direction. This effect occurs because shadowing and masking will occlude lit areas when the viewing and lighting directions differ greatly. See Figure 9.29. Retroreflection tends to give rough surfaces a flat appearance. See Figure 9.30.

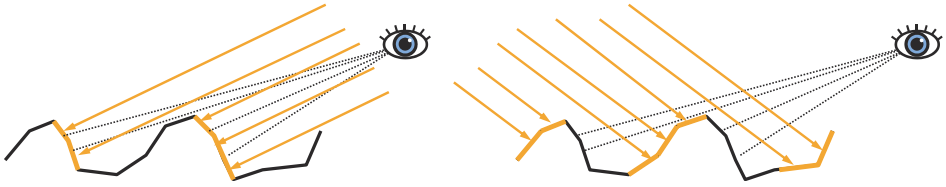

Figure 9.29. Retroreflection due to microscale roughness. Both figures show a rough surface with low Fresnel reflectance and high scattering albedo, so subsurface reflectance is visually important. On the left, the viewing and lighting directions are similar. The parts of the microgeometry that are brightly lit are also the ones that are most visible, leading to a bright appearance. On the right, the viewing and lighting directions differ greatly. In this case, the brightly lit areas are occluded from view and the visible areas are shadowed, leading to a darker appearance.

Figure 9.30. Photographs of two objects exhibiting non-Lambertian, retroreflective behavior due to microscale surface roughness. *(Photograph on the right courtesy of Peter-Pike Sloan.)*

9.7 Microfacet Theory

Many BRDF models are based on a mathematical analysis of the effects of microgeometry on reflectance called *microfacet theory*. This tool was first developed by researchers in the optics community [124]. It was introduced to computer graphics in 1977 by Blinn [159] and again in 1981 by Cook and Torrance [285]. The theory is based on the modeling of microgeometry as a collection of *microfacets*.

Each of these tiny facets is flat, with a single microfacet normal $\mathbf{m}$. The microfacets individually reflect light according to the micro-BRDF $f_\mu(\mathbf{l}, \mathbf{v}, \mathbf{m})$, with the combined reflectance across all the microfacets adding up to the overall surface BRDF. The usual choice is for each microfacet to be a perfect Fresnel mirror, resulting in a specular microfacet BRDF for modeling surface reflection. However, other choices are possible. Diffuse micro-BRDFs have been used to create several local subsurface scattering models [574, 657, 709, 1198, 1337]. A diffraction micro-BRDF was used to create a shading model combining geometrical and wave optics effects [763].

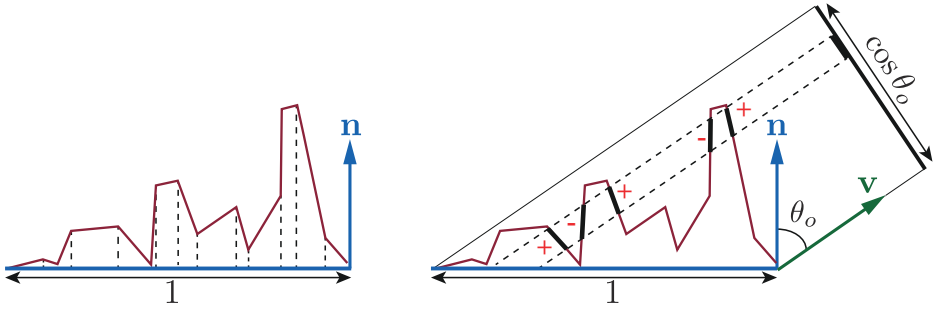

Figure 9.31. Side view of a microsurface. On the left, we see that integrating $D(\mathbf{m})(\mathbf{n} \cdot \mathbf{m})$, the microfacet areas projected onto the macrosurface plane, yields the area (length, in this side view) of the macrosurface, which is 1 by convention. On the right, integrating $D(\mathbf{m})(\mathbf{v} \cdot \mathbf{m})$, the microfacet areas projected onto the plane perpendicular to $\mathbf{v}$, yields the projection of the macrosurface onto that plane, which is $\cos\theta_o$ or $(\mathbf{v} \cdot \mathbf{n})$. When the projections of multiple microfacets overlap, the negative projected areas of the backfacing microfacets cancel out the "extra" frontfacing microfacets. *(After a figure by Matej Drame.)*

An important property of a microfacet model is the statistical distribution of the microfacet normals $\mathbf{m}$. This distribution is defined by the surface's *normal distribution function*, or NDF. Some references use the term *distribution of normals* to avoid confusion with the Gaussian normal distribution. We will use $D(\mathbf{m})$ to refer to the NDF in equations.

The NDF $D(\mathbf{m})$ is the statistical distribution of microfacet surface normals over the microgeometry surface area [708]. Integrating $D(\mathbf{m})$ over the entire sphere of microfacet normals gives the area of the microsurface. More usefully, integrating $D(\mathbf{m})(\mathbf{n} \cdot \mathbf{m})$, the projection of $D(\mathbf{m})$ onto the macrosurface plane, gives the area of the macrosurface patch that is equal to 1 by convention, as shown on the left side of Figure 9.31. In other words, the projection $D(\mathbf{m})(\mathbf{n} \cdot \mathbf{m})$ is normalized:

$$\int_{\mathbf{m} \in \Theta} D(\mathbf{m})(\mathbf{n} \cdot \mathbf{m}) d\mathbf{m} = 1. \tag{9.21}$$

The integral is over the entire sphere, represented here by Θ, unlike previous spherical integrals in this chapter that integrated over only the hemisphere centered on $\mathbf{n}$, represented by Ω. This notation is used in most graphics publications, though some references [708] use Ω to denote the complete sphere. In practice, most microstructure models used in graphics are heightfields, which means that $D(\mathbf{m}) = 0$ for all directions $\mathbf{m}$ outside Ω. However, Equation 9.21 is valid for non-heightfield microstructures as well.

More generally, the projections of the microsurface and macrosurface onto the plane perpendicular to any view direction $\mathbf{v}$ are equal:

$$\int_{\mathbf{m} \in \Theta} D(\mathbf{m})(\mathbf{v} \cdot \mathbf{m}) d\mathbf{m} = \mathbf{v} \cdot \mathbf{n}. \tag{9.22}$$

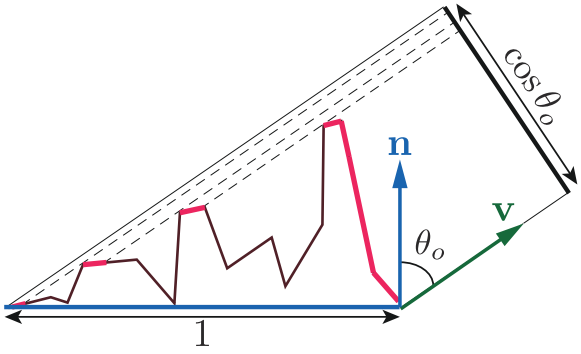

Figure 9.32. Integrating the projected areas of the visible microfacets (in bright red) yields the projected area of the macrosurface onto the plane perpendicular to $\mathbf{v}$.

The dot products in Equations 9.21 and 9.22 are not clamped to 0. The right side of Figure 9.31 shows why. Equations 9.21 and 9.22 impose constraints that the function $D(\mathbf{m})$ must obey to be a valid NDF.

Intuitively, the NDF is like a histogram of the microfacet normals. It has high values in directions where the microfacet normals are more likely to be pointing. Most surfaces have NDFs that show a strong peak at the macroscopic surface normal $\mathbf{n}$. Section 9.8.1 will cover several NDF models used in rendering.

Take a second look at the right side of Figure 9.31. Although there are many microfacets with overlapping projections, ultimately for rendering we care about only the visible microfacets, i.e., the microfacets that are closest to the camera in each overlapping set. This fact suggests an alternative way of relating the projected microfacet areas to the projected macrogeometry area: The sum of the projected areas of the visible microfacets is equal to the projected area of the macrosurface. We can express this mathematically by defining the *masking function* $G_1(\mathbf{m}, \mathbf{v})$, which gives the fraction of microfacets with normal $\mathbf{m}$ that are visible along the view vector $\mathbf{v}$. The integral of $G_1(\mathbf{m}, \mathbf{v})D(\mathbf{m})(\mathbf{v} \cdot \mathbf{m})^+$ over the sphere then gives the area of the macrosurface projected onto the plane perpendicular to $\mathbf{v}$:

$$\int_{\in\Theta} G_1(\mathbf{m}, \mathbf{v})D(\mathbf{m})(\mathbf{v} \cdot \mathbf{m})^+ d\mathbf{m} = \mathbf{v} \cdot \mathbf{n}, \tag{9.23}$$

as shown in Figure 9.32. Unlike Equation 9.22, the dot product in Equation 9.23 is clamped to zero. This operation is shown with the x^+ notation introduced in Section 1.2. Backfacing microfacets are not visible, so they are not counted in this case. The product $G_1(\mathbf{m}, \mathbf{v})D(\mathbf{m})$ is the *distribution of visible normals* [708].

While Equation 9.23 imposes a constraint on $G_1(\mathbf{m}, \mathbf{v})$, it does not uniquely determine it. There are infinite functions that satisfy the constraint for a given microfacet normal distribution $D(\mathbf{m})$ [708]. This is because $D(\mathbf{m})$ does not fully specify the mi-

crosurface. It tells us how many of the microfacets have normals pointing in certain directions, but not how they are arranged.

Although various G_1 functions have been proposed over the years, the dilemma as to which one to use has been solved (at least for now) in an excellent paper by Heitz [708]. Heitz discusses the Smith masking function, which was initially derived for Gaussian normal distributions [1665] and later generalized to arbitrary NDFs [202]. Heitz shows that out of the masking functions proposed in the literature, only two—the Smith function and the Torrance-Sparrow "V-cavity" function [1779]—obey Equation 9.23 and are thus mathematically valid. He further shows that the Smith function is a much closer match to the behavior of random microsurfaces than the Torrance-Sparrow function. Heitz also proves that the Smith masking function is the only possible function that both obeys Equation 9.23 and possesses the convenient property of *normal-masking independence*. This means that the value of $G_1(\mathbf{m}, \mathbf{v})$ does not depend on the direction of $\mathbf{m}$ as long as $\mathbf{m}$ is not backfacing i.e., as long as $\mathbf{m} \cdot \mathbf{v} \geq 0$. The Smith G_1 function has the following form:

$$G_1(\mathbf{m}, \mathbf{v}) = \frac{\chi^+(\mathbf{m} \cdot \mathbf{v})}{1 + \Lambda(\mathbf{v})}, \tag{9.24}$$

where $\chi^+(x)$ is the positive characteristic function

$$\chi^+(x) = \begin{cases} 1, & \text{where } x > 0, \\ 0, & \text{where } x \leq 0. \end{cases} \tag{9.25}$$

The Λ (lambda) function differs for each NDF. The procedure to derive Λ for a given NDF is described in publications by Walter et al. [1833] and Heitz [708].

The Smith masking function does have some drawbacks. From a theoretical standpoint, its requirements are not consistent with the structure of actual surfaces [708], and may even be physically impossible to realize [657]. From a practical standpoint, while it is quite accurate for random surfaces, its accuracy is expected to decrease for surfaces with a stronger dependency between normal direction and masking, such as the surface shown in Figure 9.28, especially if the surface has some repetitive structure (as do most fabrics). Nevertheless, until a better alternative is found, it is the best option for most rendering applications.

Given a microgeometry description including a micro-BRDF $f_\mu(\mathbf{l}, \mathbf{v}, \mathbf{m})$, normal distribution function $D(\mathbf{m})$, and masking function $G_1(\mathbf{m}, \mathbf{v})$, the overall macrosurface BRDF can be derived [708, 1833]:

$$f(\mathbf{l}, \mathbf{v}) = \int_{\mathbf{m} \in \Omega} f_\mu(\mathbf{l}, \mathbf{v}, \mathbf{m}) G_2(\mathbf{l}, \mathbf{v}, \mathbf{m}) D(\mathbf{m}) \frac{(\mathbf{m} \cdot \mathbf{l})^+}{|\mathbf{n} \cdot \mathbf{l}|} \frac{(\mathbf{m} \cdot \mathbf{v})^+}{|\mathbf{n} \cdot \mathbf{v}|} d\mathbf{m}. \tag{9.26}$$

This integral is over the hemisphere Ω centered on $\mathbf{n}$, to avoid collecting light contributions from under the surface. Instead of the masking function $G_1(\mathbf{m}, \mathbf{v})$, Equation 9.26 uses the *joint masking-shadowing function* $G_2(\mathbf{l}, \mathbf{v}, \mathbf{m})$. This function, derived from

G_1, gives the fraction of microfacets with normal $\mathbf{m}$ that are visible from two directions: the view vector $\mathbf{v}$ and the light vector $\mathbf{l}$. By including the G_2 function, Equation 9.26 enables the BRDF to account for masking as well as shadowing, but not for interreflection between microfacets (see Figure 9.27 on page 329). The lack of microfacet interreflection is a limitation shared by all BRDFs derived from Equation 9.26. Such BRDFs are somewhat too dark as a result. In Sections 9.8.2 and 9.9, we will discuss some methods that have been proposed to address this limitation.

Heitz [708] discusses several versions of the G_2 function. The simplest is the separable form, where masking and shadowing are evaluated separately using G_1 and multiplied together:

$$G_2(\mathbf{l}, \mathbf{v}, \mathbf{m}) = G_1(\mathbf{v}, \mathbf{m})G_1(\mathbf{l}, \mathbf{m}). \tag{9.27}$$

This form is equivalent to assuming that masking and shadowing are uncorrelated events. In reality they are not, and the assumption causes over-darkening in BRDFs using this form of G_2.

As an extreme example, consider the case when the view and light directions are the same. In this case G_2 should be equal to G_1, since none of the visible facets are shadowed, but with Equation 9.27 G_2 will be equal to G_1^2 instead.

If the microsurface is a heightfield, which is typically the case for microsurface models used in rendering, then whenever the relative azimuth angle ϕ between $\mathbf{v}$ and $\mathbf{l}$ is equal to 0°, $G_2(\mathbf{l}, \mathbf{v}, \mathbf{m})$ should be equal to $\min(G_1(\mathbf{v}, \mathbf{m}), G_1(\mathbf{l}, \mathbf{m}))$. See Figure 9.17 on page 311 for an illustration of ϕ. This relationship suggests a general way to account for correlation between masking and shadowing that can be used with any G_1 function:

$$G_2(\mathbf{l}, \mathbf{v}, \mathbf{m}) = \lambda(\phi)G_1(\mathbf{v}, \mathbf{m})G_1(\mathbf{l}, \mathbf{m}) + (1 - \lambda(\phi)) \min(G_1(\mathbf{v}, \mathbf{m}), G_1(\mathbf{l}, \mathbf{m})), \tag{9.28}$$

where $\lambda(\phi)$ is some function that increases from 0 to 1 as the angle ϕ increases. Ashikhmin et al. [78] suggested a Gaussian with a standard deviation of 15° (~ 0.26 radians):

$$\lambda(\phi) = 1 - e^{-7.3\phi^2}. \tag{9.29}$$

A different λ function was proposed by van Ginneken et al. [534]:

$$\lambda(\phi) = \frac{4.41\phi}{4.41\phi + 1}. \tag{9.30}$$

Regardless of the relative alignment of the light and view directions, there is another reason that masking and shadowing at a given surface point are correlated. Both are related to the point's height relative to the rest of the surface. The probability of masking increases for lower points, and so does the probability of shadowing. If the Smith masking function is used, this correlation can be precisely accounted for by the *Smith height-correlated masking-shadowing function*:

$$G_2(\mathbf{l}, \mathbf{v}, \mathbf{m}) = \frac{\chi^+(\mathbf{m} \cdot \mathbf{v})\chi^+(\mathbf{m} \cdot \mathbf{l})}{1 + \Lambda(\mathbf{v}) + \Lambda(\mathbf{l})}. \tag{9.31}$$

Heitz also describes a form of Smith G_2 that combines direction and height correlation:

$$G_2(\mathbf{l}, \mathbf{v}, \mathbf{m}) = \frac{\chi^+(\mathbf{m} \cdot \mathbf{v})\chi^+(\mathbf{m} \cdot \mathbf{l})}{1 + \max\left(\Lambda(\mathbf{v}), \Lambda(\mathbf{l})\right) + \lambda(\mathbf{v}, \mathbf{l}) \min\left(\Lambda(\mathbf{v}), \Lambda(\mathbf{l})\right)}, \tag{9.32}$$

where the function $\lambda(\mathbf{v}, \mathbf{l})$ could be an empirical function such as the ones in Equations 9.29 and 9.30, or one derived specifically for a given NDF [707].

Out of these alternatives, Heitz [708] recommends the height-correlated form of the Smith function (Equation 9.31) since it has a similar cost to the uncorrelated form and better accuracy. This form is the most widely used in practice [861, 947, 960], though some practitioners use the separable form (Equation 9.27) [214, 1937].

The general microfacet BRDF (Equation 9.26) is not used directly for rendering. It is used to derive a closed-form solution (exact or approximate) given a specific choice of micro-BRDF f_μ. The first example of this type of derivation will be shown in the next section.

9.8 BRDF Models for Surface Reflection

With few exceptions, the specular BRDF terms used in physically based rendering are derived from microfacet theory. In the case of specular surface reflection, each microfacet is a perfectly smooth Fresnel mirror. Recall that such mirrors reflect each incoming ray of light in a single reflected direction. This means that the micro-BRDF $f_\mu(\mathbf{l}, \mathbf{v}, \mathbf{m})$ for each facet is equal to zero unless $\mathbf{v}$ is parallel to the reflection of $\mathbf{l}$. For given $\mathbf{l}$ and $\mathbf{v}$ vectors, this configuration is equivalent to the case where the microfacet normal $\mathbf{m}$ is aligned with a vector pointing exactly halfway between $\mathbf{l}$ and $\mathbf{v}$. This vector is the *half vector* $\mathbf{h}$. See Figure 9.33. It is computed by adding $\mathbf{v}$ and $\mathbf{l}$ and normalizing the result:

$$\mathbf{h} = \frac{\mathbf{l} + \mathbf{v}}{||\mathbf{l} + \mathbf{v}||}. \tag{9.33}$$

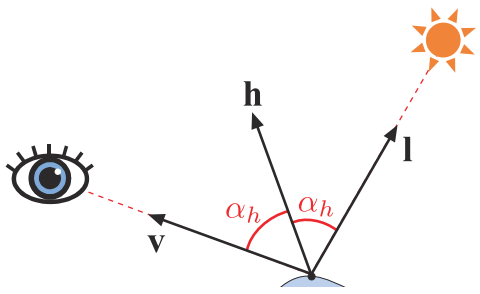

Figure 9.33. The half vector $\mathbf{h}$ forms equal angles (shown in red) with the light and view vectors.

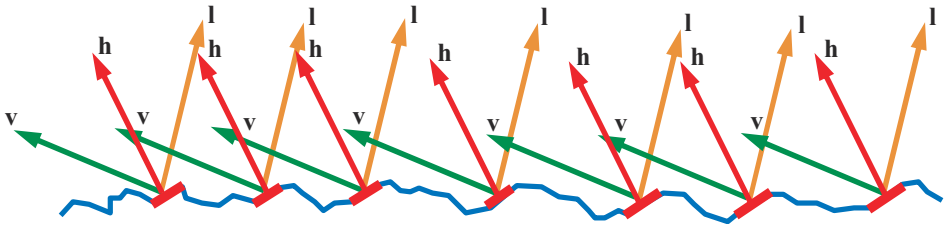

Figure 9.34. Surface composed of microfacets. Only the red microfacets, which have their surface normal aligned with the half vector $\mathbf{h}$, participate in the reflection of light from the incoming light vector $\mathbf{l}$ to the view vector $\mathbf{v}$.

When deriving a specular microfacet model from Equation 9.26, the fact that the Fresnel mirror micro-BRDF $f_\mu(\mathbf{l}, \mathbf{v}, \mathbf{m})$ is equal to zero for all $\mathbf{m} \neq \mathbf{h}$ is convenient, since it collapses the integral into an evaluation of the integrated function at $\mathbf{m} = \mathbf{h}$. Doing so yields the specular BRDF term

$$f_{\text{spec}}(\mathbf{l}, \mathbf{v}) = \frac{F(\mathbf{h}, \mathbf{l}) G_2(\mathbf{l}, \mathbf{v}, \mathbf{h}) D(\mathbf{h})}{4|\mathbf{n} \cdot \mathbf{l}||\mathbf{n} \cdot \mathbf{v}|}. \tag{9.34}$$

Details on the derivation can be found in publications by Walter et al. [1833], Heitz [708], and Hammon [657]. Hammon also shows a method to optimize the BRDF implementation by calculating $\mathbf{n} \cdot \mathbf{h}$ and $\mathbf{l} \cdot \mathbf{h}$ without calculating the vector $\mathbf{h}$ itself.

We use the notation f_{spec} for the BRDF term in Equation 9.34 to denote that it models only surface (specular) reflection. In a full BRDF, it will likely be paired with an additional term that models subsurface (diffuse) shading. To provide some intuition on Equation 9.34, consider that only those microfacets that happen to have their normals aligned with the half vector ($\mathbf{m} = \mathbf{h}$) are correctly oriented to reflect light from $\mathbf{l}$ into $\mathbf{v}$. See Figure 9.34. Thus, the amount of reflected light depends on the concentration of microfacets with normals equal to $\mathbf{h}$. This value is given by $D(\mathbf{h})$, the fraction of those microfacets that are visible from both the light and view directions, which is equal to $G_2(\mathbf{l}, \mathbf{v}, \mathbf{h})$, and the portion of light reflected by each of those microfacets, which is specified by $F(\mathbf{h}, \mathbf{l})$. In the evaluation of the Fresnel function, the vector $\mathbf{h}$ substitutes for the surface normal, e.g., when evaluating the Schlick approximation in Equation 9.16 on page 320.

The use of the half vector in the masking-shadowing function allows for a minor simplification. Since the angles involved can never be greater than 90°, the χ^+ terms in Equations 9.24, 9.31, and 9.32 can be removed.

9.8.1 Normal Distribution Functions

The normal distribution function has a significant effect on the appearance of the rendered surface. The shape of the NDF, as plotted on the sphere of microfacet normals, determines the width and shape of the cone of reflected rays (the specular

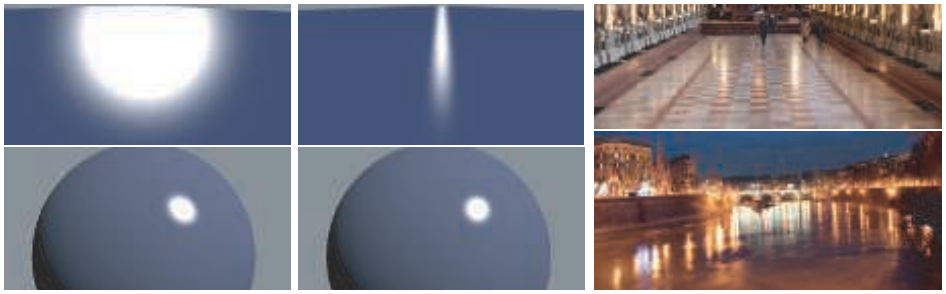

Figure 9.35. The images on the left are rendered with the non-physical Phong reflection model. This model's specular lobe is rotationally symmetrical around the reflection vector. Such BRDFs were often used in the early days of computer graphics. The images in the center are rendered with a physically based microfacet BRDF. The top left and center show a planar surface lit at a glancing angle. The top left shows an incorrect round highlight, while the center displays the characteristic highlight elongation on the microfacet BRDF. This center view matches reality, as shown in the photographs on the right. The difference in highlight shape is far more subtle on the sphere shown in the lower two rendered images, since in this case the surface curvature is the dominating factor for the highlight shape. *(Photographs courtesy of Elan Ruskin.)*

lobe), which in turn determines the size and shape of specular highlights. The NDF affects the overall perception of surface roughness, as well as more subtle visual aspects such as whether highlights have a distinct edge or are surrounded by haze.

However, the specular lobe is not a simple copy of the NDF shape. It, and thus the highlight shape, is distorted to a greater or lesser degree depending on surface curvature and view angle. This distortion is especially strong for flat surfaces viewed at glancing angles, as shown in Figure 9.35. Ngan et al. [1271] present an analysis of the reason behind this distortion.

Isotropic Normal Distribution Functions

Most NDFs used in rendering are *isotropic*—rotationally symmetrical about the macroscopic surface normal $\mathbf{n}$. In this case, the NDF is a function of just one variable, the angle θ_m between $\mathbf{n}$ and the microfacet normal $\mathbf{m}$. Ideally, the NDF can be written as expressions of $\cos\theta_m$ that can be computed efficiently as the dot product of $\mathbf{n}$ and $\mathbf{m}$.

The Beckmann NDF [124] was the normal distribution used in the first microfacet models developed by the optics community. It is still widely used in that community today. It is also the NDF chosen for the Cook-Torrance BRDF [285, 286]. The normalized Beckmann distribution has the following form:

$$D(\mathbf{m}) = \frac{\chi^+(\mathbf{n}\cdot\mathbf{m})}{\pi\alpha_b^2(\mathbf{n}\cdot\mathbf{m})^4}\exp\left(\frac{(\mathbf{n}\cdot\mathbf{m})^2-1}{\alpha_b^2(\mathbf{n}\cdot\mathbf{m})^2}\right). \tag{9.35}$$

The term $\chi^+(\mathbf{n}\cdot\mathbf{m})$ ensures that the value of the NDF is 0 for all microfacet normals

that point under the macrosurface. This property tells us that this NDF, like all the other NDFs we will discuss in this section, describes a heightfield microsurface. The α_b parameter controls the surface roughness. It is proportional to the root mean square (RMS) slope of the microgeometry surface, so $\alpha_b = 0$ represents a perfectly smooth surface.

To derive the Smith G_2 function for the Beckmann NDF, we need the corresponding Λ function, to plug into Equation 9.24 (if using the separable form of G_2), 9.31 (for the height correlated form), or 9.32 (for the direction and height correlated form).

The Beckmann NDF is *shape-invariant*, which simplifies the derivation of Λ. As defined by Heitz [708], an isotropic NDF is shape-invariant if the effect of its roughness parameter is equivalent to scaling (stretching) the microsurface. Shape-invariant NDFs can be written in the following form:

$$D(\mathbf{m}) = \frac{\chi^+(\mathbf{n}\cdot\mathbf{m})}{\alpha^2(\mathbf{n}\cdot\mathbf{m})^4}\, g\left(\frac{\sqrt{1-(\mathbf{n}\cdot\mathbf{m})^2}}{\alpha(\mathbf{n}\cdot\mathbf{m})}\right), \tag{9.36}$$

where g represents an arbitrary univariate function. For an arbitrary isotropic NDF, the Λ function depends on two variables. The first is the roughness α, and the second is the incidence angle of the vector ($\mathbf{v}$ or $\mathbf{l}$) for which Λ is computed. However, for a shape-invariant NDF, the Λ function depends only on the variable a:

$$a = \frac{\mathbf{n}\cdot\mathbf{s}}{\alpha\sqrt{1-(\mathbf{n}\cdot\mathbf{s})^2}}, \tag{9.37}$$

where $\mathbf{s}$ is a vector representing either $\mathbf{v}$ or $\mathbf{l}$. The fact that Λ depends on only one variable in this case is convenient for implementation. Univariate functions can be more easily fitted with approximating curves, and can be tabulated in one-dimensional arrays.

The Λ function for the Beckmann NDF is

$$\Lambda(a) = \frac{\mathrm{erf}(a)-1}{2} + \frac{1}{2a\sqrt{\pi}}\exp(-a^2). \tag{9.38}$$

Equation 9.38 is expensive to evaluate since it includes erf, the error function. For this reason, an approximation [1833] is typically used instead:

$$\Lambda(a) \approx \begin{cases} \frac{1-1.259a+0.396a^2}{3.535a+2.181a^2}, & \text{where } a < 1.6, \\ 0, & \text{where } a \geq 1.6. \end{cases} \tag{9.39}$$

The next NDF we will discuss is the Blinn-Phong NDF. It was widely used in computer graphics in the past, though in recent times it has been largely superseded by other distributions. The Blinn-Phong NDF is still used in cases where computation is at a premium (e.g., on mobile hardware) because it is less expensive to compute than the other NDFs discussed in this section.

The Blinn-Phong NDF was derived by Blinn [159] as a modification of the (non-physically based) Phong shading model [1414]:

$$D(\mathbf{m}) = \chi^+(\mathbf{n} \cdot \mathbf{m})\frac{\alpha_p + 2}{2\pi}(\mathbf{n} \cdot \mathbf{m})^{\alpha_p}. \tag{9.40}$$

The power α_p is the roughness parameter of the Phong NDF. High values represent smooth surfaces and low values represent rough ones. The values of α_p can go arbitrarily high for extremely smooth surfaces—a perfect mirror would require $\alpha_p = \infty$. A maximally random surface (uniform NDF) can be achieved by setting α_p to 0. The α_p parameter is not convenient to manipulate directly since its visual impact is highly nonuniform. Small numerical changes have large visual effects for small α_p values, but large values can be changed significantly without much visual impact. For this reason, α_p is typically derived from a user-manipulated parameter via a nonlinear mapping. For example, $\alpha_p = m^s$, where s is a parameter value between 0 and 1 and m is an upper bound for α_p in a given application. This mapping was used by several games, including *Call of Duty: Black Ops*, where m was set to a value of 8192 [998].

Such "interface mappings" are generally useful when the behavior of a BRDF parameter is not perceptually uniform. These mappings are used to interpret parameters set via sliders or painted in textures.

Equivalent values for the Beckmann and Blinn-Phong roughness parameters can be found using the relation $\alpha_p = 2\alpha_b^{-2} - 2$ [1833]. When the parameters are matched in this way, the two distributions are quite close, especially for relatively smooth surfaces, as can be seen in the upper left of Figure 9.36.

The Blinn-Phong NDF is not shape-invariant, and an analytic form does not exist for its Λ function. Walter et al. [1833] suggest using the Beckmann Λ function in conjunction with the $\alpha_p = 2\alpha_b^{-2} - 2$ parameter equivalence.

In the same 1977 paper [159] in which Blinn adapted the Phong shading function into a microfacet NDF, he proposed two other NDFs. Of these three distributions, Blinn recommended one derived by Trowbridge and Reitz [1788]. This recommendation was not widely heeded, but 30 years later the Trowbridge-Reitz distribution was independently rediscovered by Walter et al. [1833], who named it the *GGX distribution*. This time, the seed took root. Within a few years, adoption of the GGX distribution started spreading across the film [214, 1133] and game [861, 960] industries, and today it likely is the most often-used distribution in both. Blinn's recommendation appears to have been 30 years ahead of its time. Although "Trowbridge-Reitz distribution" is technically the correct name, we use the GGX name in this book since it is firmly established.

The GGX distribution is

$$D(\mathbf{m}) = \frac{\chi^+(\mathbf{n} \cdot \mathbf{m})\alpha_g^2}{\pi\left(1 + (\mathbf{n} \cdot \mathbf{m})^2\left(\alpha_g^2 - 1\right)\right)^2}. \tag{9.41}$$

The roughness control provided by the α_g parameter is similar to that provided by the Beckmann α_b parameter. In the Disney principled shading model, Burley [214] exposes

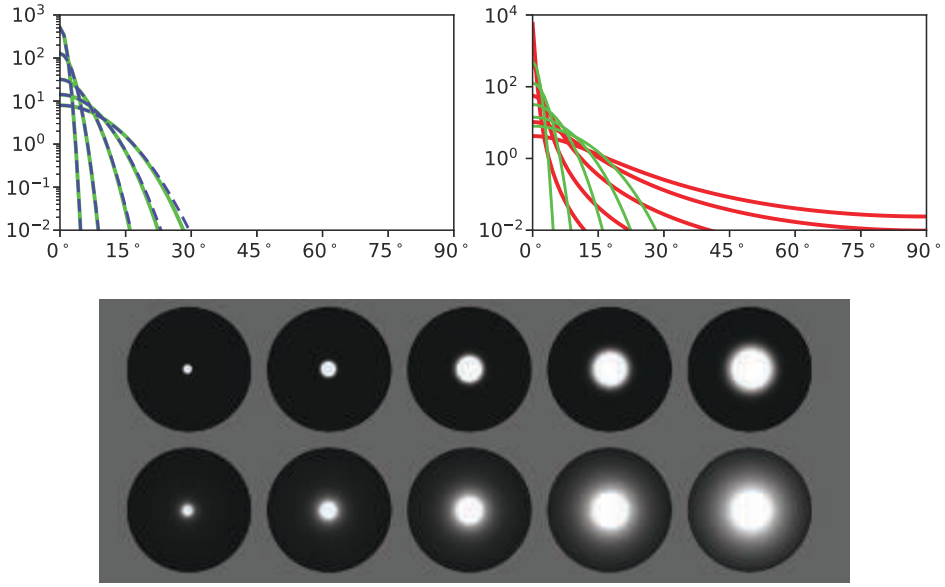

Figure 9.36. On the upper left, a comparison of Blinn-Phong (dashed blue) and Beckmann (green) distributions for values of α_b ranging from 0.025 to 0.2 (using the parameter relation $\alpha_p = 2\alpha_b^{-2} - 2$). On the upper right, a comparison of GGX (red) and Beckmann (green) distributions. The values of α_b are the same as in the left plot. The values of α_g have been adjusted by eye to match highlight size. These same values have been used in the spheres rendered on the bottom image. The top row uses the Beckmann NDF and the bottom row uses GGX.

the roughness control to users as $\alpha_g = r^2$, where r is the user-interface roughness parameter value between 0 and 1. Exposing r as a slider value means that the effect changes in a more linear fashion. This mapping has been adopted by most applications that use the GGX distribution.

The GGX distribution is shape-invariant, and its Λ function is relatively simple:

$$\Lambda(a) = \frac{-1 + \sqrt{1 + \frac{1}{a^2}}}{2}. \tag{9.42}$$

The fact that the variable a appears in Equation 9.42 only as a^2 is convenient, since the square root in Equation 9.37 can be avoided.

Due to the popularity of the GGX distribution and the Smith masking-shadowing function, there has been a focused effort to optimize the combination of the two. Lagarde observes [960] that the height-correlated Smith G_2 for GGX (Equation 9.31) has terms that cancel out when combined with the denominator of the specular microfacet BRDF (Equation 9.34). The combined term can be simplified thusly:

$$\frac{G_2(\mathbf{l}, \mathbf{v})}{4|\mathbf{n} \cdot \mathbf{l}||\mathbf{n} \cdot \mathbf{v}|} \Longrightarrow \frac{0.5}{\mu_o\sqrt{\alpha^2 + \mu_i(\mu_i - \alpha^2\mu_i)} + \mu_i\sqrt{\alpha^2 + \mu_o(\mu_o - \alpha^2\mu_o)}}. \tag{9.43}$$

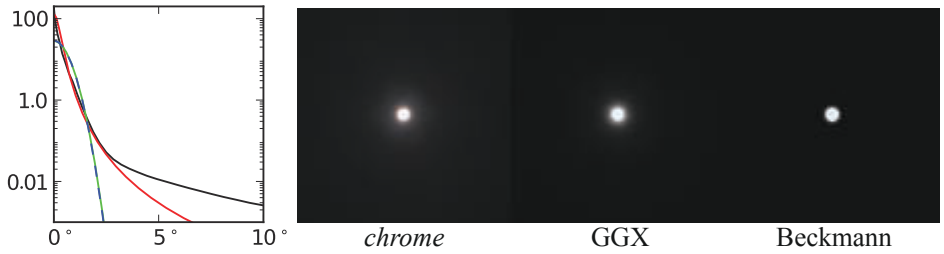

Figure 9.37. NDFs fit to measured chrome from the MERL database. On the left, we have plots of the specular peak against θ_m for chrome (black), GGX (red; $\alpha_g = 0.006$), Beckmann (green; $\alpha_b = 0.013$), and Blinn-Phong (blue dashes; $n = 12000$). Rendered highlights are shown on the right for chrome, GGX, and Beckmann. *(Figure courtesy of Brent Burley [214].)*

The equation uses the variable replacement $\mu_i = (\mathbf{n} \cdot \mathbf{l})^+$ and $\mu_o = (\mathbf{n} \cdot \mathbf{v})^+$ for brevity. Karis [861] proposes an approximated form of the Smith G_1 function for GGX:

$$G_1(\mathbf{s}) \approx \frac{2(\mathbf{n} \cdot \mathbf{s})}{(\mathbf{n} \cdot \mathbf{s})(2 - \alpha) + \alpha}, \tag{9.44}$$

where $\mathbf{s}$ can be replaced with either $\mathbf{l}$ or $\mathbf{v}$. Hammon [657] shows that this approximated form of G_1 leads to an efficient approximation for the combined term composed of the height-correlated Smith G_2 function and the specular microfacet BRDF denominator:

$$\frac{G_2(\mathbf{l}, \mathbf{v})}{4|\mathbf{n} \cdot \mathbf{l}||\mathbf{n} \cdot \mathbf{v}|} \approx \frac{0.5}{\text{lerp}\left(2|\mathbf{n} \cdot \mathbf{l}||\mathbf{n} \cdot \mathbf{v}|, |\mathbf{n} \cdot \mathbf{l}| + |\mathbf{n} \cdot \mathbf{v}|, \alpha\right)}, \tag{9.45}$$

which uses the linear interpolation operator, $\text{lerp}(x, y, s) = x(1 - s) + ys$.

When comparing the GGX and Beckmann distributions in Figure 9.36, it is apparent that the two have fundamentally different shapes. GGX has narrower peaks than Beckmann, as well as longer "tails" surrounding those peaks. In the rendered images at the bottom of the figure, we can see that GGX's longer tails create the appearance of a haze or glow around the core of the highlight.

Many real-world materials show similar hazy highlights, with tails that are typically longer even than those of the GGX distribution [214]. See Figure 9.37. This realization has been a strong contributor to the growing popularity of the GGX distribution, as well as the continuing search for new distributions that would fit measured materials even more accurately.

Burley [214] proposed the *generalized Trowbridge-Reitz* (GTR) NDF with a goal of allowing for more control over the NDF's shape, specifically the tails of the distribution:

$$D(\mathbf{m}) = \frac{k(\alpha, \gamma)}{\pi \left(1 + (\mathbf{n} \cdot \mathbf{m})^2 \left(\alpha_g^2 - 1\right)\right)^\gamma}. \tag{9.46}$$

The γ argument controls the tail shape. When $\gamma = 2$, GTR is the same as GGX. As the value of γ decreases, tails of the distribution become longer, and as it increases,

they become shorter. At high values of γ, the GTR distribution resembles Beckmann. The $k(\alpha, \gamma)$ term is the normalization factor, which we give in a separate equation, since it is more complicated than those of other NDFs:

$$k(\alpha,\gamma) = \begin{cases} \frac{(\gamma-1)\left(\alpha^2-1\right)}{\left(1-(\alpha^2)^{(1-\gamma)}\right)}, & \text{where } \gamma \neq 1 \text{ and } \alpha \neq 1, \\ \frac{\left(\alpha^2-1\right)}{\ln(\alpha^2)}, & \text{where } \gamma = 1 \text{ and } \alpha \neq 1, \\ 1, & \text{where } \alpha = 1. \end{cases} \tag{9.47}$$

The GTR distribution is not shape-invariant, which complicates finding its Smith G_2 masking-shadowing function. It took three years after publication of the NDF for a solution for G_2 to be published [355]. This G_2 solution is quite complex, with a table of analytical solutions for certain values of γ (for intermediate values, interpolation must be used). Another issue with GTR is that the parameters α and γ affect the perceived roughness and "glow" in a non-intuitive manner.

Student's t-distribution (STD) [1491] and *exponential power distribution* (EPD) [763] NDFs include shape control parameters. In contrast to GTR, these functions are shape-invariant with respect to their roughness parameters. At the time of writing, these are newly published, so it is not clear whether they will find use in applications.

Instead of increasing the complexity of the NDF, an alternative solution to better matching measured materials is to use multiple specular lobes. This idea was suggested by Cook and Torrance [285, 286]. It was experimentally tested by Ngan [1271], who found that for many materials adding a second lobe did improve the fit significantly. Pixar's *PxrSurface* material [732] has a "roughspecular" lobe that is intended to be used (in conjunction with the main specular lobe) for this purpose. The additional lobe is a full specular microfacet BRDF with all the associated parameters and terms. Imageworks employs a more surgical approach [947], using a mix of two GGX NDFs that are exposed to the user as an extended NDF, rather than an entire separate specular BRDF term. In this case, the only additional parameters needed are a second roughness value and a blend amount.

Anisotropic Normal Distribution Functions

While most materials have isotropic surface statistics, some have significant anisotropy in their microstructure that noticeably affects their appearance, e.g., Figure 9.26 on page 329. To accurately render such materials, we need BRDFs, especially NDFs that are anisotropic as well.

Unlike isotropic NDFs, anisotropic NDFs cannot be evaluated with just the angle θ_m. Additional orientation information is needed. In the general case, the microfacet normal $\mathbf{m}$ needs to be transformed into the *local frame* or *tangent space* defined by the normal, tangent, and bitangent vectors, respectively, $\mathbf{n}$, $\mathbf{t}$, and $\mathbf{b}$. See Figure 6.32 on page 210. In practice, this transformation is typically expressed as three separate dot products: $\mathbf{m} \cdot \mathbf{n}$, $\mathbf{m} \cdot \mathbf{t}$, and $\mathbf{m} \cdot \mathbf{b}$.

When combining normal mapping with anisotropic BRDFs, it is important to make sure that the normal map perturbs the tangent and bitangent vectors as well as the normal. This procedure is often done by applying the *modified Gram-Schmidt* process to the perturbed normal $\mathbf{n}$ and the interpolated vertex tangent and bitangent vectors $\mathbf{t}_0$ and $\mathbf{b}_0$ (the below assumes that $\mathbf{n}$ is already normalized):

$$\begin{aligned} \mathbf{t}' &= \mathbf{t}_0 - (\mathbf{t}_0 \cdot \mathbf{n})\mathbf{n} \quad \Longrightarrow \mathbf{t} = \frac{\mathbf{t}'}{\|\mathbf{t}'\|}, \\ \left.\begin{aligned} \mathbf{b}' &= \mathbf{b}_0 - (\mathbf{b}_0 \cdot \mathbf{n})\mathbf{n}, \\ \mathbf{b}'' &= \mathbf{b}' - (\mathbf{b}' \cdot \mathbf{t})\mathbf{t} \end{aligned}\right\} &\Longrightarrow \mathbf{b} = \frac{\mathbf{b}''}{\|\mathbf{b}''\|}. \end{aligned} \tag{9.48}$$

Alternatively, after the first line the orthogonal $\mathbf{b}$ vector could be created by taking the cross product of $\mathbf{n}$ and $\mathbf{t}$.

For effects such as brushed metal or curly hair, per-pixel modification of the tangent direction is needed, typically provided by a *tangent map*. This map is a texture that stores the per-pixel tangent, similar to how a normal map stores per-pixel normals. Tangent maps most often store the two-dimensional projection of the tangent vector onto the plane perpendicular to the normal. This representation works well with texture filtering and can be compressed similar to normal maps. Some applications store a scalar rotation amount instead, which is used to rotate the tangent vector around $\mathbf{n}$. Though this representation is more compact, it is prone to texture filtering artifacts where the rotation angle wraps around from 360° to 0°.

A common approach to creating an anisotropic NDF is to generalize an existing isotropic NDF. The general approach used can be applied to any shape-invariant isotropic NDF [708], which is another reason why shape-invariant NDFs are preferable. Recall that isotropic shape-invariant NDFs can be written in the following form:

$$D(\mathbf{m}) = \frac{\chi^+(\mathbf{n} \cdot \mathbf{m})}{\alpha^2(\mathbf{n} \cdot \mathbf{m})^4}\, g\left(\frac{\sqrt{1 - (\mathbf{n} \cdot \mathbf{m})^2}}{\alpha(\mathbf{n} \cdot \mathbf{m})}\right), \tag{9.49}$$

with g representing a one-dimensional function that expresses the shape of the NDF. The anisotropic version is

$$D(\mathbf{m}) = \frac{\chi^+(\mathbf{n} \cdot \mathbf{m})}{\alpha_x \alpha_y (\mathbf{n} \cdot \mathbf{m})^4}\, g\left(\frac{\sqrt{\frac{(\mathbf{t} \cdot \mathbf{m})^2}{\alpha_x^2} + \frac{(\mathbf{b} \cdot \mathbf{m})^2}{\alpha_y^2}}}{(\mathbf{n} \cdot \mathbf{m})}\right). \tag{9.50}$$

The parameters α_x and α_y represent the roughness along the direction of $\mathbf{t}$ and $\mathbf{b}$, respectively. If $\alpha_x = \alpha_y$, Equation 9.50 reduces back to the isotropic form.

The G_2 masking-shadowing function for the anisotropic NDF is the same as the isotropic one, except that the variable a (passed into the Λ function) is calculated differently:

$$a = \frac{\mathbf{n} \cdot \mathbf{s}}{\sqrt{\alpha_x^2(\mathbf{t} \cdot \mathbf{s})^2 + \alpha_y^2(\mathbf{b} \cdot \mathbf{s})^2}}, \tag{9.51}$$

where (as in Equation 9.37) $\mathbf{s}$ represents either $\mathbf{v}$ or $\mathbf{l}$.

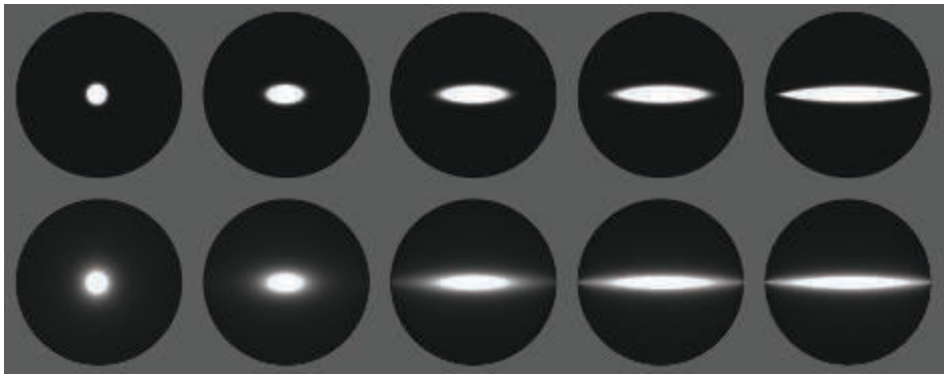

Figure 9.38. Spheres rendered with anisotropic NDFs: Beckmann in the top row and GGX in the bottom row. In both rows α_y is held constant and α_x is increased from left to right.

Using this method, anisotropic versions have been derived for the Beckmann NDF,

$$D(\mathbf{m}) = \frac{\chi^+(\mathbf{n}\cdot\mathbf{m})}{\pi\alpha_x\alpha_y(\mathbf{n}\cdot\mathbf{m})^4}\exp\left(-\frac{\frac{(\mathbf{t}\cdot\mathbf{m})^2}{\alpha_x^2}+\frac{(\mathbf{b}\cdot\mathbf{m})^2}{\alpha_y^2}}{(\mathbf{n}\cdot\mathbf{m})^2}\right), \tag{9.52}$$

and the GGX NDF,

$$D(\mathbf{m}) = \frac{\chi^+(\mathbf{n}\cdot\mathbf{m})}{\pi\alpha_x\alpha_y\left(\frac{(\mathbf{t}\cdot\mathbf{m})^2}{\alpha_x^2}+\frac{(\mathbf{b}\cdot\mathbf{m})^2}{\alpha_y^2}+(\mathbf{n}\cdot\mathbf{m})^2\right)^2}. \tag{9.53}$$

Both are shown in Figure 9.38.

While the most straightforward way to parameterize anisotropic NDFs is to use the isotropic roughness parameterization twice, once for α_x and once for α_y, other parameterizations are sometimes used. In the Disney principled shading model [214], the isotropic roughness parameter r is combined with a second scalar parameter k_{aniso} with a range of $[0, 1]$. The α_x and α_y values are computed from these parameters thusly:

$$\begin{aligned} k_{\text{aspect}} &= \sqrt{1-0.9\,k_{\text{aniso}}}, \\ \alpha_x &= \frac{r^2}{k_{\text{aspect}}}, \\ \alpha_y &= r^2\,k_{\text{aspect}}. \end{aligned} \tag{9.54}$$

The 0.9 factor limits the aspect ratio to $10:1$.

Imageworks [947] use a different parameterization that allows for an arbitrary degree of anisotropy:

$$\begin{aligned} \alpha_x &= r^2\,(1 + k_{\text{aniso}}), \\ \alpha_y &= r^2\,(1 - k_{\text{aniso}}). \end{aligned} \tag{9.55}$$

9.8.2 Multiple-Bounce Surface Reflection

As mentioned earlier in Section 9.7, the microfacet BRDF framework does not account for light that is reflected ("bounced") from the microsurface multiple times. This simplification causes some energy loss and over-darkening, especially for rough metals [712].

A technique used by Imageworks [947] combines elements from previous work [811, 878] to create a term that can be added to the BRDF to simulate multiple-bounce surface reflection:

$$f_{\text{ms}}(\mathbf{l}, \mathbf{v}) = \frac{\overline{F}\,\overline{R_{\text{sF1}}}}{\pi(1 - \overline{R_{\text{sF1}}})\big(1 - \overline{F}(1 - \overline{R_{\text{sF1}}})\big)}\big(1 - R_{\text{sF1}}(\mathbf{l})\big)\big(1 - R_{\text{sF1}}(\mathbf{v})\big), \tag{9.56}$$

where R_{sF1} is the directional albedo (Section 9.3) of f_{sF1}, which is the specular BRDF term with F_0 set to 1. The function R_{sF1} depends on the roughness α and elevation angle θ. It is relatively smooth, so it can be precomputed numerically (using Equation 9.8 or 9.9) and stored in a small two-dimensional texture. Imageworks found that 32×32 resolution is sufficient.

The function $\overline{R_{\text{sF1}}}$ is the cosine-weighted average value of R_{sF1} over the hemisphere. It depends only on α, so it can be stored in a one-dimensional texture, or an inexpensive curve could be fitted to the data. Since R_{sF1} is rotationally symmetric about $\mathbf{n}$, $\overline{R_{\text{sF1}}}$ can be computed with a one-dimensional integral. We also use the change of variables $\mu = \cos\theta$ (see Equation 9.6 on page 312):

$$\begin{aligned} \overline{R_{\text{sF1}}} &= \frac{\int_{\mathbf{s}\in\Omega} R_{\text{sF1}}(\mathbf{s})(\mathbf{n}\cdot\mathbf{s})d\mathbf{s}}{\int_{\mathbf{s}\in\Omega}(\mathbf{n}\cdot\mathbf{s})d\mathbf{s}} = \frac{1}{\pi}\int_{\phi=0}^{2\pi}\int_{\mu=0}^{1} R_{\text{sF1}}(\mu)\,\mu\,d\mu\,d\phi \\ &= 2\int_{\mu=0}^{1} R_{\text{sF1}}(\mu)\,\mu\,d\mu. \end{aligned} \tag{9.57}$$

Finally, $\overline{F}$ is the cosine-weighted average of the Fresnel term, computed in the same way:

$$\overline{F} = 2\int_{\mu=0}^{1} F(\mu)\,\mu\,d\mu. \tag{9.58}$$

Imageworks provide a closed-form solution to Equation 9.58 in the case that the generalized Schlick form (Equation 9.18) is used for F:

$$\overline{F} = \frac{2p^2 F_{90} + (3p+1)F_0}{2p^2 + 3p + 1}. \tag{9.59}$$

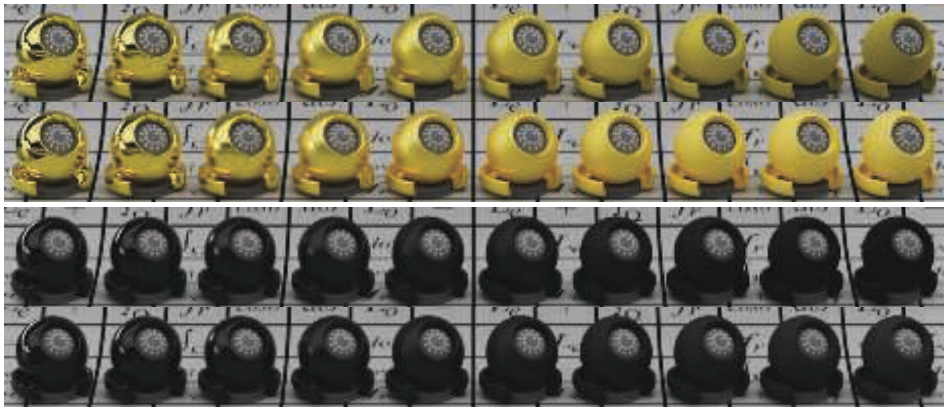

Figure 9.39. In all rows the roughness of the surface increases from left to right. The top two rows show a gold material. The first row is rendered without the Imageworks multiple-bounce term, and the second is rendered with the multiple-bounce term. The difference is most noticeable for the rougher spheres. The next two rows show a black dielectric material. The third row is rendered without the multiple-bounce term, and the fourth row has the multiple-bounce term applied. Here the difference is more subtle, since the specular reflectance is much lower. *(Figure courtesy of Christopher Kulla [947].)*

If the original Schlick approximation is used (Equation 9.16), then the solution simplifies to

$$\overline{F} = \frac{20}{21}F_0 + \frac{1}{21}. \tag{9.60}$$

In the case of anisotropy, Imageworks use an intermediate roughness between α_x and α_y for the purpose of computing f_{ms}. This approximation avoids the need to increase the dimensionality of the R_{sF1} lookup table, and the errors it introduces are small.

The results of the Imageworks multiple-bounce specular term can be seen in Figure 9.39.

9.9 BRDF Models for Subsurface Scattering

In the previous section we discussed surface, or specular, reflection. In this section we will discuss the other side of the issue, namely what happens to light refracted under the surface. As we discussed in Section 9.1.4, this light undergoes some combination of scattering and absorption, and part of it is re-emitted back out of the original surface. We will focus here on BRDF models for local subsurface scattering, or diffuse surface response, in opaque dielectrics. Metals are irrelevant, since they do not have any significant subsurface light interaction. Dielectric materials that are transparent or exhibit global subsurface scattering will be covered in Chapter 14.

We start our discussion of diffuse models with a section on the property of diffuse color and the possible values this color can have in real-world materials. In the following subsection we explain the effect of surface roughness on diffuse shading, and the criteria for choosing whether to use a smooth-surface or rough-surface shading model for a given material. The last two subsections are devoted to the smooth-surface and rough-surface models themselves.

9.9.1 Subsurface Albedo

The subsurface albedo ρ_{ss} of an opaque dielectric is the ratio between the energy of the light that escapes a surface compared to the energy of the light entering into the interior of the material. The value of ρ_{ss} is between 0 (all light is absorbed) and 1 (no light is absorbed) and can depend on wavelength, so ρ_{ss} is modeled as an RGB vector for rendering. For authoring, ρ_{ss} is often referred to as the diffuse color of the surface, just as the normal-incidence Fresnel reflectance F_0 is typically referred to as the specular color. The subsurface albedo is closely related to the scattering albedo discussed in Section 14.1.

Since dielectrics transmit most incoming light rather than reflecting it at the surface, the subsurface albedo ρ_{ss} is usually brighter and thus more visually important than the specular color F_0. Since it results from a different physical process than the specular color—absorption in the interior instead of Fresnel reflectance at the surface—ρ_{ss} typically has a different spectral distribution (and thus RGB color) than F_0. For example, colored plastic is composed of a clear, transparent substrate with pigment particles embedded in its interior. Light reflecting specularly will be uncolored, while light reflecting diffusely will be colored from absorption by the pigment particles; for example, a red plastic ball has a white highlight.

Subsurface albedo can be thought of as the result of a "race" between absorption and scattering—will the light be absorbed before it has had a chance to be scattered back out of the object? This is why foam on a liquid is much brighter than the liquid itself. The process of frothing does not change the absorptivity of the liquid, but the addition of numerous air-liquid interfaces greatly increases the amount of scattering. This causes most of the incoming light to be scattered before it has been absorbed, resulting in a high subsurface albedo and bright appearance. Fresh snow is another example of a substance with a high albedo. There is considerable scattering in the interfaces between snow granules and air, but little absorption, leading to a subsurface albedo of 0.8 or more across the visible spectrum. White paint is slightly less, about 0.7. Many substances encountered in daily life, such as concrete, stone, and soil, average between 0.15 and 0.4. Coal is an example of a material with extremely low subsurface albedo, close to 0.0.

The process by which many materials become darker when wet is the inverse of the liquid froth example. If the material is porous, water penetrates into spaces formerly filled with air. Dielectric materials have an index of refraction that is much closer to water than to air. This decrease in the relative index of refraction decreases the

scattering inside the material, and light travels longer distances (on average) before escaping the material. This change causes more light to be absorbed and the subsurface albedo becomes darker [821].

It is a common misconception (even reflected in well-regarded material authoring guidelines [1163]) that values of ρ_{ss} should never go below a lower limit of about 0.015–0.03 (30–50 in 8-bit nonlinear sRGB encoding) for realistic material authoring. However, this lower limit is based on color measurements that include surface (specular) as well as subsurface (diffuse) reflectance, and is thus too high. Actual materials can have lower values. For example, the Federal specification for the "OSHA Black" paint standard [524] has a Y value of 0.35 (out of 100). Given the measurement conditions and surface gloss, this Y corresponds to a ρ_{ss} value of about 0.0035 (11 in 8-bit nonlinear sRGB encoding).

When acquiring spot values or textures for ρ_{ss} from real-world surfaces, it is important to separate out the specular reflectance. This extraction can be done via careful use of controlled lighting and polarization filters [251, 952]. For accurate color, calibration should be performed as well [1153].

Not every RGB triple represents a plausible (or even physically possible) value for ρ_{ss}. Reflectance spectra are more restricted than emissive spectral power distributions: They can never exceed a value of 1 for any wavelength, and they are typically quite smooth. These limitations define a volume in color space that contains all plausible RGB values for ρ_{ss}. Even the relatively small sRGB color gamut contains colors outside this volume, so care must be taken when setting values for ρ_{ss} to avoid specifying unnaturally saturated and bright colors. Besides reducing realism, such colors can cause over-bright secondary reflections when precomputing global illumination (Section 11.5.1). The 2015 paper by Meng et al. [1199] is a good reference for this topic.

9.9.2 Scale of Subsurface Scattering and Roughness

Some BRDF models for local subsurface scattering take account of surface roughness—typically by using microfacet theory with a diffuse micro-BRDF f_μ—and some do not. The deciding factor for which type of model to use is not simply how rough the surface is, though this is a common misconception. The correct deciding factor relates to the relative size of the surface irregularities and the subsurface scattering distances.

See Figure 9.40. If the microgeometry irregularities are larger than the subsurface scattering distances (top left of figure), then the subsurface scattering will exhibit microgeometry-related effects such as retroreflection (Figure 9.29 on page 331). For such surfaces a rough-surface diffuse model should be used. As mentioned above, such models are typically based on microfacet theory, with the subsurface scattering treated as local to each microfacet, thus only affecting the micro-BRDF f_μ.

If the scattering distances are all larger than the irregularities (top right of Figure 9.40), then the surface should be considered flat for the purpose of modeling subsurface scattering, and effects such as retroreflection will not occur. Subsurface

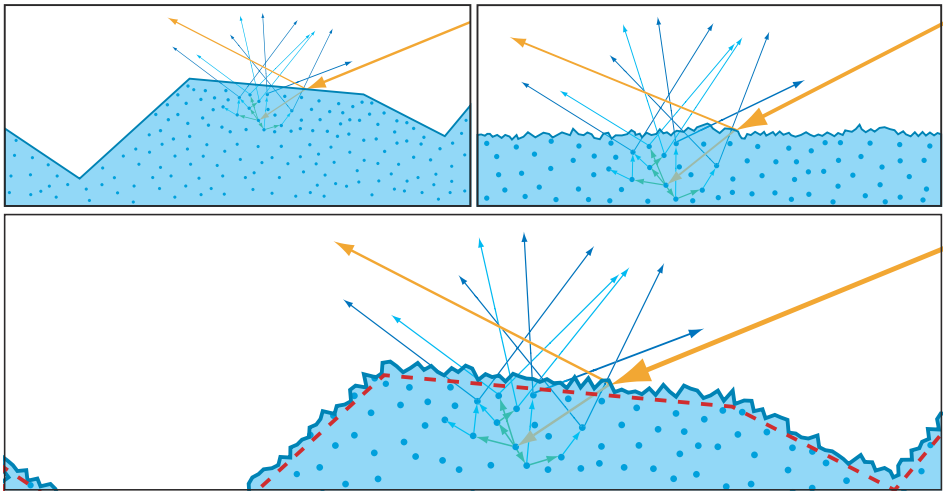

Figure 9.40. Three surfaces with similar NDFs but different relationships between the scale of the microgeometry and the subsurface scattering distances. On the top left, the subsurface scattering distances are smaller than the surface irregularities. On the top right, scattering distances are larger than the surface irregularities. The bottom figure shows a microsurface with roughness at multiple scales. The dashed red line represents the effective surface that only contains microstructure larger than the subsurface scattering distances.

scattering is not local to a microfacet, and cannot be modeled via microfacet theory. In this case, a smooth-surface diffuse model should be used.

In the intermediate case where the surface has roughness at scales both larger and smaller than the scattering distances, then a rough-surface diffuse model should be used, but with an *effective surface* that includes only irregularities larger than the scattering distances. Both diffuse and specular reflectance can be modeled with microfacet theory, but each with a different roughness value. The specular term will use a value based on the roughness of the actual surface, and the diffuse term will use a lower value, based on the roughness of the effective surface.

The scale of observation also ties into this, since it determines the definition of "microgeometry." For example, the moon is often cited as a case where rough-surface diffuse models should be used, since it exhibits significant retroreflection. When we look at the moon from the earth, the scale of observation is such that even a five-foot boulder is "microgeometry." Thus it is not surprising that we observe rough-surface diffuse effects such as retroreflection.

9.9.3 Smooth-Surface Subsurface Models

Here we will discuss smooth-surface subsurface models. These are appropriate for modeling materials where the surface irregularities are smaller than the subsurface scattering distances. Diffuse shading is not directly affected by surface roughness in

such materials. If the diffuse and specular terms are coupled, which is the case for some of the models in this section, then surface roughness may affect diffuse shading indirectly.

As mentioned in Section 9.3, real-time rendering applications often model local subsurface scattering with a Lambertian term. In this case the BRDF diffuse term is ρ_{ss} over π:

$$f_{diff}(\mathbf{l}, \mathbf{v}) = \frac{\rho_{ss}}{\pi}. \tag{9.61}$$

The Lambertian model does not account for the fact that light reflected at the surface is not available for subsurface scattering. To improve this model, there should be an energy trade-off between the surface (specular) and subsurface (diffuse) reflectance terms. The Fresnel effect implies that this surface-subsurface energy trade-off changes with incident light angle θ_i. With increasingly glancing incidence angles, the diffuse reflectance decreases as the specular reflectance increases. A basic way to account for this balance is to multiply the diffuse term by one minus the Fresnel part of the specular term [1626]. If the specular term is that of a flat mirror, the resulting diffuse term is

$$f_{diff}(\mathbf{l}, \mathbf{v}) = (1 - F(\mathbf{n}, \mathbf{l}))\frac{\rho_{ss}}{\pi}. \tag{9.62}$$

If the specular term is a microfacet BRDF term, then the resulting diffuse term is

$$f_{diff}(\mathbf{l}, \mathbf{v}) = (1 - F(\mathbf{h}, \mathbf{l}))\frac{\rho_{ss}}{\pi}. \tag{9.63}$$

Equations 9.62 and 9.63 result in a uniform distribution of outgoing light, because the BRDF value does not depend on the outgoing direction $\mathbf{v}$. This behavior makes some sense, since light will typically undergo multiple scattering events before it is re-emitted, so its outgoing direction will be randomized. However, there are two reasons to suspect that the outgoing light is not distributed perfectly uniformly. First, since the diffuse BRDF term in Equation 9.62 varies by incoming direction, Helmholtz reciprocity implies that it must change by outgoing direction as well. Second, the light must undergo refraction on the way out, which will impose some directional preference on the outgoing light.

Shirley et al. proposed a coupled diffuse term for flat surfaces that addresses the Fresnel effect and the surface-subsurface reflectance trade-off, while supporting both energy conservation and Helmholtz reciprocity [1627]. The derivation assumes that the Schlick approximation [1568] (Equation 9.16) is used for Fresnel reflectance:

$$f_{diff}(\mathbf{l}, \mathbf{v}) = \frac{21}{20\pi}(1 - F_0)\rho_{ss}\left(1 - \left(1 - (\mathbf{n} \cdot \mathbf{l})^+\right)^5\right)\left(1 - \left(1 - (\mathbf{n} \cdot \mathbf{v})^+\right)^5\right). \tag{9.64}$$

Equation 9.64 applies only to surfaces where the specular reflectance is that of a perfect Fresnel mirror. A generalized version that can be used to compute a reciprocal, energy-conserving diffuse term to couple with any specular term was proposed by

Ashikhmin and Shirley [77] and further refined by Kelemen and Szirmay-Kalos [878]:

$$f_{\text{diff}}(\mathbf{l},\mathbf{v}) = \rho_{\text{ss}}\frac{\big(1 - R_{\text{spec}}(\mathbf{l})\big)\big(1 - R_{\text{spec}}(\mathbf{v})\big)}{\pi\big(1 - \overline{R_{\text{spec}}}\big)}. \tag{9.65}$$

Here, R_{spec} is the directional albedo (Section 9.3) of the specular term, and $\overline{R_{\text{spec}}}$ is its cosine-weighted average over the hemisphere. The value R_{spec} can be precomputed using Equation 9.8 or 9.9 and stored in a lookup table. The average $\overline{R_{\text{spec}}}$ is computed the same way as a similar average we encountered earlier: $\overline{R_{\text{sF1}}}$ (Equation 9.57).

The form in Equation 9.65 has some clear similarities to Equation 9.56, which is not surprising, since the Imageworks multiple-bounce specular term is derived from the Kelemen-Szirmay-Kalos coupled diffuse term. However, there is one important difference. Here, instead of R_{sF1} we use R_{spec}, the directional albedo of the full specular BRDF term including Fresnel, and with the multiple-bounce specular term f_{ms} as well, if one is used. This difference increases the dimensionality of the lookup table for R_{spec} since it depends not only on the roughness α and elevation angle θ, but on the Fresnel reflectance as well.

In Imageworks' implementation of the Kelemen-Szirmay-Kalos coupled diffuse term, they use a three-dimensional lookup table, with the index of refraction as the third axis [947]. They found that the inclusion of the multiple-bounce term in the integral made R_{spec} smoother than R_{sF1}, so a $16\times16\times16$ table was sufficient. Figure 9.41 shows the result.

If the BRDF uses the Schlick Fresnel approximation and does not include a multiple-bounce specular term, then the value of F_0 can be factored out of the integral. Doing so allows us to use a two-dimensional table for R_{spec}, storing two quantities in each entry, instead of a three-dimensional table, as discussed by Karis [861]. Alternatively, Lazarov [999] presents an analytic function that is fitted to R_{spec}, similarly factoring F_0 out of the integral to simplify the fitted function.

Both Karis and Lazarov use the specular directional albedo R_{spec} for a different purpose, related to image-based lighting. More details on that technique can be found in Section 10.5.2. If both techniques are implemented in the same application, then the same table lookups can be used for both, increasing efficiency.

These models were developed by considering the implications of energy conservation between the surface (specular) and subsurface (diffuse) terms. Other models have been developed from physical principles. Many of these models rely on the work of Subrahmanyan Chandrasekhar (1910–1995), who developed a BRDF model for a semi-infinite, isotropically scattering volume. As demonstrated by Kulla and Conty [947], if the mean free path is sufficiently short, this BRDF model is a perfect match for a scattering volume of arbitrary shape. The Chandrasekhar BRDF can be found in his book [253], though a more accessible form using familiar rendering notation can be found in Equations 30 and 31 of a paper by Dupuy et al. [397].

Since it does not include refraction, the Chandrasekhar BRDF can be used to model only *index-matched surfaces*. These are surfaces where the index of refraction

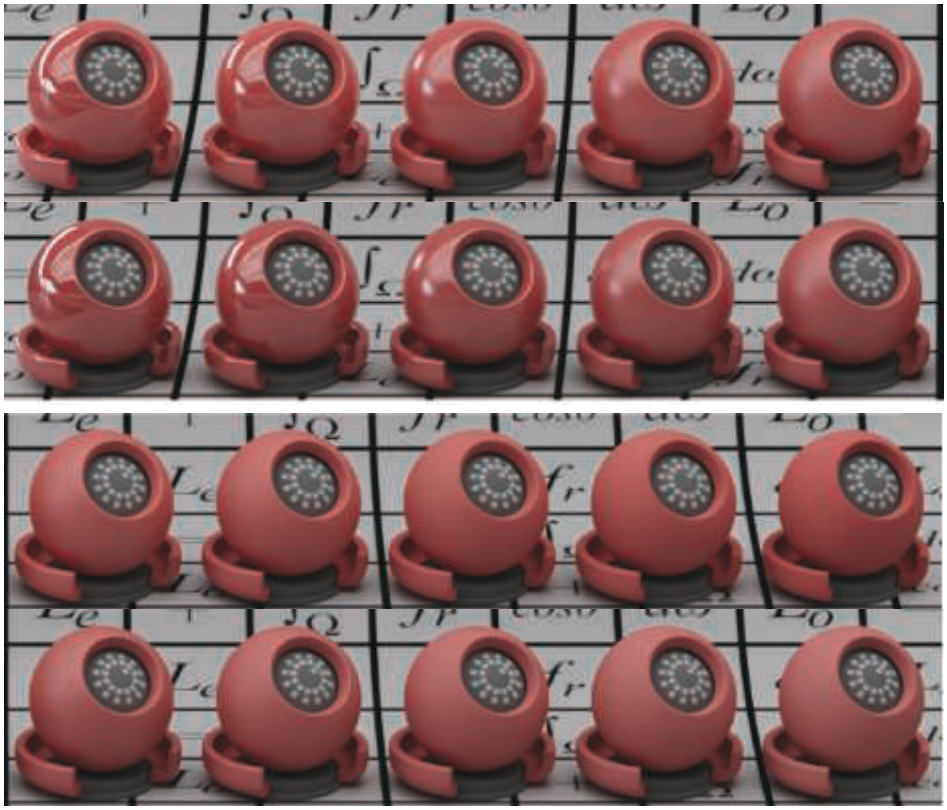

Figure 9.41. The first and third rows show a specular term added to a Lambertian term. The second and fourth rows show the same specular term used with a Kelemen-Szirmay-Kalos coupled diffuse term. The top two rows have lower roughness values than the bottom two. Within each row, roughness increases from left to right. *(Figure courtesy of Christopher Kulla [947].)*

is the same on both sides, as in Figure 9.11 on page 304. To model non-index-matched surfaces, the BRDF must be modified to account for refraction where the light enters and exits the surface. This modification is the focus of work by Hanrahan and Krueger [662] and Wolff [1898].

9.9.4 Rough-Surface Subsurface Models

As part of the Disney principled shading model, Burley [214] included a diffuse BRDF term designed to include roughness effects and match measured materials:

$$f_{\text{diff}}(\mathbf{l},\mathbf{v}) = \chi^+(\mathbf{n}\cdot\mathbf{l})\chi^+(\mathbf{n}\cdot\mathbf{v})\frac{\rho_{\text{ss}}}{\pi}\big((1-k_{\text{ss}})f_{\text{d}} + 1.25\,k_{\text{ss}}f_{\text{ss}}\big), \tag{9.66}$$

where

$$\begin{aligned} f_{\mathrm{d}} &= \Big(1 + (F_{\mathrm{D90}} - 1)(1 - \mathbf{n}\cdot\mathbf{l})^5\Big)\Big(1 + (F_{\mathrm{D90}} - 1)(1 - \mathbf{n}\cdot\mathbf{v})^5\Big), \\ F_{\mathrm{D90}} &= 0.5 + 2\sqrt{\alpha}\,(\mathbf{h}\cdot\mathbf{l})^2, \\ f_{\mathrm{ss}} &= \left(\frac{1}{(\mathbf{n}\cdot\mathbf{l})(\mathbf{n}\cdot\mathbf{v})} - 0.5\right) F_{\mathrm{SS}} + 0.5, \\ F_{\mathrm{SS}} &= \Big(1 + (F_{\mathrm{SS90}} - 1)(1 - \mathbf{n}\cdot\mathbf{l})^5\Big)\Big(1 + (F_{\mathrm{SS90}} - 1)(1 - \mathbf{n}\cdot\mathbf{v})^5\Big), \\ F_{\mathrm{SS90}} &= \sqrt{\alpha}\,(\mathbf{h}\cdot\mathbf{l})^2, \end{aligned} \tag{9.67}$$

and α is the specular roughness. In the case of anisotropy, an intermediate value between α_x and α_y is used. This equation is often referred to as the *Disney diffuse* model.

The subsurface term f_{ss} is inspired by the Hanrahan-Krueger BRDF [662] and intended as an inexpensive replacement for global subsurface scattering on distant objects. The diffuse model blends between f_{ss} and the f_{d} rough diffuse term based on the user-controlled parameter k_{ss}.

The Disney diffuse model has been used for films [214], as well as games [960] (though without the subsurface term). The full Disney diffuse BRDF also includes a sheen term, which is intended primarily for modeling fabrics, but also helps compensate for the energy lost due to the lack of a multiple-bounce specular term. The Disney sheen term will be discussed in Section 9.10. Several years later, Burley presented [215] an updated model designed to integrate with global subsurface scattering rendering techniques.

Since the Disney diffuse model uses the same roughness as the specular BRDF term, it may have difficulties modeling certain materials. See Figure 9.40. However, it would be a trivial modification to use a separate diffuse roughness value.

Most other rough-surface diffuse BRDFs have been developed using microfacet theory, with various different choices for the NDF D, micro-BRDF f_μ, and masking-shadowing function G_2. The most well-known of these models was proposed by Oren and Nayar [1337]. The Oren-Nayar BRDF uses a Lambertian micro-BRDF, a spherical Gaussian NDF, and the Torrance-Sparrow "V-cavity" masking-shadowing function. The full form of the BRDF models one secondary bounce. Oren and Nayar also included a simplified "qualitative" model in their paper. Several improvements to the Oren-Nayar model have been proposed over the years, including optimizations [573], tweaks to make the "qualitative" model more closely resemble the full model without increasing its cost [504], and changing the micro-BRDF to a more accurate smooth-surface diffuse model [574, 1899].

The Oren-Nayar model assumes a microsurface with quite different normal distribution and masking-shadowing functions than those used in current specular models. Two diffuse microfacet models were derived using the isotropic GGX NDF and height-

correlated Smith masking-shadowing function. The first model, by Gotanda [574], is the result of numerically integrating the general microfacet equation (Equation 9.26), using as the micro-BRDF the specular coupled diffuse term in Equation 9.64. An analytic function was then fitted to the numerically integrated data. Gotanda's BRDF does not account for interreflections between facets, and the fitted function is relatively complex.

Using the same NDF, masking-shadowing function, and micro-BRDF as Gotanda, Hammon [657] simulates the BRDF numerically, including interreflections. He shows that interreflections are important for this microfacet configuration, representing as much as half of the total reflectance for rougher surfaces. However, the second bounce contains almost all of the missing energy, so Hammon uses data from a two-bounce simulation. Also, likely because the addition of interreflections smoothed out the data, Hammon was able to fit a fairly simple function to the simulation results:

$$f_{\text{diff}}(\mathbf{l},\mathbf{v}) = \chi^+(\mathbf{n}\cdot\mathbf{l})\chi^+(\mathbf{n}\cdot\mathbf{v})\frac{\rho_{\text{ss}}}{\pi}\big((1-\alpha_g)f_{\text{smooth}} + \alpha_g f_{\text{rough}} + \rho_{\text{ss}} f_{\text{multi}}\big), \tag{9.68}$$

where

$$\begin{aligned} f_{\text{smooth}} &= \frac{21}{20}(1-F_0)\left(1-(1-\mathbf{n}\cdot\mathbf{l})^5\right)\left(1-(1-\mathbf{n}\cdot\mathbf{v})^5\right), \\ f_{\text{rough}} &= k_{\text{facing}}(0.9-0.4\,k_{\text{facing}})\left(\frac{0.5+\mathbf{n}\cdot\mathbf{h}}{\mathbf{n}\cdot\mathbf{h}}\right), \\ k_{\text{facing}} &= 0.5+0.5(\mathbf{l}\cdot\mathbf{v}), \\ f_{\text{multi}} &= 0.3641\alpha_g, \end{aligned} \tag{9.69}$$

and α_g is the GGX specular roughness. For clarity, the terms here have been factored slightly differently than in Hammon's presentation. Note that f_{smooth} is the coupled diffuse BRDF from Equation 9.64 without the ρ_{ss}/π factor, since this is multiplied in Equation 9.68. Hammon discusses "hybrid" BRDFs that substitute other smooth-surface diffuse BRDFs for f_{smooth}, to increase performance or improve compatibility with assets authored under older models.

Overall, Hammon's diffuse BRDF is inexpensive and is based on sound theoretical principles, although he did not show comparisons with measured data. One caveat is that the assumption that surface irregularities are larger than scattering distances is fundamental to the derivation of the BRDF, which may limit the types of materials it can accurately model. See Figure 9.40.

The simple Lambertian term shown in Equation 9.61 is still implemented by many real-time rendering applications. Besides the Lambertian term's low computational cost, it is easier to use with indirect and baked lighting than other diffuse models, and the visual differences between it and more sophisticated models are often subtle [251, 861]. Nevertheless, the continuing quest for photorealism is driving an increase in the use of more accurate models.

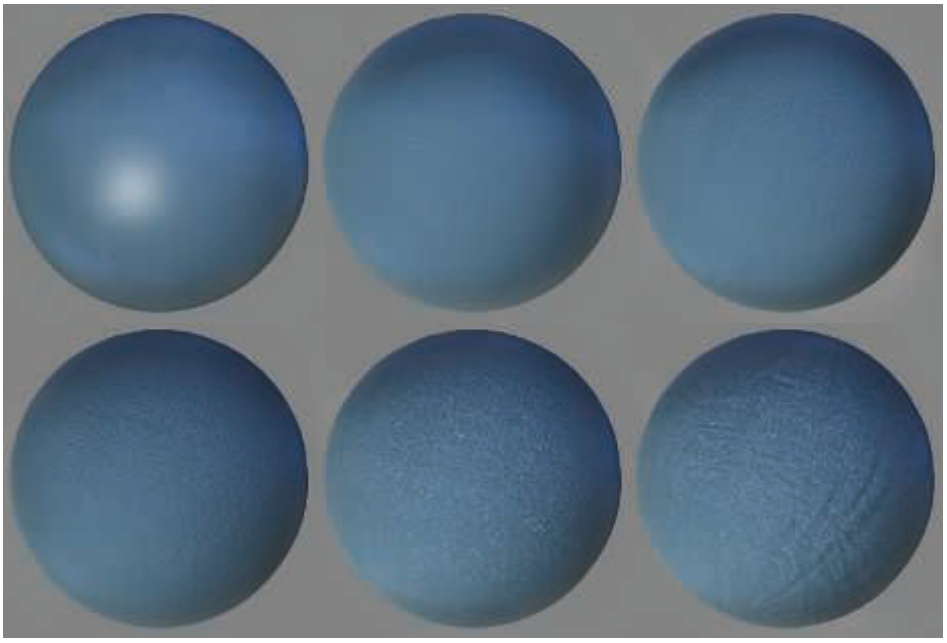

Figure 9.42. A material using the cloth system built for the game *Uncharted 4*. The upper left sphere has a standard BRDF with a GGX microfacet specular and Lambertian diffuse. The upper middle sphere uses the fabric BRDF. Each of the other spheres adds a different type of per-pixel variation, going from left to right and top to bottom: fabric weave details, fabric aging, imperfection details, and small wrinkles. *(UNCHARTED 4 A Thief's End ©/™ 2016 SIE. Created & developed by Naughty Dog LLC.)*

9.10 BRDF Models for Cloth

Cloth tends to have microgeometry that is different from other types of materials. Depending on the fabric type, it may have highly repetitive woven microstructures, cylinders (threads) protruding vertically from the surface, or both. As a result, cloth surfaces have characteristic appearances that typically require specialized shading models, such as anisotropic specular highlights, asperity scattering [919] (bright edge effects caused by light scattering through protruding, translucent fibers), and even color shifts with view direction (caused by threads of different colors running through the fabric).

Aside from the BRDF, most fabrics have high-frequency spatial variation that is also key to creating a convincing cloth appearance [825]. See Figure 9.42.

Cloth BRDF models fall into three main categories: empirical models created from observation, models based on microfacet theory, and micro-cylinder models. We will go over some notable examples from each category.

9.10.1 Empirical Cloth Models

In the game *Uncharted 2* [631], cloth surfaces use the following diffuse BRDF term:

$$f_{\text{diff}}(\mathbf{l},\mathbf{v}) = \frac{\rho_{\text{ss}}}{\pi}\Big(k_{\text{rim}}\big((\mathbf{v}\cdot\mathbf{n})^{+}\big)^{\alpha_{\text{rim}}} + k_{\text{inner}}\big(1-(\mathbf{v}\cdot\mathbf{n})^{+}\big)^{\alpha_{\text{inner}}} + k_{\text{diff}}\Big), \tag{9.70}$$

where k_{rim}, k_{inner}, and k_{diff} are user-controlled scaling factors for a rim lighting term, a term to brighten forward-facing (inner) surfaces, and a Lambertian term, respectively. Also, α_{rim} and α_{inner} control the falloff of the rim and inner terms. This behavior is non-physical, since there are several view-dependent effects but none that depend on the light direction.

In contrast, the cloth in *Uncharted 4* [825] uses either a microfacet or micro-cylinder model, depending on cloth type (as detailed in the following two sections) for the specular term and a "wrap lighting" empirical subsurface scattering approximation for the diffuse term:

$$f_{\text{diff}}(\mathbf{l},\mathbf{v})(\mathbf{n}\cdot\mathbf{l})^{+} \Rightarrow \frac{\rho_{\text{ss}}}{\pi}\Big(\mathbf{c}_{\text{scatter}} + (\mathbf{n}\cdot\mathbf{l})^{+}\Big)^{\overline{+}}\frac{(\mathbf{n}\cdot\mathbf{l}+w)^{\overline{+}}}{1+w}. \tag{9.71}$$

Here we use the $(x)^{\overline{+}}$ notation introduced in Section 1.2, which indicates a clamp between 0 and 1. The odd notation $f_{\text{diff}}(\mathbf{l},\mathbf{v})(\mathbf{n}\cdot\mathbf{l})^{+} \Rightarrow \ldots$ indicates that this model affects the lighting as well as the BRDF. The term on the right side of the arrow replaces the term on the left side. The user-specified parameter $\mathbf{c}_{\text{scatter}}$ is a scattering color, and the value w, with range $[0,1]$, controls the wrap lighting width.

For modeling cloth, Disney use their diffuse BRDF term [214] (Section 9.9.4) with a sheen term added to model asperity scattering:

$$f_{\text{sheen}}(\mathbf{l},\mathbf{v}) = k_{\text{sheen}}\mathbf{c}_{\text{sheen}}\big(1-(\mathbf{h}\cdot\mathbf{l})^{+}\big)^{5}, \tag{9.72}$$

where k_{sheen} is a user parameter that modulates the strength of the sheen term. The sheen color $\mathbf{c}_{\text{sheen}}$ is a blend (controlled by another user parameter) between white and the luminance-normalized value of ρ_{ss}. In other words, ρ_{ss} is divided by its luminance to isolate its hue and saturation.

9.10.2 Microfacet Cloth Models

Ashikhmin et al. [78] proposed using an inverted Gaussian NDF to model velvet. This NDF was slightly modified in subsequent work [81], which also proposed a variant form of the microfacet BRDF for modeling materials in general, with no masking-shadowing term and a modified denominator.

The cloth BRDF used in the game *The Order: 1886* [1266] combines the modified microfacet BRDF and a generalized form of the velvet NDF from Ashikhmin and Premože's later report [81] with the diffuse term from Equation 9.63. The generalized

Figure 9.43. The Imageworks sheen specular term added to a red diffuse term. From left to right, the sheen roughness values are $\alpha = 0.15$, 0.25, 0.40, 0.65, and 1.0. *(Figure courtesy of Alex Conty [442].)*

velvet NDF is

$$D(\mathbf{m}) = \frac{\chi^+(\mathbf{n}\cdot\mathbf{m})}{\pi(1+k_{\text{amp}}\alpha^2)}\left(1 + \frac{k_{\text{amp}}\exp\left(\frac{(\mathbf{n}\cdot\mathbf{m})^2}{\alpha^2\left((\mathbf{n}\cdot\mathbf{m})^2-1\right)}\right)}{\left(1-(\mathbf{n}\cdot\mathbf{m})^2\right)^2}\right), \tag{9.73}$$

where α controls the width of the inverted Gaussian and k_{amp} controls its amplitude. The full cloth BRDF is

$$f(\mathbf{l},\mathbf{v}) = \big(1-F(\mathbf{h},\mathbf{l})\big)\frac{\rho_{\text{ss}}}{\pi} + \frac{F(\mathbf{h},\mathbf{l})D(\mathbf{h})}{4\big(\mathbf{n}\cdot\mathbf{l}+\mathbf{n}\cdot\mathbf{v}-(\mathbf{n}\cdot\mathbf{l})(\mathbf{n}\cdot\mathbf{v})\big)}. \tag{9.74}$$

A variation of this BRDF was used in the game *Uncharted 4* [825] for rough fabrics such as wool and cotton.

Imageworks [947] use a different inverted NDF for a sheen term that can be added to any BRDF:

$$D(\mathbf{m}) = \frac{\chi^+(\mathbf{n}\cdot\mathbf{m})(2+\frac{1}{\alpha})\big(1-(\mathbf{n}\cdot\mathbf{m})^2\big)^{\frac{1}{2\alpha}}}{2\pi}. \tag{9.75}$$

Although there is no closed-form solution to the Smith masking-shadowing function for this NDF, Imageworks were able to approximate the numerical solution with an analytical function. Details on the masking-shadowing function and on energy conservation between the sheen term and the rest of the BRDF are discussed by Estevez and Kulla [442]. See Figure 9.43 for some examples rendered using the Imageworks sheen term.

Each of the cloth models we have seen so far are limited to specific types of fabric. The models discussed in the next section attempt to model cloth in a more general way.

9.10.3 Micro-Cylinder Cloth Models

The micro-cylinder models used for cloth are quite similar to those used for hair, so the discussion of hair models in Section 14.7.2 can provide additional context. The idea

behind these models is that the surface is assumed to be covered with one-dimensional lines. Kajiya and Kay developed a simple BRDF model for this case [847], which was given a solid theoretical foundation by Banks [98]. It is alternately known as the *Kajiya-Kay BRDF* or the *Banks BRDF*. The concept is based on the observation that a surface composed of one-dimensional lines has an infinite number of normals at any given location, defined by the *normal plane* perpendicular to the tangent vector $\mathbf{t}$ at that location. Although many newer micro-cylinder models have been developed from this framework, the original Kajiya-Kay model still sees some use, due to its simplicity. For example, in the game *Uncharted 4* [825], the Kajiya-Kay BRDF was used for the specular term of shiny fabrics such as silk and velvet.

Dreamworks [348, 1937] use a relatively simple and artist-controllable micro-cylinder model for fabric. Textures can be used to vary the roughness, color, and thread direction, which can point out of the surface plane for modeling velvet and similar fabrics. Different parameters can be set for the warp and weft threads to model complex color-changing fabrics, such as shot silk. The model is normalized to be energy-conserving.

Sadeghi et al. [1526] proposed a micro-cylinder model based on measurements from fabric samples as well as individual threads. The model also accounts for inter-thread masking and shadowing between threads.

In some cases actual hair BSDF models (Section 14.7) are used for cloth. RenderMan's *PxrSurface* material [732] has a "fuzz" lobe that uses the R term from the hair model by Marschner et al. [1128] (Section 14.7). One of the models implemented in a real-time cloth rendering system by Wu and Yuksel [1924, 1926] is derived from a hair model used by Disney for animated films [1525].

9.11 Wave Optics BRDF Models

The models we have discussed in the last few sections rely on geometrical optics, which treats light as propagating in rays rather than waves. As discussed on page 303, geometrical optics is based on the assumption that any surface irregularities are either smaller than a wavelength or larger than about 100 wavelengths.

Real-world surfaces are not so obliging. They tend to have irregularities at all scales, including the 1–100 wavelength range. We refer to irregularities with such sizes as *nanogeometry* to distinguish them from the microgeometry irregularities discussed in earlier sections, which are too small to be individually rendered but larger than 100 light wavelengths. The effects of nanogeometry on reflectance cannot be modeled by geometrical optics. These effects depend on the wave nature of light and *wave optics* (also called *physical optics*) is required to model them.

Surface layers, or films, with thicknesses close to a light wavelength also produce optical phenomena related to the wave nature of light.

In this section we touch upon wave optics phenomena such as diffraction and thin-film interference, discussing their (sometimes surprising) importance in realistically rendering what otherwise can seem to be relatively mundane materials.

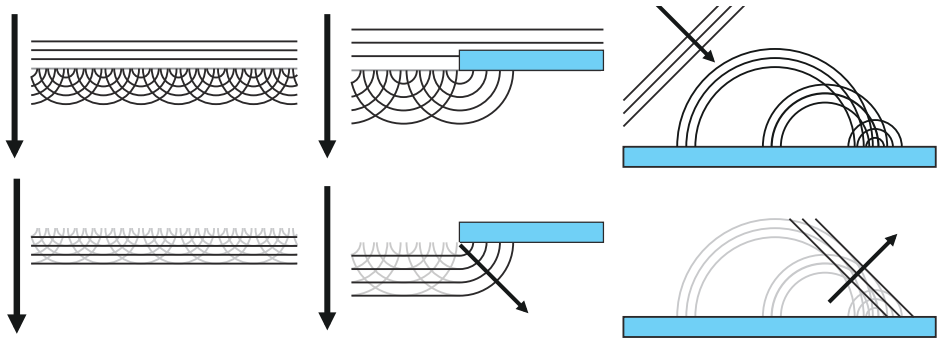

Figure 9.44. To the left, we see a planar wavefront propagating in empty space. If each point on the wavefront is treated as the source of a new spherical wave, the new waves interfere destructively in all directions except forward, resulting in a planar wavefront again. In the center, the waves encounter an obstacle. The spherical waves at the edge of the obstacle have no waves to their right that destructively interfere with them, so some waves diffract or "leak" around the edge. To the right, a planar wavefront is reflected from a flat surface. The planar wavefront encounters surface points on the left earlier than points on the right, so the spherical waves emitting from surface points on the left have had more time to propagate and are therefore larger. The different sizes of spherical wavefronts interfere constructively along the edge of the reflected planar wavefront, and destructively in other directions.

9.11.1 Diffraction Models

Nanogeometry causes a phenomenon called *diffraction*. To explain it we make use of the *Huygens-Fresnel principle*, which states that every point on a wavefront (the set of points that have the same wave phase) can be treated as the source of a new spherical wave. See Figure 9.44. When waves encounter an obstacle, the Huygens-Fresnel principle shows that they will bend slightly around corners, which is an example of diffraction. This phenomenon cannot be predicted by geometrical optics. In the case of light incident on a planar surface, geometrical optics does correctly predict that light will be reflected in a single direction. That said, the Fresnel-Huygens principle provides additional insight. It shows that the spherical waves on the surface line up just right to create the reflected wavefront, with waves in all other directions being eliminated through destructive interference. This insight becomes important when we look at a surface with nanometer irregularities. Due to the different heights of the surface points, the spherical waves on the surface no longer line up so neatly. See Figure 9.45.

As the figure shows, light is scattered in different directions. Some portion of it is specularly reflected, i.e., adds up to a planar wavefront in the reflection direction. The remaining light is diffracted out in a directional pattern that depends on certain properties of the nanogeometry. The division between specularly reflected and diffracted light depends on the height of the nanogeometry bumps, or, more precisely, on the variance of the height distribution. The angular spread of diffracted light around the

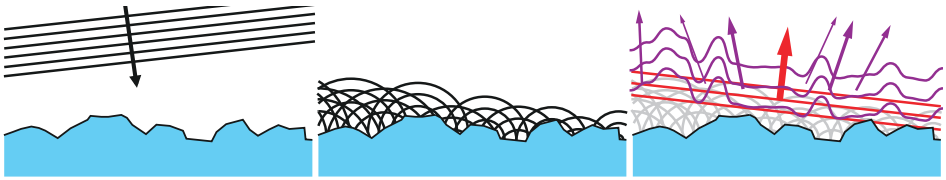

Figure 9.45. On the left we see planar wavefronts incident to a surface with rough nanogeometry. In the center we see the spherical waves formed on the surface according to the Fresnel-Huygens principle. On the right we see that after constructive and destructive interference has occurred, some of the resulting waves (in red) form a planar reflected wave. The remainder (in purple) are diffracted, with different amounts of light propagating in each direction, depending on wavelength.

specular reflection direction depends on the width of the nanogeometry bumps relative to the light wavelength. Somewhat counter-intuitively, wider irregularities cause a smaller spread. If the irregularities are larger than 100 light wavelengths, the angle between the diffracted light and the specularly reflected light is so small as to be negligible. Irregularities of decreasing size cause a wider spread of diffracted light, until the irregularities become smaller than a light wavelength, at which point no diffraction occurs.

Diffraction is most clearly visible in surfaces with periodic nanogeometry, since the repeating patterns reinforce the diffracted light via constructive interference, causing a colorful iridescence. This phenomena can be observed in CD and DVD optical disks and certain insects. While diffraction also occurs in non-periodic surfaces, the computer graphics community has assumed for many years that the effect is slight. For this reason, with a handful of exceptions [89, 366, 686, 1688], the computer graphics literature has mostly ignored diffraction for many years.

However, recent analysis of measured materials by Holzschuch and Pacanowski [762] has shown that significant diffraction effects are present in many materials, and may explain the continuing difficulty of fitting these materials with current models. Follow-up work by the same authors [763] introduced a model combining microfacet and diffraction theory, via the use of the general microfacet BRDF (Equation 9.26) with a micro-BRDF that accounts for diffraction. In parallel, Toisoul and Ghosh [1772, 1773] presented methods for capturing the iridescent diffraction effects resulting from periodic nanogeometry, and for rendering them in real time with point light sources as well as image-based lighting.

9.11.2 Models for Thin-Film Interference

Thin-film interference is a wave optics phenomenon that occurs when light paths reflecting from the top and bottom of a thin dielectric layer interfere with each other. See Figure 9.46.

The different wavelengths of light either interfere constructively or destructively, depending on the relationship between the wavelength and the path length difference.

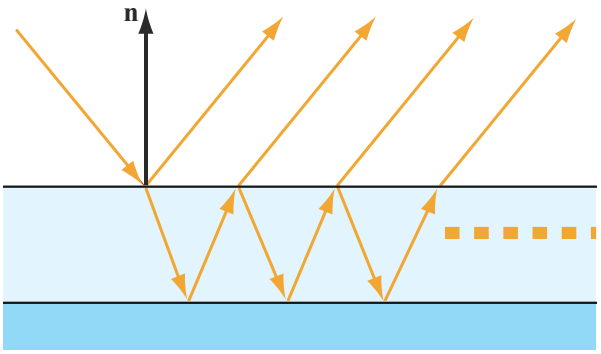

Figure 9.46. Light incident to a thin film on top of a reflective substrate. Besides the primary reflection, there are multiple paths of light refracting, reflecting from the substrate, and either reflecting from the inside of the top thin film surface or refracting through it. These paths are all copies of the same wave, but with short phase delays caused by difference in path length, so they interfere coherently with each other.

Since the path length difference changes with angle, the end result is an iridescent color shift as different wavelengths transition between constructive and destructive interference.

The reason that the film needs to be thin for this effect to occur is related to the concept of *coherence length*. This length is the maximum distance by which a copy of a light wave can be displaced and still interfere coherently with the original wave. This length is inversely proportional to the *bandwidth* of the light, which is the range of wavelengths over which its spectral power distribution (SPD) extends. Laser light, with its extremely narrow bandwidth, has an extremely long coherence length. It can be miles, depending on the type of laser. This relationship makes sense, since a simple sine wave displaced by many wavelengths will still interfere coherently with the original wave. If the laser was truly monochromatic, it would have an infinite coherence length, but in practice lasers have a nonzero bandwidth. Conversely, light with an extremely broad bandwidth will have a chaotic waveform. It makes sense that a copy of such a waveform needs to be displaced only a short distance before it stops interfering coherently with the original.

In theory, ideal white light, which is a mixture of all wavelengths, would have a coherence length of zero. However, for the purposes of visible-light optics, the bandwidth of the human visual system (which senses light only in the 400–700 nm range) determines the coherence length, which is about 1 micrometer. So, in most cases the answer to the question "how thick can a film get before it no longer causes visible interference?" is "about 1 micrometer."

Similarly to diffraction, for many years thin-film interference was thought of as a special case effect that occurs only in surfaces such as soap bubbles and oil stains. However, Akin [27] points out that thin-film interference does lend a subtle coloration

Figure 9.47. A leather material rendered without (on the left) and with (on the right) thin-film interference. The specular coloration caused by thin-film interference increases the realism of the image. *(Image by Atilla Akin, Next Limit Technologies [27].)*

to many everyday surfaces, and shows how modeling this effect can increase realism. See Figure 9.47. His article caused the level of interest in physically based thin-film interference to increase considerably, with various shading models including RenderMan's *PxrSurface* [732] and the Imageworks shading model [947] incorporating support for this effect.

Thin-film interference techniques suitable for real-time rendering have existed for some time. Smits and Meyer [1667] proposed an efficient method to account for thin-film interference between the first- and second-order light paths. They observe that the resulting color is primarily a function of the path length difference, which can be efficiently computed from the film thickness, viewing angle, and index of refraction. Their implementation requires a one-dimensional lookup table with RGB colors. The contents of the table can be computed using dense spectral sampling and converted to RGB colors as a preprocess, which makes the technique quite fast. In the game *Call of Duty: Infinite Warfare*, a different fast thin-film approximation is used as part of a layered material system [386]. These techniques do not model multiple bounces of light in the thin film, as well as other physical phenomena. A more accurate and computationally expensive technique, yet still targeted at real-time implementation, is presented by Belcour and Barla [129].

9.12 Layered Materials

In real life, materials are often layered on top of one another. A surface may be covered with dust, water, ice, or snow; it may be painted with lacquer or some other coating

for decorative or protective reasons; or it may have multiple layers as part of its basic construction, such as many biological materials.

One of the simplest and most visually significant cases of layering is a *clear coat*, which is a smooth transparent layer over a substrate of some different material. An example is a smooth coat of varnish over a rough wood surface. The Disney principled shading model [214] includes a clear-coat term, as do the Unreal Engine [1802], RenderMan's *PxrSurface* material [732], and the shading models used by Dreamworks Animation [1937] and Imageworks [947], among others.

The most notable visual result of a clear-coat layer is the double reflection resulting from light reflecting off both the clear-coat and the underlying substrate. This second reflection is most notable when the substrate is a metal, since then the difference between the indices of refraction of the dielectric clear coat and the substrate is largest. When the substrate is a dielectric, its index of refraction is close to that of the clear coat, causing the second reflection to be relatively weak. This effect is similar to underwater materials, shown in Table 9.4 on page 325.

The clear-coat layer can also be tinted. From a physical point of view, this tinting is the result of absorption. The amount of absorbed light depends on the length of the path that light travels through the clear-coat layer, according to the Beer-Lambert law (Section 14.1.2). This path length depends on the angles of the view and light, as well as the index of refraction of the material. The simpler clear-coat implementations, such as those in the Disney principled model and the Unreal Engine, do not model this view-dependence. Others do, such as the implementations in *PxrSurface* and the Imageworks and Dreamworks shading models. The Imageworks model further allows for concatenating an arbitrary number of layers of different types.

In the general case, different layers could have different surface normals. Some examples include rivulets of water running over flat pavement, a smooth sheet of ice on top of bumpy soil, or wrinkled plastic wrap covering a cardboard box. Most layered models used by the movie industry support separate normals per layer. This practice is not common in real-time applications, though the Unreal Engine's clear-coat implementation supports it as an optional feature.

Weidlich and Wilkie [1862, 1863] propose a layered microfacet model, with the assumption that the layer thickness is small compared to the size of the microfacets. Their model supports an arbitrary number of layers, and tracks reflection and refraction events from the top layer down to the bottom and back up again. It is simple enough for real-time implementation [420, 573], but does not account for multiple reflections between layers. Jakob et al. [811, 812] present a comprehensive and accurate framework for simulating layered materials, including multiple reflections. Although not suitable for real-time implementation, the system is useful for ground-truth comparisons, and the ideas used may suggest future real-time techniques.

The game *Call of Duty: Infinite Warfare* uses a layered material system [386] that is especially notable. It allows users to composite an arbitrary number of material layers. It supports refraction, scattering, and path-length-based absorption between

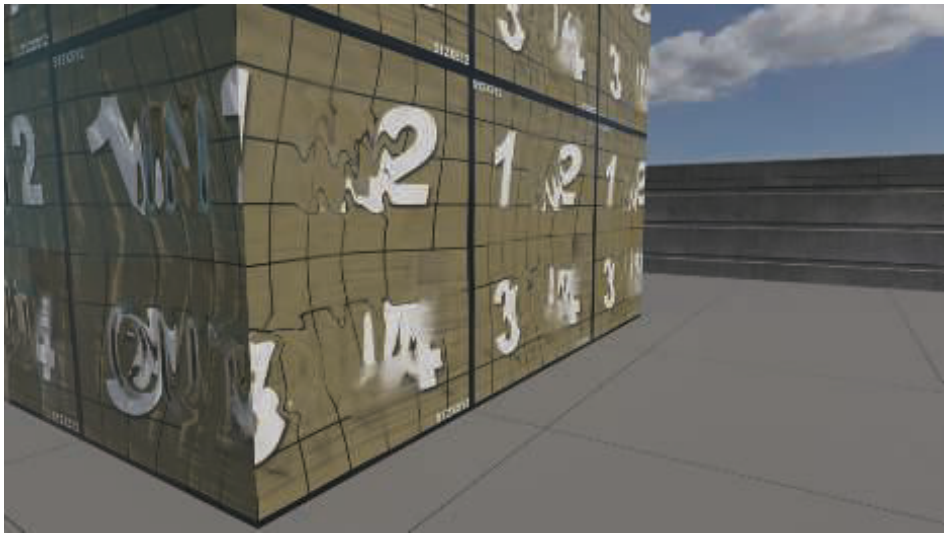

Figure 9.48. Test surface showing various features of the *Call of Duty: Infinite Warfare* multilayer material system. The material simulates a geometrically complex surface with distortion and scattering, although each side is constructed from only two triangles. *(Image courtesy of Activision Publishing, Inc. 2018.)*

layers, as well as different surface normals per layer. Combined with a highly efficient implementation, this system enables real-time materials of unprecedented complexity, particularly impressive for a game running at 60 Hz. See Figure 9.48.

9.13 Blending and Filtering Materials

Material blending is the process of combining the properties, i.e., the BRDF parameters, of multiple materials. For example, to model a sheet of metal with rust spots, we could paint a mask texture to control the rust spot locations and use it to blend between the material properties (specular color F_0, diffuse color ρ_{ss}, and roughness α) of rust and metal. Each of the materials being blended can also be spatially varying, with parameters stored in textures. Blending can be done as a preprocess to create a new texture, often referred to as "baking," or on the fly in the shader. Although the surface normal $\mathbf{n}$ is technically not a BRDF parameter, its spatial variation is important for appearance, so material blending typically includes normal map blending as well.

Material blending is critical to many real-time rendering applications. For example, the game *The Order: 1886* has a complex material blending system [1266, 1267, 1410]

that allows users to author arbitrarily deep stacks of materials drawn from an extensive library and controlled by various spatial masks. Most of the material blending is done as an offline preprocess, but certain compositing operations can be deferred to runtime as needed. This runtime processing is typically used for environments, to add unique variations to tiled textures. The popular material authoring tools *Substance Painter* and *Substance Designer* use a similar approach for material compositing, as does the *Mari* texture painting tool.

Blending texture elements on the fly provides a diverse set of effects while conserving memory. Games employ material blending for various purposes, such as:

- Displaying dynamic damage on buildings, vehicles, and living (or undead) creatures [201, 603, 1488, 1778, 1822].
- Enabling user customization of in-game equipment and clothing [604, 1748].
- Increasing visual variety in characters [603, 1488] and environments [39, 656, 1038]. See Figure 20.5 on page 891 for an example.

Sometimes one material is blended on top of another with less than 100% opacity, but even fully opaque blends will have pixels (or texels, if baking into textures) on mask boundaries where a partial blend needs to be performed. In either case, the strictly correct approach would be to evaluate the shading model for each material and blend the results. However, blending the BRDF parameters and then evaluating the shading once is much faster. In the case of material properties that have a linear or nearly linear relationship to the final shaded color, such as the diffuse and specular color parameters, little or no error is introduced by such interpolation. In many cases, even for parameters with a highly nonlinear relationship to the final shaded color (such as specular roughness), the errors introduced along mask boundaries are not objectionable.

Blending normal maps requires special consideration. Often good results can be achieved by treating the process as a blend between height maps from which the normal maps are derived [1086, 1087]. In some cases, such as when overlaying a detail normal map on top of a base surface, other forms of blending are preferable [106].

Material filtering is a topic closely related to material blending. Material properties are typically stored in textures, which are filtered via mechanisms such as GPU bilinear filtering and mipmapping. However, these mechanisms are based on the assumption that the quantity being filtered (which is an input to the shading equation) has a linear relationship to the final color (the output of the shading equation). Linearity again holds for some quantities, but not in general. Artifacts can result from using linear mipmapping methods on normal maps, or on textures containing nonlinear BRDF parameters, such as roughness. These artifacts can manifest as specular aliasing (flickering highlights), or as unexpected changes in surface gloss or brightness with a change in the surface's distance from the camera. Of these two, specular aliasing is much more noticeable; techniques for mitigating these artifacts are often referred to as *specular antialiasing* techniques. We will now discuss several of these methods.

Figure 9.49. On the left, the cylinder is rendered with the original normal map. In the center, a much lower-resolution normal map containing averaged and renormalized normals is used, as shown in the bottom left of Figure 9.50. On the right, the cylinder is rendered with textures at the same low resolution, but containing normal and gloss values fitted to the ideal NDF, as shown in the bottom right of Figure 9.50. The image on the right is a significantly better representation of the original appearance. This surface will also be less prone to aliasing when rendered at low resolution. *(Image courtesy of Patrick Conran, ILM.)*

9.13.1 Filtering Normals and Normal Distributions

The lion's share of material filtering artifacts (primarily from specular aliasing), as well as the most frequently used solutions for them, are related to the filtering of normals and normal distribution functions. Because of its importance, we will discuss this aspect in some depth.

To understand why these artifacts occur and how to solve them, recall that the NDF is a statistical description of subpixel surface structure. When the distance between the camera and surface increases, surface structures that previously covered multiple pixels may be reduced to subpixel size, moving from the realm of bump maps into the realm of the NDF. This transition is intimately tied to the mipmap chain, which encapsulates the reduction of texture details to subpixel size.

Consider how the appearance of an object, such as the cylinder at the left in Figure 9.49, is modeled for rendering. Appearance modeling always assumes a certain scale of observation. *Macroscale* (large-scale) geometry is modeled as triangles, *mesoscale* (middle-scale) geometry is modeled as textures, and *microscale* geometry, smaller than a single pixel, is modeled via the BRDF.

Given the scale shown in the image, it is appropriate to model the cylinder as a smooth mesh (macroscale) and to represent the bumps with a normal map (mesoscale). A Beckmann NDF with a fixed roughness α_b is chosen to model the microscale normal distribution. This combined representation models the cylinder appearance well at this scale. But, what happens when the scale of observation changes?

Study Figure 9.50. The black-framed figure at the top shows a small part of the surface, covered by four normal-map texels. Assume that we are rendering the surface at a scale such that each normal map texel is covered by one pixel on average. For

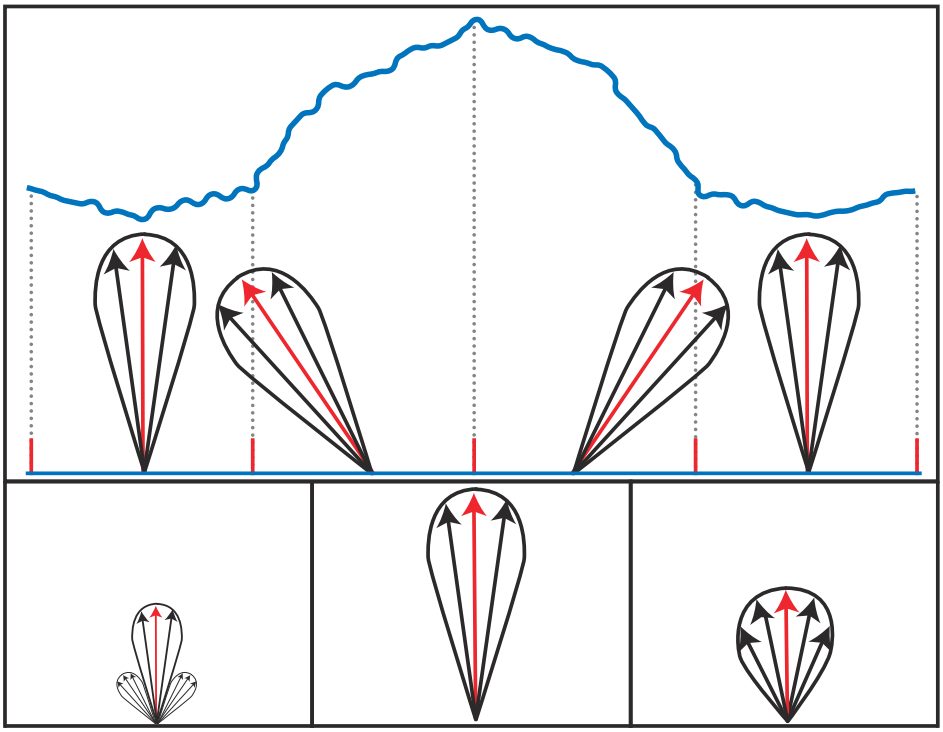

Figure 9.50. Part of the surface from Figure 9.49. The top row shows the normal distributions (the mean normal shown in red) and the implied microgeometry. The bottom row shows three ways to average the four NDFs into one, as done in mipmapping. On the left is the ground truth (averaging the normal distributions), the center shows the result of averaging the mean (normal) and variance (roughness) separately, and the right shows an NDF lobe fitted to the averaged NDF.

each texel, the normal (which is the average, or mean, of the distribution) is shown as a red arrow, surrounded by the Beckmann NDF, shown in black. The normals and NDF implicitly specify an underlying surface structure, shown in cross section. The large hump in the middle is one of the bumps from the normal map, and the small wiggles are the microscale surface structure. Each texel in the normal map, combined with the roughness, can be seen as collecting the distribution of normals across the surface area covered by the texel.

Now assume that the camera has moved further from the object, so that one pixel covers all four of the normal map texels. The ideal representation of the surface at this resolution would exactly represent the distribution of all normals collected across the larger surface area covered by each pixel. This distribution could be found by averaging the NDFs in the four texels of the top-level mipmap. The lower left figure shows this ideal normal distribution. This result, if used for rendering, would most accurately represent the appearance of the surface at this lower resolution.

The bottom center figure shows the result of separately averaging the normals, the mean of each distribution, and the roughness, which corresponds to the width of each. The result has the correct average normal (in red), but the distribution is too narrow. This error will cause the surface to appear too smooth. Worse, since the NDF is so narrow, it will tend to cause aliasing, in the form of flickering highlights.

We cannot represent the ideal normal distribution directly with the Beckmann NDF. However, if we use a roughness map, the Beckmann roughness α_b can be varied from texel to texel. Imagine that, for each ideal NDF, we find the oriented Beckmann lobe that matches it most closely, both in orientation and overall width. We store the center direction of this Beckmann lobe in the normal map, and its roughness value in the roughness map. The results are shown on the bottom right. This NDF is much closer to the ideal. The appearance of the cylinder can be represented much more faithfully with this process than with simple normal averaging, as can be seen in Figure 9.49.

For best results, filtering operations such as mipmapping should be applied to normal distributions, not normals or roughness values. Doing so implies a slightly different way to think about the relationship between the NDFs and the normals. Typically the NDF is defined in the local tangent space determined by the normal map's per-pixel normal. However, when filtering NDFs across different normals, it is more useful to think of the combination of the normal map and roughness map as defining a skewed NDF (one that does not average to a normal pointing straight up) in the tangent space of the underlying geometrical surface.

Early attempts to solve the NDF filtering problem [91, 284, 658] used numerical optimization to fit one or more NDF lobes to the averaged distribution. This approach suffers from robustness and speed issues, and is not used much today. Instead, most techniques currently in use work by computing the variance of the normal distribution. Toksvig [1774] makes a clever observation that if normals are averaged and not renormalized, the length of the averaged normal correlates inversely with the width of the normal distribution. That is, the more the original normals point in different directions, the shorter the normal averaged from them. He presents a method to modify the NDF roughness parameter based on this normal length. Evaluating the BRDF with the modified roughness approximates the spreading effect of the filtered normals.

Toksvig's original equation was intended for use with the Blinn-Phong NDF:

$$\alpha'_p = \frac{\|\overline{\mathbf{n}}\|\alpha_p}{\|\overline{\mathbf{n}}\| + \alpha_p(1 - \|\overline{\mathbf{n}}\|)}, \tag{9.76}$$

where α_p is the original roughness parameter value, α'_p is the modified value, and $\|\overline{\mathbf{n}}\|$ is the length of the averaged normal. The equation can also be used with the Beckmann NDF by applying the equivalence $\alpha_p = 2\alpha_b^{-2} - 2$ (from Walter et al. [1833]), since the shapes of the two NDFs are quite close. Using the method with GGX is less straightforward, since there is no clear equivalence between GGX and Blinn-Phong (or Beckmann). Using the α_b equivalence for α_g gives the same value at the center

of the highlight, but the highlight appearance is quite different. More troubling, the variance of the GGX distribution is undefined, which puts this variance-based family of techniques on shaky theoretical ground when used with GGX. Despite these theoretical difficulties, it is fairly common to use Equation 9.76 with the GGX distribution, typically using $\alpha_p = 2\alpha_g^{-2} - 2$. Doing so works reasonably well in practice.

Toksvig's method has the advantage of accounting for normal variance introduced by GPU texture filtering. It also works with the simplest normal mipmapping scheme, linear averaging without normalization. This feature is particularly useful for dynamically generated normal maps such as water ripples, for which mipmap generation must be done on the fly. The method is not as good for static normal maps, since it does not work well with prevailing methods of compressing normal maps. These compression methods rely on the normal being of unit length. Since Toksvig's method relies on the length of the average normal varying, normal maps used with it may have to remain uncompressed. Even then, storing the shortened normals can result in precision issues.

Olano and Baker's LEAN mapping technique [1320] is based on mapping the covariance matrix of the normal distribution. Like Toksvig's technique, it works well with GPU texture filtering and linear mipmapping. It also supports anisotropic normal distributions. Similarly to Toksvig's method, LEAN mapping works well with dynamically generated normals, but to avoid precision issues, it requires a large amount of storage when used with static normals. A similar technique was independently developed by Hery et al. [731, 732] and used in Pixar's animated films to render subpixel details such as metal flakes and small scratches. A simpler variant of LEAN mapping, CLEAN mapping [93], requires less storage at the cost of losing anisotropy support. LEADR mapping [395, 396] extends LEAN mapping to also account for the visibility effects of displacement mapping.

The majority of normal maps used in real-time applications are static, rather than dynamically generated. For such maps, the *variance mapping* family of techniques is commonly used. In these techniques, when the normal map's mipmap chain is generated, the variance that was lost through averaging is computed. Hill [739] notes that the mathematical formulations of Toksvig's technique, LEAN mapping, and CLEAN mapping could each be used to precompute variance in this way, which removes many of the disadvantages of these techniques when used in their original forms. In some cases the precomputed variance values are stored in the mipmap chain of a separate variance texture. More often, these values are used to modify the mipmap chain of an existing roughness map. For example, this method is employed in the variance-mapping technique used in the game *Call of Duty: Black Ops* [998]. The modified roughness values are computed by converting the original roughness values to variance values, adding in the variance from the normal map, and converting the result back to roughness. For the game *The Order: 1886*, Neubelt and Pettineo [1266, 1267] use a technique by Han [658] in a similar way. They convolve the normal map NDF with the NDF of their BRDF's specular term, convert the result to roughness, and store it in a roughness map.

For improved results at the cost of some extra storage, variance can be computed in the texture-space x- and y-directions and stored in an anisotropic roughness map [384, 740, 1823]. By itself this technique is limited to axis-aligned anisotropy, which is typical in man-made surfaces but less so in naturally occurring ones. At the cost of storing one more value, oriented anisotropy can be supported as well [740].

Unlike the original forms of Toksvig, LEAN, and CLEAN mapping, variance mapping techniques do not account for variance introduced by GPU texture filtering. To compensate for this, variance mapping implementations often convolve the top-level mip of the normal map with a small filter [740, 998]. When combining multiple normal maps, e.g., detail normal mapping [106], care needs to be taken to combine the variance of the normal maps correctly [740, 960].

Normal variance can be introduced by high-curvature geometry as well as normal maps. Artifacts resulting from this variance are not mitigated by the previously discussed techniques. A different set of methods exists to address geometry normal variance. If a unique texture mapping exists over the geometry (often the case with characters, less so with environments) then the geometry curvature can be "baked" into the roughness map [740]. The curvature can also be estimated on the fly, using pixel-shader derivative instructions [740, 857, 1229, 1589, 1775, 1823]. This estimation can be done when rendering the geometry, or in a post-process pass, if a normal buffer is available.

The approaches discussed so far focus on specular response, but normal variance can affect diffuse shading as well. Taking account of the effect of normal variance on the $\mathbf{n} \cdot \mathbf{l}$ term can help increase accuracy of both diffuse and specular shading since both are multiplied by this factor in the reflectance integral [740].

Variance mapping techniques approximate the normal distribution as a smooth Gaussian lobe. This is a reasonable approximation if every pixel covers hundreds of thousands of bumps, so that they all average out smoothly. However, in many cases a pixel many cover only a few hundred or a few thousand bumps, which can lead to a "glinty" appearance. An example of this can be seen in Figure 9.25 on page 328, which is a sequence of images showing a sphere with bumps that diminish in size from image to image. The bottom right image shows the result when the bumps are small enough to average into a smooth highlight, but the images on the bottom left and bottom center show bumps that are smaller than a pixel but not small enough to average smoothly. If you were to observe an animated rendering of these spheres, the noisy highlight would appear as glints that sparkle in and out from frame to frame.

If we were to plot the NDF of such a surface, it would look like the left image in Figure 9.51. As the sphere animates, the $\mathbf{h}$ vector moves over the NDF and crosses over bright and dark areas, which causes the "sparkly" appearance. If we were to use variance mapping techniques on this surface, it would effectively approximate this NDF with a smooth NDF similar to the one on the right of Figure 9.51, losing the sparkly details.

In the film industry this is often solved with extensive supersampling, which is not feasible in real-time rendering applications and undesirable even in offline rendering.

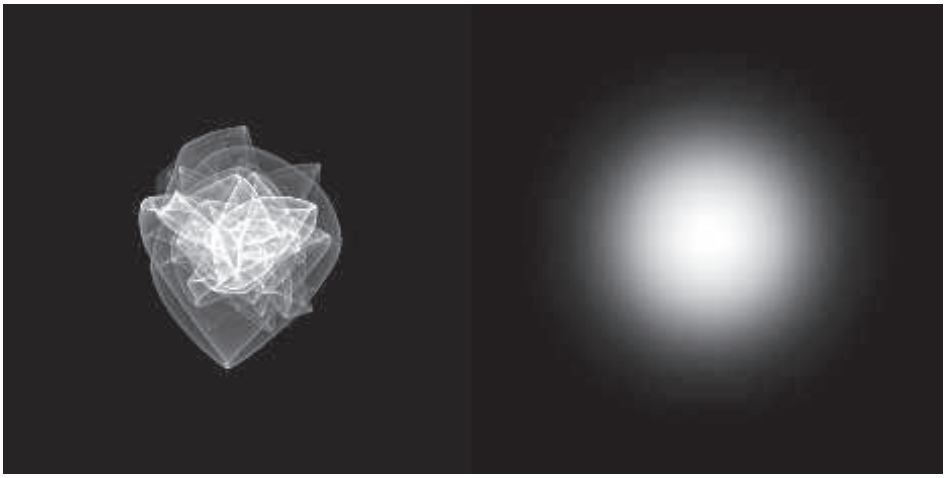

Figure 9.51. On the left is the NDF of a small patch (a few dozen bumps on a side) of a random bumpy surface. On the right is a Beckmann NDF lobe of approximately the same width. *(Image courtesy of Miloš Hašan.)*

Several techniques have been developed to address this issue. Some are unsuitable for real-time use, but may suggest avenues for future research [84, 810, 1941, 1942]. Two techniques have been designed for real-time implementation. Wang and Bowles [187, 1837] present a technique that is used to render sparkling snow in the game *Disney Infinity 3.0*. The technique aims to produce a plausible sparkly appearance rather than simulating a particular NDF. It is intended for use on materials such as snow that have relatively sparse sparkles. Zirr and Kaplanyan's technique [1974] simulates normal distributions on multiple scales, is spatially and temporally stable, and allows for a wider variety of appearances.

We do not have space to cover all of the extensive literature on material filtering, so we will mention a few notable references. Bruneton et al. [204] present a technique for handling variance on ocean surfaces across scales from geometry to BRDF, including environment lighting. Schilling [1565] discusses a variance mapping-like technique that supports anisotropic shading with environment maps. Bruneton and Neyret [205] provide a thorough overview of earlier work in this area.

Further Reading and Resources

McGuire's *Graphics Codex* [1188] and Glassner's *Principles of Digital Image Synthesis* [543, 544] are good references for many of the topics covered in this chapter. Some parts of Dutré's *Global Illumination Compendium* [399] are a bit dated (the BRDF models section in particular), but it is a good reference for rendering mathematics (e.g., spherical and hemispherical integrals). Glassner's and Dutré's references are both freely available online.

For the reader curious to learn more about the interactions of light and matter, we recommend Feynman's incomparable lectures [469] (available online), which were invaluable to our own understanding when writing the physics parts of this chapter. Other useful references include *Introduction to Modern Optics* by Fowles [492], which is a short and accessible introductory text, and *Principles of Optics* by Born and Wolf [177], a heavier (figuratively and literally) book that provides a more in-depth overview. *The Physics and Chemistry of Color* by Nassau [1262] describes the physical phenomena behind the colors of objects in great thoroughness and detail.

Chapter 10
Local Illumination

"Light makes right."
—Andrew Glassner

In Chapter 9 we discussed the theory of physically based materials, and how to evaluate them with punctual light sources. With this content we can perform shading computations by simulating how lights interact with surfaces, in order to measure how much radiance is sent in a given direction to our virtual camera. This spectral radiance is the scene-referred pixel color that will be converted (Section 8.2) to the display-referred color a given pixel will have in the final image.

In reality, the interactions that we need to consider are never punctual. We have seen in Section 9.13.1 how, in order to correctly evaluate shading, we have to solve the integral of the surface BRDF response over the entire *pixel footprint*, which is the projection of the pixel area onto the surface. This process of integration can also be thought as an antialiasing solution. Instead of sampling a shading function that does not have a bound on its frequency components, we pre-integrate.

Up to this point, the effects of only point and directional light sources have been presented, which limits surfaces to receive light from a handful of discrete directions. This description of lighting is incomplete. In reality, surfaces receive light from all incoming directions. Outdoors scenes are not just lit by the sun. If that were true, all surfaces in shadow or facing away from the sun would be black. The sky is an important source of light, caused by sunlight scattering from the atmosphere. The importance of sky light can be seen by looking at a picture of the moon, which lacks sky light because it has no atmosphere. See Figure 10.1.

On overcast days, and at dusk or dawn, outdoor lighting is all sky light. Even on a clear day, the sun subtends a cone when seen from the earth, so is not infinitesimally small. Curiously, the sun and the moon both subtend similar angles, around half a degree, despite their enormous size difference—the sun is two orders of magnitude larger in radius than the moon.

In reality, lighting is never punctual. Infinitesimal entities are useful in some situations as cheap approximations, or as building blocks for more complete models. In order to form a more realistic lighting model, we need to integrate the BRDF

Figure 10.1. Image taken on the moon, which has no sky light due to the lack of an atmosphere to scatter sunlight. This image shows what a scene looks like when it is lit by only a direct light source. Note the pitch-black shadows and lack of any detail on surfaces facing away from the sun. This photograph shows Astronaut James B. Irwin next to the Lunar Roving Vehicle during the Apollo 15 mission. The shadow in the foreground is from the Lunar Module. Photograph taken by Astronaut David R. Scott, Commander. *(Image from NASA's collection.)*

response over the full hemisphere of incident directions on the surface. In real-time rendering we prefer to solve the integrals that the rendering equation (Section 11.1) entails by finding closed-form solutions or approximations of these. We usually avoid averaging multiple samples (rays), as this approach tends to be much slower. See Figure 10.2.

This chapter is dedicated to the exploration of such solutions. In particular, we want to extend our shading model by computing the BRDF with a variety of non-punctual light sources. Often, in order to find inexpensive solutions (or any at all), we will need to approximate the light emitter, the BRDF, or both. It is important to evaluate the final shading results in a perceptual framework, understanding what elements matter most in the final image and so allocate more effort toward these.

We start this chapter with formulae to integrate analytic area light sources. Such emitters are the principal lights in the scene, responsible for most of the direct lighting

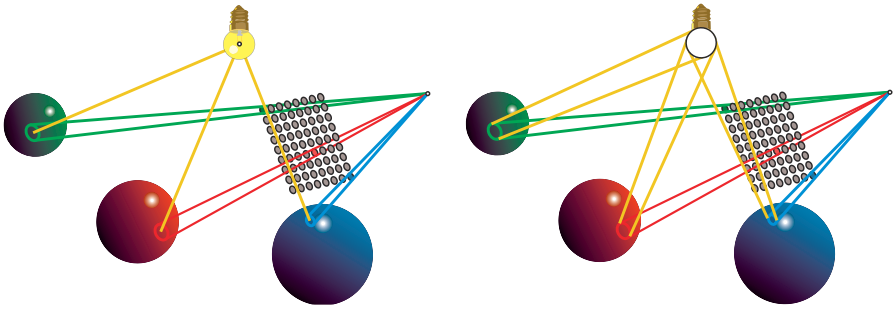

Figure 10.2. On the left, the integrals we have seen in Chapter 9: surface area and punctual light. On the right, the objective of this chapter will be to extend our shading mathematics to account for the integral over the light surface.

intensity, so for these we need to retain all of our chosen material properties. Shadows should be computed for such emitters, as light leaks will result in obvious artifacts. We then investigate ways to represent more general lighting environments, ones that consist of arbitrary distributions over the incoming hemisphere. We typically accept more approximated solutions in these cases. Environment lighting is used for large, complex, but also less intense sources of light. Examples include light scattered from the sky and clouds, *indirect light* bouncing off large objects in the scene, and dimmer direct area light sources. Such emitters are important for the correct balance of the image that would otherwise appear too dark. Even if we consider the effect of indirect light sources, we are still not in the realm of *global illumination* (Chapter 11), which depends on the explicit knowledge of other surfaces in the scene.

10.1 Area Light Sources

In Chapter 9 we described idealized, infinitesimal light sources: punctual and directional. Figure 10.3 shows the incident hemisphere on a surface point, and the difference between an infinitesimal source and an *area light source* with a nonzero size. The light source on the left uses the definitions discussed in Section 9.4. It illuminates the surface from a single direction $\mathbf{l}_c$. Its brightness is represented by its color $\mathbf{c}_{\text{light}}$, defined as the reflected radiance from a white Lambertian surface facing toward the light. The point or directional light's contribution to the outgoing radiance $L_o(\mathbf{v})$ in direction $\mathbf{v}$ is $\pi f(\mathbf{l}_c, \mathbf{v})\mathbf{c}_{\text{light}}(\mathbf{n} \cdot \mathbf{l}_c)^+$ (note the x^+ notation for clamping negative numbers to zero, introduced in Section 1.2). Alternatively, the brightness of the area light source (on the right) is represented by its radiance L_l. The area light *subtends* a solid angle ω_l from the surface location. Its contribution to the outgoing radiance in direction $\mathbf{v}$ is the integral of $f(\mathbf{l}, \mathbf{v})L_l(\mathbf{n} \cdot \mathbf{l})^+$ over ω_l.

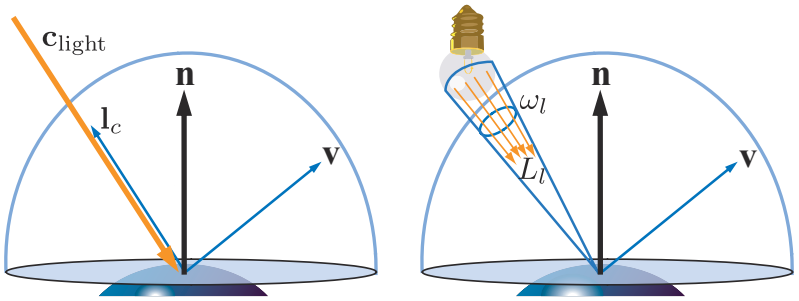

Figure 10.3. A surface illuminated by a light source, considering the hemisphere of possible incoming light directions defined by the surface normal $\mathbf{n}$. On the left, the light source is infinitesimal. On the right, it is modeled as an area light source.

The fundamental approximation behind infinitesimal light sources is expressed in the following equation:

$$L_o(\mathbf{v}) = \int_{\mathbf{l}\in\omega_l} f(\mathbf{l},\mathbf{v})L_l(\mathbf{n}\cdot\mathbf{l})^+ d\mathbf{l} \approx \pi f(\mathbf{l}_c,\mathbf{v})\mathbf{c}_{\text{light}}(\mathbf{n}\cdot\mathbf{l}_c)^+. \tag{10.1}$$

The amount that an area light source contributes to the illumination of a surface location is a function of both its radiance (L_l) and its size as seen from that location (ω_l). As we have seen in Section 9.4, point and directional light sources are approximations that cannot be realized in practice since their zero solid angle implies an infinite radiance. Understanding the visual errors that are introduced by the approximation will help to know when to use it, and what approach to take when it cannot be used. These errors will depend on two factors: how large the light source is, measured by the solid angle it covers from the shaded point, and how glossy the surface is.

Figure 10.4 shows how the specular highlight size and shape on a surface depends on both the material roughness and the size of the light source. For a small light source, one that subtends a tiny solid angle compared to the view angle, the error is small. Rough surfaces also tend to show the effect of the light source size less than polished ones. In general, both the area light emission toward a surface point and the specular lobe of the surface BRDF are spherical functions. If we consider the set of directions where the contributions of these two functions are significant, we obtain two solid angles. The determining factor in the error is proportional to the relative size of the emission angle compared to the size of the BRDF specular highlight solid angle.

Finally, note that the highlight from an area light can be approximated by using a punctual light and increasing the surface roughness. This observation is useful for deriving less-costly approximations to the area light integral. It also explains why in practice many real-time rendering system produce plausible results using only punctual sources: Artists compensate for the error. However, doing so is detrimental,

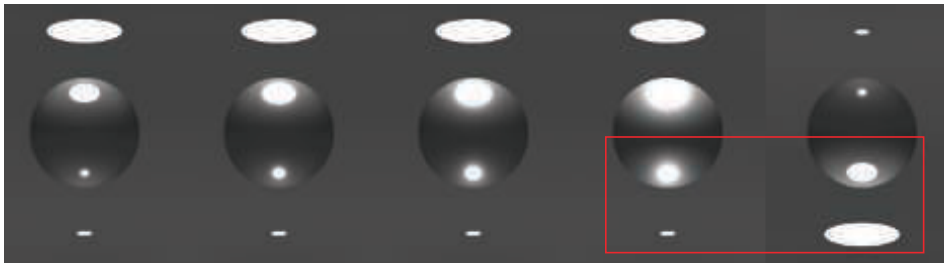

Figure 10.4. From left to right, the material of the sphere increases in surface roughness, using the GGX BRDF. The rightmost image replicates the first in the series, flipped vertically. Notice how the highlight and shading caused by a large disk light on a low-roughness material can look similar to the highlight caused by a smaller light source on a much rougher material.

as it couples material properties with the particular lighting setup. Content created this way will not look right when the lighting scenario is altered.

For the special case of Lambertian surfaces, using a point light for an area light can be exact. For such surfaces, the outgoing radiance is proportional to the irradiance:

$$L_o(\mathbf{v}) = \frac{\rho_{\text{ss}}}{\pi} E, \tag{10.2}$$

where ρ_{ss} is the subsurface albedo, or diffuse color, of the surface (Section 9.9.1). This relationship lets us use the equivalent of Equation 10.1 for computing irradiance, which is much simpler:

$$E = \int_{\mathbf{l} \in \omega_l} L_l (\mathbf{n} \cdot \mathbf{l})^+ d\mathbf{l} \approx \pi \mathbf{c}_{\text{light}} (\mathbf{n} \cdot \mathbf{l}_c)^+. \tag{10.3}$$

The concept of *vector irradiance* is useful to understand how irradiance behaves in the presence of area light sources. Vector irradiance was introduced by Gershun [526], who called it the *light vector*, and further extended by Arvo [73]. Using vector irradiance, an area light source of arbitrary size and shape can be accurately converted into a point or directional light source.

Imagine a distribution of radiance L_i coming into a point $\mathbf{p}$ in space. See Figure 10.5. We will assume for now that L_i is wavelength-independent and thus can be represented as a scalar. For every infinitesimal solid angle $d\mathbf{l}$ centered on an incoming direction $\mathbf{l}$, a vector is constructed that is aligned with $\mathbf{l}$ and has a length equal to the (scalar) radiance incoming from that direction times $d\mathbf{l}$. Finally, all these vectors are summed to produce the vector irradiance $\mathbf{e}$:

$$\mathbf{e}(\mathbf{p}) = \int_{\mathbf{l} \in \Theta} L_i(\mathbf{p}, \mathbf{l}) \, \mathbf{l} \, d\mathbf{l}, \tag{10.4}$$

where Θ indicates that the integral is performed over the entire sphere of directions.