Physically-Based Shading Models in Film and Game Production

SIGGRAPH 2010 Course Notes

Course Organizer

Naty Hoffman
Activision Studio Central

Presenters

Yoshiharu Gotanda
tri-Ace

Adam Martinez
Sony Pictures Imageworks

Ben Snow
ILM

Course Description

Physically grounded shading models have been known for many years, but they have only recently started to replace the "ad-hoc" models in common use for both film and game production. Compared to "ad-hoc" models, which require laborious tweaking to produce high-quality images, physically-based, energy-conserving shading models easily create materials that hold up under a variety of lighting environments. These advantages apply to both photorealistic and stylized scenes, and to game development as well as production of CG animation and computer VFX. Surprisingly, physically-based models are not more difficult to implement or evaluate than the traditional "ad-hoc" ones.

This course begins with a short explanation of the physics of light-matter interaction and how it is expressed in simple shading models. Then several speakers discuss specific examples of how shading models have been used in film and game production. In each case, the advantages of the new models are demonstrated, and drawbacks or issues arising from their usage are discussed. The course also includes descriptions of specific production techniques related to physically-based shading.

Level of Difficulty: Intermediate

Intended Audience

Practitioners from the videogame, CG animation, and VFX fields, as well as researchers interested in shading models.

Prerequisites

Basic familiarity with computer graphics, illumination and shading models in particular.

Course Website

All course materials can be found at `http://renderwonk.com/publications/s2010-shading-course`

Contact

Address questions or comments to `s2010course@renderwonk.com`

About the Presenters

Yoshiharu Gotanda is the CEO and CTO of tri-Ace, Inc, which is a game development studio in Japan.

Naty Hoffman is a Technical Director at Activision Studio Central, where he assists Activision's worldwide studios with graphics research and development. Prior to joining Activision in 2008, Naty worked for two years on *God of War III* at SCEA Santa Monica Studio. Naty has also worked at Naughty Dog (where he had an instrumental role in the development of the ICE libraries for first-party PS3 developers), at Westwood Studios (where he was graphics lead on *Earth and Beyond*) and at Intel as a microprocessor architect, assisting in the definition of the SSE and SSE2 instruction set extensions.

Adam Martinez is a Computer Graphics supervisor for Sony Pictures Imageworks and a member of the Shading Department, which oversees all aspects of shader writing and production rendering at Imageworks. He is a pipeline developer, look development artist, and technical support liaison for productions at the studio and he is one of the primary architects of Imageworks' rendering strategy behind *2012* and *Alice In Wonderland*. Adam started his career in commercial post houses and animation boutiques in New York City as a freelance computer graphics artist. He began his work in film visual effects on the project *Cremaster 3* by artist-filmmaker Matthew Barney. Since then he has served as both effects and lighting technical director, CG supervisor and pipeline developer for various studios in the San Francisco Bay Area. At ESC Entertainment, Adam led the effects team in the creation of complex insect crowd simulation tools for *Constantine* and destruction effects for *Matrix:Revolutions*. As computer graphics supervisor for The Orphanage on *Superman Returns*, Adam oversaw the creation of a ballistics simulation and rendering system. At Lucas Animation Adam was both a rendering pipeline developer and CG concept artist for television and feature animation. Adam's primary interest is in simulation and the construction of complex, but highly usable, systems for dynamic effects and rendering. Adam has a BA from Rutgers University.

Ben Snow studied computing and film at the University of Canberra. He started in Computer Graphics at while traveling in the U.K., then returned to Australia to set up the computer animation department for a company in Sydney. In 1994, Snow left Australia to join Industrial Light & Magic. At ILM he played a leading role in the R&D development for *Twister*, *Deep Impact*, *The Mummy* and *Pearl Harbor*. In 2002 he became visual effects supervisor on *Star Wars: Episode II—Attack of the Clones* for which he was honored with an Academy Award nomination for best achievement in visual effects. Snow also received Academy Award nominations for his work on *Pearl Harbor* and *Iron Man*. Snow went to Weta digital in October 2004 to work as a visual effects supervisor on *Peter Jackson's King Kong*. Returning to ILM in 2006, Snow visual effects supervised *Iron Man*, *Terminator Salvation*, and *Iron Man 2*. He's currently supervising ILM's work on *Pirates of the Carribean: On Stranger Tides*.

Presentation Schedule

2:00–2:30	**Background: Physically-Based Shading** *(Hoffman)*
2:–3:00	**Practical Implementation of Physically-Based Shading Models at tri-Ace** *(Gotanda)*
3:00–3:30	**Crafting Physically Motivated Shading Models for Game Development** *(Hoffman)*
3:30–3:45	**Break**
3:45–4:30	**Terminators and Iron Men: Image-Based Lighting and Physical Shading at ILM** *(Snow)*
4:30–5:00	**Faster Photorealism in Wonderland: Physically-Based Shading and Lighting at Sony Pictures Imageworks** *(Martinez)*
5:00–5:15	**Conclusion, Q&A** *(Gotanda, Hoffman, Martinez)*

Background: Physically-Based Shading

by Naty Hoffman

In this section of the course notes, we will go over the fundamentals behind physically-based shading models, starting with a qualitative description of the underlying physics, followed by a quantitative description of the relevant mathematical models, and finally discussing how these mathematical models can be implemented for shading.

The Physics of Shading

The physical phenomena underlying shading are those related to the interaction of light with matter. To understand these phenomena, it helps to have a basic understanding of the nature of light.

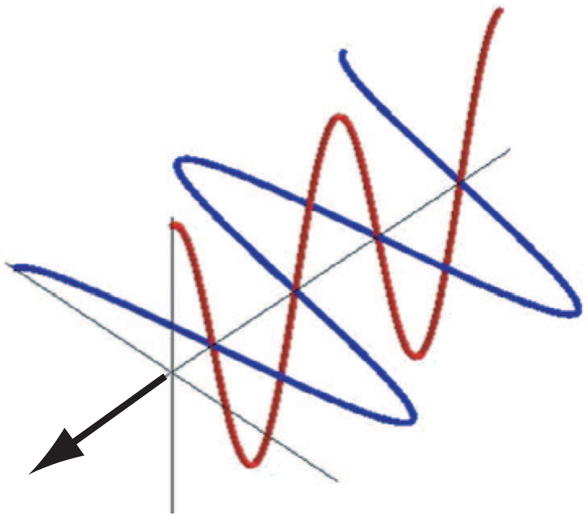


Figure 1: Light is an electromagnetic transverse wave.

Light is an electromagnetic *transverse wave*, which means that it oscillates in directions perpendicular to its propagation (see Figure 1).

Since light is a wave, it is characterized by its *wavelength*—the distance from peak to peak. Electromagnetic wavelengths cover a very wide range but only a tiny part of this range (about 400 to 700 nanometers) is visible to humans and thus of interest for shading (see Figure 2).

The effect matter has on light is defined by a property called the *refractive index*. The refractive index is a complex number; its real part measures how the matter affects the speed of light (slowing

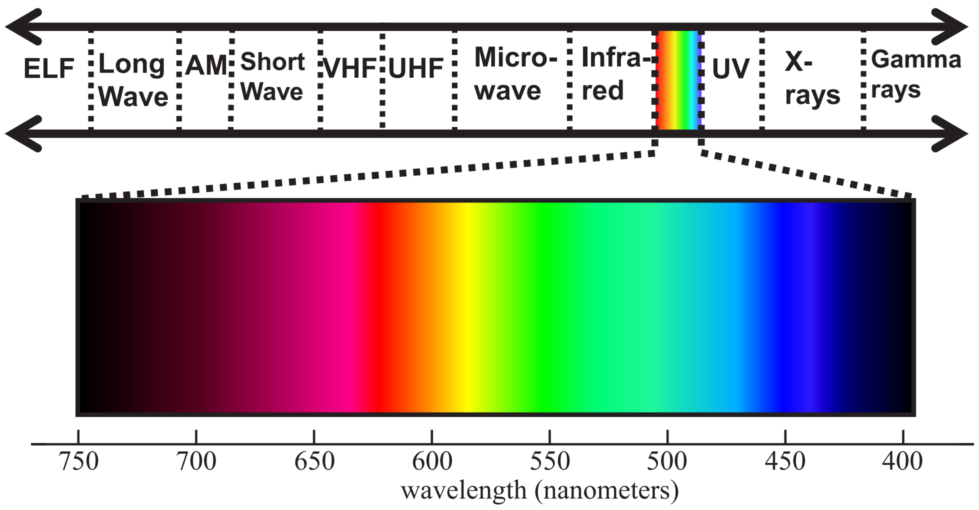


Figure 2: The visible spectrum.

it down relative to its speed in a vacuum) and the imaginary part determines whether the light is *absorbed* (converted to other forms of energy) as it propagates. The refractive index may vary as a function of light wavelength.

Homogeneous Media

The simplest case of light-matter interaction is light propagating through a *homogeneous medium*. This is a region of matter with uniform index of refraction (at the scale of the light wavelength; in the case of visible light this means that any variations much smaller than 100 nanometers or so don't count).

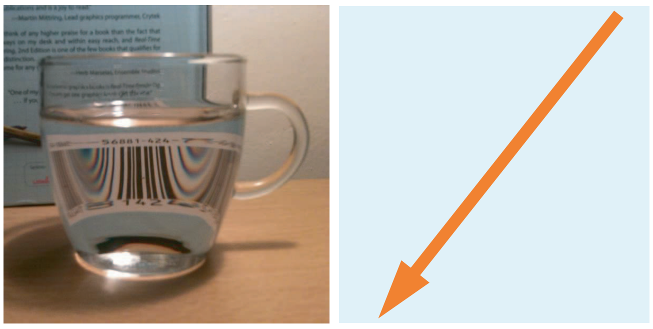


Figure 3: Light in transparent media like water and glass (left) just keeps on propagating in a straight line at the same intensity and color (right).

A *transparent* medium is one in which the complex part of the index of refraction is very low for visible light wavelengths; this means that there is no significant absorption and any light propagating through the medium just keeps on going in a straight line, unchanged. Examples of transparent media include water and glass (see Figure 3).

If a homogeneous medium does have significant absorptivity in the visible spectrum, it will absorb some amount of light passing through it. The farther the distance traveled by the light, the higher the absorption. However, the direction of the light will not change, just its intensity (and, if the absorptivity is selective to certain visible wavelengths, the color)—see Figure 4.

Note that the scale as well as the absorptivity of the medium matters. for example, water actually absorbs a little bit of visible light, especially on the red end of the spectrum. On a scale of inches this

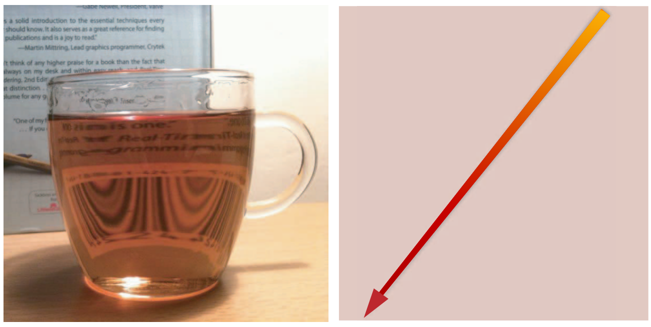


Figure 4: Light propagating through clear, absorbent media (left) continues in a straight line, but loses intensity (and may change color) with distance (right).

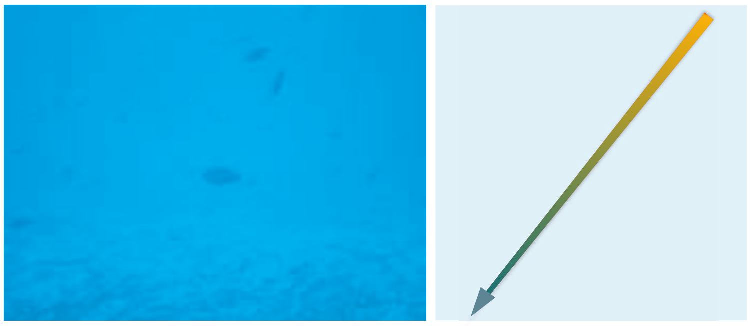


Figure 5: The slight absorptivity of water becomes significant over larger distances.

is negligible (as seen in Figure 3) but it is quite significant over many feet of distance; see Figure 5.

Scattering

In homogeneous media, light always continues propagating in a straight line and does not change its direction (although its amount can be reduced by absorption). A *heterogeneous medium* has variations in the index of refraction. If the index of refraction changes slowly and continuously, then the light bends in a curve. However, if the index of refraction changes abruptly, over a short distance (compared to the light wavelength), then the light *scatters*; it splits into multiple directions. Note that scattering does not change the overall amount of light.

Microscopic particles induce an isolated “island” where the refraction index differs from surrounding regions. This causes light to scatter continuously over all possible outgoing directions (see Figure 6). Note that the distribution of scattered light over different directions is typically not uniform and depends on the type of particle. Some cause forward scattering (more light goes in the forward direction), some cause backscattering (more light goes in the reverse of the original direction), and some have complex distributions with “spikes” in certain directions.

In *cloudy media*, the density of scattering elements is sufficient to somewhat randomize light propagation direction (Figure 7). In *translucent* or *opaque media* the density of scattering elements is so high that the light direction is completely randomized (Figure 8).

Like absorption, scattering depends on scale; a medium such as clean air which has negligible scattering over distances of a few feet causes substantial light scattering over many miles (Figure 9).

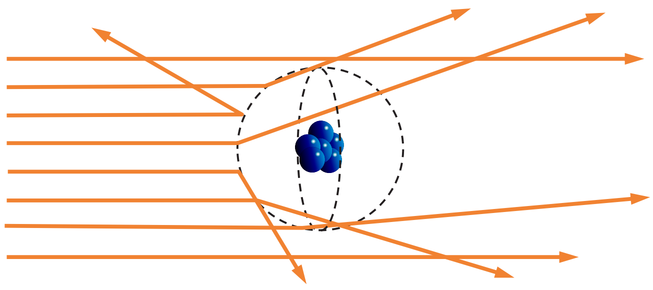


Figure 6: Particles cause light to scatter in all directions.

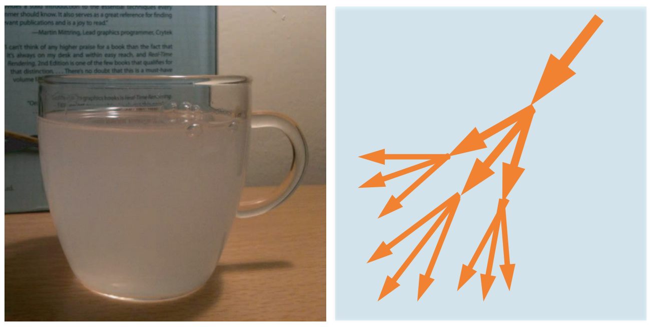


Figure 7: Light in cloudy media (left) has its direction somewhat randomized as it propagates (right).

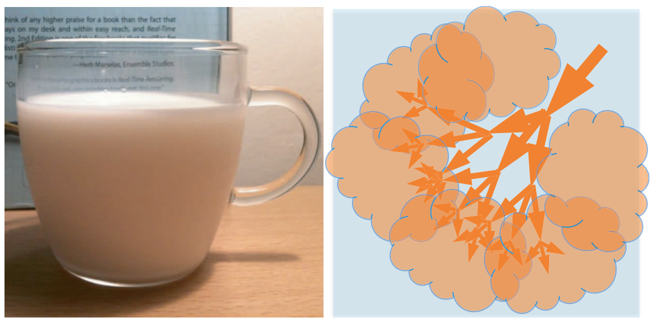


Figure 8: Light in translucent or opaque media (left) has its direction completely randomized as it propagates (right).

Figure 9: Even clean air causes considerable light scattering over a distance of miles.

Media Appearance

Previous sections discussed two different modes of interaction between matter and light. Regions of matter with complex-valued refraction indices cause absorption—the amount of light is lessened over distance (potentially also changing the light color if absorption occurs preferentially at certain wavelengths), but the light's direction does not change. On the other hand, rapid changes in the index of refraction cause scattering—the direction of the light changes (splitting up into multiple directions), but the overall amount or spectral distribution of the light does not change. There is a third mode of interaction—*emission*, where new light is created from other forms of energy (the opposite of absorption). This occurs in light sources, but it doesn't come up often in shading. Figure 10 illustrates the three modes of interaction.

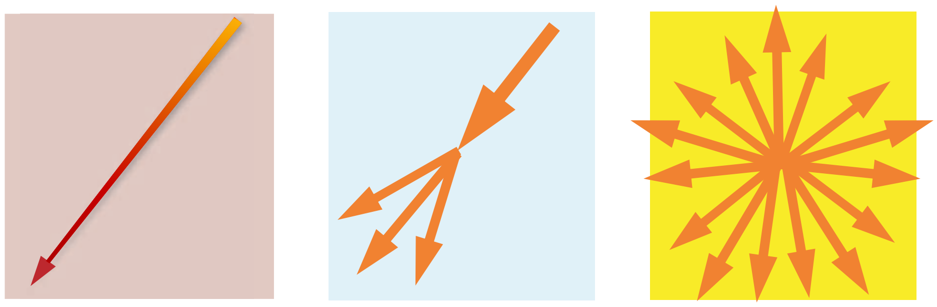


Figure 10: The three modes of interaction between light and matter: absorption (left), scattering (middle), and emission (right).

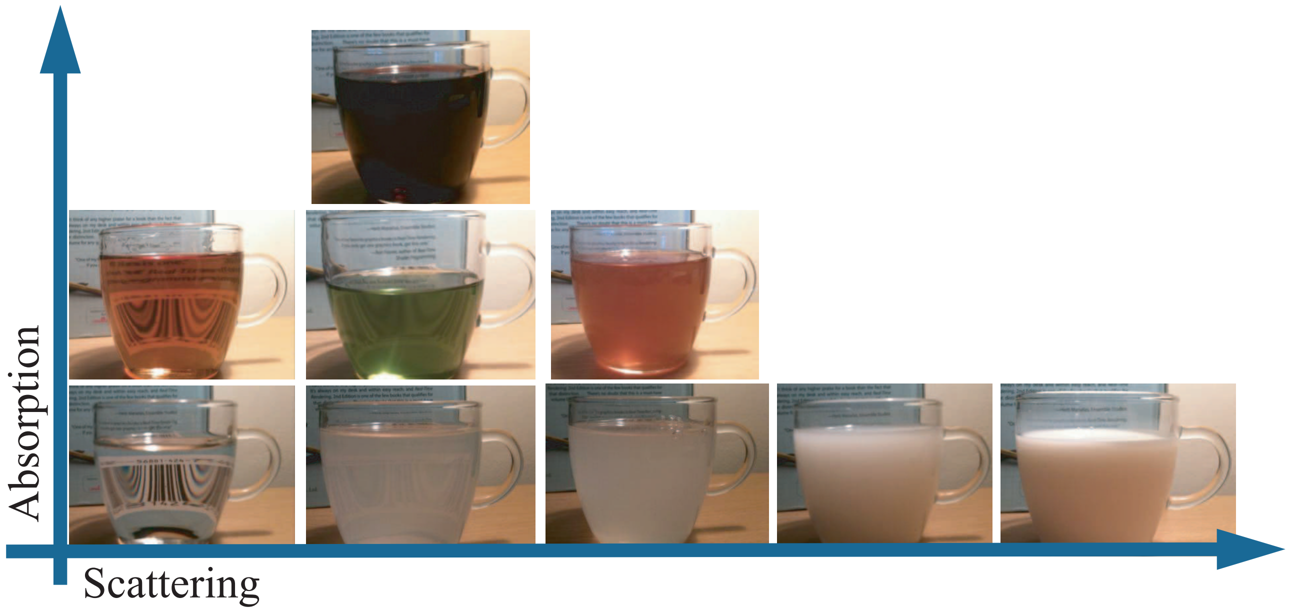


Figure 11: Media with varying amounts of light absorption and scattering.

Most media both scatter and absorb light to some degree. Each medium's appearance depends on the relative amount of scattering and absorption present. Figure 11 shows media with various combinations of scattering and absorptivity.

Scattering at a Planar Boundary

Maxwell's equations can be used to compute the behavior of light when the index of refraction changes, but in most cases analytical solutions do not exist. There is one special case which does have a solution, and it is of great relevance for shading. This is the case of an infinite, perfectly flat planar boundary between two volumes with different refractive indices. This is a good description of an object surface, with the refractive index of air on one side of the boundary, and the refractive index of the object on the other. The solutions to Maxwell's equations in this special case are called the *Fresnel equations*.

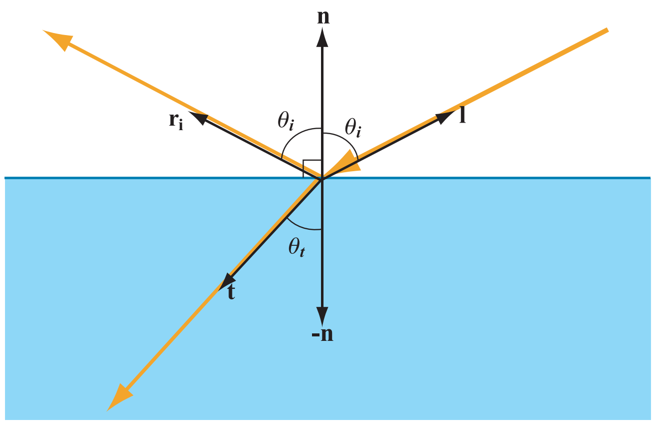


Figure 12: Refractive index changes at planar boundaries cause light to scatter in two directions (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

Although real object surfaces are not infinite, in comparison with the wavelength of visible light they can be treated as such. As for being "perfectly flat", an objection might be raised that no object's surface can truly be flat—if nothing else, individual atoms will form pico-scale "bumps". However, as with everything else, the scale relative to the light wavelength matters. It is indeed possible to make surfaces that are perfectly flat at the scale of hundreds of nanometers—such surfaces are called *optically flat* and are typically used for high-quality optical instruments such as telescopes.

In the special case of a planar refractive index boundary, instead of scattering in a continuous fashion over all possible directions, light splits into exactly two directions: reflection and refraction (see Figure 12).

As you can see in Figure 12, the angle of reflection is equal to the incoming angle, but the angle of refraction is different. The angle of refraction depends on the refractive index of the medium (if you are interested in the exact math, look up *Snell's Law*). The proportions of reflected and refracted light are described by the Fresnel equations, and will be discussed in a later section.

Non-Optically-Flat Surfaces

Of course, most real-world surfaces are not polished to the same tolerances as telescope mirrors. What happens with surfaces that are not optically flat? In most cases, there are indeed irregularities present which are much larger than the light wavelength, but too small to be seen or resolved (i.e., they are smaller than the coverage area of a single pixel or shading sample). In this case, the surface behaves like a large collection of tiny optically flat surfaces. The surface appearance is the aggregate result of many points with different surface orientations—each point reflects incoming light in a slightly different direction (see Figure 13).

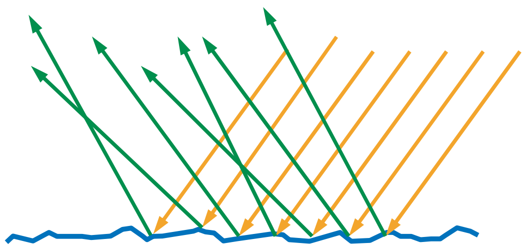


Figure 13: Visible reflections from non-optically flat surfaces are the aggregate result of reflections from many surface points with different orientations (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

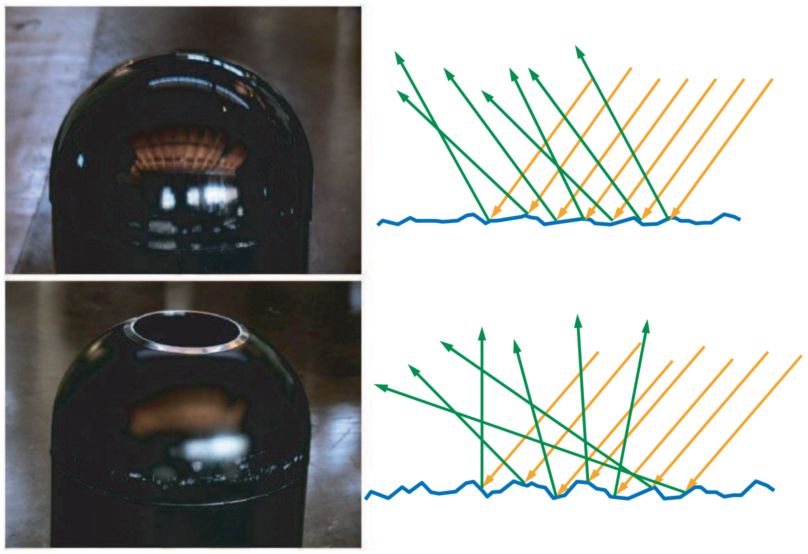


Figure 14: On the top row, the surface is relatively smooth; the surface orientation only varies slightly, resulting in a small variance in reflected light directions and thus sharper reflections. The surface on the bottom row is rougher; different points on the surface have widely varying orientations, resulting in a high variance in reflected light directions and thus blurry reflections. Note that both surfaces appear smooth at the visible scale—the roughness difference is at the microscopic scale (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

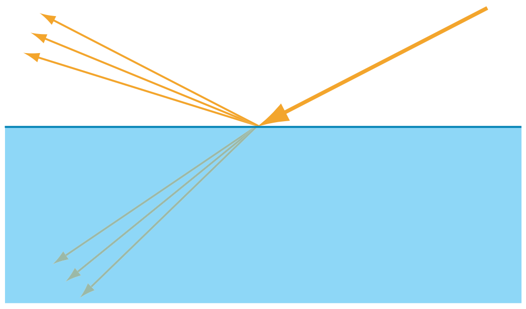


Figure 15: When viewed macroscopically, non-optically flat surfaces can be treated as reflecting (and refracting) light in multiple directions (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

The rougher the surface is at this microscopic scale, the blurrier the reflections as the surface orientations diverge more strongly from the overall, macroscopic surface orientation (see Figure 14).

For shading purposes, it is common to treat this *microgeometry* statistically and view the surface as reflecting (and refracting) light in multiple directions (see Figure 15).

Subsurface Scattering

What happens to the refracted light? This depends on the composition of the object. Metals have very high absorption coefficients (imaginary part of refractive index) in the visible spectrum. All refracted light is immediately absorbed (soaked up by free electrons). On the other hand, non-metals (also referred to as dielectrics or insulators) behave as regular participating media once the light is refracted inside them, exhibiting the range of absorption and scattering behaviors we covered in previous sections. In most cases, some of the refracted light is scattered enough to be re-emitted out of the same surface. Both of these cases are illustrated in Figure 16.

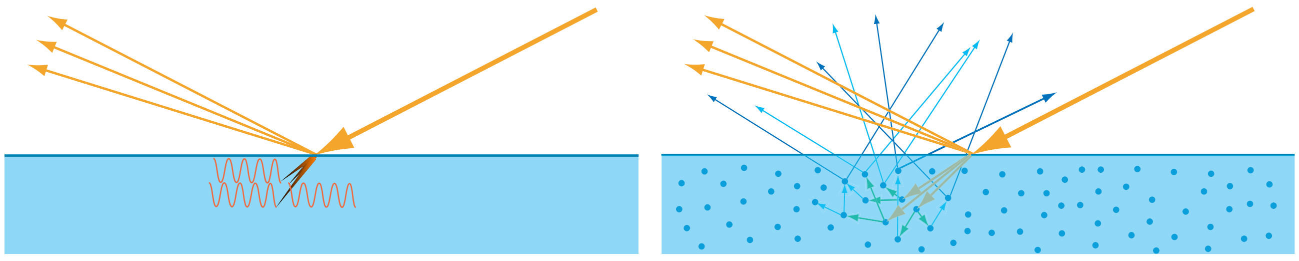


Figure 16: In metals (on the left), all refracted light energy is immediately absorbed by free electrons; in non-metals (on the right) refracted light energy scatters until it re-emerges from the surface, typically after undergoing partial absorption (*images from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

On the right side of Figure 16, you can see that the subsurface-scattered light (denoted with blue arrows) is emitted from various points on the surface, at varying distances from the original entrance point of the light. Figure 17 shows the relationship between these distances and the pixel size in two cases. On the upper left, the pixel is larger than the entry-to-exit subsurface scattering distances. In this case, the entry-to-exit distances can be ignored and the subsurface scattered light can be assumed to enter and exit the surface at the same point, as seen on the upper right. This allows shading to be handled as a completely local process; the outgoing light at a point only depends on incoming light at the same point. On the bottom of Figure 17, the pixel is smaller than the entry-to-exit distances. In this case, the shading of each point is affected by light impinging on other points. To capture this effect, local shading will not suffice and specialized rendering techniques need to be used. These are typically referred to as "subsurface scattering" techniques, but it is important to note that "ordinary" diffuse shading is the result of the same physical phenomena (subsurface scattering of refracted light). The only difference is the scattering distance relative to the scale of observation. This insight tells us that materials which are commonly thought of as exhibiting "subsurface scattering" behavior can be handled with regular diffuse shading at larger distances (e.g. the skin of a distant character) . On the other hand, materials which are thought of as exhibiting "regular diffuse shading" behavior will have a "subsurface scattering" appearance when viewed very close up (e.g. an extreme close-up of a small plastic toy).

The Mathematics of Shading

The measurement of electromagnetic radiation in general (including visible light) is called *radiometry.* There are various radiometric quantities used to measure light over surfaces, over directions, etc.; we

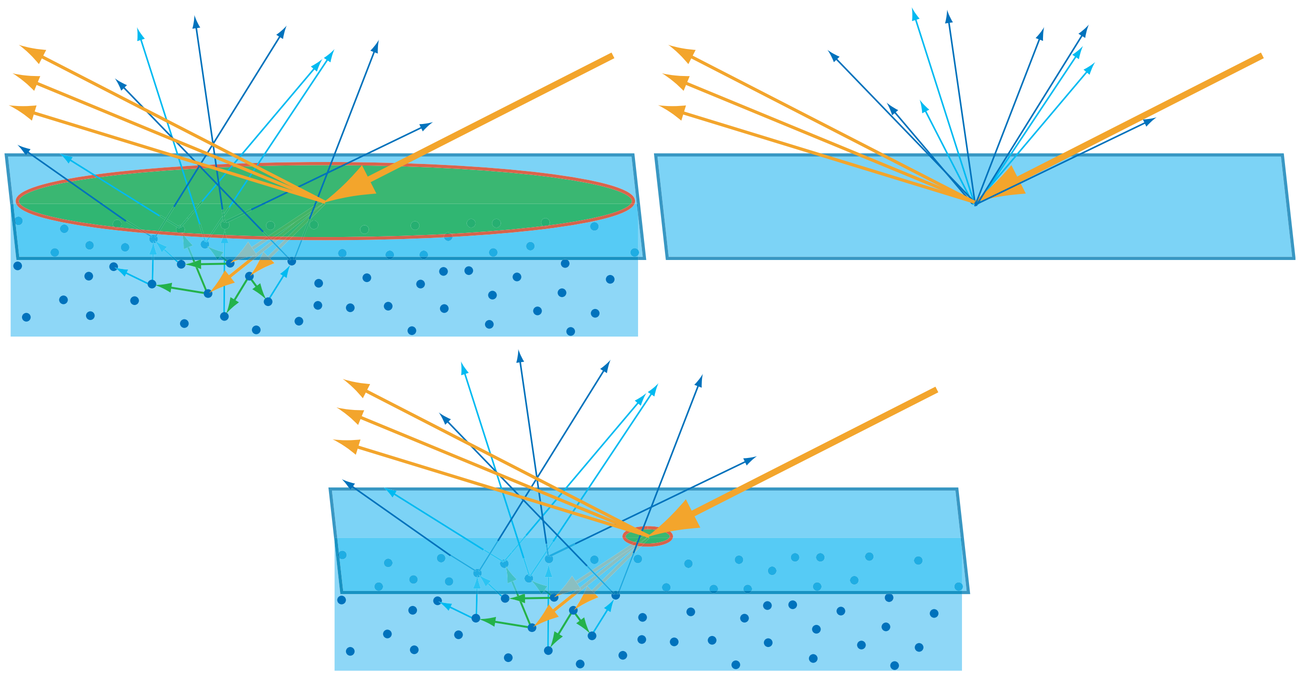


Figure 17: On the upper left, the pixel (green circle with red border) is larger than the distances traveled by the light before it exits the surface. In this case, the outgoing light can be assumed to be emitted from the entry point (upper right). On the bottom, the pixel is smaller than the scattering distances; these distances cannot be ignored if realistic shading is desired. (*images from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

will only concern ourselves with *radiance*, which is used to quantify the magnitude of light along a single ray[1]. We will use the common radiometric notation L to denote radiance; when shading a surface point, L_i denotes radiance incoming to the surface and L_o denotes outgoing radiance.

Radiance (like other radiometric quantities) is a *spectral quantity* - the amount varies as a function of wavelength. In theory, to express visible-light radiance a continuous spectral distribution needs to be stored. Dense spectral samples are indeed used in some specialized rendering applications, but for all production (film and game) rendering, RGB triples are used instead. An explanation of how these triples relate to spectral distributions can also be found in many websites and books, including *Real-Time Rendering* [20].

The BRDF

It is most commonly assumed that shading can be handled locally (as illustrated on the upper right of Figure 17). In this case, how a given surface point responds to light only depends on the incoming (light) and outgoing (view) directions. In this document, we will use $\mathbf{v}$ to denote a unit-length vector pointing along the outgoing direction and $\mathbf{l}$ to denote a unit-length vector pointing opposite to the incoming direction (it is convenient to have all vectors point away from the surface). The surface's response to light is quantified by a function called the BRDF (Bidirectional Reflectance Distribution Function), which we will denote as $f(\mathbf{l}, \mathbf{v})$. Each direction (incoming and outgoing) can be parameterized with two numbers (e.g. polar coordinates), so the overall dimensionality of the BRDF is four. In many cases, rotating the light and view directions around the surface normal does not affect the BRDF. Such *isotropic BRDFs* can be parameterized with three angles (see Figure 18). In practice, the number of

[1]An explanation of other radiometric quantities can be found in various texts, including Chapter 7 of the 3rd edition of *Real-Time Rendering* [20] and Dutré's *Global Illumination Compendium* [10].

angles used to compute a given BRDF commonly varies from one to five—some commonly used angles are shown in Figure 19.

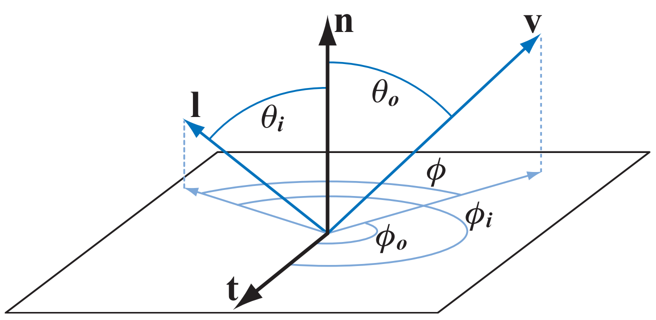


Figure 18: The BRDF depends on incoming and outgoing directions; these can be parameterized with four angles, or three in the case of isotropic BRDFs. Here $\mathbf{n}$ is the surface normal vector, $\mathbf{l}$ is the incoming light direction vector, $\mathbf{v}$ is the outgoing (view) direction vector, and $\mathbf{t}$ is a tangent vector defining a preferred direction over the surface (this is only used for *anisotropic BRDFs* where the reflection behavior changes when light and view vector are rotated around $\mathbf{n}$). (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

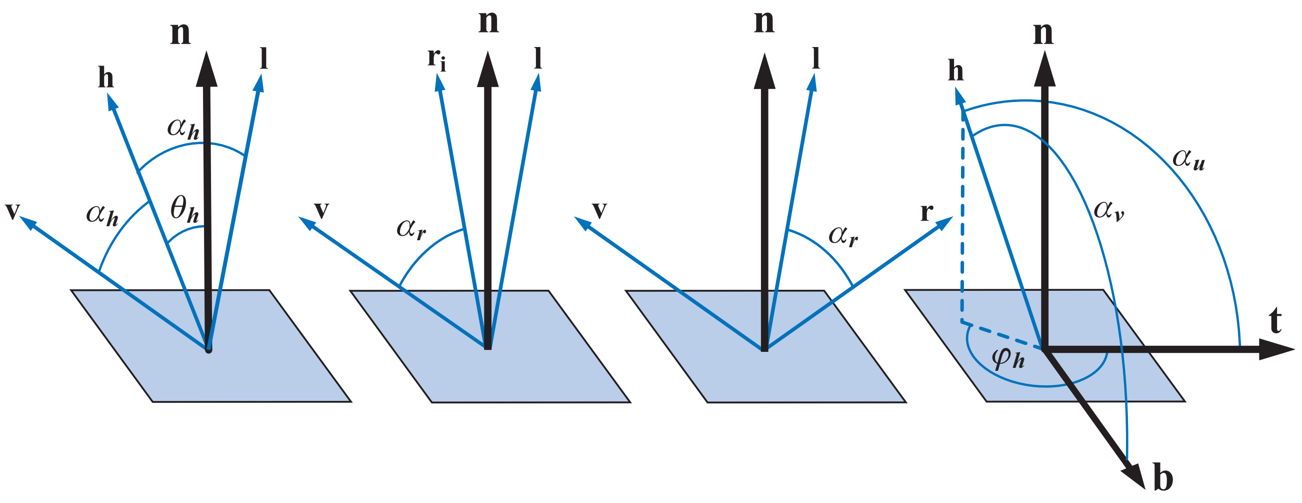


Figure 19: Examples of some angles which are commonly used in BRDF evaluation, in addition to those in Figure 18 (*images from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

In principle, the BRDF is only defined for light and view directions above the surface; in other words, the dot products $(\mathbf{n}\cdot\mathbf{l})$ and $(\mathbf{n}\cdot\mathbf{v})$ must both be non-negative (recall that the dot product between two unit-length vectors is equal to the cosine of the angle between them; if this is negative, then the angle exceeds 90°). In production shading, situations arise when shading needs to be performed for angles outside this range (for example, normal mapping can result in normal vectors backfacing to the view vector). This is typically handled in practice by clamping the dot product to 0, but other approaches are possible [25].

The BRDF can be intuitively interpreted in two ways; both are valid. The first interpretation is that given a ray of light incoming from a certain direction, the BRDF gives the relative distribution of reflected and scattered light over all outgoing directions above the surface. The second interpretation is that for a given view direction, the BRDF gives the relative contribution of light from each incoming direction to the outgoing light. both interpretations are illustrated in Figure 20.

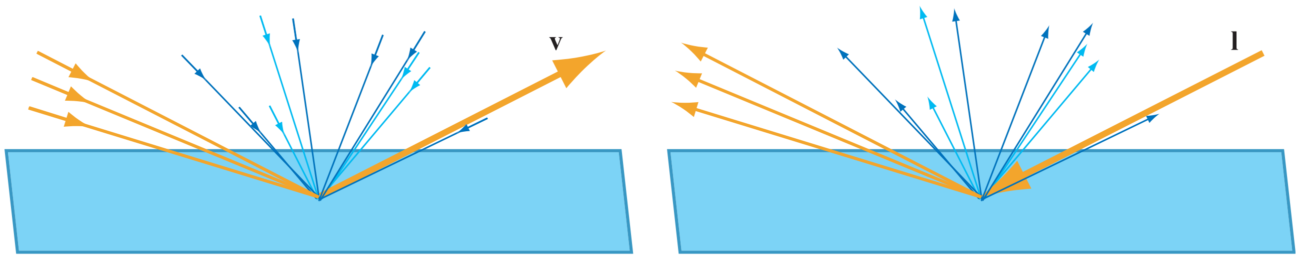


Figure 20: On the left side, we see one interpretation of the BRDF - that for a given outgoing (view) direction, it specifies the relative contributions of incoming light. On the right side we see an alternative interpretation - that for a given incoming light direction, it specifies the distribution of outgoing light. (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

The BRDF is a spectral quantity. In theory the input and output wavelengths would need to be additional BRDF inputs, increasing its dimensionality. However, in practice there is no cross-talk between the individual weavelengths[2]; each wavelength of outgoing light is only affected by that same wavelength in the incoming light. This means that instead of treating input and output wavelengths as BRDF inputs, we (more simply) treat the BRDF as an spectral-valued function that is multiplied with spectral-valued light colors. In production shading, this means an RGB-valued BRDF multiplied by RGB-valued light colors.

The BRDF is used in the *reflectance equation*[3]:

$$L_o(\mathbf{v}) = \int_\Omega f(\mathbf{l},\mathbf{v}) \otimes L_i(\mathbf{l})(\mathbf{n}\cdot\mathbf{l})d\omega_i. \tag{1}$$

Although this equation may seem a bit daunting, its meaning is straightforward: outgoing radiance equals the integral (over all directions above the surface) of incoming radiance times the BRDF and a cosine factor. If you are not familiar with integrals, you can think of them as a kind of continuous weighted average. The $\otimes$ symbol is used here to denote component-wise vector multiplication; it is used because both BRDF and light color are spectral (RGB) vectors.

Not any arbitrary function over incoming and outgoing directions can make sense as a BRDF. It is commonly recognized that are two properties a BRDF must have to be *physically plausible*: *reciprocity* and *energy conservation*. Reciprocity simply means that the BRDF has the same value if $\mathbf{l}$ and $\mathbf{v}$ are swapped:

$$f(\mathbf{l},\mathbf{v}) = f(\mathbf{v},\mathbf{l}). \tag{2}$$

Energy conservation refers to the fact that a surface cannot reflect more than 100% of incoming light energy. Mathematically, it is expressed via the following equation:

$$\forall \mathbf{l}, \int_\Omega f(\mathbf{l},\mathbf{v})(\mathbf{n}\cdot\mathbf{v})d\omega_o \leq 1. \tag{3}$$

This means that for any possible light direction $\mathbf{l}$, the integral of the BRDF times a cosine factor over outgoing directions $\mathbf{v}$ must not exceed 1.

The phenomena described by the BRDF includes (at least for non-metals) two distinct physical phenomena—surface reflection and subsurface scattering. Since each of these phenomena has different behavior, BRDFs typically include a separate term for each one. The BRDF term describing surface reflection is usually called the *specular term* and the term describing subsurface scattering is called the *diffuse term*; see Figure 21.

[2]There are two physical phenomena involving such crosstalk—fluorescence and phosphorescence; but they rarely occur in production shading.

[3]The reflectance equation is a special case of the *rendering equation* [16].

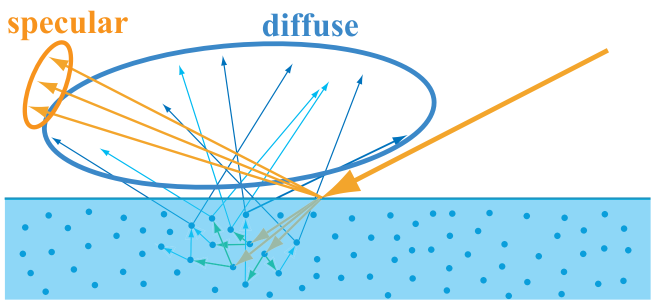


Figure 21: BRDF specular terms are typically used for surface reflection, and BRDF diffuse terms for subsurface scattering. (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

Surface Reflectance (Specular Term)

The basis for most physically-based specular BRDF terms is *microfacet theory*. This theory was developed to describe surface reflection from general (non-optically flat) surfaces. The basic assumption underlying microfacet theory is that the surface is composed of many *microfacets*, too small to be seen individually. Each microfacet is assumed to be optically flat. As mentioned in the previous section, an optically flat surface splits light into exactly two directions—reflection and refraction.

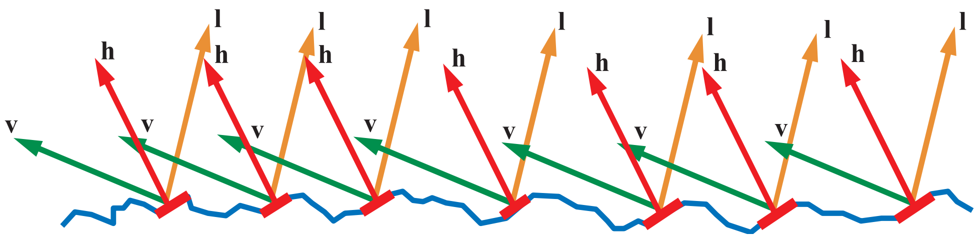


Figure 22: Microfacets with $\mathbf{m} = \mathbf{h}$ are oriented to reflect $\mathbf{l}$ into $\mathbf{v}$—other microfacets do not contribute to the BRDF. (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

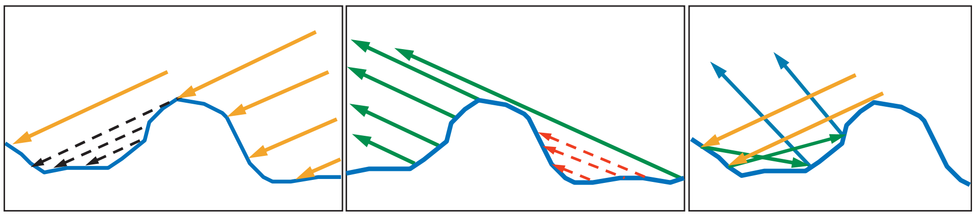


Figure 23: On the left, we see that some microfacets are occluded from the direction of $\mathbf{l}$, so they are shadowed and do not receive light (so they cannot reflect any). In the center, we see that some microfacets are not visible from the view direction $\mathbf{v}$, so of course any light reflected from them will not be seen. In both cases these microfacets do not contribute to the BRDF. In reality, shadowed light does not simply vanish; it continues to bounce from the microfacets and some of it does make its way into the view direction (as see on the right side). These *interreflections* are ignored by microfacet theory. (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

Each of these microfacets reflects light from a given incoming direction into a single outgoing direction which depends on the orientation of the microfacet normal $\mathbf{m}$. When evaluating a BRDF term, both the light direction $\mathbf{l}$ and the view direction $\mathbf{v}$ are specified. This means that of all the millions of microfacets on the surface, only those that happen to be angled just right to reflect $\mathbf{l}$ into $\mathbf{v}$ have any contribution to the BRDF value. In Figure 22 we can see that these *active microfacets* have their surface normal $\mathbf{m}$ oriented exactly halfway between $\mathbf{l}$ and $\mathbf{v}$. The vector halfway between $\mathbf{l}$ and $\mathbf{v}$ is called the *half-vector* or *half-angle vector*; we will denote it as $\mathbf{h}$.

Not all microfacets for which $\mathbf{m} = \mathbf{h}$ will contribute to the reflection; some are blocked by other microfacets from the direction of $\mathbf{l}$ (*shadowing*), from the direction of $\mathbf{v}$ (*masking*), or from both. Microfacet theory assumes that all shadowed light is lost from the specular term; in reality, due to multiple surface reflections some of it will eventually be visible, but this is not accounted for in microfacet theory. This is not typically a major source of error in most cases (rough metal surfaces are a possible exception). The various types of light-microfacet interaction are shown in Figure 23.

With these assumptions (optically flat microfacets, no interreflections), a specular BRDF term can be derived from first principles ([1, 25]). The microfacet specular BRDF term has the following form[4]:

$$f_{\mu\text{facet}}(\mathbf{l},\mathbf{v}) = \frac{F(\mathbf{l},\mathbf{h})G(\mathbf{l},\mathbf{v},\mathbf{h})D(\mathbf{h})}{4(\mathbf{n}\cdot\mathbf{l})(\mathbf{n}\cdot\mathbf{v})} \tag{4}$$

We will go into each of the terms in more detail, but first a quick summary. $F(\mathbf{l},\mathbf{h})$ is the Fresnel reflectance of the active microfacets as a function of the light direction $\mathbf{l}$ and the active microfacet normal $\mathbf{m} = \mathbf{h}$. $G(\mathbf{l},\mathbf{v},\mathbf{h})$ is the proportion of microfacets (of the ones with $\mathbf{m} = \mathbf{h}$) which are *not* shadowed or masked, as a function of the light direction $\mathbf{l}$, the view direction $\mathbf{v}$, and the active microfacet normal $\mathbf{m} = \mathbf{h}$. $D(\mathbf{h})$ is the microfacet normal distribution function evaluated at the active microfacet normal $\mathbf{m} = \mathbf{h}$; in other words, the concentration of microfacets with normals equal to $\mathbf{h}$. Finally, the denominator $4(\mathbf{n}\cdot\mathbf{l})(\mathbf{n}\cdot\mathbf{v})$ is a correction factor which accounts for quantities being transformed between the local space of the microfacets and that of the overall macrosurface.

Fresnel Reflectance Term

The Fresnel reflectance term computes the fraction of light reflected from an optically flat surface. Its value depends on two things: the incoming angle (angle between light vector and surface normal) and the refractive index of the material. Since the refractive index may vary over the visible spectrum, the Fresnel reflectance is a spectral quantity—for production purposes, an RGB triple. We also know that each of the RGB values have to lie within the 0 to 1 range, since a surface cannot reflect less than 0% or more than 100% of the incoming light. Since we are only concerned with active microfacets for which $\mathbf{m} = \mathbf{h}$, the incidence angle for Fresnel reflectance is actually the one between $\mathbf{l}$ and $\mathbf{h}$.

The full Fresnel equations are somewhat complex, and the required material parameter (complex refractive index sampled densely over the visible spectrum) is not particularly convenient for artists (to say the least). However, a simpler expression with more convenient parametrization can be derived by inspecting the behavior of these equations for real-world materials. With this in mind, let us inspect the graph in Figure 24.

The materials selected for this graph represent a wide variety. Despite this, some common elements can be seen. Reflectance is almost constant for incoming angles between 0° and about 45°. The reflectance changes more significantly (typically, but not always, increasing somewhat) between 45° and about 75°. Finally, between 75° and 90° reflectance always goes rapidly to 1 (white, if viewed as an RGB triple). Since the Fresnel reflectance stays close to the value for 0° over most of the range, we can think of this value $F(0^\circ)$ as the *characteristic specular reflectance* of the material. This value has all the properties of what is typically thought of as a "color"—it is composed of RGB values between 0 and 1, and it is a measure of selective reflectance of light. For this reason, we will also refer to this value as the *specular color* of the surface, denoted as $\mathbf{c}_{\text{spec}}$.

[4]Note that cases where one or both of the dot products in the denominator are negative or zero need to be handled, although in theory this is outside the domain over which the BRDF is defined. In practice, this is handled by clamping the dot products to a very small positive value, though some authors [25] recommend using absolute value instead of clamping.

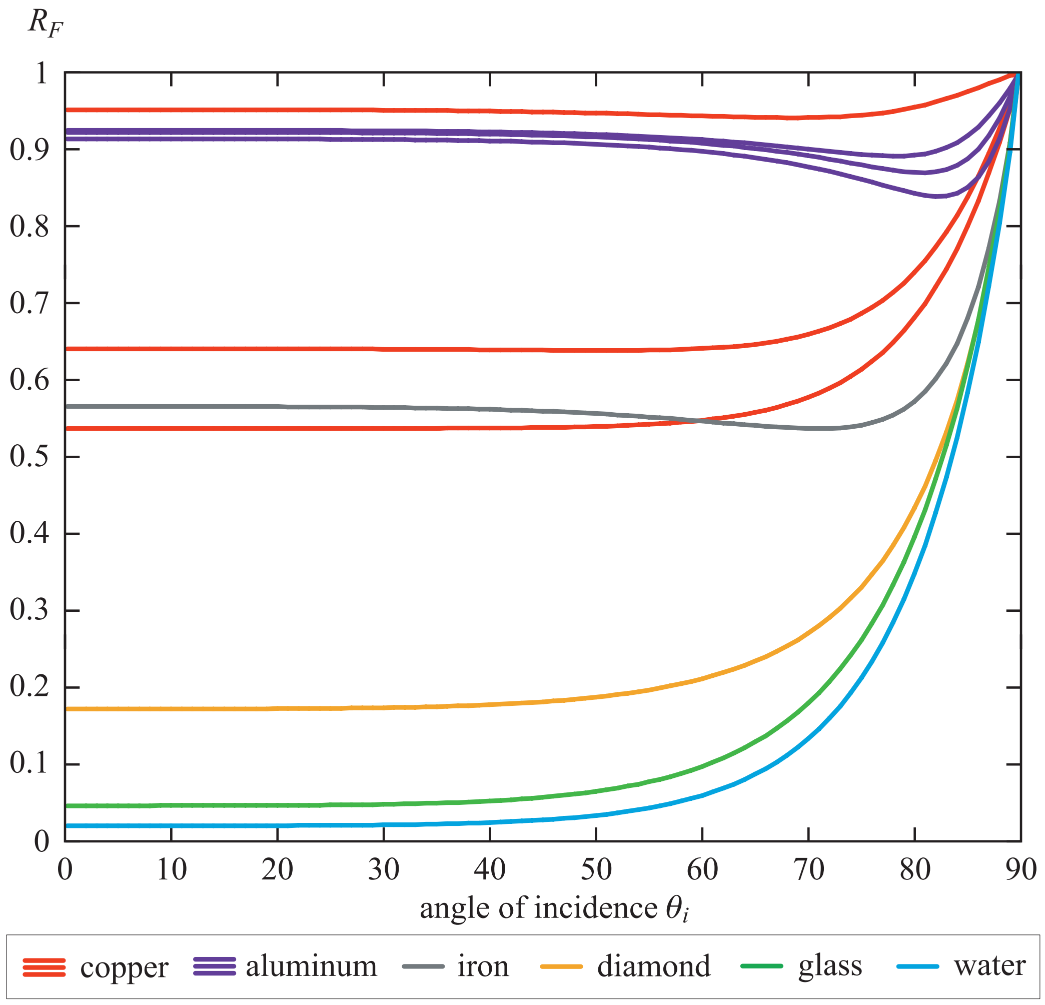


Figure 24: Fresnel reflectance for external reflection from a variety of substances. Since copper and aluminum have significant variation in their reflectance over the visible spectrum, their reflectance is shown as three separate curves for R, G, and B. Copper's R curve is highest, followed by G, and finally B (thus its reddish color). Aluminum's B curve is highest, followed by G, and finally R. (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

$\mathbf{c}_{\text{spec}}$ looks like an ideal parameter for a Fresnel reflectance approximation, and indeed Schlick [22] gives a cheap and reasonably accurate approximation that uses it:

$$F_{\text{Schlick}}(\mathbf{c}_{\text{spec}}, \mathbf{l}, \mathbf{n}) = \mathbf{c}_{\text{spec}} + (1 - \mathbf{c}_{\text{spec}})(1 - (\mathbf{l} \cdot \mathbf{n}))^5 \tag{5}$$

This approximation is widely used in computer graphics. In the special case of active microfacets, $\mathbf{h}$ must be substituted for the surface normal $\mathbf{n}$:

$$F_{\text{Schlick}}(\mathbf{c}_{\text{spec}}, \mathbf{l}, \mathbf{h}) = \mathbf{c}_{\text{spec}} + (1 - \mathbf{c}_{\text{spec}})(1 - (\mathbf{l} \cdot \mathbf{h}))^5 \tag{6}$$

To know which values are reasonable to assign to $\mathbf{c}_{\text{spec}}$, it is instructive to look at the values of $F(0°)$ for various real-world materials. These can be found in Table 1. Values are given in both linear and gamma (sRGB) space; we recommend anyone unfamiliar with the importance of computing shading in linear space and the issues involved in converting input from gamma space consult some of the articles on the topic ([14, 15]).

Material	$F(0°)$ **(Linear)**	$F(0°)$ **(sRGB)**	**Color**
Water	0.02,0.02,0.02	0.15,0.15,0.15	⬛
Plastic / Glass (Low)	0.03,0.03,0.03	0.21,0.21,0.21	⬛
Plastic High	0.05,0.05,0.05	0.24,0.24,0.24	⬛
Glass (High) / Ruby	0.08,0.08,0.08	0.31,0.31,0.31	⬛
Diamond	0.17,0.17,0.17	0.45,0.45,0.45	⬛
Iron	0.56,0.57,0.58	0.77,0.78,0.78	⬜
Copper	0.95,0.64,0.54	0.98,0.82,0.76	🟧
Gold	1.00,0.71,0.29	1.00,0.86,0.57	🟨
Aluminum	0.91,0.92,0.92	0.96,0.96,0.97	⬜
Silver	0.95,0.93,0.88	0.98,0.97,0.95	⬜

Table 1: Values of $F(0°)$ for various materials. (*table from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

When inspecting Table 1, several things stand out. One is that metals have significantly higher values of $F(0°)$ than non-metals. Iron is a very dark metal, and it reflects more than 50% of incoming light at 0°. Recall that metals have no sub-surface reflectance; a bright specular color and no diffuse color is the distinguishing visual characteristic of metals. On the other hand diamond, one of the brightest non-metals, reflects only 17% of incoming light at 0°; most non-metals reflect significantly less than that. Very few materials have values in the "no mans land" between 20% and 40%; these are typically semiconductors and other exotic materials which are unlikely to appear in production shading situations. The same is true for values lower than 2% (the $F(0°)$ value of water). In fact, ruling out metals, gemstones, and crystals, pretty much any material you are likely to see outside a laboratory will have a narrow range of $F(0°)$ values—between 2% and 5%.

Normal Distribution Term

In most surfaces, the microfacet's orientations are not uniformly distributed. Microfacet normals closer to the macroscopic surface normal tend to appear with higher frequency. The exact distribution is defined via the *microfacet normal distribution function* $D(\mathbf{m})$. Unlike $F()$, the value of $D()$ is not restricted to lie between 0 and 1—although values must be non-negative, they can be arbitrarily large. Also unlike $F()$, the function $D()$ is not spectral or RGB-valued, but scalar-valued. In microfacet BRDF terms, $D()$ is evaluated for the direction $\mathbf{h}$, to help determine the concentration of active microfacets (those for which $\mathbf{m} = \mathbf{h}$). This is why the normal distribution term appears in Equation 4 as $D(\mathbf{h})$.

The function $D()$ determines the size, brightness, and shape of the specular highlight. Several different normal distribution functions appear in the graphics literature, all are somewhat Gaussian-like, with some kind of "roughness" or variance parameter (anisotropic functions typically have two variance parameters). As the surface roughness decreases, the concentration of the microfacet normals $\mathbf{m}$ around the overall surface normal $\mathbf{n}$ increases, and the values of $D(\mathbf{m})$ can become very high (in the limit, for a perfect mirror, the value is infinity at $\mathbf{m} = \mathbf{n}$). Walter et. al. [25] discuss the correct normalization of the distribution function, and give several examples; more examples can be found in other papers [2, 3, 19, 26].

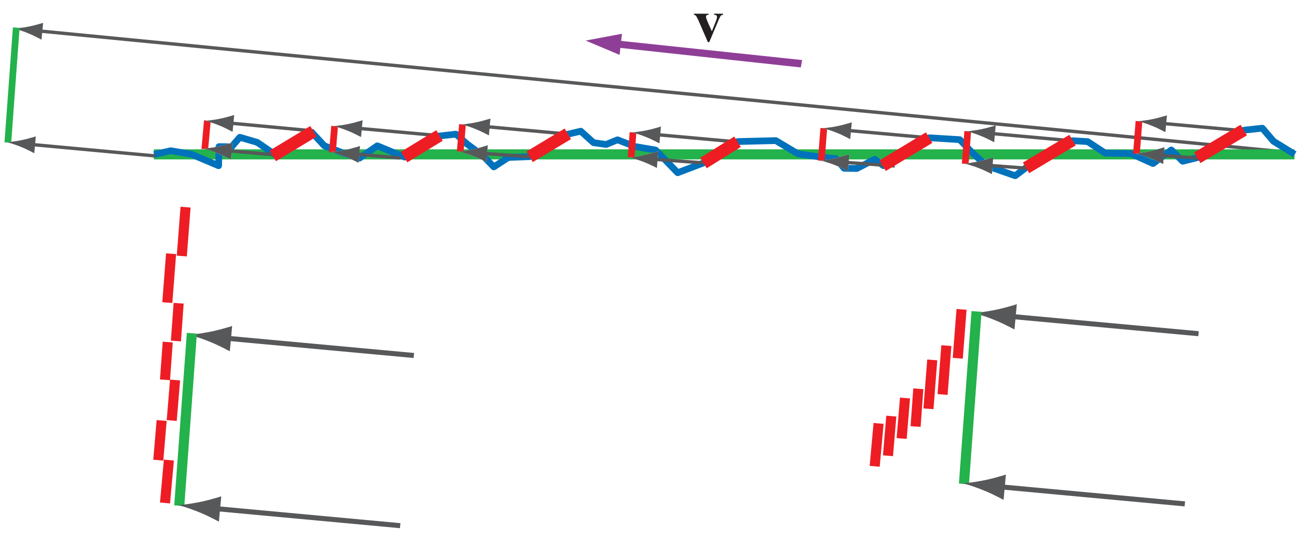


Figure 25: On the top the flat macroscopic surface is shown in green, and the rugged microscopic surface is shown in blue. The facets for which $\mathbf{m} = \mathbf{h}$ are marked in red. The projection of the macroscopic surface area (length in this 2D side illustration) onto the view direction (in other words, its foreshortened surface area) is shown as a green line on the upper left. The projected areas of the individual red microfacets are shown as separate red lines. On the bottom left the areas of the red microfacets are added up without accounting for masking, resulting in an active area greater than the total area. This is illogical, and more importantly can result in the BRDF reflecting more energy than it receives. On the right we see that the red areas are combined in a way that accounts for masking. The overlapping areas are no longer counted multiple times, and we see that the correct active area is smaller than the total area. When the viewing angle is lower, then the effect will be even more pronounced—ignoring the effects of masking could lead to the BRDF reflecting thousands of times the amount of energy received or more (the amount of reflected energy would go to infinity in the limit as the angle goes to $90°$).

Shadowing-Masking Term

The shadowing and masking term $G(\mathbf{l}, \mathbf{v}, \mathbf{h})$ is also often called the *geometry term* in the BRDF literature. The function $G(\mathbf{l}, \mathbf{v}, \mathbf{m})$ represents the probability that microfacets with a given normal $\mathbf{m}$ will be visible from both the light direction $\mathbf{l}$ and the view direction $\mathbf{v}$. In the microfacet BRDF, $\mathbf{m}$ is replaced with $\mathbf{h}$ (for similar reasons as in the previous two terms). Since the function $G()$ represents a probability, its values are scalars and are constrained to lie between 0 and 1. As in the case of $D()$, there are various analytical expressions for $G()$ in the literature [2, 3, 7, 8, 17, 19, 25]; these are typically approximations based on some simplified model of the surface. The $G()$ function typically does not introduce any new parameters to the BRDF; it either has no parameters, or uses the roughness parameters of the $D()$ function. In many cases, the shadowing-masking term partially cancels out the $(\mathbf{n}\cdot\mathbf{l})(\mathbf{n}\cdot\mathbf{v})$ denominator in Equation 4, replacing it with some other expression such as $\max(\mathbf{n}\cdot\mathbf{l}, \mathbf{n}\cdot\mathbf{v})$.

The shadowing-masking term is essential for BRDF energy conservation—without such a term the BRDF can reflect arbitrarily more light energy than it receives. A key part of the microfacet BRDF is the ratio between the active area (combined area of the microfacets which reflect light energy from $\mathbf{l}$ to $\mathbf{v}$) and the total area (of the macroscopic surface). If shadowing and masking are not accounted for, then the active area may exceed the total area, an obvious impossibility which can lead to the BRDF not conserving energy, in some cases by a huge amount (see Figure 25).

Microfacet Models

The choice of $D()$ and $G()$ functions is independent; they can be mixed and matched from different microfacet models. Most papers proposing a new microfacet BRDF model are best understood as introducing a new $D()$ and / or $G()$ function.

Once the $D()$ and $G()$ functions have been chosen, the full BRDF is determined by selecting parameter values. Microfacet BRDFs have compact parameterizations, typically only consisting of one RGB value for ($\mathbf{c}_{\text{spec}}$) and one scalar for roughness (two in the case of anisotropic BRDFs).

Subsurface Reflectance (Diffuse Term)

Although there are several models for subsurface local reflection in the literature, the most widely-used one by far is the *Lambertian* BRDF term. The Lambertian BRDF is actually a constant value; the well-known cosine or $(\mathbf{n} \cdot \mathbf{l})$ factor is part of the reflection equation, not the BRDF (as we saw in Equation 1). The exact value of the Lambertian BRDF is:

$$f_{\text{Lambert}}(\mathbf{l}, \mathbf{v}) = \frac{\mathbf{c}_{\text{diff}}}{\pi}. \tag{7}$$

Here $\mathbf{c}_{\text{diff}}$ is the fraction of light which is diffusely reflected. As in the case of $\mathbf{c}_{\text{spec}}$, it is an RGB value with R, G, and B restricted to the $0 - 1$ range, and corresponds closely to what most people think of as a "surface color". This parameter is typically referred to as the *diffuse color*.

Non-Lambertian diffuse terms attempt to model either the trade-off between specular and diffuse terms at glancing angles [2, 3, 17, 24], or the effects of surface roughness at a scale larger than the scattering distance [21].

Implementing Shading

In the previous section, we saw the mathematical models that are typically employed to describe surface shading. In this section, we will discuss how such models are implemented in film and game production renderers.

General Lighting

In the most general case, the BRDF must be integrated against incoming light from all different directions. This includes not only primary light sources (with area) but also skylight and accurate reflections of other objects in the scene. To fully solve this, global illumination algorithms are required. Detailed descriptions of these algorithms are outside the domain of this talk; more details can be found in various references ([18, 11]), as well as Adam Martinez's talk in this course, *Faster Photorealism in Wonderland: Physically-Based Shading and Lighting at Sony Pictures Imageworks*.

Punctual Light Sources

A far more restricted, but common production lighting environment is comprised of one or more *punctual light sources*. These are the classic computer graphics point, directional, and spot lights (more complex variants are also used [4]). Since they are infinitely small and infinitely bright, they aren't physically realizable or realistic, but they do produce reasonable results in many cases and are computationally convenient. Punctual light sources are parameterized by the light color $\mathbf{c}_{\text{light}}$ and the light direction vector $\mathbf{l}_{\mathbf{c}}$. For artist convenience, $\mathbf{c}_{\text{light}}$ does not correspond to a direct radiometric measure of the light's intensity; it is specified as the color a white Lambertian surface would have when illuminated by the light from a direction parallel to the surface normal ($\mathbf{l_c} = \mathbf{n}$). Like other color quantities we have seen, $\mathbf{c}_{\text{light}}$ is spectral (RGB)-valued, but unlike them its range is unbounded.

The primary advantage of punctual light sources is that they greatly simplify the reflection equation (Equation 1), as we will show here. We will start by defining a tiny area light source centered on $\mathbf{l_c}$, with a small angular extent ε. This tiny area light illuminates a shaded surface point with the incoming radiance function $L_{\text{tiny}}(\mathbf{l})$. The incoming radiance function has the following two properties:

$$\forall \mathbf{l} | \angle(\mathbf{l}, \mathbf{l_c}) > \varepsilon, L_{\text{tiny}}(\mathbf{l}) = 0. \tag{8}$$

$$\text{if } \mathbf{l_c} = \mathbf{n}, \text{ then } \mathbf{c}_{\text{light}} = \frac{1}{\pi} \int_{\Omega} L_{\text{tiny}}(\mathbf{l})(\mathbf{n} \cdot \mathbf{l}) d\omega_i. \tag{9}$$

The first property says that no light is incoming for any light directions which form an angle greater than ε with $\mathbf{l_c}$. In other words, the light does not produce any light outside its angular extent of ε. The second property follows from the definition of $\mathbf{c}_{\text{light}}$, applying Equations 1 and 7 with $\mathbf{c}_{\text{diff}} = 1$. Equation 9 still holds in the limit as ε goes to 0:

$$\text{if } \mathbf{l_c} = \mathbf{n}, \text{ then } \mathbf{c}_{\text{light}} = \lim_{\varepsilon \to 0} \left(\frac{1}{\pi} \int_{\Omega} L_{\text{tiny}}(\mathbf{l})(\mathbf{n} \cdot \mathbf{l}) d\omega_i \right). \tag{10}$$

Since $\mathbf{l_c} = \mathbf{n}$ and $\varepsilon \to 0$, we can assume $(\mathbf{n} \cdot \mathbf{l}) = 1$ which gives us:

$$\mathbf{c}_{\text{light}} = \lim_{\varepsilon \to 0} \left(\frac{1}{\pi} \int_{\Omega} L_{\text{tiny}}(\mathbf{l}) d\omega_i \right). \tag{11}$$

Note that Equation 11 is independent of the value of $\mathbf{l_c}$, so it is true for any valid light orientation, not just $\mathbf{l_c} = \mathbf{n}$. Simple rearrangement isolates the value of the integral in the limit:

$$\lim_{\varepsilon \to 0} \left(\int_{\Omega} L_{\text{tiny}}(\mathbf{l}) d\omega_i \right) = \pi \mathbf{c}_{\text{light}}. \tag{12}$$

Now we shall apply our tiny area light to a general BRDF, and look at its behavior in the limit as ε goes to 0:

$$L_o(\mathbf{v}) = \lim_{\varepsilon \to 0} \left(\int_{\Omega} f(\mathbf{l}, \mathbf{v}) \otimes L_{\text{tiny}}(\mathbf{l})(\mathbf{n} \cdot \mathbf{l}) d\omega_i \right) = f(\mathbf{l_c}, \mathbf{v}) \otimes \lim_{\varepsilon \to 0} \left(\int_{\Omega} L_{\text{tiny}}(\mathbf{l}) d\omega_i \right) (\mathbf{n} \cdot \mathbf{l_c}). \tag{13}$$

Substituting Equation 12 into the right part of Equation 13 gives us the final punctual light equation:

$$L_o(\mathbf{v}) = \pi f(\mathbf{l_c}, \mathbf{v}) \otimes \mathbf{c}_{\text{light}} \underline{(\mathbf{n} \cdot \mathbf{l_c})}. \tag{14}$$

Compared to the original reflectance equation, we have replaced the integral with a straightforward evaluation of the BRDF, which is much simpler to compute. Note the line under the dot product $\underline{(\mathbf{n} \cdot \mathbf{l_c})}$; this is our notation for clamping to 0. In other words, $\underline{x} \equiv \max(x, 0)$. The dot product is clamped to handle punctual lights which are behind the surface; these should have no contribution (rather than a negative contribution).

In the case of directional light sources (such as the Sun), both $\mathbf{l_c}$ and $\mathbf{c}_{\text{light}}$ are constant over the scene. In the case of other punctual light types such as point lights and spotlights, both will vary. In reality, $\mathbf{c}_{\text{light}}$ would fall off proportionally to the inverse square distance, but in practice other falloff functions are often used.

If multiple punctual light sources are illuminating the surface, Equation 14 is computed multiple times and the results summed. Punctual light sources are rarely used by themselves, since the lack of any illumination coming from other directions is noticeable, especially with highly specular surfaces. For this reason punctual light sources are typically combined with some kind of ambient or environmental lighting; these types of lighting will be discussed below.

Ambient Lighting

Here we define ambient lighting as some numerical representation of low-frequency lighting, ranging from a single constant light color and intensity over all incoming directions to more complex representations such as spherical harmonics (SH). Often this type of lighting environments is only applied to the diffuse BRDF term; more high-frequency image-based lighting are applied to the specular term. However, it is possible to apply ambient lighting environments to the specular BRDF term. Yoshiharu Gotanda's talk in this course, *Practical Implementation of Physically-Based Shading Models at tri-Ace* gives a specular implementation for constant and SH ambient, a recent presentation by Bungie [5] discusses applying the Cook-Torrance [7, 8] specular term to SH lighting, and a ShaderX^7 article by Schüler [23] describes an implementation of a physically-based specular term with hemispherical lighting.

Image-Based Lighting

Image-based lighting is typically done with environment maps. These maps represent distant lighting. If they are sampled at an appropriate reference position (say, at the center of an object) they can be a very good representation of reflections from distant objects. To correctly handle local shading with a general BRDF and an environment map[5], many samples are required. Importance sampling helps to keep the number of samples to a somewhat more manageable number (at least for film rendering). Another approach that can be used, either by itself (an approximate solution, but suitable for games) or in combination with importance sampling, is environment map prefiltering. More information on importance sampling can be found at another course this year [6], as well as as Adam Martinez's talk in this course. Other aspects of image-based lighting in film production are discussed in Ben Snow's talk in this course, *Terminators and Iron Men: Image-Based Lighting and Physical Shading at ILM*, and some aspects of shading with environment maps for video games are discussed in Naty Hoffman's other talk, *Crafting Physically Motivated Shading Models for Game Development.*

Further Reading

Chapter 7 of the 3rd edition of "Real-Time Rendering" [20] provides a broad overview of physically-based shading models, going into somewhat more depth than these course notes. For even greater depth, consider reading Glassner's *Principles of Digital Image Synthesis* [12, 13], or *Digital Modeling of Material Appearance* [9] by Dorsey, Rushmeier, and Sillion.

Dutré's free online *Global Illumination Compendium* [10] is a useful reference for BRDFs, radiometric math, and much else.

Acknowledgments

The author would like to thank A K Peters for permission to use images from the book *Real-Time Rendering, 3rd edition*, and also Paul Edelstein, Yoshiharu Gotanda and Dimitar Lazarov for many thought-provoking discussions on physically-based shading models.

[5]global illumination effects such as interreflections on an object can also be handled, but in that case you are effectively using the environment map as a light source for a global illumination renderer.

Bibliography

[1] Ashikhmin, Michael, Simon Premože, and Peter Shirley, "A Microfacet-Based BRDF Generator," *Computer Graphics (SIGGRAPH 2000 Proceedings)*, pp. 67–74, July 2000. http://www.cs.utah.edu/~shirley/papers/facets.pdf Cited on p. 13

[2] Ashikhmin, Michael, and Peter Shirley, "An Anisotropic Phong Light Reflection Model," Technical Report UUCS-00-014, Computer Science Department, University of Utah, June 2000. www.cs.utah.edu/research/techreports/2000/pdf/UUCS-00-014.pdf Cited on p. 15, 16, 17

[3] Ashikhmin, Michael, Simon Premože, and Peter Shirley, "An Anisotropic Phong BRDF Model," *journal of graphics tools*, vol. 5, no. 2, pp. 25–32, 2000. www.cs.utah.edu/~shirley/papers/jgtbrdf.pdf Cited on p. 15, 16, 17

[4] Barzel, Ronen, "Lighting Controls for Computer Cinematography" *journal of graphics tools*, vol. 2, no. 1, pp. 1–20, 1997. Cited on p. 17

[5] Chen, Hao, "Lighting and Material of Halo 3," *Game Developers Conference*, March 2008. http://www.bungie.net/images/Inside/publications/presentations/lighting_material.zip Cited on p. 19

[6] Colbert, Mark, Simon Premože, and Guillaume François, "Importance Sampling for Production Rendering," *SIGGRAPH 2010 Course Notes*, 2010. http://sites.google.com/site/isrendering/ Cited on p. 19

[7] Cook, Robert L., and Kenneth E. Torrance, "A Reflectance Model for Computer Graphics," *Computer Graphics (SIGGRAPH '81 Proceedings)*, pp. 307–316, July 1981. Cited on p. 16, 19

[8] Cook, Robert L., and Kenneth E. Torrance, "A Reflectance Model for Computer Graphics," *ACM Transactions on Graphics*, vol. 1, no. 1, pp. 7–24, January 1982. http://graphics.pixar.com/library/ReflectanceModel/ Cited on p. 16, 19

[9] Dorsey, Julie, Holly Rushmeier, and François Sillion, *Digital Modeling of Material Appearance*, Morgan Kaufmann, 2007. Cited on p. 19

[10] Dutré, Philip, *Global Illumination Compendium*, 1999. http://www.graphics.cornell.edu/~phil/GI Cited on p. 9, 19

[11] Dutré, Philip, Kavita Bala, and Philippe Bekaert, *Advanced Global Illumination*, second edition, A K Peters Ltd., 2006. Cited on p. 17

[12] Glassner, Andrew S., *Principles of Digital Image Synthesis*, vol. 1, Morgan Kaufmann, 1995. Cited on p. 19

[13] Glassner, Andrew S., *Principles of Digital Image Synthesis*, vol. 2, Morgan Kaufmann, 1995. Cited on p. 19

[14] Gritz, Larry, and Eugene d'Eon, "The Importance of Being Linear," in Hubert Nguyen, ed., *GPU Gems 3*, Addison-Wesley, pp. 529–542, 2007. http://http.developer.nvidia.com/GPUGems3/gpugems3_ch24.html Cited on p. 14

[15] Hoffman, Naty, "Adventures with Gamma-Correct Rendering," Renderwonk blog. http://renderwonk.com/blog/index.php/archive/adventures-with-gamma-correct-rendering/ Cited on p. 14

[16] Kajiya, James T., "The Rendering Equation," *Computer Graphics (SIGGRAPH '86 Proceedings)*, pp. 143–150, August 1986. http://www.cs.brown.edu/courses/cs224/papers/kajiya.pdf Cited on p. 11

[17] Kelemen, Csaba, and Lázló Szirmay-Kalos, "A Microfacet Based Coupled Specular-Matte BRDF Model with Importance Sampling," *Eurographics 2001*, short presentation, pp. 25–34, September 2001. `http://www.fsz.bme.hu/~szirmay/scook_link.htm` Cited on p. 16, 17

[18] Křivánek, Jaroslav, Marcos Fajardo, Per H. Christensen, Eric Tabellion, Michael Bunnell, David Larsson, and Anton Kaplanyan, "Global Illumination Across Industries," *SIGGRAPH 2010 Course Notes*, 2010. `http://www.graphics.cornell.edu/~jaroslav/gicourse2010/` Cited on p. 17

[19] Kurt, Murat, László Szirmay-Kalos, and Jaroslav Křivánek, "An Anisotropic BRDF Model for Fitting and Monte Carlo Rendering," *Computer Graphics*, vol. 44, no. 1, pp. 1–15, 2010. `http://www.graphics.cornell.edu/~jaroslav/` Cited on p. 15, 16

[20] Akenine-Möller, Tomas, Eric Haines, and Naty Hoffman, *Real-Time Rendering*, third edition, A K Peters Ltd., 2008. `http://realtimerendering.com/` Cited on p. 9, 19

[21] Oren, Michael, and Shree K. Nayar, "Generalization of Lambert's Reflectance Model," *Computer Graphics (SIGGRAPH 94 Proceedings)*, pp. 239–246, July 1994. `http://www.cs.columbia.edu/CAVE/projects/oren/` Cited on p. 17

[22] Schlick, Christophe, "An Inexpensive BDRF Model for Physically based Rendering," *Computer Graphics Forum*, vol. 13, no. 3, Sept. 1994, pp. 149–162. `http://dept-info.labri.u-bordeaux.fr/~schlick/DOC/eur2.html` Cited on p. 14

[23] Schüler, Christian, "An Efficient and Physically Plausible Real Time Shading Model," in Wolfgang Engel, ed., *ShaderX*^{7}, Charles River Media, pp. 175–187, 2009. Cited on p. 19

[24] Shirley, Peter, Helen Hu, Brian Smits, Eric Lafortune, "A Practitioners' Assessment of Light Reflection Models," *Pacific Graphics '97*, pp. 40–49, October 1997. `http://www.graphics.cornell.edu/pubs/1997/SHSL97.html` Cited on p. 17

[25] Walter, Bruce, Stephen R. Marschner, Hongsong Li, Kenneth E. Torrance, "Microfacet Models for Refraction through Rough Surfaces," *Eurographics Symposium on Rendering (2007)*, 195–206, June 2007. `http://www.cs.cornell.edu/~srm/publications/EGSR07-btdf.html` Cited on p. 10, 13, 15, 16

[26] Ward, Gregory, "Measuring and Modeling Anisotropic Reflection," *Computer Graphics (SIGGRAPH '92 Proceedings)*, pp. 265–272, July 1992. `http://radsite.lbl.gov/radiance/papers/sg92/paper.html` Cited on p. 15

Physically Based Shading Models in Film and Game Production:

Practical Implementation at tri-Ace

Yoshiharu Gotanda
tri-Ace, Inc.

1. Introduction

In this paper, we present our practical examples of physically based shading models that we implemented. In the game industry, traditional ad-hoc shading models are mainly used because of performance, though recently, physically based models have been attracting more attention. In our studio, artists had some difficulty setting parameters for physically correct materials using the following ad hoc shading model:

$$\sigma = R_d(N \cdot L) + R_s F(f_0)(N \cdot L)(N \cdot H)^n G \,, \tag{1}$$

where R_d is the diffuse color, R_s is the specular color, N is the normal vector, L is the light vector, H is the halfway vector, $F(f_0)$ is the Fresnel function, and n is the cosine power that is often called shininess which represents the roughness of a surface. Lastly, G is the geometry attenuation factor. Some of the above parameters are typically stored in textures.

The human eye doesn't perceive light linearly and the brain recognizes the brightness of light in non-linear space (similar to logarithmic space) with a high dynamic range. On the other hand, display devices used for video games typically have an 8-bit color resolution which is insufficient to represent real world, high dynamic range, light intensity. Due to these two reasons, artists can set material parameters incorrectly via their intuition, even though the real world material properties have more dynamic variance. For example, if there is an object that has a 1,000 times stronger specular than another object, an artist may set only a 10 times stronger specular parameter, because the artist felt that was the correct value.

In our case, this problem often happened by the Fresnel parameter being incorrectly set. For our implementation, we used Schlick's approximation[1] as shown in Equation 2. The approximation is faster than the original equation and produces a good shading result.

$$F(f_0) = f_0 + (1 - f_0)(1 - E \cdot H)^5 \,. \tag{2}$$

f_0 is the normal specular reflectance. This can be computed with:

$$f_0 = \left(\frac{1-n}{1+n} \right)^2 , \tag{3}$$

where n is the complex refractive index[1]. For typical dielectric materials, n is between 1.3 and 1.7. Using the average value of 1.5, f_0 evaluates to 0.04 and Equation 2 evaluates to 0.04 in the normal direction and 1.0 in the direction perpendicular to the normal. As a result, the reflection ratio between the normal and glancing directions becomes 25. Since such a large specular variance is not intuitively acceptable for artists, they incorrectly set a value like 0.3 or 0.5 to f_0. This causes the specular in the normal direction to become too

[1] If n is a complex number, it is important that this equation be calculated using n as a complex number.

strong compared to reality. The artist then senses something isn't right and reduces specular intensity to try and compensate. Consequently, this leads to specular at glancing angles becoming too weak. As a typical example, because the specular at glancing angles is weak, the highlight on the edge of a back lit object cannot be accurately represented. As a solution, we changed parameters in our tool to use either complex refractive indices, n, or prebuilt material templates such that artists do not have the opportunity to provide incorrect inputs for the Fresnel equation.

Compared with these issues, physically based shading models allow us to easily manipulate the parameters. We can reduce the number of parameters or textures and the shader still produces physically correct results.

2. Customized Blinn-Phong model

The following equation is our BRDF model based on the Blinn-Phong model[2]:

$$\rho = \frac{R_d}{\pi}\left((1 - F_{diff}(f_0)\right) + \frac{(n+2)}{4\pi(2 - 2^{-\frac{n}{2}})} \cdot \frac{F_{spec}(f_0)(N \cdot H)^n}{\max(N \cdot L, N \cdot E)}. \tag{4}$$

$F_{diff}(f_0)$ and $F_{spec}(f_0)$ are Fresnel functions. We used Schlick's approximation for them:

$$F_{diff}(f_0) = f_0 + (1 - f_0)(1 - N \cdot L)^5, \tag{5}$$

$$F_{spec}(f_0) = f_0 + (1 - f_0)(1 - E \cdot H)^5. \tag{6}$$

The BRDF shown in Equation 4 basically follows the laws of energy conservation. However, the function violates reciprocity in the diffuse term for performance.

The normalization factor $(n+2)/4\pi(2 - 2^{-\frac{n}{2}})$ is calculated with our specular BRDF with Neumann-Neumann BRDF[3] as:

$$f_{r\_spec} = \frac{(H \cdot N)^n}{\max(N \cdot E, N \cdot L)}. \tag{7}$$

In order to acquire the normalization factor, this BRDF with the cosine term is integrated and must satisfy the laws of energy conservation as follows:

$$c \cdot \int_\Omega \frac{(H \cdot N)^n (N \cdot L)}{\max(N \cdot E, N \cdot L)} d\omega \le 1 . \tag{8}$$

Choosing c in this inequality to satisfy the energy conservation requirements, the normalization factor can be computed. Since integrating the integral analytically over all light (L) directions is impossible, we assume that the maximum reflected energy occurs when $L = N$. Therefore the inequality becomes:

$$c \cdot \int_\Omega \frac{(H \cdot N)^n}{\max(N \cdot E, 1)} d\omega \le 1. \tag{9}$$

However, $N \cdot E$ is always less than or equal to 1 and the inequality can be simplified to:

$$c \cdot \int_{\Omega} (H \cdot N)^n d\omega \leq 1 . \tag{10}$$

Then the reciprocal of c becomes:

$$\begin{aligned} &\int_{\Omega} (H \cdot N)^n d\omega \\ &= \int_{-\pi}^{\pi} \int_{0}^{\frac{\pi}{2}} (\cos\frac{\theta}{2})^n \sin\theta d\theta d\varphi \\ &= \frac{4\pi(2 - 2^{-\frac{n}{2}})}{n+2} . \end{aligned} \tag{11}$$

This normalization factor is a little expensive to compute in real-time, therefore we approximate it with a linear function. The coefficients of the linear function are determined using the least square method by fitting them in the range of n = 0 to 1,000:

$$\frac{n+2}{4\pi(2 - 2^{-\frac{n}{2}})} \approx 0.0397436n + 0.0856832 . \tag{12}$$

Figure 1 shows the difference between the original normalization factor and the approximated one.

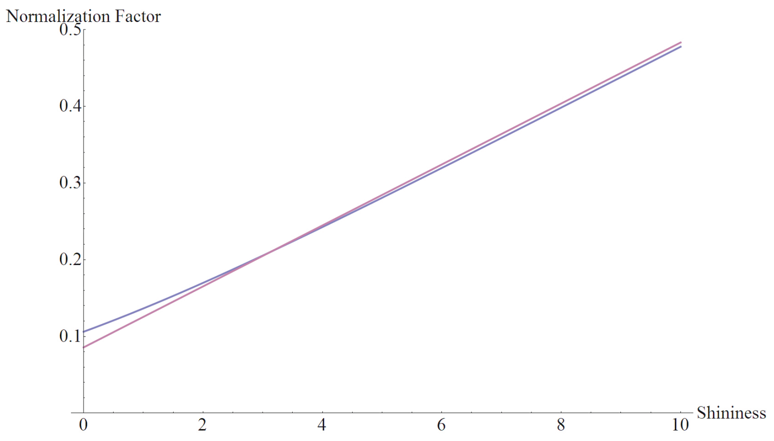


Figure 1: *Normalization Factor Comparison. The blue line is the original factor and the red line is the approximated factor with the linear function. The graph only shows the factor between shininess of 0 and 10. The error between shininess of 0 and 1,000 is almost negligible.*

The diffuse term in our BRDF model is an approximation of a more physically correct model. First we assume that the diffuse component is a perfect lambertian[2]. Then, the incident light reflects as specular and diffuse reflections at the shading point. (In this case, we don't think of other types of reflections.) The amount of specular reflection is decided by the Fresnel equation with respect to the reflection angle. Due to energy conservation, the diffuse component can be computed as the rest of specular reflection. The reflection angle is equivalent to the incident angle, so the Fresnel equation in the diffuse term is calculated with respect to the incident angle.

2 This is obviously a wrong assumption because it violates the reciprocity. It is not an assumption from physics, from simplicity for performance.

However, the diffuse term is a little expensive and approximating it only produces subtle differences. Therefore, the diffuse term is also approximated in our final implementation as:

$$\rho = \frac{R_d}{\pi}(1 - f_0) + (0.0397436n + 0.0856832)\frac{F_{spec}(f_0)(N \cdot H)^n}{\max(N \cdot L, N \cdot E)}. \tag{13}$$

Figure 2 shows rendering results with our customized BRDF.

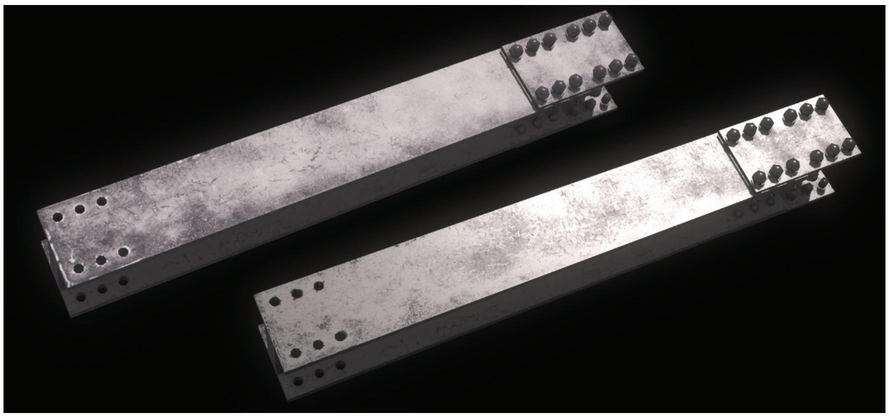


Figure 2 (a): *A comparison only with a shininess map. The left steel frame is rendered with an ad-hoc model and the right one is rendered with our model. The right frame looks more natural compared to the left frame, though it looks a flat material.*

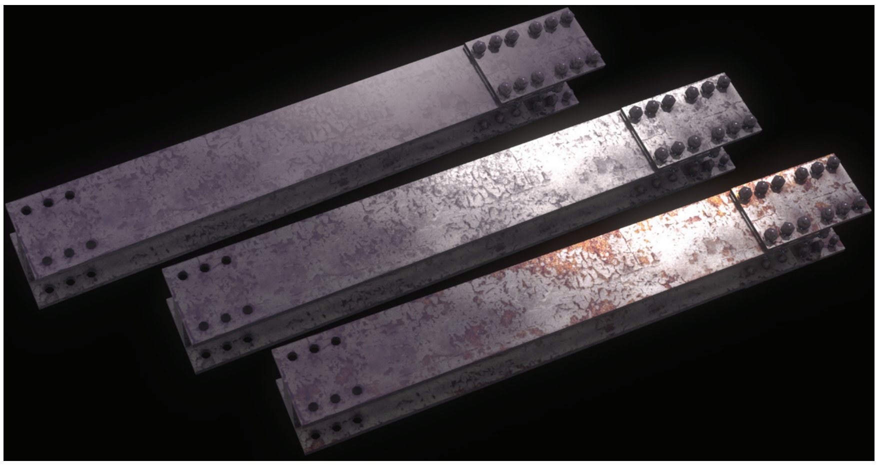


Figure 2 (b): *A comparison only with a reflectance map. The steel frames from the top left are rendered with an ad-hoc model, our model and our spectral model. The texture is used as a specular color map for the ad-hoc model that looks like it is just varying the design of the steel frame. Compared with it, our model looks to have more varied surface materials. Moreover, spectral model represents rust well.*

The BRDF model shown in Equation 13 is isotropic. In order to cover other types of materials, we implemented other variations of models which are anisotropic, spectral and metal versions. Because of the performance reasons, the basic model only covers isotropic, monochrome (non-spectral) and dielectric materials. The anisotropic model supports two different shininess parameters along with the tangent vector and binormal vector. The spectral model supports three different f_0 parameters, especially for materials which have different refractive indices at different wavelengths. The metal model[4] is designed for materials which have a complex refractive index especially with a large imaginary number. The Fresnel function for these kinds of metals has a distinctive curve compared to other materials. Figure 3 shows rendering results of these variation shaders.

In addition to the BRDF model based on the Blinn-Phong model, we also implemented other BRDF models such as Ashikhmin-Shirley[5] model, Marschner[6] model and Kajiya-Kay[7] model. Figure 4 shows rendering results of these shaders and comparisons with our Blinn-Phong based shaders.

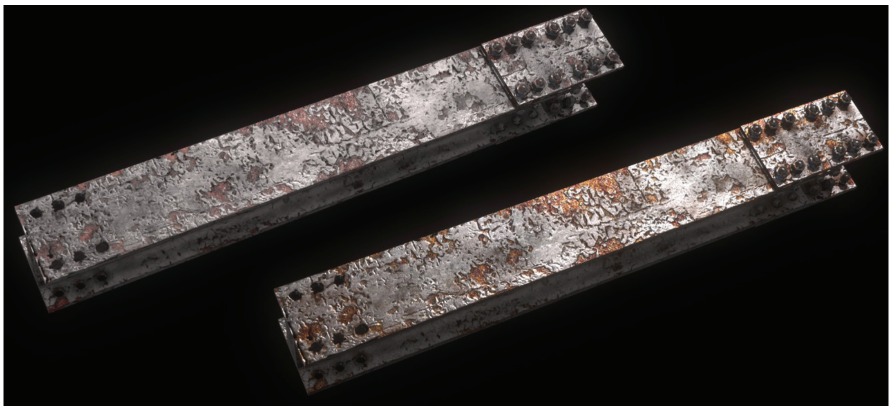


Figure 3(a): *The left steel frame is rendered with an ad-hoc model and the right one is rendered with our spectral model. Both frames have all textures such as reflectance, shininess maps, etc. With respect to the steel part and rusty part of both steel frames, it's easier to notice the difference between the materials on the right one.*

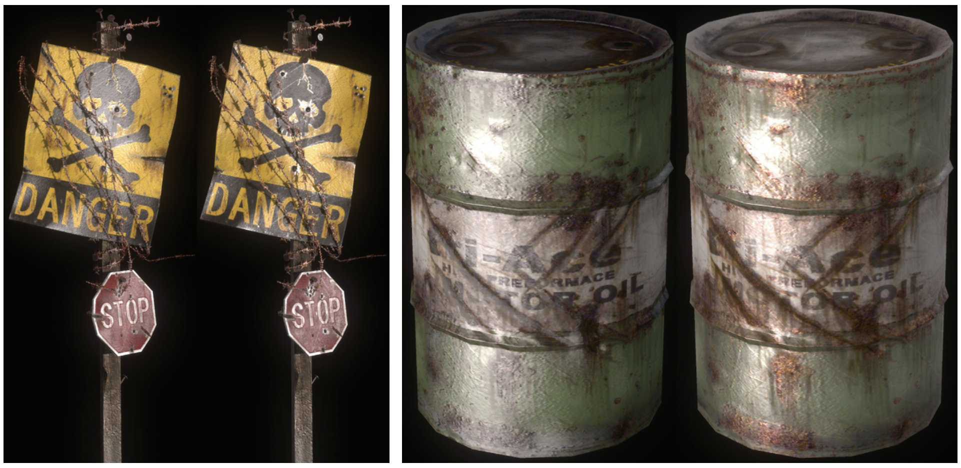


Figure 3 (b): *From the left: (a) ad-hoc model vs. our model, (b) ad-hoc model vs. spectral model. The left object uses an ad-hoc model and the right object uses our model in both images. In the image (a), reflectance is changed with respect to the black and yellow pigments in the sign. However, with the ad-hoc model, the black and yellow pigments look like they use the same material. Different materials are rendered well for the drum on the right using a reflectance map.*

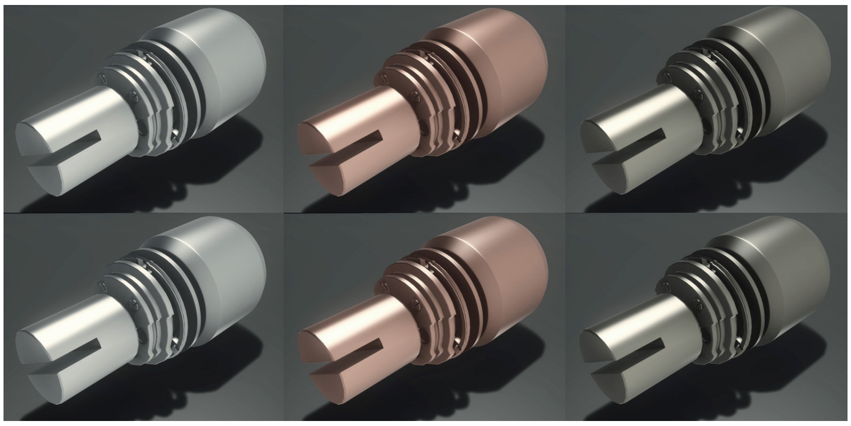


***Figure 3 (c)**: Comparisons of our metal model and spectral model. The top bolts are rendered with our spectral model and the bottom bolts are rendered with our metal models. Materials from the left are aluminum, copper and titanium. Specular reflection looks slightly different.*

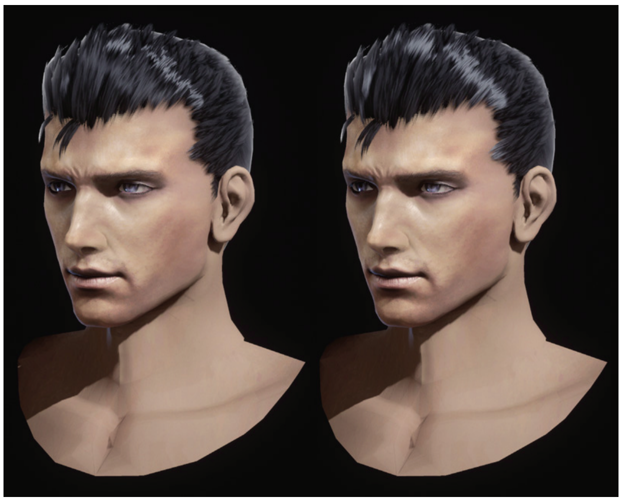


***Figure 4 (a)**: The hair of the left character is rendered with Marschner model and the hair of the right character is rendered with Kajiya-Kay model.*

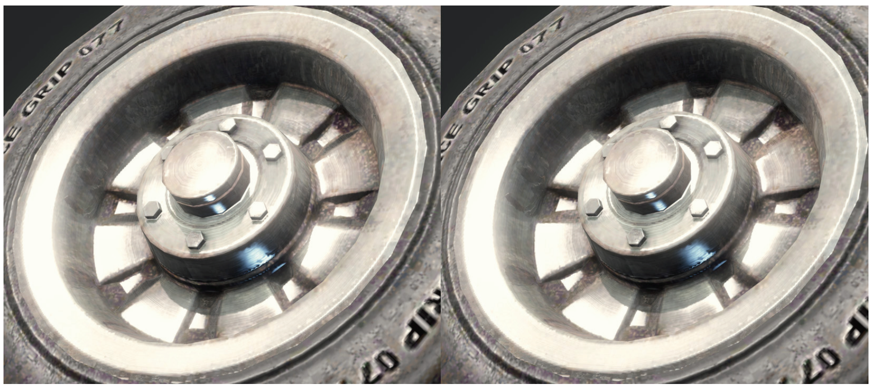


***Figure 4 (b)**: The wheel on the left is rendered with Ashikhmin-Shirley model, the wheel on the right is rendered with our anisotropic model. Our anisotropic model can achieve the close result with a less computational cost than Ashikhmin-Shirley model.*

3. Ambient Shading Improvement

Implementing the physically-based shading models introduced in the previous section, the quality of typical materials can be drastically and easily improved. However, in typical game engines, these BRDF models are used for direct (punctual) light sources such as directional, point and spot lights. Although deferred based shading or lighting techniques are becoming popular, multiple BRDF models don't get along well with deferred based techniques because different BRDF models increase the number of deferred shading passes or require dynamic branches in the deferred shader. Whether using forward shading or deferred shading, ambient lighting is still important for real-time rendering in games. However, because ambient lighting has been invented for performance, ambient lighting typically has a different pass from direct light shading.

There are two standard implementations for ambient shading. One is only treating only the diffuse term with ambient light. With this simplification, objects that only have a specular component such as metals are rendered completely black in a scene that only uses ambient lighting. The second implementation solves the problem of the first by using one arbitrary constant between diffuse and specular terms. In both implementations, the ambient term is computed as a constant and is view independent; though if it existed in the real world, it should be computed as spherical area lighting with a proper BRDF.

As a result of the ambient shading simplification, rendering quality degrades in scenes mainly lit by ambient light such as shadowed areas, the inside of a house lit only by daylight (no artificial lights), cloudy outside areas, etc. Figure 5 shows a sample scene with this problem.

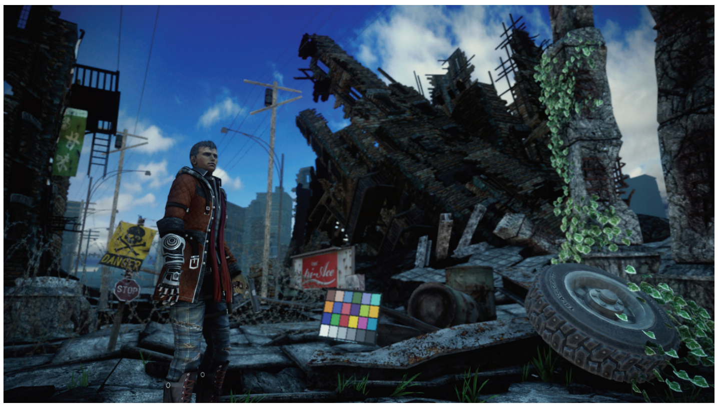


***Figure 5**: A scene with only ambient lighting. Because there are no specular components, material differences are difficult to see.*

On the other hand, ambient lighting has been improved. In the past, ambient light was a constant color vector in the scene and the color vector was same for every pixel and/or vertex. Recently, ambient lighting color has been changed to be dependent on location[3]. Examples would be Hemisphere lighting, Spherical Harmonic (SH) lighting[8][9], and Ambient Occlusion[10]. Hemisphere lighting or SH are one of the most reasonable approximations of spherical area lighting and are widely used for real-time rendering. SH coefficients,

[3] If the calculation is done on the CPU then the object location is used. If done on the GPU then the vertex or pixel location is used.

irradiance[11], or values similar to irradiance are stored in a voxel structure and lighting vectors are interpolated according to shading position. These kinds of techniques are widely used in pseudo real-time global illumination. However, these improvements for ambient lighting only change incoming light according to the shading point. Even if physically correct material parameters are set for every object, they seem to have the same material in a scene dominated by ambient lighting. Figure 6 shows objects with variety of materials using ambient lighting and ambient occlusion.

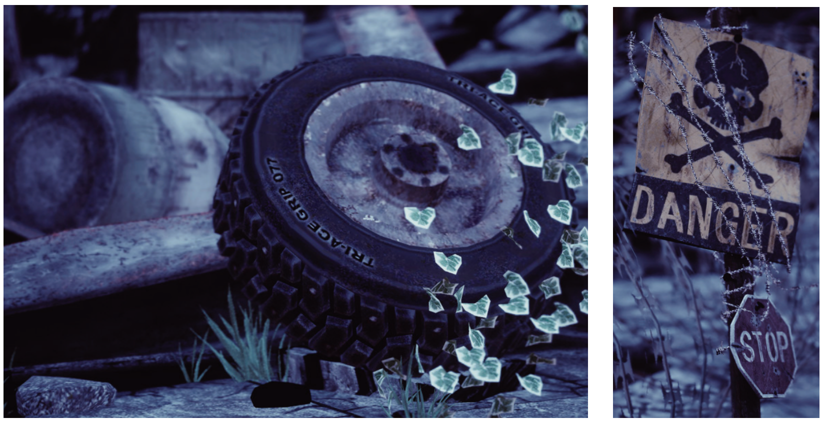


Figure 6: *Screenshots rendered with only ambient lighting. Compared to other screenshots with direct lighting, it is difficult to distinguish material differences. In the real world, though it is also difficult to distinguish.*

Artists noticed this problem from the beginning. In one project, even though they tried painting ambient occlusion or normal mapping textures carefully, quality only marginally improved. Therefore, they solved the problem by placing secondary lights, like rim lights or fill lights, manually. In another project, an artist built an original shader that improved quality using our highly flexible shader system[12] [13]. However, both solutions were based on artists' intuition and were not physically correct.

For solving the previously mentioned problem, we came up with a novel BRDF model called Ambient BRDF. After implementing this model, ambient shading computes the specular and diffuse components properly and the ambient term is no longer a constant. In other words, ambient lighting is regarded as area lighting in our shader. In practice, first we integrated a BRDF integral as:

$$f(n, f_0, \omega) = \int_{\Omega} \rho(n, f_0, \omega, \omega')(N \cdot L) d\omega' , \tag{14}$$

where ρ is a BRDF model, n is shininess, f_0 is normal specular reflectance, ω is the eye vector and ω' is the light vector. In our case, the specular term in Equation 13 is used for computing the specular component of the function ρ. Since this integral is difficult to integrate analytically, we integrated it numerically offline and stored it in a linear (non-swizzled) volume texture. For creating the texture, we developed an application that was firstly used for the experiment of our ambient BRDF models. After that, we tried both a linear texture and swizzled texture, and the linear texture was faster. With our texture, the U coordinate represents $E \cdot N$, the V coordinate represents shininess, and W coordinate is used for f_0. Figure 7 shows the volume texture computed by our implementation. The volume texture stores the specular term itself and is directly used as the specular term in a pixel shader. The diffuse term computation is approximated with $R_d \cdot (1 - \text{specular term})$ but ideally, it should be stored in another texture because due to geometry attenuation, (specular term + diffuse term) does not always equal one. If this approximation is used, the reflectance gets too strong at glancing angles; however, we used it for performance.

***Figure** 7*: *Part of the volume texture computed by our application.*

With the integral, only specular and diffuse terms can be computed. Therefore, the ambient shading result will vary due to material parameters and viewing angle. For a more realistic result, we computed the color terms of ambient light. When using Spherical Harmonics, the diffuse lighting color vector is evaluated with a normal vector and the specular lighting color vector is evaluated with a reflection vector. For both vectors, the color could be approximated depending on the number of SH coefficients. Since the specular component requires a lot of coefficients and no specular cosine lobe is taken into account, the specular color in ambient shading is coarsely approximated. When using image based lighting (environment map), the diffuse lighting color vector is computed with a diffuse cube map or SH coefficients which are calculated from an environment map. The specular color vector is computed with a pre-filtered mipmapped environment map.

For spectral materials, we fetch the texture three times with different f_0. For anisotropic materials, we fetch the texture with the average of two different shininesses. If we implement anisotropic ambient BRDF with texture, we need 4D texture. Therefore, we decouple the 4D texture to two 3D textures. However, compared to the difference between this experiment and the average version, we thought that it was too costly. As a result, we decided to use the current implementation.

Figure 8 shows results of rendering with Ambient BRDF and Figure 9 shows a performance comparison.

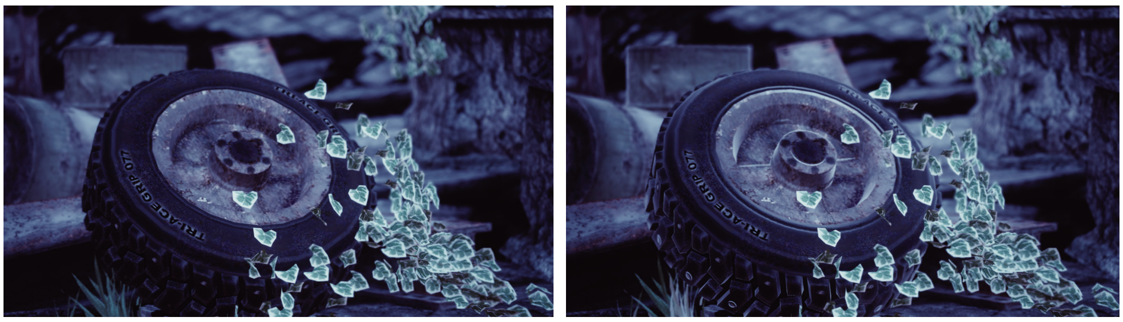
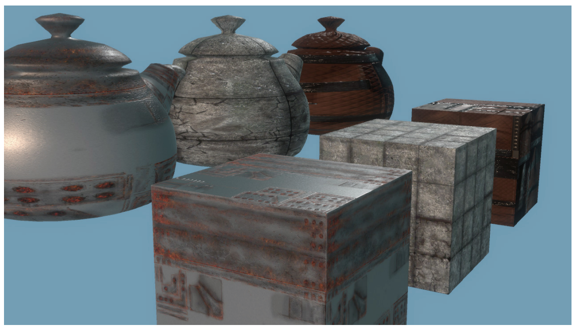


***Figure** 8*: *The image on the left was rendered without Ambient BRDF and the image on the right was rendered with Ambient BRDF. The both images were rendered with only ambient lighting. In the right image, the shading result from the specular component was added on the edges of the tire or wheel. As a result, the right image's material differences are slightly more recognizable than the left image.*



	Ad-hoc model	*Customized model*	*Anisotropic model*	*Spectral model*	*Metal model*	*Ashikhmin*
Ambient BRDF off	3.13	3.71	4.67	4.02	3.48	5.15
Ambient BRDF on	N/A	4.37	5.13	4.61	3.83	5.76

Figure 9*: Performance of different shaders is compared with the scene shown on the image. The numbers indicate rendering times in millisecond on GPU for PlayStation 3. Each object has an albedo map, normal map and combined map (R: Reflectance, G: Shininess). The scene is rendered by 1280x720.*

4. Limitations and future work

Our customized physically-based Blinn-Phong model doesn't handle reciprocity and roughness in the diffuse component. The diffuse roughness component is very important for some materials. Oren-Nayer or other approximation models would be necessary to achieve higher quality results.

Additionally, multi-layered BRDF is not supported. A single layer BRDF is not enough to represent objects with coating, organic materials, and so on. We will need to implement multi-layered BRDF models in real-time.

Our Ambient BRDF model is only an approximation; in the future we can compute more accurate real time representations on more powerful GPUs.

5. Conclusion

With physically based shading models, artists can easily manipulate parameters and textures compared to ad-hoc models. Physically based shading models are fast enough to run in real-time, allow us to render realistic images, and give us true HDR images. The Ambient BRDF model automatically improves image quality in a scene dominated by ambient lighting instead of using ad-hoc methods such as placing secondary lights. As a result secondary lights like rim lights or fill lights can be used for their original purposes. Lastly, Figure 10 shows rendering results.

Acknowledgements

The author would like to thank to Tatsuya Shoji for helping with the research and implementation of physically based shading models, Bart Sekura and Elliott Davis for reviewing the paper and slides, Kenichi Kanekura, Kazuki Shigeta, Kenichi Kaneko and Ryo Mizukami for creating beautiful graphics samples, and my fellow course speakers, especially Nathaniel Hoffman for reviewing.

References

[1] Christophe Schlick. An Inexpensive BRDF Model for Physically-based Rendering. *Computer Graphics Forum,* 13(3):233-246, 1994.

[2] James F. Blinn. Models of Light Reflection for Computer Synthesized Pictures. *Proceedings of the 4th annual conference on Computer graphics and interactive techniques*, 1977.

[3] László Neumann, Attila Neumann, László Szirmay-Kalos. Compact Metallic Reflectance Models. *Computer Graphics Forum*, 1999

[4] István Lazányi, László Szirmay-Kalos. Fresnel Term Approximation for Metals. In *Short Paper Proceedings of WSCG*, pp.77-80, 2005.

[5] Michael Ashikhmin and Peter Shirley. An Anisotropic Phong BRDF Model. *UUCS-00-014*, 2000.

[6] Stephen R. Marschner, Henrik Wann Jensen, Mike Cammarano, Steve Worley and Pat Hanrahan. Light Scattering from Human Hair Fibers. *ACM Transactions on Graphics*, 22, 3(July), 281-290, 2003.

[7] James T. Kajiya. Anisotropic Reflection Models. In *Proceedings of SIGGRAPH 85*, ACM Press, 15–21, 1985.

[8] Robin Green. Spherical Harmonic Lighting: The Gritty Details. *http://www.research.scea.com/gdc2003/spherical-harmonic-lighting.html*, 2003.

[9] Peter-Pike Sloan, Jan Kautz, John Snyder. Precomputed Radiance Transfer for Real-Time Rendering in Dynamics, Low-Frequency Lighting Environments. *ACM Transactions on Graphics* 21, 3, 527–536, 2002.

[10] S. Zhukov, A. Iones, G.Kronin. An ambient light illumination model. In *Rendering Techniques '98 (Proceedings of the Eurographics Workshop on Rendering)*, 45–55, 1998.

[11] Gene Greger, Peter Shirley, Philip M. Hubbard, Donald P. Greenberg. The Irradiance Volume. *IEEE Compute Graphics & Applications*, 1998

[12] Yoshiharu Gotanda, Tatsuya Shoji. Shader Kanrijirei – Jiyudoto Hikikaeni – (Postmortem of Shader Management). In *CESA Developers Conference*, 2008.

[13] Yoshiharu Gotanda. STAR OCEAN 4: Flexible Shader Management and Post-Processing. In *Game Developers Conference*, 2009.

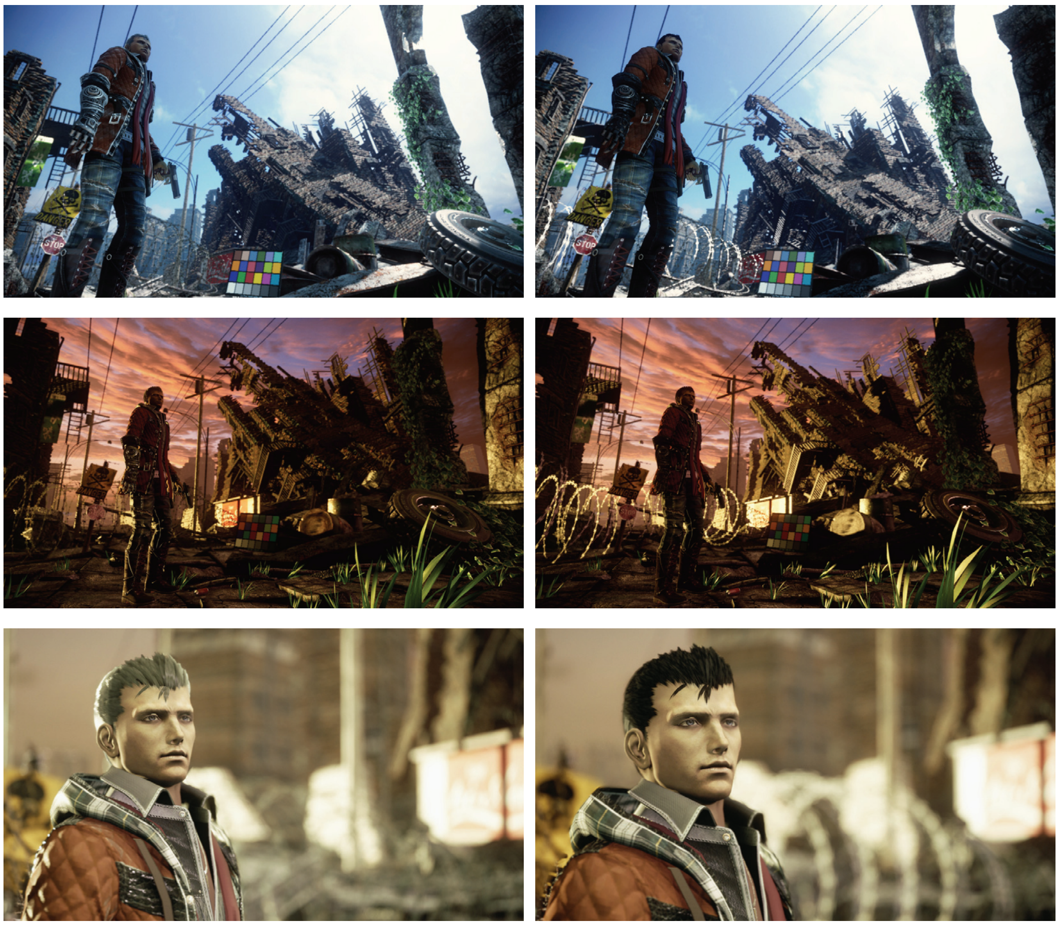


Figure 10: *The images on the left are rendered with our physically based shading models and the other images on the right are rendered with ad-hoc models. Due to our artists' good work, the right images still look very good. The left images keep consistent material appearance with all kinds of lighting such as daylight, sunset, or under shadow.*

Appendix

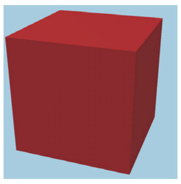
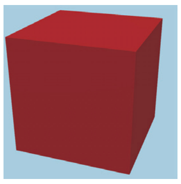
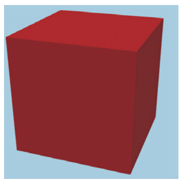
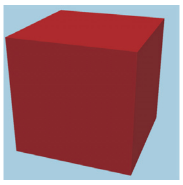
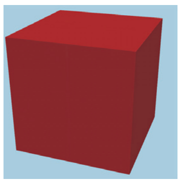
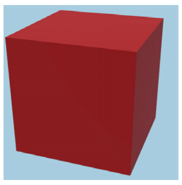
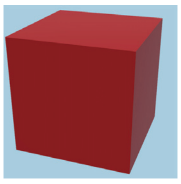
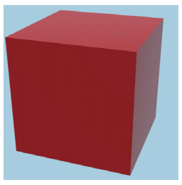
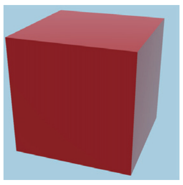
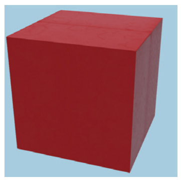
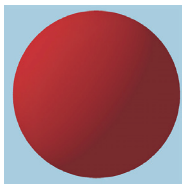
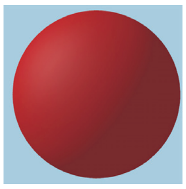
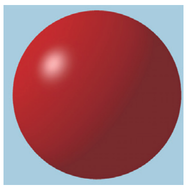
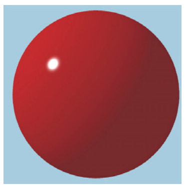
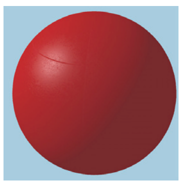
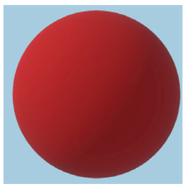
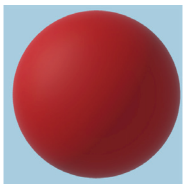
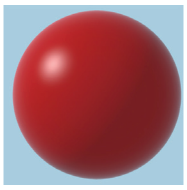
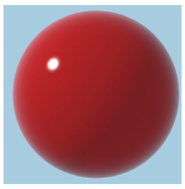
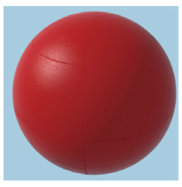
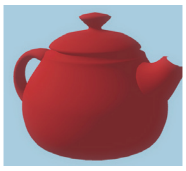
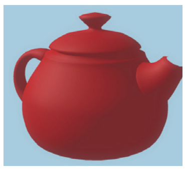
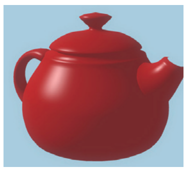
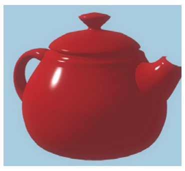
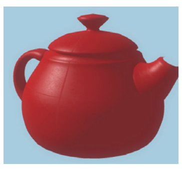
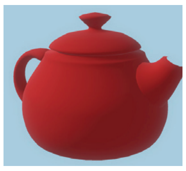
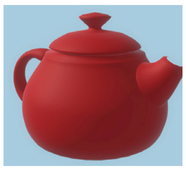
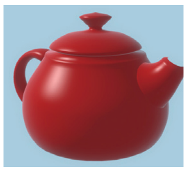
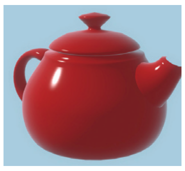
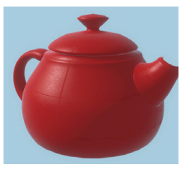
	Shininess 1	Shininess 10	Shininess 100	Shininess 1000	Shininess Map
Box (AB OFF)					
Box (AB ON)					
Sphere (AB OFF)					
Sphere (AB ON)					
Teapot (AB OFF)					
Teapot (AB ON)					

Table 1: *Ambient BRDF comparison with different settings. Each column shows different shininess settings. Each row shows different objects and Ambient BRDF settings (ON or OFF). Images with Ambient BRDF look surrounded by the area light.*

Crafting Physically Motivated Shading Models for Game Development

by Naty Hoffman

In this section of the course notes, we discuss the design of shading models that are both physically based and appropriate for game development use.

Motivation and Infrastructure

Motivation

The first question many game developers ask in connection with physically-based shading models is "Why bother?". This is a valid question, since games do not aim at an exact physical simulation of light transport[1]. However, we shall see many practical advantages for games in adopting these models.

With shading models that are based on physical principles, it is easier to achieve photorealistic and cinematic looks. Objects that use such shading models retain their basic appearance when the lighting and viewing conditions change; they have a robustness that is often not afforded by ad-hoc shading hacks. Art asset creation is also easier; less "slider tweaking" and adjustment of "fudge factors" is needed to achieve high visual quality, and the material interface exposed to the artists is simple, powerful and expressive.

For graphics programmers and shader writers, physically based shaders are easier to troubleshoot. When something appears too bright, too dark, too green, too shiny, etc. then it is much easier to reason about what is happening in the shader when the various terms and parameters have a physical meaning. It is also easier to extend such shaders to add new features, since physical reasoning can be used to determine e.g., which subexpression in the shader should be affected by ambient occlusion, or how an environment map should be combined with existing shading terms.

There have been several articles in the press over the last few years [28, 29, 30] highlighting these advantages in the case of film production; most of the content of these articles applies equally to game development.

Infrastructure

There are several basic features a game rendering engine needs to have to get the most benefit from physically-based shading models. Shading needs to occur in linear space, with inputs and outputs

[1]Rendering applications which do have exact simulation as their goal are called *predictive rendering* applications, since they aim to predict the image that would have been created if the scene and lighting environment existed in the real world. Such applications include architectural previsualization and some types of CAD (Computer-Aided Design).

correctly transformed (*gamma-correct rendering*), the engine should have some support for lighting and shading values with high dynamic range (*HDR*), and there needs to be an appropriate transformation from scene to display colors (*tone mapping*).

Gamma-Correct Rendering

When artists author shader inputs such as textures, light colors, and vertex colors, a *non-linear encoding* is typically used for numerical representation and storage. This means that physical light intensities are not linearly proportional to numerical values. Pixel values stored in the frame buffer after rendering use similar nonlinear encodings.

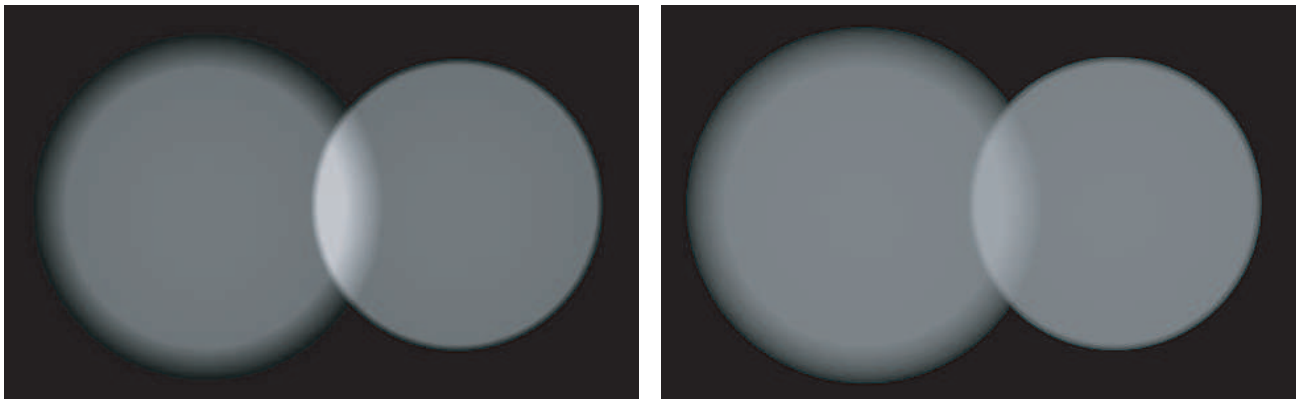


Figure 1: This figure shows a grey flat surface illuminated by two overlapping spotlights. In the left image, shading computations have been performed on nonlinear (sRGB) encoded values, so the addition of the two lights in the overlapping region results in an overly bright region that does not correspond to the expected brightness from summing the two lights. On the right the shading computations are performed on linearly encoded values, and the result appears correct. (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*)

These encodings are primarily used to make efficient use of limited bit precision. Although such nonlinear encodings have steps between successive integer values which are not *physically uniform* (the amount of light energy added at each step varies over the range), they are (somewhat) *perceptually uniform* (the perceived change in brightness at each step does not vary much over the range). This allows for fewer bits to be used without banding.

The two nonlinear encodings most commonly used in game graphics are *sRGB* (used by computer monitors) and *ITU-R Recommendation BT.709* (used by HDTV displays). The official sRGB specification is the IEC 61966-2-1:1999 standard, a working draft of which is available online [18]. Both standards are described in detail elsewhere on the Internet, and in a comprehensive book on video encoding by Charles Poynton [26].

Since shading inputs and outputs use nonlinear encodings, then by default shading computations will be performed on nonlinearly encoded values, which is incorrect and can lead to "$1 + 1 = 3$" situations such as the one shown in Figure 1. To avoid this, shading inputs need to be converted to linear values, and the shader output needs to be converted to the appropriate nonlinear encoding. In principle, these conversions can be done in hardware by the GPU (for textures and render targets) or in a post-process (for shader constants and vertex colors). However, if blending operations are performed in the frame buffer, then difficulties may ensue with platforms like the Playstation 3, which performs alpha blending incorrectly with sRGB render targets. sRGB texture filtering is also not always performed correctly. However, MIP-maps can (and should) always be generated in the correct space. Problems are also caused by the fact that the exact nonlinear encoding varies from platform to platform, especially in the case of consoles.

Although converting a game engine to linear shading generally improves visuals, there are often unintended consequences that need to be addressed. Light distance falloff, Lambert falloff, spotlight angular falloff, soft shadow edge feathering, vertex interpolation all will appear differently now that they are happening in linear space. This may require some retraining or readjustment by the artists, and a few rare cases (like vertex interpolation) might need to be fixed in the shader.

More details on how to convert a game engine to be gamma-correct can be found online [10, 16, 35].

HDR Values

Realistic rendering requires handling much higher intensity values than display maximum white (this maximum is typically mapped to a pixel intensity of 1.0). Values with a much larger range than the display range are referred to as *High Dynamic Range* (HDR) values. HDR values are needed before shading (e.g., lightmaps, environment maps) and shading can also produce such values (e.g., specular highlights). Although these values cannot be displayed directly, they can still affect the final image via effects such as bloom, fog, depth of field and motion blur.

The most straightforward way to handle HDR values is to store them in a wide format with 16 or even 32 bits per channel. However, such wide formats can be prohibitively expensive to use for textures and render targets on current-generation consoles.

One popular solution is to use some type of compressed encoding to store HDR values in low-precision (typically 8 bits per channel) render targets [8, 20]. A second approach is to render multiple (typically two) exposures into different render targets [35]. Unlike compressed encodings, this option has the advantage of hardware blending and filtering support. The cheapest approach is to tone-map to an LDR buffer at the end of the pixel shader. This approach is typically combined with hacks to extract bloom masks out of LDR data. Careful scaling of HDR values to fit in low-precision render targets can improve the results of this approach [19]. Textures such as light maps and environment maps can also be scaled to fit in low-precision texture formats. With careful management of lighting and exposure, ranges greater than $25 - 100$ times display white are rarely needed. In sRGB space this corresponds to just a $4 - 8$ range, which can fit well in 10-bit-per-channel textures (8-bit-per-channel or even DXT in a pinch). Giving artists manual control over the exposure often works better than a more automatic approach.

Tone Mapping

Tone Mapping is the process of converting HDR scene intensity values to display intensities in a perceptually convincing manner. It is common in computer graphics to do with with a smooth curve of some kind [27]. The most effective curves are derived from film emulsion characteristics; these "S-shaped" curves produce pleasing and realistic images. Another term for this mapping "from scene to screen" is *color rendering.* The notes from a recent SIGGRAPH course [17] go into detail on color rendering practice in game and film production; a GDC presentation [11] and subsequent blog posts [12, 13, 14] by John Hable discuss how filmic tone mapping curves were used in *Uncharted 2: Among Thieves.*

Making an Ad-hoc Game Shading Model Physically Plausible

The initial generations of graphics accelerators did not have programmable shaders, and imposed a fixed-function shading model on games for several years. Once programmable shading was introduced, game developers were used to the fixed-function models and often extended them instead of developing new models from scratch. For this reason, many physically incorrect properties of the old fixed-function model persist in common usage today.

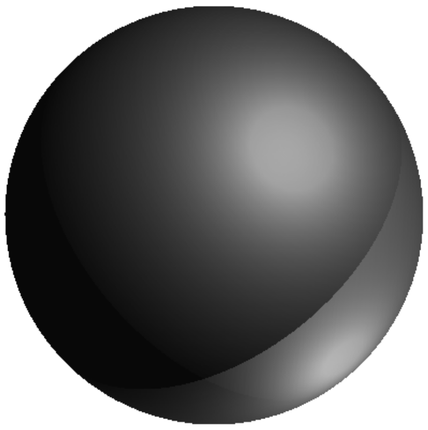


Figure 2: Conditionally setting the specular term to 0 when the light is behind the surface can introduce distracting discontinuities (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

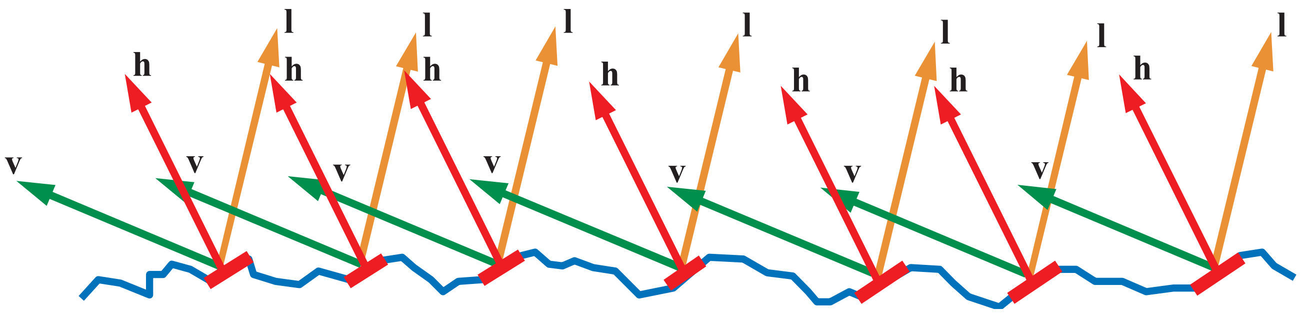


Figure 3: Microfacets with $\mathbf{m} = \mathbf{h}$ are oriented to reflect $\mathbf{l}$ into $\mathbf{v}$—other microfacets do not contribute to the BRDF. (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

We will start with a fairly representative game shading model based on Phong's original model [25]. We will show here the equation for a single punctual light source (note that a game will typically have multiple punctual lights and additional terms for ambient light, environment maps, etc.):

$$L_o(\mathbf{v}) = \left(\mathbf{c}_{\text{diff}} \underline{(\mathbf{n} \cdot \mathbf{l_c})} + \begin{cases} \mathbf{c}_{\text{spec}} \underline{(\mathbf{r_v} \cdot \mathbf{l_c})}^{\alpha_p}, & \text{if } (\mathbf{n} \cdot \mathbf{l_c}) > 0 \\ 0, & \text{otherwise} \end{cases} \right) \otimes \mathbf{c}_{\text{light}}. \quad (1)$$

The notation is the same as in the background talk: $L_o(\mathbf{v})$ is the outgoing radiance in the view direction, $\mathbf{v}$ is the view vector, $\mathbf{c}_{\text{diff}}$ is the diffuse color, $\mathbf{n}$ is the normal vector, $\mathbf{l_c}$ is the punctual light's direction vector, $\mathbf{c}_{\text{spec}}$ is the specular color, α_p is the specular power, $\mathbf{c}_{\text{light}}$ is the punctual light color, $\otimes$ denotes RGB vector multiplication, and the line under the dot product $\underline{(\mathbf{n} \cdot \mathbf{l_c})}$ is the notation for clamping to 0. There is one new vector— $\mathbf{r_v}$ is the view vector reflected about the normal.

Like the clamp on the diffuse dot product, the conditional on the specular term is there to remove the contributions of punctual lights behind the surface. However, this conditional does not make physical sense and worse, can introduce severe artifacts (see Figure 2).

We will modify the shader to avoid specular from backfacing lights in a different way. Instead of a conditional, we will multiply the specular term by $\underline{(\mathbf{n} \cdot \mathbf{l_c})}$. This makes sense since this cosine term is not actually part of the BRDF, but of the rendering equation. Recall the punctual light rendering equation from the background talk in this course:

$$L_o(\mathbf{v}) = \pi f(\mathbf{l_c}, \mathbf{v}) \otimes \mathbf{c}_{\text{light}} \underline{(\mathbf{n} \cdot \mathbf{l_c})}. \tag{2}$$

After replacing the conditional with multiplication by the cosine term, we get the following equation, which is simpler, faster to compute, and does not suffer from discontinuity artifacts:

$$L_o(\mathbf{v}) = \left(\mathbf{c}_{\text{diff}} + \mathbf{c}_{\text{spec}} \underline{(\mathbf{r_v} \cdot \mathbf{l_c})}^{\alpha_p}\right) \otimes \mathbf{c}_{\text{light}} \underline{(\mathbf{n} \cdot \mathbf{l_c})}. \tag{3}$$

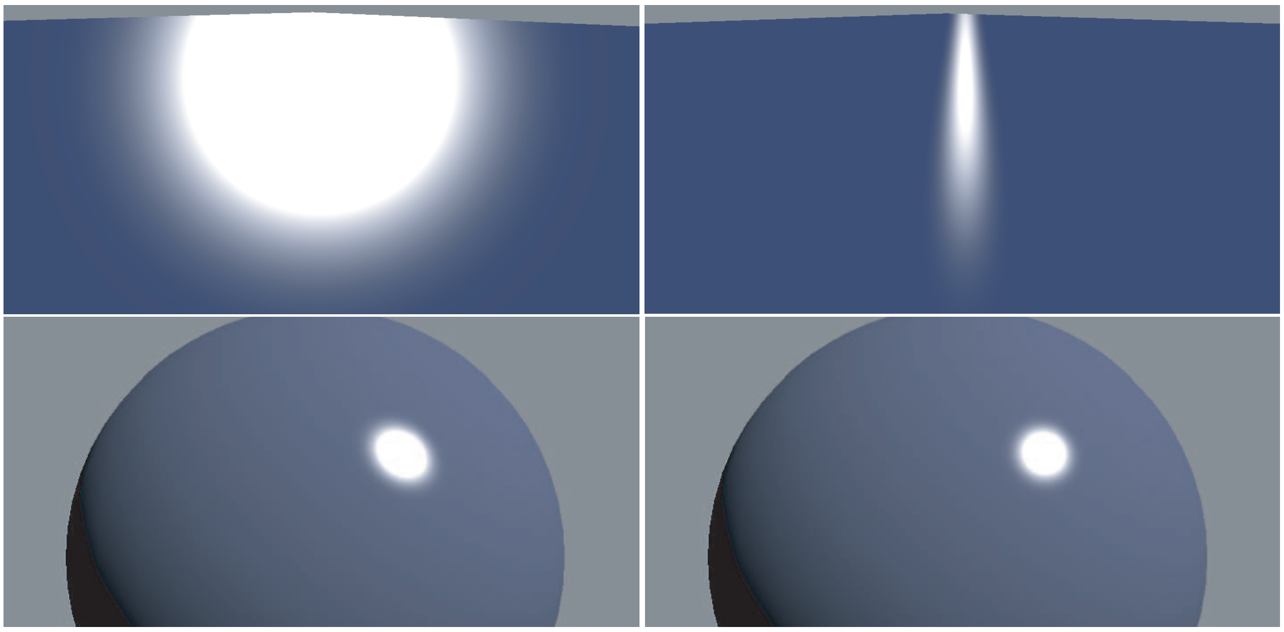


Figure 4: On the left, we see two scenes rendered with the original Phong shading model. On the right, we see the same scenes rendered with the Blinn-Phong model. Although the differences are subtle on the bottom row (sphere), they are very noticeable on the top row (flat plane). (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

Let's now focus on the specular term. What is the physical meaning of the dot product between the reflected view vector and the light? It doesn't seem to correspond to anything from microfacet theory. Blinn's modification [2] to the Phong model (typically referred to as the *Blinn-Phong model*) is very similar to Equation 3, but it uses the more physically meaningful half-vector. Recall (from the background talk): the half-vector is the direction to which the microfacet normals $\mathbf{m}$ need be oriented to reflect $\mathbf{l}$ into $\mathbf{v}$ (see Figure 3)—the reflection vector has no such physical significance. Changing from Phong to Blinn-Phong gives us the following model:

$$L_o(\mathbf{v}) = \underline{(\mathbf{n} \cdot \mathbf{h})}^{\alpha_p} \mathbf{c}_{\text{spec}} \otimes \mathbf{c}_{\text{light}} \underline{(\mathbf{n} \cdot \mathbf{l_c})}. \tag{4}$$

Although Blinn-Phong is more physically meaningful than the original Phong, it is valid to ask whether this makes any practical difference for production shading. Figure 4 compares the visual appearance of the two models. For round objects the two are similar, but for lights glancing off flat surfaces like floors, they are very different. Phong has a round highlight and Blinn-Phong has an elongated thin highlight. If we compare to real-world photographs (Figure 5) then it is clear that Blinn-Phong is much more realistic.

So far using microfacet theory to improve our game shading model has been successful. Let's try some more microfacet theory, starting by comparing our current shading model with a microfacet BRDF model lit by a punctual light source:

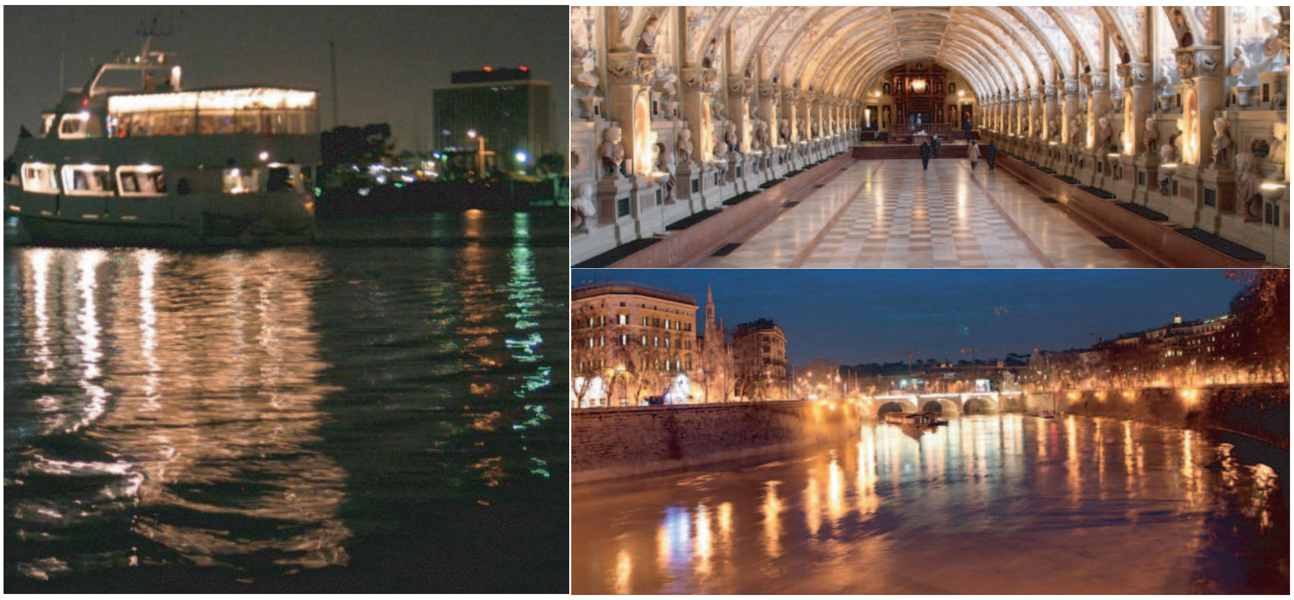


Figure 5: The real world displays elongated thin highlights, similar to those predicted by the Blinn-Phong model and very different from those predicted by the original Phong model. (*photographs from "Real-Time Rendering, 3rd edition" used with permission from A K Peters and the photographer, Elan Ruskin*).

$$L_o(\mathbf{v}) = \pi \frac{\boxed{D(\mathbf{h})}\, G(\mathbf{l_c}, \mathbf{v}, \mathbf{h})}{4\underline{(\mathbf{n} \cdot \mathbf{l_c})}\, \underline{(\mathbf{n} \cdot \mathbf{v})}} \boxed{F(\mathbf{l_c}, \mathbf{h})} \otimes \boxed{\mathbf{c}_{\text{light}} \underline{(\mathbf{n} \cdot \mathbf{l_c})}} \tag{5}$$

$$L_o(\mathbf{v}) = \boxed{\underline{(\mathbf{n} \cdot \mathbf{h})}^{\alpha_p}} \boxed{\mathbf{c}_{\text{spec}}} \otimes \boxed{\mathbf{c}_{\text{light}} \underline{(\mathbf{n} \cdot \mathbf{l_c})}}.$$

There appear to already be several important similarities; we have highlighted the parts that correspond most closely with matching colors. What are the minimal changes required to turn our model into a full-fledged microfacet BRDF?

First, we see that the cosine power term already resembles a microfacet distribution function evaluated with $\mathbf{m} = \mathbf{h}$. However, to convert the cosine power term into a microfacet distribution function it must be correctly normalized. Any microfacet distribution needs to fulfill the requirement that the sum of the microfacet areas is equal to the macrosurface area. More precisely, the sum of the signed projected areas of the microfacets needs to equal the signed projected area of the macroscopic surface; this must hold true for any viewing direction [37]. Mathematically, this means that the function must fulfil this equation for any $\mathbf{v}$:

$$(\mathbf{v} \cdot \mathbf{n}) = \int_{\Theta} D(\mathbf{m})(\mathbf{v} \cdot \mathbf{m}) d\omega_m. \tag{6}$$

Note that the integral is over the entire sphere, not just the hemisphere, and the cosine factors are not clamped. This equation holds for any kind of microsurface, not just heightfields. In the special case, $\mathbf{v} = \mathbf{n}$:

$$1 = \int_{\Theta} D(\mathbf{m})(\mathbf{n} \cdot \mathbf{m}) d\omega_m. \tag{7}$$

The Blinn-Phong cosine power term can be made to obey this equation by multiplying it with a simple normalization factor:

$$D_{\text{BP}}(\mathbf{m}) = \frac{\alpha_p + 2}{2\pi} \underline{(\mathbf{n} \cdot \mathbf{m})}^{\alpha_p}. \tag{8}$$

The next term that needs to be modified is $\mathbf{c}_{\text{spec}}$. As we saw in the background talk, although the specular reflectance of a given material stays almost constant over a wide range of directions, it always goes to 100% white at extremely glancing angles. This effect can be simply modeled by replacing $\mathbf{c}_{\text{spec}}$ with $F_{\text{Schlick}}(\mathbf{c}_{\text{spec}}, \mathbf{l}, \mathbf{h})$ (the background talk course notes give more details on the Schlick approximation). Not all games use a constant $\mathbf{c}_{\text{spec}}$ as in our example "game shading model". Many games do use the Schlick approximation for Fresnel, but unfortunately it is often used incorrectly. The most common error is to use the Schlick equation to interpolate a scalar "Fresnel factor" to 1 instead of interpolating $\mathbf{c}_{\text{spec}}$ to 1. This interpolated "Fresnel factor" is then multiplied with $\mathbf{c}_{\text{spec}}$. This is bad for several reasons. Instead of interpolating the surface specular color to white at the edges, this "Fresnel term" instead darkens it at the center. The artist has to specify the edge color instead of the much more intuitive center color, and in the case of colored specular there is no way to get the correct result. Worse still, the superfluous "Fresnel factor" parameter is added to the ones the artist needs to manipulate, sometimes even stored per-pixel in a texture, wasting storage space. It is true that this "Fresnel model" is slightly cheaper to compute than the correct one, but given the lack of realism and the awkwardness for the artist, the tiny performance difference is not worth it.

Another common error is to use the wrong angle for the Fresnel term. Both environment maps and specular highlights require that the specular color be modified by a Fresnel term, but it is not the same term in both cases. The appropriate angle to use when computing environment map Fresnel is the one between $\mathbf{n}$ and $\mathbf{v}$, while the angle to use for specular highlight Fresnel is the one between $\mathbf{l}$ and $\mathbf{h}$ (or equivalently, between $\mathbf{v}$ and $\mathbf{h}$). This is because specular highlights are reflected by microfacets with surface normals equal to $\mathbf{h}$. Either out of unfamiliarity with the underlying theory or out of temptation to save a few cycles, it is common for developers to use the angle between $\mathbf{n}$ and $\mathbf{v}$ for both environment map Fresnel and specular highlight Fresnel. This temptation should be resisted—when this angle is used for specular highlight Fresnel, any surface which is glancing to the view direction will receive brightened highlights regardless of light direction. This will lead to overly bright highlights throughout the scene, often forcing the use of some hack factor to darken the highlights back down and dooming any chance of achieving realistic specular reflectance.

Looking back at Equation 5, we see that part of the microfacet model has no corresponding term in our modified game specular model. This "orphan term" is the shadowing / masking, or geometry term $G(\mathbf{l_c}, \mathbf{v}, \mathbf{h})$ divided by the "foreshortening factors" $\underline{(\mathbf{n} \cdot \mathbf{l_c})}\,\underline{(\mathbf{n} \cdot \mathbf{v})}$. We refer to this ratio as the *visibility term* since it combines factors accounting for microfacet self-occlusion and foreshortening. Since our modified specular model has no visibility term, we will simply set it to 1. This is the same as setting the geometry term to be equal to the product of the two foreshortening factors, defining the following *implicit geometry term*:

$$G_{\text{implicit}}(\mathbf{l_c}, \mathbf{v}, \mathbf{h}) = \underline{(\mathbf{n} \cdot \mathbf{l_c})}\,\underline{(\mathbf{n} \cdot \mathbf{v})}. \tag{9}$$

This is actually a plausible geometry term for a heightfield microsurface (which is what the Blinn-Phong normal distribution function corresponds to, since it is zero for all backfacing microfacets). $G_{\text{implicit}}()$ is equal to 1 when $\mathbf{l} = \mathbf{n}$ and $\mathbf{v} = \mathbf{n}$, which is correct for a heightfield (no microfacets are occluded from the direction of the macrosurface normal). It goes to 0 for either glancing view angles or glancing light angles, which again is correct (the probability of a microfacet being occluded by other microfacets increases with viewing angle, going to 100% in the limit). Given that this geometry factor actually costs less than zero shader cycles to compute (it cancels out the foreshortening factors so we don't need to divide by them), it has very good "bang per buck".

When comparing $G_{\text{implicit}}()$ to other geometry terms from the literature, we find that it goes to 0 too quickly–it is slightly too dark at moderately glancing angles. In other words, adding a more accurate geometry factor will have the result of somewhat brightening the specular term.

If we plug all these terms (Schlick Fresnel approximation, correctly normalized Blinn-Phong normal distribution function, and implicit geometry term) into the microfacet BRDF in Equation 5, we get

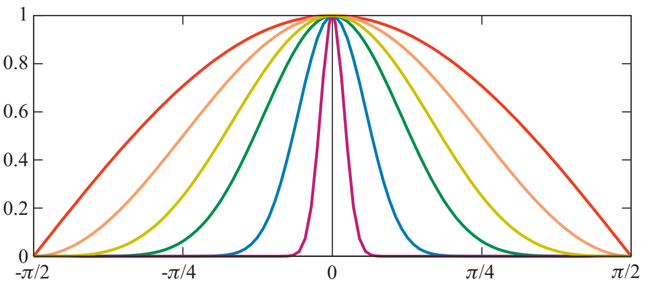


Figure 6: This graph shows the relationship between highlight brightness (y-axis) and angle (x-axis) for a Blinn-Phong term without a normalization factor; the different colors indicate various values of α_p. Note that the center of the highlight is always the same brightness regardless of the value of α_p, which is unrealistic. Furthermore, the overall reflected energy (which can be thought of as roughly corresponding to the volume under the surface created by rotating the curve around the y-axis) decreases as the specular power increases. This is undesirable; the overall reflected energy should only be affected by the parameter $\mathbf{c}_{\text{spec}}$, not α_p. (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

the following shading model:

$$\frac{\alpha_p + 2}{8} \underline{(\mathbf{n} \cdot \mathbf{h})}^{\alpha_p} F_{\text{Schlick}}(\mathbf{c}_{\text{spec}}, \mathbf{l_c}, \mathbf{h}) \otimes \mathbf{c}_{\text{light}} \underline{(\mathbf{n} \cdot \mathbf{l_c})}. \tag{10}$$

Besides the Fresnel term, the only difference between this model and the one in Equation 4 is the $(\alpha_p + 2)/8$ normalization factor, which results from multiplying the $(\alpha_p + 2)/2\pi$ normal distribution function normalization factor with the $\pi/4$ constant from Equation 5.

This normalization factor is hugely important for realism and ease of artist control. Unfortunately, it is not commonly used in game development, so we will spend some time discussing its advantages[2].

Values for the specular power parameter α_p commonly range from 0 to tens of thousands. This means that without the normalization factor, the specular term will be anywhere from four times too bright to thousands of times too dark. This error is large enough to make considerations of correct Fresnel values irrelevant. Omitting the normalization factor makes it extremely difficult for artists to create realistic-looking materials, especially when α_p varies per pixel (as it should). This is one of the primary reasons why the materials in many games look either like plastic or like chrome.

The graphs in Figures 6 and 7 show how highlight brightness varies with angle and specular power, with and without the normalization factor. Figure 8 shows the effect of the normalization factor on simple rendered images; unfortunately, we didn't have time to prepare comparison images with more complex materials, where the difference is even more noticeable.

The normalization factor also has significant advantages for art asset creation. It clearly separates the surface *material* (controlled by $\mathbf{c}_{\text{spec}}$) from its *roughness* (controlled by α_p). With this factor, varying the value of α_p by reading it from a texture (typically called a *roughness map* or *gloss map*) becomes a very effective way to control surface appearance. The values in this texture will simultaneously control highlight width and intensity, as opposed to just controlling the width as in a non-normalized shader.

Another advantage of the normalization factor is that it enables using real-world $F(0°)$ Fresnel values for $\mathbf{c}_{\text{spec}}$ (see the appropriate table in the background talk course notes), resulting in a realistic appearance similar to the desired material. Recall from the background talk that the vast majority of

[2]The fact that the normalization factor causes the reflected intensity $L_o(\mathbf{v})$ to be higher than the light intensity $\mathbf{c}_{\text{light}}$ may seem like a violation of energy conservation, but it is not. The apparent oddity results from the definition of $\mathbf{c}_{\text{light}}$. It is true that the reflection of a light source can never be more intense than the radiance observed when looking directly at the light. However, the definition of $\mathbf{c}_{\text{light}}$ is not related to the brightness of a light source when observed directly, but to the brightness of a white diffuse surface illuminated by the light source.

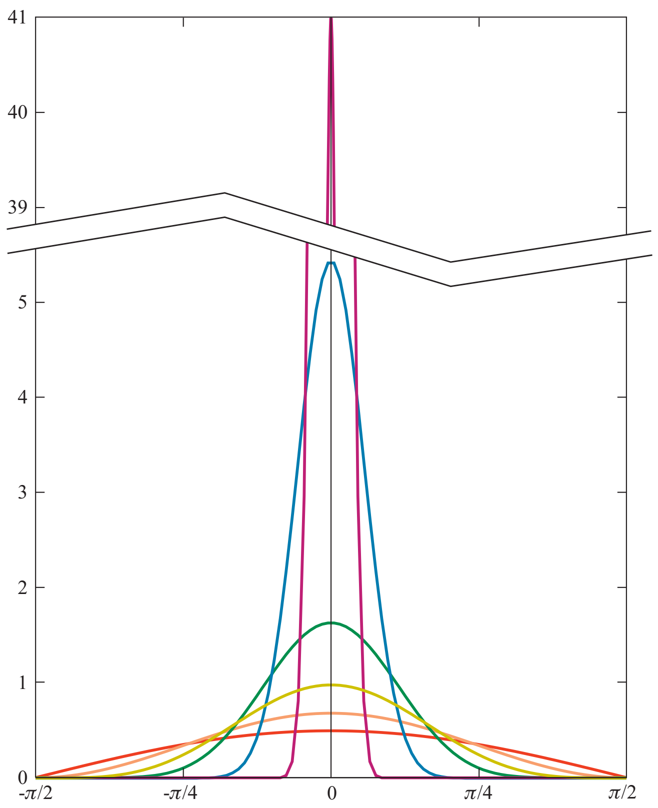


Figure 7: This graph shows the relationship between highlight brightness (y-axis) and angle (x-axis) for a Blinn-Phong term with a normalization factor; the different colors indicate various values of α_p. Note that the center of the highlight becomes brighter as α_p increases, which corresponds to the behavior of real-world surfaces. The overall reflected energy stays constant as α_p is changed. (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

real-world materials (anything that isn't a gem, crystal or metal) has a narrow range of $F(0°)$ values, between 0.02 and 0.06. For surfaces comprised of such materials, the variation in α_p will have a much greater effect on highlight intensity then the exact value of $\mathbf{c}_{\text{spec}}$. These materials can get very good results with a constant value of $\mathbf{c}_{\text{spec}}$ somewhere in the appropriate range, perhaps 0.04. This removes one texture from the shader, leaving only the normal map, diffuse color map, and gloss map. Such a reduction in textures is good for several reasons; it saves storage and texture read instructions, and perhaps more importantly it saves the artist from having to author another texture. Given the large impact of art asset creation cost on game budgets, this benefit is not to be underestimated.

A texture for $\mathbf{c}_{\text{spec}}$ thus becomes an "advanced" shader feature, and the specular power or gloss map is a "basic feature" which all shaders should have (this is, sadly, the reverse of current practice). For these "advanced" materials the artist needs to take care in painting values for $\mathbf{c}_{\text{spec}}$, using tables of real-world values as reference. It should also be noted that there is no such thing as "a surface without specular". Shaders without specular terms are commonly used in games for "matte-appearing" materials. However, in reality such materials have $\mathbf{c}_{\text{spec}}$ values around 0.03-0.06, and very low values of α_p (around 0.1-2.0). At glancing angles, even the most "matte' surfaces have noticeable specular appearance; the lack of this effect is another reason why so many game environments appear unrealistic.

As mentioned above, all objects should use roughness maps to vary α_p per-pixel. Artists should paint fine detail into these maps; real-world surfaces are covered with scratches, uneven wear patterns, pores, grooves, and other features which cause the microscopic roughness (modeled by α_p) to vary. The gloss map is closely tied to the normal map; when generating MIP-maps for both maps the variation in

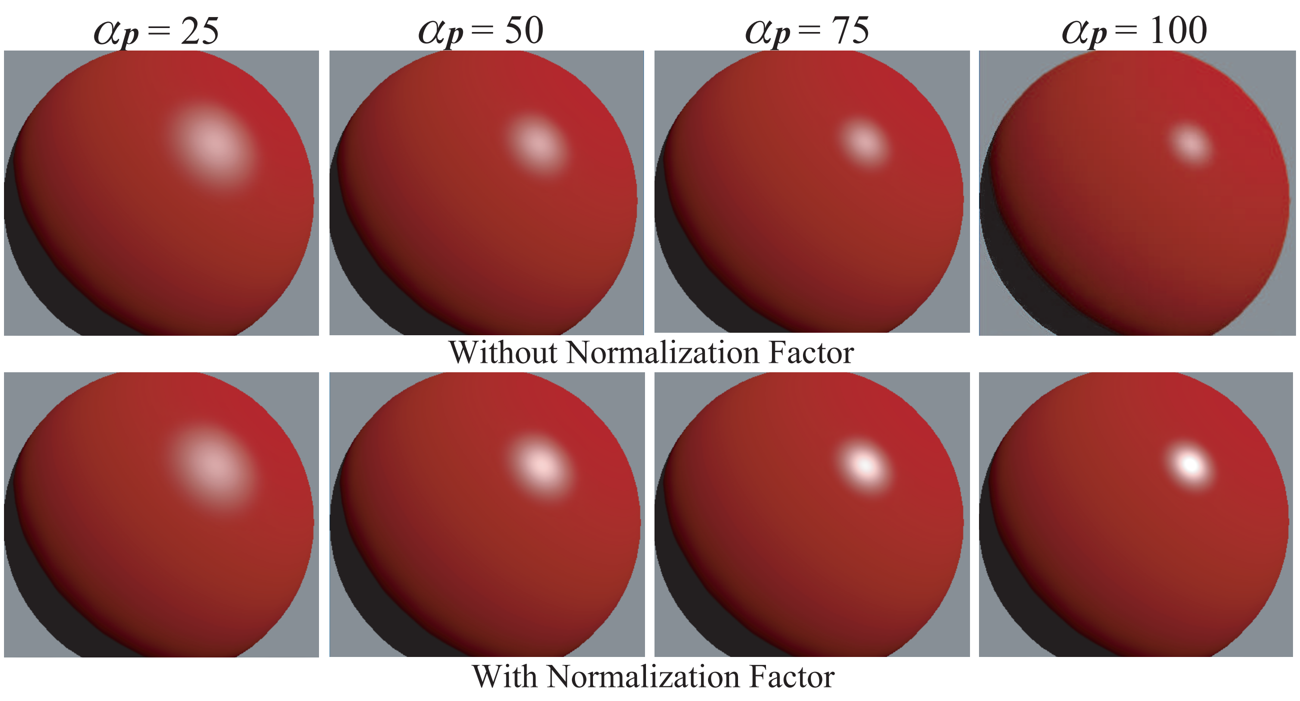


Figure 8: Rendered images of a red plastic sphere. The bottom row of images was rendered with a normalization factor applied to the specular term, using $\mathbf{c}_{\text{spec}} = 0.05$ (an appropriate value for plastic). The top row of images was rendered without a normalization factor, using a value of $\mathbf{c}_{\text{spec}}$ chosen so that the two leftmost images match. The intent is to render spheres made of the same material (red plastic) but with differing surface smoothness. It can be seen that in the bottom row, the highlight grows much brighter as it gets narrower, which is the correct behavior—the outgoing light is concentrated in a narrower cone. In the top row, the highlight remains equally bright as it narrows, so there is a loss of energy and surface reflectance appears to decrease. (*image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters*).

normals should be used to modify the values of α_p [15, 23, 31, 32, 36]. If done correctly, this technique can greatly improve visuals at little or no runtime cost.

For best results, we have found that storing a nonlinear function of α_p in the gloss map helps to utilize limited precision and makes them more intuitive to paint. A good example function is $\alpha_p = (\alpha_{\max})^s$ where $\alpha_{\max}$ is a constant set to the highest specular power that will be used in the game, and s is a $0 - 1$ value read from the gloss map.

Environmental and Ambient Light

Environment maps (typically cube maps in game development) are important when using physical shading models. Since they have no diffuse color, all exposed metal surfaces should use environment maps, but it is worth considering using them everywhere, even on "matte" surfaces. The exact content of the environment map typically does not matter. With a few exceptions (such as a racing game where there is a smooth curved object in the center of the player's attention), incorrectly-shaped reflections are rarely noticed by players. However, it is important for the average color and intensity of the environment map to match the diffuse ambient or indirect lighting, otherwise the material's appearance will change. If both are derived from local samples in the game level (typically precomputed), then they will match by default.

However, it is much easier to vary diffuse ambient lighting continuously over the game environment than to do the same for environment maps. For this reason a way to "track" the environment map

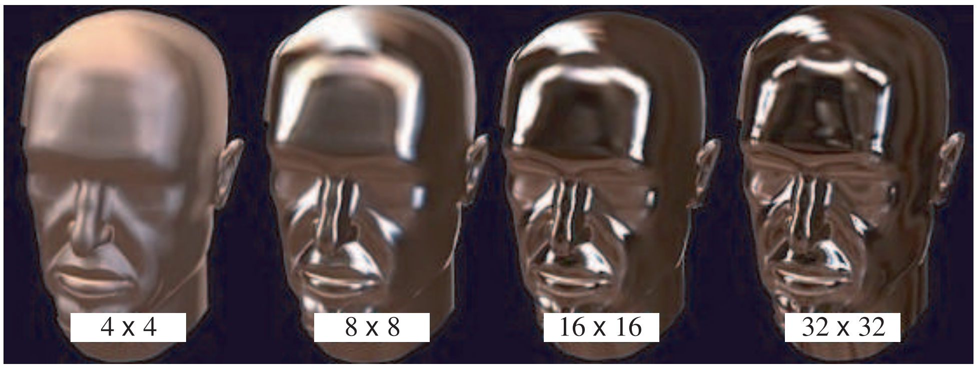


Figure 9: An example of a cube map mip chain filtered with Gaussian lobes of increasing width. Note that this creates a visual impression of changing surface roughness. (*CubeMapGen image from "Real-Time Rendering, 3rd edition" used with permission from A K Peters and AMD.*

to the diffuse ambient is useful. This can be done in a straightforward manner by "normalizing" the environment map (dividing it by its average value) in a pre-process, and then multiplying it by the diffuse ambient in the shader. The diffuse ambient value used should be averaged over all normal directions, in the case of spherical harmonics this would be the 0-order SH coefficient. This can produce quite good results even if the original environment map does not contain an image of the level or even of the same game (e.g., real-world light probes can be used).

Shading with environment maps is reasonably straightforward. The same $\mathbf{c}_{\text{spec}}$ value used for specular highlights should be applied, albeit with a slightly different Fresnel factor. As mentioned earlier, $F_{\text{Schlick}}(\mathbf{c}_{\text{spec}}, \mathbf{v}, \mathbf{n})$ should be used for environment maps and $F_{\text{Schlick}}(\mathbf{c}_{\text{spec}}, \mathbf{l}, \mathbf{h})$ (or the equivalent $F_{\text{Schlick}}(\mathbf{c}_{\text{spec}}, \mathbf{v}, \mathbf{h})$) for specular highlights. If it is desired to use the environment map directly for diffuse shading (instead of "tracking" it to some other diffuse ambient representation as described above), then the environment map should be prefiltered using a cosine term and stored in either a separate low-resolution environment map, the bottom MIP of the specular environment map, spherical harmonics coefficients, or some similar representation.

It is crucial to blur the environment map based on the value of α_p. For low values of α_p the environment map should be very blurry, and for very high values it should be sharp. Low specular powers should blur the environment map and specular highlight by the same amount. Fortunately, most of the hard work of blurring the environment map can be done in a preprocess. The blurring (filtering) should use full HDR values (they can be clipped to a lower range after filtering). It is recommended to use AMD's CubeMapGen library [6]; it has several important features for filtering cube maps that other texture processing libraries lack.

Once the environment map has been properly prefiltered, the shader just needs to select the appropriate MIP level based on the value of α_p. This is particularly effective when combined with per-pixel variation of α_p via a gloss map. Figure 9 shows a simple example of the visual results that can be achieved by prefiltering the cube map and selecting the MIP level in the shader.

If the $\alpha_p = (\alpha_{\max})^s$ gloss-map-to-specular-power mapping is used (as discussed above), then the desired MIP level is a simple linear function of s. The exact function can be calibrated by comparing a prefiltered black environment map with a single HDR texel to a directional light source.

When selecting MIP level in the shader, it is important to compare the desired MIP (calculated from α_p or s) to the MIP level which would be computed automatically by the hardware for a regular cube map lookup. The lower-resolution of the two MIP levels should be used. A straightforward method is to store the MIP level either in a separate cubemap or in the alpha channel of the environment map, and perform two cubemap lookups; one to get the automatic MIP level, and the final lookup. If the alpha channel of the environment map is used, the RGB from the first lookup (which corresponds to an un-blurred reflection) can be used for effects like the "clear coat" double reflection found in metallic car paint.

In principle, environment maps should contain HDR data. It is most important to perform filtering on the full range. Some extended range is needed in the shader as well, but this can typically be handled by scaling to fit into low-precision formats. Since specular color will never be darker than 0.02, the environment map will saturate to white after it reaches a value 50 times brighter than display white (you may need a bit more if you want bloom from environment maps). As discussed in the HDR section, in sRGB space the range required tends to be quite low (100 in linear space corresponds to around 8 in sRGB space).

Other representations of indirection and environmental lighting, such as ambient terms or spherical harmonics can be applied to specular BRDFs. Yoshiharu Gotanda's talk in this course, *Practical Implementation of Physically-Based Shading Models at tri-Ace* gives a specular implementation for constant and SH ambient, a recent presentation by Bungie [3] discusses applying the Cook-Torrance [4, 5] specular term to SH lighting, and a ShaderX^7 article by Schüler [33] describes an implementation of a physically-based specular term with hemispherical lighting.

Fine-Tuning and Future Directions

This section discusses various lessons learned when helping game teams with the transition to physically-based shading models.

Overbright Highlights

Perhaps the most frequent complaint which arises after changing the shading models and reflectance values to physically correct ones, is that the specular is "too bright". There are several reasons for this; we will discuss the two most common ones.

The first reason is related to the behavior of the Fresnel term. In games, fine cracks and divots are typically modeled as normal maps rather than geometry. Since computing bump self-shadowing in the shader is too expensive for most games, it is common for artists to manually darken the diffuse and specular colors in the crevices instead. The aim is to avoid bright shading and highlights from deep crevices that should by all rights remain dark. However, a Fresnel term such as F_{Schlick} will brighten even the darkest specular colors at glancing angles, causing bright highlights to appear in deep cracks.

We are aware of two solutions to this problem. The first solution, proposed by Schüler [33], is to modify the Schlick approximation so that any values under a certain threshold are unaffected. Since we know that no real-world material has a value of $F(0°)$ lower than 0.02, any values of $\mathbf{c}_{\text{spec}}$ lower than this can be assumed to be the result of "prebaked" bump occlusion and left as is, without applying the Fresnel effect. This technique is effective, but it cannot handle more subtle or partial occlusions—occlusion is "all or nothing". For games which have separate ambient occlusion (AO) textures, another possible solution is to apply the AO texture to the specular term. While applying AO to direct lighting is technically incorrect, as long as care is taken to only apply small-scale occlusion (like AO or cavity maps) and not large-scale occlusion (like screen-space ambient occlusion—SSAO) to the specular term then the results are not too bad.

On the other hand, environment maps are more similar to ambient lighting and should have all AO terms applied to them, regardless of scale. Perceptual studies [9] have found that applying AO to environment maps is a reasonable compromise when more accurate (and expensive) forms of reflection occlusion are impractical (which is almost always the case for games).

Another common reason for over-bright specular highlights is related to the diffuse color ($\mathbf{c}_{\text{diff}}$) values. If material artists are not careful, it is easy for them to make the diffuse colors much too dark, making the specular term appear too bright. If the game engine supports setting manual exposure values, dark diffuse values will typically cause the artists to over-expose, again making the specular values appear too bright. To avoid this, it is important to set exposure values using well-known principles such as the Ansel Adams zone system [1]. Basically an unshadowed diffuse white surface should expose to a value a little under display white to leave headroom for specular highlights. Any photo references used for diffuse textures should be carefully calibrated ("dividing out" the lighting). Textures painted from scratch should be carefully visualized as they will appear in game (the OpenColorIO [24] open source project includes relevant workflow examples).

Unsolved Problems and Future Work

Specular highlights and regular (non-prefiltered) environment maps have a well-defined Fresnel terms. In each case there is only one normal vector of interest; **n** for environment maps and **h** for microfacet specular highlights. However, in the case of prefiltered environment maps representing reflections from rough surfaces, there are many different microfacet orientations that contribute to the final color. A cheap Fresnel approximation that represents this case with reasonable accuracy would be a useful development.

Another unsolved problem occurs in the context of very smooth surfaces with high specular powers. Such materials are important to model e.g., wet surfaces. However, the punctual light approximation breaks down in this case, yielding extremely intense highlights of subpixel size that are unrealistic and alias badly. What we would like to see is a sharp reflection of the shape of the light source, which requires some kind of area light approximation which is fast enough to use in games.

A third problem is related to the visual differences between original Phong and Blinn-Phong that were discussed in a previous section. With a single environment map lookup, the visual results are similar to original Phong. Is there an inexpensive way to get stretched "Blinn-Phong-style" reflections from environment maps?

And finally there are a variety of geometry terms in the literature. Do any of them provide a visual improvement over the "cheaper-than-free" implicit geometry function G_{implicit} that is worth the extra cost? One candidate is the geometry factor proposed by Kelemen et. al. [21]. This is an approximation to the Cook-Torrance geometry factor [4, 5] but it is far cheaper to compute:

$$\frac{G_{\text{CT}}(\mathbf{l_c}, \mathbf{v}, \mathbf{h})}{\underline{(\mathbf{n} \cdot \mathbf{l_c})}\,\underline{(\mathbf{n} \cdot \mathbf{v})}} \approx \frac{1}{(\mathbf{l_c} \cdot \mathbf{h})^2}. \tag{11}$$

Just divide by the square of a dot product (which needs to be computed in any case for Fresnel) to get a reasonably close approximation to the full Cook-Torrance geometry factor divided by the foreshortening terms.

Another geometry factor that could be of interest is the one by Smith [34]. It is considered to be more accurate than the Cook-Torrance one, and takes account of surface roughness. Walter [37] gives an approximation to this factor. In Adam Martinez's talk in this course, *Faster Photorealism in Wonderland: Physically-Based Shading and Lighting at Sony Pictures Imageworks*, the use of this approximation for film production shading is discussed. It should be noted that even this approximation is significantly more costly than the Kelemen one; perhaps a cheaper one could be found for game use?

Acknowledgments

The author would like to thank A K Peters for permission to use images from the book *Real-Time Rendering, 3rd edition*, Elan Ruskin for permission to use his photographs, AMD for permission to use the CubeMapGen image, and Paul Edelstein, Yoshiharu Gotanda and Dimitar Lazarov for many thought-provoking discussions on physically-based shading models.

Further Reading

Chapter 7 of the 3rd edition of "Real-Time Rendering" [22] surveys various shading models appropriate for real-time use. More detail can be found in the book *Digital Modeling of Material Appearance* [7] by Dorsey, Rushmeier, and Sillion.

Bibliography

[1] Adams, Ansel, *The Negative: Exposure and Development*, Morgan and Lester, 1948 (we recommend reading the latest 1995 edition). Cited on p. 13

[2] Blinn, James F., "Models of Light Reflection for Computer Synthesized Pictures," *ACM Computer Graphics (SIGGRAPH '77 Proceedings)*, pp. 192–198, July 1977. `http://research.microsoft.com/apps/pubs/default.aspx?id=73852` Cited on p. 5

[3] Chen, Hao, "Lighting and Material of Halo 3," *Game Developers Conference*, March 2008. `http://www.bungie.net/images/Inside/publications/presentations/lighting_material.zip` Cited on p. 12

[4] Cook, Robert L., and Kenneth E. Torrance, "A Reflectance Model for Computer Graphics," *Computer Graphics (SIGGRAPH '81 Proceedings)*, pp. 307–316, July 1981. Cited on p. 12, 13

[5] Cook, Robert L., and Kenneth E. Torrance, "A Reflectance Model for Computer Graphics," *ACM Transactions on Graphics*, vol. 1, no. 1, pp. 7–24, January 1982. `http://graphics.pixar.com/library/ReflectanceModel/` Cited on p. 12, 13

[6] "CubeMapGen," *AMD GPU Tools website*, `http://developer.amd.com/gpu/cubemapgen/Pages/default.aspx` Cited on p. 11

[7] Dorsey, Julie, Holly Rushmeier, and François Sillion, *Digital Modeling of Material Appearance*, Morgan Kaufmann, 2007. Cited on p. 14

[8] Ericson, Christer, "Converting RGB to LogLuv in a fragment shader," "Real-Time Collision Detection" blog, July 9, 2007. `http://realtimecollisiondetection.net/blog/?p=15` Cited on p. 3

[9] Green, Paul, Jan Kautz, and Frédo Durand, "Efficient Reflectance and Visibility Approximations for Environment Map Rendering," *Computer Graphics Forum*, vol. 26, no. 3, pp. 495–502, 2007. `http://people.csail.mit.edu/green/` Cited on p. 12

[10] Gritz, Larry, and Eugene d'Eon, "The Importance of Being Linear," in Hubert Nguyen, ed., *GPU Gems 3*, Addison-Wesley, pp. 529–542, 2007. `http://http.developer.nvidia.com/GPUGems3/gpugems3_ch24.html` Cited on p. 3

[11] Hable, John, "Uncharted 2: HDR Lighting," *Game Developers Conference*, March 2010. `http://filmicgames.com/archives/6` Cited on p. 3

[12] Hable, John, "Filmic Tonemapping Operators," "Filmic Games" blog, May 5, 2010. `http://filmicgames.com/archives/75` Cited on p. 3

[13] Hable, John, "Why Reinhard Desaturates Your Blacks," "Filmic Games" blog, May 17, 2010. `http://filmicgames.com/archives/183` Cited on p. 3

[14] Hable, John, "Why a Filmic Curve Saturates Your Blacks," "Filmic Games" blog, May 24, 2010. `http://filmicgames.com/archives/190` Cited on p. 3

[15] Han, Charles, Bo Sun, Ravi Ramamoorthi, and Eitan Grinspun, "Frequency Domain Normal Map Filtering," *ACM Transactions on Graphics (SIGGRAPH 2007)*, vol. 26, no. 3, 28:1–28:11, July, 2007. `http://www.cs.columbia.edu/cg/normalmap/` Cited on p. 10

[16] Hoffman, Naty, "Adventures with Gamma-Correct Rendering," "RenderWonk" blog, August 3, 2007. `http://renderwonk.com/blog/index.php/archive/adventures-with-gamma-correct-rendering/` Cited on p. 3

[17] Hoffman, Naty, Haarm-Pieter Duiker, Joseph Goldstone, Yoshiharu Gotanda, Dominic Glynn, Lou Levinson, Josh Pines, and Jeremy Selan, "Color Enhancement and Rendering in Film and Game Production," *SIGGRAPH 2010 Course Notes*, 2010. `http://renderwonk.com/publications/s2010-color-course/` Cited on p. 3

[18] International Electrotechnical Commission, "IEC/4WD 61966-2-1: Colour Measurement and Management in Multimedia Systems and Equipment - Part 2-1: Default RGB Colour Space - sRGB," CIE Division 8 website, May 28, 1998. `http://www.colour.org/tc8-05/Docs/colorspace/61966-2-1.pdf` Cited on p. 2

[19] Kaplanyan, Anton, "CryENGINE 3: Reaching the Speed of Light," *SIGGRAPH 2010 Advanced Real-Time Rendering in 3D Graphics and Games course notes*, 2010. `http://advances.realtimerendering.com/s2010/index.html` Cited on p. 3

[20] Karis, Brian, "RGBM color Encoding," "Graphic Rants" blog, April 28, 2009. `http://graphicrants.blogspot.com/2009/04/rgbm-color-encoding.html` Cited on p. 3

[21] Kelemen, Csaba, and Lázló Szirmay-Kalos, "A Microfacet Based Coupled Specular-Matte BRDF Model with Importance Sampling," *Eurographics 2001*, short presentation, pp. 25–34, September 2001. `http://www.fsz.bme.hu/~szirmay/scook_link.htm` Cited on p. 13

[22] Akenine-Möller, Tomas, Eric Haines, and Naty Hoffman, *Real-Time Rendering*, third edition, A K Peters Ltd., 2008. `http://realtimerendering.com/` Cited on p. 14

[23] Olano, Marc, and Dan Baker. "LEAN Mapping," *ACM Symposium on Interactive 3D Graphics and Games (I3D 2010)*, pp. 181–188, February 2010. `http://www.cs.umbc.edu/~olano/papers/` Cited on p. 10

[24] "OpenColorIO," *Sony Pictures Imageworks open source website*, `http://opensource.imageworks.com/?p=opencolorio` Cited on p. 13

[25] Phong, Bui Tuong, "Illumination for Computer Generated Pictures," *Communications of the ACM*, vol. 18, no. 6, pp. 311–317, June 1975. `http://jesper.kalliope.org/blog/library/p311-phong.pdf` Cited on p. 4

[26] Poynton, Charles, *Digital Video and HDTV: Algorithms and Interfaces*, Morgan Kaufmann, 2003. Cited on p. 2

[27] Reinhard, Erik, Mike Stark, Peter Shirley, and James Ferwerda, "Photographic Tone Reproduction for Digital Images," *ACM Transactions on Graphics (SIGGRAPH 2002)*, vol. 21, no. 3, pp. 267–276, July 2002. `http://www.cs.utah.edu/~reinhard/cdrom/` Cited on p. 3

[28] Robertson, Barbara, "Rampant Risk-Taking," *Computer Graphics World*, vol. 31, no. 7, pp. 10–17, July 2008. `http://www.cgw.com/Publications/CGW/2008/Volume-31-Issue-7-July-2008-/Rampart-Risk-Taking.aspx` Cited on p. 1

[29] Robertson, Barbara, "Shades of the Future," *Computer Graphics World*, vol. 32, no. 5, pp. 18–24, May 2009. `http://www.cgw.com/Publications/CGW/2009/Volume-32-Issue-5-May-2009-/Shades-of-the-Future.aspx` Cited on p. 1

[30] Robertson, Barbara, "Rare Mettle," *Computer Graphics World*, vol. 33, no. 5, pp. 16–22, May 2010. `http://www.cgw.com/Publications/CGW/2010/Volume-33-Issue-5-May-2010-/Rare-Mettle.aspx` Cited on p. 1

[31] Schilling, Andreas, "Towards Real-Time Photorealistic Rendering: Challenges and Solutions," *ACM SIGGRAPH/Eurographics Workshop on Graphics Hardware* Los Angeles, CA, pp. 7–16, August 1997. `http://graphics.stanford.edu/courses/cs448-05-winter/Schilling-1997-Towards.pdf` Cited on p. 10

[32] Schilling, Andreas, "Antialiasing of Environment Maps," *Computer Graphics Forum*, vol. 20, no. 1, pp. 5–11, March 2001. `http://www.gris.uni-tuebingen.de/fileadmin/user_upload/Paper/Schilling-2001-Antialiasing.pdf` Cited on p. 10

[33] Schüler, Christian, "An Efficient and Physically Plausible Real Time Shading Model," in Wolfgang Engel, ed., *ShaderX*^{7}, Charles River Media, pp. 175–187, 2009. Cited on p. 12

[34] Smith, Bruce G., "Geometrical Shadowing of a Random Rough Surface," *IEEE Transactions on Antennas and Propagation*, vol. 15, no. 5, pp. 668–671, September 1967. Cited on p. 13

[35] Tchou, Chris, "HDR The Bungie Way," *Gamefest*, August 2006. `http://www.microsoft.com/downloads/details.aspx?FamilyId=995B221D-6BBD-4731-AC82-D9524237D486&displaylang=en` Cited on p. 3

[36] Toksvig, Michael, "Mipmapping Normal Maps," *journal of graphics tools*, vol. 10, no. 3, pp. 65–71, 2005. `http://developer.nvidia.com/object/mipmapping_normal_maps.html` Cited on p. 10

[37] Walter, Bruce, Stephen R. Marschner, Hongsong Li, Kenneth E. Torrance, "Microfacet Models for Refraction through Rough Surfaces," *Eurographics Symposium on Rendering (2007)*, 195–206, June 2007. `http://www.cs.cornell.edu/~srm/publications/EGSR07-btdf.html` Cited on p. 6, 13

Terminators and Iron Men: Image-based lighting and physical shading at ILM.

Ben Snow

Industrial Light and Magic

In the Computer Graphics visual effects community, we've learned to live with the accusation that our work looks fake or looks "too CG". In the business of looking at images, everyone is an expert and will offer their opinion.

Of course, we don't *want* our stuff to look computer generated. We spend most of our waking moments trying to get away from that. This course is about some of the ways we do that and I'm going to use some of the film projects I've worked on at ILM to give examples of how. I'll start with a reminder of what we're trying to recreate, particularly in the Visual Effects context. Then I'll give you an overview of how we've tried to improve that over the years, starting with the introduction of ambient occlusion on Pearl Harbor through the energy conserving, image-based, importance sampled approaches we used on Terminator Salvation and Iron Man 2. Finally, I'll talk a little bit about the process (and challenges) of capturing images and environment information on set.

What are we reproducing here?

In computer graphics a lot of the time we're trying to reproduce the reality that the viewer sees with their own eyes in the world around them.

But with visual effects for film, we're really trying to produce *filmed* reality, or more correctly the view of the world that is captured on film or another medium. And it's an important distinction. One of the things we noticed while doing the planes on Pearl Harbor was the way the highlights would flare out in the real footage. But the filmed references and 8 bit log digital scans compressed the dynamic range of the scene, so when we balanced the ambient and reflection lights the highlights tended to turn into mush. On that film we compensated by boosting the brightest values in the environment map. And further we add an additional bloom to the hottest highlights in the composite – trying to reproduce what happened to them in the medium of film.

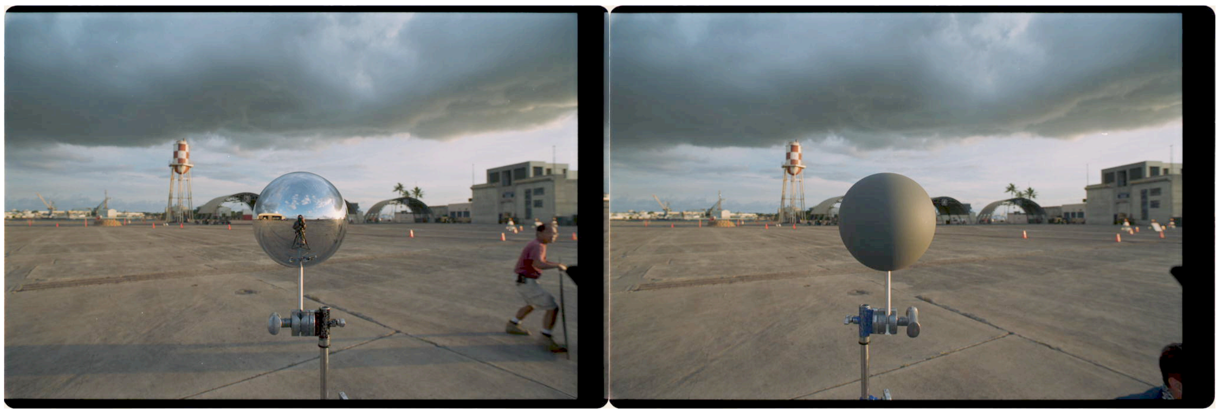
It's not enough to just match the reality of the lighting of the set. This has to be viewed as very much a starting point. As artists we need to use our aesthetic judgment to make our creatures correctly integrated into the scene (and here the physically based techniques discussed in this course are primary), and also presented in an aesthetically pleasing way the supports the story being told by the shot. It's great when the director of photography has some real reference object like a partial Iron Man suit to light when they're lighting the shot. When that isn't the case its great when they're at least thinking of what the effect would be, and light the scene with that at least partially in mind. In that case our reference material and HDRIs are a good foundation and might give a good result out of the box. But equally often the scene won't be lit with the character in mind, and so our captured environments give us a good foundation for the reality of the scene but even more than usual need to be complemented artistically to create a truly effective image.

Another thing to remember is that there are going to be crew around when the material is shot. While you don't want the crew perfectly reflected in your lovely Iron Man suit, you don't always need them to clear out either – if Iron Man had been in when they shot the footage, the crew would be in his reflection, but probably made less apparent with some dulling spray.

The 90s – light probes, textures and environment maps

At ILM in the mid '90s during the first wave of post Jurassic Park films, we were using two types of light probes for calibrating our CG lighting back at ILM with the lighting from the location. The first was a chrome sphere for capturing environment maps and providing a 360 degree picture of the light on the subject and the second a grey sphere as an easily matched reference for how the lighting would affect a known object in the scene.

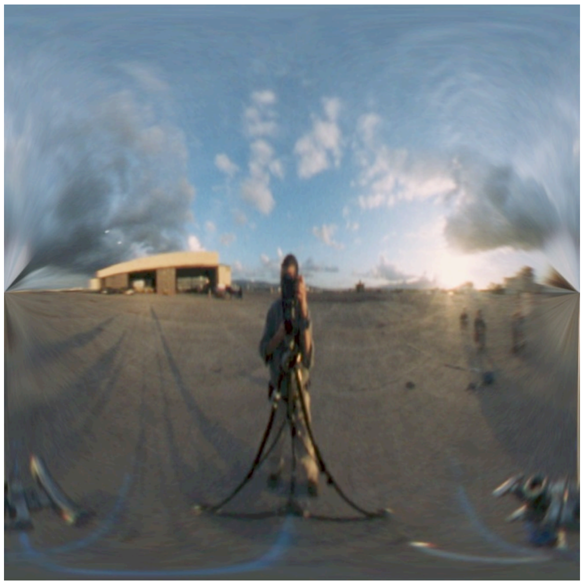
Where possible we still shoot a reflective "chrome" sphere and diffuse "gray" sphere on location, but now we also shoot High Dynamic Range images as well.



CHROME AND GREY SPHERE ON LOCATION (c) 2001 Industrial Light and Magic. All Rights Reserved.

If we can match the grey sphere materials on the computer, we get a way of evaluating how closely our CG lighting matches the lighting on the set.

The chrome sphere gives us a look at that lighting, and we can unwrap it into a spherical environment map that can be accessed with a standard Renderman environment call. We mostly used it for reflections. The chrome sphere gives us a 360 degree view of the scene but the back hemisphere is very distorted at the grazing angle. Also, by the time you shoot it on film and scanned it, you often ended up with an image a few pixels across – way too small to get a ton of detail. One solution to the latter was to shoot still images in four directions with a stills camera and map them on a cube for reflection.



CHROME BALL DERIVED ENVIRONMENT MAP EG(c) 2001 Industrial Light and Magic. All Rights Reserved.

In the early 90s we also relied a lot on painted textures, and fairly simple shaders (straight out of the Renderman companion) maintained by a group of technical artists. We put a lot of faith in our textures, and we had very talented texture artists. The problem with relying too much on textures this way is that you end up with painted in detail like highlights and shadows that doesn't respond correctly when you change the environment, which leads to a bunch of do-over's of look development for different scenes in the movie. So as we improved our look development standards we tried to watch out for that sort of thing.

We were using Cook-Torrance specular functions (and still use a modified version of that in our generic material) but investigated other specular functions like La Fortune. Lighting wise we in general used the standard early '90s CG setup of ambient and reflection environments with spot lights and one or more directional lights for the lighting. People who worked in that environment will remember the evils of ambient and reflection lights which were not correctly shadowed.

Too Much Ambient isn't good for the brain. Mars Attacks! (c) 1996 Warner Bros. Pictures. All Rights Reserved.

On various films we tried things like replacing the ambient with multiple spot lights of low intensity or even with their shadows turned down from 100% opacity.

The rock monster from Galaxy Quest. It was difficult to reproduce all the bounce in this environment and still get realistic shadowing. We used a large number of lights of low intensity.(c) 1999 Dreamworks Pictures. All Rights Reserved.

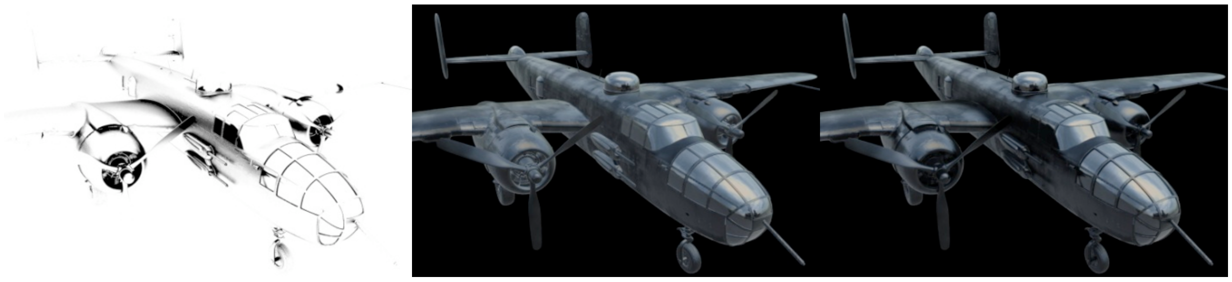
Pearl Harbor and the development of "production ready" IBL.

So, in the rich tradition of visual effects, both Computer Graphics and more traditional, we approach the end of the '90s with a bunch of hacks. Of course ray tracing and global illumination were already in use in production and constant improvements were cropping up each year at Siggraph. But we were, and still are to an extent, in love with the look of our Renderman renders, and were already dealing with scenes of such heavy complexity that ray tracing and global illumination were not practical solutions. At that time we started using Reflection Occlusion and Ambient environments to provide us with a production ready approach to global illumination – a way of providing similar looks to ray tracing and global illumination, but with less expense.

These techniques used a ray-traced occlusion pass that was generated independently of the final lighting. The information in these passes was used as part of the final render calculations in Renderman, but we were able to change our materials and lighting without having to calculate these passes.

Reflection occlusion was developed during Speed II as a way of occluding or shadowing the reflections on the CG cruise ship created for that movie. It was put into full production on Star Wars: Episode 1. It addresses the problem of reflections not being correctly occluded when you use an all encompassing reflection environment. Single channel reflection maps like the one show on the left below are used to attenuate reflection in areas that are either self occluding or blocked by other objects in the scene.

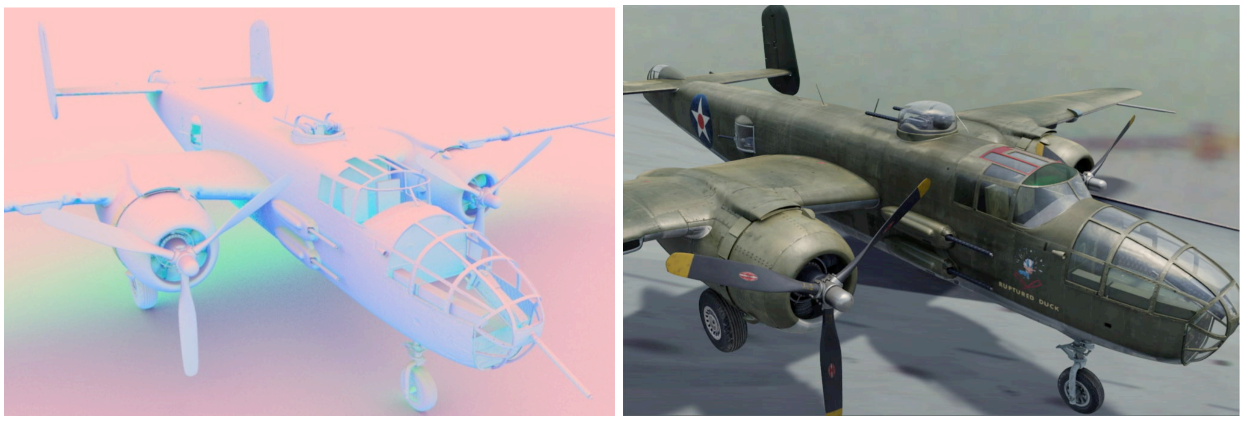
Surface shaders read the occlusion maps and then attenuate the environment to provide us with realistic reflections as shown in the second and third images.



B25 REFLECTION OCCLUSION PASS, B25 WITH REFLECTIONS, B25 WITH REFLECTION OCCLUSION.(c) 2001 Industrial Light and Magic. All Rights Reserved.

Ambient environments is a technique that gave us a way of getting diffuse fill light illumination that was more like what we'd get from global illumination. We introduced this technique on Pearl Harbor. For that film Michael Bay challenged us to come up with a more realistic looking computer graphics airplane than had been achieved before. We also had a need for a computer graphics ships but felt that we hadn't been able to get them to the level of realism that had been achieved with miniatures.

I felt that global illumination was our best bet for the latter, and Ken McGaugh and Hilmar Koch started investigating whether we could get our CG ship model to render in mental Ray. Meanwhile Hayden Landis was working on the airplane solution and felt that the multiple fill lights solution we'd used to approximate more even environmental fill was too cumbersome. He wanted to try using a blurred version of the reflection environment as a big ambient light to provide fill. The problem was that the ambient light wouldn't be shadowed properly but we decided to give Hayden a couple of weeks to try and come up with a solution. Working with Ken and Hilmar, he came up with ambient occlusion, a technique that, like reflection occlusion, uses a pre-rendered occlusion map accessed at render time to give the scene realistic shadowing. In addition, they developed a way to derive directional information so that a given surface point would be illuminated by the most appropriate part of the ambient environment map. Below is an example of the ambient environment render with the different channels used to indicate a light direction, and the final beauty render.



Ambient Occlusion Render, Beauty render with Key Light and Ambient Environment Light only.(c) 2001 Industrial Light and Magic. All Rights Reserved.

Ambient occlusion is achieved by casting rays in a hemisphere around the surface normal for every surface point. The final occlusion amount is dependent on the number of rays that hit other surfaces or objects in the scene. Since the strongest diffuse contribution comes from the general direction of the surface normal, the result is weighted to favor samples that are cast in that direction. But beyond that, we do a first pass to calculate the average direction of the available light arriving at a surface by averaging together the unoccluded rays that hit the surface. This is weighted with the surface normal to bend the direction of the lookup to the appropriate direction in the environment map. We use the term "bent normals" to refer to this effect.

Here are some final images from Pearl Harbor

(c) 2001 Touchstone Pictures. All Rights Reserved.

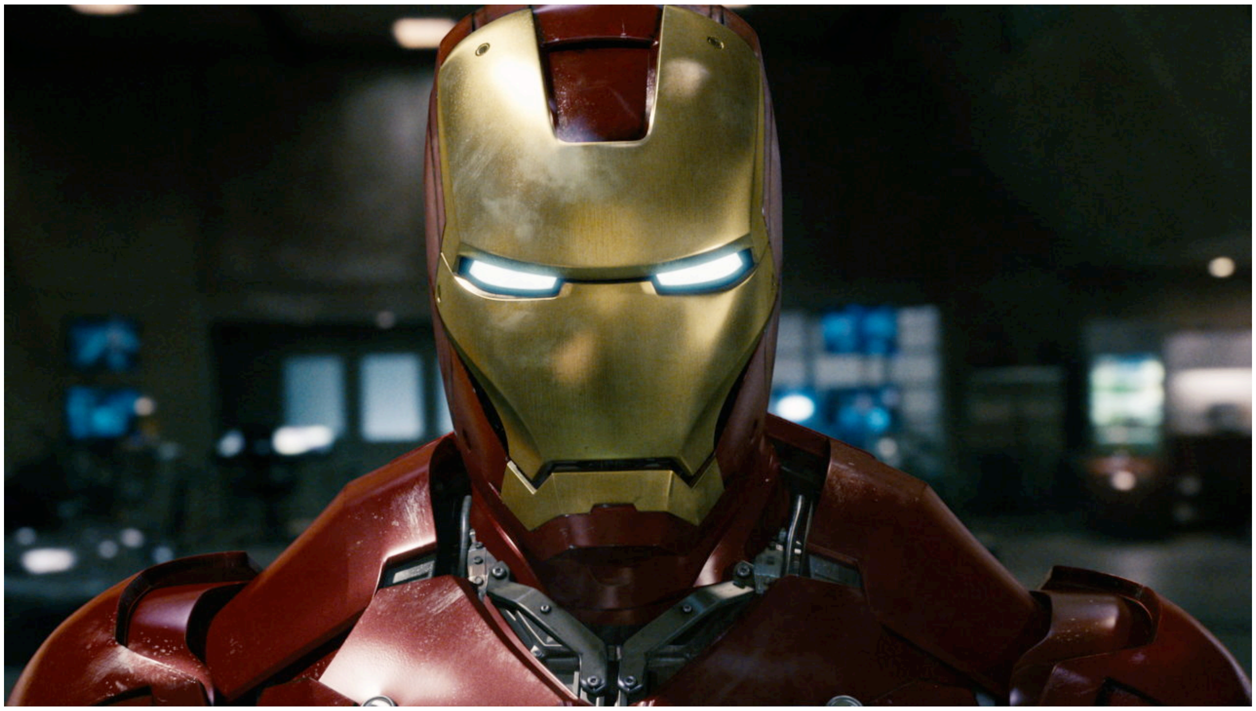
(c) 2001 Touchstone Pictures. All Rights Reserved.

Reflection occlusion and ambient occlusion were great tools that gave us a definite boost in both realism and speed, and they've served us in good stead ever since. But they're not the end of the story.

Around the time of Pearl Harbor we started seeing some very interesting developments in image based lighting with Paul Debevec's presentations of using High Dynamic Range Images and global illumination to make very realistic renderings. At ILM we developed ways to improve the dynamic range of our composited images with the introduction of the openExr format.

On Hulk we started using Sphereon rigs for capturing high resolution and high dynamic range images of the scene, and developed tools to recreate the set and project that captured image material onto rough proxy geometry for ray tracing and faking diffuse reflections. We also made tools to calculate the location of lights in the scene by projecting rays from the capture location to the brightest points in the projected image. This was quite an expensive process at the time and so wasn't generally used in production at ILM until several years later.

Iron Man and better metals.



(c) 2008 MVLFFLLC.TM&(c) 2008 Marvel Entertainment. All rights reserved.

By the time of Iron Man we'd begun to heavily use High Dynamic range images captured on set with digital cameras. We'd also developed a variety of more realistic car finish materials for Transformers, particularly a specular clear-coat ripple and some metal flakes materials. When ILM was approached about doing a test for Iron Man to help make director Jon Favreau more comfortable with the large amount of computer graphics he'd need to use on his film, we were able to leverage the Transformers material to quickly put together a fairly realistic metal suit for the test.

Shot from an early Iron Man test.(c) 2008 MVLFFLLC.TM&(c) 2008 Marvel Entertainment. All rights reserved.

The Iron Man armor for the film was originally been discussed as largely computer graphics, because it had to have a lot of additional functionality that would be impossible to do with a practical costume. However, financial reality set in, and the film-makers decided to get several suits built by Stan Winston's company (which later became Legacy effects). The aim was to try and get many of the shots with practical suits, and then use Computer Graphics to complement that and for things like an aerial dogfight, where it didn't make sense to do practical stuff (although we did talk about things like strapping a stuntman in the suit to the side of a helicopter).

The test and what we were seeing from Transformers told us that our car materials were looking pretty good. We'd also settled on a solid approach for acquiring High Dynamic Range images. But I hadn't really been happy with our raw metal surfaces. Chrome was pretty good, but brushed metal and dull metal surfaces tended to rely a lot on texture maps to add detail, and seemed to come out feeling like a fairly diffuse painted oily metal. Also the Winston/Legacy beauty suits were spectacular. They'd actually chromed and gilded the helmets and parts of the suits and then added paint and other finishes making them look great. As we finished photography it was clear our suits were going to have to hold up very closely with the Legacy suits.

Practical Iron Man suit created by Stan Winston studios (c) 2008 MVLFFLLC.TM&(c) 2008 Marvel Entertainment. All rights reserved.

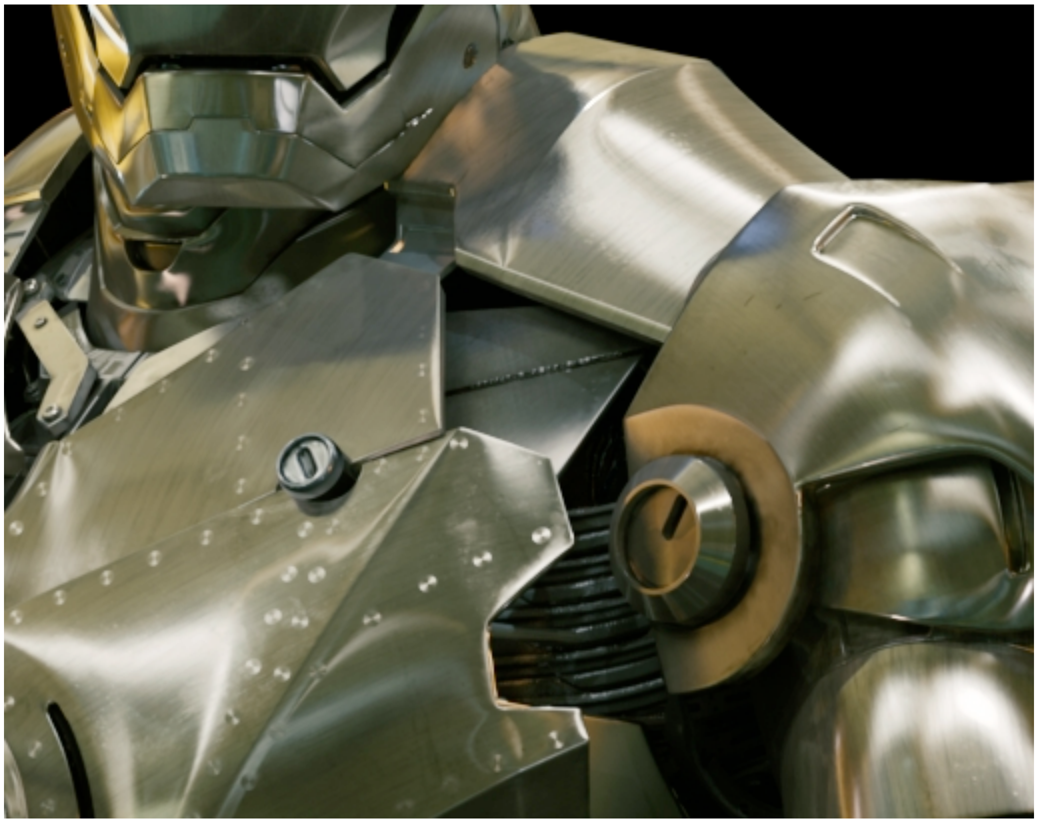
One of the things we decided to do on Iron Man was to try and acquire sampled BRDFs for certain materials. The Legacy effects team made some small references spheres for us of the different suit finishes, including the red armor, the gold armor and the brushed metal of the Mark 2 silver suit. We also had them make 1 inch square swatches with the same finishes and sent them off to have BRDFs sampled.

We got back data on the red armor, and some useful stuff on the gold, but the brushed chrome was a problem. It was a little too complex for the agency doing the sampling.

At the same time, the materials we created for using the sampled BRDF were proving to be a limitation. You could plug in the BRDF table and get quite a nice highlight out of it, but it wasn't terribly easy to art direct on top of that, or tweak or even, initially to change the base color. In the end the sampled BRDFs were useful in helping us better understand how the materials responded to light, but not actually used in and of themselves in our materials for the show.

Based on that analysis we came up a new anisotropic specular function to help improve our brushed metal surfaces.

Anisotropic Brushed Metal



The anisotropic specular function we used on Iron Man added separate X and Y specular exponent values along with a direction-of-anisotropy vector (e.g. "scratch direction"). It allowed separate specular exponent values per R,G,B color channel, giving a color fringe used in the second specular highlight in our shaders. This provided the macroscopic "stretched highlights" characteristic of brushed metal seen from a distance. The meshes were UV laid out so that the primary brushing direction was always exactly vertical in the texture, eliminating the need for a direction map. The specular ended up being close to a regular Cook-Torrance for compatibility with the existing controls. Our modifications were for speed, to add the anisotropic qualities and to normalize it. We'd considered Ward and Ashikmin-Shirley's approach but mostly went with out own approach for backward compatibility issues.

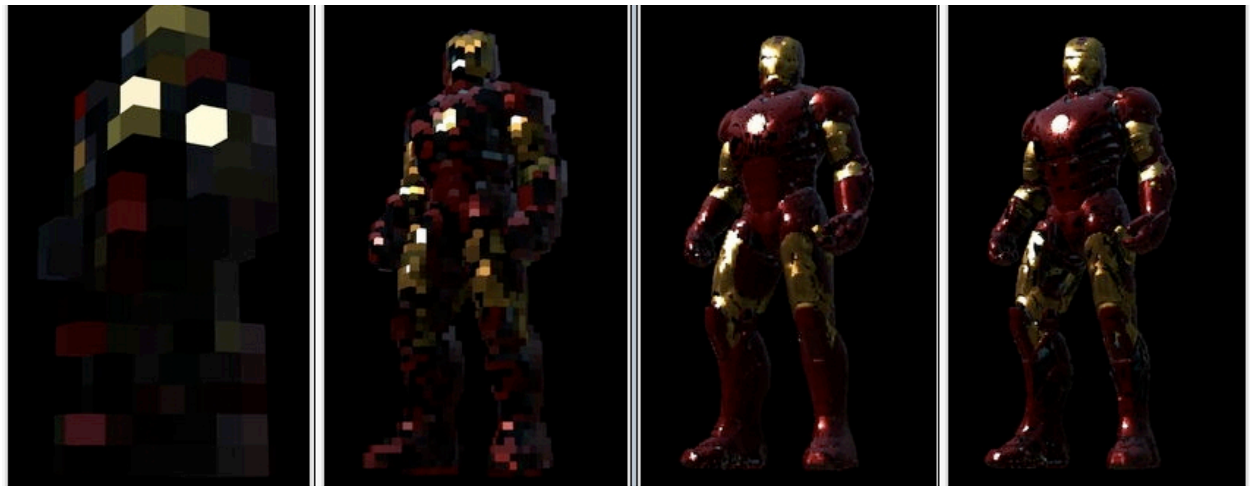
The actual brush lines were painted as a standard specular attenuation map (so they mush together and fade out when further from camera), but we also modified the shader to allow darkening the diffuse and reflection components by the specular map to further enhance the visible brush lines.

This was great for the specular highlights from our individual lights, but we'd also been increasing our use of environment lighting based on high dynamic range images, and when the silver suit was in a more environment light based setup we didn't get the lovely bow tie highlights in the reflections of lights coming from in the environment sphere (because at the time that information was treated as separate reflection). We also figured out a way to get anisotropic *reflections* with environment spheres, but it definitely felt like double the work – something better was needed.

We were able to use some better controls to allow selective ray tracing in Renderman that Transformers had come up with (ray tracing still being pretty expensive in that package). We needed to limit the number of trace bounces for speed. However, our materials basically didn't send out a ray if the reflection occlusion pass told them not to. We also set up base colors that were close to the textured colors so that we could turn off some of the texture maps during the ray tracing passes.

Brick Maps for Ray Traced reflections.

In our need for speed we also increased our use of Renderman brickmaps for cacheing surface data for ray traced self reflections. Brickmaps are Pixar's name for a volumetric texture map. Instead of a tradition 2-D texture map projected mapped onto an object, a brick map defines the color at each voxel that was occupied by some geometry when the brickmap was created. Like textures, brickmaps are mip-mapped in Renderman, meaning they have several nested levels of detail as shown below.



(c) 2008 MVLFFLLC.TM&(c) 2008 Marvel Entertainment. All rights reserved.
(c) 2008 Industrial Light & Magic. All Rights Reserved.

Use of High Dynamic Range Images

By the time of Iron Man, use of High Dynamic Range images for environment lighting was fairly well established at ILM. We were using the images mostly as environment maps for reflection and ambient illumination, still much like the unwrapped chrome spheres from Pearl Harbor. For Iron Man we ended up shooting "highish" dynamic range images with 3 bracketed exposures, (0, -3, +3 stops).

In general we created hero environment maps for each set or location, with one or two variations for different lighting conditions that were shared across the sequence. Much of the lighting was still done with traditional lights even with having the High Dynamic Range images and better tools, although we did start using area lights and reflection cards more heavily.

Look developing the CG suits to match Practical

As mentioned, the aim had been to shoot as much as possible with the practical suits and augment with computer graphics. In the end, by far the majority of the suits were computer graphics – something like 300 CG suit shots vs. 40 practical suit shots. This was because the practical beauty suit was great looking but heavy and difficult for the actors to move in. The lighter stunt suits didn't look as good, even for more action shots. And Jon Favreau was preferring the movements our animators were getting in the animated suits.

It was clear that our Computer Graphics suits had to match the practical suits. With the techniques described above in our toolkit, we were able to get fairly close matches to the practical, which was vital due to the need to cut back and forth.

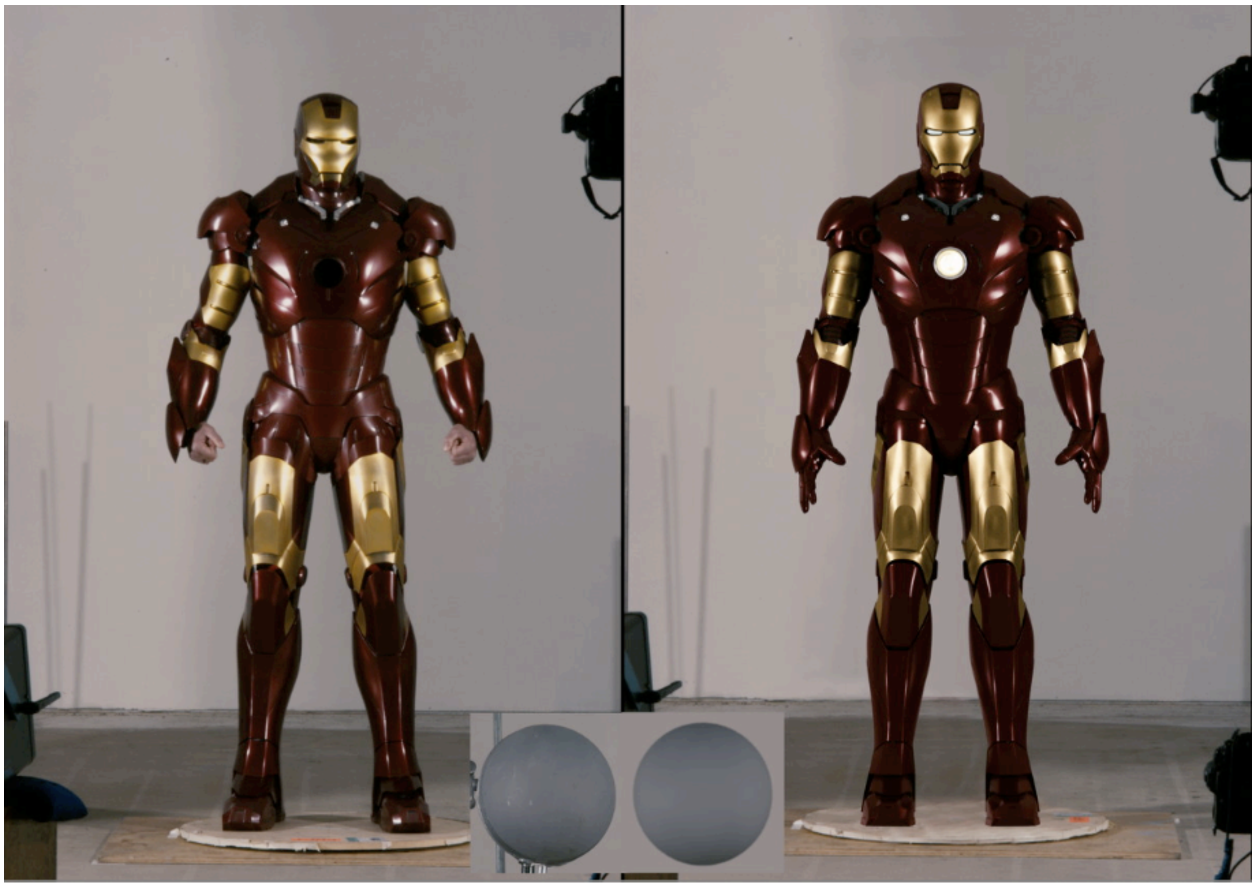
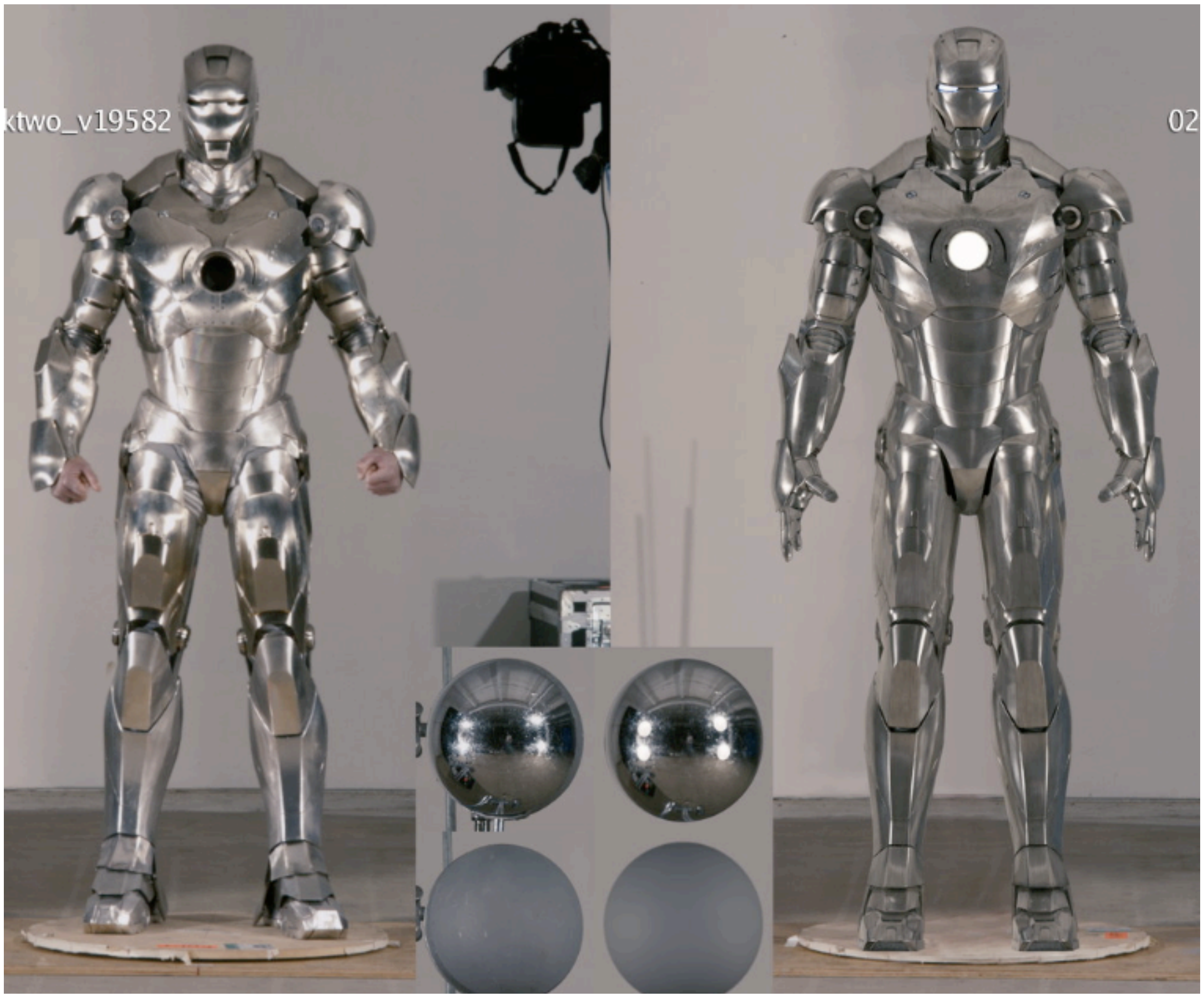


IMAGE: SIDE BY SIDE CG and PRACTICAL(c) 2008 MVLFFLLC.TM&(c) 2008 Marvel Entertainment. All rights reserved.



These looked good in the turntables and also in the shots, but a couple of things became clear. Firstly, the shader setup we were using made it possible to create unbalanced lighting that affected different components (ambient, diffuse, specular, reflection) across different areas of a model, and particularly between different models.

Ambient and diffuse are really the same thing, and specular and reflection are really the same thing, but we had different controls for each, and they would drift out of alignment as the look development artist was working on separate parts of the model, and the balance between the components would be inconsistent. In addition, the way we mixed lights and reflection cards to get a good looking image was rather arbitrary and up to the individual artist, and each of those lighting instruments only affected a subset of all the lighting components – the reflection lights wouldn't give diffuse illumination on a surface and wasn't affected by the surface's specular BRDF function, the CG lights gave too perfect and shiny highlights that lacked the complexity of real lighting. The look could change radically depending on the instrument type used by the artist.

I was also pushing to match more of the lighting that I was seeing on set – the bounce cards, larger light sources like Kino Flos etc., that the D.P. Matty LaBatique used to make the practical suits look good. Area lights were still felt to be a little expensive, so some artists used reflection cards instead – again leading to different results. In the end, we tried to coral the lighting by having a sequence lead set up the instruments to be used in a sequence and prove those with a couple of test shots. The other artists would start with that setup. But it was still a time consuming process getting each shot to match and look good, and we felt that a new approach to lighting was needed.

The other issue arising from the large set of controls over the surface and illumination properties was that we'd need to make sure we ran our turntables in a variety of environments reflecting the environments the creature would appear in. On Iron Man we made a strong effort to making sure our main assets would work in all sorts of conditions, but it was sometimes a struggle.

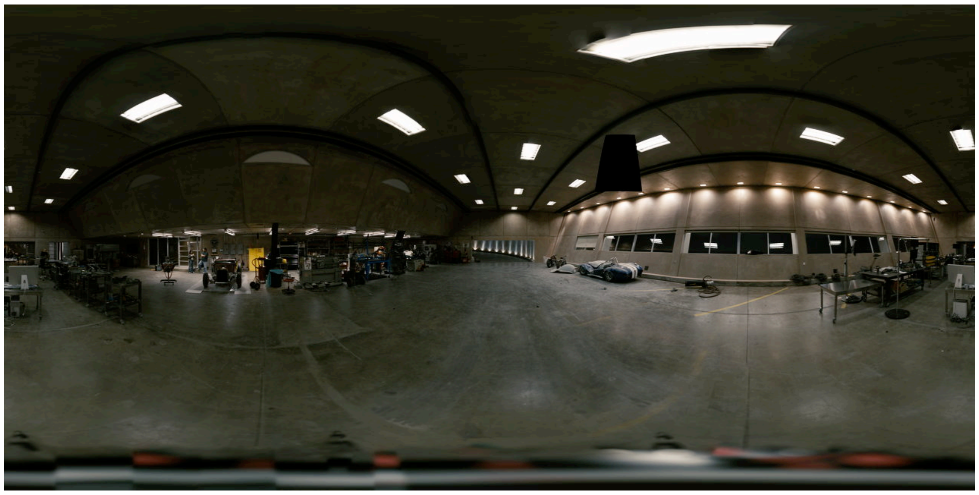
IMAGES: Iron Man in Daylight, Iron Man at night.(c) 2008 MVLFFLLC.TM&(c) 2008 Marvel Entertainment. All rights reserved.

EXAMPLE FROM IRON MAN, MARK 2 SILVER SUIT IN TONY STARK'S WORKSHOP

The beginning of the sequence where Tony Stark tests his silver Iron Man flying suit for the first time was a particular challenge showing some of the tools above. We'd shoot footage with the practical Silver Suit created by Winston/Legacy effects for most of the shots, that set a high bar. We knew we'd have to integrate and replace those with our CG silver suit to make the different control surfaces move around as Tony tested the suit.

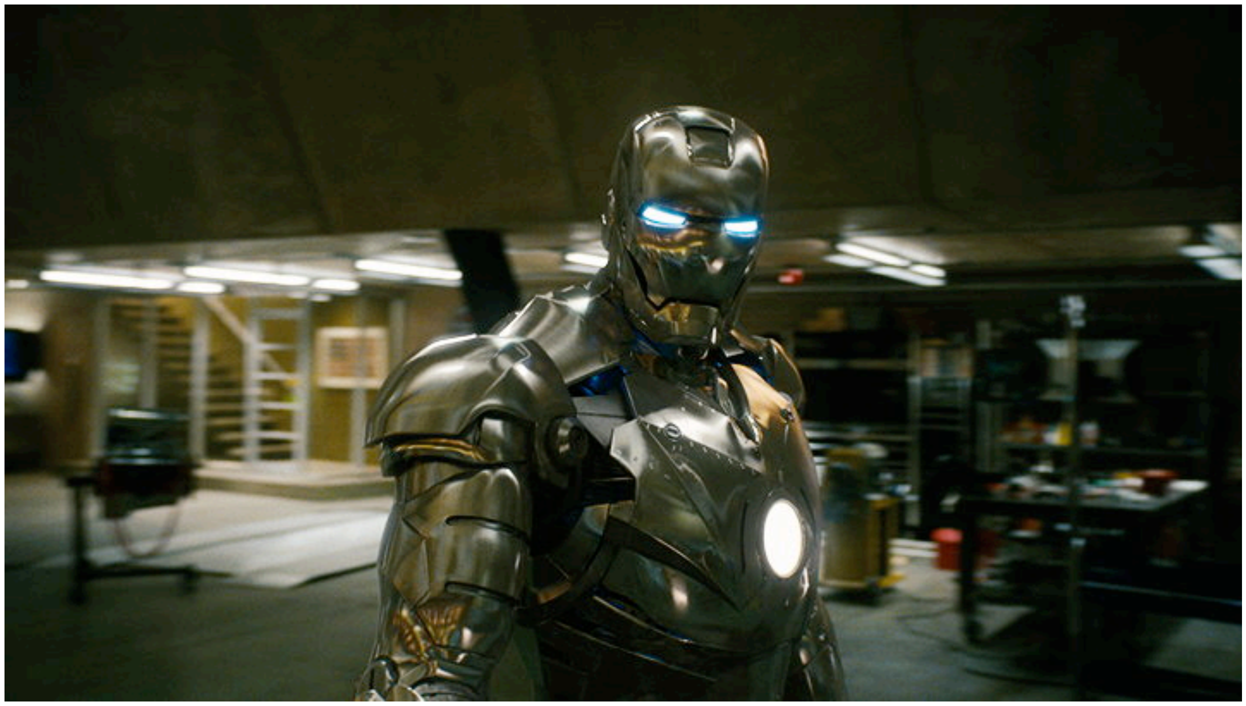
IMAGE: PRACTICAL SUIT(c) 2008 MVLFFLLC.TM&(c) 2008 Marvel Entertainment. All rights reserved.

We started with our "high-ish dynamic range" environment of the workshop.



(c) 2008 MVLFFLLC.TM&(c) 2008 Marvel Entertainment. All rights reserved.

We created a lighting environment with a combination of lights and reflection cards. The Anisotropic materials described above were key to nailing the brushed metal look.



CG Render of Silver Iron Man suit in the workshop. (c) 2008 MVLFFLLC.TM&(c) 2008 Marvel Entertainment. All rights reserved.

Terminator Salvation and more physically correct materials in Renderman.

After Iron Man I moved on to Terminator Salvation, with many of the same ILM people who had collaborated on Iron Man. We were again faced with a bunch of bare, brushed and oily metal and with cutting our computer graphics with practical metallic looking puppets created by Legacy Effects. If anything we had even less of a budget than the already tightly budgeted Iron Man and wanted to leverage image-based lighting for quality, consistency and speed. We wanted to avoid the double work we'd run into with the anisotropic specular and reflections for the silver suit, and we wanted to try and avoid having to rework the materials for the different lighting environments we knew our Terminators would play in.

On Terminator we were faced with fairly dynamic environments filled with practical sparks, fireballs and explosions on the one hand, and harsh exterior desert environments shot in and around Albuquerque in New Mexico. On top of that the D.P., Shane Hurlbut, and director McG had decided on a very harsh and unforgiving digital intermediate process that would great increase the contrast of the image.

Another concern was that unless the computer graphics really matched the dynamic range of what was being photographed we'd be faced with seeing our computer graphics drifting away from the live-action plate – either the black levels would be too bright and flashed in the low end of the computer graphics part of the image, or the bright highlights would drift away in the high end.

In the few years leading up to Iron Man and Terminator, there had been a lot of interesting discussion and presentations at Siggraph and other conferences about physically based materials and lighting, and the move to energy conserving materials, and importance sampled ray tracing. We decided we wanted to use an energy conserving, importance sampled, image-based shading setup.

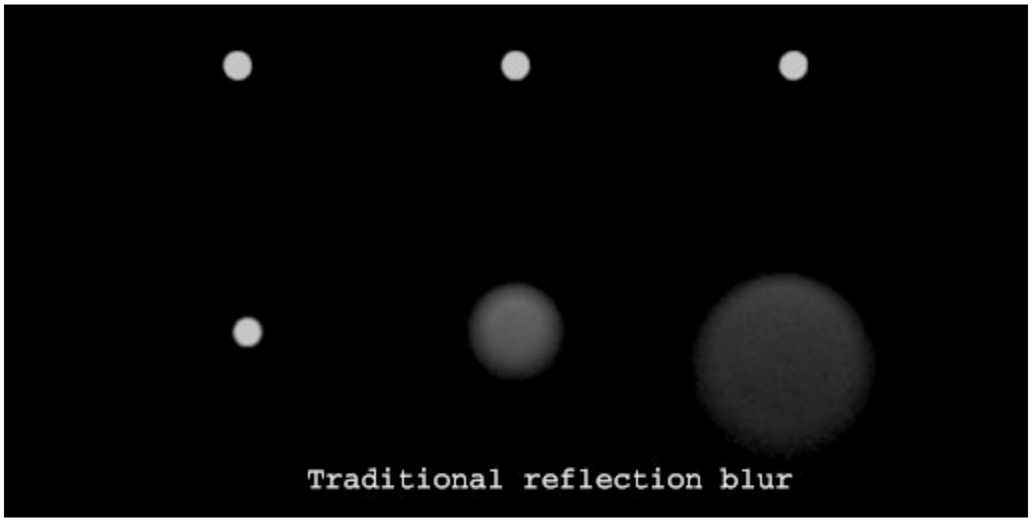
The goal was a simpler, more intuitive and physically based system of lighting and rendering.

Energy Conservation

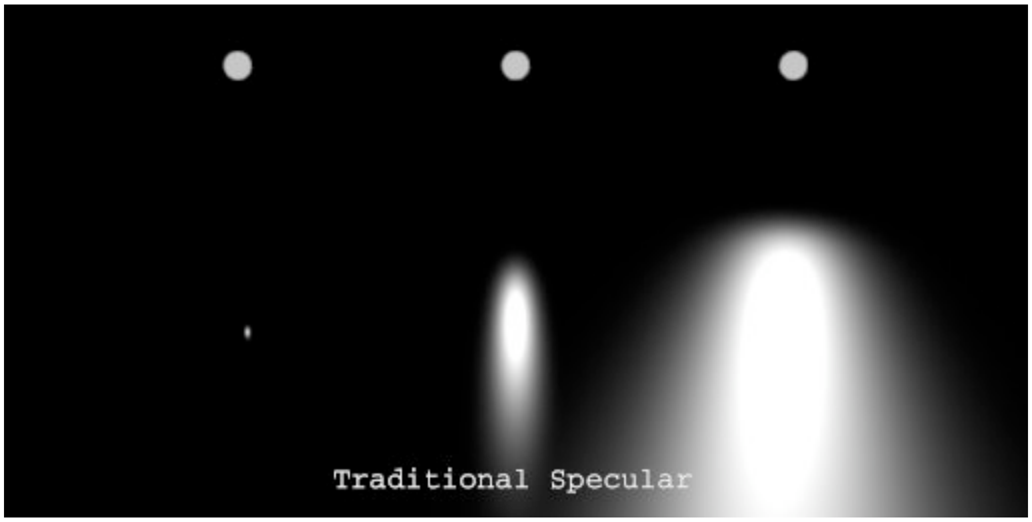
The system we arrived at introduced energy conservation into our shaders. This means that the amount of light that reflects or bounces off a surface can never be more than the amount of light hitting the surface. As described in Naty's introduction to this course, in the physical world incoming light can be reflected off a surface, absorbed by a surface or transmitted through it. Ironically, some of our older shading functions created a situation (for example in Fresnel areas) where more light could be reflected off a surface than was being emitted by the light! To make our shading models more physically accurate, we introduced a new, normalized specular function, improved light falloff response, and importance sampled raytracing. The new tools also combined what we used to think of as the separate components of specular and reflection together into one component we called specular. Likewise what we used to call ambient and diffuse now are combined into diffuse. In addition we created new importance sampled image based lights so we don't use separate ambient and reflection environment lights. So instead of having to manually balance ambient, diffuse, specular and reflection in a material we just have to balance diffuse vs specular. Rather than losing artistic controls it makes it a lot faster to hone in on a more realistic and balanced look because the material is behaving more like something in the real world.

Normalized Specular

Let's start with looking at the traditional setup, where we treated specular and reflection differently. The image below shows reflections in a ground plane of three 50% grey spheres hovering above three different ground planes (apologies that the ground planes themselves aren't clearly shown in these images. In the lower half of the image you see the reflections of the spheres, with each of the ground plane materials having a different reflection blur. As we increase the reflection blur the image of the reflected sphere gets darker because at each pixel we are now gathering light from a wider and wider cone of directions, so the average value of the pixel becomes less.



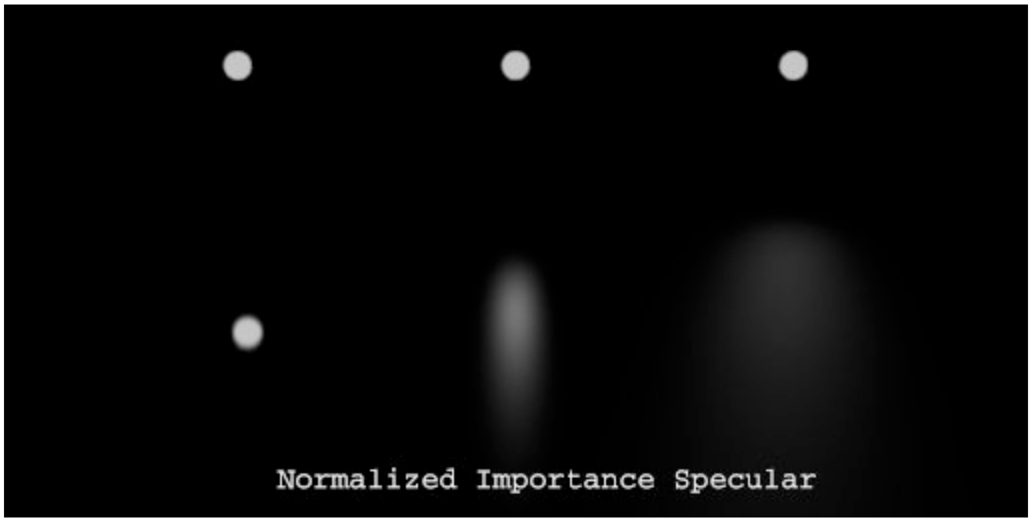
But it didn't work that way with our specular in the traditional ILM lighting scheme. In the image below we have a point light of intensity 0.5 at the position of each of the spheres. These point lights are illuminating three surfaces but this time we're varying the specsize (and not including any raytraced reflection to make the idea clearer). The specular from the point light doesn't get darker as the specsize increases, indeed the specular model we have been using for years at ILM actually gets much brighter with grazing angles so the actual specular values are very hard to predict.



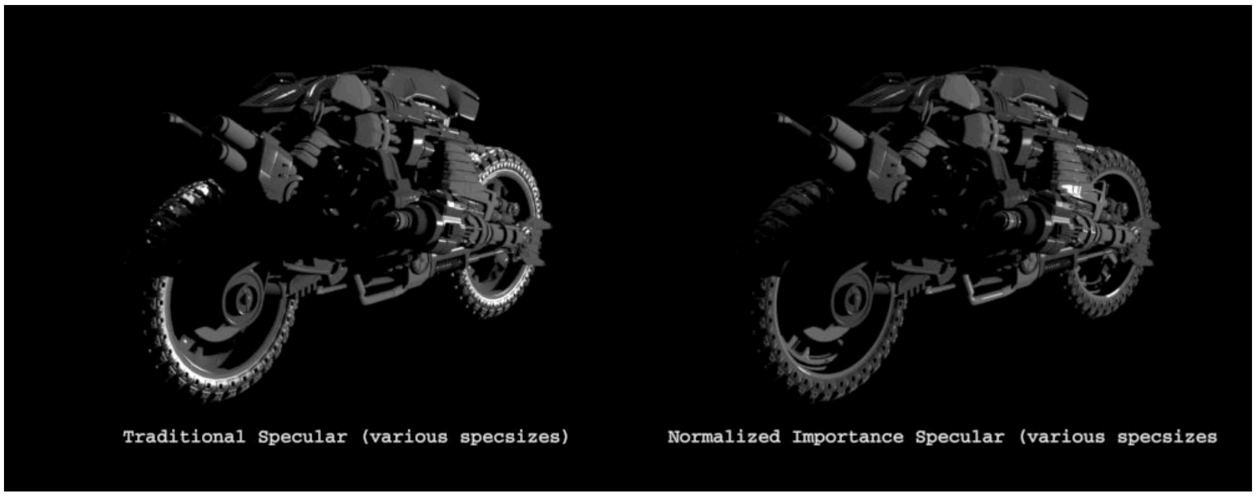
Adjusting the gain of the specular to compensate for different sizes was left to the person doing the look development, so for a model which has a number of different specsizes the way that those surfaces respond to reflections (rougher surfaces should have darker reflections in this example) and the way they responded to specular from spotlights was inconsistent when it should be the same.

So a normalized specular function behaves in the same way that a reflection does. As the specsize increases, the intensity of the specular goes down, and if you looked at a graph of specular response vs reflection angle (ie the specular lobe) then the area under this graph (which corresponds to the spread of energy of the reflection) always sums to 1.0. The lobe is either thin and tall for a small specsize, or short and fat for a large specsize, but the area of the lobe is always exactly the same.

Here is the same render using the new normalized specular model. The specular is the same shape as before but the intensity behaves as it should, with larger specsizes producing dimmer specular reflection.



So let's apply this to the look development on a real world model (or real world in terms of a Terminator movie). Below are two images of a mototerminator with varying specsizes for the various different materials but with the specGain set to 1. The left image is rendered with the old traditional shaders and rhe right with the new importance shaders. The look development person would have to go in and start balancing all the specGains by hand but with the new shaders its looks better out of the box, with the duller surfaces like the tires out of the box reflecting light in the correct balance relative to the shinier surfaces.

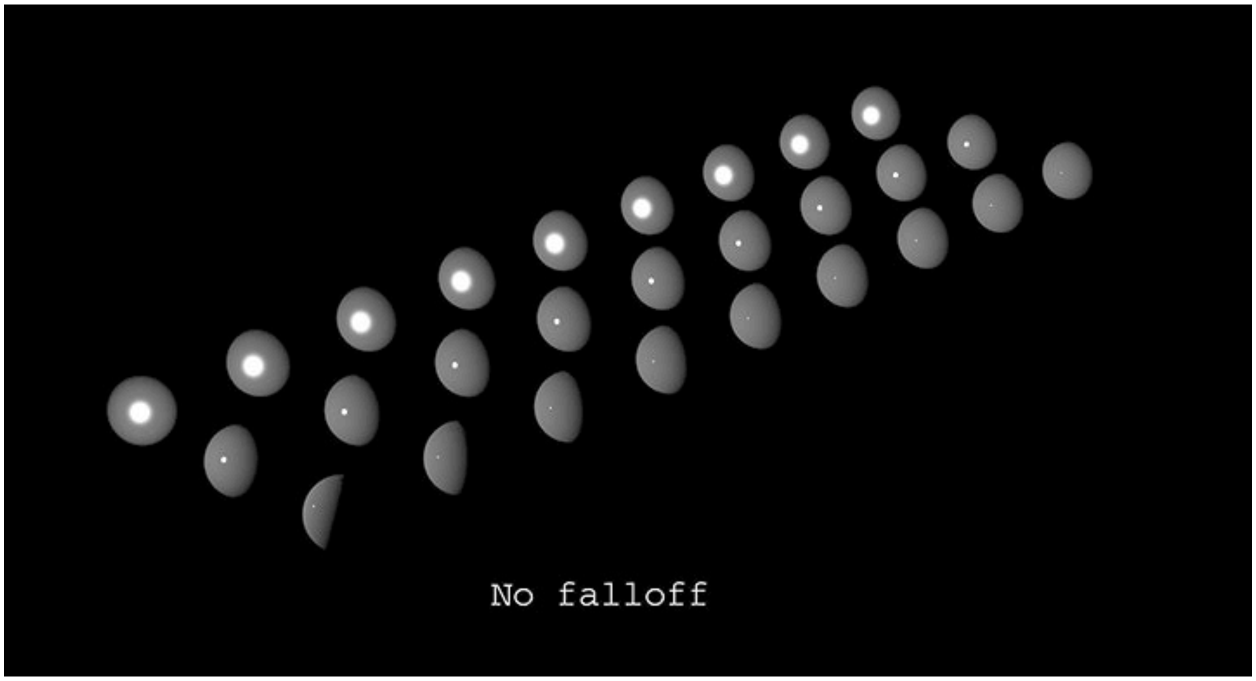

Traditional Specular (various specsizes)
Normalized Importance Specular (various specsizes

Specular Falloff

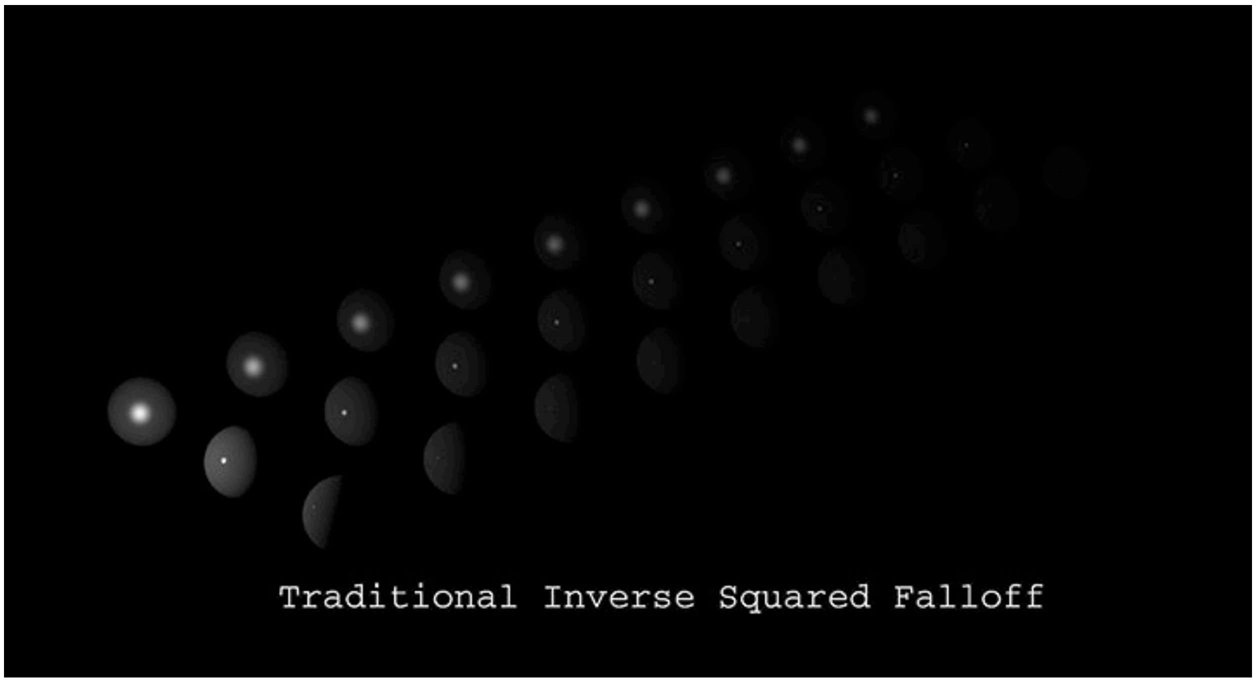
Another feature of the new specular is the ability to get more physically plausible specular falloff as lights recede from surfaces.

Traditionally at ILM, our lights defaulted to having no falloff. We had controls in the light that gave us the ability to add a 1/r2 falloff or a smoothstep falloff between a near and far distance or various other ways of defining falloff.   The falloff would affect both diffuse and specular contributions from the light but even after we added separate control of specular falloff it still didn't account for how the roughness of different surfaces should inform falloff.   When you're lighting a single creature in an environment this is not such a big issue, but as we start rendering more all-cg scenes having a physically inspired model can really help in achieving realistic lighting conditions.

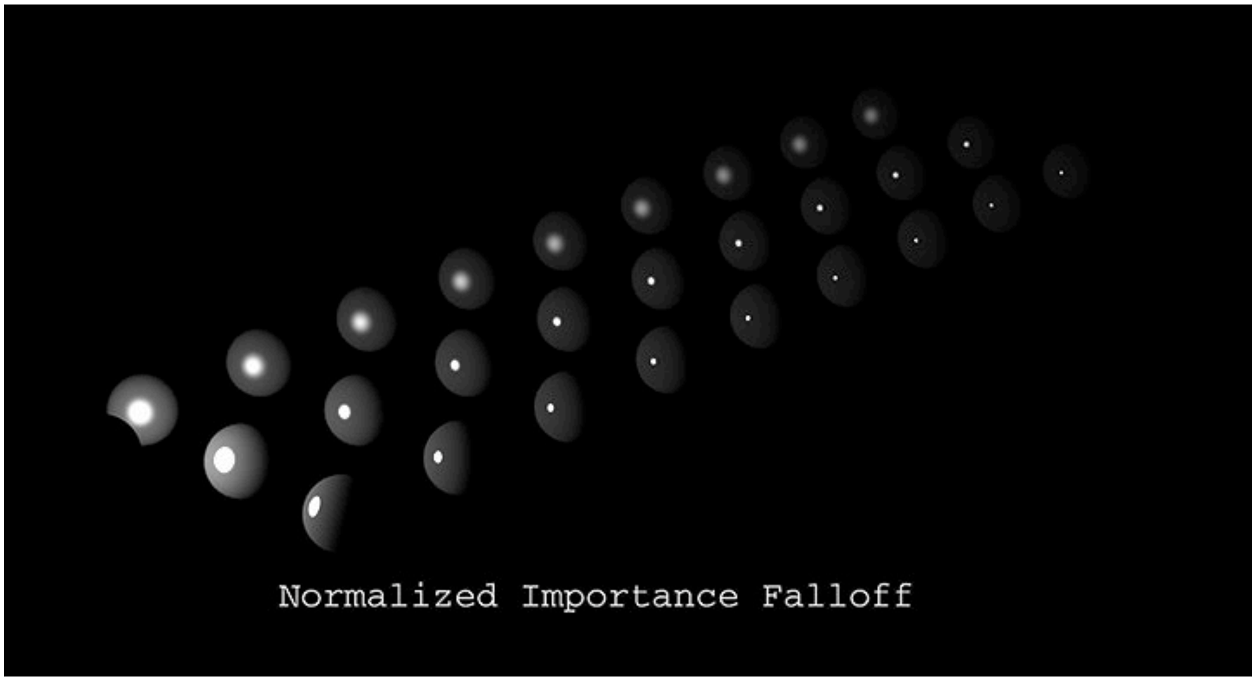
The three images below help illustrate this. There are three spheres, each with different roughness. The first is chrome-sphere-ey (specsize 0.003), the middle a glossier metal (specsize 0.1) and the far one has broad specular (specsize 0.2).



The first render uses the older shaders with no falloff on the light.



The second uses the older shaders with Inverse squared falloff.



The last render uses a spot light again but with normalized-importance surface shaders. Note how on the rougher surfaces the intensity of the highlight falls off faster. This effect is unachievable with the old shader set.

If you've been used to taking the notion of inverse square falloff as the way things should work, this might seem a little odd. While light does "falloff" in the real world, when you are looking directly at something, or looking at something in a shiny surface as the object gets further away you see more of it. With specular reflection an object or light source will only get darker as it occupies a smaller portion of the specular lobe. For tight speculars, like chrome spheres or mirrors the light source has to get a really long way away before it starts to look dimmer. For broad speculars it will dim much more quickly.

For this to work we had to introduce the concept of a light having a physical size. For distant lights like the sun we define the light size as an angle rather than a physical radius and this angle will be the same for every surface regardless of how far it is from the light.

For the implementation we used on Terminator we gave ourselves the option of decoupling diffuse from this behavior, because some of our team felt there were times we might not want to use falloff. This sparked a holy war amongst our team, since to be physically correct the diffuse should fall off as well. There needed to be a shift in the way we thought about lighting, or more correctly, in the way were used to cheating lighting. As our materials and lights became more physically correct we had to treat them more like real lights, including allowing ourselves much greater light intensities and getting used to the light dimming as our slightly dull objects drove flew or swam away from the light.

Normalized Importance Sampled Raytracing

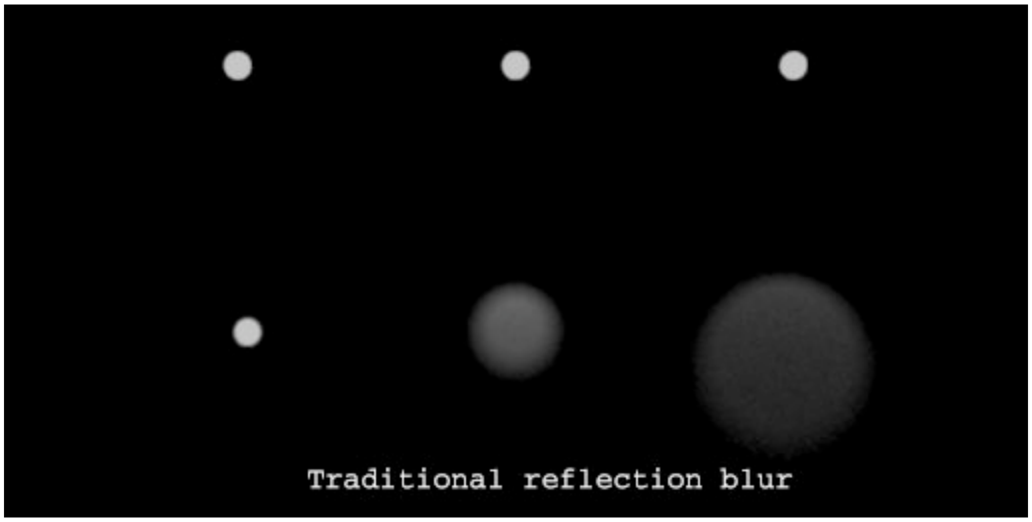
Importance sampling involves trying to sample your scene as efficiently as possible, by focusing your sampling on the points where you get the most bang for the buck. We do this in two places.

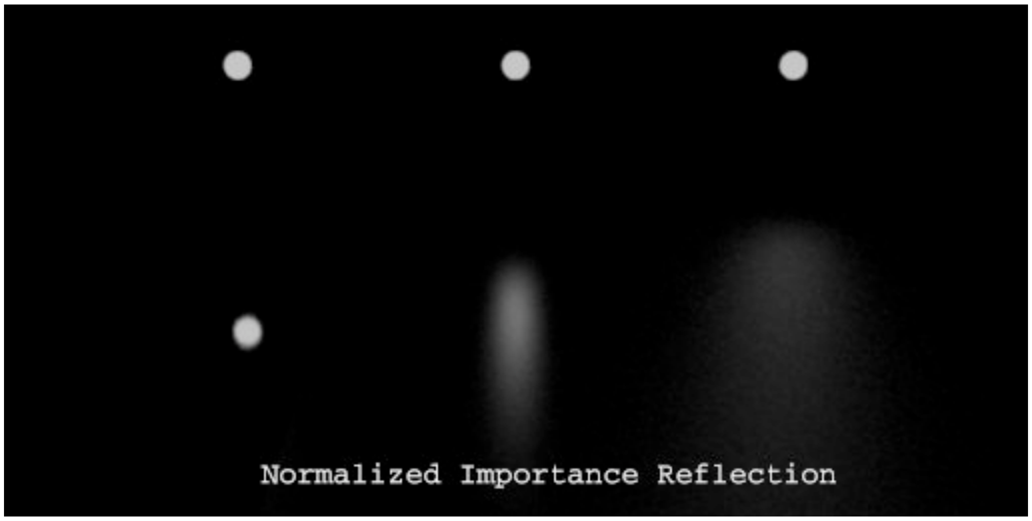
First we importance sample any images used for lighting. We want to concentrate our samples around the brightest points of the image and not waste time on the dark portions that don't contribute much light.

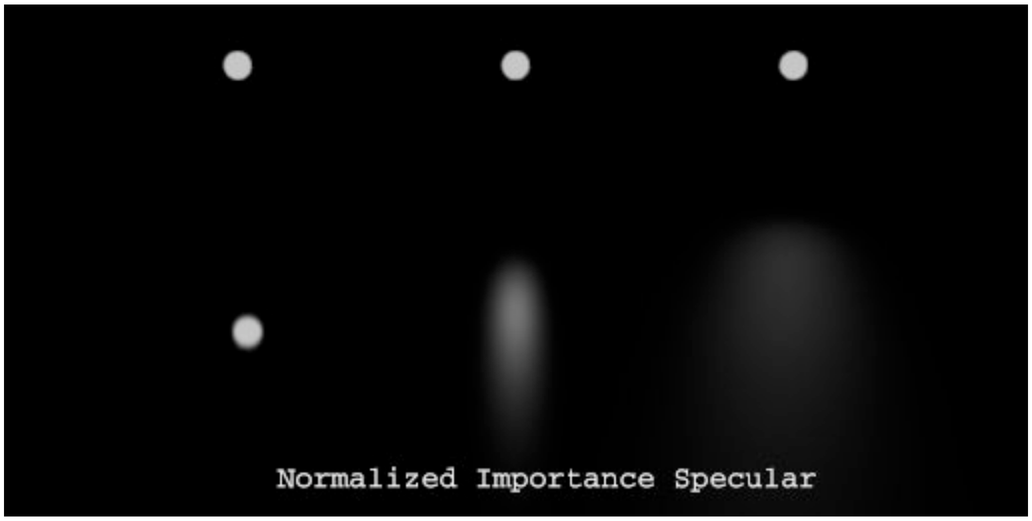
Secondly, when we cast out rays to get blurry reflections, we importance sample the specular function to determine which are the ray directions that will provide most of the lighting contribution, much like the bent normals we'd used back on Pearl Harbor. We also determine the direction which will provide least contribution. Both are taken into account to determine the best possible samples that will accurately capture both the light information and surface response.

A similar technique is used by Sony Picture Imageworks. Refer to the next talk in this course.

Below is a comparison of rendering with our old traditional reflection blur and our normalized importance ray-traced specular reflection.



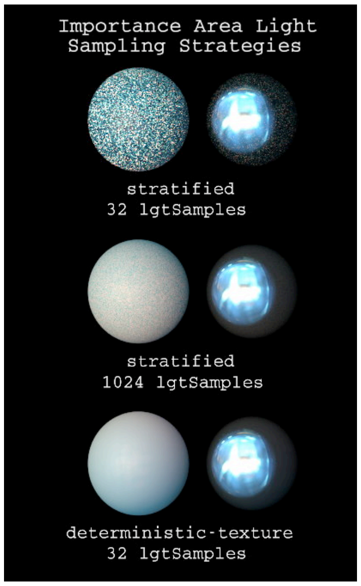




The last image above is the normalized specular image (the second image above) but with the lightRadius set to the same size as the reflected spheres, for comparison with the ray-traced reflection. As you can see from the similarity of these images, reflections from objects, and specular from lights is treated the same way.   This, we feel, is more like the real world.

Importance sampled ray tracing still comes with some cost, and while the time taken to place the lights and get a realistic result has been reduced, the render times have gone up. Sampling strategy is very important, and we were initially getting some rather noisy results.   We introduced sampling controls that increased as the specular sizes needed to increase. We added secondary sampling controls for where a shader is being called by another shader to resolve inter-reflection, indirect diffuse illumination or refraction so by setting these lower than primary values we could avoid an explosion of sampling.

Our preferred importance sampling strategy on Terminator was deterministic texture, which gave us smoother diffuse with fewer samples. However, the deterministic scheme can result in artifacts like banding in the shadows. We also used stratified sampling, which fixes the latter problem but can result in noticeable aliasing with very bright values in environment textures.



Since Teminator we have improved the texture sampling and on Iron Man 2 implemented a “mediancut-interleaved” sampling method, although the improved texture sampling on that show worked very well also.

For more information on importance sampling, I refer you to Pharr and Humphrey’s “Physically Based Rendering” which I believe has a new edition out at Siggraph. There is also a siggraph course on Tuesday at 9am entitled “Importance Sampling for Production Rendering “ at which Simon Premoze, who contributed greatly to our new tools, is one of the presenters.

SWITCHING TO THE ENERGY-CONSERVING, IMPORTANCE SAMPLED, IMAGE-BASED MATERIALS

The development of the new materials and lights took a bit longer than expected for our production needs for Terminator Salvation and so we decided early on to start with the traditional lighting and then commit to using the new approaches on a couple of sequences. We left the moto-terminator and giant Harvester sequences as using the traditional approaches and decided to use the new approaches for our skeleton terminators – the old, clunky T-600 that attacks John Connor in the beginning of the movie, and the evolved T-800 skeleton that attacks him in the Terminator factory at the end. This was because the beginning represented a high contrast environment where we had to cut back and forth with a practical Legacy effects T-600 puppet, and that end factory sequence was largely going to be illuminated with flashing lights, smoke and explosions and could benefit from a more image based approach. As work progressed we also made the resistance A-10 Warthog airplanes and the chase down the canyon using the new setup.

Our test case was a CG version of the practical Tow Truck in the film. While we wanted the new tools to give us results that were close to, but an improvement upon the older materials, the goal was to match reality – or our filmed version of reality – and so we used a real world target for our initial testing, as well as a suite of methodical, more technical tests.

A couple of points here. We put a lot of effort into establishing a fairly solid Image-only based setup for our generic turntable environment (seen in the Truck turntable below). It's particularly valuable to use IBL for look development, because it soley focuses the effort on material setups. With our solid generic environment we felt more confident in being able to transfer an asset between quite different environments without having to greatly re-jig the look. For instance, with a "pure" IBL setup you don't end up baking the response of specific lights in your BRDF parameters. There is a school of thought which suggests that you'll do a better job of judging your material if the lighting field in which the observation is made is complex and high frequency (something you get pretty much out of the box with the real world captured IBL data).

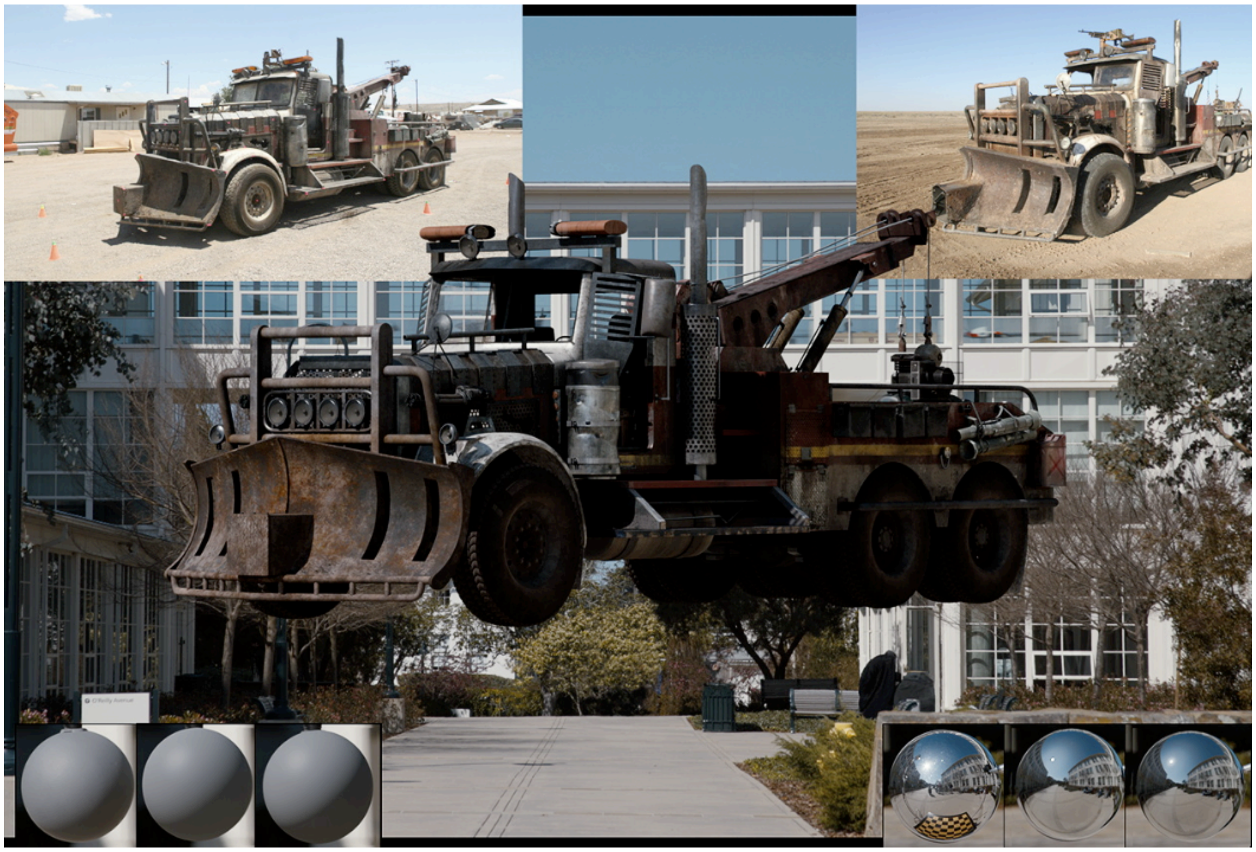


IMAGE: Turntable frame of C.G. Tow Truck (c) 2009 Industrial light and Magic. All rights reserved.

MATERIALS EXAMPLE – INITIAL T-600/JOHN CONNOR TERMINATOR FIGHT

Here is our CG T-600 endo-skeleton. In the end this is featured in all but a couple of the shots in that initial fight.

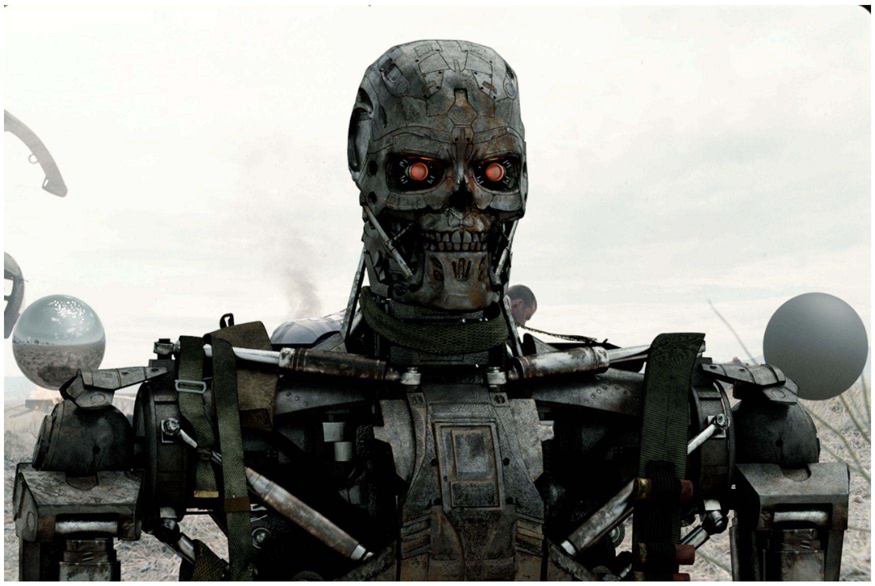


IMAGE: T-600 turntable. (c) 2009 Industrial light and Magic. All rights reserved.

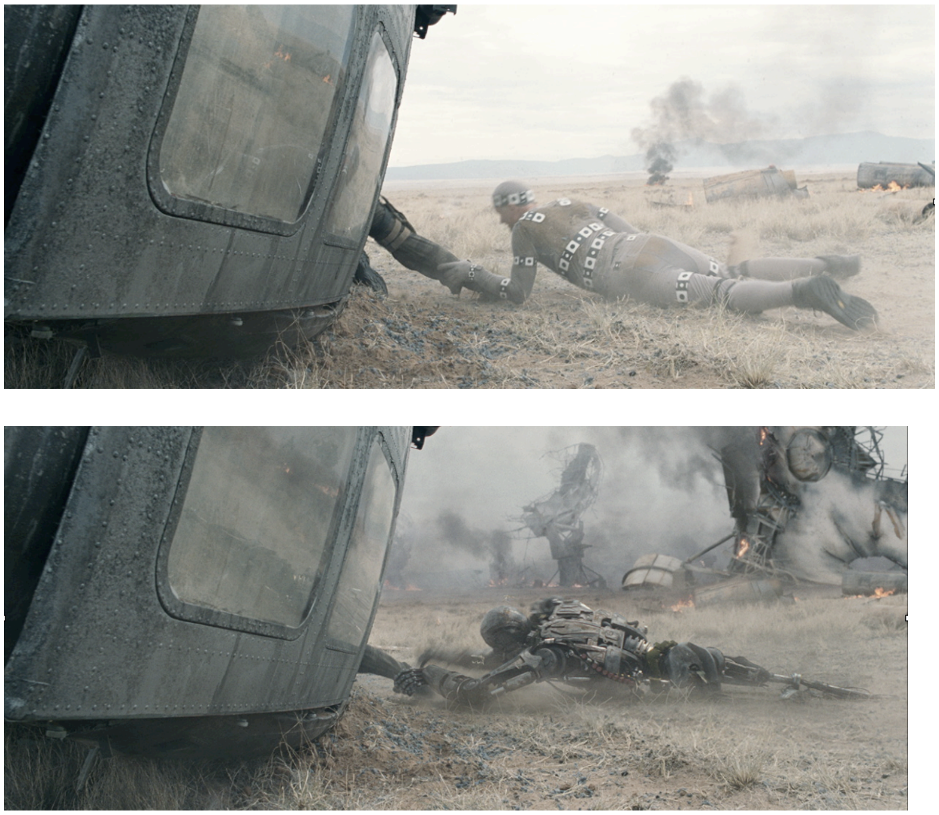


IMAGE: VE fight before/after. (c) 2009 Industrial light and Magic. All rights reserved.

As we started TD's out on the sequence we found there was a learning curve. Initially, the renders were a lot slower than what they were used to. However, this was offset by not having as many pre-passes for ambient occlusion, brick-mapping etc. so the complexity of the scenes was less. Another issue was that the TDs are so used to a bunch of 3D and 2D cheats to make their shots look good that they found it odd to be working in a more physically correct world where things didn't work the way they used to in the more artificial traditional approach. We almost had a holy war over eliminating the ability to control falloff in the lights, and ended up re-implementing it as described above.

We also asked them to use much more of an image-based approach, and so the environment sphere became a lot more important. The instruments used to add shadows were different – you couldn't just add some barn doors or paint a cuculoris in a slide map – you instead had to use more real-world approaches like adding flags and reflectors. A lot of the time the artists would basically go in and paint the environment map. Shots where the terminator transitioned from outside to inside the downed helicopter during the fight were a problem, due to the need to move between two environments.

In the end for speed we did set things up to allow us to selectively mix in ambient environments as well as pre-computed point-cloud approaches for occlusion and indirect diffuse calculations. It was a bit of a struggle, but we felt the new tools helped the realism and as the tool set matured were definitely the way to move forward.

IMAGE-BASED LIGHTING ON TERMINATOR SALVATION - THE T-800 FIGHT

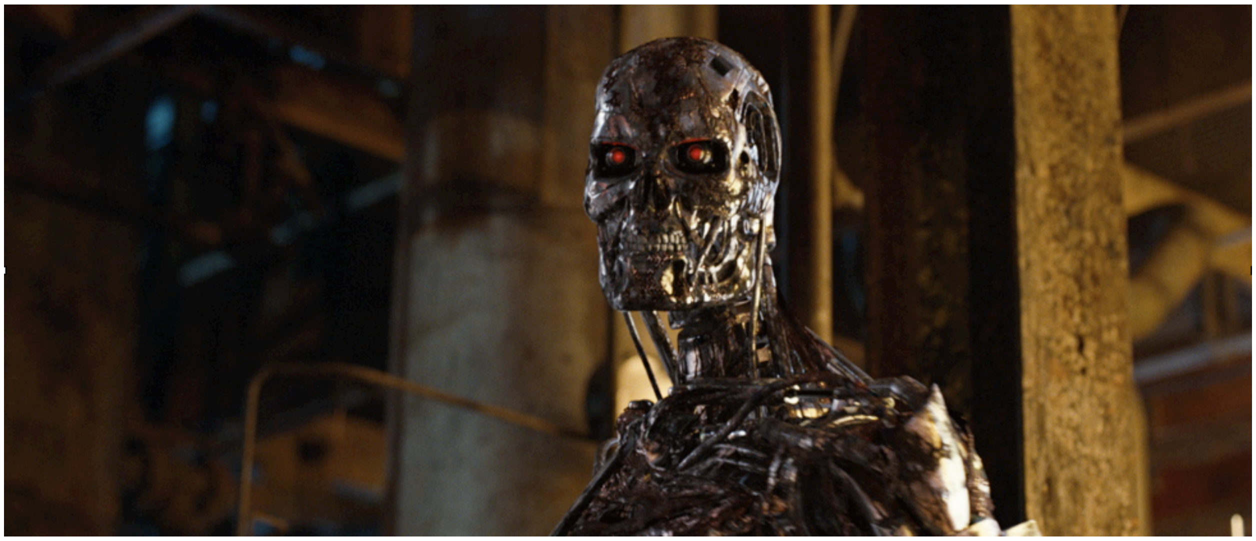


IMAGE: T-800 in the Terminator Factory

The terminator factory sequence at the end of Terminator Salvation, where the John Connor and Marcus characters face off a charred T-600 endo skeleton was ideal for an image-based approach. D.P. Shane Hurlbutt was lighting the sequence with a lot of pyrotechnic effects creating a chaotic, rapidly changing environment. The issues of specular and reflection being treated differently in the old shader set could create a variety of headaches with all the reflections, but the new energy conserving materials solved this problem for us nicely.

Capturing the environment
This was our first problem. We don't have a good way of capturing high-dynamic range moving images. Our solution was to have the film-makers hold the dynamic effects like explosions and freeze the strobes (where possible) for our HDRI digital stills. Then we'd shoot a chrome sphere with all of the effects. Finally, we shot a whole bunch of the pyro elements on film, both for use as elements to add even more chaos in the 2D composites, and to use as reflection/area lights in the scene.

Creating the sequence foundation
To help smooth over the transition to our new energy conserving shader set, we decided, like on Iron Man, to have our sequence supervisors set up the environment maps and lighting instruments for the sequence. Beyond that, 'though, we had the sequence supervisor actually run a first lighting pass for each of the shots. One of the benefits of the new approach was that the shots looked better out of the box just using the environment maps, and with the lead TDs setting up the shots, we'd always have a high standard first take. It would also mean that if need be we could get a real-looking render out even during animation if we were having trouble getting the animation approved by the director – to remove one more variable.

The sequence lead would start by finding a hero HDRI image set for a shot or group of shots, and have that stitched into a lighting sphere. All the bright lights would be painted out and either images of the lights from the sphere image, or images of the different pyro events we'd filmed separately would be used. The sequence lead would document the different environments and lighting instruments available to the artist.

Choreography of the lights
When the TD picked up the shot, they found a well set up shot file including the environment and many of the lighting tools they needed already set up and running. But they still had to position the lights and start using the running footage to match the chaotic lighting going on in the background. This was greatly helped by a hardware rendering tool Pat Conran wrote for the show called layer cake, which gave fast GL previews of the lighting.

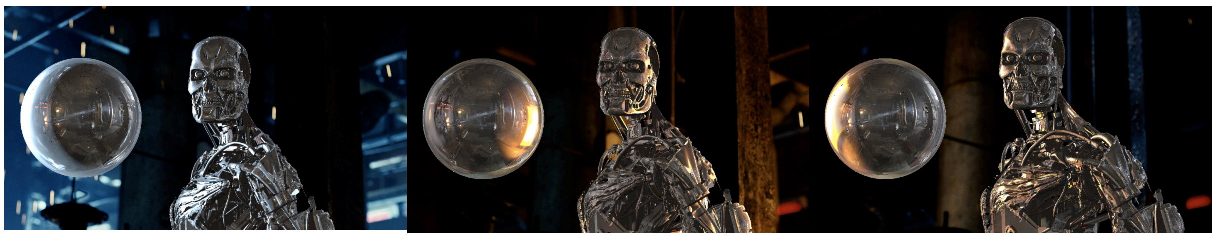
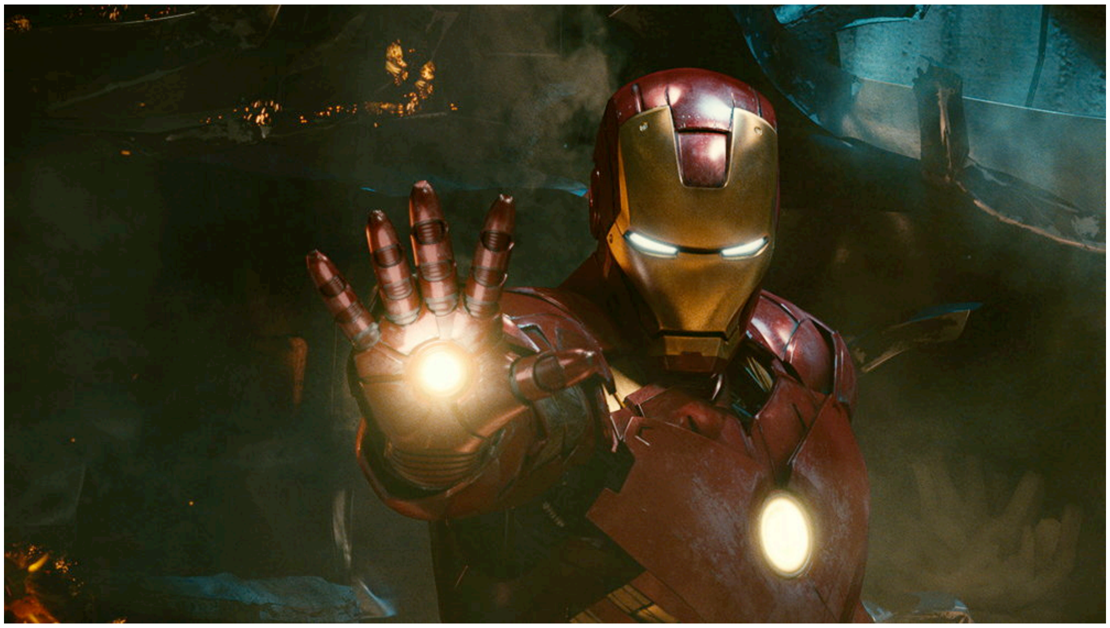


IMAGE: A sequence of frames from hardware preview. Note the element cards in the spheres. (c) 2009 Industrial light and Magic. All rights reserved.

This approach to lighting was new to us all, and as a diagnosis tool we added a feature that rendered a chrome sphere automatically centered in front of the camera for the shot, and baked it into the movie files we used to review the shots in dailies. This gave me a sense, at a glance, of the lighting choreography that the artists were using, and is a tool I really like to have in the tool set.

Iron Man 2: Putting the new shaders and lights to work.



(c) 2010 MVLFFLLC.TM&(c) 2010 Marvel Entertainment. All rights reserved.

Going into Iron Man 2, we were able to put the new energy conserving, image-based tools to work on the whole production. The shaders and lights themselves had matured, and we'd removed some of the hacks from the transition phase so that, for example, you couldn't override the falloff on lights as easily. The first thing we did was to update the materials for the Iron Man Mark 2 and 3 suits seen in the previous film to our new shading setup.

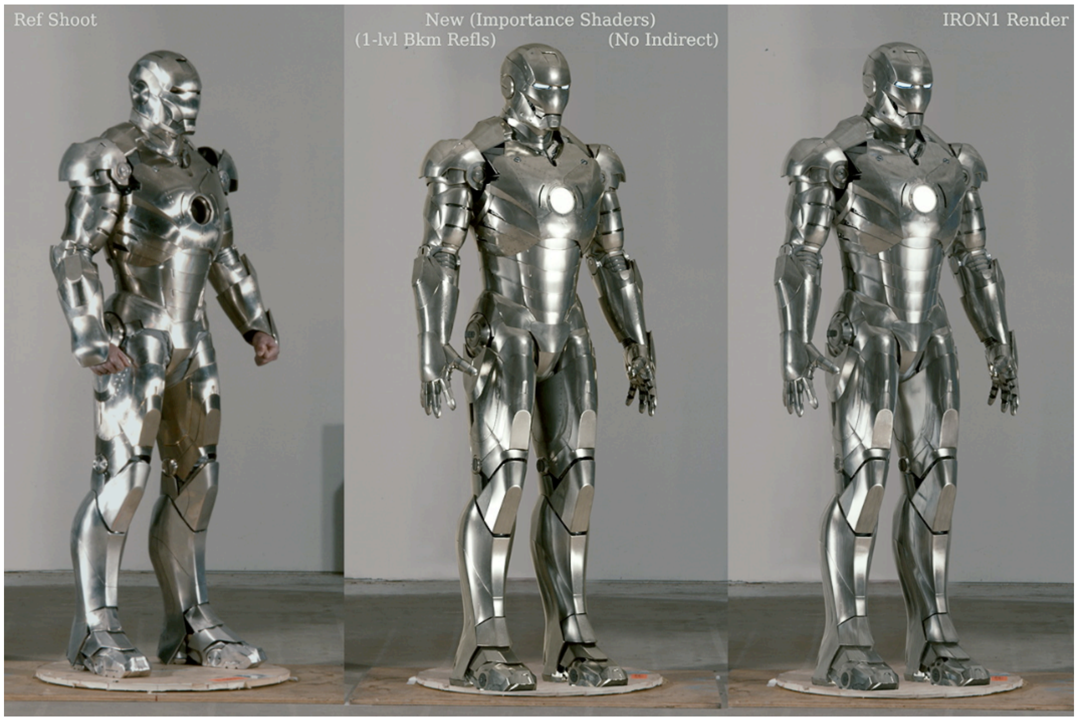


IMAGE: Iron Mark 2 comparison.

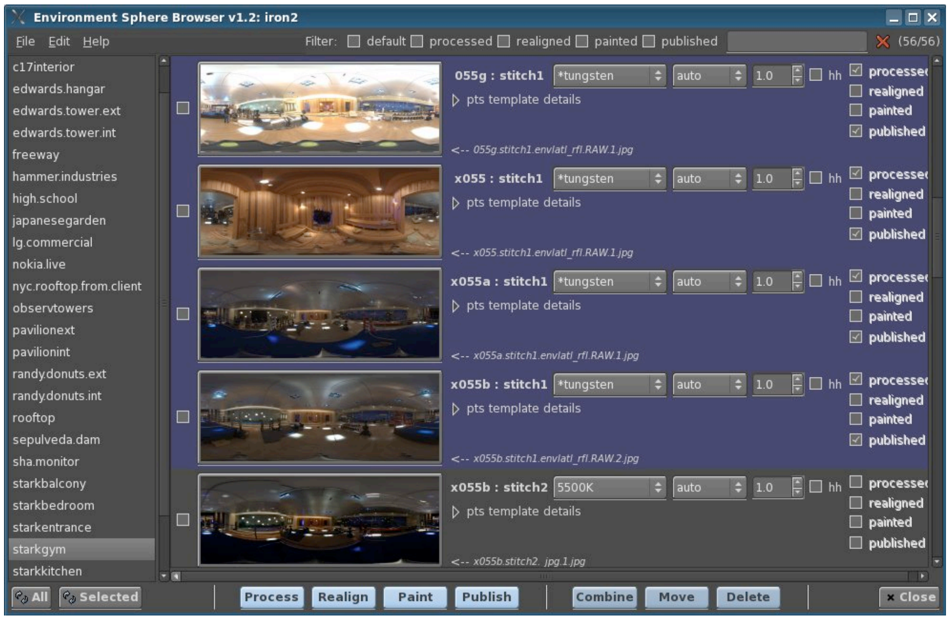
(c) 2010 MVLFFLLC. TM&(c) 2010 Marvel Entertainment. All rights reserved.

IMAGE: Iron Mark 3 comparison. (c) 2010 MVLFFLLC.TM&(c) 2010 Marvel Entertainment. All rights reserved.

New tools to make image based lighting easier

Doug Smythe and the Iron Man 2 team created a set of tools and procedures to create a smooth end-to-end image based lighting pipeline for Iron Man 2.

The first was a tool to organize, stitch and publish the HDR environment sphere images.

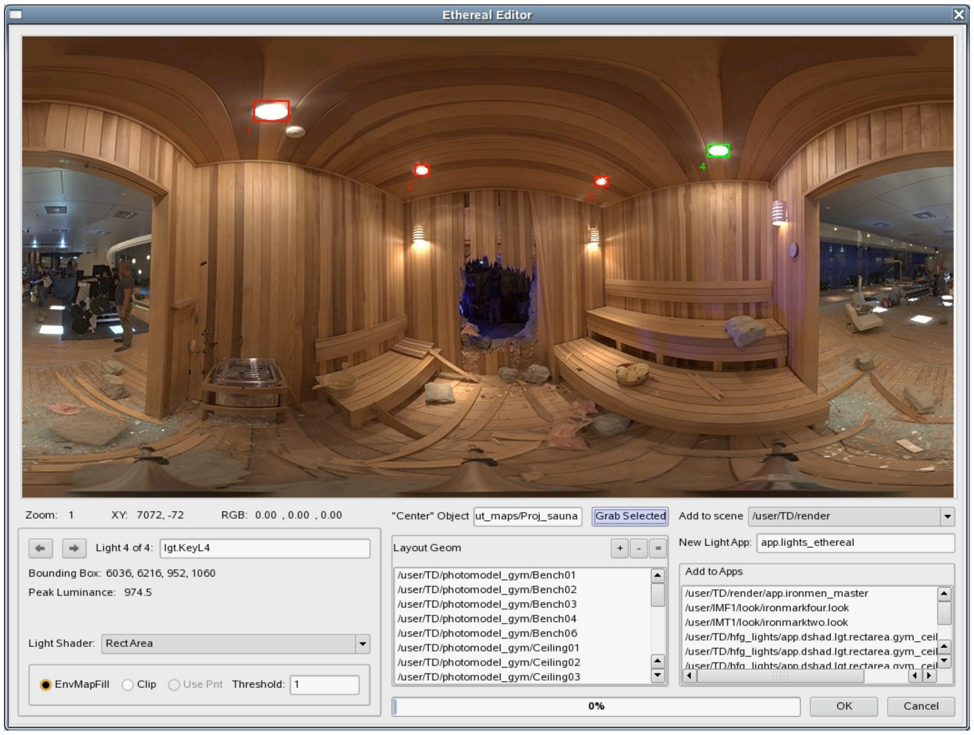

Environment Sphere Browser v1.2: iron2
File Edit Help
Filter: default processed realigned painted published (56/56)
All Selected Process Realign Paint Publish Combine Move Delete Close

c17interior
edwards.hangar
edwards.tower.ext
edwards.tower.int
freeway
hammer.industries
high.school
japanesegarden
lg.commercial
nokia.live
nyc.rooftop.from.client
observtowers
pavilionext
pavilionint
randydonuts.ext
randydonuts.int
rooftop
sepulveda.dam
sha.monitor
starkbalcony
starkbedroom
starkentrance
starkgym
starkkitchen

055g : stitch1	*tungsten	auto	1.0	hh	☑ processed ☐ realigned ☐ painted ☑ published	<-- 055g.stitch1.envlatl_rfl.RAW.1.jpg
x055 : stitch1	*tungsten	auto	1.0	hh	☑ processed ☐ realigned ☐ painted ☑ published	<-- x055.stitch1.envlatl_rfl.RAW.1.jpg
x055a : stitch1	*tungsten	auto	1.0	hh	☑ processed ☐ realigned ☐ painted ☑ published	<-- x055a.stitch1.envlatl_rfl.RAW.1.jpg
x055b : stitch1	*tungsten	auto	1.0	hh	☑ processed ☐ realigned ☐ painted ☑ published	<-- x055b.stitch1.envlatl_rfl.RAW.2.jpg
x055b : stitch2	5500K	auto	1.0	hh	☐ processed ☐ realigned ☐ painted ☐ published	<-- x055b.stitch2. jpg.1.jpg

The Environments Browser. (c) 2010 Industrial light and Magic. All rights reserved.

Once stitched we used another internal software tool which we updated to load the env sphere image and semi-automatically create a lighting rig which included textured area lights created for bright areas in the image.



Zoom: 1 XY: 7072, -72 RGB: 0.00 , 0.00 , 0.00

"Center" Object: ut_maps/Proj_sauna

Add to scene: /user/TD/render

Light 4 of 4: lgt.KeyL4

Bounding Box: 6036, 6216, 952, 1060

Peak Luminance: 974.5

Light Shader: RectArea

[✓ EnvMapFill] Clip Use Pnt Threshold: 1

Layout Geom

```
/user/TD/photomodel_gym/Bench01
/user/TD/photomodel_gym/Bench02
/user/TD/photomodel_gym/Bench03
/user/TD/photomodel_gym/Bench04
/user/TD/photomodel_gym/Bench06
/user/TD/photomodel_gym/Ceiling01
/user/TD/photomodel_gym/Ceiling02
/user/TD/photomodel_gym/Ceiling03
```

New Light App: app.lights_ethereal

Add to Apps

```
/user/TD/render/app.ironmen_master
/user/IMF1/look/ironmarkfour.look
/user/IMT1/look/ironmarktwo.look
/user/TD/hfg_lights/app.dshad.lgt.rectarea.gym_ceil
/user/TD/hfg_lights/app.dshad.lgt.rectarea.gym_ceil
/user/TD/hfg_lights/app.dshad.lgt.rectarea.gym_ceil
```

0%

The area light extraction/light rig creation tool. Squares represent lights tagged for extraction. (c) 2010 Industrial Light & Magic. All rights reserved.

The artist can manually tag the lights they want removed from the environment map and recreated as area lights. Our matchmovers modeled the whole of the sets in rough geometries, and position the HDR env sphere was photographed from. The light rig tool traced a positon out onto the set to get a rough light position and scale for the area light.

Finally a tool was created to create local environment sphere textures and simultaneously output data in the form of pointclouds or brickmaps, or create textured area light representing geometry that the CG character interacts with so indirect bounce and reflections are locally accurate.   Shadowing and local illumination effects from other CG characters could be included in these passes.

Matching the lights that they're using on set

Some of the scenes in Iron Man 2 were lit with the practical suits on set and with the lights were set up by the D.P. with the suits in mind. While this is an ideal case, and certainly such practical references should be encouraged where possible, in Iron Man 2 there were circumstances where the set lighting wasn't perfectly balanced for the suits. In this case we were able to look at times where the D.P. had lit the suits and use images of those lights as lighting instruments in our other scenes.

An example was a small Kino light he'd use to get kicks off the practical suits. It had one warm and one cooler Kino Flo tube in a reflector box a couple of feet wide. We created a CG version of this based on an image of the light to use in scenes like the end battle to add kicks on the suits.

Issues with image based.

In general the new image based tools worked out very well indeed. It certainly made it faster for artists to get a shot looking good pretty much out of the box, and allowed us to focus on the more creative lighting challenges for the show. The image based tools worked particularly well for the daylight exteriors on Terminator Salvation, and for the dynamic factory environment where the lighting was driven by pyro events that we could reproduce on cards. Combined with the normalized importance materials, the image-based approach definitely gives us more realistic results earlier in the rendering process than we've had before. However, there are some issues involved in their use.

- There is a learning curve for artists used to the traditional approaches. In lighting a shot I did for Iron Man 2 I found the new approach to be terrific. I was able to cut the lighting instruments out of the HDR environment capture instruments, position them in the right places and effectively start with a scene in my computer that matched with what I'd seen on set. But I've done a lot of set work and was familiar with the tools used. Some of the artists took to the new setup well, others found adapting to losing some of their previous cheats a little harder to adapt to.
- The better your image collection on set, the easier the path to real-looking renders. We shot environment spheres for most shots (tips on the way to shoot these is below), but in addition I recommend shooting images of the lights themselves used.
- When editing the HDR environment images, make sure you don't lose dynamic range.
- Beware the infinite nature of the lights coming from an environment sphere. If you use an HDRI sphere and a character isn't centered in the location the sphere was shot from for the whole shot, the reflections and illumination can start to play up. That's because the light intensities were correct for that spot, but the light images, being infinitely far away if you're just using the sphere, will remain the same wherever the character is positioned. This was a problem in the kitchen for the House Fight in Iron Man 2, where there were a bunch of recessed lights in the ceiling. These looked great on the practical suit reference pieces we were able to shoot on set, but in our CG environment spheres our lights were not, of course recessed, so when a character moved away from the light he was centered under he was still being illuminated by that light to the same intensity. Using a more hand-built combination of lights and reflectors as we'd done for the comparable Tony's workshop on Iron Man 1 was ultimately a bit more successful. In these cases you can recreate the lighting more correctly in the computer to match the scene rather than relying on the sphere alone. Using projections onto geometry can help as well, but, again, you're limited by the infinite nature of the lights in the source material. In a couple of shots from the Kitchen Fight of Iron Man 2 we actually painted out a bunch of the lights on camera.
- Don't forget the aesthetics. Its great to have your Iron Man look more real out of the box, but remember, the box may not always be lit with him in mind, not with him doing this particular

action. Use cards and reflectors and flags like they do in the real world to shadow offensive lights in the environment or bounce more light into the scene.

Practical tools for recording the lighting in the real (filmed) world

Filmed references

When shooting the scene, even with us using high dynamic range images as our most important tool for capturing information about the lighting and environment, it's still important to shoot visual effects references.

"Setiquette" and relationships on set.

Film shoots are fast and furious, but the crews prefer clear decisions, consistency and predicatability. You need to make sure you have relationships with the right people to get you the information you need, and you want to get people used to what you're going to need to document you environment. Your data gathering crew has to be fast and ready to move as soon as the opportunity to shoot references presents itself.

As a vfx supervisor, I usually communicate with the director and director of photography. However, the most important relationship for getting you the references to help you recreate the reality of the lighting on set is with the first assistant director and their team. You should make sure they're aware of what passes you need as soon as you can. The camera team, particularly the camera operator and first a.c. are important – you'll often want the camera operator to provide a "human motion control" reproduction of his camera moves for a clean plate, and you'll need the a.c.'s help to stay on top of camera information – lens, focal distance, and stereo information etc. Likewise the grip crew who control the cranes and dollies and who can save your butt with a piece of bluescreen thrown up on short notice. The gaffer and his crew are also people you want as allies – you want them to keep the lights and reflectors and diffusers in place for shooting your references and HDRIs so the scene matches what it was. The script supervisor will often keep track of camera information as well, and you can help each other keep track of information. You can keep the script supervisor informed of why on earth they're shooting all these passes which can help you down the track as editorial tries to sort out what to send.

The film set is a complicated place, with an odd sort of hierarchy. You and your crew need to be sensitive and away of what is going on, to stay out of the way as much you can, but to make sure you get the references you need.

What references to get

You certainly need an HDRI that is relevant to the lighting of the scene, and I'll talk a bit more about that in a moment. Beyond (and often before) that, I recommend you also capture the traditional light probe references. Spheres, a Macbeth chart possibly, and ideally some reference related to what you're going to be creating.

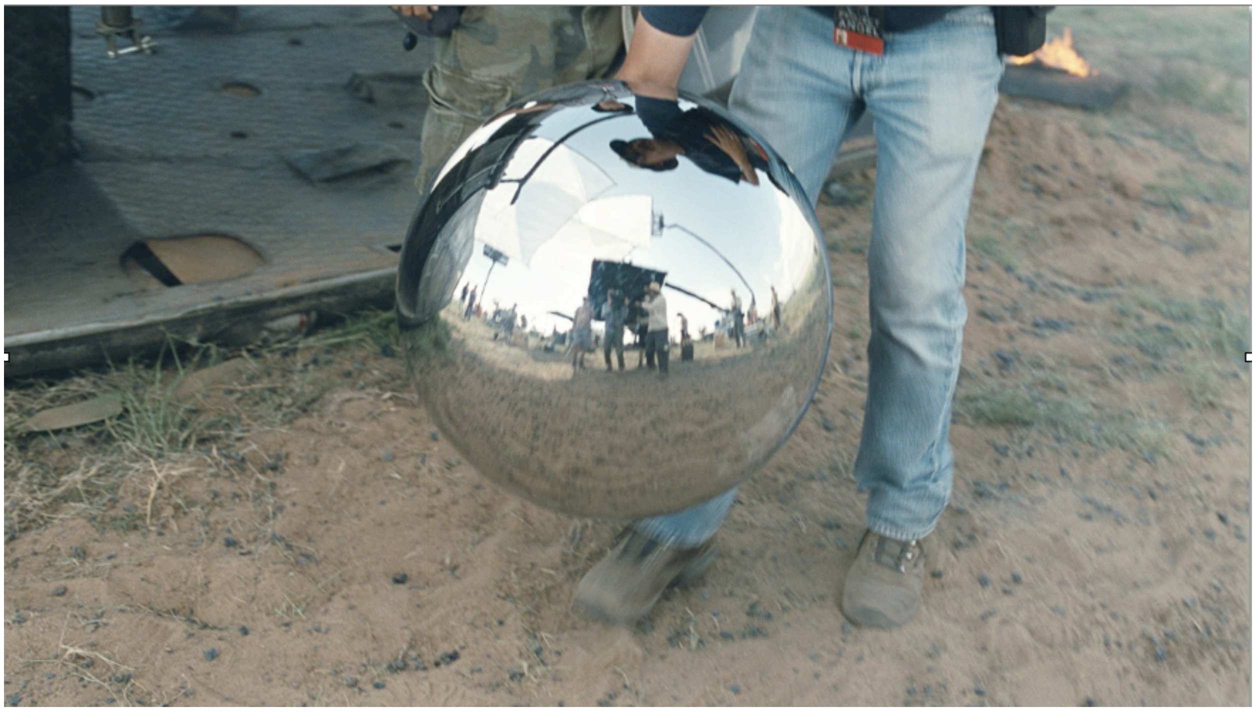
Chrome/Gray Spheres

A couple of years back I had a very talented visual effects supervisor tease me for wanting chrome and grey spheres even with shooting the high dynamic range image sets to document the scene. Its true that the HDRI is our most important reference but the chrome/grey spheres offer the following advantages:

- They are on the medium that the plate material is on, and if correctly captured and converted to your image data format, they represent the same colors as that medium. Thus, and this is the important part, you can use them to calibrate the color of you HDRI which is often shot in a different way using different equipment.
- They are a fast way of getting up to the moment information. While it might be hard to perfectly preserve the lighting, position of the performers etc. with an HDRI, you can often quickly get the chrome/grey sphere into place.
- It's insurance. In case you don't get and HDRI or the data gets messed up.
- They are moving footage. You can capture dynamic lighting setups with fire puffers, steam, strobe lights etc.
- They are familiar. More on this below.

There are several types of chrome grey spheres, and you can pay quite a lot for different ones. The one I usually use is a portable half-chrome/half-grey setup that you breaks apart for transport. Its quick and light and lets you do the job efficiently.

I've also used slightly larger separate full chrome and full grey spheres, which are good for multiple camera setups and for moving through a scene, and big ones for long lens and air to ground type scenarios.



Guidelines for shooting the chrome and grey spheres are as follows.

- Get the crew into the habit. I mentioned the importance of predictability and consistency. You want the crew to expect and indeed call for the spheres and digital stills if the shot is a visual effects shot.

- I generally ask for reference passes if I'm adding computer graphics to a shot. Even for bluescreen elements it can be relevant to shoot the spheres at least - for example where you're adding a 3D prosthetic or piece of armor to the person of object you're shooting against blue. The spheres can help in matching the bluescreen lighting to the background (hopefully you've shot the plate first and can use the sphere pass from that to help guide the lighting on the bluescreen element).

- Make sure the lighting is consistent with the plate photography. The references are not much use if the grips disassemble the reflector cards and the electric crew turn off the key light for the reference passes. Ideally everyone should hold their positions for the references as if you're shooting another take, even the actors in some cases. Don't expect a whole lot of success on the latter, but sometimes you can get in there on at least one of the setups and capture a representative HDRI and refs with performers or stand-ins. With some D.P.s this can be a nightmare because they tweak the lighting with every take. At a certain point you have to get what you can.

- Make sure the spheres are as big as possible in frame. If the camera is on a zoom, have them zoom in, but do that AFTER shooting any lens calibration references you might be using. If it's a prime – which is what we prefer for visual effects work – you might need the big spheres

- Make sure the spheres are in the right spot. It's not very useful to shoot spheres that aren't in the lighting that represents where the computer graphics will go. Of course they're not a ton of use if they're tiny in frame either. If it's a relatively consistent lighting environment (eg daylight exterior without a bunch of reflectors etc.) I use two sets – one back where the cg will go and one closer to the camera so its big in frame.

- Hold the sphere out in front of you. You don't want to see the person holding the sphere reflected in the sphere, particularly if you're going to unwrap it for an environment map.

- Don't shadow the spheres with your body – again you want the sphere to be in the lighting the added object would be in – and that probably doesn't include you.

- Make sure the lights go through their performance, ideally with the sphere and camera in a locked position. For the chrome sphere its useful to have it static so you can pull an environment map if you wish.

- Move the spheres through the space if the computer graphics object moving through the space. This is where separate spheres are handy – you can hold them out from your body and apart and offset so they don't shadow one another and the grey doesn't take too much of the real estate in the chrome reflection.

- Camera move is a judgement call. You want some moments of camera and sphere stable, particularly for a chrome sphere if you're pulling an environment from it.

- Be ready, and have your crew ready, at all times when they're shooting VFX shots. You need to get in there as soon as they announce they're moving on. If there is something non camera or lighting related that delays the shooting the A.D. might want to grab the references while waiting. This is not always ideal since things can change but you can always call to shoot them again if things change enough to justify it.
- When in doubt, shoot refs. If the director swears blind it won't be visual effects but you see how it could, its usually worth shooting the refs, particularly if by now your on-set crew is a polished machine. If not, make sure the data gatherer and script supervisor note the nearest set of references that could at least inform this take.
- Don't be sloppy – make sure you and your crew are shooting the spheres properly and that you don't just dash them off, even knowing there is some forgiveness if you're using them to just calibrate the HDRI

How it goes down on set

Before the take. If something in the take is going to adjust the set irrecoverably (eg its going to be blown to bits), you might want to shoot references and even a clean plate before they do the actual effect.

As mentioned, be ready to swoop in with references when the A.D. calls for them. Make sure you tell the A.D. you want references and ideally what they are for every setup at some time before or during the photography.

You need to keep an ear out – ideally you're at or near video village. If the director seems to be happy with the take, get ready to get your team out there with the refs. If you hear the "cut, we got it, moving on" type call, you need to swoop in. Sometimes a call like "spheres " (crews sometimes like "balls" better), "hold the lights" etc. may be necessary. If you're lucky the crew will pretty soon be calling for the spheres themselves. In general, have them shoot them for VFX shots to get into the habit. Don't try and be too nice a guy and overthink it – remember consistency, and also take care in what you shoot – so they know to take care as well.

For complex shots particularly they should be informed ahead of time, but a lot of what we do is on the fly, so you certainly need to make sure they know what you'll be asking for as soon as they start doing takes of a scene. If it's a big special effects pass or stunt, you might request a clean plate or some references before the first take that would mess up the shot, so that you can capture the environment that you'll be adding your computer graphics to in its "before" state – there is often no going back.

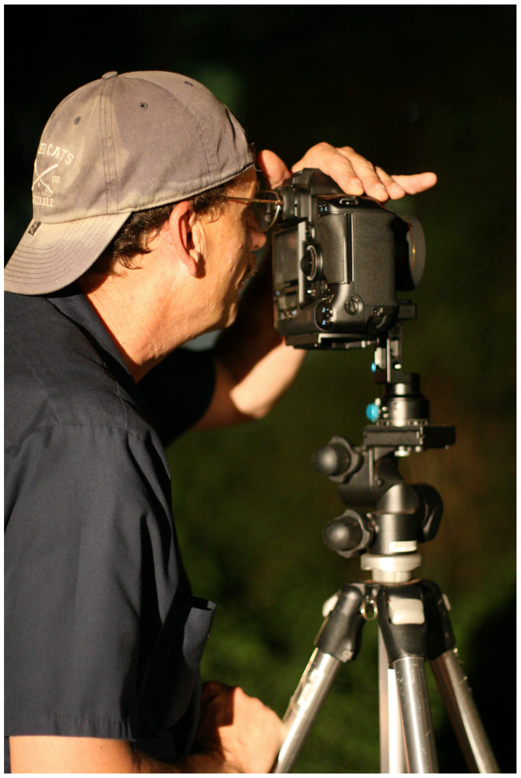
Capturing HDRIs on set.

There are several ways of capturing HDRIs on set, but we try to keep a fairly simple approach, mostly in the interests of flexibility and speed. The rules for capturing lighting references apply here.

In theory we're trying to capture the whole dynamic range of the scene, including the brightest lights including the sun. In practice we often end up replacing the brightest lights with lights we have control over, so there is some forgiveness. We've arrived at a resolution of no less than 8k around for the final stitched environment sphere

Not everyone will agree with me on this, and there are several approaches to capturing high dynamic range images on set, including some quite cool semi-automated approaches like the box device created by Hoyt Yeatman.

However, we've found that sending an on set data gatherer out with a simpler setup as described below works very well. It is reliable, flexible and, importantly fast. It is also film set friendly, and sometimes we'll be able to take an appropriate image off to one side or sneak in and grab it between takes, saving further time in the expensive day of shooting.



Recommended equipment:

Canon 1DS mk3 Camera body. This model has a full-frame sensor and with an 8mm lens you can capture a scene with three directions. Other cameras have smaller capture areas and require multiple directions to cover a full sphere.

Sigma 8mm fisheye lens

Nodal Ninja

Tripod

Remote shutter trigger for camera body.

0.6 ND (2-stop) filter for lens.

We setup the camera as shown with the 8mm fisheye lens in the Nodal Nija head mounted on the tripod. Position the lens so it is as close to nodal as possible.

We shoot in Manual exposure mode with manual focus and image stabilization off if the lens or body has it. We remove any extraneous filters or rings from the lens.

In good, not too rushed shooting conditions we'd shoot the following for a direct-sunlit exterior environment:

- 7 exposures, 3 stops separation, center exposure 1/32 sec
- Aperture f/16, ISO 100
- Add 0.6 ND (2-stop) filter to the lens (you don't want this for interior or reference shooting, 'though.

We adjust to taste in darker situations or situations where we need to move a little more quickly than this would entail (eg the director or AD is shouting at you). Aperture decreases and ISO increases are usually the first things that get changed in the heat of battle.

ACKNOWLEDGEMENTS

I'd like to thank the following people. Pat Conran, Doug Smythe and Hayden Landis all of whom generously allowed me to shamelessly crib from their various show reports, ILM internal documentation and previous courses. Pat's normalized importance documentation was particularly helpful and supplied many of the technical renders in the section on Terminator Salvation. Also Christophe Hery, Naty Hoffman and Philippe Rebours for their valuable advice and suggestions and Lanny Cermak for modelling.

Reflection Occlusion was originally developed by Stefen Fangmeir on Speed II with contributions by Barry Armor, Dan Goldman, Ken McGaugh, Will Anielewicz and Christope Hery.

Ambient environments were developed largely by Hayden Landis, Ken McGaugh and Hilmar Koch with refinements by Kevin Sprout, Dan Goldman and Doug Sutton.

Christophe Hery and Simon Premoze developed the BRDF and anisotropic tools for Iron Man and implemented much of the Normalized importance shaders and lights on the R&D side, with Pat Conran, David Meny, Philippe Rebours making big contributions on the production side.

Thanks to the ILM crews on all the above productions, and the terrific support staff and R&D team.

Finally, thanks to Siggraph organizers for providing this forum for learning and sharing and to my fellow speakers for their enthusiasm and inspiration.

Course Notes, Siggraph 2010
Los Angeles, CA, USA

Faster Photorealism in Wonderland:

*Physically based shading and lighting at
Sony Pictures Imageworks*

Presented By:
Adam Martinez, Cg Supervisor
amartinez@imageworks.com

Sony Pictures Imageworks, Culver City, CA

Biographical Information

Adam Martinez is a Computer Graphics supervisor for Sony Pictures Imageworks and a member of the Shading Department, which oversees all aspects of shader writing and production rendering at Imageworks. He is a pipeline developer, look development artist, and technical support liaison for productions at the studio and he is one of the primary architects of Imageworks' rendering strategy behind *2012* and *Alice In Wonderland*. Adam started his career in commercial post houses and animation boutiques in New York City as a freelance computer graphics artist. He began his work in film visual effects on the project "Cremaster 3" by artist-filmmaker Matthew Barney. Since then he has served as both effects and lighting technical director, cg supervisor and pipeline developer for various studios in the San Francisco Bay Area. At ESC Entertainment, Adam led the effects team in the creation of complex insect crowd simulation tools for *Constantine* and destruction effects for *Matrix:Revolutions*. As computer graphics supervisor for The Orphanage on *Superman Returns*, Adam oversaw the creation of a ballistics simulation and rendering system. At Lucas Animation Adam was both a rendering pipeline developer and cg concept artist for television and feature animation. Adam's primary interest is in simulation and the construction of complex, but highly usable, systems for dynamic effects and rendering. Adam has a BA from Rutgers University. This is his first ever Siggraph presentation.

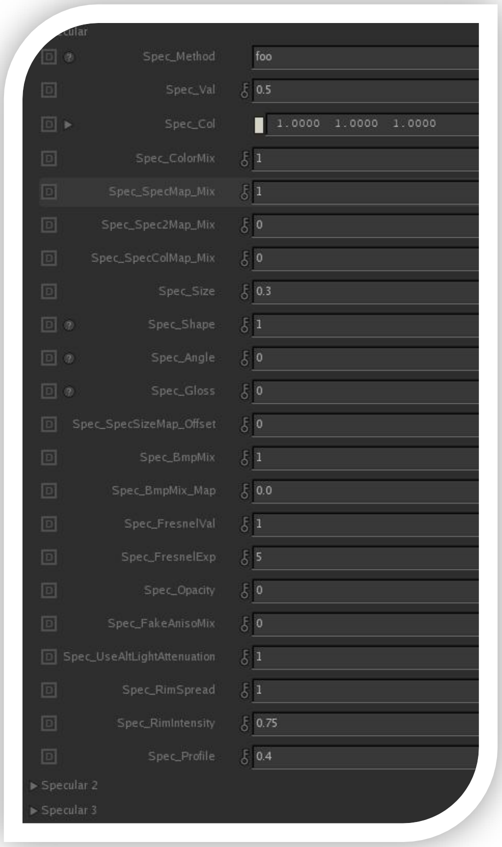
Introduction

Films such as *Cloudy with a Chance of Meatballs*, *2012* and *Alice in Wonderland* (*Alice*) have represented dramatic changes in the way Sony Pictures Imageworks produces imagery. We have moved away from the traditional biased, multi-pass rendering system to a largely single-pass, global illumination system incorporating modern ray-tracing and physically based shading techniques. We will discuss how these changes applied to our work on *Alice*, and look at specific examples from the film. Topics discussed will include: motivations behind moving to a physically based rendering system and how such motivations would ideally manifest, micro-facet illumination models, global illumination and layered materials. We will talk about the impact of these developments on our creative approach and lighting procedures in production, as well as some of the unintended consequences of these changes.

Recent History

For projects as recent as *Watchmen*, Sony Pictures Imageworks employed a rendering system that does not natively account for global illumination effects. This system was heavily pass-dependant: shadows were depth-mapped, color bleeding came from point clouds, and reflection or environment occlusion was rendered in a separate pass, often in a separate renderer. Biased techniques such as shadow map and point cloud generation meant smaller sub-pipelines had to be developed and maintained. These techniques are largely phenomenological and physically inaccurate. Fidelity of the final result was subject to the quality of the passes that were generated previously. In addition, there was an enormous potential for version mismatches between separate passes. Even so, all of these entities had to be in place before anything approaching a photorealistic render could be accomplished.

These auxiliary pipelines required significant development and maintenance depending on the needs of a particular show or sequence. In addition, the amount of data generated per frame was so large that disk storage and synchronization of elements became critical aspects to the production. If any single shadow or occlusion pass was not in sync with the latest animation or light rig, it meant a lot of work hours tracking down the source of the divergence, and a lot of machine time re-generating the data. In the worst possible cases, a user would just be resigned to regenerating all the data on every update.

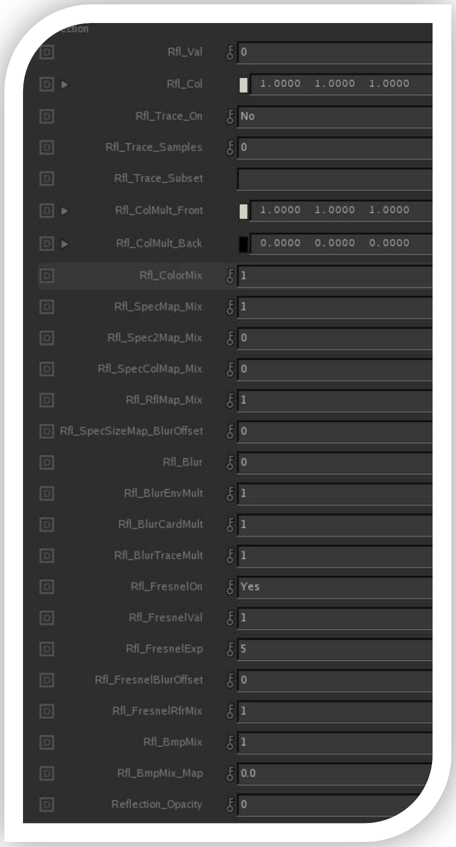


cular

Spec_Method	foo
Spec_Val	0.5
Spec_Col	1.0000 1.0000 1.0000
Spec_ColorMix	1
Spec_SpecMap_Mix	1
Spec_Spec2Map_Mix	0
Spec_SpecColMap_Mix	0
Spec_Size	0.3
Spec_Shape	1
Spec_Angle	0
Spec_Gloss	0
Spec_SpecSizeMap_Offset	0
Spec_BmpMix	1
Spec_BmpMix_Map	0.0
Spec_FresnelVal	1
Spec_FresnelExp	5
Spec_Opacity	0
Spec_FakeAnisoMix	0
Spec_UseAltLightAttenuation	1
Spec_RimSpread	1
Spec_RimIntensity	0.75
Spec_Profile	0.4

▶ Specular 2

▶ Specular 3



ection

Rfl_Val	0
Rfl_Col	1.0000 1.0000 1.0000
Rfl_Trace_On	No
Rfl_Trace_Samples	0
Rfl_Trace_Subset	
Rfl_ColMult_Front	1.0000 1.0000 1.0000
Rfl_ColMult_Back	0.0000 0.0000 0.0000
Rfl_ColorMix	1
Rfl_SpecMap_Mix	1
Rfl_Spec2Map_Mix	0
Rfl_SpecColMap_Mix	0
Rfl_RflMap_Mix	1
Rfl_SpecSizeMap_BlurOffset	0
Rfl_Blur	0
Rfl_BlurEnvMult	1
Rfl_BlurCardMult	1
Rfl_BlurTraceMult	1
Rfl_FresnelOn	Yes
Rfl_FresnelVal	1
Rfl_FresnelExp	5
Rfl_FresnelBlurOffset	0
Rfl_FresnelRfrMix	1
Rfl_BmpMix	1
Rfl_BmpMix_Map	0.0
Reflection_Opacity	0

All of our shaders were based on the traditional phenomenological "Phong" model, and variations. There was no relationship between the direct illumination specular response and reflection response of the shader. Parameters for color, value and Fresnel response were replicated multiple times, meaning that the same phenomenon was essentially being dialed twice.

This shading system suffered from "feature creep" and near over-development. Parameter names adhered to very CG specific terminology there was very little notion of real world phenomena in the user interface. Terms such as "Specular Size" and "Specular Angle", have no real-world meaning and require a certain amount of education and familiarity to dial successfully.

Massive numbers of parameters and a monolithic shading architecture meant that users had only one recourse for feature implementation or experimentation; the shader developer. This also meant that productions mutated these shaders independently, which led to a certain amount of divergence across shows, and a potential loss of technology when a show wrapped.

Substantial interface real estate and shader development was devoted to simulating area lights in a rendering system that did not natively support the notion. Most of these methods fell apart when it came to complex lighting schemes, and would require significant dialing from both the surface material and the light material to get a passable effect.

Arnold Renderer

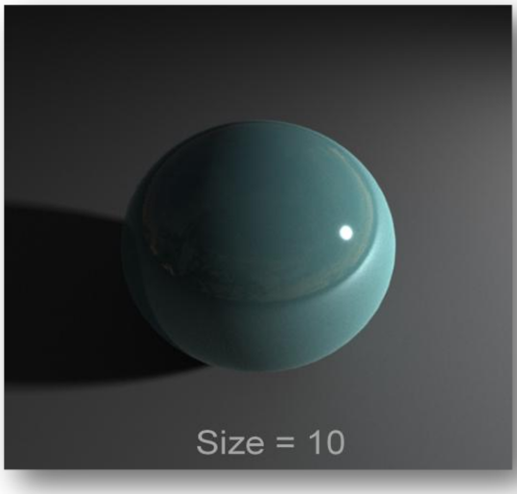
Arnold is Imageworks internal renderer, co-developed with Solid Angle SL. It is a modern global illumination ray tracer with a robust shading API and flexible scene format. For most productions, Arnold was deployed as a utility renderer; generating ray traced occlusion passes for use in compositing. On a few productions it had seen use as the primary rendering engine; Monster House and Cloudy are both full CG animated productions. Cloudy was the driving force behind shader and renderer development for Arnold which made it a much more plausible option for visual effects films. Eagle Eye used the shading system developed for Cloudy, but it was *2012* that really pushed for a new shading system that offered faster photorealism.

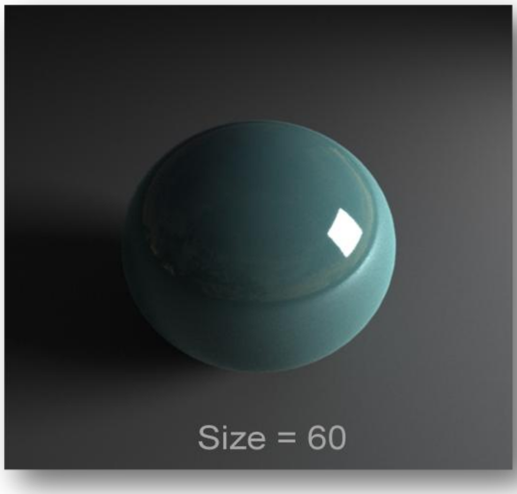
At the same time that *Alice* and *2012* were beginning to evaluate the rendering options, there was a lot of discussion throughout the facility about development philosophy in general. Rob Bredow, Imageworks' CTO, made it clear that the push towards using Arnold, and technologies like it, was to provide more photo-realistic imagery out of the gate, but to also make our human time more productive. This was highly informed by his experience supervising Cloudy, where he witnessed Arnold's capacity for consistency in shot production, provided the requisite time was spent in asset development. This idea coincided with another facility mandate to take interface simplification very seriously. This idea was intended to provide a framework for reducing interface clutter and, ideally, making is so that a multiple doctorate was not required to shade a frog. The task was put before us in the shading group to develop a system that was human readable, intuitive to use, and deployed at the facility level. To top it all off, we had to come up with strategies and programs to educate users on how to use these new tools.

For *Alice* specifically, there was a strong desire to leverage the power of the ray tracer to implement a physically based shading system, but also to expand the capabilities of the shading system during the look development stage. Previously, realism was difficult to control and maintain in the face of creative demands. Supervisors wanted a system that would allow them to start the creative process sooner, and maintain a consistent level of realism.

2012 and *Alice* committed to using Arnold, and a yet-to-be developed shading system. This was going to be no small chore. Developing shaders for the renderer was a known quantity, but how to deploy them and what form they would take to the end user was a major question. Other issues we needed to address included the pipeline infrastructure required to handle the predicted volume of data going through the renderer. This also made us sharply aware that our render times were going to be significant, and we knew we were going to have to balance physical accuracy with reasonable times.

Part of the development strategy was informed by the functions of the renderer and how switching to it inherently changed the pipeline and production approach. Lack of maps or need for occlusion passes did away with all of the previous secondary pipelines, and their associated maintenance costs. Light sources in Arnold are implemented as area lights natively, which means no extra development in the shader is required to support them.





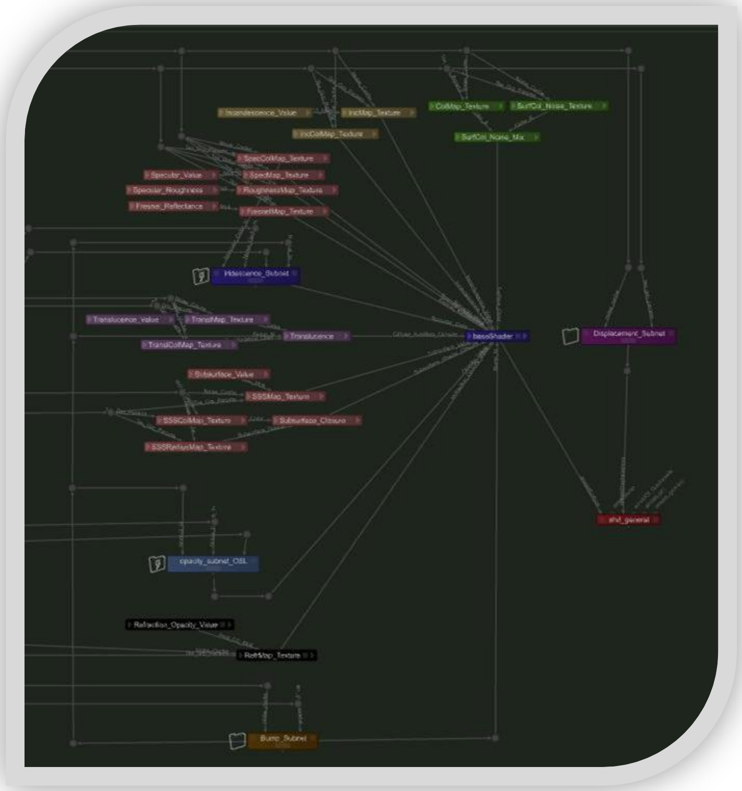
This illustration demonstrates the effect of area light size on a scene. The only variable changes was the size of the light source. On the left, you can see the change in the size of the specular highlight, the softness of the shadow and the diffuse falloff, or "wrap" around the surface normal terminator. The shader has no parameters to control this behavior, it is purely a function of the light source and Arnold's sampling engine. This not only demonstrates a key feature of the lighting pipeline, but is also a great example of interface simplification.

While global illumination is accomplished by indirect sampling means, point clouds weren't entirely extinguished. Arnold has a built-in point based subsurface scattering implementation, but this is entirely generated by the renderer inline to the process and uses the same scene information as the beauty render. In addition, the subsurface effects is informed by and reflected in the global illumination solution by default. As a quick side-note, we did implement our own traced sss used extensively throughout the film, based on Henrik Wann-Jensen's 2001 paper, *A Practical Model for Subsurface Light Transport*.

We also implemented a photon mapped caustics solution which was initially developed for the Watchmen, glass palace sequence. This was based on the methods described in Jensen's 2004 course *A Practical Guide to Global Illumination Using Photon Mapping*. This system followed a similar path in that the photon map was generated at beauty render-time with the actual scene data. Energy is emitted by light sources into the scene and is either reflected, transmitted or absorbed. This energy is stored in a point cloud representation of the photons and then referenced during the illumination stage of the beauty render.

The success of Arnold's global illumination effects depend on the exchange of energy between surfaces. The renderer developers and the shading department both encouraged rendering as much geometry in camera as possible, and only splitting out passes when required by compositing needs. Various memory management strategies built into the renderer helped with this, with minimal user intervention required for special circumstances.

Because Arnold is a stochastic ray-tracer, implementing real-world lighting tools such as bounce cards was a relatively simple matter. We advocated the use of bounce cards and even incandescent emissive geometry in scenes, over the use of special purpose CG light sources. As a result of the Arnold renderer's robust HDRI sampling, most, if not all sequence lighting started out with a sky dome, which was an image mapped sphere encompassing the entire scene. This special purpose entity was the primary source of mood and context for a particular sequence, and usually stayed constant for each shot.



Designing the Shading System

The shading department made some initial design decisions that had far-reaching implications for how we worked as shader developers, and how we maintained the shading system for productions. The decision to use a node based network for shading was motivated primarily by these maintenance concerns. A networked shader can be portioned out to different developers, parts of it can be repurposed, and nodes can be developed once, and re-used in many places.

In the illustration above you can see a portion of our general shading network. The shading system centered around a root illumination node with encapsulated all of the basic lighting procedures. Procedural and map based texture sub-networks feed this root node with parameter information, such as surface color, specular roughness etc.

When it came to establishing default values, the motivating factor was to give users a more accurate image. This meant enabling reflections, global illumination up front, and for optional shading components supplying sensible and realistic default values without causing a dramatic slowdown in turn around.

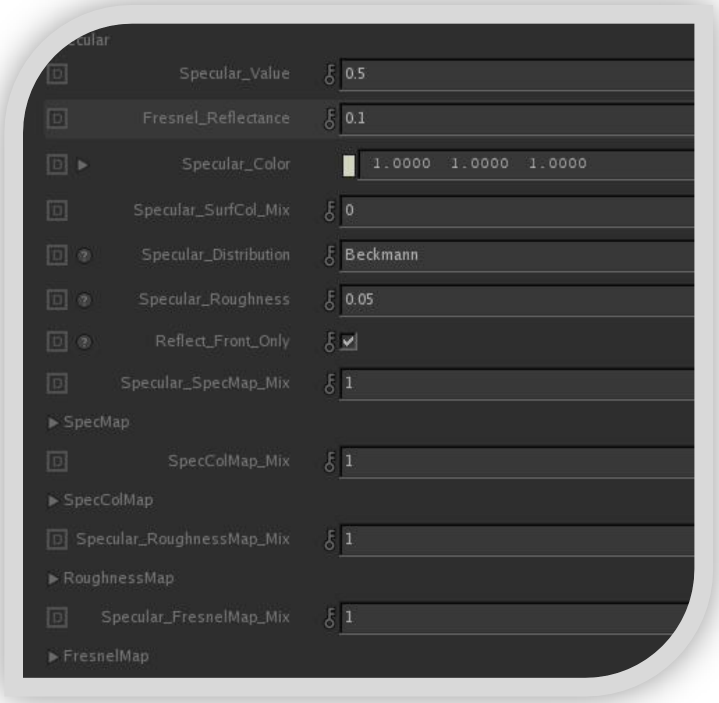
By far the most critical aspect was to involve the users on *2012* and *Alice* from the beginning of the design process. This level of involvement was an opportunity for two-way communication between the developers and the artists. Artists gained early familiarity with the tools as they were being developed and regular feedback and feature requests ensured minimal wasted effort on the development side.

Implementation Details

Much of the information in this section is explained in much greater detail by the source material. Our implementations followed the source material fairly accurately.

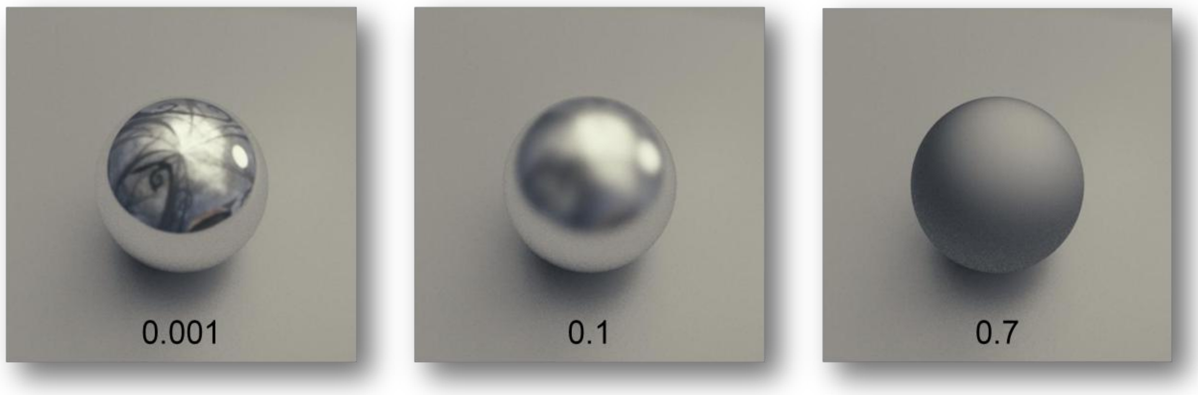
For general specular response our shaders employ the Cook-Torrance microfacet functions from Bruce Walter and co. 2007 paper *Microfacet Models for Refraction Through Rough Surfaces*. In this paper the authors describe a flexible bsdf architecture with brdf, sampling and weighting functions for multiple distributions in both reflective and transmissive rendering contexts.

In [Walter07], the reflection term is described by equation 20. It is defined as a Fresnel term, F, a shadow masking term G and a microfacet distribution term D. In our implementation, F is the Schlick approximation. G and D are variable depending on the distribution selected. Equations 25 and 27 describe the Beckmann distribution, which is also our default distribution in the shading system at Imageworks. G is a rational approximation of a gaussian distribution of microfacet normals. Equations 28 and 29 are the sampling equations to generate microsurface normals from random number pairs in a Monte-Carlo sampling scheme.

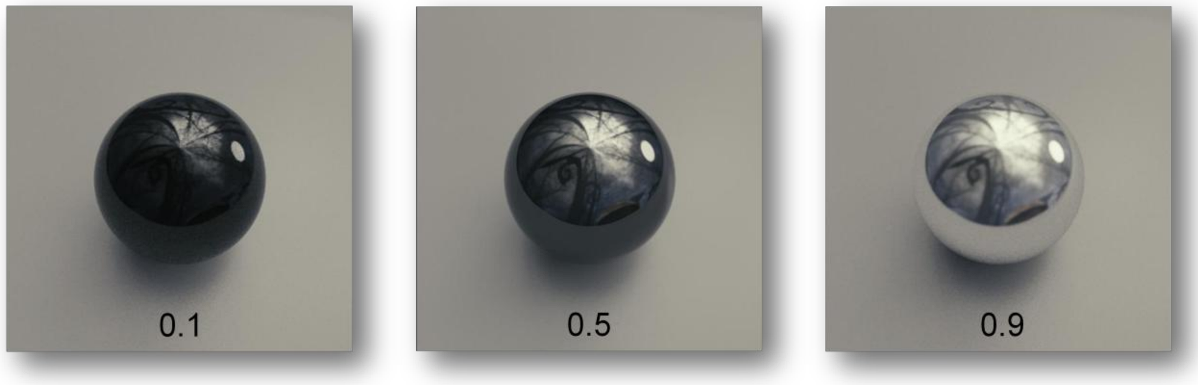

cular

Specular_Value	0.5
Fresnel_Reflectance	0.1
Specular_Color	1.0000 1.0000 1.0000
Specular_SurfCol_Mix	0
Specular_Distribution	Beckmann
Specular_Roughness	0.05
Reflect_Front_Only	✓
Specular_SpecMap_Mix	1
▸ SpecMap	
SpecColMap_Mix	1
▸ SpecColMap	
Specular_RoughnessMap_Mix	1
▸ RoughnessMap	
Specular_FresnelMap_Mix	1
▸ FresnelMap	

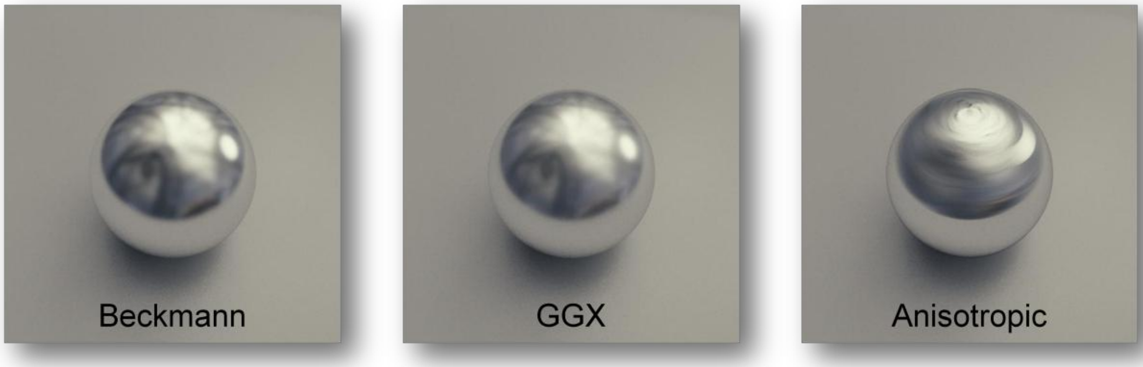
The parameterization of these functions is deceptively simple: roughness and index of refraction. And yet the model controls direct specular reflections from cg sources, indirect sampled reflections and refractions. What we get from this is a common appearance of surface roughness between all three of these contexts. This contributes greatly to an overall physically plausible look, and we realize a significantly reduced parameter set over our old shaders. In addition the term “roughness” is a commonly used term to describe real world surfaces, and it translates quite well into our system. Frensel Reflectance is a bit more obscure: we use it to describe what is actually the index of refraction, or eta in the Schlick approximation, (the Fresnel term), of the previous equations.



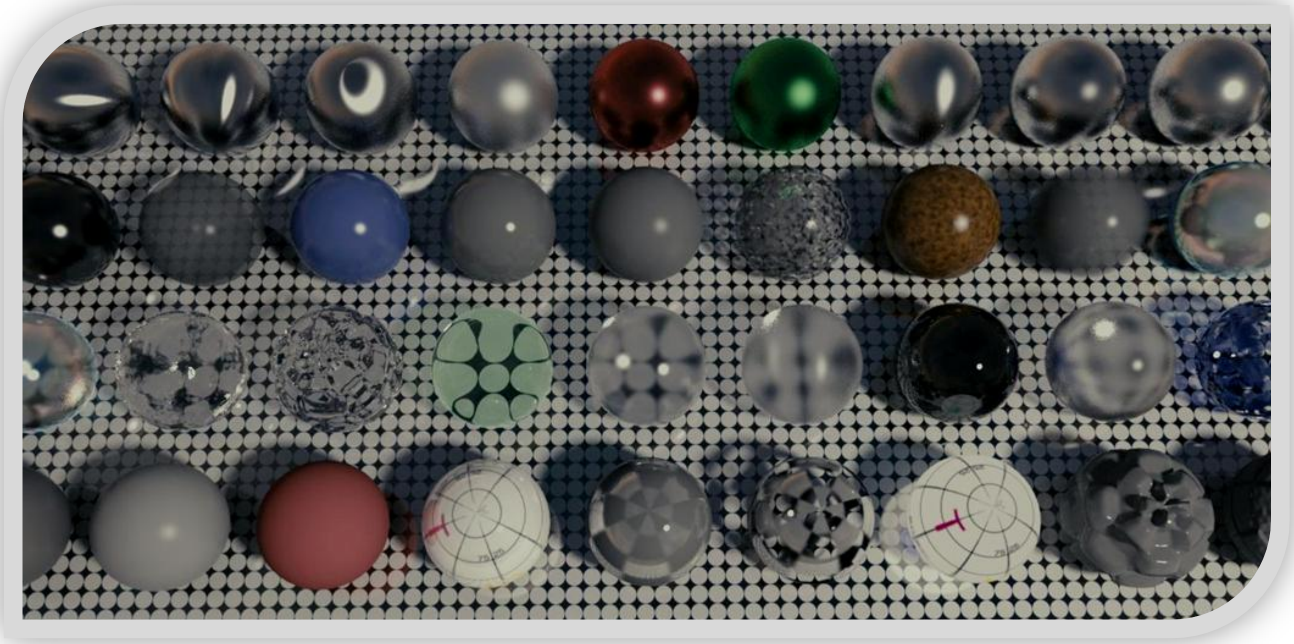
Roughness is a normalized parameter from 0-1 and controls the spread, or “blurriness” of the direct and indirect reflections. At high values it approaches a perfect diffuser, but resolving this requires a significant number of samples.



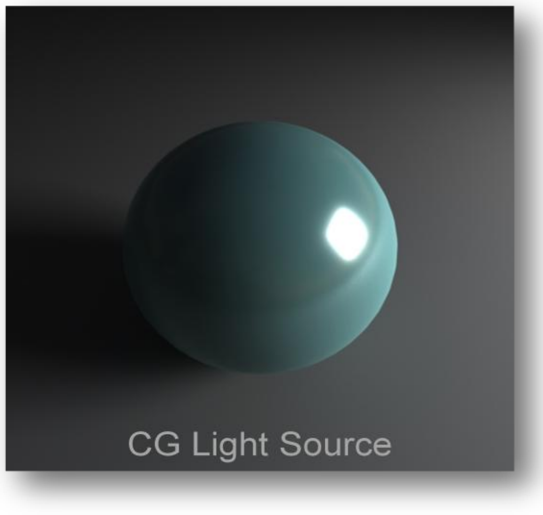
The fresnel response is also a normalized parameter from 0-1. At low values, around 0.02, the material behaves as a perfect dielectric, like plastic. At 1, the material is a near-perfect conductor, or metallic surface. In our implementation we used the Schlick approximation for normal-incidence reflectance.

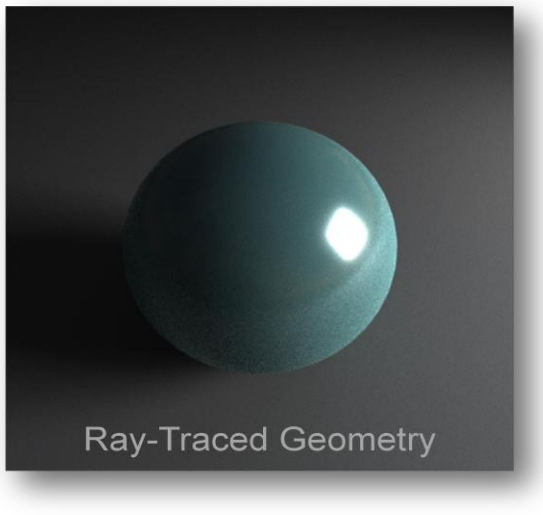


These are three examples of the most commonly used specular distributions in the shading system. Beckmann is the default and is widely used for most materials. GGX is a distribution introduced in the Walter paper and we found that is samples slightly better at very low roughness settings, which makes it a good candidate for corneas and other wet surfaces. The anisotropic distribution is based off of the work of Ashikhminh-Shirley and is the primary specular model used for metal surfaces on *Alice*.



This illustration shows our standard shader network with a variety of parameter settings. This image is used to test shader behavior as new versions are released. You can see a variety of distributions and settings here including anisotropic reflections, with various patterns, as well as anisotropic refractions. In our implementation we consciously decided to decouple the index of refraction of the reflection component from that of the refraction component for creative reasons. Absolem, the caterpillar is an example of this use. The refractions of his eye are raytraced, but the effect of the eyeball filling the lens had to be dialed separately from the glassy reflections at the surface interface.

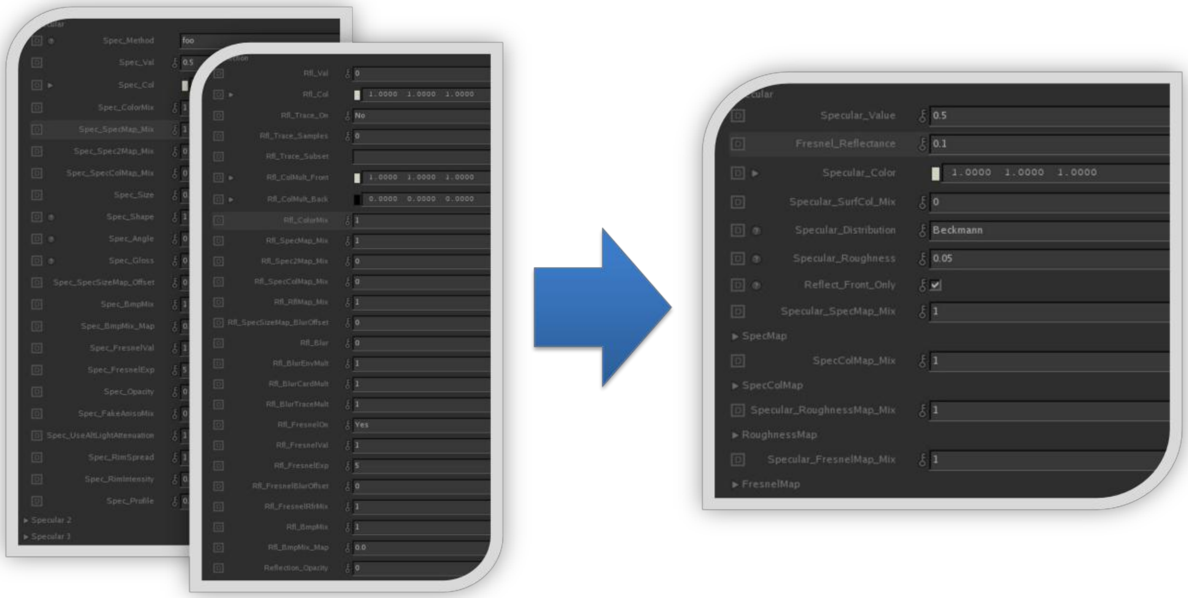




This illustration demonstrates the coherency of our implementation across direct and indirect illumination sampling methods. On the right is the sphere rendered with a cg light source reflected using the just the shading model. On the left is a card with an emissive material, it is being traced in the reflections using the sample distribution.   This behavior makes the architecture very flexible in production by allowing artists to substitute lights for geometry and vice versa as the need may arise, and get a predictable result.

Development Successes

In summary, the implementation of a microfacet BSDF architecture provided numerous benefits to the shading system: We realize visual coherence between the direct and indirect reflection methods as well as consistent behavior from refractions, and our shaders provide realistic response that is consistent across all lighting conditions. Since we combine the specular illumination and reflection terms into a single interface, and by virtue of the renderer's native behavior, we also accomplished substantial interface simplification and a streamlined the user experience.



Spec_Method	foo
Spec_Val	0.5
Spec_Col	
Spec_ColorMix	1
Spec_SpecMap_Mix	1
Spec_Spec2Map_Mix	0
Spec_SpecColMap_Mix	0
Spec_Size	0
Spec_Shape	1
Spec_Angle	0
Spec_Gloss	0
Spec_SpecSizeMap_Offset	0
Spec_BmpMix	1
Spec_BmpMix_Map	0.
Spec_FresnelVal	1
Spec_FresnelExp	5
Spec_Opacity	0
Spec_FakeAnisoMix	0
Spec_UseAltLightAttenuation	1
Spec_RimSpread	1
Spec_RimIntensity	0.
Spec_Profile	0.

▸ Specular 2
▸ Specular 3

Rfl_Val	0
Rfl_Col	1.0000 1.0000 1.0000
Rfl_Trace_On	No
Rfl_Trace_Samples	0
Rfl_Trace_Subset	
Rfl_ColMult_Front	1.0000 1.0000 1.0000
Rfl_ColMult_Back	0.0000 0.0000 0.0000
Rfl_ColorMix	1
Rfl_SpecMap_Mix	1
Rfl_Spec2Map_Mix	0
Rfl_SpecColMap_Mix	0
Rfl_RflMap_Mix	1
Rfl_SpecSizeMap_BlurOffset	0
Rfl_Blur	0
Rfl_BlurEnvMult	1
Rfl_BlurCardMult	1
Rfl_BlurTraceMult	1
Rfl_FresnelOn	Yes
Rfl_FresnelVal	1
Rfl_FresnelExp	5
Rfl_FresnelBlurOffset	0
Rfl_FresnelRfrMix	1
Rfl_BmpMix	1
Rfl_BmpMix_Map	0.0
Reflection_Opacity	0

Specular_Value	0.5
Fresnel_Reflectance	0.1
Specular_Color	1.0000 1.0000 1.0000
Specular_SurfCol_Mix	0
Specular_Distribution	Beckmann
Specular_Roughness	0.05
Reflect_Front_Only	✓
Specular_SpecMap_Mix	1

▸ SpecMap

SpecColMap_Mix	1

▸ SpecColMap

Specular_RoughnessMap_Mix	1

▸ RoughnessMap

Specular_FresnelMap_Mix	1

▸ FresnelMap

Dozens of parameters were reduced to a few parameters that appropriately affected multiple components. The user should not have to adjust any other settings to get the same response from direct light sources and traced reflections. Dialing the simpler shader is significantly easier, but didn't we lose some flexibility? How did this change affect other departments in the color and lighting pipeline? There was certainly concerns regarding the combination of illumination and traced reflections, and the impact across many of the departments was significant.

With the simplified ui came a much smaller learning curve on how to use the shaders, but more development was needed to accomplish complex hybrid looks that *Alice* would require. It required a shift in the thinking of our look development artists from composing a highlight on a surface, to critical thinking of the nature of the surface itself. Early on in the process I would advise artists to ask themselves: How would this surface feel to the touch? Smooth or rough? Is this material more like plastic, or glass? Stone or metal? Answering these questions, one could arrive at initial values for roughness and Fresnel reflectance and then go from there. Alternatively, we had contextual help dialogs that listed example settings.

In the technology arena, we had to design tools that would allow the artists to create complex looking materials without having to develop a new shader every time a feature needed to be added. Our node based network shading system allowed us develop a straight forward material layering scheme. Material layering has been present in commercial software for many years. However Imageworks traditionally avoided such schemes due to consistency and efficiency concerns, not for lack of trying. Our approach leveraged a simple interface that stacked shaders in an intuitive way.

Material Layering

Material layering allowed for much greater flexibility in the shading system and room for experimentation with hybrid materials. Red knight for example is made up of dirt over black paint, over red paint, over metal. Each layer has very particular settings based on look desires. The presence of each layer is determined by an opacity control map. Most layers can make use of the same maps. The opacity mask for the red paint in this example, is also one of the bump maps for that layer. Variations on this character were as simple as swapping out the opacity mask for the black paint.

The production also realized less dependence on shader writers; look dev artists could easily add their own secondary specular response, or other shading effects and composite a material without too many restrictions. Material Layering became a critical aspect to the definition of looks on *Alice*: it was used to add slime on frogs, puddles of water on cobblestone, moss on stone.

Look Development

The lighting environments that were set up for *Alice* were approximated from the physical studio lighting used on the actors during reference photography and costume acquisitions. Many of these early light rigs became standard light setups for the look development turntables.

In the past, this was common practice to use a different light rig for characters than for environments. In a global illumination context it doesn't make sense to use two different light rigs; indirect illumination depends on a homogenous environment. Therefore sequence and shot lighting was always designed with the characters in context.
The initial expectation was that combined specular and reflection parameters would result in reduced ability to art-direct and dial the looks. In practice however, the imagery looked more correct out of the box, we almost never received direction to dial specular illumination and reflection independently.

It required some education on the part of the texture painters to distinguish specular intensity from "blurriness." Roughness was a parameter that had never been present in prior shading systems at Imageworks. However, with microfacet models it is a phenomenon that is critical to the realism of any material.

Texture Painting

All of these developments had a significant impact on the texture painting departments as well. Fewer parameters in the system and more combined controls meant an overall reduction in the number of maps that were needed for an asset. In addition, the modular nature of our new shading system allowed look developers to experiment with texture maps for unintended purposes. Control maps are for the most part floating point data, intended to be used for attenuation, bump or utility purposes in the shader. These were always painted in a 0-1 normalized range and dialed by the look development artist using a set of built-in correction controls available for each map. Simple masks were by far the most common maps used on creatures, since they defined regions where a material layer would be active, but they could also be used to mask other procedural effects etc. Variations among multiple characters can be as simple as a single shader setting on a shader or material layer, or as complex as an entirely different set of color and control maps. The red knights are an example of map-based and procedural variations in practice. Each particular red knights numerical designation is a simple mask on the black paint layer. Dirt and grime are modified by procedurally modifying parameters on the materials at render time.

Hair Shading

The grooming, animation and rendering of hair on *Alice* is worthy of a course all its own. The geometric complexity of hair makes it very challenging to sample efficiently in a ray-traced global illumination context. We had to establish new techniques and procedures for dealing with the amount of hair on the show.

Shadows on and from hair were always ray traced in the same manner as other geometry in the scene. This was a major source of fidelity and clarity in the renders of hairy creatures, but also of massive render times if not managed carefully. All of our hair received indirect light from the skin and surrounding environment by default. However, the hair did not occlude itself in the indirect diffuse solution. While completely possible, the sampling requirements and render time overhead of tracing the hair-to-hair occlusion was too prohibitive but in most cases this was a acceptable compromise. For these same reasons, the hair did not trace glossy reflections.

Traditionally look development artists leveraged the inaccuracies of deep shadow maps and point clouds to create a sense of softness and hair-to-hair energy transmission. Because none of these techniques were available to accomplish softness in the hair shading we relied heavily on an accurate model of the hair volume, and careful procedural texturing of the hair opacity. Setting up a successful hair look was an exercise in balancing hair thickness with transparency. While the up front cost of setting this up per character was significant, rendering hair was much less problematic in shots than in the past.

The door mouse presented a particularly interesting challenge in the hair-to-subsurface scattering interaction.

The Catch

None of this came without a cost.

Our render times for the average frame were long. This was not unexpected: we are asking the renderer to do a lot more at one time where previously we were executing a lot of incremental steps. This cost was easily offset by the amount of time saved by not having to render a massive number of passes through the system. In addition the predictability of the resulting images meant less tweaking for consistency with surrounding shots or sequences. Overall, the net rendering costs of switching to Arnold, did not go up nearly as much as anyone thought it would.

One of the most notable problems of ray-traced imagery in production is sampling noise, and because we were essentially sampling more than we ever had before, noise was an issue. One of the major culprits of sampling noise is contrast between consecutive samples, and this could usually be tracked down to light sources with very non-physical properties. Lights with zero area, an intensity of one and no decay over distance have no real world counterpart, and tend to introduce large amounts of energy in to the global illumination solution.

Case Study: Red Queen Throne Room

Let's take a look at an example of one of the more challenging look development and lighting cases. The red queen throne room is an ornate, richly colored set with no shortage of reflective surfaces, high-contrast lighting and fine geometric detail. The mood of the sequences in this location is sinister and threatening, but still well lit.

Interiors were generally difficult scenes to tackle. In this environment the key lighting is a bright exterior source, and it is considerably brighter, and cooler than any interior sources. This situation exacerbates sample contrast and contributes to noise in both the indirect diffuse and specular solutions.

There are numerous small accent light sources, from the candles, that need to contribute to the scene as well. We found that we could either cast specular illumination from these source OR reflect the flame geometry, but rarely if ever would both be appropriate. These accent sources are very localized, but will be computed for samples that are very far away, which is a waste of resources.

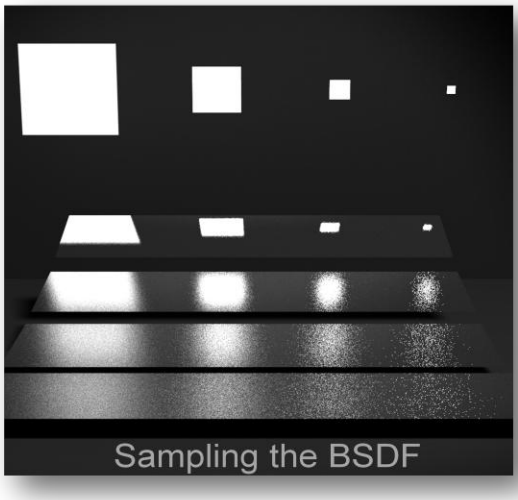
Fine geometric detail combined with highly reflective surfaces also proved to be an sampling challenge. The column bases for example are almost entirely defined by their specular response. This required multiple glossy specular bounces in order to resolve column self-reflections appropriately.   In addition, the columns and the characters have conflicting sampling requirements. Where the characters would benefit mostly from cg light sources, the columns do better reflecting geometric representations of the light sources.

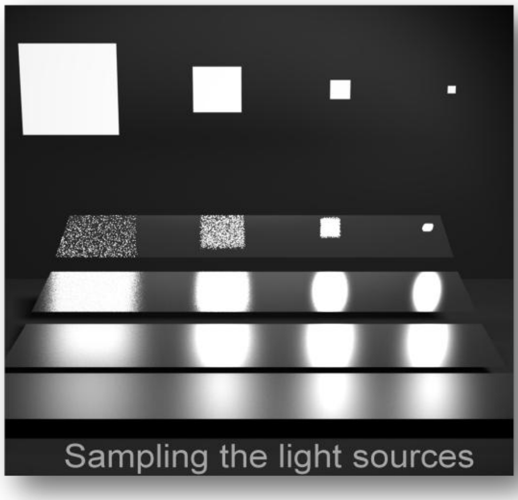
Sequences set in the red queen throne room were some of the first in the pipeline. These shots became a test bed for significant experimentation with the renderer and the shading system. One of the things we learned from this environment was the expense of indirect versus direct light sources. In most cases the indirect illumination was a fixed cost while the direct illumination cost increased as light sources were added to the scene. We found that substituting cg lights with geometry representations solved some sampling problems very efficiently.

We also implemented hooks into the shading system to cull out shader operations in certain contexts. By default, the specular component of the shader is not evaluated during the indirect diffuse solution, such a computation would be a caustic effect that would be extremely expensive to sample appropriately (but not impossible). Additionally we can cut the cost of our secondary illumination solution by removing light sources that do not significantly impact the overall result. Candle lights for example will illuminate surfaces during primary, or camera-ray shading, but will not be calculated during the indirect diffuse solution.

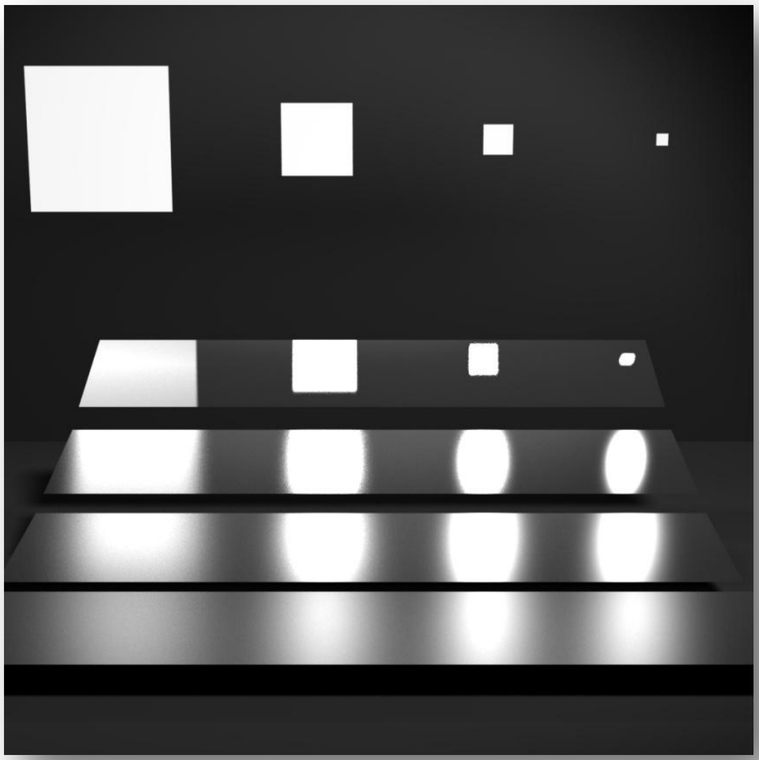
Other efficiencies were gained by altering the behavior of material layering in secondary shading. User can selectively disable the layering features in favor of a less complicated single-shader representation of the surface. By far the most effective solution to sampling problems, and render times, was Multiple Importance Sampling.

Multiple Importance Sampling: The Glossy Reflection Problem





On the right is an image of four area light sources emitting onto four planes of increasing roughness. This is sampling the light sources and evaluating the BRDF at each sample.
On the left is an image of a series of emissive cards being reflected in the same planes. This is sampling the BSDF of the shader. You can see that when sampling the light sources, the sampling scheme falls apart when large area light sources are combined with very low roughness values. On the other hand when sampling the BSDF, that case is handled quite well. However, for small area sources and high roughness values many of the reflection samples miss and we get significant noise in the results.



As presented by Eric Veach in his 1997 thesis *Monte Carlo Methods for Light Transport Simulation*, multiple importance sampling (MIS) is a modification of importance sampling designed to alleviate sampling artifacts in the glossy reflection problem.
Importance sampling depends on a probability density function (PDF) to ensure that stochastic samples are oriented in meaningful directions, and the sampling distribution does this for us. MIS introduces a balance heuristic to weight samples from both sampling methods in order to reduce variance.

MIS significantly increases the efficiency of our sampling and results in an overall reduction of noise for almost every situation. Having a meaningful PDF that corresponds accurately to the BRDF and sampling distributions is critical to the overall effectiveness of a BSDF in a multiple importance sampling scheme. The models we chose to implement had well defined PDFs that we were able to rapidly integrate into our existing shader architecture.
MIS helped enormously in reducing noise in glossy reflections. By choosing the best sampling directions for the columns versus the characters, sampling settings approached a “one-size-fits-all” ideal.

Case Study: Lighting and Rendering The Final Battle

The final battle sequence of *Alice* was particularly challenging on a number of levels. In this section, we are going to discuss the approach to lighting this sequence with a new renderer and shading system.

Final Battle: Geometric Complexity

Traditional instancing strategies broke down when we introduced procedural displacements to geometry. While scene memory use was manageable for most situations, computation of indirect shading slowed down dramatically. Limited trace distances and making small objects invisible to secondary lighting effects reduced the rendering expense of computing a the indirect lighting components, and the expense of negligible accuracy in the energy transport simulation.

Final Battle: Lighting Strategies

The final battle sequence incorporated a significant amount of set, creature and live action interaction. At the outset, it was a given that the lighting of all aspects of the sequence had to be consistent. The initial approach was to create a broadly defined lighting rig that could be distributed to all of the shots.

The stage lighting informed how the initial lighting rig would be built. Generally, actors were lit with bounce cards to the right and left of camera which served to fill in darks and soften shadow areas significantly. This on-set approach was simulated in cg using the natural falloff of emissive geometry in the indirect diffuse light computations. The overall look of the sequence was that of an overcast morning. The artwork however had also depicted a high contrast between the darks and mid tones which became a balancing act for the lighting team.

While single pass rendering was generally encouraged and successful, it was not always practical in the live action visual effects context. Usually separate passes had to be generated to separate foreground and background elements relative to a plate. Sometimes special utility passes had to be generated to integrate live action elements more appropriately. These passes still had to integrate the entire scene geometry to accurately represent shadows, bounce light and reflections. Since render times were expected to be fairly long, it was common practice to light shots at half resolution and with lower sampling settings. We were still getting all of the secondary shading effects and subsurface scattering which helped us judge the lighting more accurately than if we had rendered without those effects at full resolution. Supervisors got used to these slightly noisy renders and could provide more creative feedback earlier in the lighting process.

Final Battle: Subsurface Scattering

Subsurface scattering proved to be a special challenge for this sequence as it was a key characteristic of the white knights look. We found that using point based subsurface was too unwieldy for the renderer as each character had to generate a separate point cloud per frame. This amount of data would fill up ram rather quickly. It became much more efficient to use the ray traced subsurface scattering implementation, following [JENSEN01], as the cost was significantly cheaper for many hundreds of creatures that were small in frame. The traced scattering solution also tended to preserve more geometric and texture detail over the point cloud solution.

Conclusions

Sony Pictures Imageworks is one of the largest visual effects facilities in the world. Over the past year and a half we have managed to transform the entire rendering strategy for feature film visual effects production. The effect of these changes are numerous and profound.

We have moved away from a biased multiple pass rendering system, to a largely single pass global illumination system incorporating modern ray tracing and physically based shading techniques. Our shaders for surfaces and light sources respond much more naturally than previous implementations. Robust global illumination methods combined with efficient production practices have allowed artists to experiment freely with simulating on-set lighting practices.

We have streamlined the users interaction with the shading system. We have reduced the number of parameters and associated them with real world surface phenomena that can be discussed with clarity. We have provided a set of tools such as material layering to intuitively add visual complexity.

The impact on shot production and lighting procedures has been equally profound. Master lighting setups translate predictably to different shot settings, and lighters can get a sense of shot-to-shot consistency within a sequence much faster than with previous methods.

And finally, since the first frame rendered incorporates the full complement of shading components, including indirect diffuse bounce, image based lighting, glossy reflections and ray traced refractions, lighters and supervisors are much better equipped to make creative lighting judgments much sooner in the production schedule.

Moving Forward

At Imageworks we continue to research, experiment with and engineer new rendering solutions. We are continually developing new techniques, both procedurally and technologically, to use these tools in production. Current areas of development interest in which we are extending physically based shading ideas are: specular to diffuse energy transfer (aka caustics), subsurface scattering using ray-traced lighting and global illumination, and efficient and realistic ray traced hair shading.

Imageworks' sponsored Open Shading Language open source development incorporates a lot of the work discussed here in a generic framework for integration into almost any rendering engine. In our use of OSL internally we have found the language to dramatically improve our ability to get shaders to users rapidly. We have also seen much better consistency in the resulting imagery, owing to a much more homogenous sampling environment, and the implementation of illumination models as closures. OSL Shaders themselves do not contain explicit lightloops, but rather make calls to library functions that will be evaluated at a later stage in the rendering process.

Because OSL abstracts the illumination integration process out of the shaders, the renderer is free to make decisions about what techniques are most appropriate to solve the

lighting . This continues to be an active area of development for the renderer developers and already has yielded excellent gains in the area of IBL importance sampling.
As always, we continue to search for solutions to long render-times, but ultimately the quality of the image is the top priority.

References

[1] [WALTER07] Walter, B., Marschner, S. R., Li, H., and Torrance, K. E. Microfacet Models for Refraction through Rough Surfaces. In *Rendering Techniques (Proc. EG Symposium on Rendering)*. (2007)

[2] [VEACH97] Eric Veach Robust Monte Carlo Methods for Light Transport Simulation, Stanford University, (1997)

[3] [JENSEN01] Jensen, H. W. , S R. Marschner , Levoy M. , Hanrahan P., A practical model for subsurface light transport, *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, (2001)

[4] [JENSEN04] Jensen, H. W. , A practical guide to global illumination using ray tracing and photon mapping, *ACM SIGGRAPH 2004 Course Notes*, (2004)