

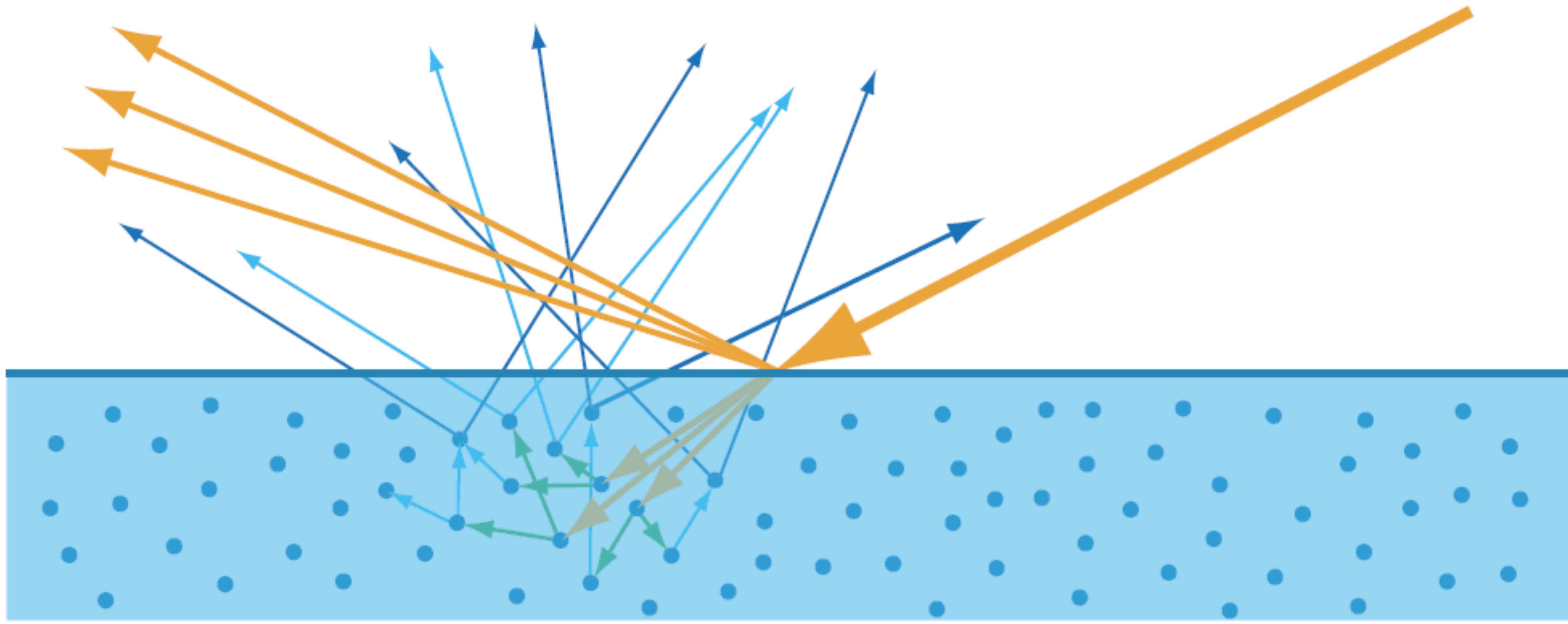
Background: Physics and Math of Shading

Naty Hoffman

2K



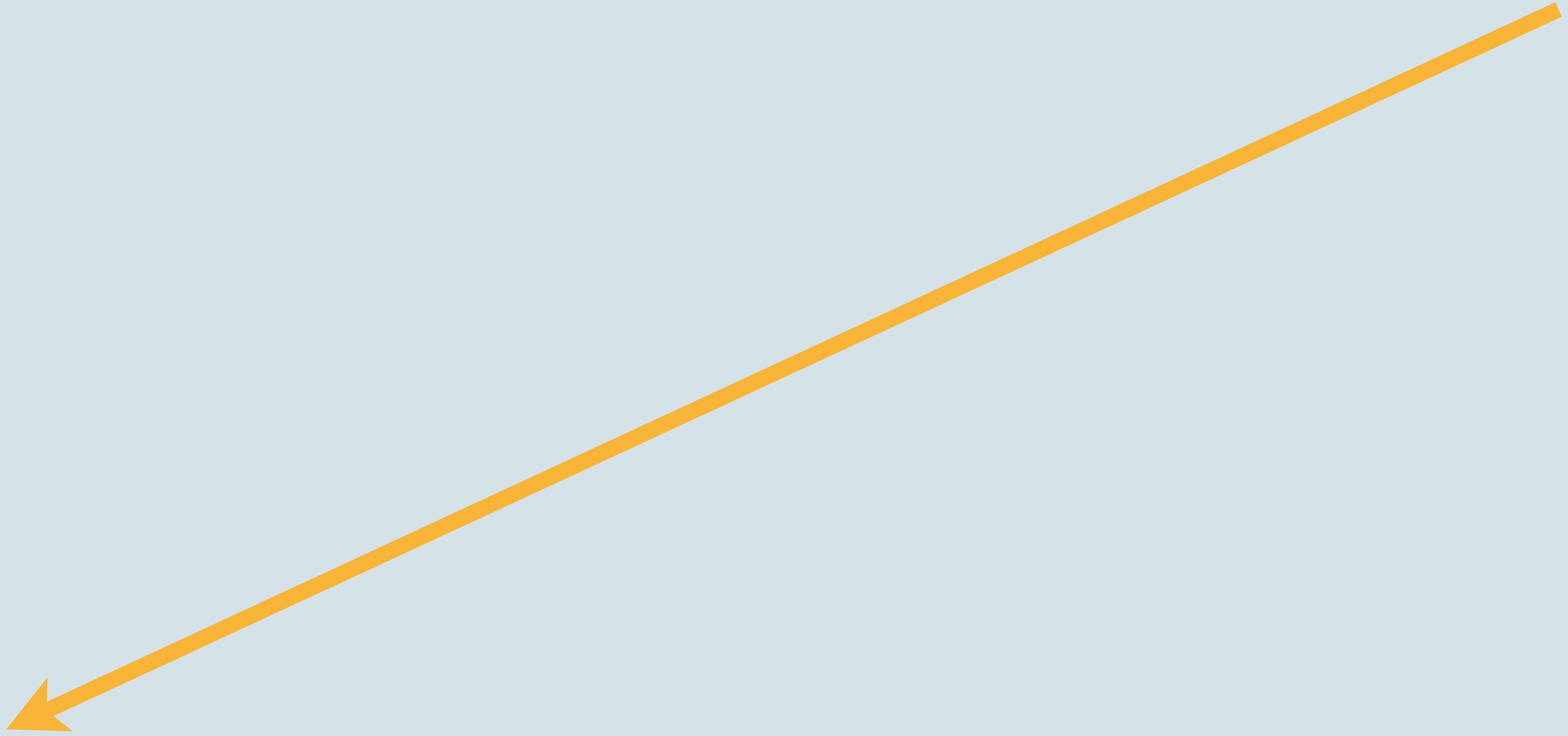
Hi. Over the next 15 minutes I'll be going from the physics underlying shading, to the math used to describe it and from there to the kind of rendering implementations we'll see in the rest of the course.



SIGGRAPH 2013



We'll start with the physics that happen when light interacts with matter.



The simplest case is light propagating through a *homogeneous medium* with exactly the same properties everywhere. In this case light moves in a straight line.

Digitized by srujanika@gmail.com

Martin Matranga, Lead graphics programmer, Crytek

think of any higher praise for a book than the fact that it sits on my desk, and within easy reach, and *Real-Time C++* by Eric Lippert is one of the few books that qualifies for such distinction.

one for any reason.

"One of my favorite things about the new system is that it's so much easier to use."

תְּמִימָנָה בְּבֵית יְהוָה וְבְבֵית מִזְבֵּחַ



Some homogeneous media don't significantly change the light's color or intensity..

—Gabe Newell, President, Valve

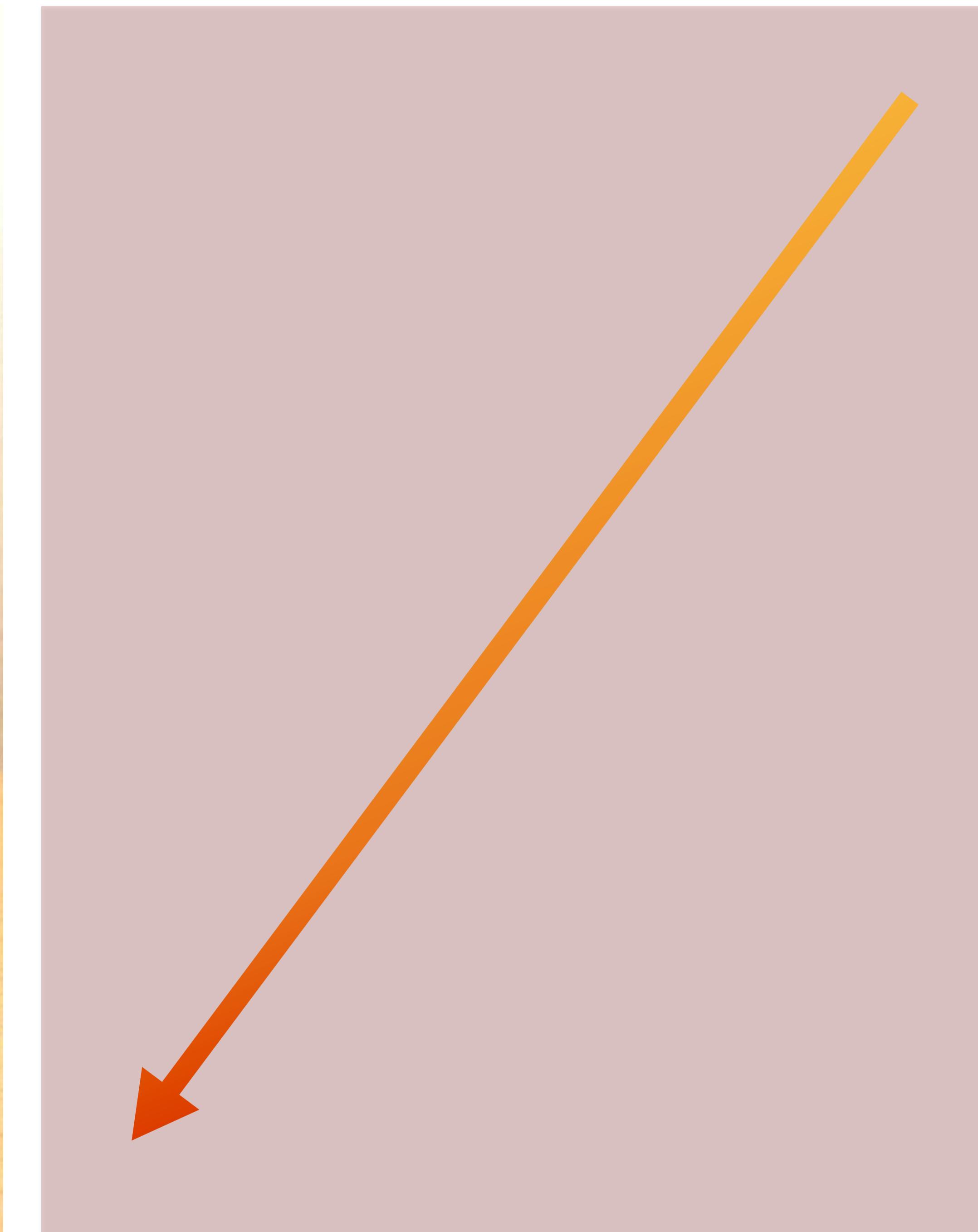
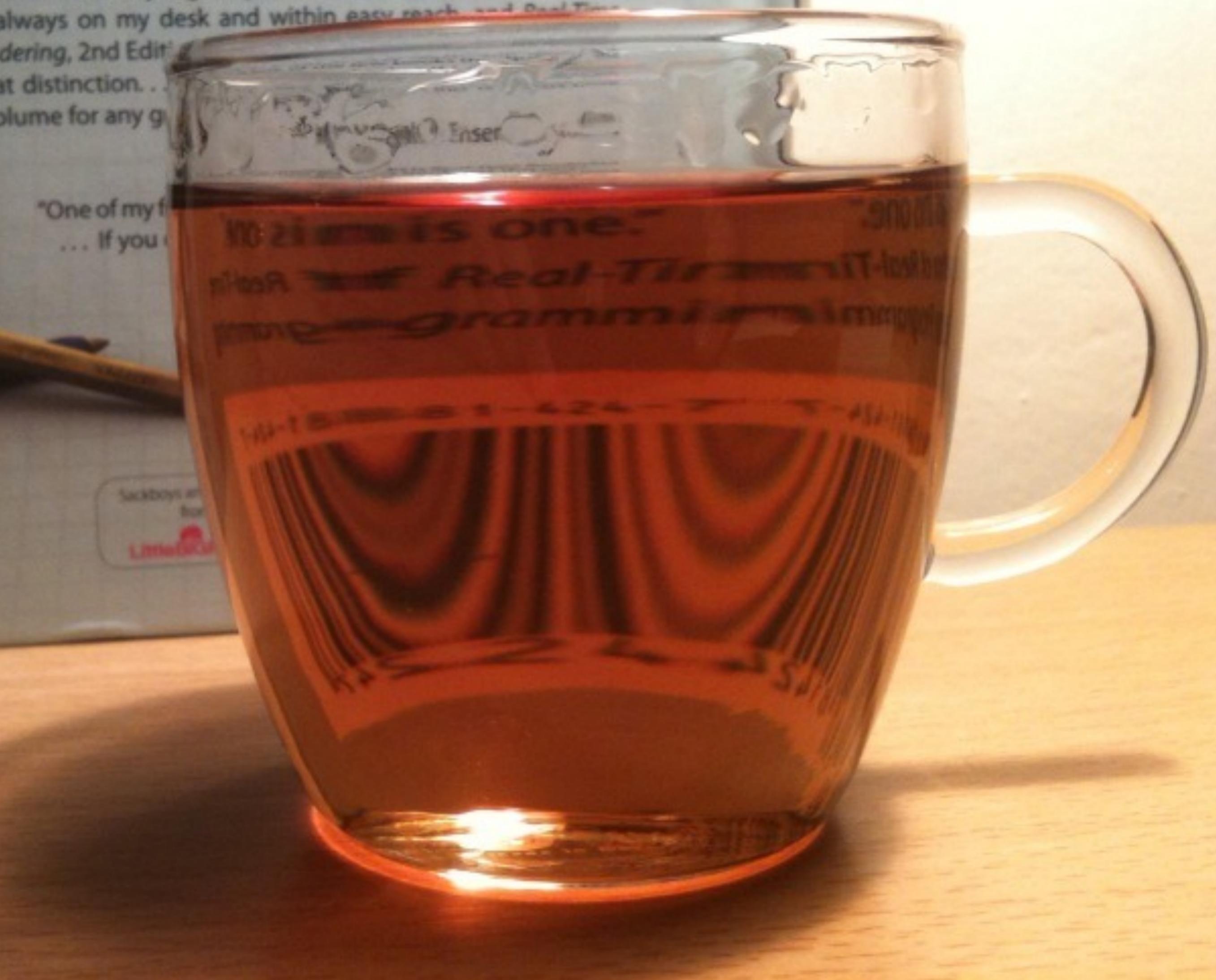
s a solid introduction to the essential techniques every
should know. It also serves as a great reference for finding
publications and is a joy to read."

—Martin Mittring, Lead graphics programmer, Crytek

I think of any higher praise for a book than the fact that
always on my desk and within easy reach... —Paul Toma
dering, 2nd Edit
it distinction...
olume for any g

"One of my f
... If you c

NO 23 THIS ONE
Real-Time Ray Tracing
Programming Examples



...while others absorb part of the visible light traveling through them, changing its intensity and potentially its color. For example, this medium absorbs more light in the blue part of the spectrum, giving it a red appearance.

Figure 1. A schematic diagram of the experimental setup for the measurement of the absorption coefficient.

Martin Mattings, Lead graphics programmer, Crystal

think of any higher praise for a book than the fact that it sits on my desk, and within easy reach, and that *Time* and *Newsweek* are one of the few books that qualifies for distinction.

www.flux.org

—
—
—
—
—

"One of my favorite things you can do is read True Stories," he says. "It's great for one's writing because it gets the mind going."

"One of my

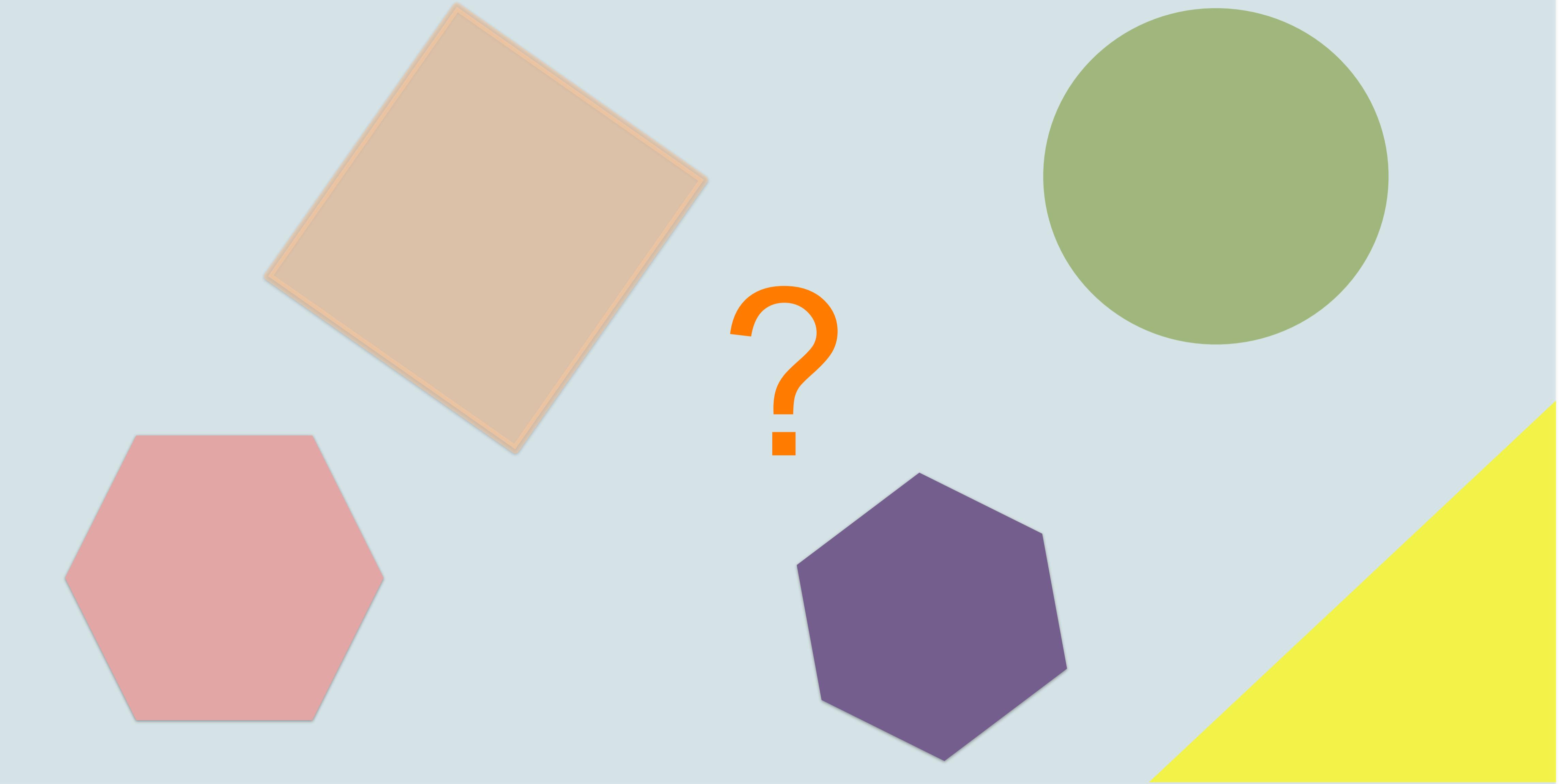
卷之三



Scale is important. For example, clean water does absorb a little light in the red end of the visible spectrum, but it's not noticeable over a few inches.



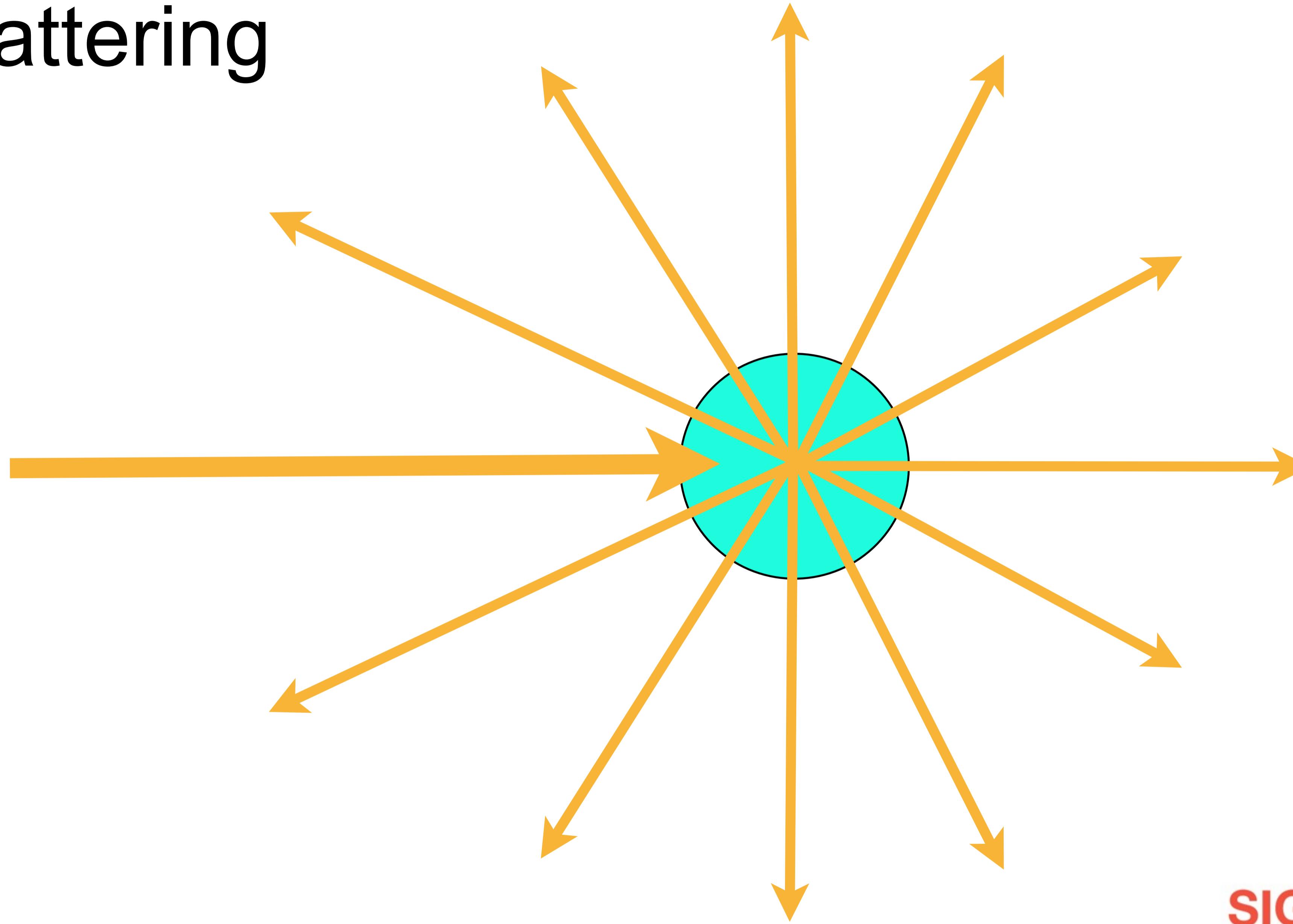
But this absorption is quite significant over distances of dozens of yards.



SIGGRAPH2013 

In an inhomogeneous medium, the index of refraction (which is the property of matter that affects light) changes. This causes light to no longer move in a straight line.

Scattering



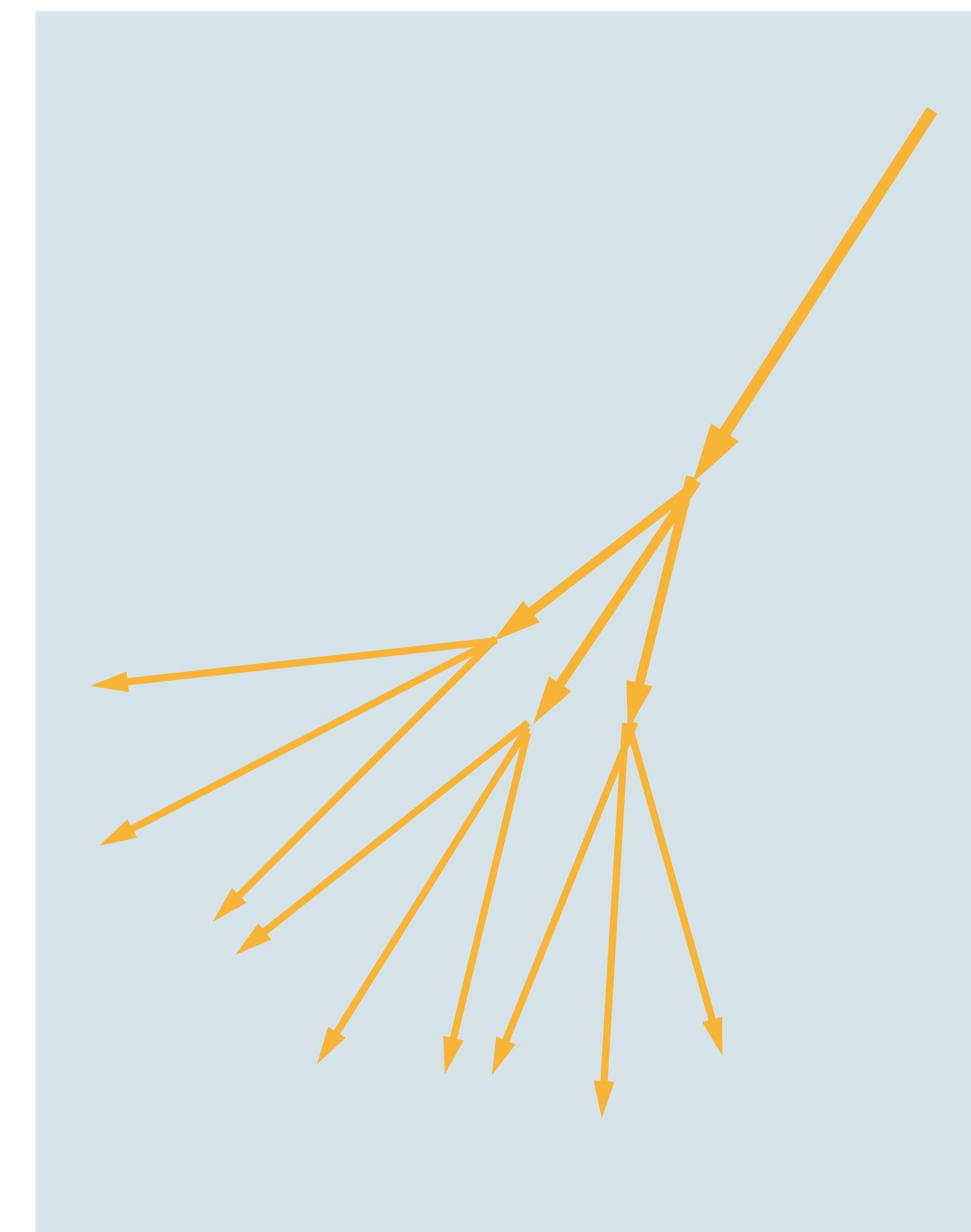
Abrupt changes in the index of refraction cause scattering, which changes the direction of light propagation.

ovides a solid introduction to the essential techniques every
nner should know. It also serves as a great reference for finding
want publications and is a joy to read."

—Martin Mittring, Lead graphics programmer, Crytek

I can't think of any higher praise for a book than the fact that
it's always on my desk and within easy reach, and *Real-Time
Rendering, 2nd Edition* is one of the few books that qualifies for
that distinction. . . . There's no doubt that this is a must-have
volume for...

"One of the best graphics books is *Real-Time Rendering*.
It's the most comprehensive book on the subject."

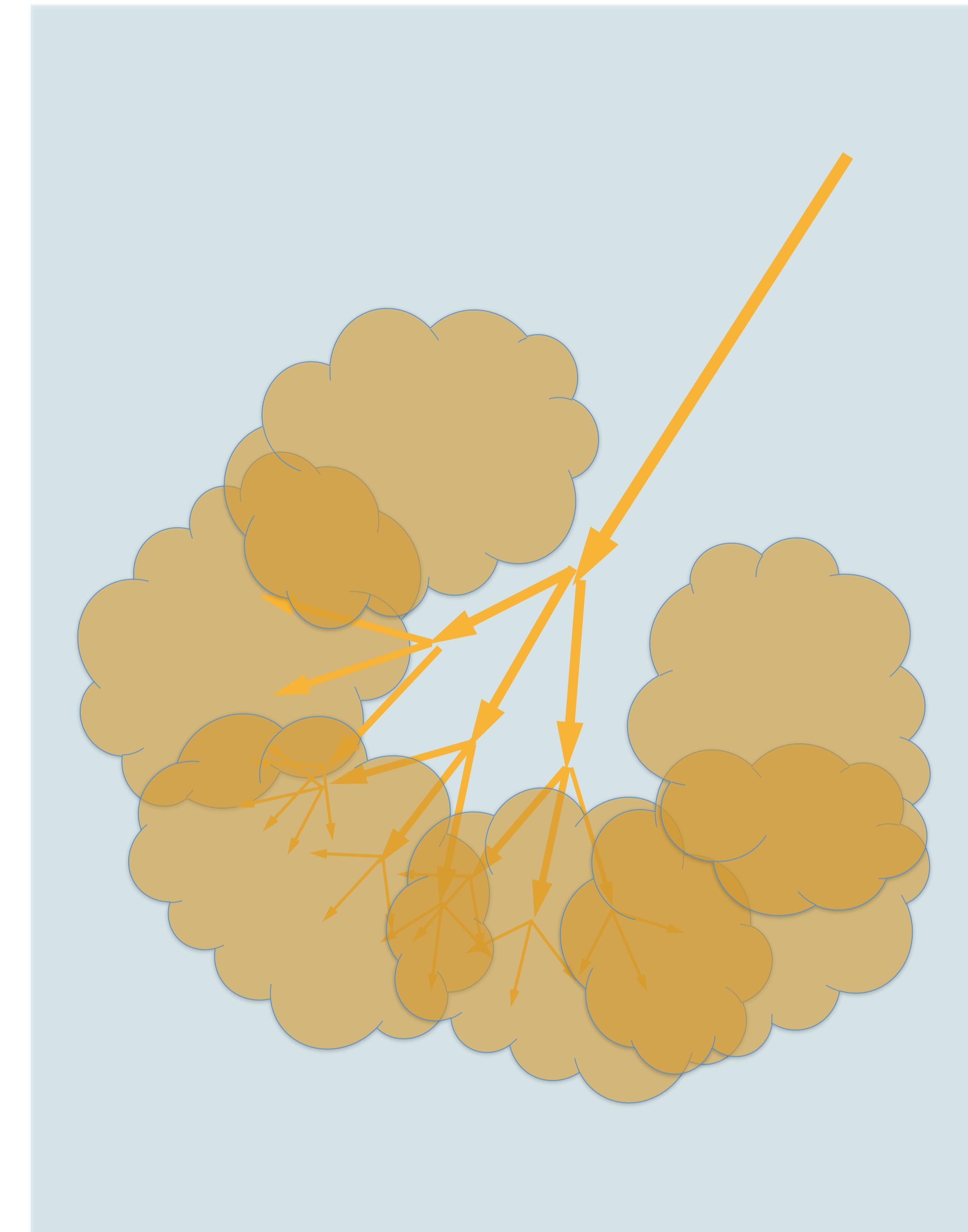


An inhomogeneous medium contains numerous scattering particles. These could be dense enough to randomize the light's direction somewhat, giving a cloudy appearance...

n't think of any higher praise for a book than the fact that
always on my desk and within easy reach, and *Real-Time
Rendering*, 2nd Edition is one of the few books that qualify for
at dist...
volume 1

—Herb Maresas, Ensemble Studios

"Or
"Read my favorite graphics books is *Real-Time Rendering*.
If you only get one graphics book, get this one."



...or to randomize it completely, giving the medium an opaque appearance.

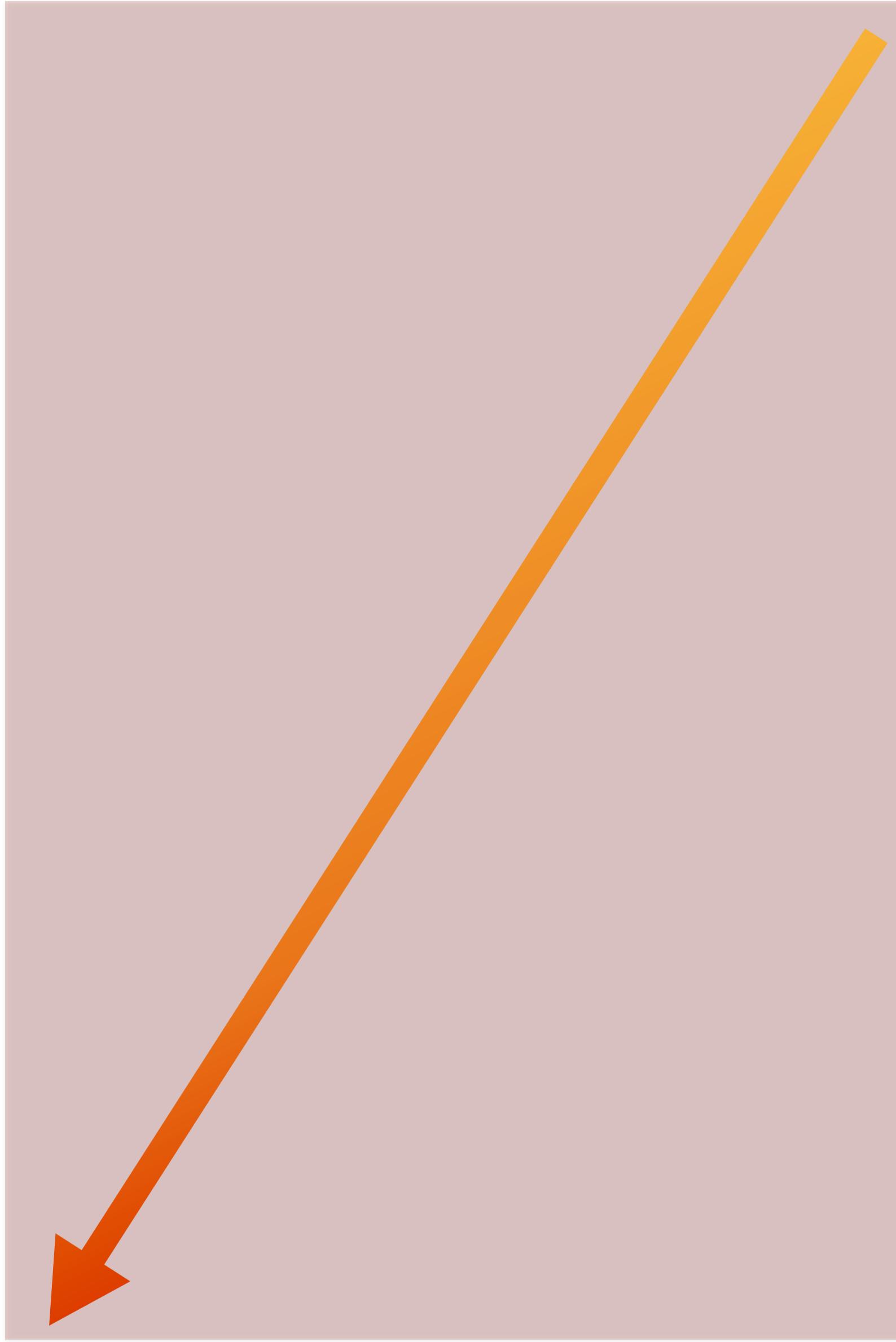


SIGGRAPH 2013

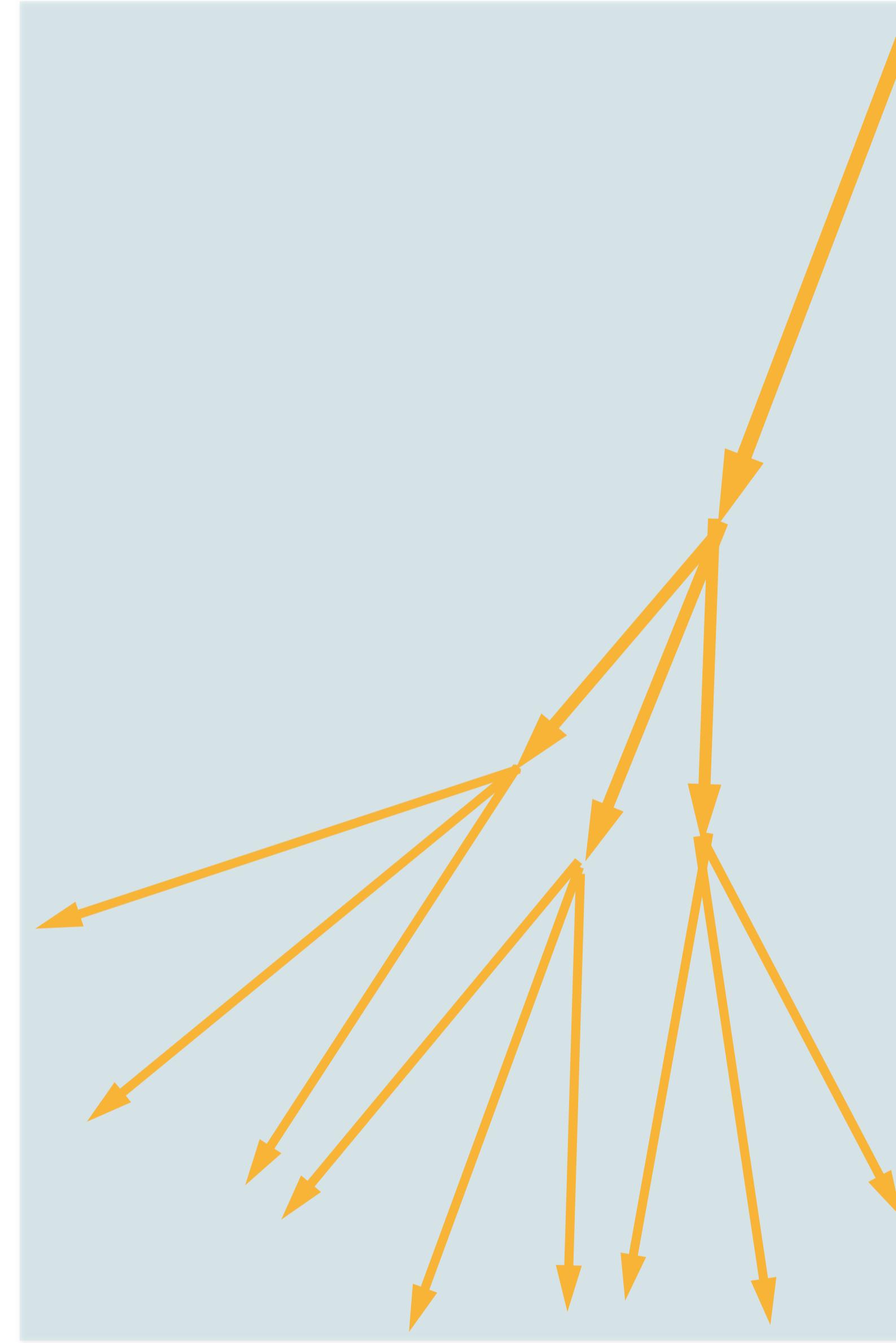


Scale also matters for scattering: for example, clean air doesn't noticeably scatter light over a few yards, but it definitely does over a distance of miles.

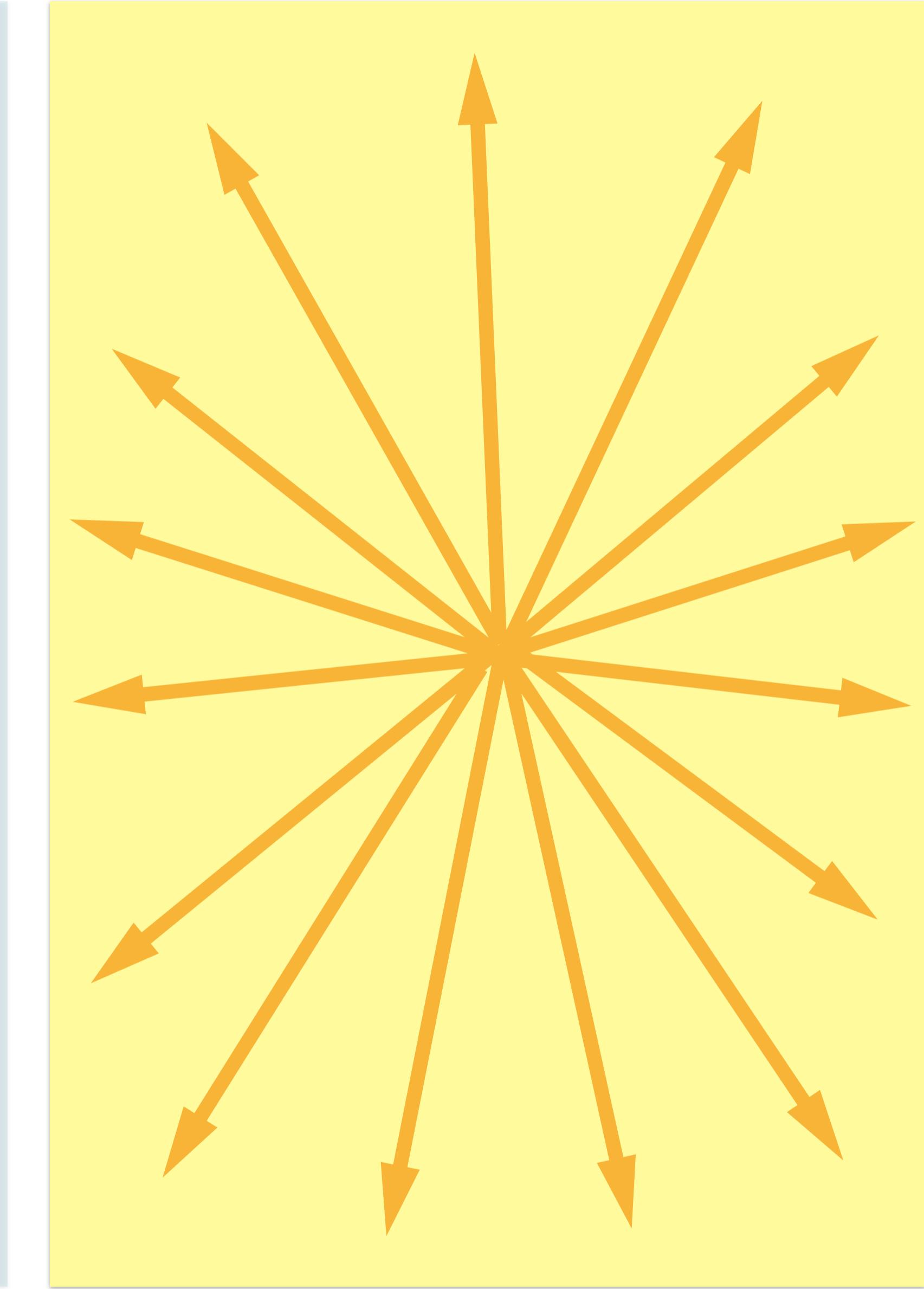
Absorption



Scattering

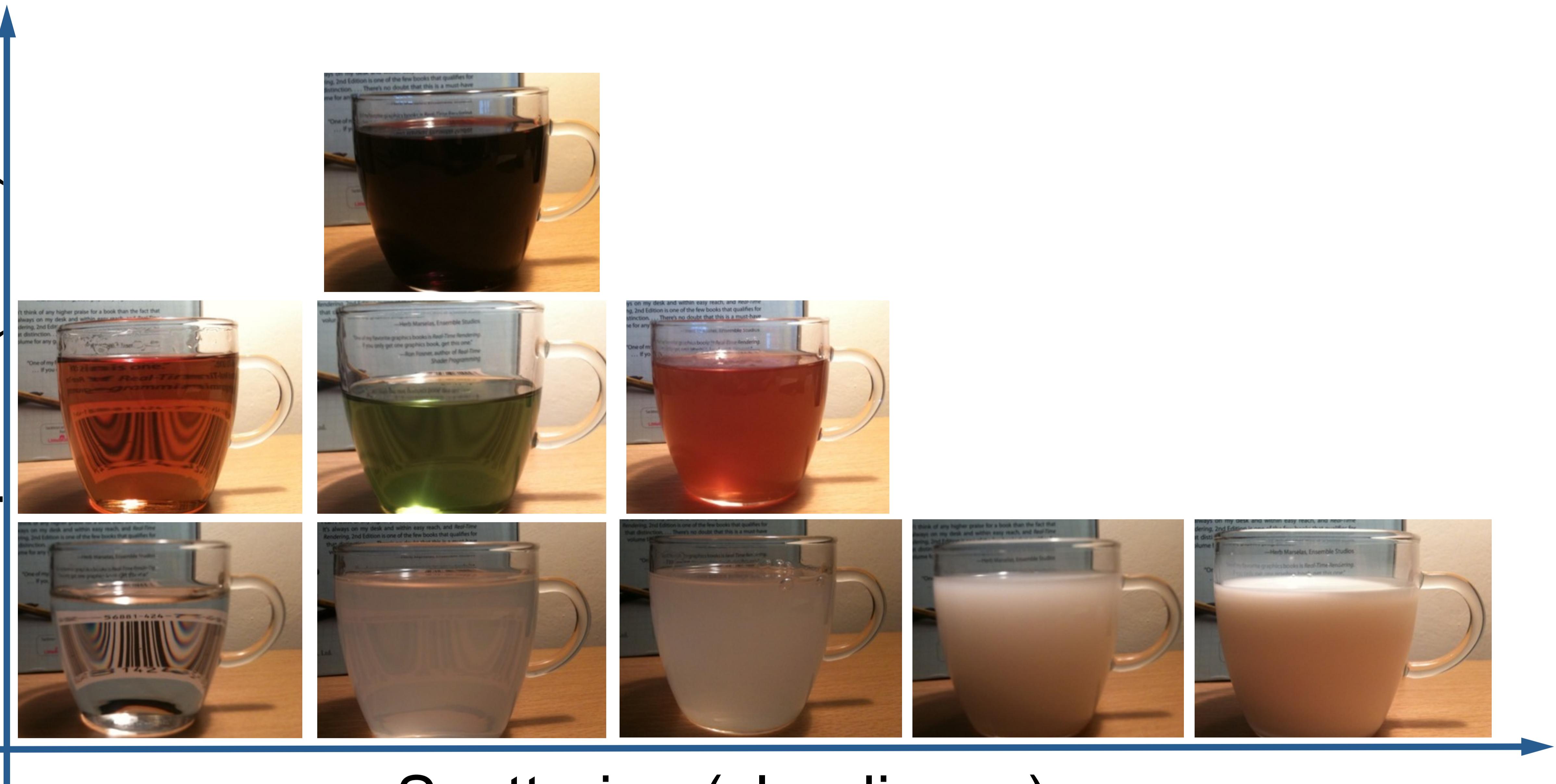


Emission



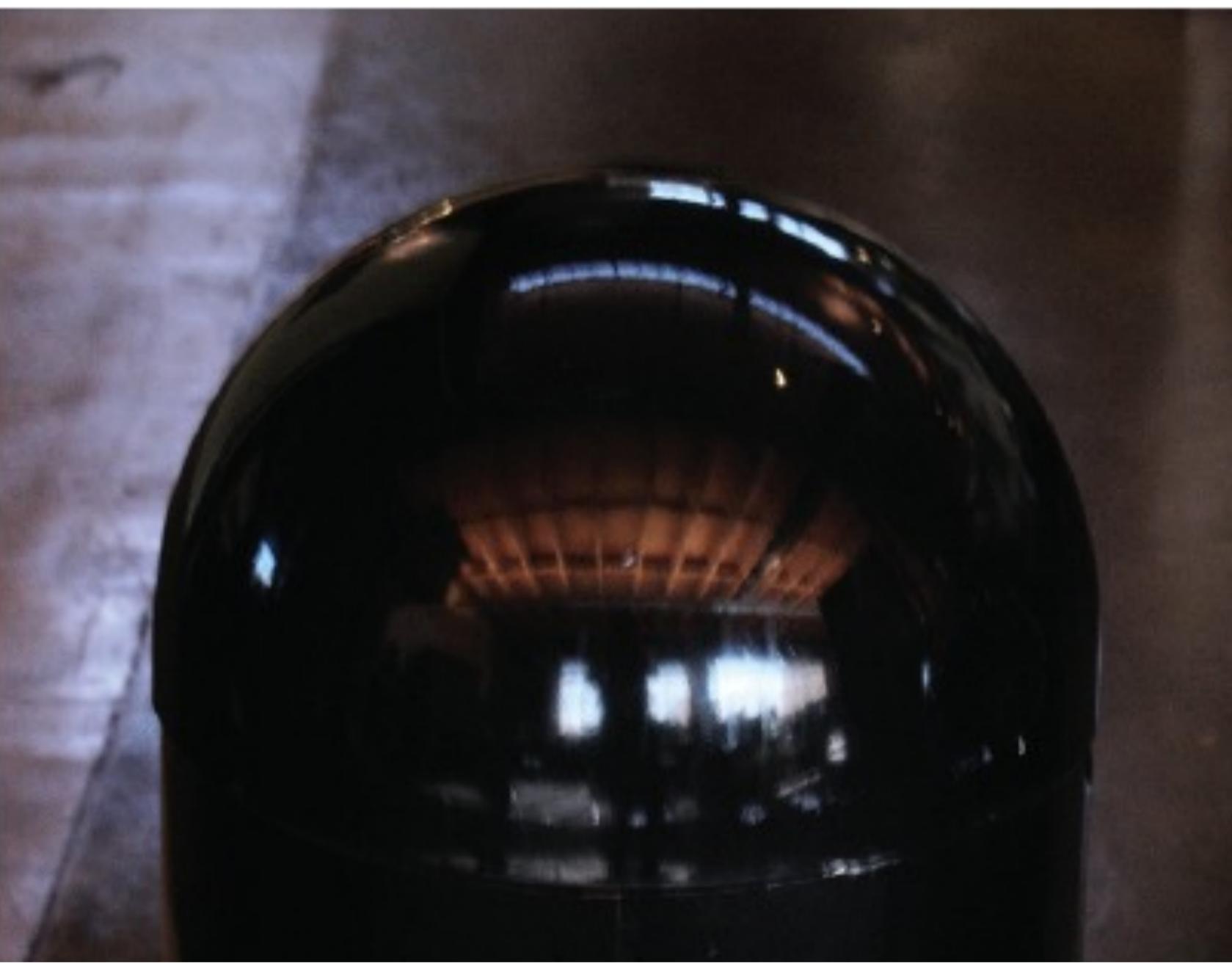
To summarize, there are three basic modes of light / matter interaction: absorption (which changes light's intensity and / or color), scattering (which changes light's direction), and emission (which creates new light; most materials don't exhibit emission and I won't further discuss it in this talk).

Absorption (color)



Scattering (cloudiness)

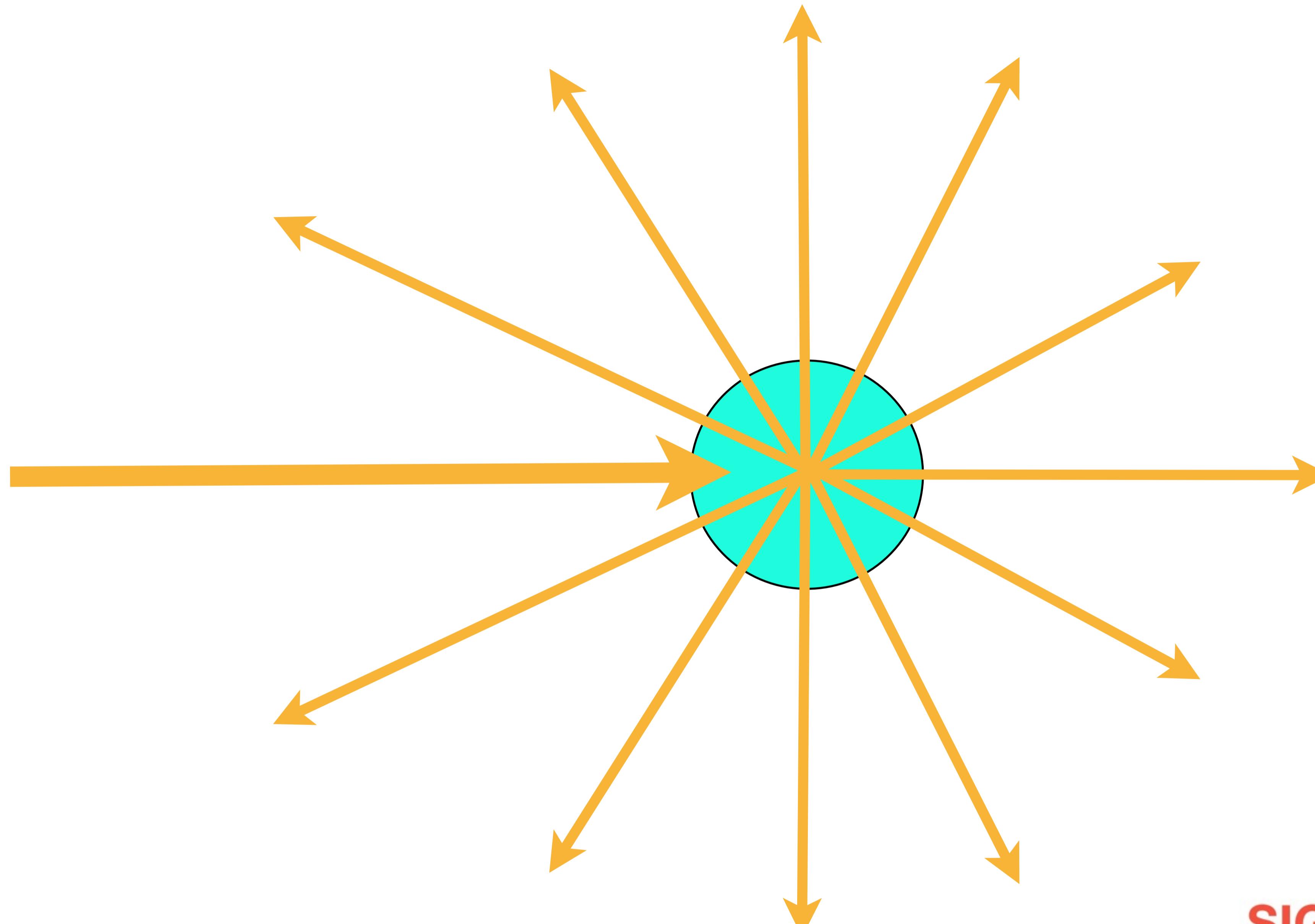
The overall appearance of a medium is determined by the combination of its absorption and scattering properties. For example, a white appearance (like the whole milk in the lower right corner) is caused by high scattering and low absorption.



SIGGRAPH 2013



While media are easy to understand, most of the time in graphics we are concerned with rendering solid objects, in particular the surfaces of these objects.



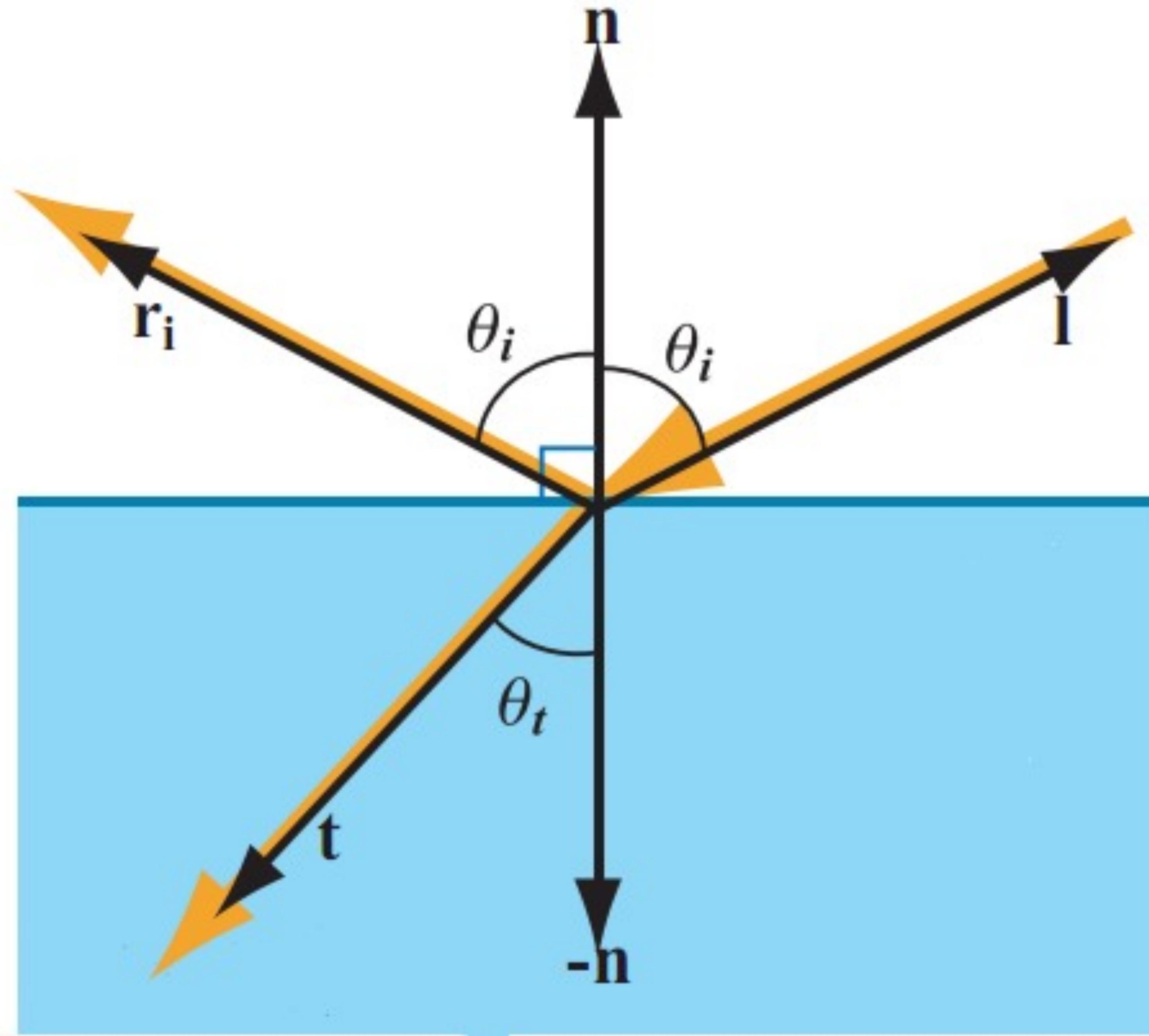
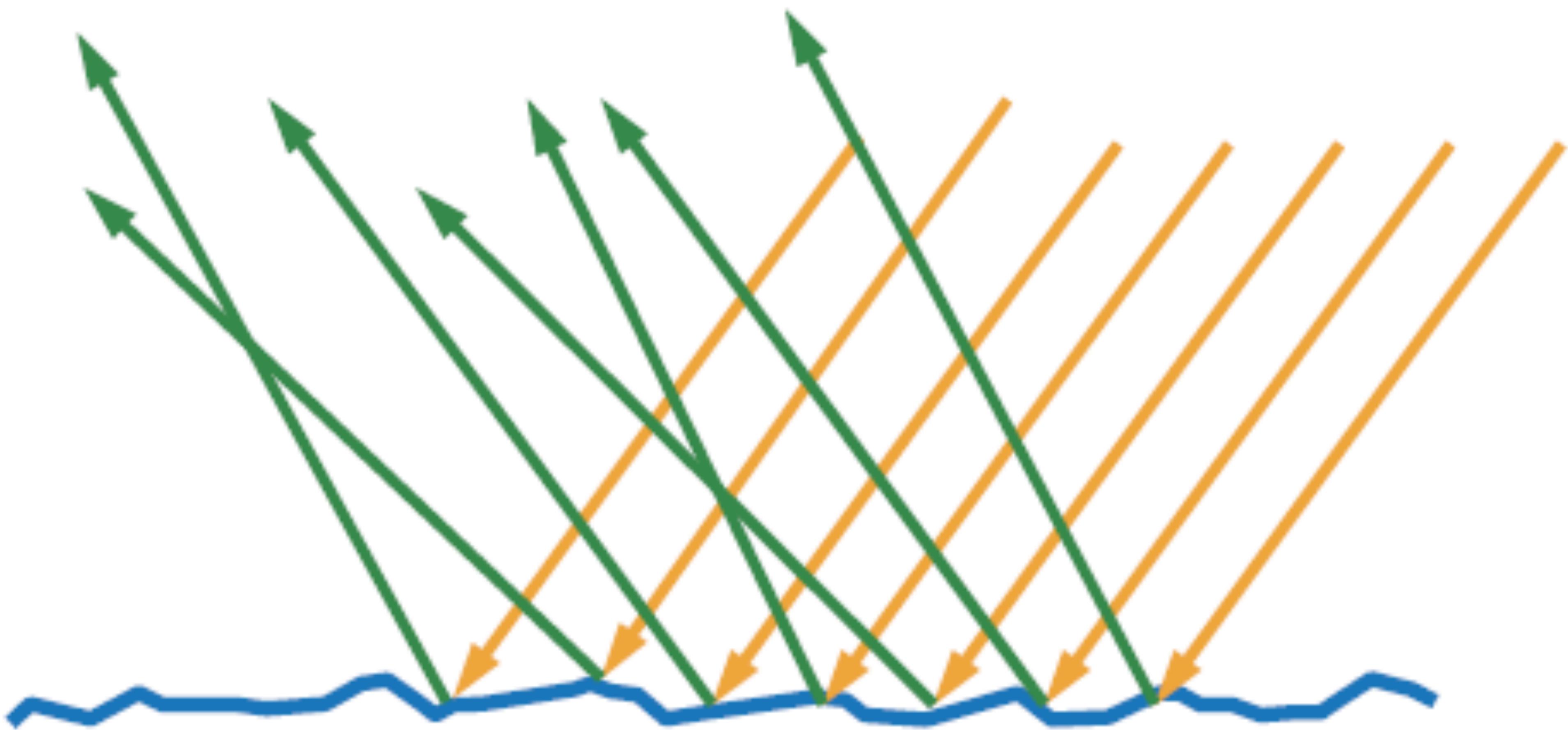


Image from "Real-Time Rendering, 3rd Edition", A K Peters 2008

A flat surface (defined as a plane separating two volumes with different indices of refraction) is another special case of scattering; such a surface scatters light into exactly two directions: reflection and refraction. In this case "flat" means *optically flat* - any irregularities are smaller than visible light wavelengths and thus do not affect visible light.

Microgeometry



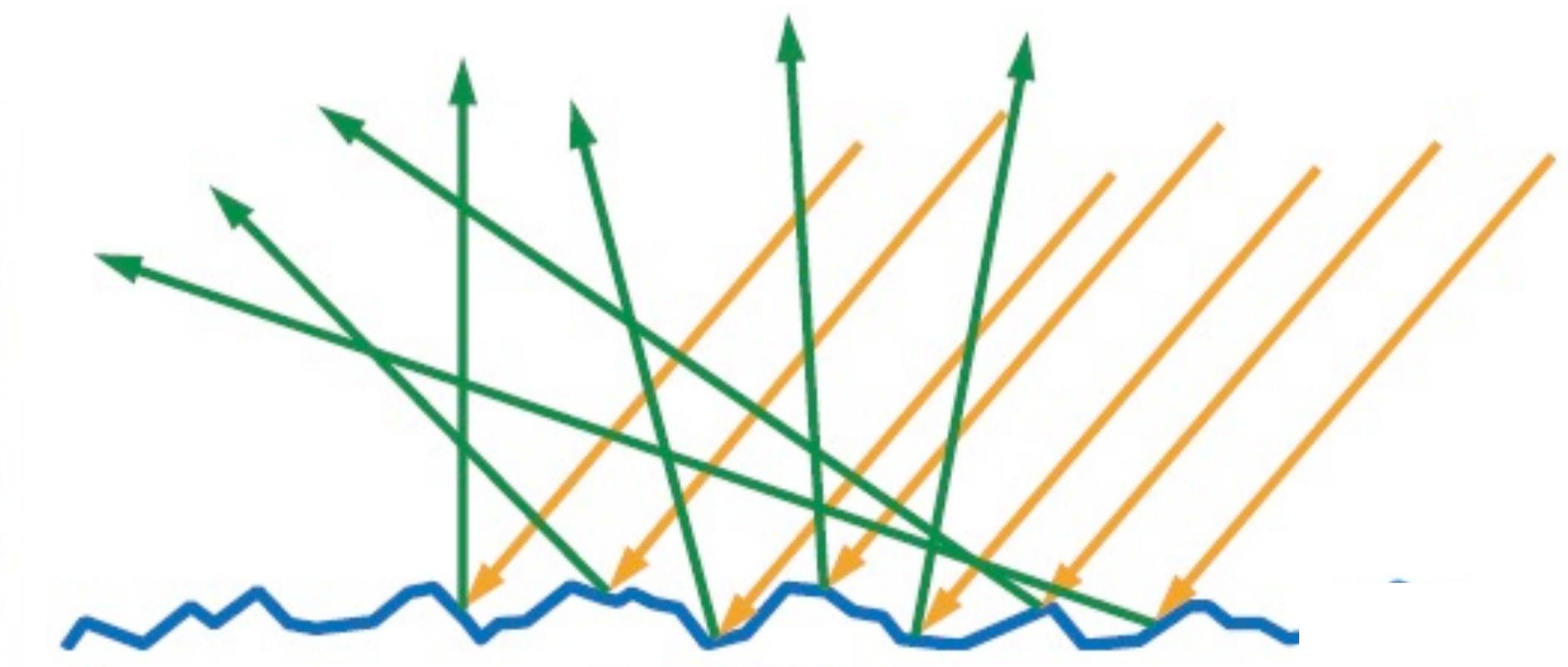
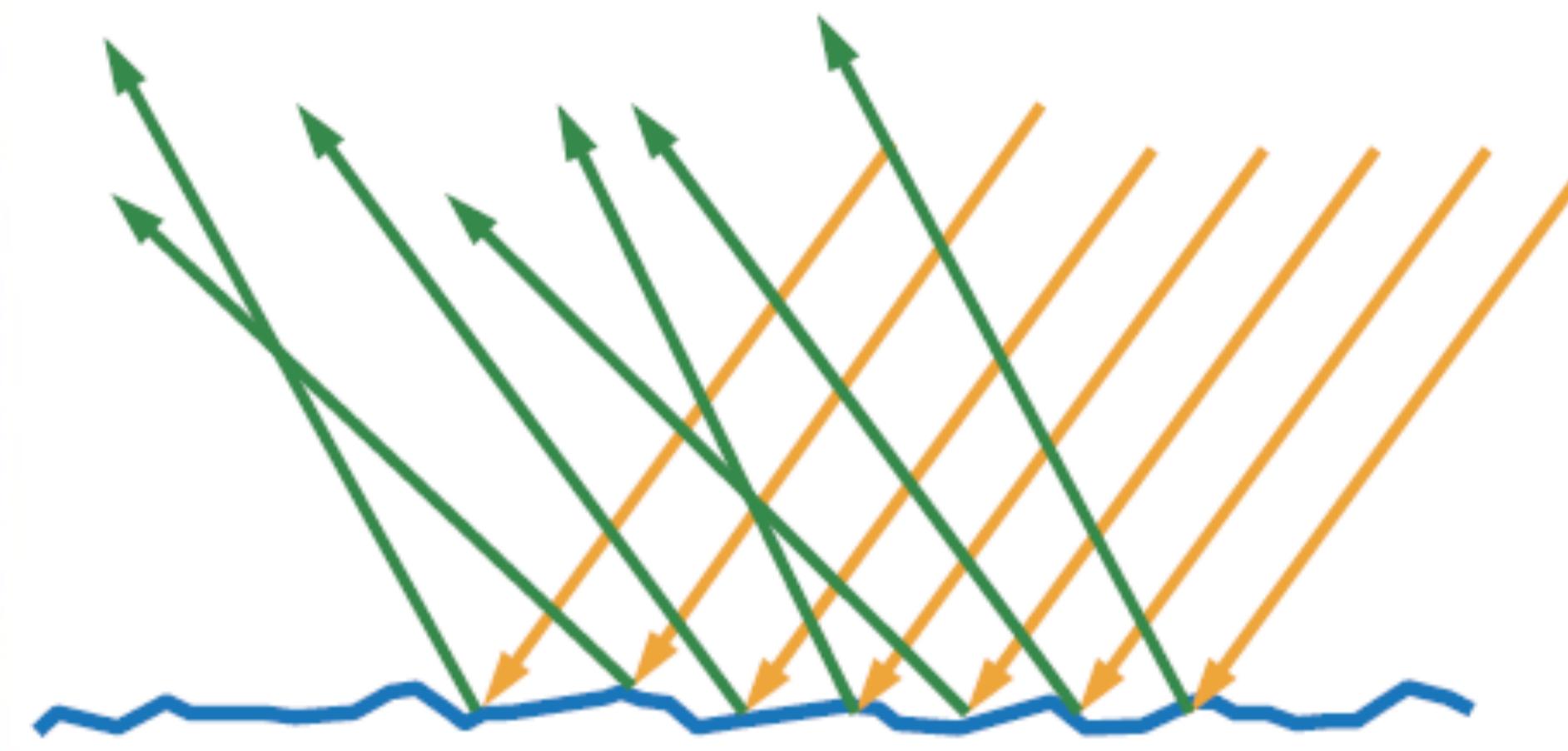
SIGGRAPH 2013



Image from "Real-Time Rendering, 3rd Edition", A K Peters 2008

Some rare real-world surfaces (like high-end telescope optics) are optically flat, but most aren't. Most have microgeometry - bumps that are bigger than a light wavelength but too small to be individually visible. Each surface point reflects (and refracts) light in a different direction - the surface appearance is the aggregate result of all the different reflection & refraction directions.

Rougher = Blurrier Reflections



SIGGRAPH 2013

Images from "Real-Time Rendering, 3rd Edition", A K Peters 2008

19

These two surfaces, equally smooth to the naked eye, differ in roughness at the microscopic scale. The surface on the top is only a little rough; incoming light rays hit bits of the surface that are angled slightly differently and get reflected to somewhat different outgoing directions, causing slightly blurred reflections. The surface on the bottom is much rougher, causing much blurrier reflections.

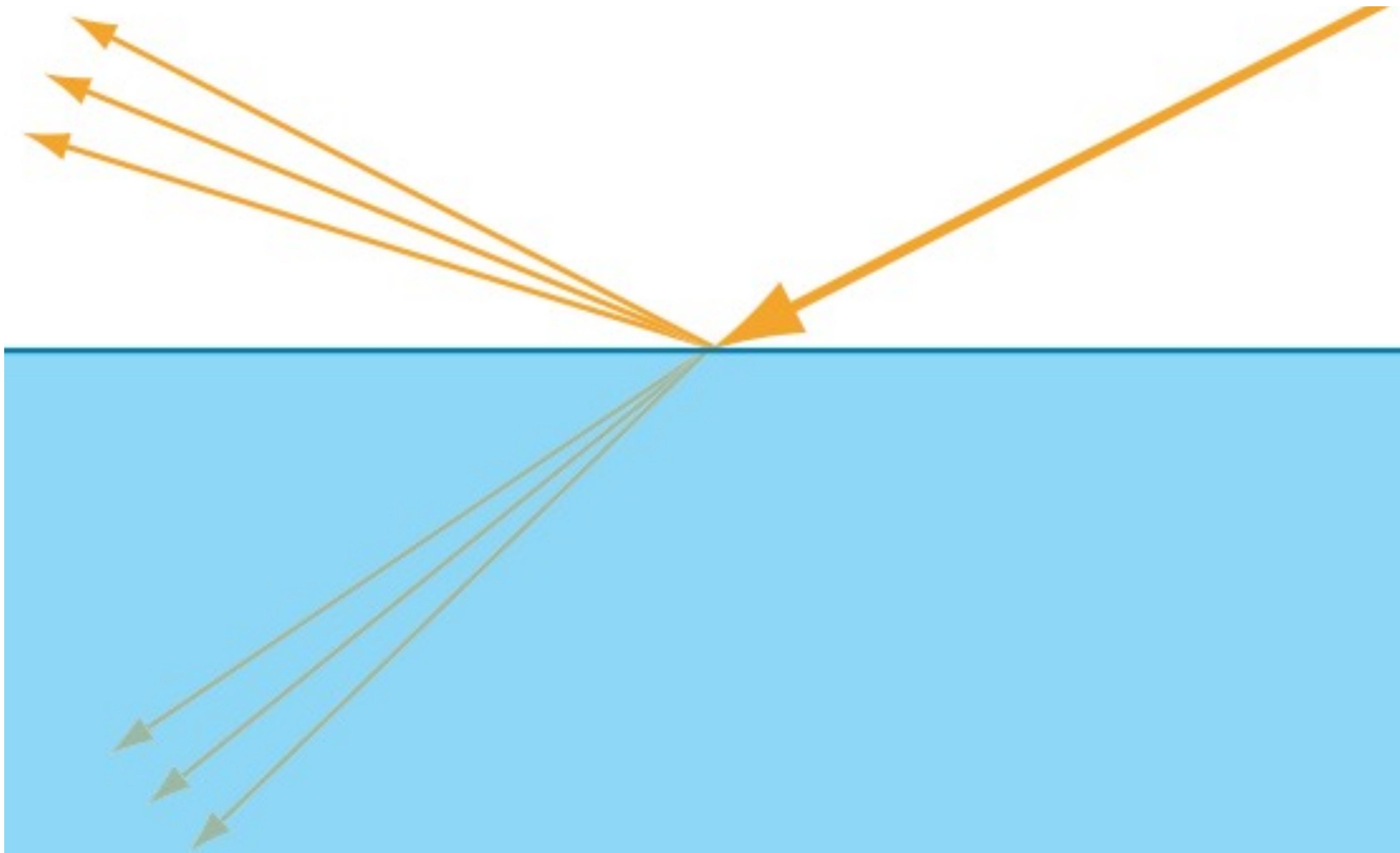
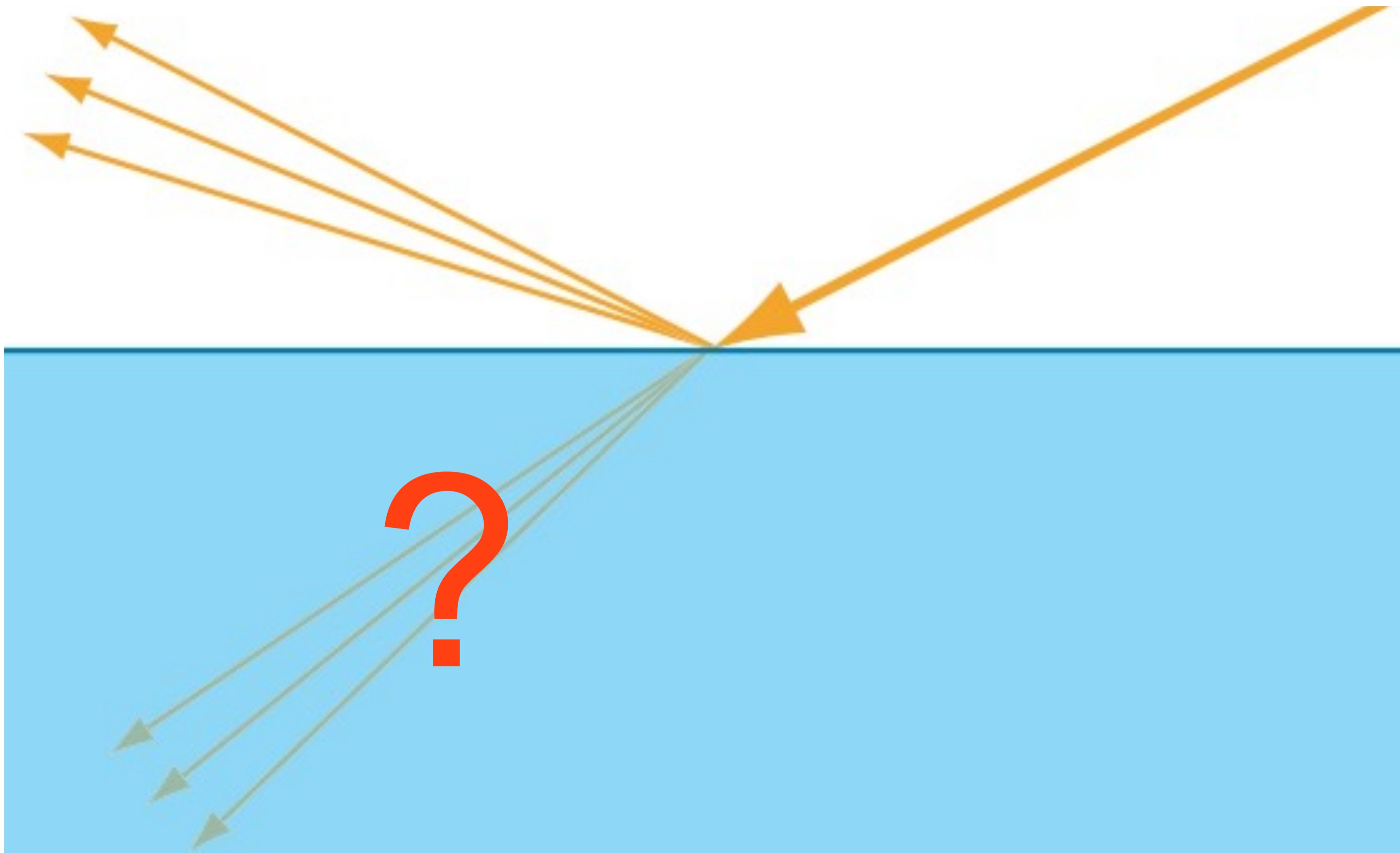


Image from "Real-Time Rendering, 3rd Edition", A K Peters 2008

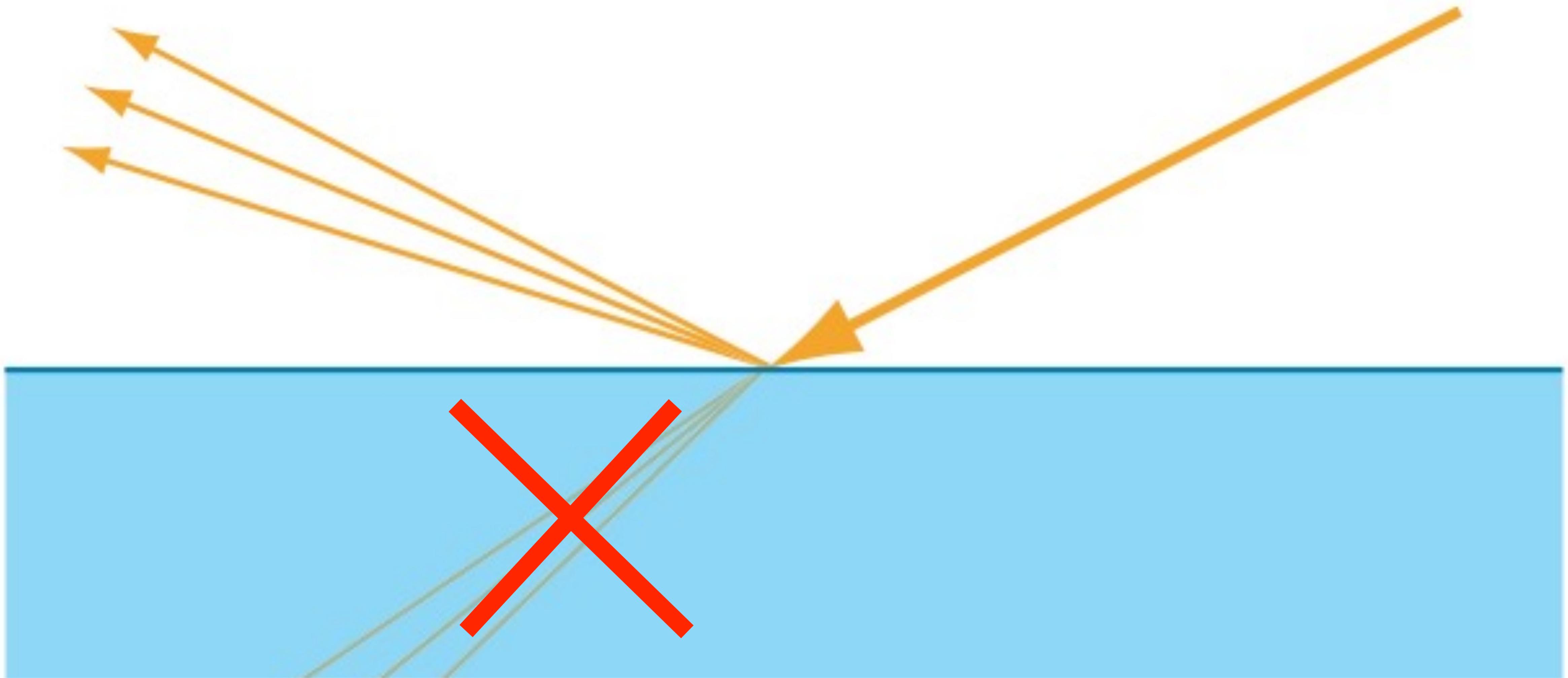
SIGGRAPH 2013



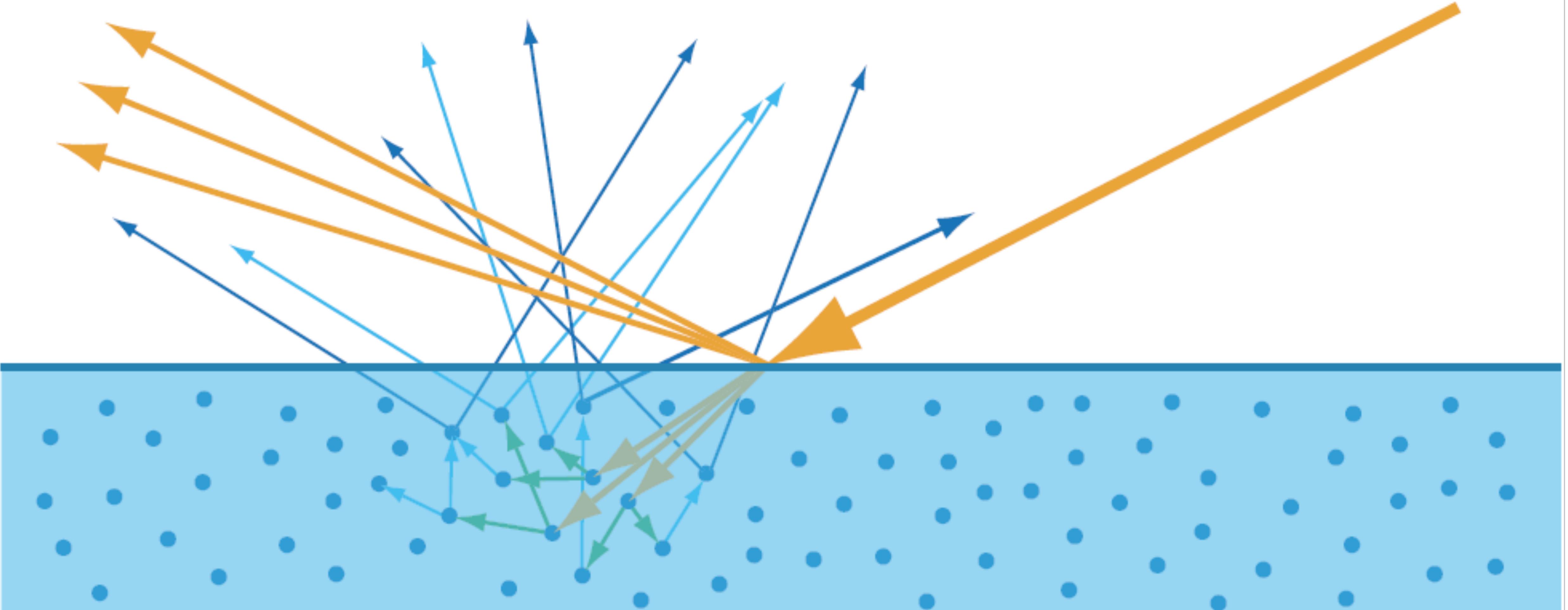
In the macroscopic view, we treat the microgeometry statistically and view the surface as reflecting (and refracting) light in multiple directions. The rougher the surface, the wider the cones of reflected and refracted directions will be.



Metals



Non-Metals

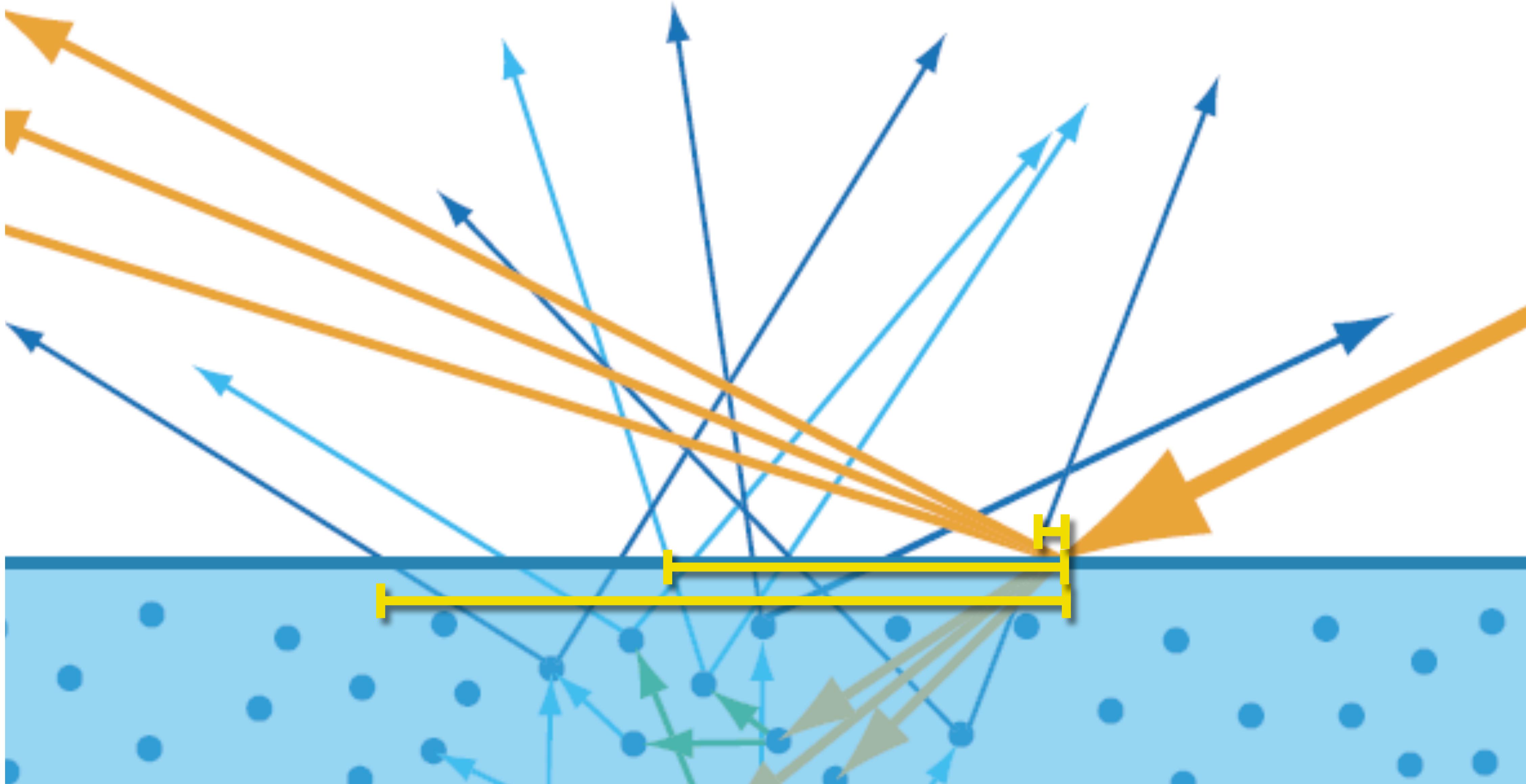


SIGGRAPH 2013

Image from "Real-Time Rendering, 3rd Edition", A K Peters 2008

23

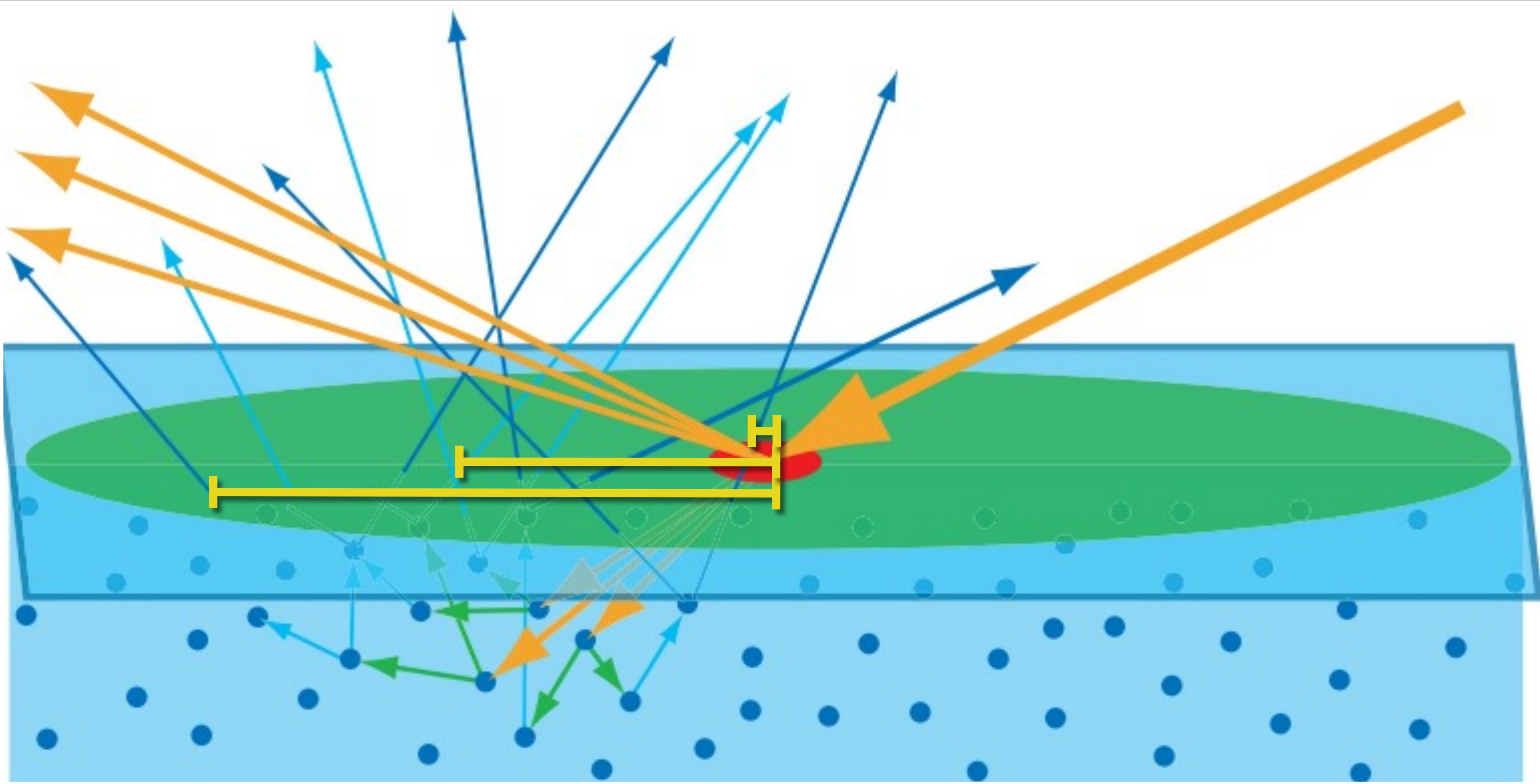
Non-metals behave like those cups of liquid we saw earlier - refracted light is scattered and / or absorbed to some degree. Unless the object is made out of a clear substance like glass or crystal, there will be enough scattering that some of the refracted light is scattered back out of the surface - these are the blue arrows you see coming out of the surface in various directions.



SIGGRAPH 2013



The re-emitted light comes out at varying distances (shown by the yellow bars) from the entry point. The distribution of distances depends on the density and properties of the scattering particles.



SIGGRAPH 2013

Image from "Real-Time Rendering, 3rd Edition", A K Peters 2008

If the pixel size (or shading sample area) is large (like the green circle) compared to the entry-exit distances, we can assume the distances are effectively zero for shading purposes.

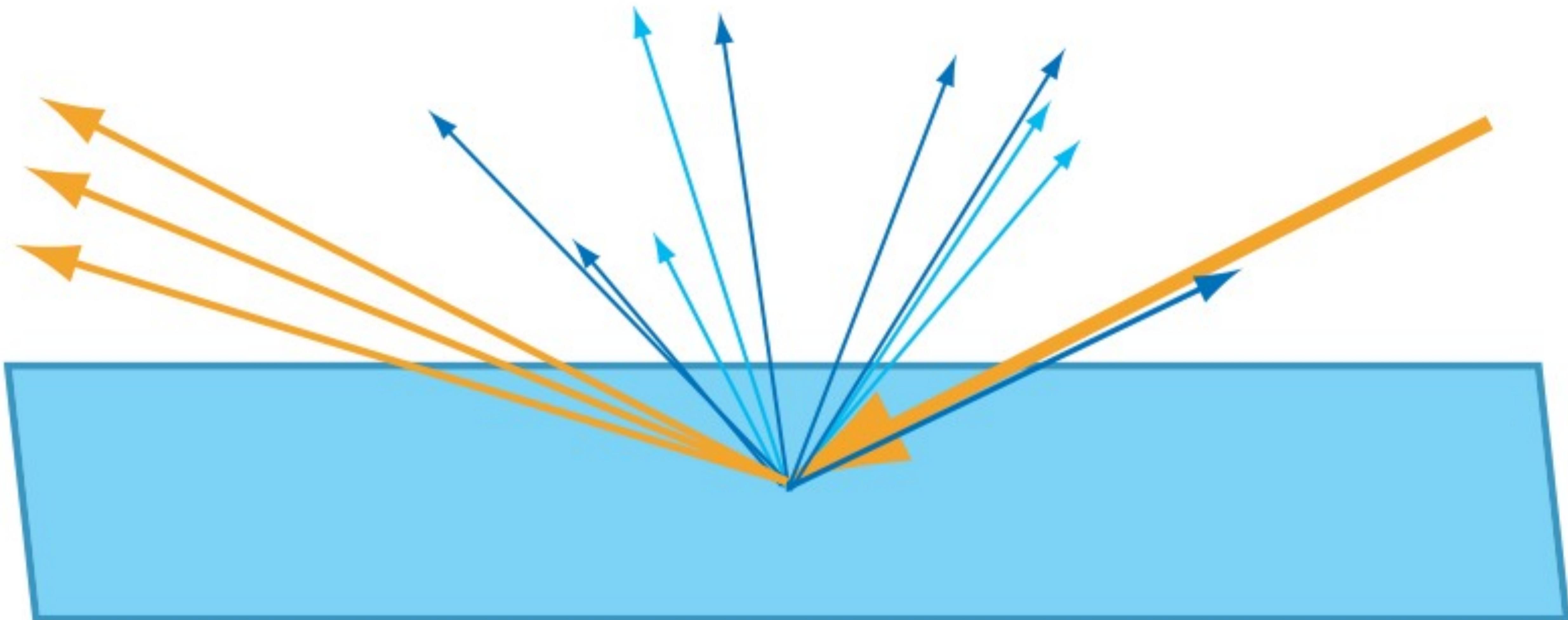


Image from "Real-Time Rendering, 3rd Edition", A K Peters 2008

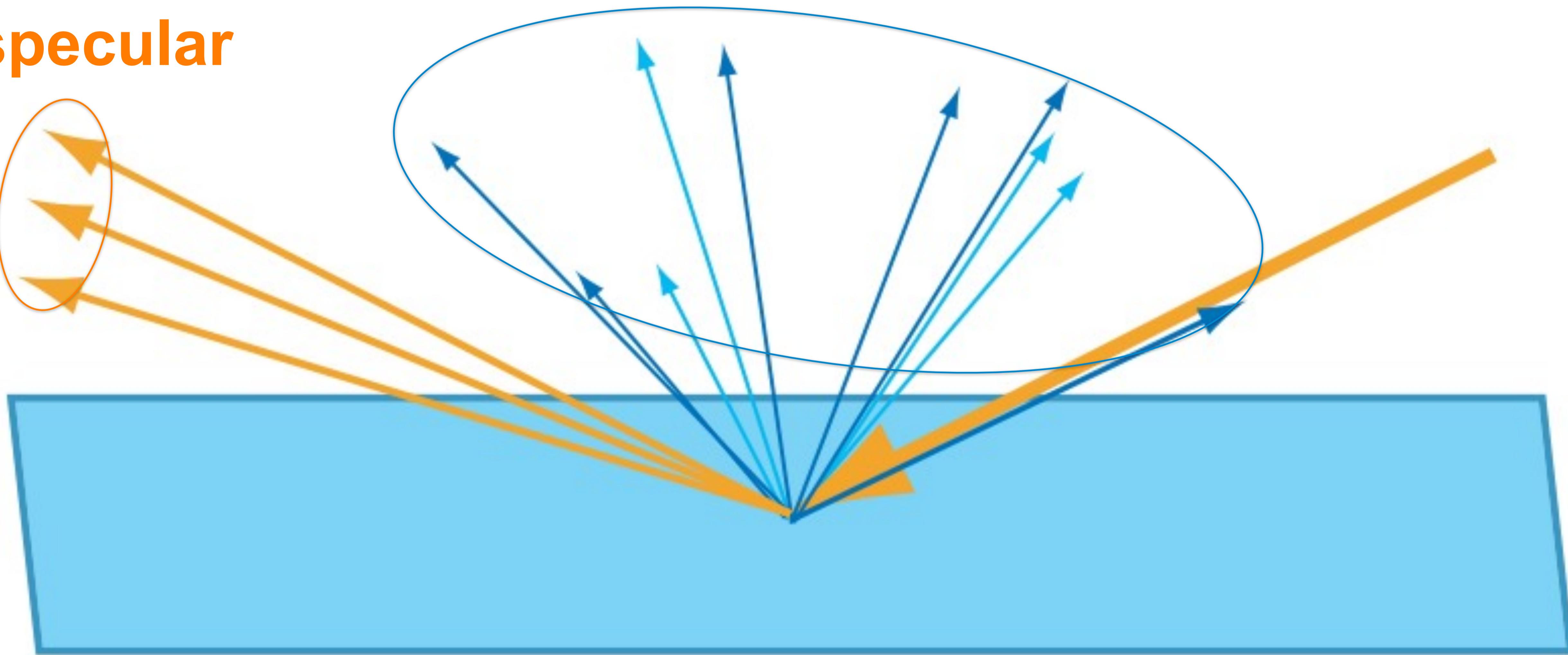
SIGGRAPH 2013

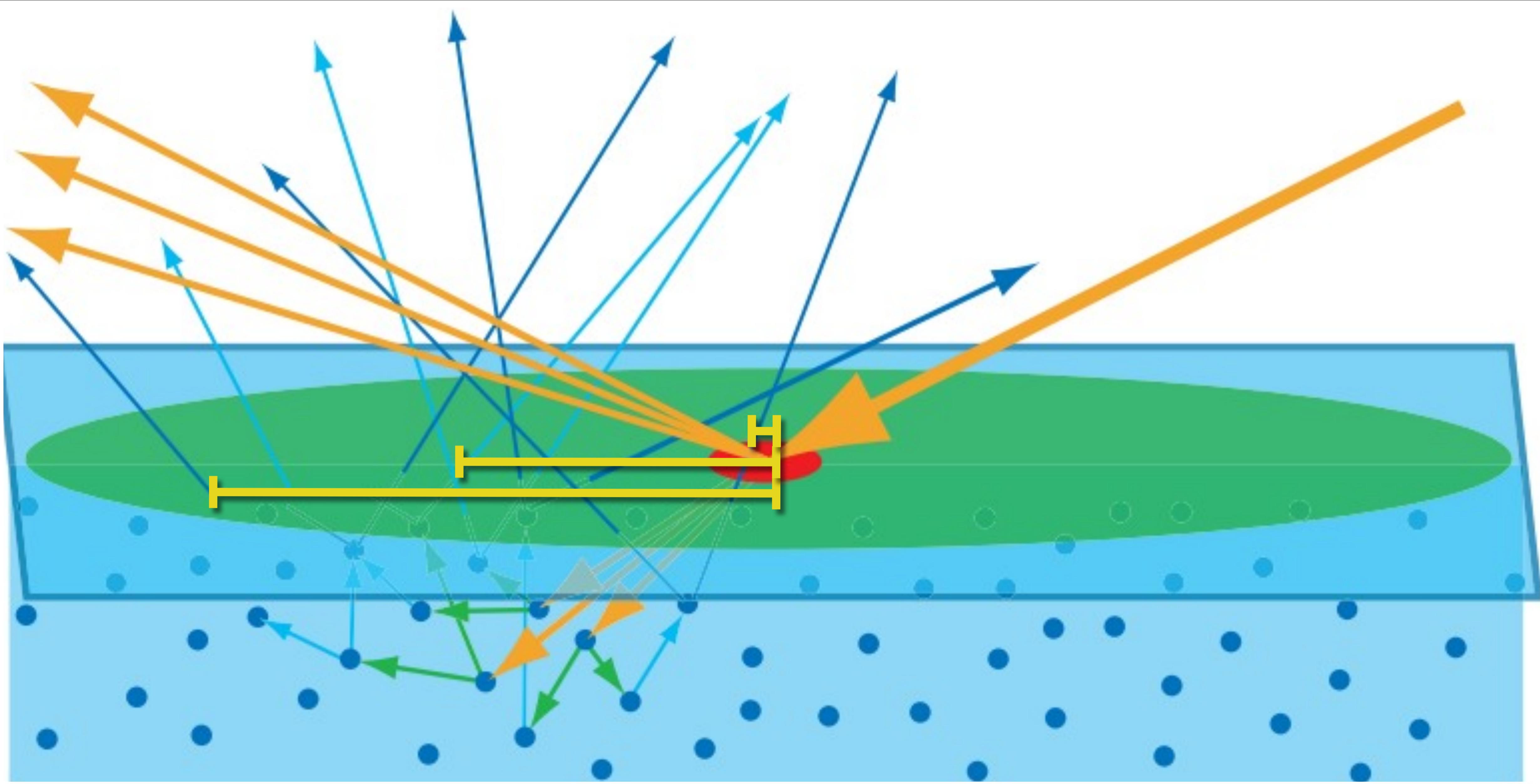


By ignoring the entry-to-exit distance, we can then compute all shading locally at a single point. The shaded color is only affected by light hitting that surface point.

specular

diffuse





SIGGRAPH 2013

If the pixel is small compared to the entry-exit distances (like the red circle), then special “subsurface scattering” rendering techniques are needed. It’s important to note that even regular diffuse shading is a result of subsurface scattering - the difference is the shading resolution compared to the scattering distance.

Physics → Math



So far we've discussed the physics of light/matter interactions. To turn these physics into mathematical models that can be used for shading, the first step is to quantify light as a number.

Radiance



Radiometry is the measurement of light. Of the various radiometric quantities, we'll use *radiance*...

Radiance

Single Ray



...which measures the intensity of light along a single ray...

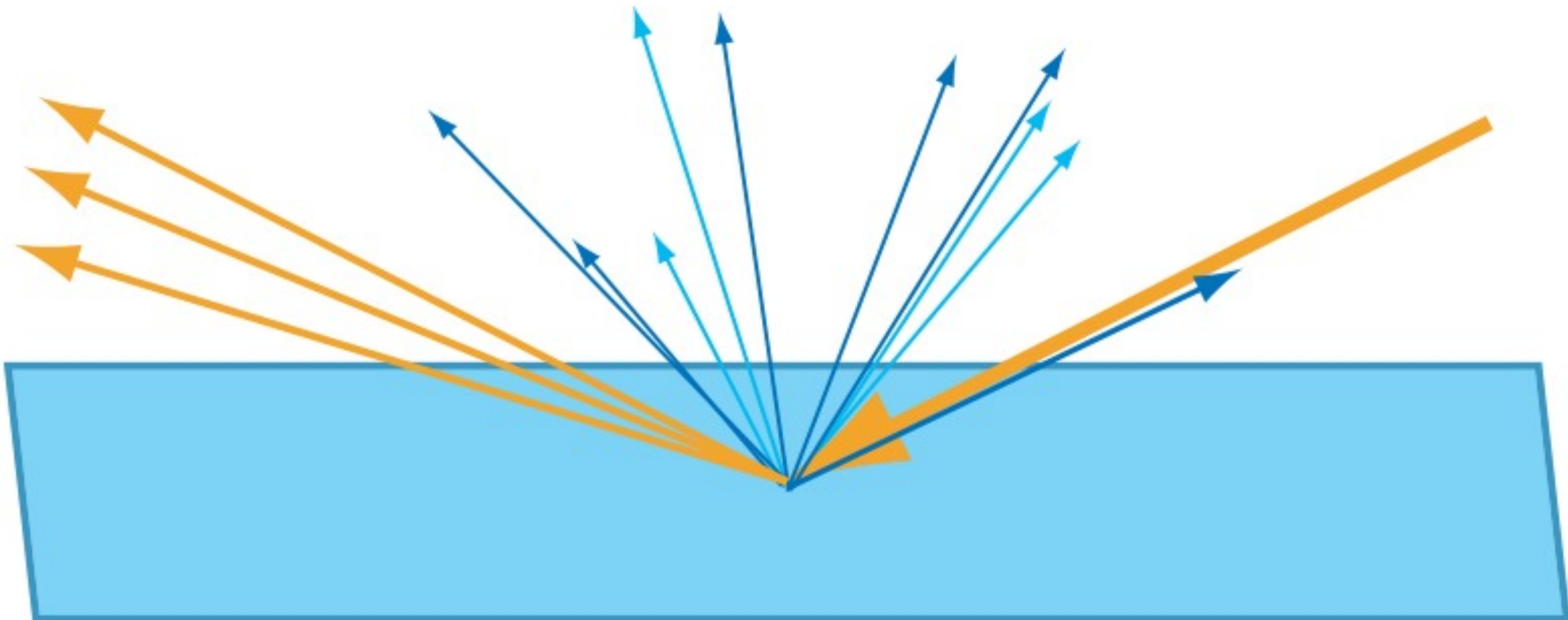
Radiance

Single Ray

Spectral/RGB

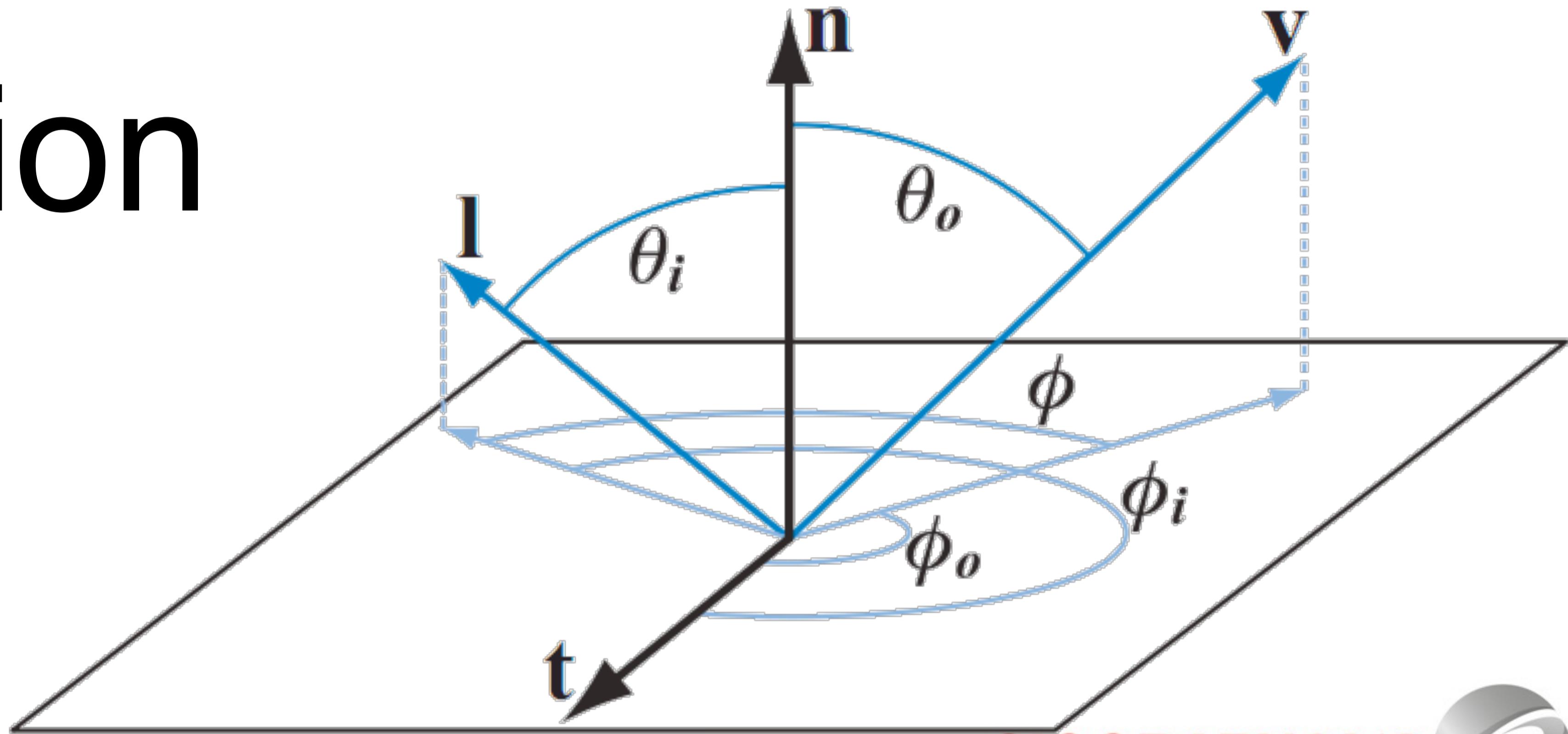


...Radiance is spectral (it varies with wavelength) - it's technically a continuous spectral power distribution but for production purposes it's represented as an RGB triplet.



Bidirectional Reflectance Distribution Function

$$f(\mathbf{l}, \mathbf{v})$$



SIGGRAPH 2013

Image from "Real-Time Rendering, 3rd Edition", A K Peters 2008

We represent this variation with the *BRDF*, a function of light direction \mathbf{l} and view direction \mathbf{v} . In principle, the BRDF is a function of the 3 or 4 angles shown in the figure. In practice, BRDF models use varying numbers of angles. Note that the BRDF is only defined for light and view vectors above the macroscopic surface; see the course notes for some tips on how to handle other cases.

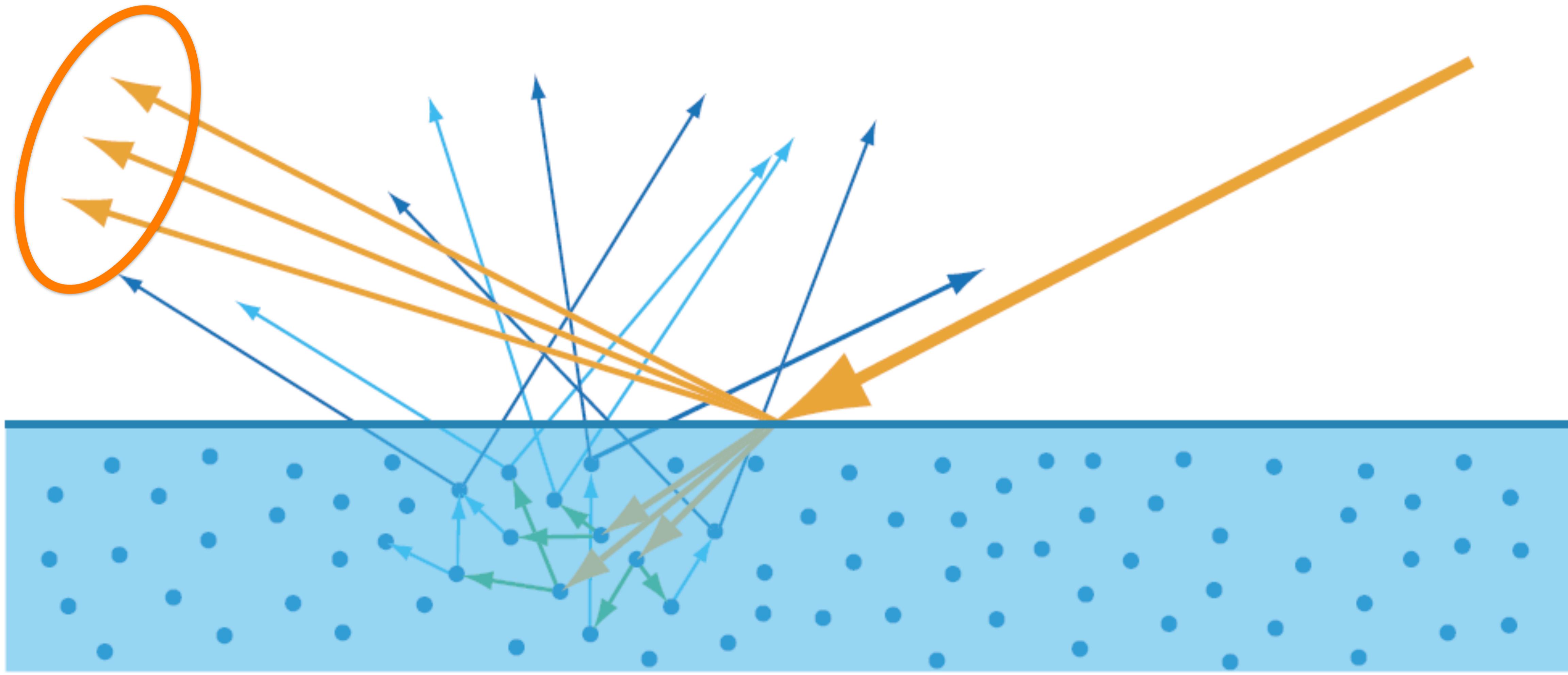
The Reflectance Equation

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



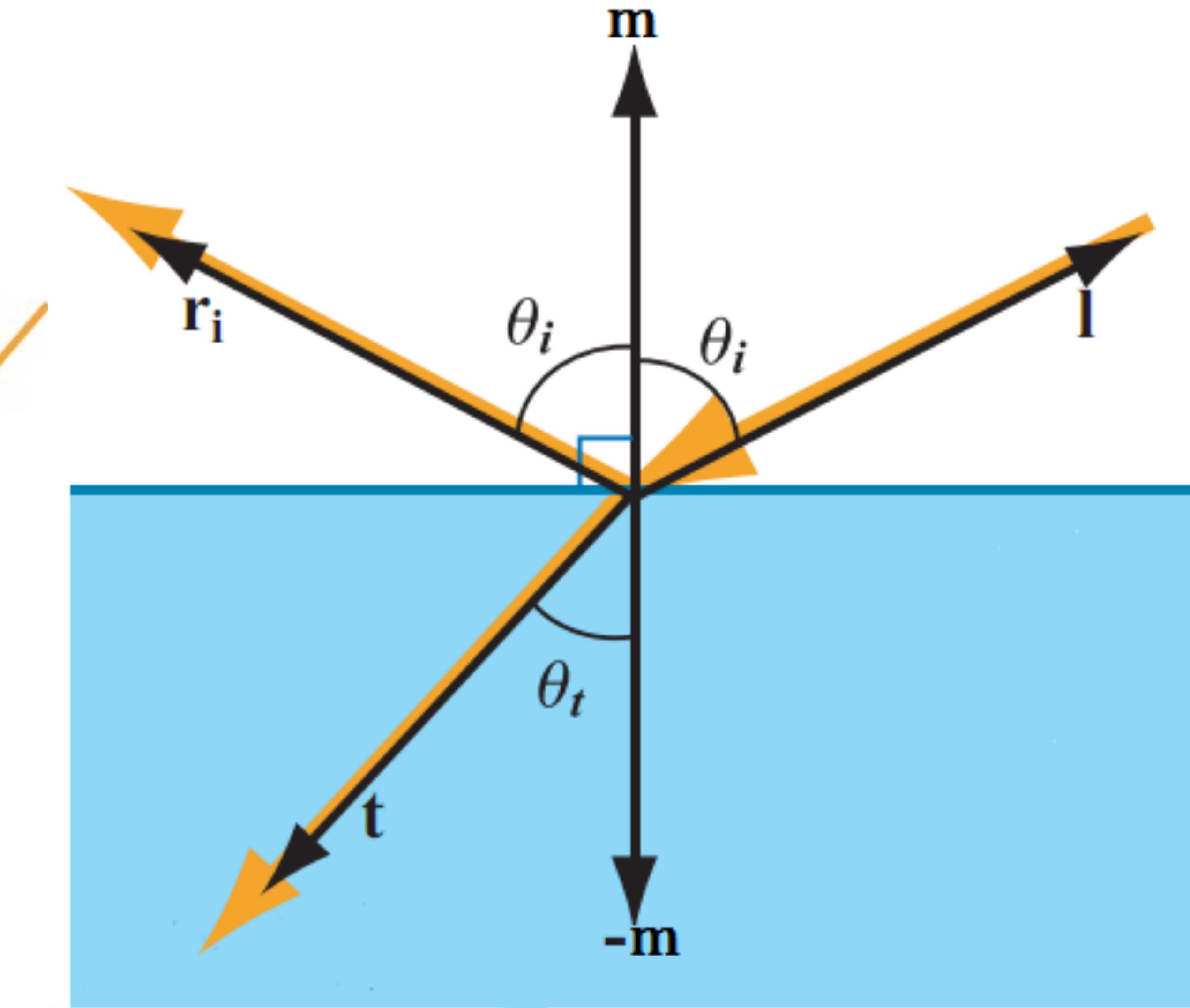
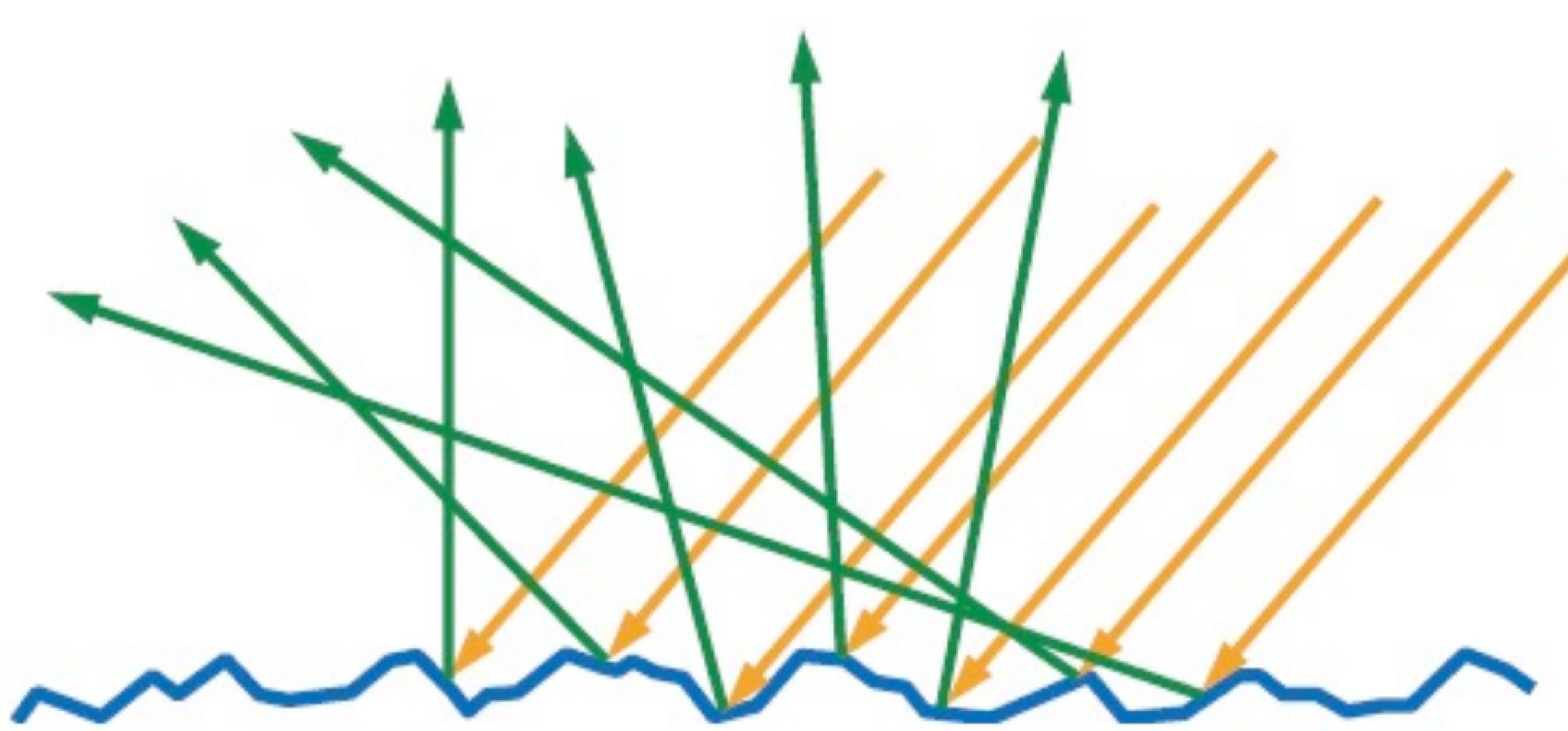
This scary-looking equation just says that outgoing radiance from a point equals the integral of incoming radiance times BRDF times a cosine factor, over the hemisphere of incoming directions. If you're not familiar with integrals you can think of this as a sort of weighted average over all incoming directions. The “X in circle” notation is from the *Real-Time Rendering* book - it means component-wise RGB multiplication.

Surface Reflection (Specular Term)



We'll start by looking at the specular term.

Microfacet Theory



Microfacet theory is a way to derive BRDFs for surface (or specular) reflection from general (non-optically flat) surfaces. It assumes the surface is composed of many *microfacets*. Each facet is a perfect mirror (optically flat), so it reflects each incoming ray of light into only one outgoing direction, which depends on the light direction \mathbf{I} and the microfacet normal \mathbf{m} .

The Half Vector

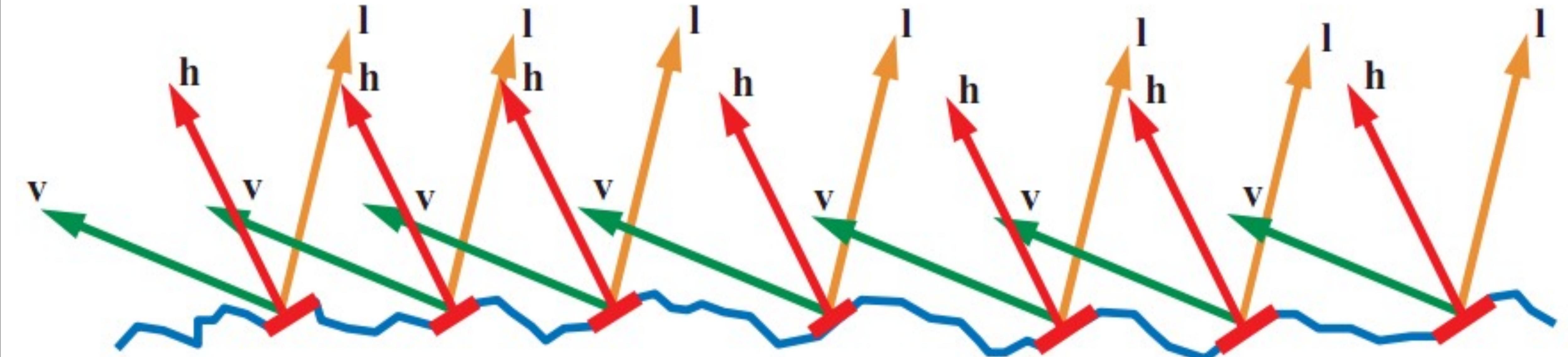


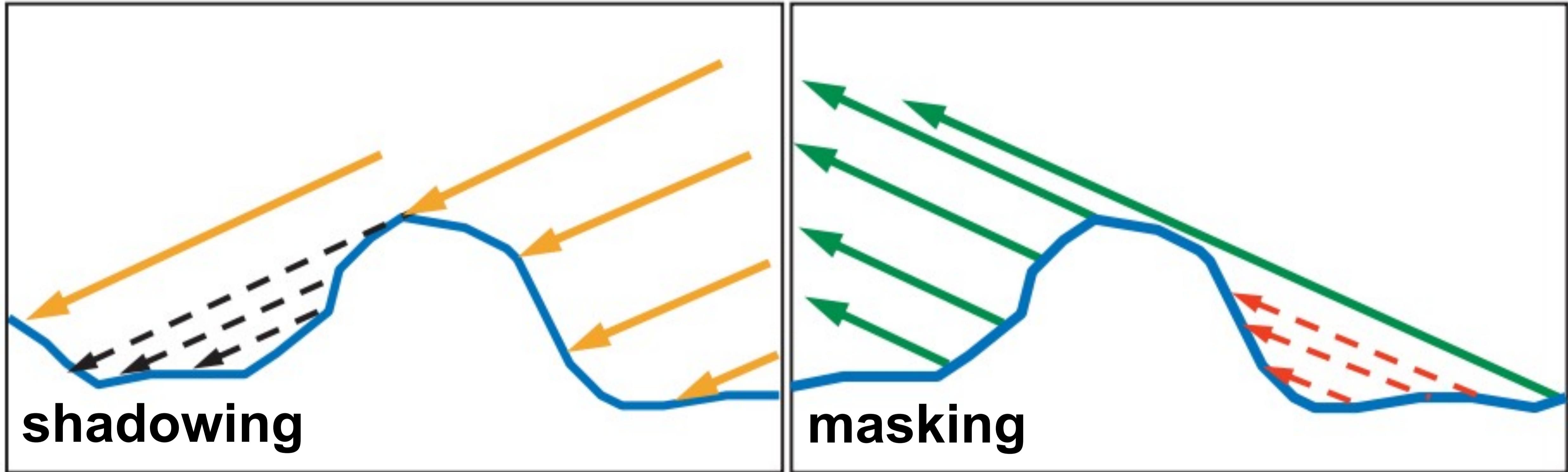
Image from "Real-Time Rendering, 3rd Edition", A K Peters 2008

SIGGRAPH 2013



Only those microfacets which happen to have their surface normal m oriented exactly halfway between l and v will reflect any visible light - this direction is the half-vector h .

Shadowing and Masking



Images from “Real-Time Rendering, 3rd Edition”, A K Peters 2008

SIGGRAPH 2013



Not all microfacets with $\mathbf{m} = \mathbf{h}$ will contribute - some will be blocked by other microfacets from either the light direction (*shadowing*) or the view direction (*masking*).

Multiple Surface Bounces

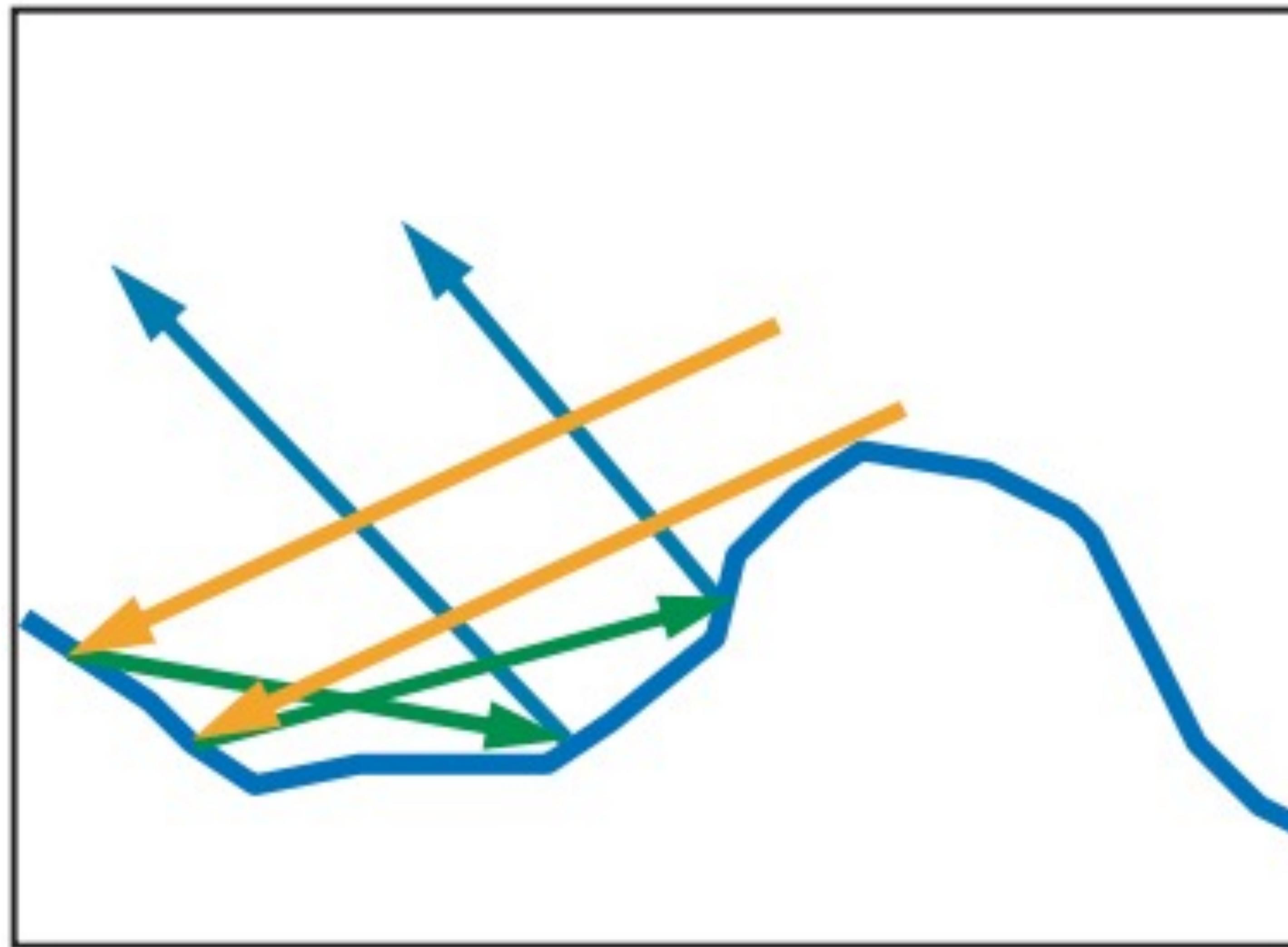


Image from “Real-Time Rendering, 3rd Edition”, A K Peters 2008

In reality, blocked light continues to bounce; some will eventually contribute to the BRDF. Microfacet BRDFs ignore this, so effectively they assume all blocked light is lost.

Microfacet BRDF

$$f(l, v) = \frac{F(l, h)G(l, v, h)D(h)}{4(n \cdot l)(n \cdot v)}$$



This is a general microfacet BRDF. I'll go over its various parts, explaining each.

Fresnel Reflectance

$$f(\mathbf{l}, \mathbf{v}) = \frac{F(\mathbf{l}, \mathbf{h})G(\mathbf{l}, \mathbf{v}, \mathbf{h})D(\mathbf{h})}{4(\mathbf{n} \cdot \mathbf{l})(\mathbf{n} \cdot \mathbf{v})}$$



The Fresnel reflectance is the fraction of incoming light that is reflected (as opposed to refracted) from an optically flat surface of a given substance. It depends on the light direction and the surface (in this case microfacet) normal. This tells us how much of the light hitting the relevant microfacets (the ones facing in the half-angle direction) is reflected.

Fresnel Reflectance

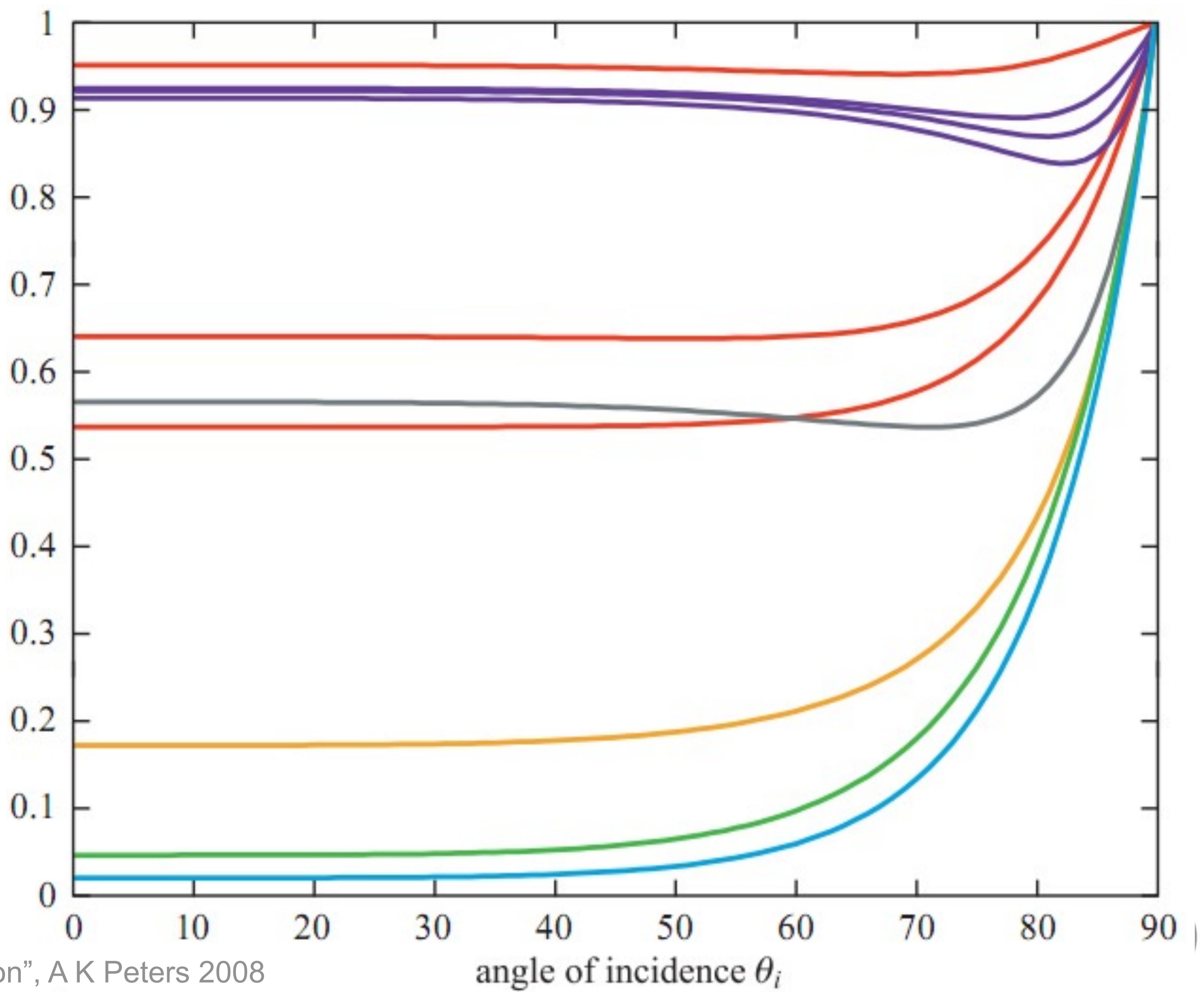
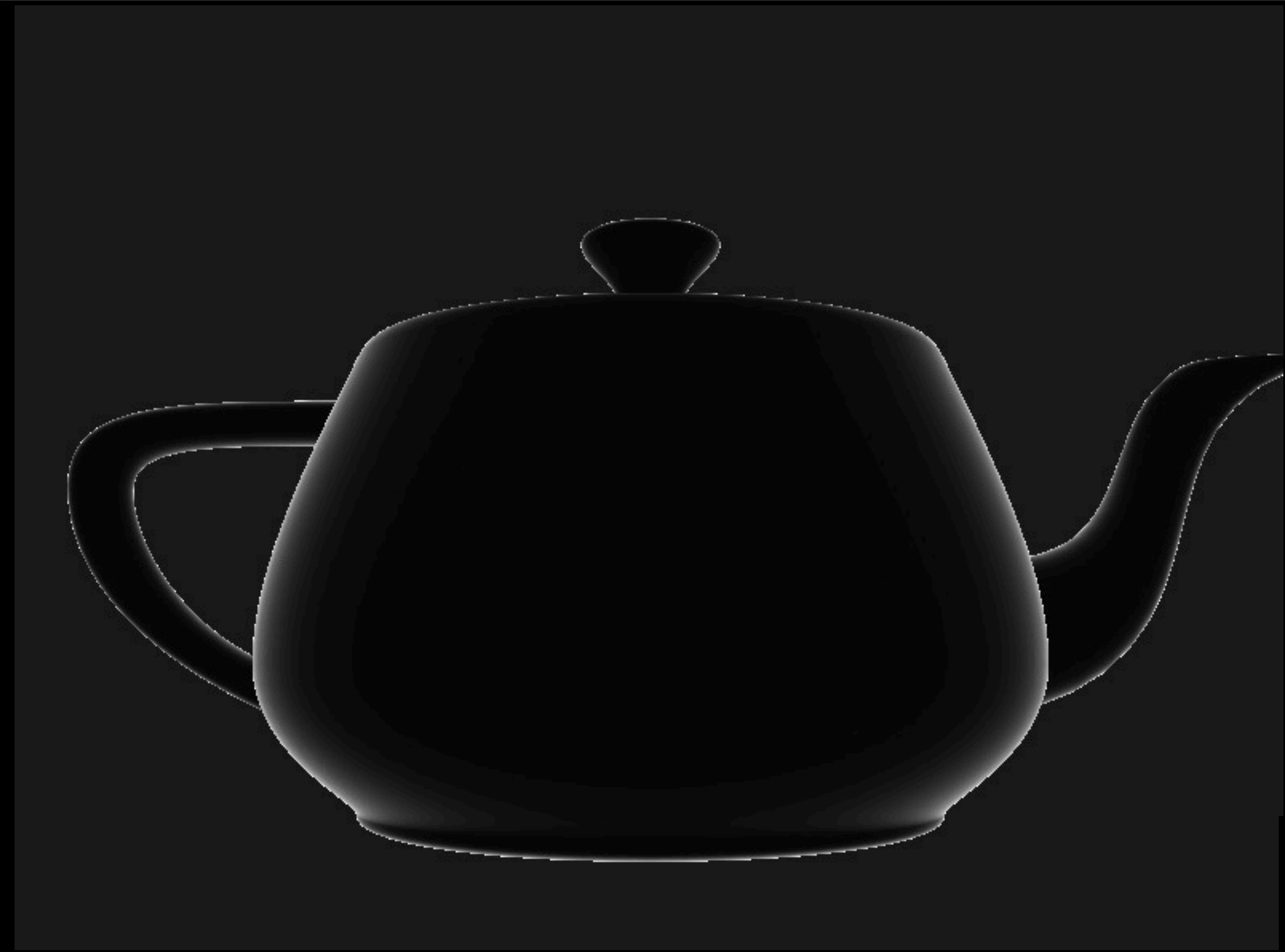


Image from "Real-Time Rendering, 3rd Edition", A K Peters 2008

angle of incidence θ_i

43

Fresnel reflectance (on the y-axis in this graph) depends on refraction index (in other words, what the object's made of) and light-to-normal angle (which is plotted here on the x-axis). In this graph, substances with three lines (copper & aluminum) have colored reflectance which is plotted separately for the R, G and B channels – the other substances, with one line, have uncolored reflectance.



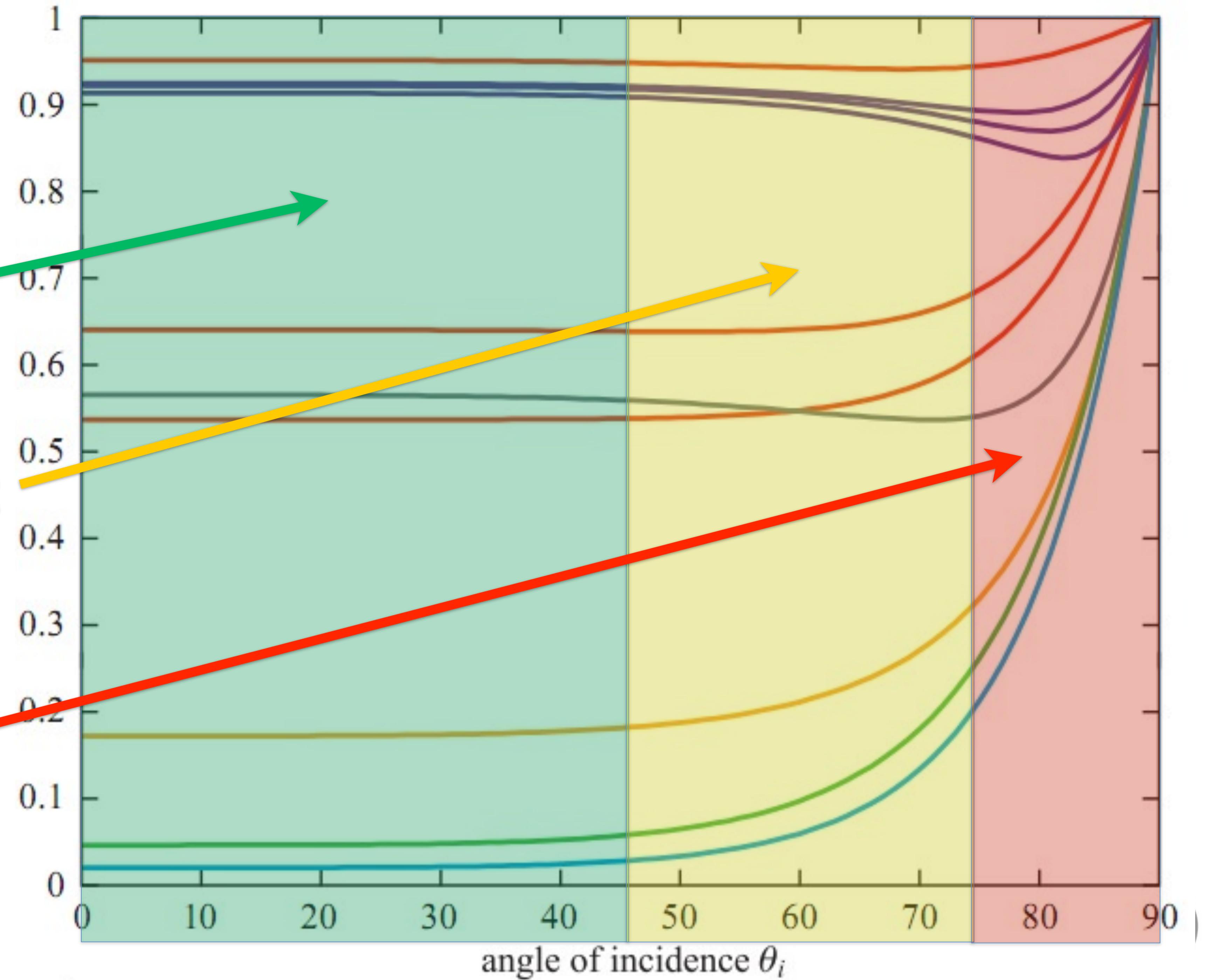
With an optically flat surface and non-directional lighting (like an overcast sky) the relevant angle for Fresnel reflectance is the one between the view and normal vectors. This image shows the Fresnel reflectance of glass (the green curve from the previous slide) over a 3D shape - see how the dark reflectance color in the center brightens to white at the edges.

Fresnel Reflectance

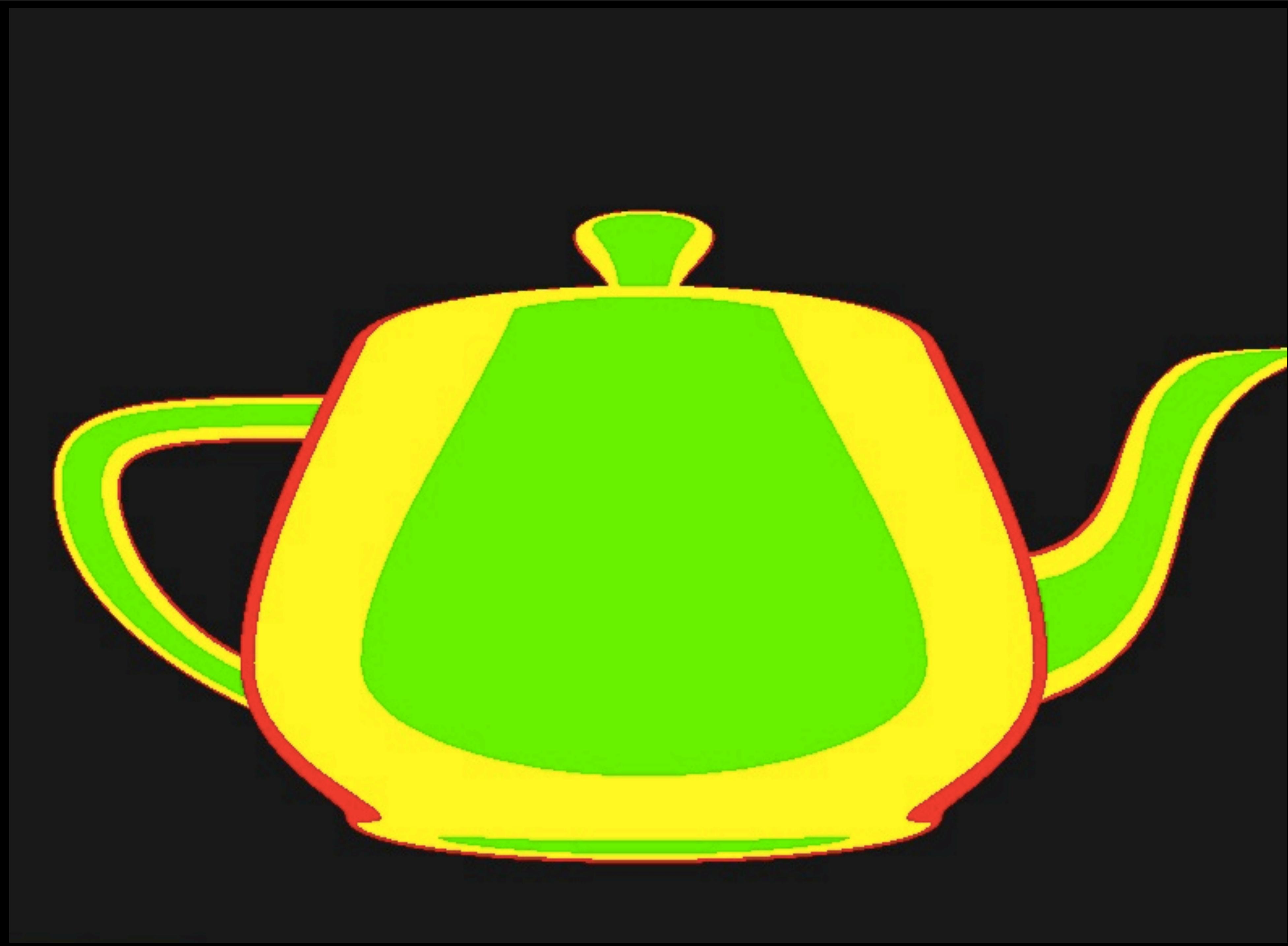
barely changes

changes somewhat

goes rapidly to 1



As angle increases, up to about 45 degrees (the green area on the graph) the Fresnel reflectance barely changes; afterwards it starts changing, first slowly (the yellow area, up to about 75 degrees) and then for very glancing angles (the red zone) it rapidly goes to 100% at all wavelengths.



Here's a visualization of the same zone colors over a 3D object. We can see that the vast majority of visible pixels are in the areas where the reflectance changes barely at all (green) or only slightly (yellow).

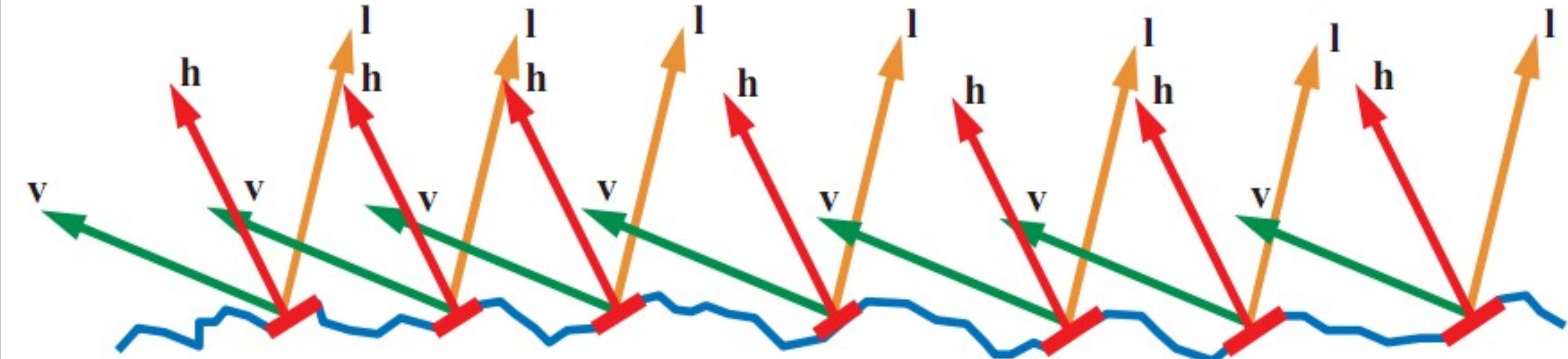


Image from "Real-Time Rendering, 3rd Edition", A K Peters 2008

SIGGRAPH 2013



Recall that in a microfacet BRDF the relevant normal direction is the **h** vector (only the microfacets with normals aligned to **h** are visible). This means that we need to use the angle between **v** and **h** for Fresnel reflectance (or **l** and **h** - it's the same angle).

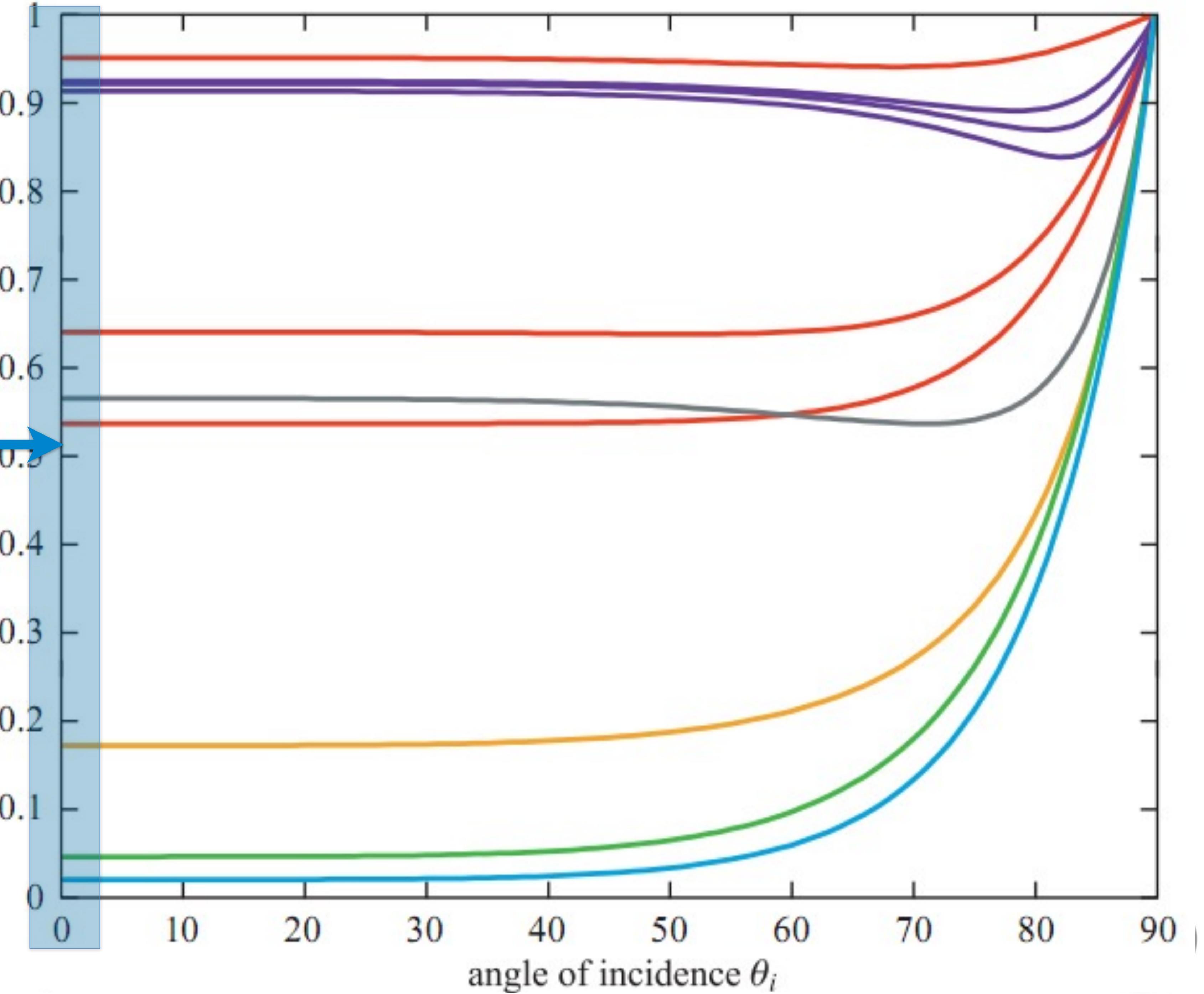
<DEMO> unlike Fresnel for surface normals, <SWITCH MODE> visualizing Fresnel zones for the **h** vector it seems like the whole object can be in the yellow or even the red zone <MOVE LIGHT AROUND>. But when we combine Fresnel with the rest of the BRDF <SWITCH MODE> then we can see that green still predominates, and red can only be seen - rarely - at the object edges <MOVE LIGHT AROUND>.

Fresnel Reflectance

$F(0^\circ)$

Is the surface's characteristic specular color:

c_{spec}



Since the reflectance over most angles is close to that at normal incidence, the normal-incidence reflectance - $F()$ at 0 degrees - is the surface's characteristic specular color.

Normal-Incidence Fresnel for Metals

Metal	$F(0^\circ)$ (Linear)	$F(0^\circ)$ (sRGB)	Color
Gold	1.00,0.71,0.29	1.00,0.86,0.57	
Silver	0.95,0.93,0.88	0.98,0.97,0.95	
Copper	0.95,0.64,0.54	0.98,0.82,0.76	
Iron	0.56,0.57,0.58	0.77,0.78,0.78	
Aluminum	0.91,0.92,0.92	0.96,0.96,0.97	



Table from “Real-Time Rendering, 3rd Edition”, A K Peters 2008

Metals have relatively bright specular colors. Note that metals have no subsurface term, so the surface Fresnel reflectance is the material’s only source of color. The “linear” and “sRGB” columns refer to whether the values are in linear or gamma space.

Normal-Incidence Fresnel for Non-Metals

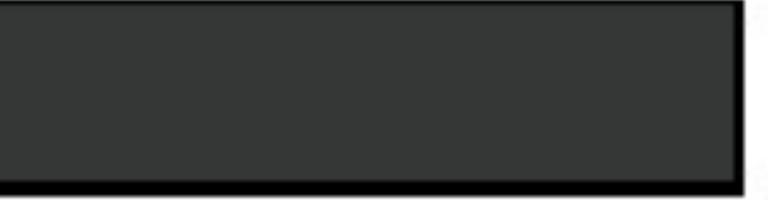
Insulator	$F(0^\circ)$ (Linear)	$F(0^\circ)$ (sRGB)	Color
Water	0.02,0.02,0.02	0.15,0.15,0.15	
Plastic / Glass (Low)	0.03,0.03,0.03	0.21,0.21,0.21	
Plastic High	0.05,0.05,0.05	0.24,0.24,0.24	
Glass (High) / Ruby	0.08,0.08,0.08	0.31,0.31,0.31	
Diamond	0.17,0.17,0.17	0.45,0.45,0.45	

Table from “Real-Time Rendering, 3rd Edition”, A K Peters 2008



Note that for non-metals the specular colors are achromatic (gray) and are relatively dark (especially if excluding gems and crystals). Most non-metals also have a subsurface (or diffuse) color in addition to their Fresnel (or specular) reflectance.

The Schlick Approximation to Fresnel

- Pretty accurate, cheap, parameterized by c_{spec}

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

- For microfacet BRDFs ($m = h$):

$$F_{\text{Schlick}}(c_{\text{spec}}, l, h) = c_{\text{spec}} + (1 - c_{\text{spec}})(1 - (l \cdot h))^5$$



The Schlick approximation to Fresnel is commonly used. It is cheap and reasonably accurate - more importantly it has a convenient parameter (normal incidence Fresnel reflectance, or specular color). As we saw previously, when using it in microfacet BRDFs the h vector is used in place of the normal.

Normal Distribution Function

$$f(\mathbf{l}, \mathbf{v}) = \frac{F(\mathbf{l}, \mathbf{h})G(\mathbf{l}, \mathbf{v}, \mathbf{h})}{4(\mathbf{n} \cdot \mathbf{l})(\mathbf{n} \cdot \mathbf{v})} D(\mathbf{h})$$



The next part of the microfacet BRDF we will discuss is the microfacet normal distribution function, or NDF. The NDF gives the concentration of microfacet normals pointing in a given direction (in this case, the half-angle direction). The NDF determines the size and shape of the highlight.

$$D_p(\mathbf{m}) = \frac{\alpha_p + 2}{2\pi} (\mathbf{n} \cdot \mathbf{m})^{\alpha_p}$$

$$D_{uabc}(\mathbf{m}) = \frac{1}{(1 + \alpha_{abc1} (1 - (\mathbf{n} \cdot \mathbf{m})))^{\alpha_{abc2}}}$$

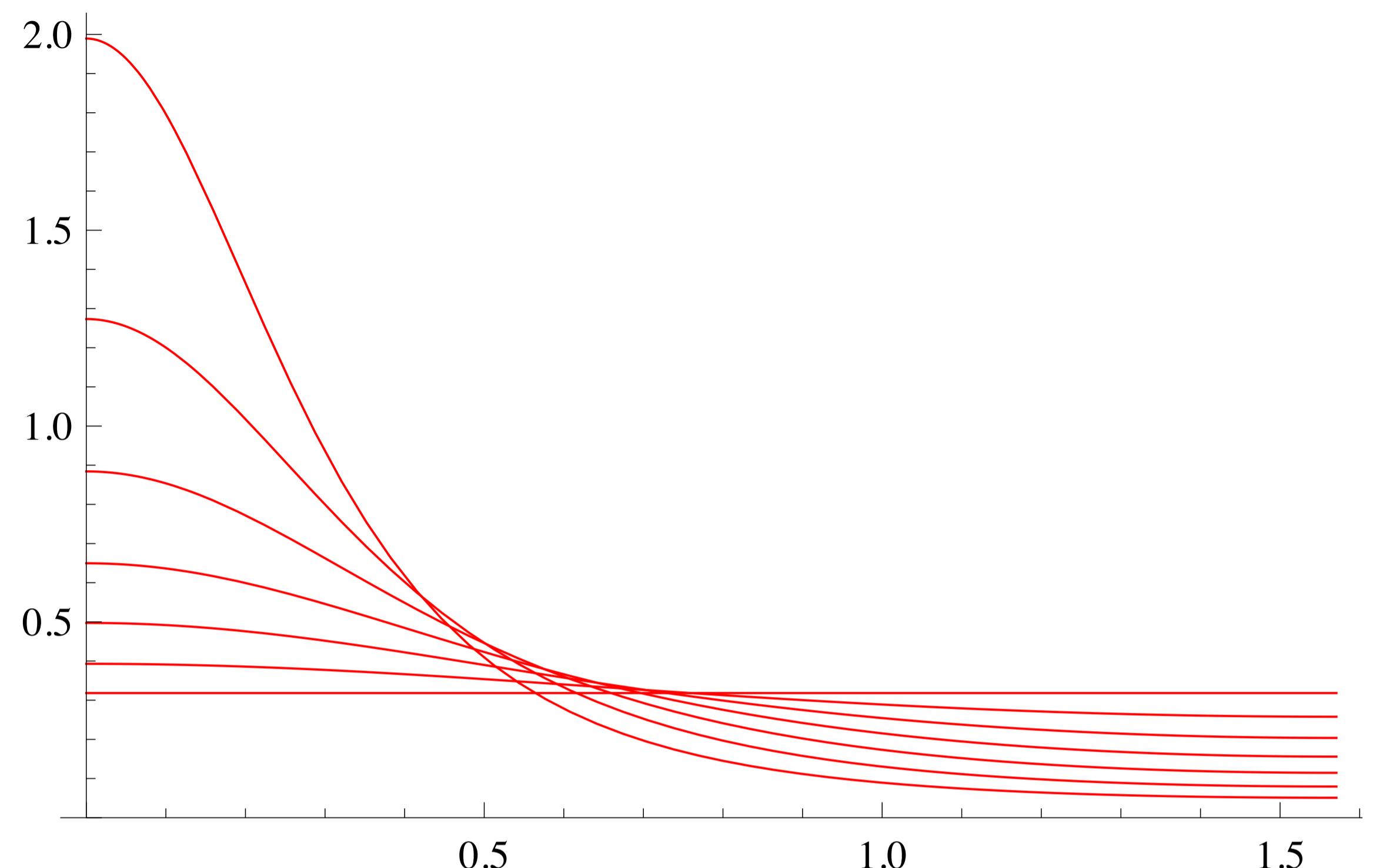
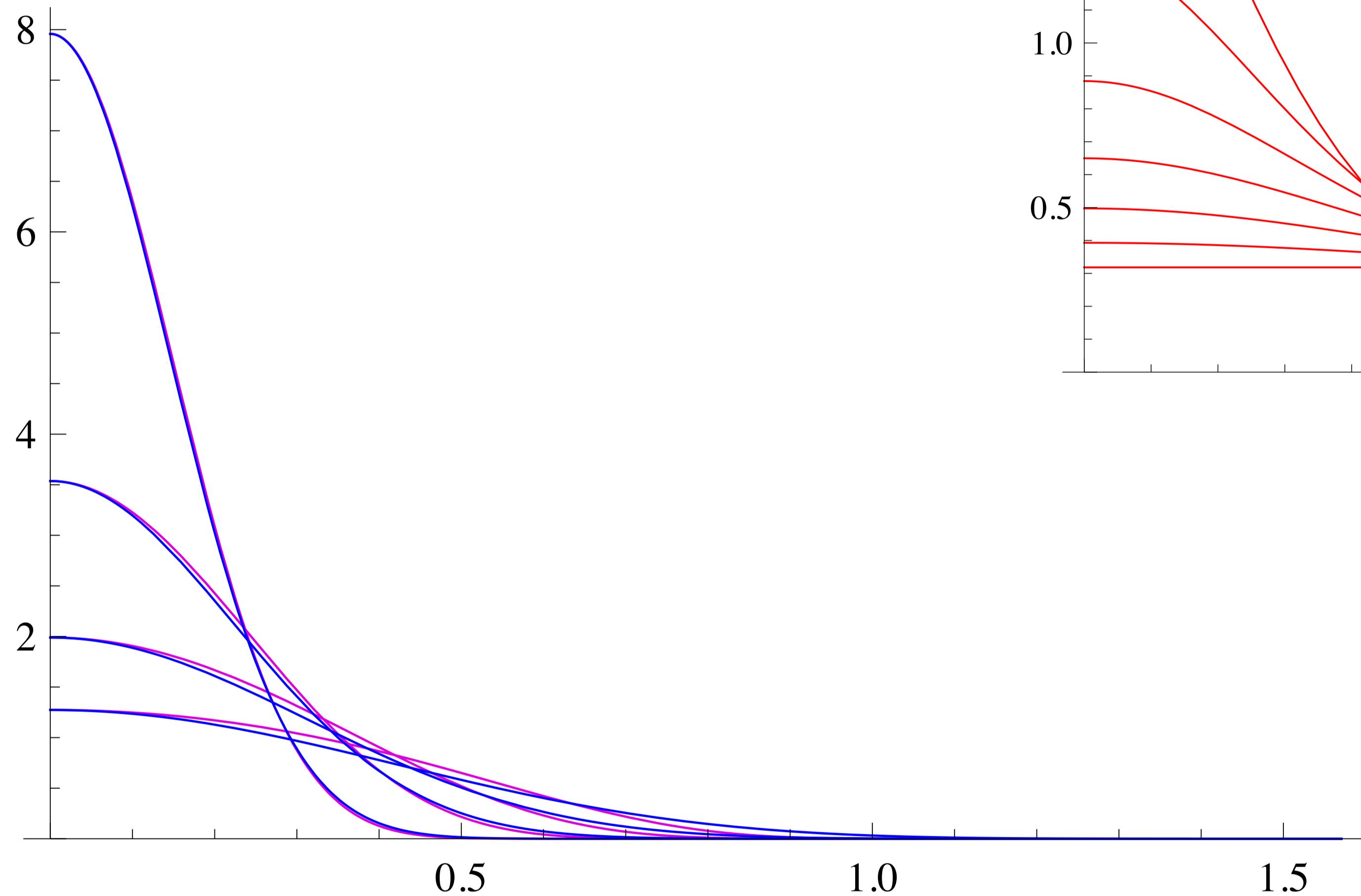
$$D_{tr}(\mathbf{m}) = \frac{\alpha_{tr}^2}{\pi ((\mathbf{n} \cdot \mathbf{m})^2 (\alpha_{tr}^2 - 1) + 1)^2}$$

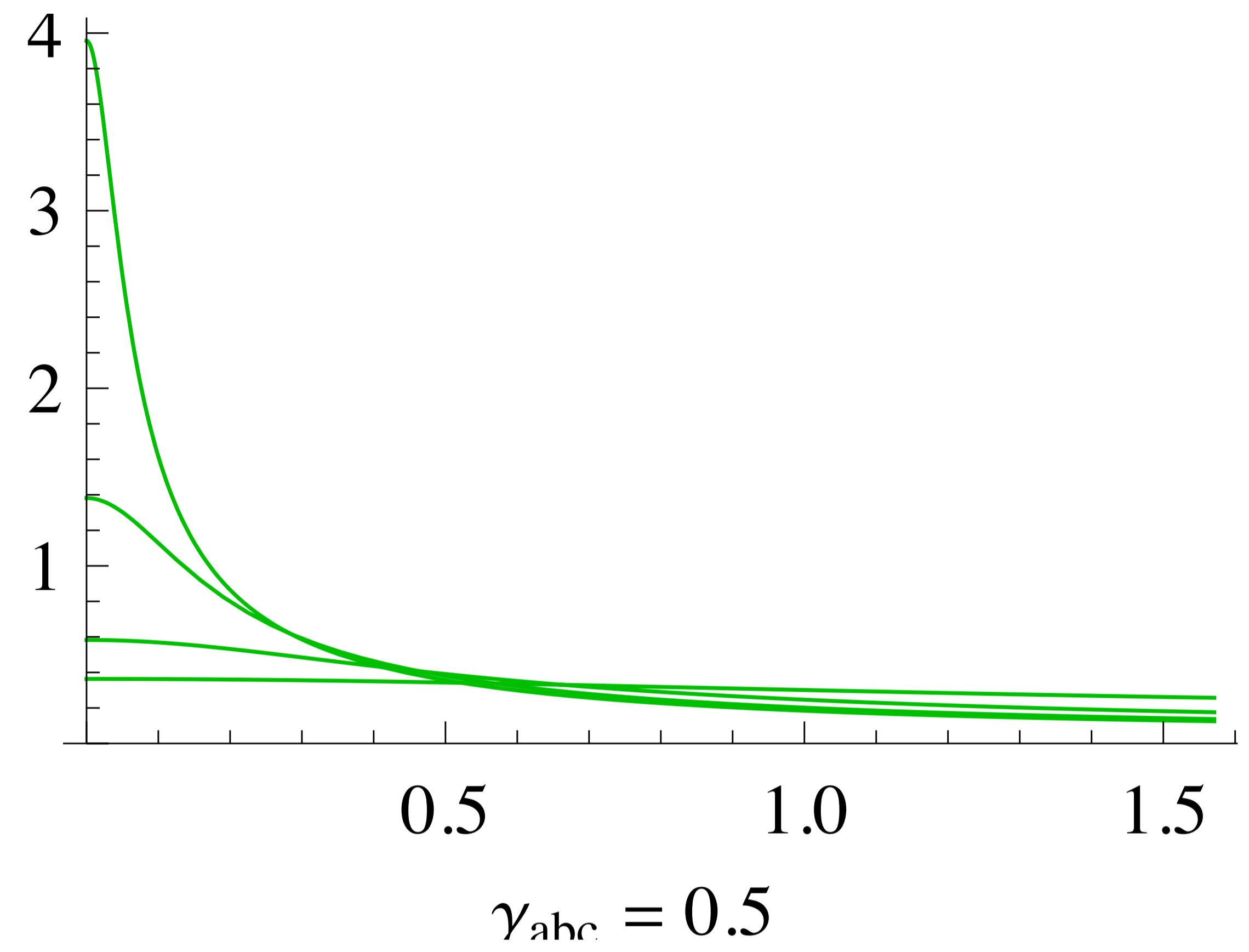
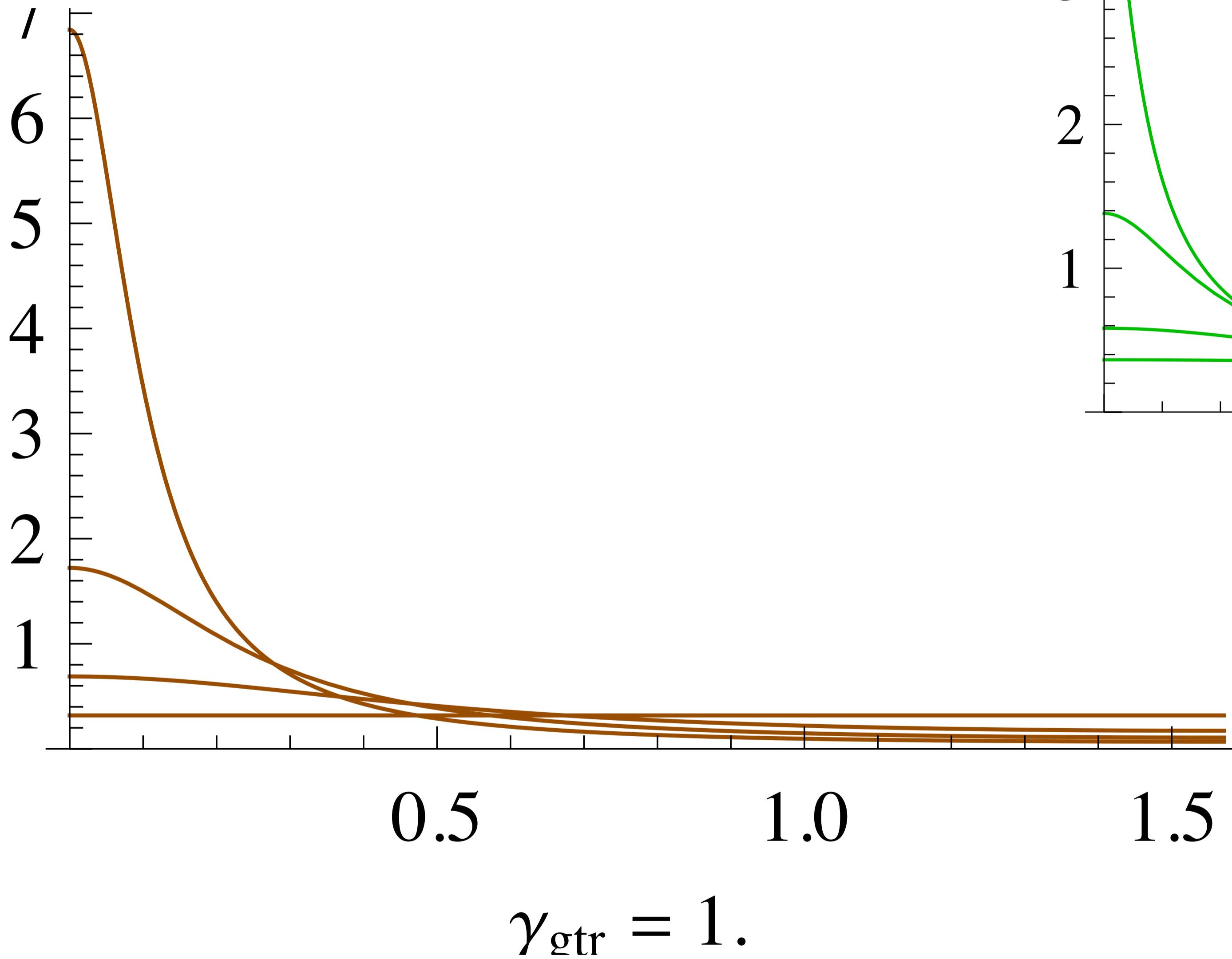
$$D_b(\mathbf{m}) = \frac{1}{\pi \alpha_b^2 (\mathbf{n} \cdot \mathbf{m})^4} e^{-\left(\frac{1 - (\mathbf{n} \cdot \mathbf{m})^2}{\alpha_b^2 (\mathbf{n} \cdot \mathbf{m})^2}\right)}$$

$$D_{sgd}(\mathbf{m}) = \frac{p22 \left[\frac{1 - (\mathbf{n} \cdot \mathbf{m})^2}{(\mathbf{n} \cdot \mathbf{m})^2} \right]}{\pi (\mathbf{n} \cdot \mathbf{m})^4}$$



The course notes detail various options for NDFs.





Others have a more “spiky” shape with long tails, leading to sharp highlights with “halos” around them.

Geometry Factor

$$f(l, v) = \frac{F(l, h) G(l, v, h) D(h)}{4(n \cdot l)(n \cdot v)}$$



The geometry factor gives the chance that a microfacet with a given orientation (again, the half-angle direction is the relevant one) is lit and visible (in other words, not shadowed and/or masked) from the given light and view directions.

The Visibility Term

$$f(l, v) = \frac{F(l, h)G(l, v, h)D(h)}{4(n \cdot l)(n \cdot v)}$$

$$V(l, v) = \frac{G(l, v, h)}{(n \cdot l)(n \cdot v)}$$

In some cases, expressions are found for the geometry factor divided by the n-dot-l-times-n-dot-v “foreshortening term”. We’ll call this combined term the “visibility term”.

Simplest Visibility Term

$$\frac{G(l, v, h)}{(n \cdot l)(n \cdot v)} = 1$$

Equivalent to:

$$G_{\text{implicit}}(l_c, v, h) = (n \cdot l_c)(n \cdot v)$$

Some BRDFs have no visibility term, effectively equating it to one. This implies an “implicit” geometry factor equal to $n \cdot l$ times $n \cdot v$. It behaves as expected, going from one when view and light are in the normal direction, to zero when either is at 90 degrees. And it’s “cheaper than free” in a sense. But it darkens too fast compared to real surfaces and it isn’t affected by roughness, which is implausible.

$$G_{ct}(\mathbf{l}, \mathbf{v}, \mathbf{h}) = \min \left(1, \frac{2(\mathbf{n} \cdot \mathbf{h})(\mathbf{n} \cdot \mathbf{v})}{(\mathbf{v} \cdot \mathbf{h})}, \frac{2(\mathbf{n} \cdot \mathbf{h})(\mathbf{n} \cdot \mathbf{l})}{(\mathbf{v} \cdot \mathbf{h})} \right)$$

$$\frac{G_{ct}(\mathbf{l}, \mathbf{v}, \mathbf{h})}{(\mathbf{n} \cdot \mathbf{l})(\mathbf{n} \cdot \mathbf{v})} \approx \frac{1}{(\mathbf{l} \cdot \mathbf{h})^2}$$

$$G_s(\mathbf{l}, \mathbf{v}, \mathbf{h}) = G_{s1}(\mathbf{l}, \mathbf{h})G_{s1}(\mathbf{v}, \mathbf{h})$$

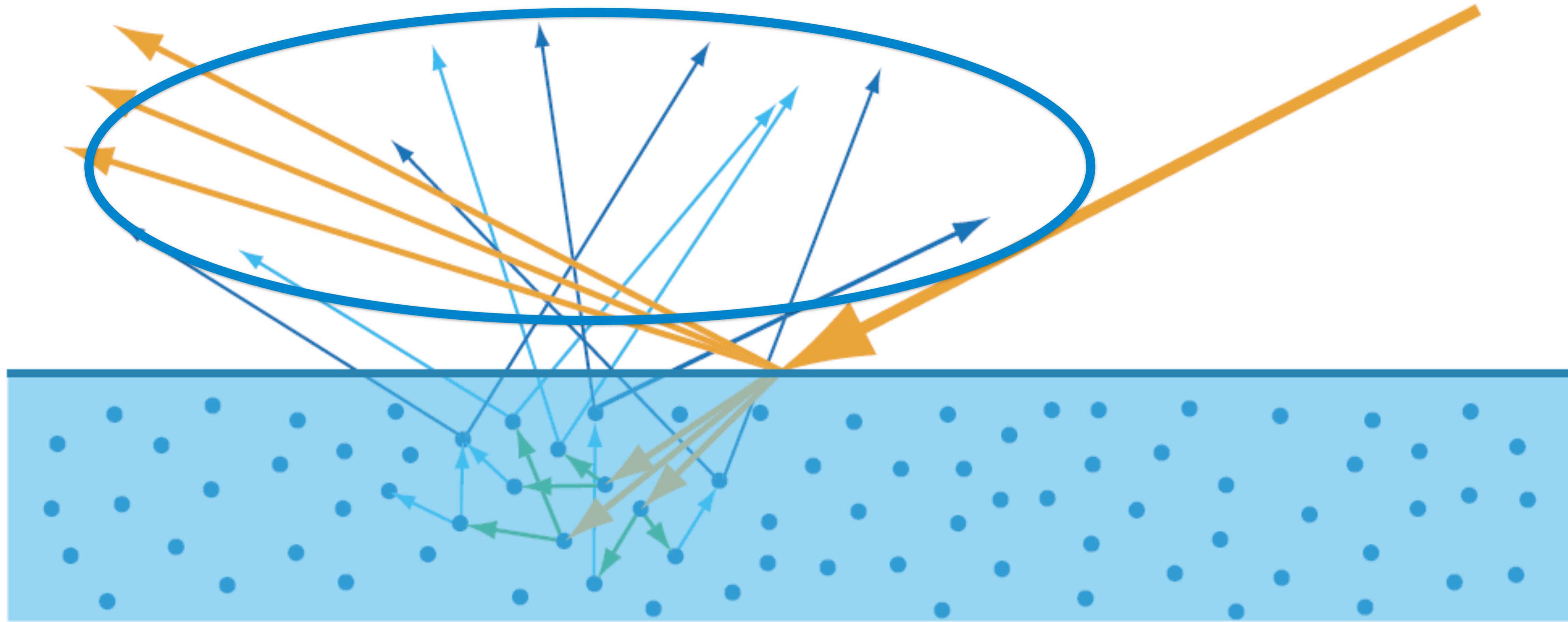


The course notes also discuss some other options for geometry factors. The best of these are influenced by the roughness, or better, by the exact shape of the NDF.

$$f(\mathbf{l}, \mathbf{v}) = \frac{F(\mathbf{l}, \mathbf{h})G(\mathbf{l}, \mathbf{v}, \mathbf{h})D(\mathbf{h})}{4(\mathbf{n} \cdot \mathbf{l})(\mathbf{n} \cdot \mathbf{v})}$$

Putting it all together, we see that the BRDF is proportional to the concentration of active microfacets (the ones with normals aligned with \mathbf{h}) times their visibility times their Fresnel reflectance. The rest of the BRDF (in the denominator) consists of correction factors relating to the various frames involved (light frame, view frame, local surface frame).

Subsurface Reflection (Diffuse Term)



SIGGRAPH 2013



Until now we've been focusing on the specular, or surface, reflection term. Next, we'll take a quick look at the diffuse (or subsurface) term.

Lambert

- Constant value ($\mathbf{n} \cdot \mathbf{l}$ is part of reflection equation):

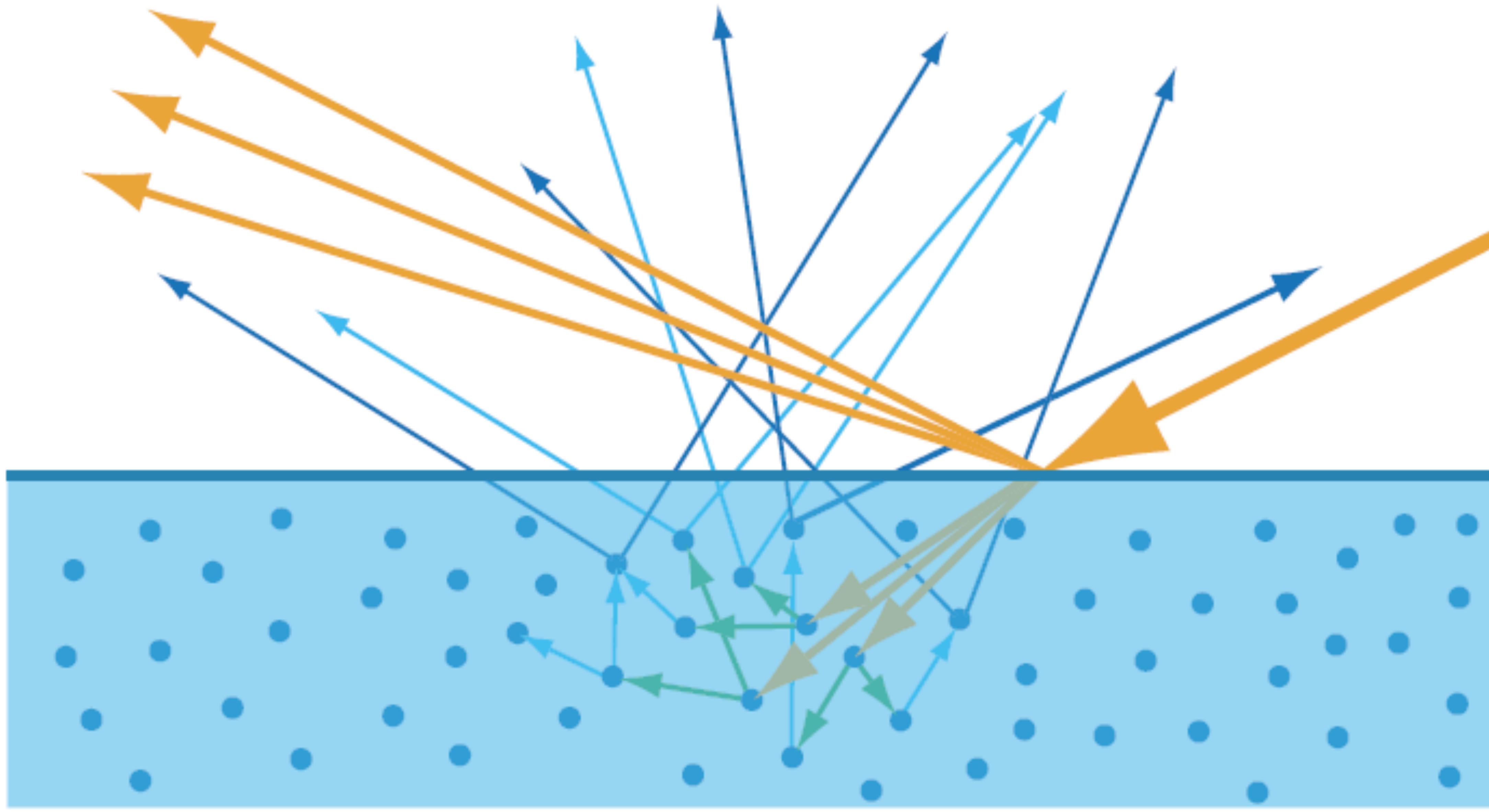
$$f_{\text{Lambert}}(\mathbf{l}, \mathbf{v}) = \frac{c_{\text{diff}}}{\pi}$$

- c_{diff} : fraction of light reflected, or diffuse color



The Lambert model is the most common diffuse term used in game and film production. By itself, it's the simplest possible BRDF - a constant value. The well-known cosine factor is part of the reflection equation, not the BRDF.

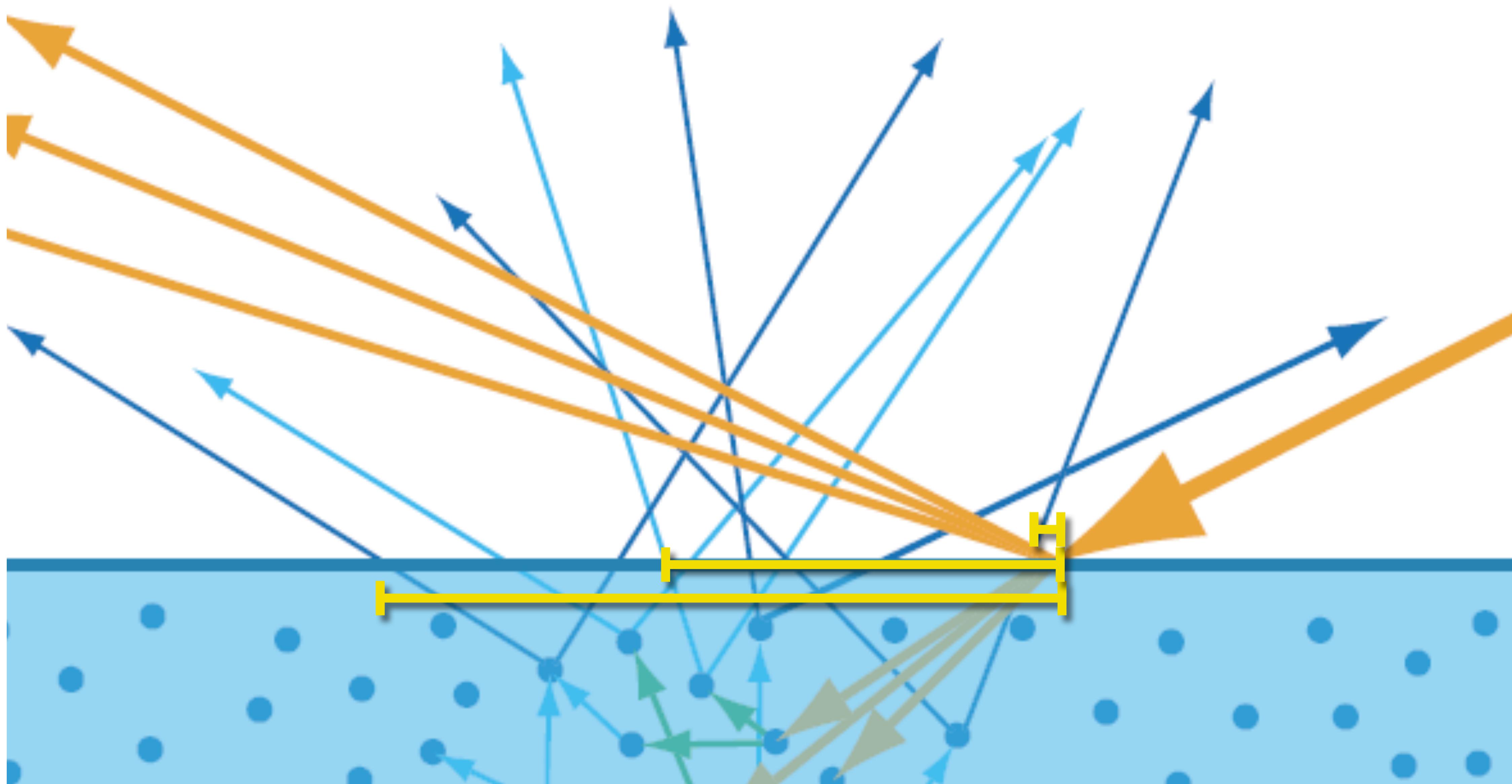
Beyond Lambert: Diffuse-Specular Tradeoff



64

There are a few important physical phenomena that Lambert doesn't account for. Diffuse comes from refracted light - since the specular term comes from surface reflection, in a sense it gets "dibs" on the incoming light and diffuse gets the leftovers. Since surface reflection goes to 100% at glancing angles, it follows that diffuse should go to 0%. The course notes discuss a few ways to model this.

Beyond Lambert: Surface Roughness



Lambert also doesn't account for surface roughness. In most cases, microscopic roughness only affects specular; diffuse reflectance at a point comes from incoming light over an area, which tends to average out any microgeometry variations. But some surfaces have microgeometry larger than the scattering distance, and these do affect diffuse reflectance. That's when you need models like Oren-Nayar.

Math → Rendering



We've talked about how to represent the physics of light / matter interactions in mathematical shading models. But how do we implement these in a renderer?

shading model + illumination model
= rendering implementation



To be implemented in a renderer, a shading model needs to be combined with a specific illumination model.

- General Lighting

- General Lighting
- Image-Based Lighting

- General Lighting
- Image-Based Lighting
- Area Light Sources



It is often advantageous to separate light sources such as the sun and lamps into specialized illumination models which take account of their brightness and area. Area light sources are commonly used in film but are difficult to implement efficiently enough to be used in games - in Brian Karis' talk later in this course we will hear of one way to do so. Instead of area light sources, games typically use...

- General Lighting
- Image-Based Lighting
- Area Light Sources
- Punctual Light Sources

- General Lighting
- Image-Based Lighting
- Area Light Sources
- Punctual Light Sources
- Ambient Light



Ambient light covers various low-frequency lighting representations, ranging from a single constant light color and intensity over all incoming directions to more complex representations such as spherical harmonics . We will now focus on punctual lights and image-based lighting due to their common use in game and film production rendering.

Punctual Light Sources

- Parameterized by light color c_{light} and direction to the light (center) position \mathbf{l}_c
- c_{light} equals radiance from a white Lambertian surface illuminated by the light at 90 degrees

Punctual Light Equation

$$L_o(\mathbf{v}) = \pi f(\mathbf{l}_c, \mathbf{v}) \otimes c_{\text{light}}(\mathbf{n} \cdot \mathbf{l}_c)$$



Implementing a shading model with punctual lights is extremely simple - just evaluate the BRDF in a single light direction and multiply by π (the derivation of this is in the course notes). Games often clamp the dot product between the light and normal vectors as a convenient method to remove the contribution of backfacing lights.

Image Based Lighting

Optically flat (mirror) surface is easy - just multiply reflection by Fresnel function; same c_{spec} , different angle:

- $F(\mathbf{v}, \mathbf{n})$ instead of $F(\mathbf{l}, \mathbf{h})$ or $F(\mathbf{v}, \mathbf{h})$

Image Based Lighting

Non-mirror (glossy/diffuse) surfaces require many samples

- Importance sampling helps
- Prefiltering (alone or with importance sampling)

Acknowledgements

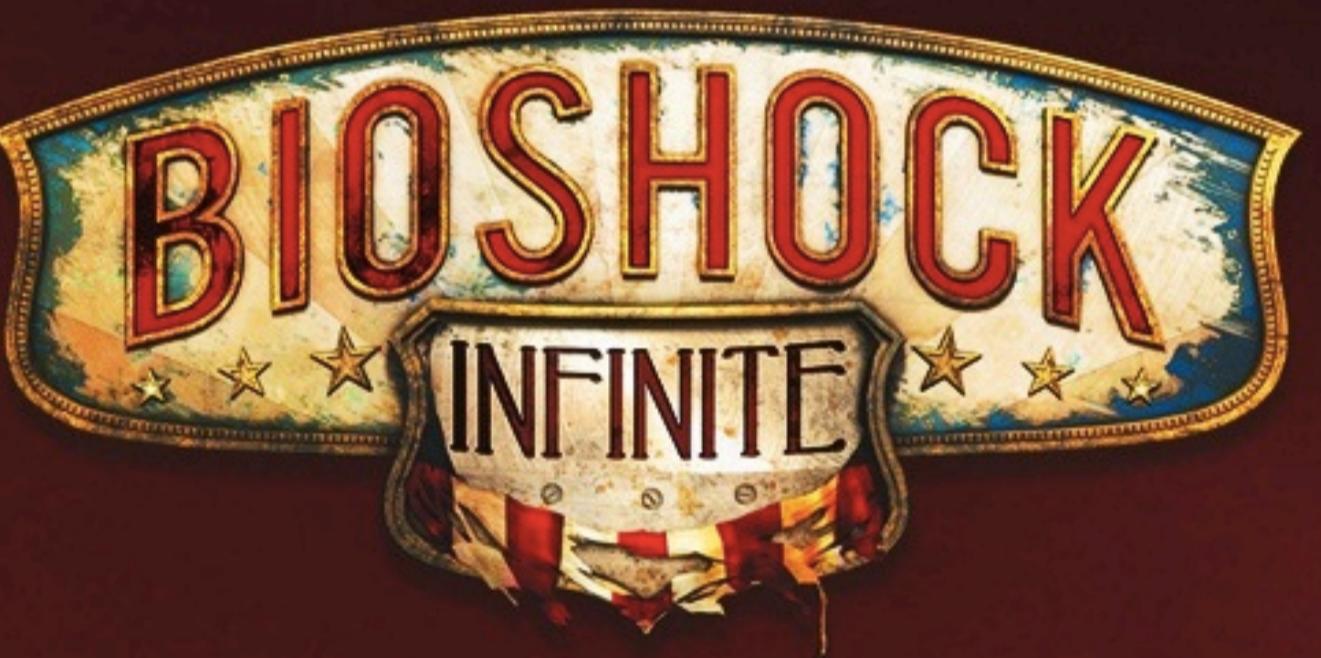
- A K Peters for permission to use RTR3 images
- Brent Burley, Paul Edelstein, Yoshiharu Gotanda, Christophe Hery, Sébastien Lagarde, Dimitar Lazarov, and Brian Smits for thought-provoking discussions
- Steve Hill for helping improve the course notes and slides, and for the WebGL framework used for the Fresnel visualization



I'd like to close by thanking some people who helped me with this talk.



IS HIRING!



And finally, I wanted to note that 2K is hiring - there are positions across many of our studios. In particular, I'm personally looking for a top-notch rendering programmer for the 2K Core Tech group.