

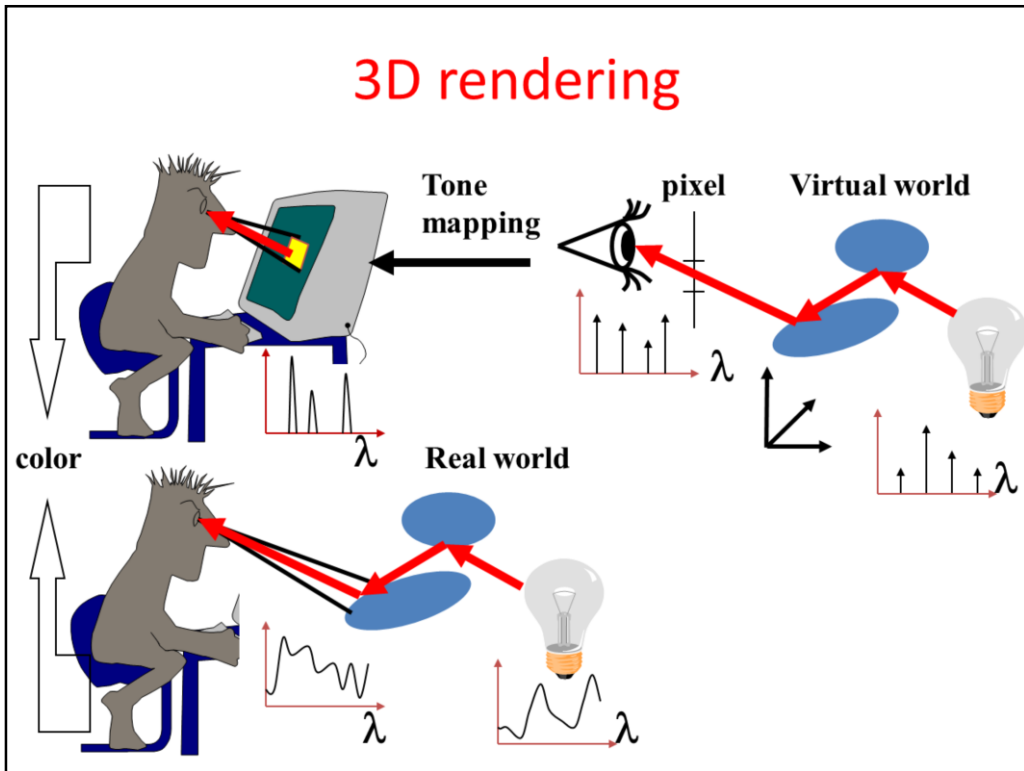
*Science is either physics or  
stamp collecting.*  
*Rutherford*

# Physics fundamentals of 3D rendering

Szirmay-Kalos László



In this lecture we review the optics fundamentals that are essential in simulating the light transfer in virtual worlds.



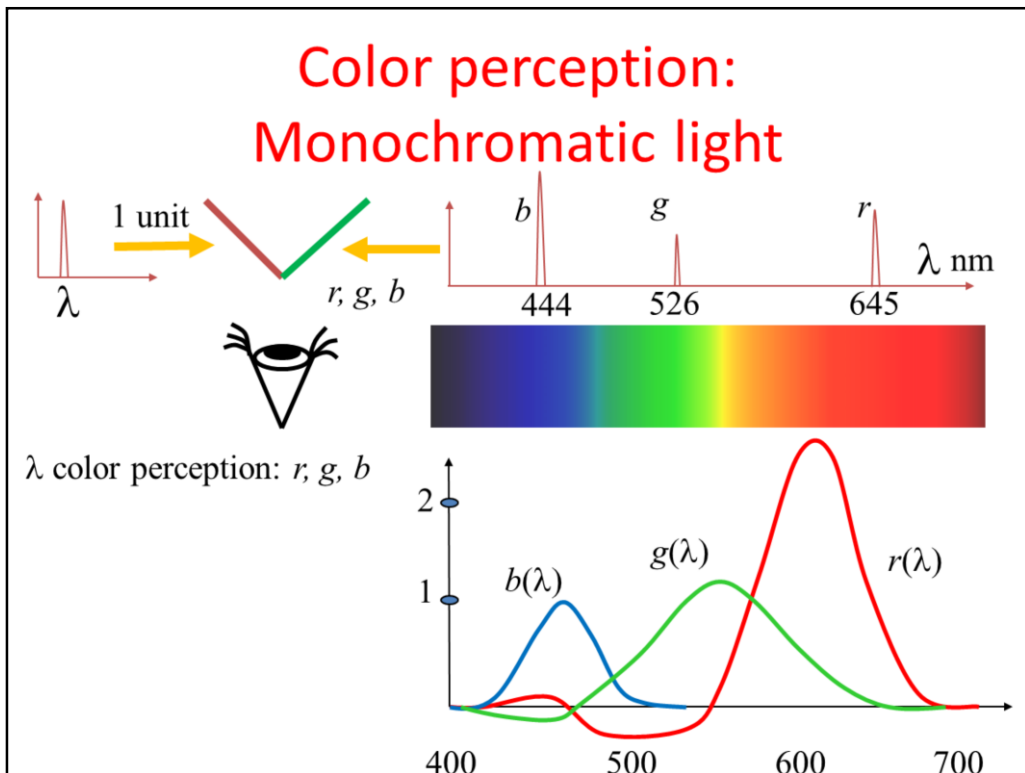
In order to compute the image, the power arriving at the eye from the solid angle of each pixel needs to be determined on different wavelengths.

We establish a virtual world model in the computer memory, where the user is represented by a single eye position and the display by a window rectangle. Then we compute the power going through a pixel toward the eye on different wavelengths, which results in a power spectrum.

If we can get the display to emit the same photons (i.e. the same number and of the same frequency) as that real surface would emit which is visible in the solid angle of the pixel, then the illusion of watching the virtual world can be created. Note that the human eye is sensitive only to the number and frequency of photons, but is not sensitive to the source of the photons, thus, we would not be able to distinguish the real world from a virtual one shown on the computer screen.

As the human eye can be cheated with red, green, and blue colors, it is enough if the display emits light on these wavelengths. The last step of rendering is the conversion of the calculated spectrum to displayable red, green and blue intensities, which is called color matching and tone mapping. If we compute the light transfer only on these wavelengths, then this step can be omitted and the resulting spectrum can be used directly to control the monitor.

One crucial question will be what physical quantity should be computed for the surfaces that describes the strength of the light intensity and when the pixel is controlled accordingly, provides the same color perception as the surface. Note that the pixel is at a different distance than the visible surface. The orientations of the display surface and of the visible surface are also different. The total emitted power would definitely be not good since it would mean less photons for the eye for farther sources.



Light is an electromagnetic wave, color is just an illusion created by the human eye and the brain. As the eye is a poor spectrometer, we can cheat it with a different spectrum, the eye cannot tell the difference. This fact is exploited by displays, which can emit light just around three wavelengths. So the task is to convert the computed spectrum to the intensities of the three lamps associated with a pixel. To solve this, we should understand how the illusion of color is created. As the illusion is deep in our brain, we can use only subjective comparative experiments to find out what color means.

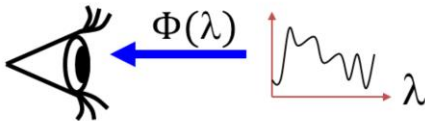
In our experiment, we have two white sheets, the first is illuminated by a unit radiance monochromatic light beam of wavelength  $\lambda$ , the other is by three lamps of controllable intensities and of wavelengths, say, 444, 526, 645 nanometers, which could be seen as red, green and blue (we could choose other reference wavelengths as well, they just have to be far enough; this particular selection is justified by the fact that there exist materials that emit light on these wavelengths). A human observer sits in front of the two white sheets and his task is to control the intensities of the three lamps in order to eliminate any perceived difference between the two sheets. If it happens, the monochromatic light and the controlled three component light provide the same color and are called **metamers**. If the same experiment is repeated in many discrete wavelengths, three **color matching functions** can be obtained, which are going to be our **model of the human color perception**.

Note that the red and the green matching function have negative parts as well, which means, for example, that the 500 nm light can be matched only if some red is added to it. So negative light here means light that is added to the other side of the matching experiment.

In the second experiment we can try to match two, three, etc. component light beams and beams of non-unit intensity. We will come to the conclusion that the corresponding  $r, g, b$  values of polychromatic light are the sums of the  $r, g, b$ , primaries of the monochromatic components, and also that if the intensity of the beam is not unit, then the  $r, g, b$  intensities

should also be multiplied by the same factor. This means that colors are linear objects.

# Color perception: Polychromatic light



$$R = \int \Phi(\lambda)r(\lambda)d\lambda$$

$$G = \int \Phi(\lambda)g(\lambda)d\lambda$$

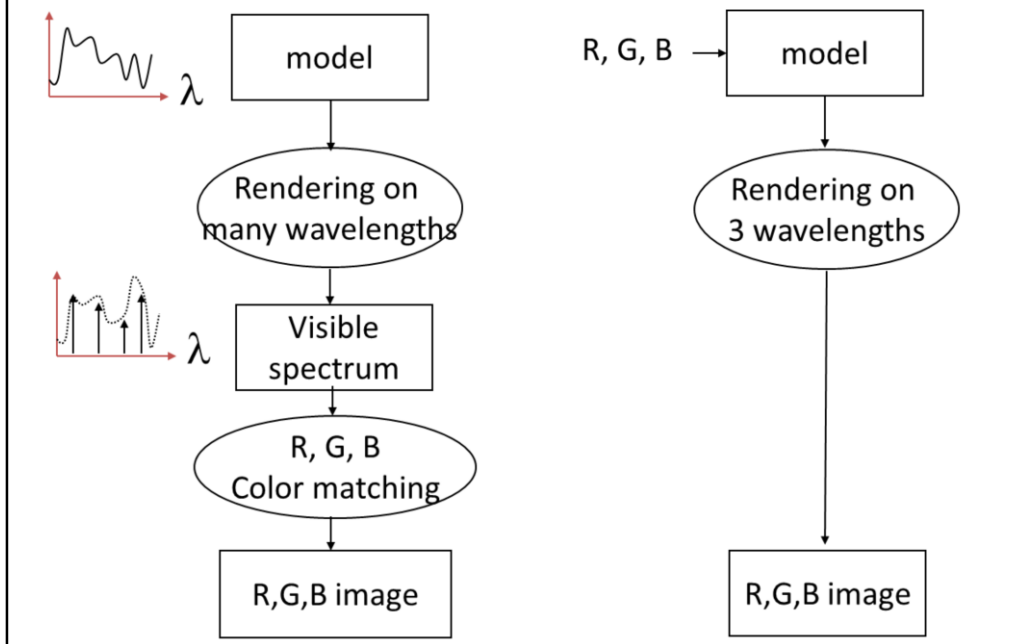
$$B = \int \Phi(\lambda)b(\lambda)d\lambda$$

Color perception is linear (Grassmann's laws)

- Two times the spectrum = Two times  $r,g,b$
- Sum of two spectra = Sum of corresponding  $r,g,b$  values

Based on these experiments, we can establish the Grassmann laws of color science. Any spectrum can be matched with three primaries by weighting the monochromatic components by the color matching functions and adding (integrating) different monochromatic components.

## Spectral versus RGB rendering



A physically plausible simulation would be executed on many wavelengths (note that wavelengths can be handled independently) resulting in a visible spectrum. The final step of rendering is the conversion of this spectrum to red, green, blue intensities, which can be set in the frame buffer, and ultimately on the display.

However, in many cases, we use an approximation. We assume that light sources emit light directly on the wavelengths of the red, green, blue. Thus, we can immediately get the  $r, g, b$  values without any integration. Note, however, that the rendering process is not linear since the products of light source intensities and material reflection parameters are computed, so this simpler option is just an approximation.

**Radiance:  $L(\mathbf{r}, \mathbf{V})$**

Power emitted by a unit visible area surface into a unit solid angle [Watt/sr/m<sup>2</sup>]

$$L(\mathbf{r}, \mathbf{V}) = \frac{\Delta\Phi}{\Delta A \cos \theta \Delta\omega}$$

$$\Delta\Phi = L(\mathbf{r}, \mathbf{V}) \Delta A \cos \theta \Delta\omega$$

If we want some efficient computation scheme, we should work with power density instead of the power, that is computed with respect to the solid angle in which the light is emitted and with respect to the size of the projected surface. The density computed as the power divided by the projected surface and the solid angle of emission is called the **radiance**.

Experience and an important theorem state that if two surfaces have the same radiance, then they look identical no matter whether they are at a different distance or have different orientation. The proof is based on the fact that if in a solid angle the eye gathers the same number and frequency of photons, then the eye would not be able to distinguish the source of the surface. Let us compute this power for two surfaces that are seen in the same solid angle and have the same radiance. The formula is obtained from the definition of the radiance, multiplying both sides by the denominator.

If the surface is closer, then its real area is smaller, but the solid angle in which the pupil of the eye can be reached from this surface is larger. Both the solid angle and the surface changes with the square of the distance and the two factors compensate each other. If the surface is not perpendicular to the viewing direction, then the surface seen in a given solid angle is larger, but the cosine factor will be proportionally smaller, so again we see no difference.

So, the conclusion is that we should compute the radiance of a surface and set the pixel of the display to have the same radiance. Then the two surfaces will be identical for the eye.

The fact that surfaces having the same radiance but at different distances look similar can also be interpreted as that the radiance does not change along a ray.

An important remark is that this independence of the distance holds only if the surface is large, or in other words when we get farther, we can see a larger part of the same surface. Point like phenomena, like small light bulbs or very distant stars, do not fall into this category. For point like objects, the radiance is inversely proportional to the square of the distance as expected.

## Wavelengths are independent



- Relativistic mass is small:  $m = E/c^2 = hf/c^2$
- Photon energy (frequency) does not change in elastic scattering

If we have, for example, red monochromatic light source in our room, then only this red color show up in the room with different intensities (with zero intensity, the red becomes black). We would never experience, say, green or yellow colors in this situation. Monochromatic light intensity may be modified but its frequency remains the same.

During the simulation of light transfer, different wavelength can be handled independently, which simplifies the computation considerably. But, what is the physical explanation of this phenomenon?

In computer graphics we consider photons in the visible wavelength range, roughly from 300 to 700 nanometer wavelengths. A photon has zero rest mass, otherwise it would not be able to fly with the speed of light. However, it has non-zero energy and impulse. The energy is proportional to frequency  $f$  of the light as stated by Einstein who invented this law when examining the photoelectric effect. He got his Nobel prize for this and not for the theory of relativity. Using the equivalence of the energy and mass ( $E = mc^2$ ), which was also published by Einstein as a short paper in 1905, we can assign a relativistic weight to the photon as the Planck constant  $h$  multiplied by the frequency and divided by the square of the speed of light.

If  $f$  is small, then this relativistic mass is small. When photons meet a material, photons collide or scatter by the electrons or less probably with the core of atoms. For photons belonging to the visible spectrum, the relativistic mass of the photon is much smaller than the mass of the electron, thus a photon bounces off the electron like a ball bounces off from a rigid wall or a billiard ball bounces off from the edge of the table. If the collision is elastic, then the photon energy is preserved and the electron does not change its energy level.

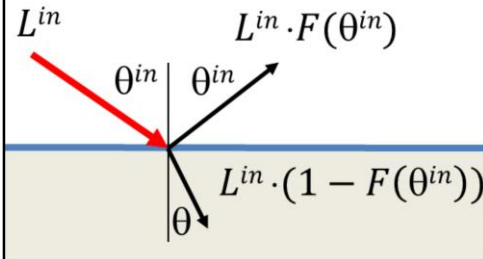
If the collision is inelastic, then the energy of the photons is absorbed by the electron, this is the photoelectric effect, and the number of photons gets smaller. The probability of inelastic scattering, i.e. the albedo associated with a collision is energy dependent.



Summarizing when photons meet electrons, their number may get smaller but their energy level and consequently their frequency remain the same. This is the reason that in computer graphics wavelengths or frequencies are handled independently.



## Smooth surfaces: Fresnel equation



„Smooth” = surface visible in a single pixel is a plane

Snellius –  
Descartes:

$$n = \frac{\sin \theta^{in}}{\sin \theta}$$

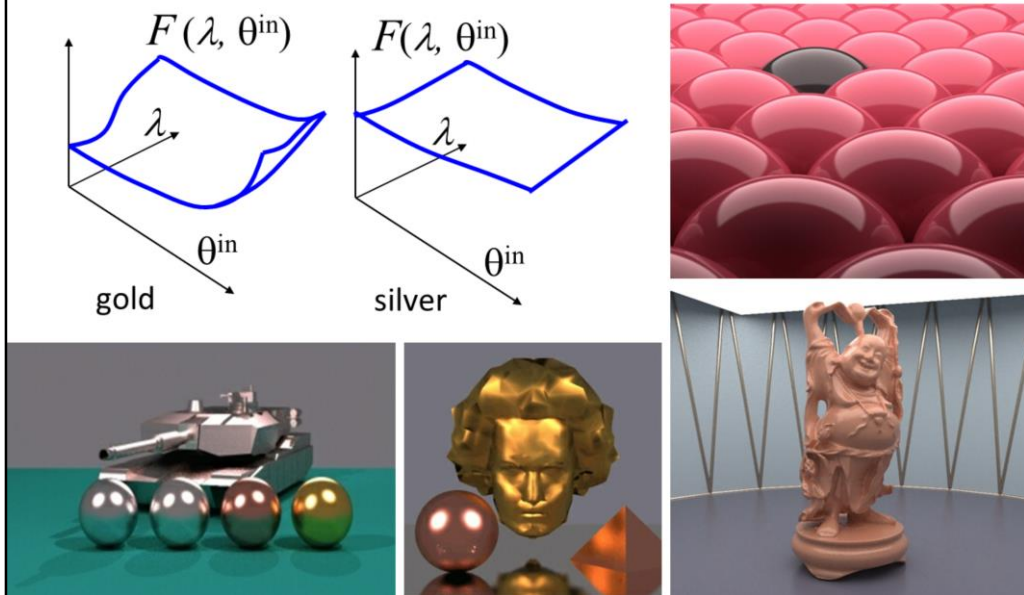
$n$ : index of refraction  
 $\kappa$ : extinction

$$F(\theta^{in}) \approx F_0 + (1 - F_0) \cdot (1 - \cos \theta^{in})^5 \quad F_0 = \frac{(n - 1)^2 + \kappa^2}{(n + 1)^2 + \kappa^2}$$

The simplest arrangement for the light object interaction is a single plane that separates the space into two half spaces of air and a given material. Suppose that a light ray arrives at the surface from the air. According to the laws of geometric optics, the illumination ray is broken to a reflection ray meeting the reflection law and a refraction ray obeying the **Snell's law of refraction**. Here  $n$  is the **index of refraction**, which expresses the ratio of speeds of light outside and inside the material. The Fresnel equations define the amount of reflected energy (i.e. the probability that a photon is reflected). The **Fresnel function** can be calculated from **index of refraction**  $n$ , **extinction**  $k$ , which describes how quickly the electromagnetic wave diminishes inside the material, incident angle  $\theta^{in}$  and refraction angle  $\theta$ . The extinction is negligible for non-metals. Here we show a simplified Fresnel term, which is accurate enough for our purposes.

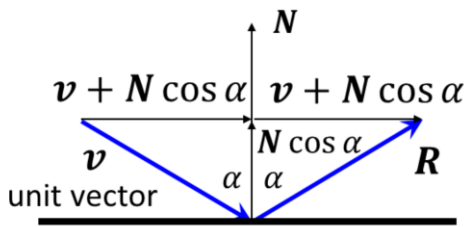
In this formula,  $F_0$  expresses the reflection probability of the photon when it arrives from perpendicular direction.  $F_0$  is a function of the index of refraction and the extinction. Let us take glass as an example. The extinction coefficient of glass is negligible since glass is non metal. The index of refraction is 1.5. Thus  $F_0$  is 4%. Indeed, when we look at a glass window perpendicularly, the image is determined by our own reflection with only 4%, while the outside view determines its 96%. However, when the direction is not perpendicular,  $F$  gets larger and reaches 100% at grazing angles. Glass becomes mirrors at grazing angles.

# Fresnel function



The Fresnel function depends on the wavelength and on the incident angle. When we see an object, we can observe surfaces of many different orientations, so we perceive the Fresnel function as a whole. This is why we can distinguish gold from copper although both of them are yellow.

## Reflection direction



$$\cos \alpha = -\mathbf{v} \cdot \mathbf{N}$$

$$\mathbf{R} = \mathbf{v} + 2\mathbf{N} \cos \alpha$$

```
vec3 reflect(vec3 inDir, vec3 normal) {
    return inDir - normal * dot(normal, inDir) * 2.0f;
};
```

```
vec3 Fresnel(vec3 inDir, vec3 normal) {
    float cosa = -dot(inDir, normal);
    vec3 one(1, 1, 1);
    vec3 F0 = ((n-one)*(n-one) + kappa*kappa) /
              ((n+one)*(n+one) + kappa*kappa);
    return F0 + (one - F0) * pow(1-cosa, 5);
}
```

$$F_0 = \frac{(n-1)^2 + \kappa^2}{(n+1)^2 + \kappa^2}$$

To render smooth surfaces, we should compute the ideal reflection direction. Assume that incident direction  $\mathbf{v}$  and surface normal  $\mathbf{N}$  are unit length vectors.

Incident direction  $\mathbf{v}$  is decomposed to a component parallel to the normal and a component that is perpendicular to it. The length of the parallel component is the projection of  $\mathbf{v}$  onto  $\mathbf{N}$ , which can be obtained with the dot product. The perpendicular component is obtained by requiring the sum of parallel and perpendicular components be equal to  $\mathbf{v}$ .

Then, the reflection direction is built up from these two components.

For the amount of reflected light, we also need the Fresnel function for this direction, which is the direct implementation of our approximating formula.

For the computer simulation, we need two functions. Function `reflect` calculates the ideal reflection direction. Function `Fresnel` determines the portion of light being reflected. As the index of refraction and the extinction are wavelength dependent, the Fresnel function is of type `vec3` since it can have a different value on the wavelengths of red, green, and blue.

## Refraction direction

Snellius-  
Descartes

$$n = \frac{\sin \alpha}{\sin \beta}$$

$$N_{\perp} = \frac{v + N \cos \alpha}{\sin \alpha}$$

$$T = N_{\perp} \sin \beta - N \cos \beta$$


---


$$T = v/n + N(\cos \alpha/n - \cos \beta)$$

$$\cos \beta = \sqrt{1 - \sin^2 \beta} = \sqrt{1 - \sin^2 \alpha/n^2}$$

$$T = v/n + N(\cos \alpha/n - \sqrt{1 - (1 - \cos^2 \alpha)/n^2})$$

```

vec3 refract(vec3 inDir, vec3 normal, float ns) {
    float cosa = -dot(inDir, normal);
    float disc = 1 - (1 - cosa * cosa) / ns / ns;
    if (disc < 0) return vec3(0, 0, 0);
    return inDir/ns + normal * (cosa/ns - sqrt(disc));
}

```

The refraction direction calculation is also similar. The refraction direction  $v_t$  is expressed as a combination of the normal vector and a vector that is perpendicular to the normal,  $N_{\perp}$ . These vectors should be combined with weights  $\cos(\beta)$  and  $\sin(\beta)$  where  $\beta$  is the refraction angle.

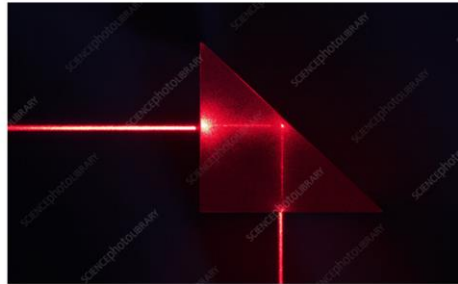
$N_{\perp}$  is expressed from  $v + N \cos(\alpha)$  by dividing it with its length  $\sin(\alpha)$ .

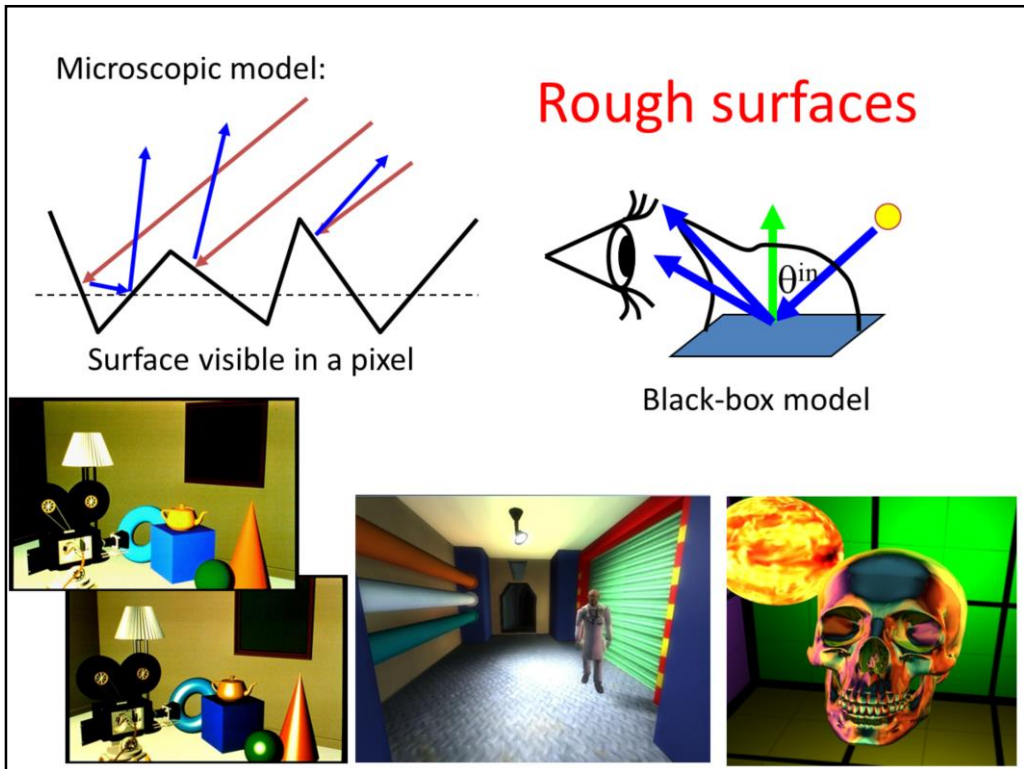
Then  $\sin(\beta)/\sin(\alpha)$  is replaced by the reciprocal of the index of refraction. Function  $\cos(\beta)$  is expressed with  $\sin(\beta)$ , for which we apply the same trick, and replaced with  $\sin(\alpha)$ .

Finally,  $\sin(\alpha)$  is expressed from  $\cos(\alpha)$ , which is available as the dot product of the surface normal and the incident direction.

The final formula for refraction direction  $T$  involves a square root, where the discriminator in the square root can be negative if relative refraction index  $n$  is less than 1. This relative refraction index is less than 1 if the ray arrives at the boundary from the optically denser medium, for example, when the ray is originally in water and meets the surface separating water from air. Negative discriminator indicates that there is no refraction direction because the light gets fully reflected. This situation is called the total internal reflection and the threshold angle when it first occurs is the Brewster angle. The picture in the left upper corner shows a swimmer (László Cseh, olympic silver medalist) who is below the water while the camera is also at the bottom of the pool. The water surface acts as a perfect mirror reflecting the back of the swimmer and also the bottom of the pool.

# Total internal reflection





Surfaces are usually not perfectly smooth, so they reflect light not just in the ideal reflection direction but practically in all possible directions. On microscopic scale, we can imagine these rough surfaces as a random collection of **ideal mirror microfacets** that reflect light according to their random orientation.

As we see not a single microfacet in a pixel, but a large collection of them, we perceive the average radiance reflected by this collection.

Photons may have a single scattering on these microfacets when the average is maximum around the ideal reflection direction of the mean surface. On the other hand, photons may get scattered multiple times, when they “forget” their original direction, so the reflection lobe will be roughly uniform.

Instead of following a probabilistic reasoning, for the sake of simplicity, we handle these rough surfaces as a black-box, i.e. empirical model. That is, we describe the behavior of the surface based on everyday experience without any structural analysis. By experience, we say that a rough surface reflects light into all directions, but more light is reflected into the neighborhood of the ideal reflection direction.

# Light-matter interaction

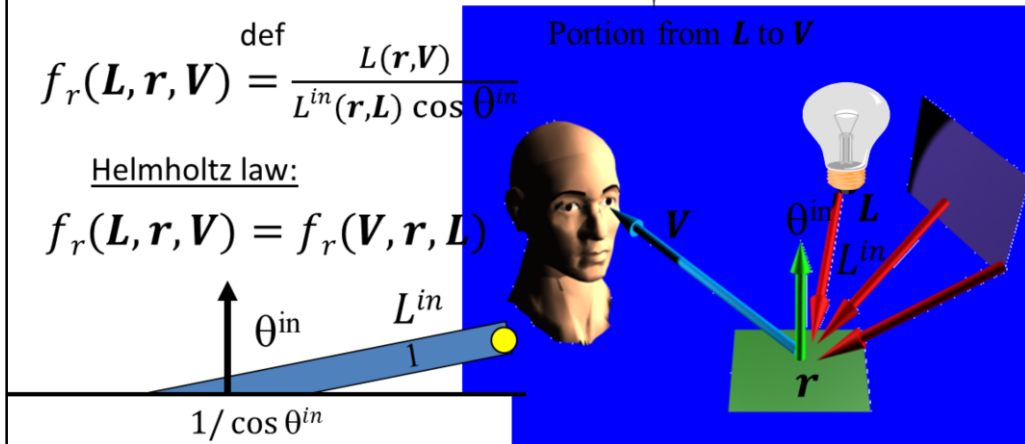
Radiance = Irradiance · BRDF

$$L(\mathbf{r}, \mathbf{V}) = L^{in}(\mathbf{r}, \mathbf{L}) \cdot \cos \theta^{in} \cdot f_r(\mathbf{L}, \mathbf{r}, \mathbf{V})$$

$$f_r(\mathbf{L}, \mathbf{r}, \mathbf{V}) \stackrel{\text{def}}{=} \frac{L(\mathbf{r}, \mathbf{V})}{L^{in}(\mathbf{r}, \mathbf{L}) \cos \theta^{in}}$$

Helmholtz law:

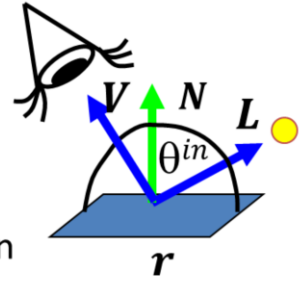
$$f_r(\mathbf{L}, \mathbf{r}, \mathbf{V}) = f_r(\mathbf{V}, \mathbf{r}, \mathbf{L})$$



The reflected radiance of a surface depends on the irradiance and the likelihood of the reflection. The irradiance is the incident radiance and a **geometric factor** that expresses that the illumination is weaker if the light arrives from a non-perpendicular direction since a unit cross section light beam illuminates a larger surface on which the photons are distributed. The likelihood of reflection, or the ration of the reflected radiance and the incident irradiance is expressed by the **Bi-directional Reflectance Distribution Function (BRDF)**  $f_r$ . In real life, BRDFs are symmetric, which is referred to as the **Helmholtz's reciprocity principle**. However, in computer graphics, the BRDF is just a formula, so we can work with non-symmetric BRDFs as well, but these cannot exist in real life.



## Diffuse reflection



- Radiance = Irradiance · BRDF  
is independent of the viewing direction
- BRDF is independent of the viewing direction
- Helmholtz: BRDF is independent of the illumination direction
- BRDF is constant:  $f_r(\mathbf{L}, \mathbf{r}, \mathbf{V}) = k_d(\mathbf{r}, \lambda)$
- Diffuse reflection = very rough surfaces
  - Multiple light-matter interaction
  - colorful!

Our first model is for **very rough surfaces** where all photons get reflected multiple times. Such materials (snow, sand, wall, chalk, cloth etc.) have a matte look, they look the same from all viewing directions. Thus, the radiance, which equals to the incident radiance times the BRDF times the geometry factor, is independent of the viewing direction. Incident radiance and the geometry term are already independent of the viewing direction, thus the BRDF must also be independent of the viewing direction. According to Helmholtz reciprocity, if the BRDF is independent of the viewing direction, it must be independent of the illumination direction as well, so the BRDF is direction independent.

Diffuse surfaces correspond to very rough surfaces where a photon collides many times. The Fresnel depends on the wavelength, which is strong for metals and weak for non-metals. Even if a single reflection changes the spectrum just a little, multiple reflections amplify this effect, so the final reflected light will have a modified spectrum. Diffuse reflection is primarily responsible for the “**own color**” of the surface.

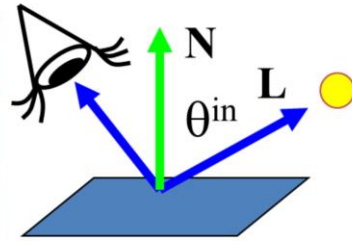
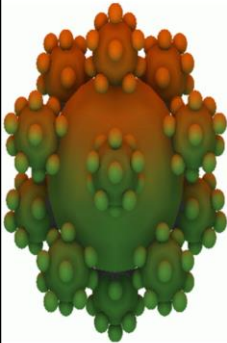


## Lambert's law

BRDF is constant, BUT the reflected radiance does depend on the illumination direction

$$L^{ref} = L^{in} k_d \cos^+ \theta^{in}$$

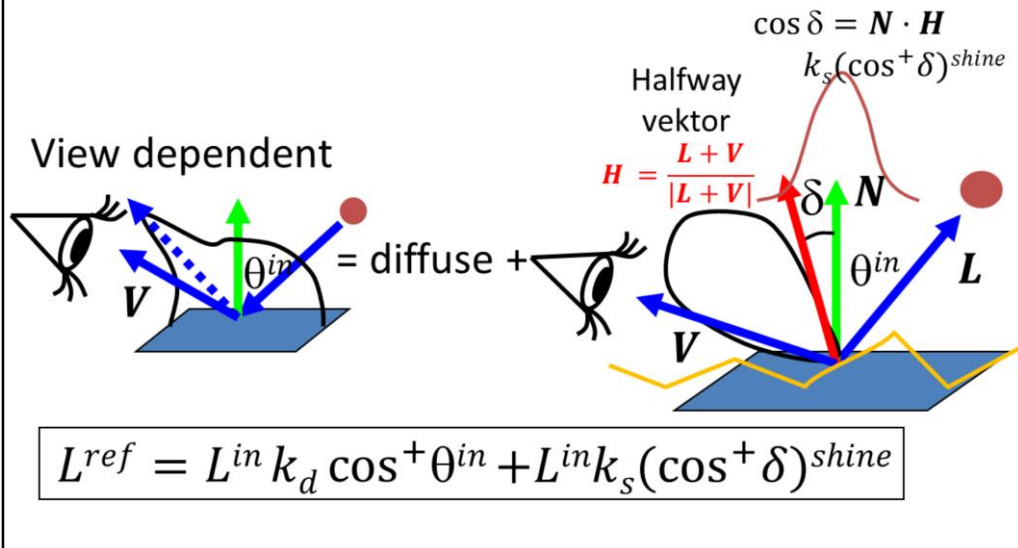
$$\cos \theta^{in} = \mathbf{N} \cdot \mathbf{L}$$



The reflected radiance is the incident radiance times the BRDF, which is constant now, and the geometry factor. So for diffuse surfaces, the reflected radiance is proportional to the cosine of the orientation angle. This cosine can be computed as a dot product of the unit surface normal and the unit illumination direction.

If the cosine is negative, i.e. the angle between the surface normal and the illumination direction is greater than 90 degrees, then the illumination is blocked by the object whose surface is considered. In such cases, the negative value is replaced by zero.

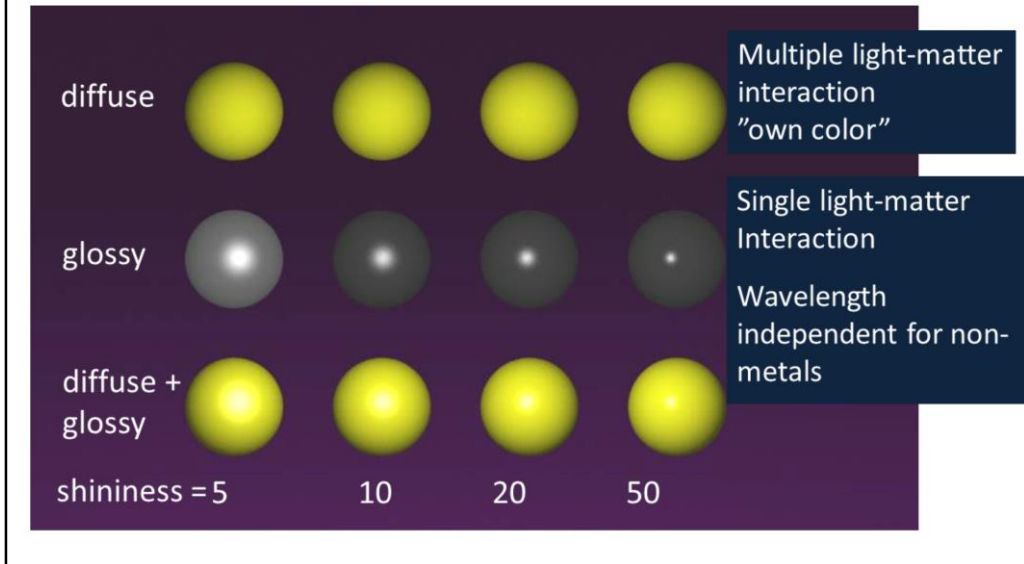
# Specular (glossy) reflection (Phong-Blinn model)



Shiny, glossy or specular surfaces also reflect the light in all possible directions, but they look differently from different viewing directions. We can observe the blurred reflection of the light sources, thus they reflect more light close to the ideal reflection direction.

We model such surfaces as a combination of diffuse reflection where the radiance is constant and a specular reflection where the radiance is great around the ideal reflection direction. According to the microfacet model, diffuse reflection is caused by multiple light microfacet interaction while specular reflection is the result of a single light microfacet interaction. In order to model the specular reflection lobe, we need a function that is maximum at the reflection direction and decreases in a controllable way if the viewing direction gets farther from the reflection direction. **Phong and Blinn proposed the  $\cos^{\text{shininess}}(\delta)$  function where  $\delta$  is the angle between the macroscopic surface normal and the microfacet normal.** The shininess exponent defines how shiny the surface is. Considering the microfacet model, this factor is proportional to the probability density that a random microfacet reflects the given light direction  $L$  exactly to the given view direction  $V$ , that is, the normal vector of the microfacet is equal to halfway vector  $H$  in between light direction  $L$  and view direction  $V$ .

## Diffuse+glossy materials



Diffuse reflection simulates multiple light-surface interaction and is colored. Specular reflection is the model of the single light-surface interaction and it is proportional to the Fresnel function. For non metals, the wavelength dependence of the Fresnel is moderate, so for non metals the specular reflection is said to be "white".

# RoughMaterial

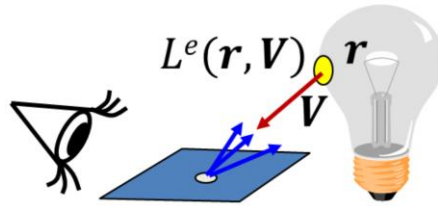


```
//          unit N          unit V          unit L
vec3 shade( vec3 normal, vec3 viewDir, vec3 lightDir,
            vec3 inRad) {
    float cosTheta = dot(normal, lightDir); // unit vecs
    if(cosTheta < 0) return vec3(0,0,0); // self shadow
    vec3 outRad = inRad * kd * cosTheta; // diffuse
    vec3 halfway = normalize(viewDir + lightDir);
    float cosDelta = dot(normal, halfway);
    if(cosDelta < 0) return outRad;
    return outRad + inRad * ks * pow(cosDelta, shine);
}
```

To simulate a rough material, we need a **shade** function that computes the reflected radiance for incident radiance **inRad**, incident direction **lightDir**, surface normal vector **normal**, and view direction **viewDir**. Since the reflected and incident radiances are spectra, we use **vec3** type for them that can be different on the wavelength of red, green and blue. The function first computes the geometry factor **cosTheta**. If this geometry factor is negative, then the back of the surface is illuminated, so the object itself occludes the light source, so the reflected radiance is zero. Then, the diffuse radiance is computed, which is increased by the glossy term if **cosDelta** is not negative.

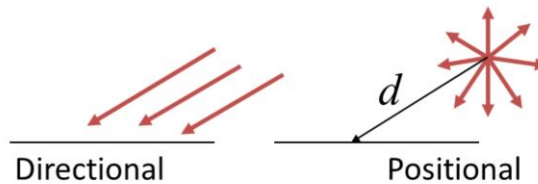
# Light sources

- Geometry + emitter radiance:



- Abstract light sources:

- Directional: rays are parallel, intensity is constant
- Positional: rays connect the light source to the shaded point, intensity decreases with the square of the distance

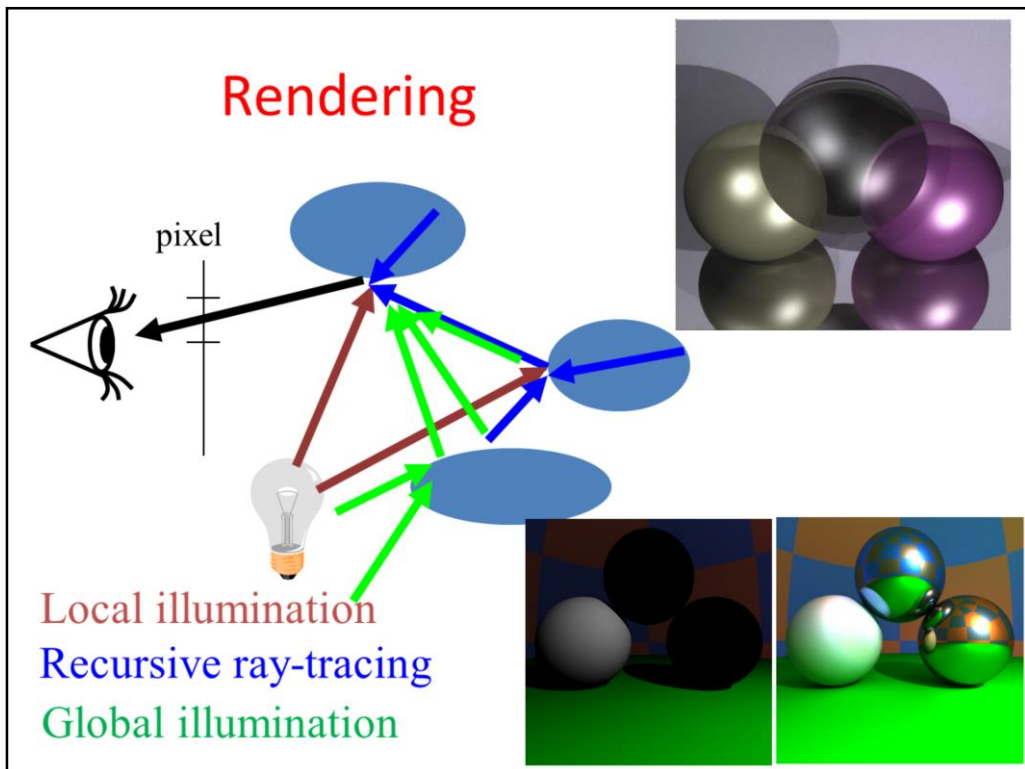


Real light sources are defined by their emission radiance,  $L^e$ . When the reflected radiance of a point is considered, the contribution of all those light source points should be added which are visible from the point of interest. This means integration. Thus, we often prefer abstract light source models, that can illuminate a surface just from a single direction, which saves integration.

In case of directional light sources, the radiance is constant everywhere, so is the illumination direction. In other words, the illumination rays are parallel. The Sun is an example for directional light source if we are on the Earth.

For point light sources, the illumination direction points from the location of the source to the illuminated point. The radiance decreases with the square of the distance.

If we ignore the dependence of the radiance on the distance, directional light sources can be considered as point sources being at infinity.



Rendering requires the determination of the surface that is visible through a pixel, then the computation of the radiance of this surface in the direction of the eye. There are different tradeoffs between accuracy of the light transport computation and the speed of the computation.

In the local illumination setting, when the radiance of a surface is calculated, we consider only the direct contribution of the light sources and ignore all indirect illumination.

In recursive ray tracing, indirect illumination is computed only for smooth surfaces, in the ideal reflection and refraction directions.

In the global illumination model, indirect illumination is taken into account for rough surfaces as well. In engineering applications we need global illumination solutions since only these provide predictable results. However, in games and real time systems, local illumination or recursive ray tracing will also be acceptable.