Scuola di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea in Informatica

Tesi di Laurea

BOX OF SUNLIGHT
Riflessione Realistica della Luce nel Rendering di Animazioni 3D

BOX OF SUNLIGHT
Realistic Reflection of Light in 3D Animation Rendering

LEONARDO ZETTI

Relatore: *Stefano Berretti*

Correlatore: *Michele Ginolfi*

Anno Accademico 2023-2024

CONTENTS

LIST OF FIGURES

"Insert citation"
— Insert citation's author

1

INTRODUCTION

The aim of this thesis is to explore a set of concepts that are fundamental for the generation of photo-realistic images in computer graphics, with a focus on the context of 3D animation.

We'll shine some light on what it means for digital, 3D art to be *physically based*. There's a lot that goes into the creation of physically based images, and our analysis will only scratch the surface of the problem. More specifically, we'll only be concerned with capturing realistic *reflection of light* on virtual materials, as there's a lot to be said on this topic alone.

During our discussion, we'll move freely between the fields of physics, mathematics, and computer science, to finally present a small computer program for physically based reflections, called `BoxOfSunlight`.

`BoxOfSunlight` was written in C++, using the OpenGL API. Among its features, it contains an implementation of the *Disney Principled BRDF*, a reflection model presented by Brent Burley in his 2012 SIGGRAPH talk *Physically Based Shading at Disney*[1] (Burley 2012).

1.1 COMPUTER GRAPHICS

Computers have the potential to be powerful tools for artistic expression. One way to create art with a computer is through *computer graphics*. *Computer graphics* can be defined as

> "The science and art of communicating visually via a computer's display and its interaction devices."
> (Hughes et al. 2014)

Advancements in this field have made computer-generated imagery a pervasive component of our day-to-day lives, ranging from special effects in movies to the *Graphical User Interfaces* (GUIs) on our phones.

1 The talk was part of the 2012 SIGGRAPH course *Practical Physically Based Shading in Film and Game Production*.

Thus, the world of computer graphics is a vast one. We should keep in mind that the problems and solutions presented here only refer to the context of "geometry-based 3D graphics", as we'll call them. Our graphics will be "geometry-based" in the sense that we'll first describe the objects we want to draw on the screen with *geometric models*[2] (lines, polygons, polygonal meshes, ...), to then *sample* them for visualization. We can imagine our process to be as follows.

1. Create a *geometric model* of the object,
2. Describe the *material* it is made of,
3. Place the object in a *virtual scene* with *light sources*,
4. Use a *virtual camera* to "take a picture" of the object (this is the so-called *sampling* phase).

Our graphics will be 3D, meaning that the geometrical models, other than width and height, will also have *depth*. Our virtual camera will then communicate that depth to the viewer through the use of *perspective*.

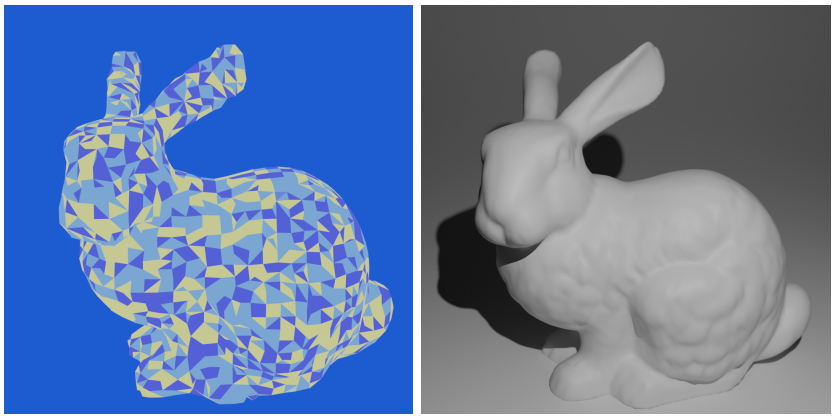

Fig. 1: The *Stanford Bunny Model*, drawn on the left as a *triangle mesh* (random color for each triangle). Model courtesy of the Stanford Computer Graphics Laboratory, rendered with Blender.

2 In the field of computer graphics, the word "model" is used to refer both to *geometric models* and *mathematical models*. A *geometric model* describes an object we want to "take a picture" of (for example, the *Stanford Bunny*). A *mathematical model* describes some physical or computational process that we use to take that picture (for example, the equations we use to tell how much light is reflected by the bunny's surface).(Hughes et al. 2014)

To restrict ourselves even further, we'll keep our focus mainly on steps 2. and 4. of the just mentioned process. Our aim will be to derive a single *mathematical model* than can describe as many materials as feasible, in relation to how they *reflect light*. We'll then use this model to record reflection effects with our virtual camera.

The final images will be generated so:

1. Light gets emitted from a light source,
2. Light hits a surface and is reflected in various amounts and directions, depending on the *material* of the surface,
3. Light that's reflected towards the virtual camera gets recorded in the final image.

We'll now give a more rigorous definition of the virtual "picture-taking" step that we're interested in. We'll refer to it as *rendering*.

1.2 RENDERING AND PBR

In the book *Physically Based Rendering, From Theory to Implementation* - an important resource for this thesis - *rendering* (or, as it has been historically referred to, *image synthesis*) is defined as

> "The process of converting a description of a three-dimensional scene into an image."
> (Pharr, Jakob, and Humphreys 2023)

Note that a *3D scene* is comprised not only of 3D objects, but also light sources and a camera. Generally speaking, the final image of this process can be generated using any rules we choose. This flexibility enables a wide range of different rendering techniques, each of which will communicate a different *message* to the viewer.

This is our final goal in graphics, to *communicate* with viewers through *meaningful images*. As Andrew Glassner put it:

> "The field of *image synthesis*, also called *rendering*, is a field of transformation: it turns the rules of geometry and physics into pictures that mean something to people."
> (Glassner 1995)

If our aim is to create art - as in the case of *animated movies*[3] - rendering can be an important tool for creative expression. The end result of a rendering process will influence the emotional responses elicited in spectators. This interplay of different fields of study - math, physics, art, computer science, psychology, to name a few - is what makes rendering, and computer graphics in general, fascinating disciplines that are worth exploring.

Fig. 2: Stylized render from Blender Studio's *Project Gold*. ©Blender Foundation

So, in theory, there are no rules on how to *render*[4] a picture. However, what we are often interested in is making our images *physically plausible*, that is, *realistic*. This is why various rendering applications - such as *RenderMan*, used to make Pixar movies - are, we say, written to be *physically-based*.

Physically Based Rendering (or *PBR* for short) is a collection of rendering techniques based on physical models of real-world light and materials (Pharr, Jakob, and Humphreys 2023)).

Our objective in this thesis will be physical plausibility, so we'll stick with PBR. As an alternative approach one could, for example, render *stylized* images to favour the expressiveness of their art over its realism.

3 In 3D animated films, each of the frames shown on screen was generated by some advanced rendering software. Achieving high levels of realism demands intense computation: applications used in movie production spend *hours* computing a *single frame*.

4 The word "render" can be used to express two different concepts. When used as a *verb*, it is the process followed by the computer to produce our image. When used as a *noun*, it refers to the final image produced.

1.3 HISTORICAL NOTES

In this paragraph, we'll go through a short summary of some of the key steps that have been taken in the research progress of physically based rendering. We will also see how PBR has been gradually adopted in film production.

Physically based rendering research started to catch on only after the 1980s. An important seminal work was Whitted's 1980 paper, *An Improved Illumination Model for Shaded Display* (Whitted 1980), which has since served as a foundation for modern *ray-traced* computer graphics which feature *global illumination*.

PBR generally makes substantial use of *ray tracing*, and for this reason, in Chapter 3, we'll give a brief description of the main ideas behind this technique. For now, let's just say that ray tracing involves - very surprisingly - tracing rays of light in a 3D scene, to determine the colors of objects. As for *global illumination*, we'll elaborate on it in Chapter 3 as well.

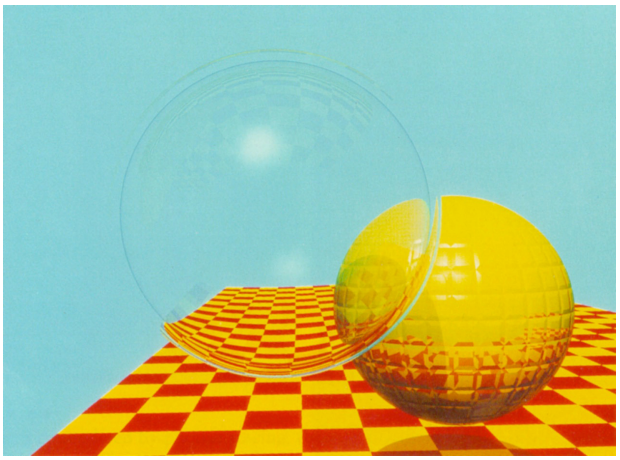

Fig. 3: One of the renders from Whitted's 1980 paper. ©ACM

We should mention that an important early contribution to PBR techniques was also given by Cook and Torrance, when they proposed a new reflection model based on *microfacet theory* (Cook and Torrance 1982). The techniques for realistic reflections described in Chapter 3 will be based on said theory.

Many other historic steps were necessary in order to make today's PBR techniques a reality. One of these was the work of Kajiya, who in 1986

introduced the *rendering equation* (Kajiya 1986). This equation describes in a concise and elegant way the problem solved by any realistic rendering application. More on the rendering equation will be said in Chapter 3.

Kajiya also introduced *path tracing*, which we can consider as an advanced form of *ray tracing* that properly takes into account *global illumination* effects, through the use of *Monte Carlo integration*.

Again, we'll cover this topic better in Chapter 3. However, to understand what we are talking about, let's put it this way. In order to do proper realistic rendering, we need to approximate the values of many (nested) definite integrals. *Monte Carlo integration* is a technique, based on random numbers, for doing just that.

Monte Carlo integration really transformed the field, allowing the creation of images unlike any before. Many important contributions were made to Monte Carlo-based efforts during the years, one of the most important ones being Veach's 1997 PhD thesis. Veach advanced key theoretical foundations of Monte Carlo rendering, and also developed new algorithms that improved its efficiency, like *bidirectional path tracing* (Pharr, Jakob, and Humphreys 2023).

It took some time to incorporate physically based rendering techniques in film production, due to its computational costs. An early example of a movie made with Monte Carlo global illumination was the short film *Bunny* (1998), by Blue Sky Studios. Its visual look was substantially different from other films and shorts of the past. Before then, renderers were mostly based on *rasterization*, a rendering technique that's faster than ray tracing, but, in general, less realistic. *Reyes* is an important example of a *rasterization-based* architecture that had been used to generate photo-realistic images (Pharr, Jakob, and Humphreys 2023).

After *Bunny*, another watershed moment came in 2001, when Marcos Fajardo presented at the SIGGRAPH conference an early version of his *Arnold* renderer. *Arnold* was able to render scenes with complex geometry, textures, and global illumination in just tens of minutes (Pharr, Jakob, and Humphreys 2023). With the help of Sony Pictures Imageworks, *Arnold* was developed into a production-capable rendering system. It is now available as a commercial product.

In the early 2000s, Pixar's *RenderMan* renderer started to support hybrid rasterization and ray-tracing algorithms. It also introduced a number of innovative algorithms for computing global illumination solutions in complex scenes (Pharr, Jakob, and Humphreys 2023).

RenderMan was recently rewritten to be a physically based ray tracer. It was first employed in its new form for the production of the 2013 movie *Monsters University* (Hery and Villemin 2013).

Fig. 4: Monsters University, ©Disney/Pixar 2013

1.4 THESIS OUTLINE

The rest of the thesis will be organized as follows.

Chapter 2 will be devoted to the properties and behaviour of light, in a physical sense.

Chapter 3 will describe mathematical models commonly used in ray tracing. This includes also techniques for realistic reflections, such as the *Disney Principled BRDF*.

In Chapter 4 an implementation of the previously discussed techniques will be presented, in the form of the small renderer `BoxOfSunlight`. The focus here will be on the architectural choices made during development.

In Chapter 5, the `BoxOfSunlight` engine will be validated. We'll see how even just by changing the values of a few input parameters, we can simulate the looks of a wide range of materials.

Finally, Chapter 6 will be dedicated to some of the shortcomings of `BoxOfSunlight`, and will contain ideas on how it could be extended into a more complete physically based renderer.

2

PHYSICS OF LIGHT

This chapter establishes the physical foundations of our Physically Based Rendering techniques. By its end, we'll have acquired important intuitions relating to three topics in the physics of light, namely, the *dual nature of light*, the *electromagnetic spectrum*, and *light-matter interactions*. The very last subject will be a physical interpretation of the *reflection of light*.

2.1 WAVE-PARTICLE DUALITY

When we create images, we are effectively using light as a tool to communicate information to the viewers. This raises an apparently trivial question.

What *is* light?

We'll give the physicist's answer. It might seem non-sensical at first, since it's actually made up of *two*, very different, answers. We say in fact that light has a *dual nature*, in the sense that there are two ways, that is, *models*, to define what light is. One represents light as a *wave*, the other as a stream of *particles*. Now the interesting part: light is actually *both* of these at the same time. There are situations in which light behaves just like a smooth, continuous wave, and you couldn't possibly think of it as being *granular*. But then, we also find cases indicating that light *is* actually composed of small, individual "energy packets", and where the wave theory is definitely not an option. In the following sub-sections, we'll dive (although a bit superficially) into a discussion about both models, and extract some facts that will be useful to us[1].

1 Various examples and definitions that follow were taken from *light*. Encyclopaedia Britannica (Stark 2025)

2.1.1 *Wave Nature of Light*

Obviously, light is important for far more reasons than just visual perception and communication. It is a central component to our lives (actually, to *life* in general as we know it), and for this reason we're usually somewhat familiar with the physical concept of it. Frequently, we say that light is an *electromagnetic wave* (or *radiation*). It's easier to visualize what a *wave* is by thinking of *mechanical waves*. For example, we can imagine light as a phenomenon similar to water waves. We say that they are similar since they are both *disturbances* that *propagate* through space. Water waves propagate as physical displacements of water, a *material medium*. By contrast, electromagnetic waves don't require any material substance to propagate.

Using a proper definition, we say that electromagnetic waves are *oscillations* in the *strengths* of *electric* and *magnetic fields*.

There's a lot to unpack here. Most importantly, we haven't talked about *fields* yet. We can think of a field as of a function that associates some physical quantity to every point in space (and time). For example, each point in Earth's atmosphere has a temperature associated with it. This temperature can be expressed as a function of spatial coordinates and time: $T(x, y, z, t)$, where T is the temperature field, x, y, and z are the spatial coordinates, and t is the time. We say that temperature is a *scalar* field. On the other hand, *electric* and *magnetic fields* are *vector fields*, since they associate *vectors* to points in space and time. We indicate the electric field as $E(x, y, z, t)$ and the magnetic field as $B(x, y, z, t)$ (E and B for short).

We won't get much deeper into the topic of fields. For us, it's sufficient to know that E and B are abstractions, in the following sense. Two electrically charged bodies, just like two magnets, exhibit forces (repellent or attractive) on each other. The forces depend on the characteristics of *both* bodies. Now, If we focus our attention on only *one* of the bodies, between the two electric charges or the two magnets, we can consider it as the source of a *field*, *electric* or *magnetic* respectively, which extends in the surrounding space. From this point of view, the force exerted on the second body of the couple is caused by the *field*. Thinking of fields instead of forces is a more abstract approach, since, putting it simply, we need to consider only one of the two bodies to describe its influence on the space surrounding it (the field it produces). Thus, we can see that E and B are abstractions which allow for more complex reasoning.

We know that E and B are needed to properly define light, which is an *oscillation* in the strengths of both fields. The definition of light might

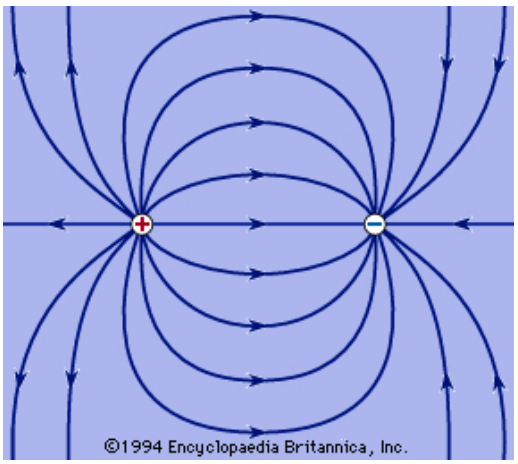

Fig. 5: "Electric field lines near equal but opposite charges." Source: *electric field*. Encyclopaedia Britannica. URL: https://www.britannica.com/science/electric-field (last accessed: 12.03.2025)

still be a bit confusing, though, so our next aim will be to *see* what an electromagnetic wave actually looks like (that is, its *graph*). In order to get there, we should start with a reminder on why we are considering the electric field E and the magnetic field B *together*.

Developments in 19th century physics, which famously culminated in *Maxwell's equations*, proved that E and B are intimately coupled; when one of them changes, the other one does as well. For example, Faraday's well-known *law of induction* describes how a moving magnet can generate electric current[2]. For this reason, we unite E and B in a single concept: the *electromagnetic field*, which describes both electric and magnetic influences produced by bodies in space.

Now, both E and B can be modeled as *time-harmonic fields* (Glassner 1995); that is, they can be described as *traveling harmonic waves*. A harmonic wave can be specified as a sinusoidal function over distance and time, which outputs the *displacement* (in our case, the strengths of the two fields). An example of a harmonic wave traveling in only the x direction is given by

$$y(x,t) = A\cos\left(\frac{2\pi}{\lambda}x - \frac{2\pi}{T}t\right) \tag{2.1}$$

where:

2 From *Faraday's law of induction*. Encyclopaedia Britannica. URL: https://www.britannica.com/science/Faradays-law-of-induction (last accessed: 12.03.2025)

- y is the vertical displacement at distance x and time t,
- A is the maximum displacement of the wave, or *amplitude*,
- λ, the *wavelength* of the wave, is the physical distance between successive crests,
- T is the *period*.

If we were standing still, on a point along the wave's path, and were to measure the time for one crest to reach us after the previous, that would get us the *period*.

Another important quantity is the wave's *frequency* ν; it is defined as $\nu = \frac{1}{T}$, and we can think of it as the "speed" at which the wave repeats. It is measured in Hertz (Hz).

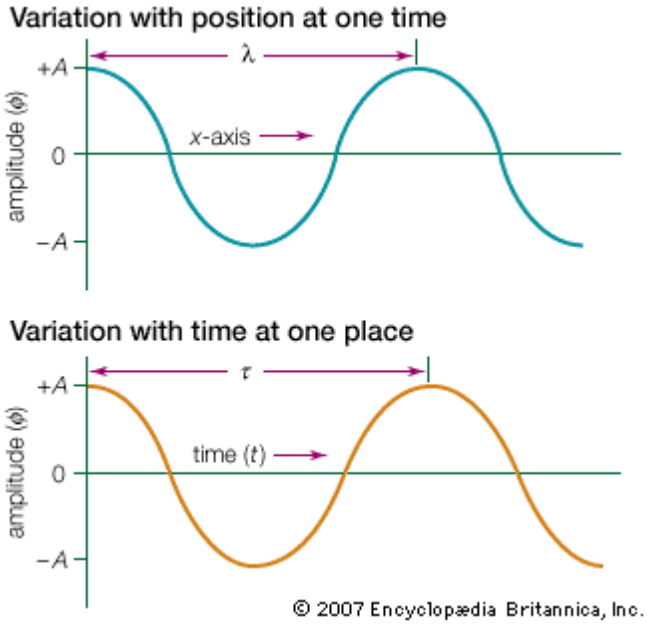

Fig. 6: "Snapshots of a harmonic wave can be taken at a fixed time to display the wave's variation with position (top) or at a fixed location to display the wave's variation with time (bottom)." Source: Encyclopaedia Britannica (Stark 2025)

Frequently, instead of writing $\frac{2\pi}{\lambda}$, we substitute it with k, the so-called *wave number*. We also use $\omega = \frac{2\pi}{T}$, which is called the *angular frequency*[3].

3 As done in *The Feynman Lectures on Physics*, Vol. I, Ch. 29 (Feynman, Leighton, and Sands 2011)

Moreover, we should mention that in the more general case, we can add a *phase shift* ϕ inside the cosine[4]. The effect of this is to, basically, shift the wave function along the x axis.

With these modifications, 2.1 becomes:

$$y(x,t) = A\cos(kx - \omega t + \phi) \tag{2.2}$$

Putting it all together, we can thus say that an *electromagnetic wave*, that is, *light*, is composed of two harmonic waves, one representing the strength of E, the other the strength of B. We also add that E and B are always perpendicular to each other, and that the magnitude and direction of propagation of an electromagnetic wave can be calculated with the cross product $E \times B$ (Glassner 1995). We can now see what the graph of an electromagnetic wave looks like by plotting E and B along the x axis (Figure 7). We define their values as

$$\begin{aligned} \vec{E}(x,t) &= E_0\cos(kx - \omega t + \phi)\hat{y} \\ \vec{B}(x,t) &= B_0\cos(kx - \omega t + \phi)\hat{z} \end{aligned} \tag{2.3}$$

To be more precise, what we are considering here is a *linearly polarized* wave, which has the property that the fields oscillate in fixed directions.

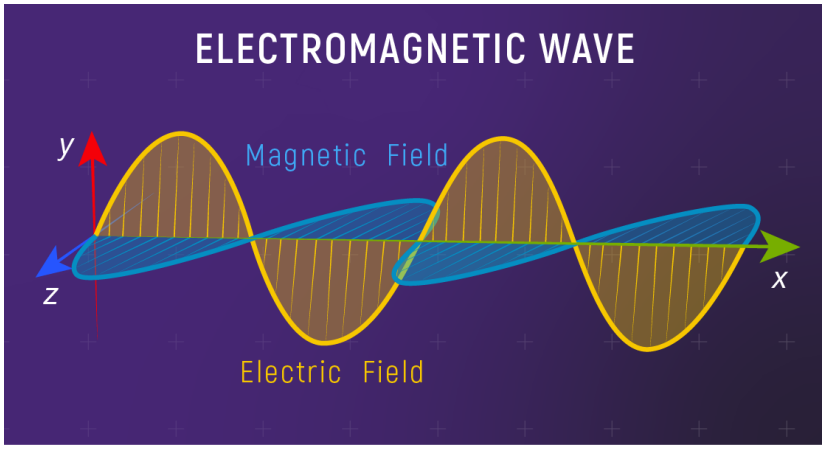

Fig. 7: Graph of a light wave. Source: NASA, ESA, CSA, Leah Hustak (STScI).

Before we move on, it's important that we address a situation that we'll encounter many times. Let's take two or more waves that are overlapping (they're "on top of each other"). Such waves are said to be in *superposition*.

4 As done in *The Feynman Lectures on Physics*, Vol. I, Ch. 21 (Feynman, Leighton, and Sands 2011)

The *superposition principle* states that, when two or more waves overlap, they produce a *new* wave, with a net displacement that is equal to the algebraic sum of the individual displacements. What this means is that the values of a wave function, be it $R(x,t)$, could actually be the result of summing up *many* different harmonic wave functions:

$$\begin{aligned} R(x,t) = A_1\cos(k_1x - \omega_1t + \phi_1) + A_2\cos(k_2x - \omega_2t + \phi_2) + ... \\ + A_n\cos(k_nx - \omega_nt + \phi_n) \end{aligned}$$

In fact, the electromagnetic waves that we'll talk about should be in general interpreted in this form. In rendering, we'll think of every electromagnetic as being composed of (infinitely) many overlapping harmonic waves. As we know, each of these harmonic waves has its own wavelength λ. Putting it simply, we can thus say that a single ray of light "contains" many wavelengths. We'll find this to be an important fact.

There are good reasons to model light as a wave. A classic demonstration of the wave nature of light is the *double-slit experiment*, first performed by Young in 1801. In this famous experiment, two parallel slits on an opaque surface are equally illuminated by a light source, and the light that passes through the slits is observed on a screen. When the slits are close together, light that strikes the screen creates a pattern of alternating bright and dark bands (Glassner 1995) (see Figure 8).

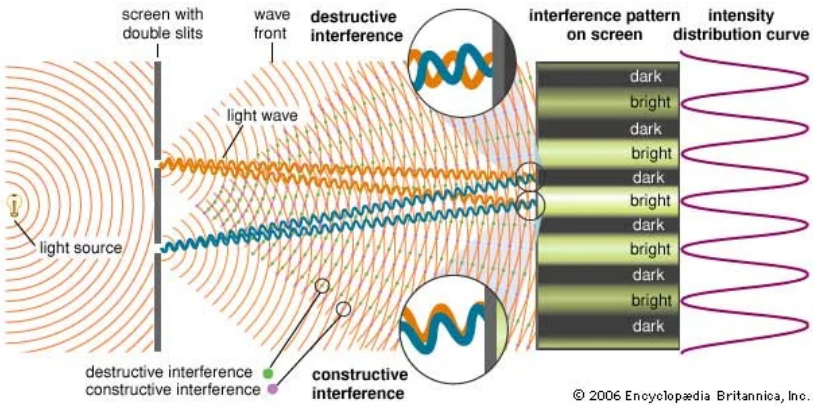

Fig. 8: "Young's double-slit experiment." Source: Encyclopaedia Britannica (Stark 2025)

The easiest way to explain this result is by positing that the light exiting the two slits has a wavelike nature. As we know, we can describe light waves with the two functions $\vec{E}(x,t)$ and $\vec{B}(x,t)$ (from (2.3)), periodic over distance and time. However, to simplify this discussion, let's express

light as a single periodic function $A(x,t)$, instead of two (with $A(x,t) = A_0 cos(kx - \omega t + \phi)$).

We start by noting that when waves pass through a slit in a barrier, the slit acts as a source of cylindrical waves (the waves propagate symmetrically beyond it). This is due to a phenomenon called *diffraction* (Glassner 1995).

In the experiment, both slits are at the same distance from the light source, so the waves leaving the two slits have the same *phase* (they're in "sync", so to speak). This means that, at any time t, the *same wave* is generated in synchrony at both slits. Now, even though the waves leaving the two slits are the same, the two distances x_1, x_2 that they travel to reach a same point on the screen will be, in general, different ($x_1 \neq x_2$). Consequently, their wave functions in that point might differ, that is, $A(x_1) \neq A(x_2)$ (with t fixed). In some points, for example, one wave arrives at its maximum and the other at its minimum, causing their sum to be 0; the waves here leave a dark spot, and we say that they *destructively interfere*. In some other points, both wave functions might arrive at their maximum, creating a bright spot. In this case, they *constructively interfere*. Between these two extremes we get different intensities due to different amounts of *interference* between the two waves.

This is exactly what causes the light-and-dark pattern that we wanted to explain (Glassner 1995).

2.1.2 *Particle Nature of Light*

Not all physical phenomena related to light can be explained by thinking of waves. Most famously, the wave model disagrees with the *photoelectric effect*, observed first by Heinrich Hertz in 1887.

Consider an experimental setup which consists in a beam of light shining onto a piece of metal, called the *cathode*. Next to the cathode, we also set up a detection device that can measure the energy of electrons striking it. What happens in this scenario is that, as soon as we shine light on the cathode, the detector starts reporting electrons. Clearly, the light that strikes the cathode triggers an expulsion of electrons from the metal. For each electron, we find its energy E to be[5]

$$E = h\nu - p \tag{2.4}$$

5 Note that the energy of the electron E is a result of its *movement* as it's expelled, and is called *kinetic energy*.

where ν is the frequency of the incident wave of light, p is a constant characteristic of the metal, and h is a factor that seems to be constant for all metals and all wavelengths (Glassner 1995). This expulsion of electrons caused by light is what we call *the photoelectric effect*.

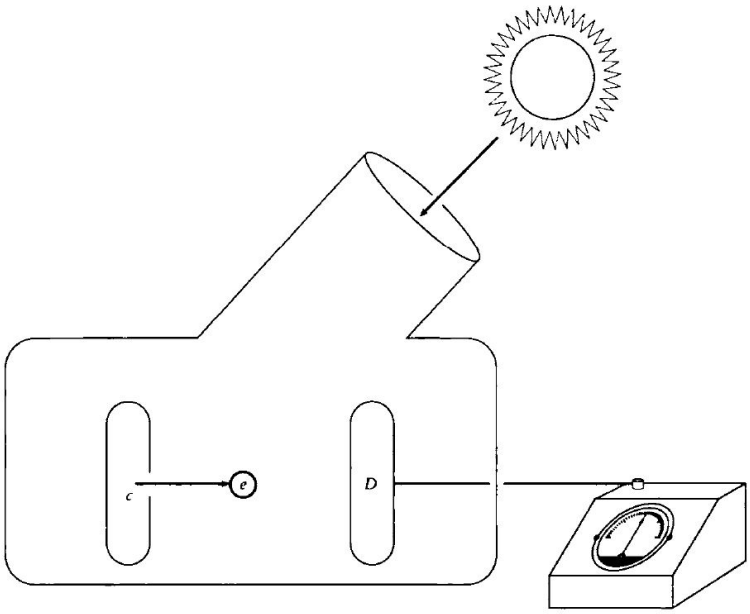

Fig. 9: "Apparatus for observing the photoelectric effect. The cathode is *c*, the detector is *D*." (Glassner 1995)

If we experiment with different light "strengths", or, to be more precise, different wave *amplitudes*, strange things start to happen. First of all, the energy E of the electrons is found to be independent of the light beam's amplitude. For example, illuminating the metal with a 40-watt light bulb rather then an otherwise identical 20-watt light bulb causes more electrons to be freed, but their individual energy does not change. According to the wave model, this doesn't make much sense, since a stronger wave should impart more energy to the electrons. Another surprising fact is that even extremely low amplitudes of light free some electrons. Again, thinking of the light beam as a wave, we would be led to believe the incident energy to be so low that it gets spread *all over* the cathode. Consequently, there wouldn't be any point with enough energy to free an electron[6] (Glassner 1995).

6 More precisely, the energy wouldn't be higher than the constant p from (2.4)

In 1905, Einstein advanced the hypothesis that the energy flowing along the incident beam is quantized into small, individual packets called *photons*. He supposed that to liberate an electron from a surface, there's a minimum amount of energy that's required. A photon that contains this amount of energy transfers it to the electron at the time of collision, resulting in the electron being freed. On the other hand, photons with energies lower than this minimum cannot induce electron emission (Glassner 1995).

Going back to the two phenomena mentioned earlier, we can now see that each photon actually interacts with each electron *independently*. Therefore, if we increase the incident beam's energy, that is, the *number* of photons, we don't increase the energy of the emitted electrons, we just produce *more* of them. On the other hand, even a beam with low energy (less photons) will trigger the emission of some electrons, if the energy carried by individual photons is over a certain threshold.

Einstein's interpretation of the *photoelectric effect* was based on the work of Max Planck, who in 1900 had proposed a mathematical construct to calculate *blackbody radiation*. (We'll mention more about *blackbodies* later.) Planck had considered electromagnetic energy as *released* and *absorbed* in discrete packets, with the energy of a packet being directly proportional to the *frequency* of the radiation:

$$E = h\nu \tag{2.5}$$

Where the constant h has since been called *Planck's constant*. Einstein's contribution, which earned him the Nobel price in 1921[7], was interpreting *light itself* as composed of energy packets, that is, *photons*.

So, the particle model is better suited to explain phenomena such as the *photoelectric effect* and *blackbody radiation*. It is still, however, indisputable that light is also a wave, due to the manifestation of *interference effects* such as the ones present in Young's experiment. But how can both theories be right?

Quantum mechanics solves this dilemma by stating that light exhibits *wave-particle duality*. Actually, this is also true for *electrons* and other discrete bits of matter; that is, they also possess wave properties such as *wavelength* and *frequency*. For example, in modern versions of the double-slit experiment, electrons have been shown to form the same interference pattern as the one demonstrated originally by Young (Stark 2025).

7 See https://www.nobelprize.org/prizes/physics/1921/summary/ (last accessed: 12.03.2025)

But how does this work? Briefly put, each particle has a *wave function* associated with it, which should be interpreted *statistically*, as suggested originally by the German physicist Max Born. In fact, the *wave function* is required to calculate the *probability* of finding a particle at any point in space (Stark 2025).

We now close our discussion on the dual nature of light. We started with what seemed like a simple question and ended up in the realm of modern physics, which have challenged not only earlier theories in the field, but our own perception of reality. It seems that to answer the question "what is light?" we need embrace a vision of the Universe where concepts that baffle the intellect, like particles being also waves, and waves being also particles, are a reality.

2.2 THE ELECTROMAGNETIC SPECTRUM

From here on, we'll alternate freely between the wave and particle models. Let's now consider the first. Changing the wavelength of an electromagnetic radiation, we can span a broad *spectrum*, from very long waves to very short. This spectrum can be divided in *bands*, each with a different name. For the sake of curiosity, we'll provide a short summary of them, based on the NASA Science article series *The Electromagnetic Spectrum* (National Aeronautics and Space Administration, Science Mission Directorate 2010).

2.2.1 *From Radio Waves to Gamma Rays*

The longest waves in the spectrum are *radio waves*. Their existence was proved by Hertz in the late 1880s, and their wavelengths range "from the length of a football to larger than our planet". As suggested by the name, they are used to transmit *radio signals*.

At the higher frequency end of the radio spectrum, *microwaves* can be found. They are distinguished from radio waves because of the technologies used to access them. They are used by microwave ovens to cook food.

Really hot objects, like fire, emit *visible light*. Other objects, such as humans, are not as hot and only emit only so-called *infrared waves*. Infrared waves are just beyond the visible spectrum of light; although the human eye cannot see them, we do sense them as *heat*.

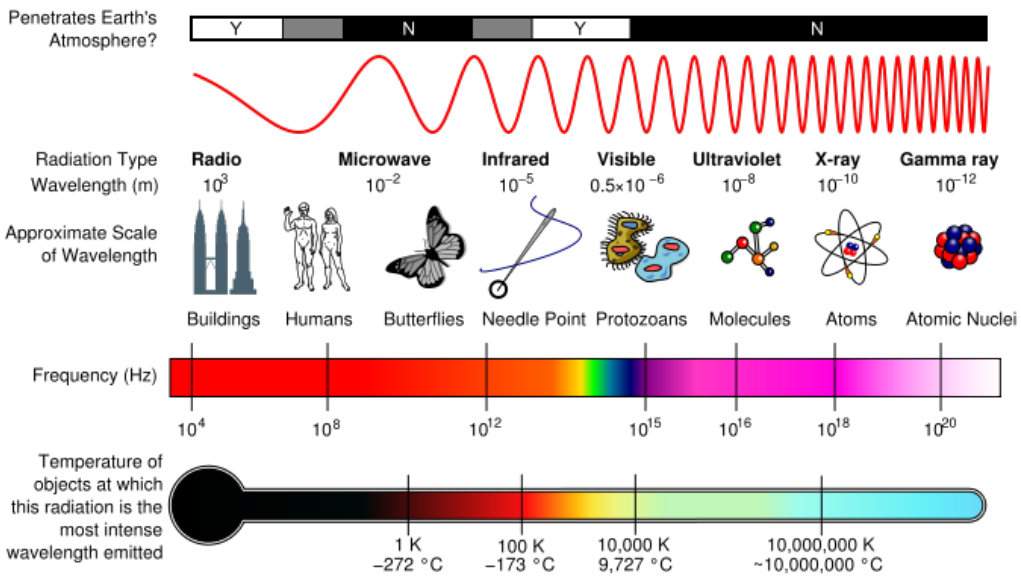

Fig. 10: "Electromagnetic Spectrum with Radiation Types." Source: *Electromagnetic Radiation*. LibreTexts, Physical and Theoretical Chemistry. (license URL: https://creativecommons.org/licenses/by/4.0/)

Visible light falls roughly in the wavelengths

$$380 < \lambda < 700$$

measured in *nanometers* (nm).
There is a direct correlation between the wavelength of a visible ray of light and the color that our eyes see in response. Short wavelengths look blueish (the shortest waves look violet), and long wavelengths red. In the middle, there's green.

Ultraviolet light (*UV*) has shorter wavelengths than visible light. The Sun is a source of ultraviolet radiation, some of which can be harmful to living organisms. Luckily, the vast majority of these are absorbed by Earth's atmosphere.

X-rays possess extremely small wavelengths, between 0.03 and 3 nanometers (that is smaller than a single atom of many elements). They are used for medical diagnosis, since bones are dense and absorb more x-rays than skin does, as rays pass through the body.

The smallest wavelengths (and highest energies) are reserved to *gamma rays*. On Earth, they are generated from events such as nuclear explosions and lightning.

We'll now focus a bit more on *visible light*, to explain how we perceive colors and, therefore, how we should treat them in a computer program.

2.2.2 *Spectral Distributions and Color*

When we'll be defining physical quantities for the measurement of light, we'll find it necessary to talk in terms of *spectral distributions*. We'll also refer to these, simply, as *spectra* (*spectrum* for the singular). A *spectral distribution* is a distribution function that gives us the amount of light as a function of wavelength (Pharr, Jakob, and Humphreys 2023).

To make an example, let's consider the energy of a ray of light. We know by now that it can be decomposed into discrete packets, each of which is carried by a photon. We are also aware of the fact that, given a light wave's frequency ν, the energy E of its photons is (from (2.5))

$$E = h\nu$$

We've mentioned earlier that a light wave can be considered as the sum of many harmonic waves, each with its own wavelength λ. A light wave's frequency ν can be calculated from its wavelength λ as

$$\nu = \frac{c}{\lambda} \tag{2.6}$$

where c is the speed of light in a vacuum (Stark 2025). Consequently, a light wave "contains" many frequencies, just like it does with wavelengths. This means that it's composed of many types of photons; that is, photons with different energies. These individual energy values are summed up to obtain the total energy carried by light.

Let's now see how the light's energy could be used to determine its color. To do so, we follow a simplified overview from the book *An Introduction to Ray Tracing* (Glassner 2019).

We know that light is composed of photons at many different frequencies (or, equivalently, wavelengths). When these photons reach our eyes, each of them will be "deciphered" as a *different color* from the visible spectrum (based on the wavelength). What happens next is that these base colors are blended together by the eye. The resulting coloring will be tinted towards the base colors for which there were more photons (of that wavelength). This means that, for example, if most of the light's energy was carried by photons with very long wavelengths, it will appear more red (it will be more intense at the corresponding wavelength). On

the contrary, if energy came mostly from short-wavelength photons, the light will look more blueish.

So, in our case, we don't really care for the light's *total* energy: it only reveals the light's "total" intensity. What we actually want to know is how much energy there is *per-wavelength* (we're asking ourselves "how red is this light?", or "how blue?"). Thus, we want to be able to *distribute* the total energy between the range of wavelengths that the light ray is made of. This is exactly the purpose of a *spectral distribution function*; in this case, it's called *spectral energy distribution* (or *SED*)[8].

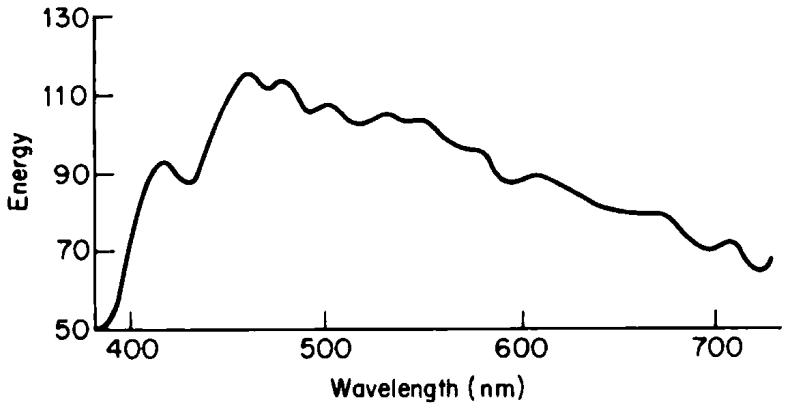

Fig. 11: "The spectrum of CIE Standard Illuminant D6500, which approximates sunlight on a cloudy day." (Glassner 2019)

Putting it simply, we thus say that colors are the result of a "blending" operation, which can be predicted through spectral distributions. At this point, we might be led to think that colors *are* spectral distributions. However, we should be careful not to mix up a purely physical concept such as spectra with the *experience* of colors, which is strongly related to human *perception*. The human visual system actually employs some tricks that we can exploit for our own rendering purposes. Most notably, we can avoid representing each color as a distribution over all wavelengths.

Since the 1800s, numerous experiments have proven that all colors perceived by the human eye can be represented using three scalar values. This is called the *tristimulus theory*, and it's been confirmed by the study of *cone cells* (Pharr, Jakob, and Humphreys 2023). *Cone cells* are light-sensitive cells on the retina which allow us to see colors. There are three

8 From the NASA IPAC Teacher Archive Research Program (NITARP) wiki, at: https://coolwiki.ipac.caltech.edu/index.php/SED_plots_introduction (last accessed: 12.03.2025)

types of cone cells, which we can call *short*, *medium* and *long*, after the wavelengths of light they are most sensitive to. To each of these cone cells, we can associate a *response function*, which describes how strongly one type of cone "responds" to beams of light of different wavelengths. Let's call these functions $k_s(\lambda)$, $k_m(\lambda)$ and $k_l(\lambda)$ (for short, medium and long cones respectively). Since each ray of light with a specific (or "single") wavelength λ produces three response values on the retina, one for each type of cone, we can visualize this response as a single point in a 3D space:

$$[k_s(\lambda), k_m(\lambda), k_l(\lambda)]^t$$

We call this space a *color space*[9]. Light rays which don't have a specific wavelength (they have "multiple" wavelengths) produce colors that are just linear combinations of the "simpler" ones; that is, the ones that *did* come from light rays with well-defined wavelengths (so, the colors are still just points in the color space) (Gortler 2012). Note that, since we're just speaking of coordinates in a 3D space, we can easily describe the same color using a different basis (any three linearly independent base colors).

It is well known that colors of computer pixels are the result of blending together red, green and blue (*RGB*), and now we can clearly see why this works. In fact, red, green and blue constitute the basis of the *RGB color space*.

In our implementation of Physically Based Rendering techniques, we'll be using RGB to represent *spectral quantities*; that is, values that vary with wavelength, like the energy of a light ray. Advanced rendering engines, such as *pbrt*, actually employ *spectral rendering*, where full spectra are used for higher precision (Pharr, Jakob, and Humphreys 2023). We'll consider the loss of information caused by RGB color spaces as negligible for visual aspects.

Before we conclude this section, we should also mention that digital images are, in general, not stored using the RGB format. More commonly, computers use the *sRGB color space*, which involves a non-linear transformation from RGB, for the purpose of storing color values more efficiently in the computer's memory (Gortler 2012). This, however, in rendering, will be our concern only at the moment of reading or storing an image

9 We should mention that what we're talking about here is called *retinal color*. The *perceived color*, the one we experience and base judgements upon, is actually the result of some non-trivial processing steps done by the human visual system. For example, our brains make us perceive the colors of objects as more or less constant, even under different illuminants. (Gortler 2012)

from/to memory, while our physically based calculations will be done in the linear, RGB space.

2.3 LIGHT-MATTER INTERACTIONS

Now we have a pretty good mental model of light, but we should remember that we also need one for materials. To build material models, we need to talk about *matter*. Matter can either *emit* light, as a light source, or *respond* to incident energy (for example, through *reflection*). We'll consider both cases.

For the moment, however, we should settle on a model to describe matter. We'll be considering the simplest form of it: the atom.

It is well-known that the classical model of the atom includes a center, the *nucleus*, composed by small particles called *protons* (with positive charge $+e$) and *neutrons*. The nucleus is also orbited by another group of much smaller particles, called *electrons* (with negative charge $-e$).

In quantum mechanics, these particles don't have a precise location. Basically, unless a particle is observed, it is *nowhere*; particles are only associated with *probabilities*. For example, instead of individual electron particles, we have a *cloud of electrons*, with the cloud being more dense where electrons are more likely to be found (Glassner 1995).

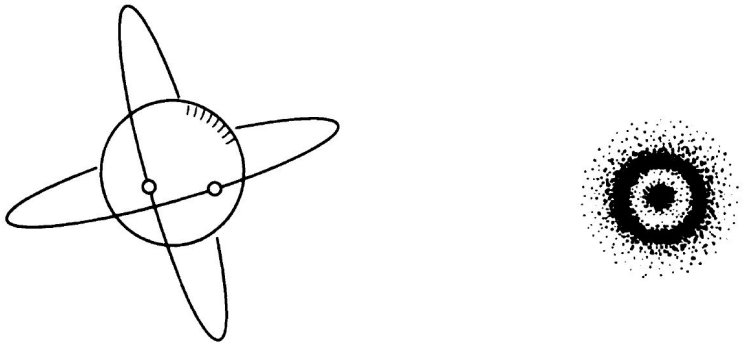

Fig. 12: The classical atom model (left) compared to a more modern view (right). (Glassner 1995)

Electrons are very susceptible to external influences, which make them move between different *states* by *absorbing* and *releasing* energy, often in the form of photons (Glassner 1995). The state of an electron at any time is given by a set of four *quantum numbers*. The *principal quantum number* n characterizes the electron's energy, with $n \epsilon \{1, 2, 3, ...\}$. Higher values of

n indicate that the electron is more likely to be found far away from the nucleus[10].

The electrons of a neutral atom normally inhabit *ground states*. Apart from these, however, there are many higher-order *excited-state energy levels*. Above, we find the *ionization continuum*, where electrons become disassociated from the atoms and are free to move away (Glassner 1995). We have already seen this in the photoelectric effect, where electrons where expelled from atoms with kinetic energy E.

Finally, we can say a couple things about molecules. A molecule is an electrically neutral, stable combination of two or more atoms. The energy transitions for electrons change when they are involved in a bonding process that brings atoms together into molecules. Also, the size (much greater compared to atoms) and translational/vibrational energy of molecules affects what energy is absorbed and emitted by their atoms (Glassner 1995).

2.3.1 *Emission*

We can classify the emission of light into two distinct types: *thermal* and *luminescent* (Glassner 1995).

Thermal emissions are caused by heat. We can explain this phenomenon as follows. It is known that, when the temperature of an object is above absolute zero, its atoms are moving. As is described by *Maxwell's equations*, the motion (more precisely, the *acceleration*) of atomic particles that hold electrical charges causes objects to emit electromagnetic radiation (Pharr, Jakob, and Humphreys 2023). In fact, we've already mentioned that humans emit light at infrared frequencies, and that, to emit light that is visible, an object needs to be much warmer. This is because, to put it simply, the atoms need to move *faster*. An incandescent light bulb is an example of a *thermal radiator*; it contains a small filament which, when heated by the flow of electricity, emits electromagnetic radiation (Pharr, Jakob, and Humphreys 2023).

When talking about thermal emission, it is common to run into the term *blackbody* (actually, we already have while discussing the particle nature of light). Blackbodies are (theoretical) perfect absorbers; they absorb all incident light, without reflecting any of it. They are also perfect emitters; that is, they convert *power* to electromagnetic radiation as efficiently as

10 From *orbital*. Encyclopaedia Britannica. URL: https://www.britannica.com/science/orbital (last accessed: 12.02.2025)

physically possible[11]. Even though true blackbodies are not physically realizable, the idea of them is still quite useful. This is because *Planck's law* specifies a way to, given a temperature, determine the emission of a blackbody as a function of wavelength. Taking a non-black body emitter, if the shape of its emitted spectral distribution is similar to the one of a blackbody at some temperature, we say that the emitter has the corresponding *color temperature*. Color temperatures over 5000 K are generally described as "cool," while those at 2700–3000K "warm". For example, the CIE standard illuminant A, which represents the average incandescent light, corresponds to a blackbody radiator of about 2856 K (Pharr, Jakob, and Humphreys 2023).

Luminescent emission is due to energy stored (perhaps for a very short time) in the material, and is determined primarily by factors different that temperature, although the temperature can affect the material (Glassner 1995). In the context of *luminescent emission*, it's important that we mention *phosphors*. Speaking broadly, a *phosphor* is a material that absorbs some form of energy, like an electromagnetic wave or an electron beam, and then emits it as light over some period of time (Glassner 1995). When certain phosphors *luminesce* from *electron excitation*, the process is called *electroluminescence*. This is used in the production of TV screens and computer monitors[12].

Given our model of the atom, we can actually explain luminescence. We already know that electrons tend to move through different states, absorbing and emitting energy in the process, and that this energy is often in the form of photons. When a photon is absorbed, it generally disappears completely (a photon cannot exist at rest, and cannot transfer only *some* of its energy), and the electron transitions into what we called an *excited-state energy level*. Generally, this can't be kept up for long, so after a while the electron will drop back to its *ground state*, emitting the difference in energy between the two states as a *new* photon. Most times, this happens under 10^{-8} seconds. This number is used to divide phosphors into two categories. If the material responds to incident energy by reradiating most of it within 10^{-8} seconds, we call it *fluorescent* (this is

11 This makes sense if we think of emission as of the reverse operation of absorption. (Pharr, Jakob, and Humphreys 2023)

12 From *phosphor*. Encyclopaedia Britannica. URL: https://www.britannica.com/science/phosphor (last accessed: 12.03.2025)

what happens, for example, in *fluorescent lamps*[13]). If the emission persists longer, we're talking about *phosphorescence* (Glassner 1995).

2.3.2 *Response*

We now consider some common events that can be triggered when light strikes matter.

To do so, we'll first move back to a simpler model of light; we won't be thinking of waves or particles, but of *rays*. This is the point of view of *geometrical optics*, where we describe the interaction of light with objects much larger than its wavelength, allowing for this abstract idea of *rays of light* (Pharr, Jakob, and Humphreys 2023).

To predict how matter will respond to light, we can use a property called the *index of refraction* (*IOR*). This is a complex number, where its real part describes how much the matter slows down the speed of light (which, as we know, is equal to c in a vacuum), and its imaginary part determines whether the light is *absorbed* as it propagates in the material (Hoffman 2012). We should mention that by *absorption* we mean the conversion of the energy of light into some other energy form, internal to the atoms. One example for this is *thermal energy*, which derives from the *movement* of atoms[14].

We'll now consider matter as a *medium* through which light is propagating (just like it does through air, for example). We can distinguish two types of media: *homogeneous* and *heterogeneous*.

We say that a medium is *homogeneous* if it presents a uniform index of refraction (at the scale of the light's wavelength). Water and glass are examples of this. We also say that they're *transparent* media, since they don't absorb visible light in any significant way. In fact, their refraction indices have a very low imaginary part for visible light wavelengths (Hoffman 2012).

On the other hand, if the homogeneous medium *does* absorb light from the visible spectrum significantly, what happens is that the light's intensity dies down as it moves farther into the medium. However, the direction of light does not change.

13 From *fluorescent lamp*. Encyclopaedia Britannica. URL: https://www.britannica.com/technology/fluorescent-lamp (last accessed: 12.03.2025)

14 From The IBM article *What is thermal energy?*, at https://www.ibm.com/think/topics/thermal-energy (last accessed: 12.03.2025)

Fig. 13: Light in transparent media like water and glass just keeps on propagating in a straight line at the same intensity and color. (Hoffman 2012)

We should note that these properties are always in function of *scale*. For instance, on the scale of many feet of distance, water actually absorbs quite a bit of light, especially red colors (Hoffman 2012).

Fig. 14: Over large distances, the slight absorptivity of water becomes noticeable. (Image by Andreas Schau, from Pixabay).

Let's turn now to *heterogeneous media*, which have variations in the index of refraction. If the index of refraction changes slowly and continuously, then the light bends in a curve. If it changes abruptly, the light *scatters*, meaning that is splits into multiple directions. We can think about light

encountering microscopic particles, which induce regions in space where the index of refraction becomes different. The type of particle affects the distribution of the scattered light's directions (Hoffman 2012).

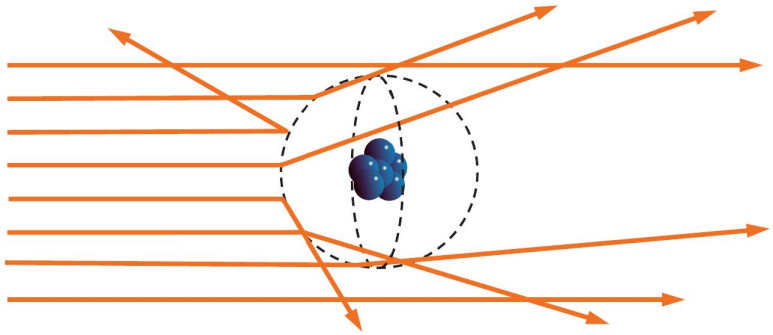

Fig. 15: "Particles cause light to scatter in all directions." (Hoffman 2012)

Media that are *translucent* (or *opaque*) contain such a high density of scattering elements that, as a result, light gets scattered in completely random directions.

Fig. 16: Example of an opaque medium. (Hoffman 2012)

Most media both scatter and absorb light, at least to some extent.

Now, we want to know what happens when light, propagating through air, hits an object's surface. An analytical solution to the problem can be found if we consider the surface of the object as infinite and perfectly flat (relatively to the light's wavelength). In this case, we get special solutions to Maxwell's equations, called the *Fresnel equations*. These equations tell

us that the material causes light to split only in two directions: *reflection* and *refraction*. They also describe the portions of reflected and refracted light (Hoffman 2012).

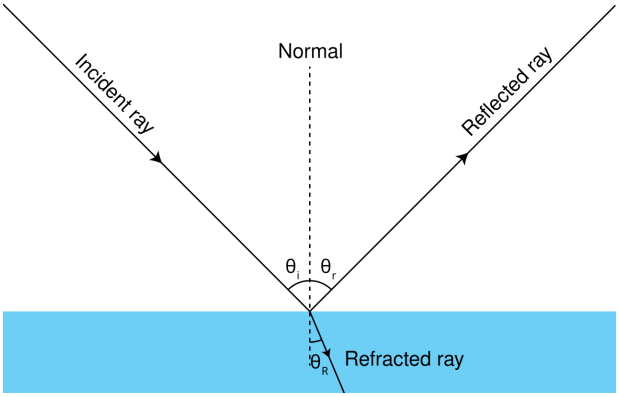

Fig. 17: An incident ray gets split into a reflected ray and a refracted ray. Source: Wikimedia Commons.

The angle formed by the reflected ray of light with the surface *normal*[15], that is, the *angle of reflection*, is equal to the angle of the incoming ray. The *angle of refraction* depends on the refractive index, and can be found through *Snell's law*.

2.3.3 *Reflection and Subsurface Scattering*

If real-world surfaces were really perfectly flat, computing realistic reflections would be extremely simple, in the sense that we would just reflect the incident light vector. In most cases, however, surfaces present irregularities larger than the light's wavelength, but too small to be seen by the eye. To account for this, we can model the surface as a large collection of small optically flat surfaces, which will cause light to be reflected in various directions, and various amounts for each direction. The resulting surface *roughness* will cause reflections to be more blurry.

Let's now consider the refracted light; what happens to it depends on the type of material. Metals immediately absorb refracted light (more precisely, their free electrons do). As for non-metals (also called *dielectrics* or *insulators*), light behaves the way we would expect, in part getting absorbed and in part scattered. Most times, the refracted light gets scattered

15 the *normal* to a surface in a given point is a vector perpendicular to the surface in that point

Fig. 18: Light gets reflected off a surface, modeled as a collection of smaller, optically flat surfaces. Image courtesy of Joey de Vries, at https://learnopengl.com/PBR/Theory. (last accessed: 12.03.2025) (license: https://creativecommons.org/licenses/by/4.0/)

so much that it's re-emitted out of the surface, in general, at a different point. This is called *subsurface scattering* (Hoffman 2012).

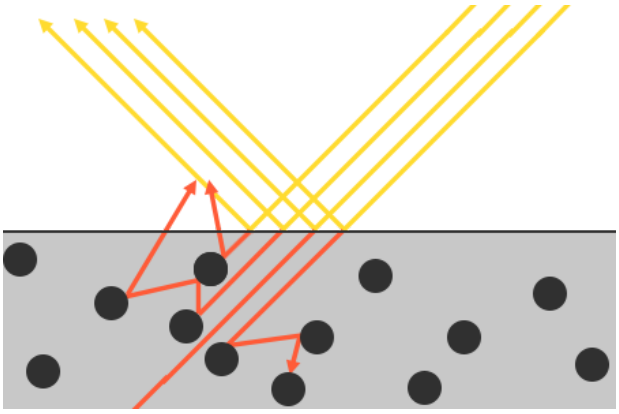

Fig. 19: The refracted light scatters until it re-emerges from the surface. Image courtesy of Joey de Vries, at https://learnopengl.com/PBR/Theory. (last accessed: 12.03.2025) (license: https://creativecommons.org/licenses/by/4.0/)

2.3.4 *Reflection, as seen by Electrodynamics*

Here, we'll form a quick intuition on how the phenomenon of reflection fits into the framework of *classical electrodynamics*. This means that we'll pretend again that light is simply a wave. Addressing the reflection of *photons* is a problem that belongs to *quantum electrodynamics* (Feynman, Leighton, and Sands 2011), and would constitute a topic too complex for this thesis.

We'll be going through a very brief summary of the Feynman lectures on electrodynamics (volume I, lectures 28-30) (Feynman, Leighton, and Sands 2011).

Remember that Maxwell's equations tell us that an accelerating charge produces an electromagnetic field. To be more precise, we consider the case where a charge is moving nonrelativistically at a very large distance.

We can also have many charges instead of one. If we can make them move together, all in the same way, we can get the resulting field just by summing the effects of the individual charges. This is a consequence of the *superposition principle*, which we have already encountered.

Let's be more concrete now, and conceive an experiment. We have two pieces of wire connected to a generator (as shown in Figure 20). The generator makes a *potential difference*, which first pulls electrons from piece A and pushes them into B, to then almost immediately reverse the effect, and make the electrons move from B into A. This way, the charges accelerate up and down in the wires. This is the same as having a single charge (summing the effects of the individual charges) accelerating up and down as though A and B were a *single* wire. When the wire is very short compared to the distance traveled by light in one oscillation period, it is called an *electric dipole oscillator*.

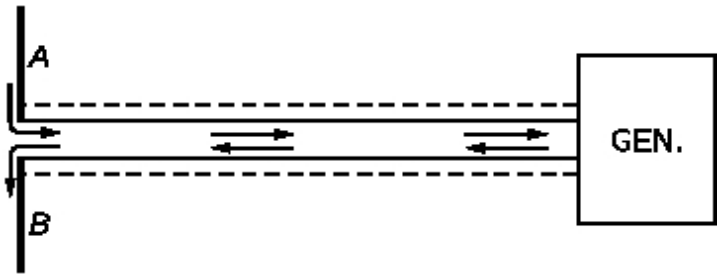

Fig. 20: "A high-frequency signal generator drives charges up and down on two wires." (Feynman, Leighton, and Sands 2011)

Our accelerating charge generates an electric field, which we pick up with an identical instrument; that is, another pair of wires just like A and B. We call this a *detector*. The incoming electric field will produce a force which will pull the electrons up on both wires or down on both wires. The resulting signal is then detected by a *rectifier* mounted between A and B, and amplified by an *amplifier*, so that we can perceive it in some form (like sound).

With this setup, we can observe that the field is strongest when the detector is parallel to the generator, and is 0 when they are perpendicular. In fact, what matters when calculating the field is the acceleration of the charge *projected* on the plane *perpendicular* to the *line of sight*.

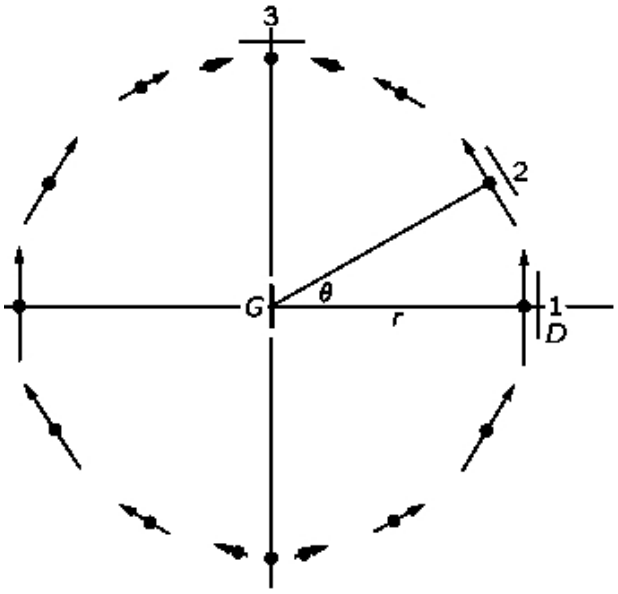

Fig. 21: The detector *D* finds a strong field when parallel to the detector *G* at point 1. The field is 0 when the detector is at 3. (Feynman, Leighton, and Sands 2011)

We now consider a situation where there are n oscillators, equally spaced at a distance d, all with the same amplitude, and lying on the same line (Figure 22). Their phases are different to the observer, both because of some intrinsic shift in phase from one to the next, and because we are looking at them at different angles, resulting in different distances traveled by the individual waves to reach us.

The intrinsic shift in phase, one to the next, is α. If we are observing in a given direction θ from the normal, there is an additional phase difference contribution $2\pi d \sin\theta/\lambda$, because of the time delay between each successive oscillator. The sum of the waves becomes

$$\begin{aligned} R = & A[\cos\omega t + \cos(\omega t + \phi) + \cos(\omega t + 2\phi) + ... \\ & + \cos(\omega t + (n-1)\phi)] \end{aligned} \tag{2.7}$$

where $\phi = \alpha + 2\pi d \sin\theta/\lambda = \alpha + k d \sin\theta$ is the net phase difference between one oscillator and the next one.

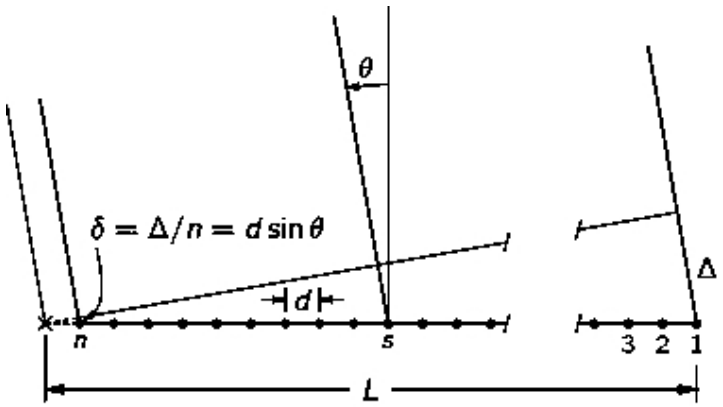

Fig. 22: "A linear array of n equal oscillators." (Feynman, Leighton, and Sands 2011)

If we consider the intensity of the resulting wave (2.7), we notice that we have a maximum when $\phi = 0$ (all the oscillators are in phase). We also get maxima, in general, when $\phi = 2\pi m$, where m is any integer (successive waves are out of phase by a multiple of 2π).

Now suppose that we have a source of electromagnetic radiation that's far away, practically at infinity, and that its light is coming in at an angle θ_{in}. The corresponding electric field will cause the electrons to move up and down in the wires, and in moving, they will generate *new* waves. This phenomenon is what we call *scattering*. In the case of a material hit by a light wave, its electrons will also start moving just like for the oscillators, generating new waves.

In *diffraction gratings*, instead of wires we have a flat piece of glass with tiny notches in it, each of which represents a source of slightly different scattered waves. Diffraction gratings can be used to split light based on its wavelength (that is, its color). In one of its forms, a diffraction grating is composed of just a plane glass sheet (transparent and colorless) with scratches on it, all equally spaced.

So, we have a light source at infinity, which sends light to the scratches at an angle θ_{in}. We are interested in the scattered beam at the angle θ_{out}, which is the angle from which we are observing the grating (we called it θ earlier). The incident light will hit the scratches one after the other, with some time delay, resulting in a phase shift between adjacent scratches. This can be calculated as $\alpha = -2\pi d \sin\theta_{in}/\lambda$. Putting it all together, we have:

$$\phi = 2\pi d \sin\theta_{out}/\lambda - 2\pi d \sin\theta_{in}/\lambda$$

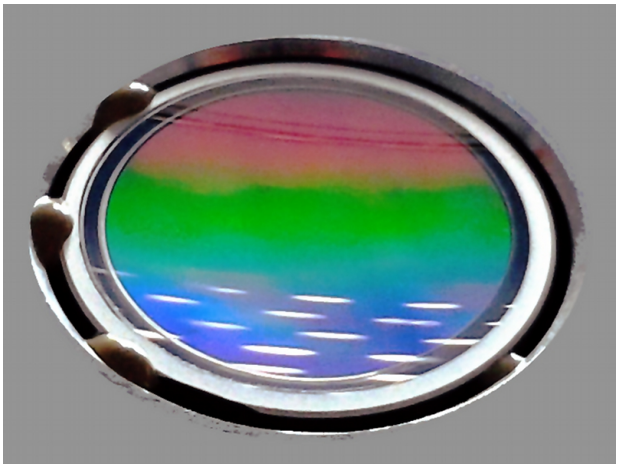

Fig. 23: A parabolic reflective diffraction grating. Source: Wikimedia Commons. (license: https://creativecommons.org/licenses/by-sa/3.0/deed.en)

To have maximum intensity, we know that ϕ should satisfy $\phi = 2\pi m$:

$$2\pi m = 2\pi d \sin\theta_{out}/\lambda - 2\pi d \sin\theta_{in}/\lambda$$

Multiplying both sides by $\lambda/2\pi$, we get

$$\lambda m = d\sin\theta_{out} - d\sin\theta_{in}$$

Now, if d is less than $\lambda/2$, this equation can have no solution except $m = 0$. In this case, we have

$$\sin\theta_{in} = \sin\theta_{out}$$

Which may mean two things: either $\theta_{out} = \pi - \theta_{in}$ or $\theta_{out} = \theta_{in}$.

- In the first case, it's as if the light goes "right through" the grating.
- In the second, we have *reflection*: the *angle of incidence* is equal to the *angle of scattering*.

We can now end this chapter with a solid physical interpretation of reflection, which goes as follows. When light hits a surface, it generates motion in the atoms (or electrons, to be more precise) of the object, causing them, in turn, to regenerate *new* electromagnetic waves. These

waves will add up, resulting in a *single* wave, produced by the object as a whole. If the spacing of the scatterers is small compared with one wavelength, the reflected wave will be strongest in intensity (only) at the angle $\theta_{\text{out}} = \theta_{\text{in}}$ (ignoring light that goes "right through"). Thus, we consider the direction of scattering as symmetric to the direction of incident light, with respect to the surface normal.

3

MATHEMATICS FOR RAY TRACING

Whereas the previous chapter was mostly a description of light phenomena in the physical world, here we wish to put together more abstract, mathematical ideas that can let us *imitate* the behaviour of light inside a computer. Our final goal is to simulate light reflections in a way that makes virtual materials feel "real", and the abstraction process that we'll be going through will lead us exactly there.

First of all, we should note that practically all nontrivial graphics programs require a geometric foundation (at least to *some* extent), this foundation being made of *vertices*, *polygons*, and similar constructs (Pharr, Jakob, and Humphreys 2023). In our case, we'll be building objects out of two simple primitive shapes: triangles and spheres. At an even lower level, we'll be dealing with *points* and *vectors* in 3D space. Vectors will also, and more importantly, be needed to define *rays of light*.

Once a 3D scene is built, it needs to be shown to the viewer. This is where *ray tracing* will come in. Its job will be to simulate the path taken by light, from the *light sources* to the *virtual camera*, after (potentially) being *reflected* by an object[1]. Light will travel along *rays*, and the reflection calculations will be conducted with the help of a powerful mathematical tool: the *Bidirectional Reflectance Distribution Function* (*BRDF*).

3.1 RAY TRACING FUNDAMENTALS

In this section we'll get an idea of how ray tracing works, and what makes it a powerful rendering technique. We'll also go through the fundamental mathematical operations that simple ray tracers[2] rely on.

1 Actually, we'll see that in ray tracing we *reverse* this path.
2 A *ray tracer* is a renderer which uses ray tracing.

3.1.1 *What is Ray Tracing*

First of all, we should point out that ray tracing is only *one* of the various rendering techniques that are commonly used in computer graphics. Another especially important method is called *rasterization*, which is at the base, for example, of the widely-adopted *OpenGL* graphics API (see Chapter 4 for more about *OpenGL*).

In a rasterization-based rendering environment such as OpenGL, 3D objects are represented as collections of triangles (also called *triangle meshes*), which are then projected onto the screen to figure out what pixels are covered by them[3], to finally determine the appropriate colors of those pixels by perform "some" computation. These final computations (which are also called *shading*) typically involve equations that simulate the way that light reflects off of the triangles, based on the materials they're made of (Gortler 2012).

Through the use of *z-buffering*, OpenGL draws on screen only the triangles that are closest to the viewer. The way this works is that OpenGL goes though the list of triangles, one by one, and, while coloring the pixels they cover, it also memorizes a *distance* for them. This distance is relative to the virtual camera, and is memorized in a buffer called the *depth buffer* (or *z-buffer*). Successive triangles are then drawn only if they are closer than what was drawn earlier, and this is checked on a per-pixel basis in the depth buffer. This way we can have triangles drawn on top of each other correctly. We refer to this as *z-buffering* because, in our 3D space, we consider the viewing direction to be along the z axis.

We can thus summarize rasterization in the following form (Algorithm 1). (Gortler 2012)

Algorithm 1 Rasterization

```
initialize z-buffer
for all triangles
    for all pixels covered by the triangle
        compute color and z
        if z is closer than what is already in the z-buffer
            update the color and z of the pixel
```

A nice property of this algorithm is the fact that each triangle in the scene is "touched" only once (Gortler 2012).

3 The process of of taking a triangle and filling in the pixels it covers is what we call the *rasterization* step. (Gortler 2012)

Ray tracing represents an alternative to rasterization, based on ideas that are closer to the physics of light. To put it intuitively, we can think of the (basic) ray tracing algorithm as a result of *inverting* of the two **for**-loops from ALgorithm 1: (Gortler 2012)

Algorithm 2 Ray Tracing

```
for all pixels on the screen
    for all objects seen in this pixel
        if this is the closest object seen at the pixel
            compute color and z
            set the color of the pixel
```

Note that here, instead of "triangles", we're using the term "objects". In fact, in ray tracing, our scene can be made up of any geometrical primitives we want. For instance, we can render smooth spheres without having to "dice them up" into triangles (Gortler 2012). So, when we say "object", we basically mean any geometrical shape.

The objects "seen" by a pixel are found by casting a *ray* in the 3D scene, and checking what geometries are *intersected* by it. As The rays are cast from the camera, in the direction of a pixel (later, we'll understand better what this means), and that pixel is given the color of the closest object.

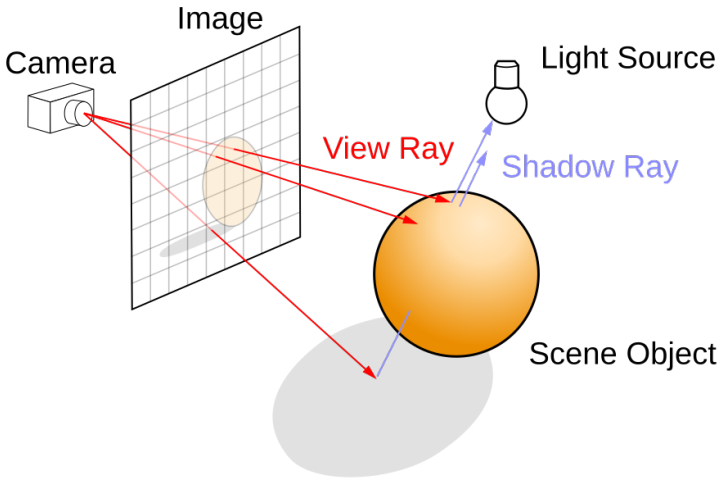

Fig. 24: Rays are cast through pixels and given the color of the intersected object. Source: Wikimedia Commons. URL: https://www.britannica.com/science/electric-field (last accessed: 19.03.2025)

In general, in ray tracing rays are cast *recursively*. For instance, once we've found the closest object hit by a ray, we can cast *another* ray, from

the hit point towards a *light source* in the scene (also called a *shadow ray*), as shown in Figure 24. This way, we can determine whether (and how much) light reaches that point, or if it's in shadow (Gortler 2012).

Going back for a moment to the physics of light, we can see that what we're doing here is following light rays *in reverse*. In the real world, light is emitted by a light source, hits an object, gets reflected, and, finally, reaches the camera. We, however, retrace this path backwards so that we're sure to be following rays that have reached the camera (we're not in interested in the rest).

One of the clear computational disadvantages of ray tracing is that, every time we cast a ray in the scene, we have to go through *all* the geometric primitives (all triangles, all spheres...) to check which ones are intersected. In other words, we conduct a *linear search* between all the geometries to find the closest one. To make this process faster, we can use *acceleration data structures*, such as *Bounding Volume Hierarchies* (*BVH*), which turn the search into a *tree traversal process*, reducing its time complexity from $O(n)$ to $O(\log n)$ (Akenine-Möller et al. 2018).

On the other hand, a great advantage of ray tracing is its ease in simulating realistic light effects such as *reflections*, *refractions*, and *smooth shadows*. This is due to the possibility of shooting rays in any direction, from any point in the scene (Akenine-Möller et al. 2018).

3.1.2 *Ray and Camera Models*

Up to this point, we've been purposely vague about the practical aspects of ray tracing. For example, we've been talking about "virtual cameras" without really knowing what they look like inside a renderer. We'll now see how ray tracers can understand the concept of a camera by expressing it in a mathematical way.

We'll be using the simplest possible model of a camera, that is, a *pinhole camera* (Gortler 2012). In a pinhole camera, we imagine having an opaque surface with a hole on it, of the size of a point. Light passes through this point and gets recorded on the camera's *film* (or *sensor*), which is just a plane behind the opaque surface. The image produced by this process is actually *mirrored*, and we later need to *flip it* to show the scene correctly. We can avoid doing this, though, by modeling the pinhole camera with the film plane *in front*, at a distance which we'll call *focal length*[4] (see

4 Following the terminology used by Peter Shirley in the book series *Ray Tracing In One Weekend*. (Shirley, Black, and Hollasch 2024)

Figure 24). Note that this wouldn't make sense in the physical world, but works just as well mathematically (Gortler 2012).

Another name we'll use to refer to the film plane is *viewport*. The viewport is divided into *pixels*, which, for us, are just equally-spaced points. Each pixel has two integer coordinates, $[x, y]^T$, with x going right in the image and y going down. The viewport has a horizontal axis $\vec{H}$ and a vertical axis $\vec{V}$ (V_u and V_v in Figure 25). Adjacent pixels are separated by a horizontal shift $\Delta\vec{h}$ or a vertical shift $\Delta\vec{v}$.

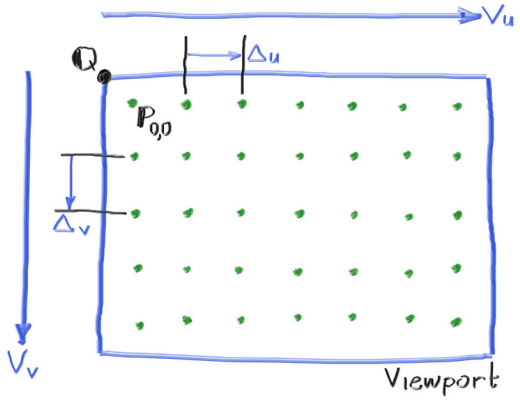

Fig. 25: "Viewport and pixel grid." (Shirley, Black, and Hollasch 2024)

The camera has a *center*, which corresponds to the *pinhole*, and is oriented along a *coordinate frame* formed by its *right*, *forward* and *up* vectors.

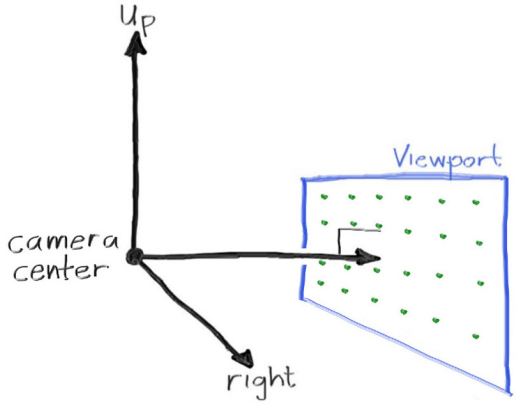

Fig. 26: "Camera geometry." (Shirley, Black, and Hollasch 2024)

To the camera, we also associate a *focal length* d, the distance of the viewport from the camera center, and a *vertical field of view* θ. These

two values, together with the aspect ratio α (width/height), are used to determine the *scale* of the viewport (Shirley, Black, and Hollasch 2024). In essence, we can say that the vertical field of view θ determines a *cone of vision*, which grows as we move farther away from the camera[5]. The viewport will constitute a "slice" of this cone, at the distance d (see Figure 27).

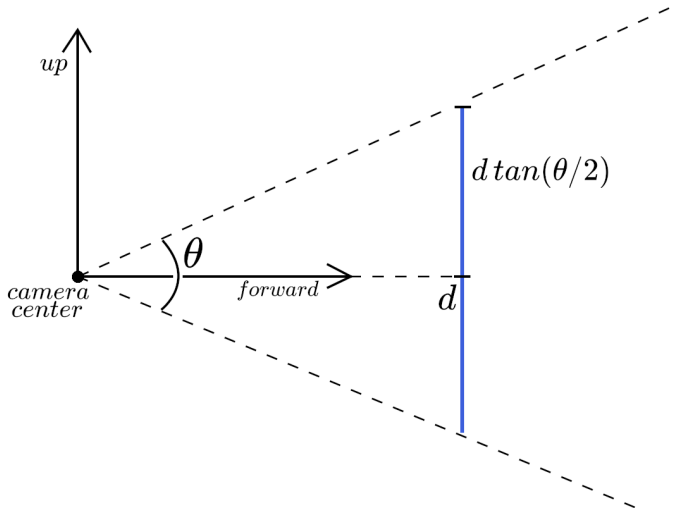

Fig. 27: Cone of vision and viewport geometry.

Thus, we can calculate the vertical size of the viewport $||\vec{V}||$ as:

$$||\vec{V}|| = 2d\,tan(\theta/2)$$

And the viewport width $||\vec{H}||$ as

$$||\vec{H}|| = \alpha||\vec{V}||$$

$\vec{V}$ and $\vec{H}$ will then be

$$\vec{V} = -||\vec{V}||\,\vec{u}$$
$$\vec{H} = ||\vec{H}||\,\vec{r}$$

Finally, $\Delta\vec{u}$ and $\Delta\vec{v}$ can be found by dividing $\vec{V}$ and $\vec{H}$ by the number of pixels, horizontal and vertical respectively.

This is all we need to describe our simple, pinhole camera. At this point, we would like to "take a picture" with it; that is, to color the viewport's

5 It's more of a "rectangular pyramid" than a "cone", actually.

pixels. To do so, we need to know what the camera *sees* along each pixel. As already mentioned, we find this out by casting a *ray* through the pixel and into the scene.

We define a ray as a function $\tilde{p}(t)$[6], which outputs a point along the ray, at a distance $t \epsilon \mathbb{R}$ from the ray origin $\tilde{o}$: (Pharr, Jakob, and Humphreys 2023)

$$\tilde{p}(t) = \tilde{o} + t\vec{D} \qquad 0 \leqslant t < \infty \tag{3.8}$$

Where $\vec{D}$ is the direction of the ray. Equation (3.8) is also called the *parametric form* of a ray.

To "shoot" the ray through a pixel with coordinates $[x, y]^T$, we pick the camera center $\tilde{c}$ as its origin, and calculate the direction $\vec{d}$ as

$$\vec{d} = (\tilde{p}_{00} + x\Delta\vec{h} + y\Delta\vec{v}) - \tilde{o}$$

Where $\tilde{p}_{00}$ is the position in 3D space of the pixel with coordinates $[0, 0]^T$.

As we know, what we do next is find the closest object intersected by the ray, and then give the pixel its color.

3.1.3 *Antialiasing*

If we send a single ray through each pixel, that is, we do *point sampling* (Shirley, Black, and Hollasch 2024), we'll quickly find that our rendered images contain *visual artifacts*, also called *aliasing artifacts*. One example for this are *jagged patterns* (or "stair steps"), which manifest at at shape edges (Figure 28).

Aliasing artifacts occur when there is too much visual complexity to fit in a single pixel (Gortler 2012). If we think about it, it's obvious why this happens. We are trying to represent on a fixed, discrete grid of points something that has (potentially) infinite resolution. so we can't expect it to look 100% right. If we zoom in on an image, sooner or later this discretization process will become obvious to the eye.

However, these artifacts can be mitigated by taking *multiple samples* around each pixel, that is, sending multiple rays, and then averaging the colors returned from those rays. In simple terms, it's as if we *blur* together multiple, smaller pixels. These types of methods are collectively known as *antialiasing* (Gortler 2012).

6 We'll be indicating points in 3D as $\tilde{p}$, with a tilde on top, to distinguish them from vectors, as done by (Gortler 2012). Also, most of the times vectors will be capitalized (e.g. $\vec{V}$) to make the formulas more compatible with their code implementations, which we'll see in Chapter 4.

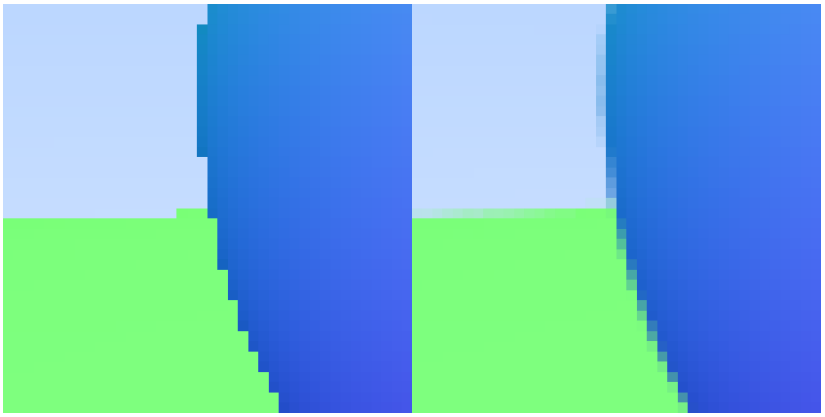

Fig. 28: "Before and after antialiasing." (Shirley, Black, and Hollasch 2024)

So, we should take multiple samples per pixel. Each of this samples corresponds to a point around the pixel, with some small offsets Δx, Δy with respect to the pixel coordinates $[x, y]^T$. The ray will be sent through the viewport's point positioned at $[x + \Delta x, t + \Delta y]^T$.

There are various ways to pick Δx and Δy. For instance, we can generate them as two random numbers, uniformly distributed in the interval $[-1, 1]$. This way, we use a sampling region which extends to the eight neighbouring pixels. We can also apply some sort of filter (like a *tent filter*), to the generated points[7]. This will make them less uniform and more distributed towards the central pixel, which is what we want, since the central pixel is more important.

3.1.4 *Ray-Object Intersections*

In ray tracing, the most fundamental computation is the intersection of a ray with an object in the scene. Our objects will be either spheres or triangles, and here we'll describe intersection methods for both.

Let's consider first the case of spheres. Ray-sphere intersection can actually be made quite simple, as we'll see, by writing the equation of the sphere in vector form (Shirley, Black, and Hollasch 2024).

A sphere is the set of points with the same distance (the sphere's *radius*) from its center. For this reason, we define a sphere by specifying

7 As done in the `smallpt` renderer. URL: https://www.kevinbeason.com/smallpt/ (last accessed: 19.03.2025)

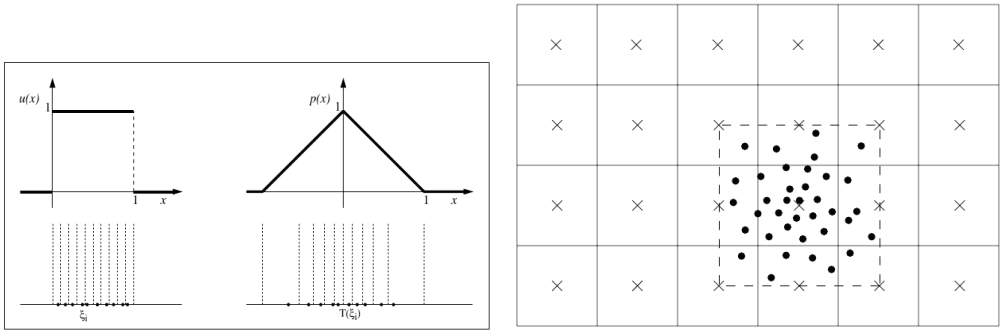

Fig. 29: A tent filter is applied to a set of uniform samples, to make the points more distributed towards the central pixel. Images from the book *Realistic Ray Tracing*, by Shirley and Morley.

$\tilde{c}$, its center, and r, its radius. If the sphere is centered in the origin, the condition we just mentioned can be expressed as

$$x^2 + y^2 + z^2 = r^2$$

For each point $\tilde{p} = [x, y, z]^T$ on the sphere. When the sphere is centered around a point $\tilde{c} = [c_x, c_y, c_z]^T$, the equation becomes:

$$(c_x - x)^2 + (c_y - y)^2 + (c_z - z)^2 = r^2$$

We rewrite this as

$$(\tilde{c} - \tilde{p}) \cdot (\tilde{c} - \tilde{p}) = r^2 \tag{3.9}$$

Remember now (from eq. (3.8)) that a ray is defined as $\tilde{p}(t) = \tilde{o} + t\vec{D}$. Saying "the ray hits the sphere" is the same as saying that there exists a point $\tilde{p}$ which satisfies both eq. (3.8) and eq. (3.9) (it's both on the ray, and on the sphere):

$$(\tilde{c} - \tilde{p}(t)) \cdot (\tilde{c} - \tilde{p}(t)) = r^2$$
$$(\tilde{c} - (\tilde{o} + t\vec{D})) \cdot (\tilde{c} - (\tilde{o} + t\vec{D})) = r^2$$

Next, we solve this for t:

$$(-t\vec{D} + (\tilde{c} - \tilde{o})) \cdot (-t\vec{D} + (\tilde{c} - \tilde{o})) = r^2$$
$$t^2\vec{D} \cdot \vec{D} - 2t\vec{D} \cdot (\tilde{c} - \tilde{o}) + (\tilde{c} - \tilde{o}) \cdot (\tilde{c} - \tilde{o}) - r^2 = 0$$

And what we get is a quadratic equation with variable t.

- If the equation has two real roots, the ray passes through the sphere. Doing so, it hits two points on the surface, one on the front and one on the back, which correspond to the two solutions for t.

- If there's only one real root, the ray is tangent to the sphere.
- If there are no real roots, the ray misses the sphere.

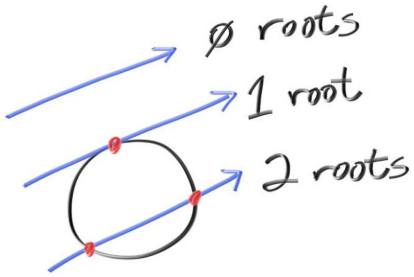

Fig. 30: "Ray-sphere intersection results." (Shirley, Black, and Hollasch 2024)

At the intersection point $\tilde{p}$, the normal $\vec{N}$ of the sphere at that point can be calculated as

$$\vec{N} = \tilde{p} - \tilde{c}$$

Let's now move on to triangles. First, we decide a standard way to calculate the triangle's normal.

Let $\tilde{v_0}, \tilde{v_1}, \tilde{v_2}$ be the three vertices of the triangle. We can calculate its normal $\vec{N}$ as a cross product between two vectors along the triangle's edges (figure ...):

$\vec{N} = \frac{(\tilde{v_1} - \tilde{v_0}) \times (\tilde{v_2} - \tilde{v_0})}{\|(\tilde{v_1} - \tilde{v_0}) \times (\tilde{v_2} - \tilde{v_0})\|}$

Where the denominator is there to normalize $\vec{N}$.

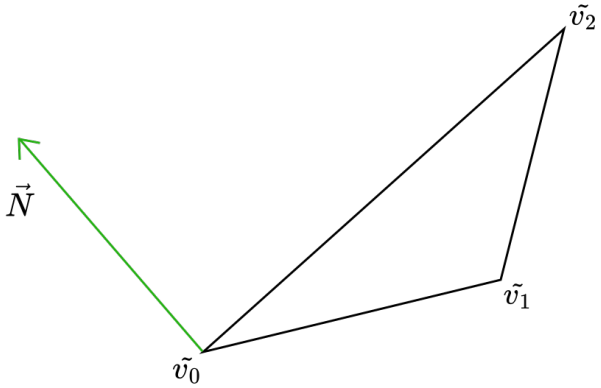

Fig. 31: Surface normal of a triangle.

A consequence of this choice is that, when looking at the triangle, if $\vec{N}$ pointing *towards us*, we're seeing the vertices in *anti-clockwise* order, while if it's pointing away from us, *clockwise*. To simplify things, we will discard ray-triangle intersections when in the second case, considering only triangles which have their normal pointed towards us[8]. *Triangle meshes* will be following this convention. In fact, they are defined as collections of *vertices* rather than triangles, the latter of which are built only at the time of *loading* the mesh into the program. When loading a triangle mesh, we'll go through the list of vertices, in their defined order, and form groups of three to build triangles. Thus, to predict the orientation of an assembled triangle, we'll just have to follow the clockwise/anticlockwise convention when defining the vertices in the mesh. That is, if we want to show a triangle on screen, we'll order its vertices in the list so that the camera sees them in clockwise order.

We can now describe ray-triangle intersection, which is usually composed of two steps (Gortler 2012). The first consists in calculating an intersection with the triangle's *plane*[9].

The equation of a plane is given by

$$ax + by + cz + d = 0$$

Where a, b, c, d are constants, and x, y, z are the coordinates of a point $\tilde{p}$ lying on the plane. A more intuitive way to express this formula is, again, its vector form (Shirley, Black, and Hollasch 2024). In fact, we can interpret a, b, c as the components of the plane's normal vector $\vec{N}$, that is, $\vec{N} = [a, b, c]^T$. Therefore, we rewrite the equation as

$$\vec{N} \cdot \tilde{p} = -d$$

We're basically calculating the projection of $\tilde{p}$ on the normal $\vec{N}$, and saying how far along $\vec{N}$ it should be. We already know $\vec{N}$, since it's also the triangle's normal.

As done with spheres, we now impose the condition $\tilde{p}(t) = \tilde{o} + t\vec{D}$, and solve for t:

$$\begin{aligned} \vec{N} \cdot (\tilde{o} + t\vec{D}) &= -d \\ \vec{N} \cdot \tilde{o} + t(\vec{N} \cdot \tilde{D}) &= -d \\ t &= -\frac{\vec{N} \cdot \tilde{o} + d}{\vec{N} \cdot \tilde{D}} \end{aligned}$$

8 This is also called *backface culling*, since we only see the "front" of the triangle. **(gortler2012)**

9 A triangle lies on exactly one plane.

Before calculating the division, we check if the denominator is zero. In that case, the ray is parallel to the plane, and there's no intersection.

So, we now use t to get the ray-plane intersection point $\tilde{p} = \tilde{p}(t) = \tilde{o} + t\vec{D}$. The next step is to check whether the point is inside the triangle.

Following our anti-clockwise convention, we want $\tilde{p}$ to be "on the left" of each of the triangle's edges[10].

Let's see what this means by taking the first edge $\tilde{v_1} - \tilde{v_0}$, and calling it $\vec{A}$. Let's also define the vector $\vec{B} = \tilde{p} - \tilde{v_0}$ and compute the cross product $\vec{C} = \vec{A} \times \vec{B}$. If $\tilde{p}$, or (to be more precise, $\vec{B}$) is on the left from $\vec{A}$, the cross product $\vec{C}$ will be pointing in the same direction as the normal $\vec{N}$. Thus, $\vec{p}$ will be on the left from the edge $\vec{A}$ if:

$$\vec{N} \cdot \vec{C} \geqslant 0$$

Where if $\vec{N} \cdot \vec{C} = 0$, $\tilde{p}$ lies along the edge.

When this holds true for each of the triangle's edges, that is, the point $\tilde{p}$ is on the left of each edge, we can say that $\tilde{p}$ is inside the triangle. Otherwise, there's no intersection.

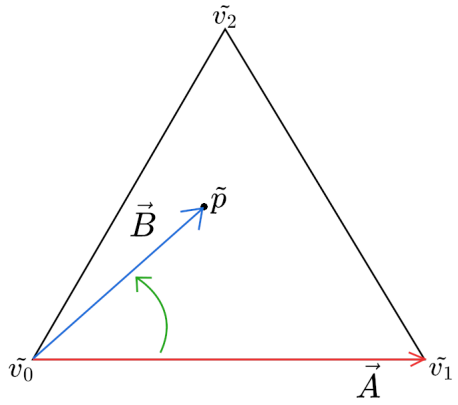

Fig. 32: $\tilde{p}$ is "to the left" from $\vec{A}$.

That's it: we can now calculate the closest intersection with an object, be it a sphere or a triangle, for any ray sent into the scene. The problem that we're left with now is to establish the color of the object's surface, which derives from reflection. As the source of the reflected light, for the moment, we use a *point light* (Gortler 2012). A point light source is

10 This is described by (Gortler 2012) as the three "sub" triangles $\Delta(\tilde{v_0}\tilde{v_1}\tilde{p})$, $\Delta(\tilde{v_0}\tilde{p}\tilde{v_2})$, $\Delta(\tilde{p}\tilde{v_1}\tilde{v_2})$ being all "clockwise".

located at a single point in the scene, and radiates light in all directions equally.

We'll see that the orientation of the *light vector* $\vec{L}$, which is directed towards the (point) light from the point of intersection, is a key factor for reflection, along with the surface normal $\vec{N}$ and the *view vector* $\vec{V}$, which points towards the viewer (that is, the camera center) (**gorler2012**). The directions of these vectors, together with the material parameters, will determine the amount and color of light reflected by the intersected surface.

3.2 RADIOMETRY

We might realize now that something is missing. In fact, we have to properly describe a way to put *light* into *numbers*. In the previous chapter, we mentioned that the "amount" of light is usually specified through *spectral distributions*, and then went on to prove that, for our purposes, we can use RGB triplets instead of full spectra. The three RGB scalar values tell us "how much" some light is red, green or blue, and are used to mix these three base colors. The issue here is that we don't have any *units* to measure light, aside from *energy*. The concept of energy is too vague for us, and can't really express the amount of light traveling along *rays*. To solve this issue we'll now go back to physics, and build a series of abstractions which will culminate in a unit of measurement called *radiance*.

We saw how ray tracing thinks of light as a ray. We recall now that we already encountered this model in Chapter 2, when talking about macroscopic descriptions of light-matter interaction (section 2.3.2). We called it the point of view of *geometrical optics*, where light interacts with objects much larger than its wavelength.

We now resort again to geometrical optics to describe quantities for the measurement of light, also called *radiometric quantities*.

Radiometry can be defined as

> "The study of the propagation of electromagnetic radiation in an environment."
> (Pharr, Jakob, and Humphreys 2023)

Geometrical optics and radiometry form the basis of many PBR algorithms, as they offer adequate models for the description of light and light scattering.

We'll now follow the process provided by (Pharr, Jakob, and Humphreys 2023) to abstractly derive some basic radiometric quantities, by taking limits over time, area, and directions. We should always keep in mind that these quantities are, in general, wavelength dependent.

3.2.1 *Energy*

We already know that the energy of light is carried by photons[11], each of which is at a particular *frequency* ν. Recall that the energy of a photon, which we'll indicate as Q here, is

$$Q = h\nu$$

Where h is Planck's constant ($h \approx 6.626 \times 10^{-34}\,m^2\,kg\,s^{-1}$).

3.2.2 *Radiant Flux*

Radiant flux, also known as *power*, is the total amount of energy passing through a *surface*, or region of space, per unit time.

$$\Phi = \lim_{\Delta t \to 0} \frac{\Delta Q}{\Delta t} = \frac{dQ}{dt}$$

It is measured in watts (W).

So, when talking about flux, we have an area, crossed by photons during infinitesimally small amounts of time. Note that, because photons are discrete quanta, it doesn't really makes sense to take limits that go to zero for differential time. However, if we assume the number of photons to be enormous with respect to the measurements we are interested in, this detail is not problematic.

3.2.3 *Irradiance and Radiant Exitance*

Let's say that we have photons passing through a finite area A. We now define the *average* density of power over this area as

$$E = \Phi/A \qquad [W/m^2]$$

and call it *irradiance* if the photons are *arriving* at the surface, and *radiant exitance* (indicated as M) if they're *leaving* it.

11 Basically, all the radiometric quantities that we'll see are just different ways of measuring photons.

To be more precise, we should, again, take a limit. This time, it's over the differential area at a point $\tilde{p}$ on A:

$$E(\tilde{p}) = \lim_{\Delta A \to 0} \frac{\Delta \Phi(\tilde{p})}{\Delta A} = \frac{d\Phi(\tilde{p})}{dA}$$

This gives us a proper definition of irradiance/radiant exitance.

We can use irradiance to understand *Lambert's law*:

> "The amount of light energy arriving at a surface is proportional to the cosine of the angle between the light direction and the surface normal."
> (Pharr, Jakob, and Humphreys 2023)

We do so as follows. Consider a light source with area A and a flux Φ, illuminating a surface. If the light is directly on top of the surface (see Figure 33), the area of the surface A_1 which receives light will be equal to A, and the irradiance

$$E_1 = \frac{\Phi}{A}$$

If, however, the light is at an angle to the surface, the area on the surface which receives light will actually be *larger* than A. If A is small, the area receiving flux, which we call A_2 now, will be approximately $A/\cos\theta$, and the irradiance

$$E_2 = \frac{\Phi \cos\theta}{A}$$

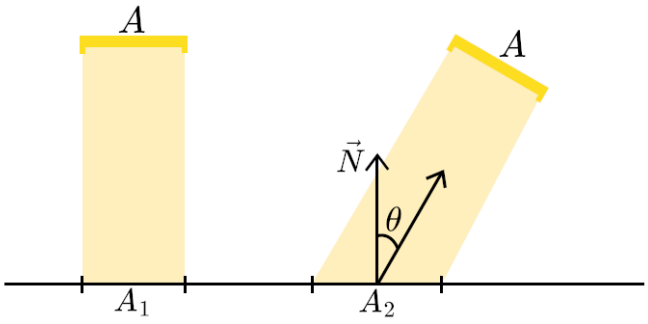

Fig. 33: Visualization of Lambert's law.

3.2.4 *Radiance*

Radiance measures irradiance with respect to *solid angles*. It is defined as

$$L(\tilde{p}, \omega) = \lim_{\Delta\omega \to 0} \frac{\Delta E_\omega(\tilde{p})}{\Delta\omega} = \frac{dE_\omega(\tilde{p})}{d\omega} \qquad [\frac{W}{m^2\,sr}]$$

where E_ω denotes irradiance at the surface when it's perpendicular to the direction ω. As we've seen, we can calculate E_ω by simply dividing for a $\cos\theta$ factor, thanks to Lambert's law.

Basically, radiance measures light incident to a surface point coming in from an infinitesimal cone of directions $d\omega$, which corresponds, approximately, to a single direction $\vec{\omega}$ (see Figure 34).

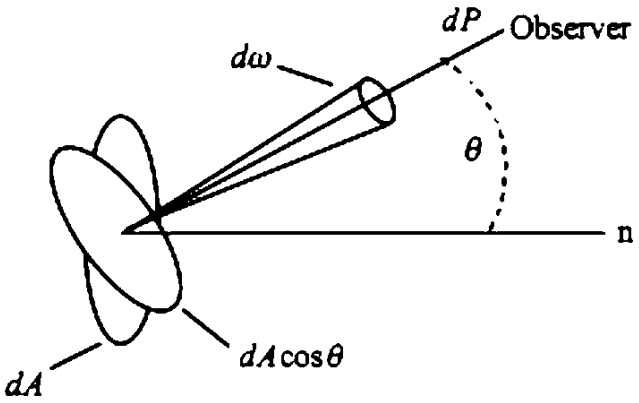

Fig. 1.

Fig. 34: The incident radiance arrives to the surface point from an infinitesimal cone of directions. Source: LibreTexts physics, Planetary Photometry (Tatum and Fairbairn). (license URL: https://creativecommons.org/licenses/by-nc/4.0/)

Between the radiometric quantities we've seen, radiance is the most fundamental. Knowing the radiance, we can calculate the irradiance E, the radiant flux Φ and the energy Q by taking *integrals*, instead of *limits*.

It can be proven that radiance remains *constant* along *rays* through empty space. This clearly makes it a perfect quantity for ray tracing.

Given a point $\tilde{p}$ on a surface, we'll use radiance both to define *incoming* and *outgoing* (that is, *reflected*) *light* at $\tilde{p}$. To express this distinction,

we'll write the *incoming radiance* as $L_i(\tilde{p}, \vec{\omega})$, and the *outgoing radiance* as $L_o(\tilde{p}, \vec{\omega})$.

3.3 BRDF

We've finally arrived at a point where we can actually discuss models for the reflection of light, called *bidirectional reflectance distribution functions* (*BRDF*). We'll start by deriving the definition of a generic BRDF from radiometric quantities.

3.3.1 *Definition*

Let's reformulate the problem of reflection, now in more precise terms. We want to compute the amount of outgoing radiance L_o, leaving the point $\tilde{p}$ of a surface in the direction $\vec{\omega_o}$ toward the viewer, as a result of some incident radiance L_i coming in from the direction $\vec{\omega_i}$ (see Figure 35).

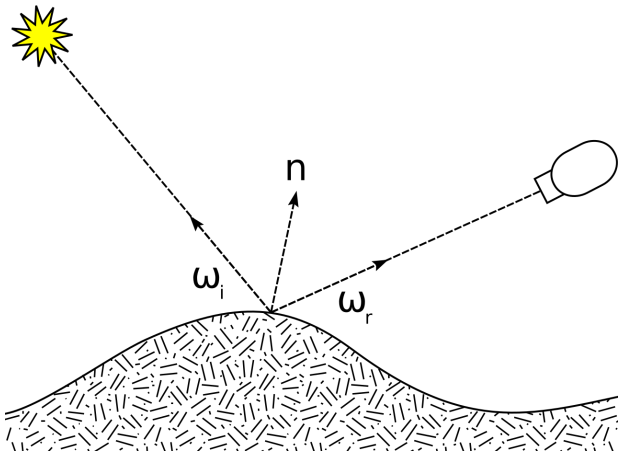

Fig. 35: Diagram showing vectors for the BRDF (ω_r is the outgoing direction). Source: Wikimedia Commons (license URL: https://creativecommons.org/licenses/by-sa/3.0/deed.en)

We could do this easily if we had a function, call it f_r, which defines the ratio between L_o and L_i. Actually, for reasons that will become clear later, it would be better if L_o were expressed in differential terms, that is, dL_o.

Now, remember that $\vec{\omega_i}$ and $\vec{\omega_o}$ can be interpreted as *differential* (that is, infinitely small) *cones of directions* $d\omega_i$, $d\omega_o$. If we do so with $\vec{\omega_i}$, we can calculate the differential irradiance coming in on $\tilde{p}$ as (Pharr, Jakob, and Humphreys 2023)

$$dE(\tilde{p}, \vec{\omega_i}) = L_i(\tilde{p}, \vec{\omega_i}) cos\theta_i d\omega_i$$

Where θ_i is the angle between the incident direction and the surface normal, also called *angle of incidence*.

This incoming irradiance induces some differential amount of reflected radiance in the direction $\vec{\omega_o}$, proportional to the irradiance:

$$dL_o(\tilde{p}, \vec{\omega_o}) \propto dE(\tilde{p}, \vec{\omega_i}) \tag{3.10}$$

The reasons their values are proportional lies in the *linearity assumption*. In fact, in geometrical optics, we assume the following fact to be true

> "The combined effect of two inputs to an optical system is always equal to the sum of the effects of each of the inputs individually." (Pharr, Jakob, and Humphreys 2023)

This is a reasonable assumption. In fact, for most real materials, the following observation can be made. If all of the light is coming in from a single small cone of directions, and we double the width of this incoming cone, the amount of flux reflected along a fixed outgoing cone, and thus the amount of radiance reflected along a fixed outgoing direction, will roughly double as well (Gortler 2012). In our case, the size of the incoming cone is given by $d\omega_i$.

The BRDF function f_r is then defined as the constant of proportionality in eq. (3.10): (Pharr, Jakob, and Humphreys 2023)

$$f_r(\tilde{p}, \vec{\omega_o}, \vec{\omega_i}) = \frac{dL_o(\tilde{p}, \vec{\omega_o})}{dE(\tilde{p}, \vec{\omega_i})} = \frac{dL_o(\tilde{p}, \vec{\omega_o})}{L_i(\tilde{p}, \vec{\omega_i}) cos\theta_i d\omega_i} \tag{3.11}$$

To describe how the reflection of light should take place on surfaces, we'll be defining BRDF functions. We'll then base our calculations, other than on the parameters $\tilde{p}, \vec{\omega_o}, \vec{\omega_i}$, also on properties of surfaces, which will effectively "tweak" the behaviour of the BRDF.

We should mention that, in PBR, BRDFs can't just be defined arbitrarily. To make a BRDF physically based, we should make sure that it obeys the following principles (Pharr, Jakob, and Humphreys 2023). - *Helmholtz reciprocity*: the incoming and outgoing directions can be swapped without affecting the value of f_r. That is, for all pairs of directions $\vec{\omega_i}$ and $\vec{\omega_o}$, $f_r(\tilde{p}, \vec{\omega_i}, \vec{\omega_o}) = f_r(\tilde{p}, \vec{\omega_o}, \vec{\omega_i})$. - *Energy conservation*: the surface can't reflect more energy than it receives.

3.3.2 *The Reflection Equation*

Even though we have the BRDF f_r, we still haven't explicitly said how it should be used. Let's rewrite eq. (3.11) as

$$dL_o(\tilde{p}, \vec{\omega_o}) = f_r(\tilde{p}, \vec{\omega_o}, \vec{\omega_i}) L_i(\tilde{p}, \vec{\omega_i}) \cos\theta_i d\omega_i$$

If we integrate this equation over all directions above $\tilde{p}$, we can now actually calculate L_o:

$$L_o(\tilde{p}, \vec{\omega_o}) = \int_{H^2(\vec{N})} f_r(\tilde{p}, \vec{\omega_o}, \vec{\omega_i}) L_i(\tilde{p}, \vec{\omega_i}) \cos\theta_i d\omega_i \tag{3.12}$$

Where $H^2(\vec{N})$ is the hemisphere centered around the normal at $\tilde{p}$, assuming that the normal is pointing outwards from the surface. This is called the *reflection equation*[12], and, during rendering, our objective will be to solve it for each point $\tilde{p}$ seen by the camera. For now, we'll only present the solution for the case of a single point light, which is very straightforward.

Let's summarize briefly the situation. In order to determine the color of a pixel, a ray has been sent out in the scene, which intersected an object at the point $\tilde{p}$. The pixel will be colored based on the reflected radiance L_o from $\tilde{p}$, in the direction $\vec{V}$ towards the camera. Remember that radiance, like any other light-measuring quantity, is spectral (it varies with wavelength). Also remember that we can express it as an RGB triplet. To calculate the reflected light, that is, $L_o(\tilde{p}, \vec{V})$, we multiply the BRDF f_r for all the incoming radiance. The BRDF is a spectral quantity as well, so its value is also just an RGB triplet. As for the incoming radiance, there's only one source where it could come from: the point light. The point light is in the direction $\vec{L}$ with respect to $\tilde{p}$, so we compute:

$$L_o(\tilde{p}, \vec{V}) = f_r(\tilde{p}, \vec{V}, \vec{L}) L_i(\tilde{p}, \vec{L}) \cos\theta_i$$

And we've calculated the color $L_o(\tilde{p}, \vec{V})$ of the pixel.

Note that $\cos\theta_i$ can be calculated as the dot product $\vec{N} \cdot \vec{L}$. To avoid negative values, we clamp it to 0. This way, we only consider light if it comes from above the surface, and ignore it otherwise.

12 The reflection equation is a special case of the *rendering equation*.(Hoffman 2012)

3.3.3 *Diffuse and Specular Terms*

In Chapter 2, we learnt about the behaviour of light when it encounters a planar boundary. We found out that, as explained by the *Fresnel equations*, light gets split into the two directions of *reflection* and *refraction*. We also defined *subsurface scattering* as the phenomenon where refracted light gets scattered so much that it's re-emitted out of the surface.

Even though subsurface scattering is different from reflection, a BRDF model can partly simulate it. In fact, BRDFs are usually divided into two separate parts, called *terms* (or *lobes*). One of them describes proper reflection and is called the *specular term*, while the other, the *diffuse term*, partially models subsurface scattering. We say "partially" because it only accounts for subsurface-scattered light that is re-emitted from (approximately) the same point where it entered (Hoffman 2012). An example is shown in Figure 36, where the BRDF terms are based on the *Phong lighting model*.

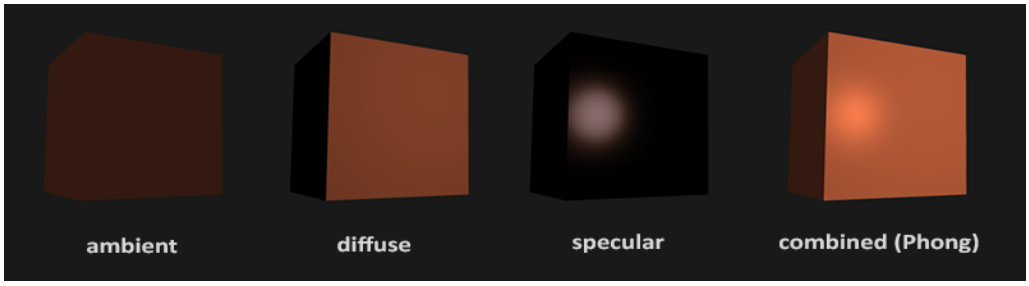

Fig. 36: The Phong lighting model. Other than diffuse and specular lighting, constant *ambient lighting* is also added to model reflection. Image courtesy of Joey de Vries, at `https://learnopengl.com/Lighting/Basic-Lighting`. (license: `https://creativecommons.org/licenses/by/4.0/`)

If we need to simulate proper subsurface light transport, meaning light exiting the surface also at points that are relatively distant from its entry, we should use a *bidirectional scattering surface reflectance distribution function* (*BSSRDF*) instead of a BRDF (Pharr, Jakob, and Humphreys 2023).

The *diffuse term* of a BRDF is usually quite simple. A common example is the *Lambertian model*, where the BRDF is just a constant value: (Hoffman 2012)

$$f_{\text{Lambert}}(\vec{L}, \vec{V}) = \frac{\rho}{\pi} \tag{3.13}$$

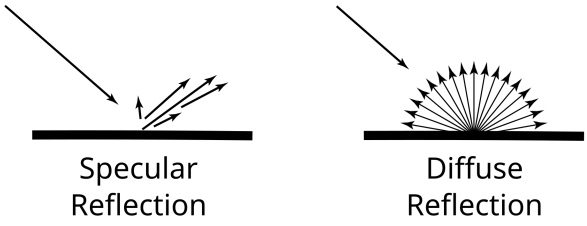

Fig. 37: Geometry of specular and diffuse reflection. Source: Wikimedia Commons. (license URL: https://creativecommons.org/licenses/by/3.0/deed.en)

Here ρ is an RGB triplet indicating the fraction of light which is diffusely reflected, and is typically referred to as the *diffuse color*. In fact, during the scattering process, light is also partially absorbed by the material, depending on wavelength. Thus, only some wavelengths will be re-emitted, and the diffuse response will be tinted towards a certain color (Burley 2012). The diffuse color ρ corresponds to what typically people think of as the *surface color* (Hoffman 2012).

What equation (3.13) means is that the outgoing radiance will be constant for all outgoing directions, that is, it's independent from $\vec{V}$ (see Figure 37). Incoming directions will influence reflection only through the $\cos\theta_i$ term in equation (3.12). Note that in (3.13) we've omitted the parameter $\tilde{p}$ from the BRDF and made it implicit, as we'll be doing from now on.

The *specular term* of a BRDF is, usually, more complex, as it's often based on *microfacet theory*.

3.4 MICROFACET THEORY

Microfacet theory allows us to generalize the idea of reflection to non-optically flat surfaces. We know from Chapter 2 that most surfaces in the real world present irregularities which are larger than visible light wavelengths, but also too small to be seen by the eye (section 2.3.3). Microfacet theory takes this into account by modeling a surface as a collection of many *microfacets*, each of which is optically flat and thus represents a perfect reflecting mirror (Hoffman 2012). Most physically plausible models, even if they're not specifically described in microfacet form, can still be interpreted through the point of view of microfacet theory (Burley 2012).

So, the specular term of a BRDF uses the idea of microfacets. Each microfacet reflects light the way we know, in a single outgoing direction depending on the orientation of the microfacet's normal, which we'll call $\vec{M}$. Since, given a light direction $\vec{L}$, we evaluate the BRDF in a single view direction $\vec{V}$, only the microfacets that happen to be angled just right between $\vec{L}$ and $\vec{V}$ can contribute to reflectance. More precisely, their surface normal $\vec{M}$ needs to be oriented exactly halfway between $\vec{L}$ and $\vec{V}$. The vector halfway between $\vec{L}$ and $\vec{V}$ actually has a specific name: the *half-vector*. It will be denoted as $\vec{H}$ (Hoffman 2012). We can calculate the half vector as (Burley 2012)

$$\vec{H} = \frac{\vec{L} + \vec{V}}{|\vec{L} + \vec{V}|}$$

Even in the case of microfacets with $\vec{M} = \vec{H}$, it's not certain that they will contribute to reflection. This is because some microfacets might be blocked by others from the direction $\vec{L}$ (*shadowing*), or from the direction $\vec{V}$ (*masking*), or from both (see Figure 38). This light is assumed to be lost.

The specular BRDF can therefore be derived in the following form (Hoffman 2012)

$$f_{\mu facet} = \frac{F(\vec{L}, \vec{H})G(\vec{L}, \vec{V}, \vec{H})D(\vec{H})}{4(\vec{N} \cdot \vec{L})(\vec{N} \cdot \vec{V})}$$

Where

- $D(\vec{H})$ is the microfacet *normal distribution function* evaluated at the half-vector $\vec{H}$. It expresses the concentration of microfacets which are oriented so that they *could* reflect light from $\vec{L}$ into $\vec{V}$.
- $G(\vec{L}, \vec{V}, \vec{H})$ is the *shadow-masking function*. It tells us the percentage of microfacets with $\vec{M} = \vec{H}$ that are not *shadowed* or *masked*. Its product with $D(\vec{H})$ gives, as a result, the concentration of *active microfacets*, which contribute to reflection.
- $F(\vec{L}, \vec{H})$ is the *Fresnel reflectance* of the active microfacets. In other terms, it's the amount of incoming light is reflected from each of the microfacets that are active.
- The denominator $4(\vec{N} \cdot \vec{L})(\vec{N} \cdot \vec{V})$ is a correction factor, which is there because of quantities being transformed between the local space of the microfacets and that of the overall macrosurface. To make sure we don't divide by zero, other than clamping the dot products to avoid negative results, we also add a very small positive value to the denominator.

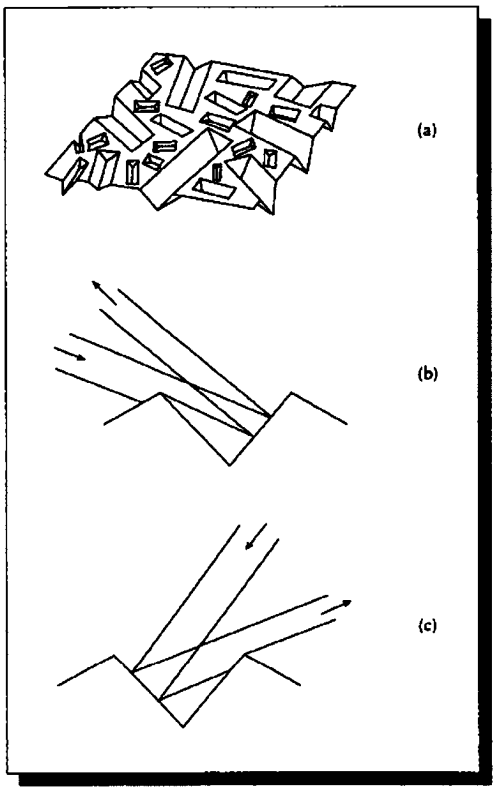

Fig. 38: (a) A surface's microfacets. (b) Shadowing. (c) Masking. (Glassner 1995)

We know present a more thorough explanation of these terms, based on Naty Hoffman's course notes *Physics and Math of Shading*, made for the 2012 SIGGRAPH course *Practical Physically Based Shading in Film and Game Production* (Hoffman 2012).

3.4.1 *Fresnel Reflectance*

Let's repeat the meaning and purpose of the Fresnel equations, one last time. The Fresnel equations tell us the portions of reflected and refracted light at a planar boundary (that is, an optically flat surface) based on the incoming angle of light and the refractive index of the material. In our case, the incoming angle is between $\vec{L}$ and $\vec{H}$, since we are only concerned with active microfacets (that is, with $\vec{M} = \vec{H}$). The Fresnel reflectance function $F(\vec{L}, \vec{H})$ is based on the Fresnel equations, and computes the amount of light that's reflected.

Since the refractive index may vary over the visible spectrum, the Fresnel reflectance should be defined as a spectral quantity (so, for us, an RGB triplet). However, sometimes the spectral nature of reflectance is ignored, and a single real number is used instead. This is done, for example, in the Disney "principled" BRDF (Burley 2012). Fresnel reflectance has to lie in the $[0, 1]$ range, since a surface cannot reflect less than 0% or more than 100% of the incoming light.

Specifying a refractive index is often inconvenient, especially for artists. For this reason, it's customary to simplify the Fresnel equations into a more intuitive form. This can be done thanks to some common patterns observed in the behaviours of real-world materials (see Figure 39).

First of all, for many materials, the reflectance is almost constant for incoming angles between $0°$ and (about) $45°$. The reflectance then goes on to change more significantly (typically increasing) between $45°$ and about $75°$. Finally, between $75°$ and $90°$, reflectance always goes rapidly to 1 ($[1, 1, 1]^T$ if viewed as an RGB triplet), making surfaces perfectly reflective at grazing angles.

So, since the Fresnel reflectance stays close to its value at $0°$ over most of the range, we can think of this value as being *characteristic* for the *material*. We denote it as F_0, and, in this thesis, we refer to it as the *base reflectance*.

A widely used approximation of Fresnel reflectance computes it as a function of the base reflectance F_0 instead of the refractive index:

$$F_{Schlick}(F_0, \vec{L}, \vec{H}) = F_0 + (1 - F_0)(1 - (\vec{L} \cdot \vec{H}))^5 \tag{3.14}$$

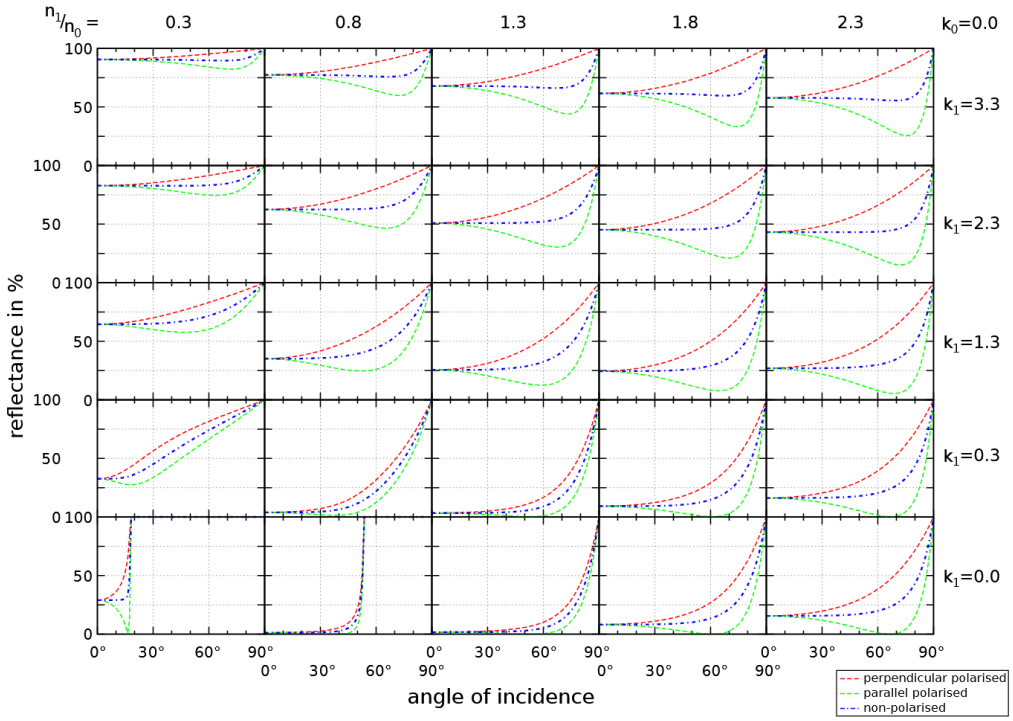

Fig. 39: "Fresnel reflection coefficients for a boundary surface between air and a variable material in dependence of the complex refractive index and the angle of incidence." Source: Wikimedia Commons. (license URL: https://creativecommons.org/licenses/by-sa/3.0/deed.en)

This called *Schlick's approximation*.

Basically, in (3.14), we're interpolating between the base reflectance F_0 and the maximum reflectance 1, following the smooth function $(1 - (\vec{L} \cdot \vec{H}))^5$, which goes from 0 to 1 as $\vec{L}$ becomes more perpendicular to $\vec{H}$ (and thus, the light is more parallel to the surface).

We've just encountered the first parameter, F_0, which will be used to define a material. To model realistic materials, the values assigned to F_0 should be reasonable. For instance, metals have much higher values of F_0 than dielectrics. In fact, we know that metals have no sub-surface scattering, so most of the incident light will be re-emitted as the specular component of the BRDF. Aside from metals (and also gemstones and crystals) pretty much any material that we see outside of laboratories has between 2% and 5% base reflectance.

3.4.2 *Normal Distribution Function*

The statistical distribution of microfacet orientations is defined via the microfacet normal distribution function $D(\vec{H})$. Most microfacets usually point in the same direction as the surface normal $\vec{N}$, resulting in a peak in value at $D(\vec{N})$. Choosing a distribution function for $D(\vec{H})$, we determine the size, brightness, and shape of specular highlights on the surface. Many times, the distribution functions are somewhat Gaussian-like with some kind of *roughness* (or *variance*) parameter, resulting in circular highlights of various sizes. However, anisotropic functions are also used.

In general, the value of $D(\vec{H})$ is not restricted to lie between 0 and 1, although its values must be non-negative.

3.4.3 *Shadow-Masking Function*

The *shadow-masking function* $G(\vec{L}, \vec{V}, \vec{H})$ represents the probability that microfacets with normal $\vec{H}$ will be visible from both the light direction $\vec{L}$ and the view direction $\vec{V}$. It's also called the *geometric attenuation term*(as done by Burley 2012). Since $G()$ represents a probability, its values are scalar and constrained to lie between 0 and 1. This function typically does not introduce any new parameters to the BRDF; it either has no parameters, or uses the roughness parameter(s) of $D()$.

The shadowing-masking function is essential for BRDF energy conservation; without it, the BRDF could reflect arbitrarily more light energy than it receives.

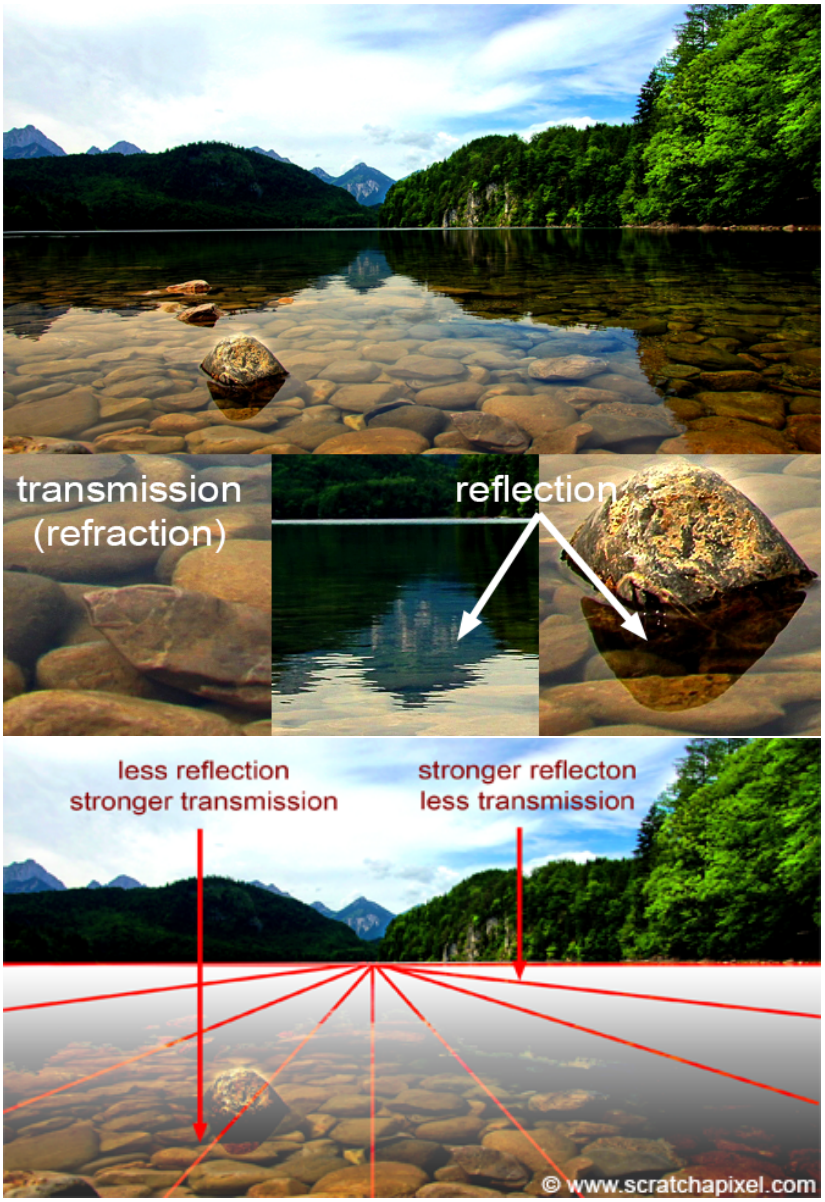

Fig. 40: "The ratio of reflected light increases as the angle between the view direction and the surface normal increases." Source: *Reflection, Refraction and Fresnel*. Scratchapixel, Introduction to Shading. (license URL: https://creativecommons.org/licenses/by-nc-nd/4.0/)

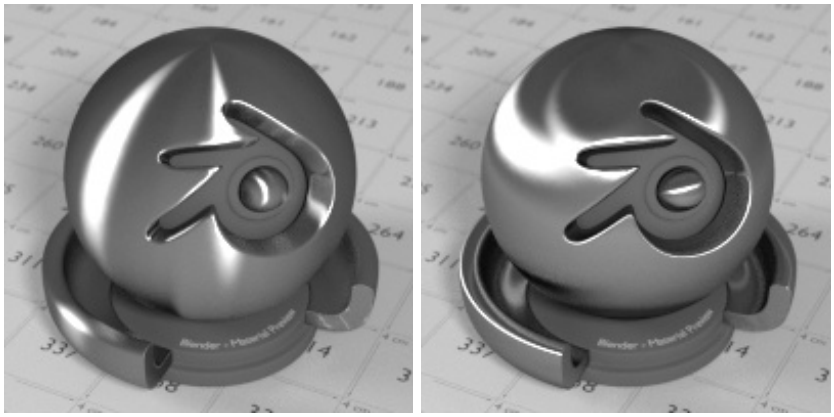

Fig. 41: Anisotropic specular highlights at 0° (left) and 90° (right). Source: *Anisotropic BSDF*. Blender 3.1 Reference Manual. (license URL: https://creativecommons.org/licenses/by-sa/4.0/)

3.5 THE DISNEY "PRINCIPLED" BRDF

We'll now see the math behind the Disney "principled" BRDF, as was described by Burley in 2012 (Burley 2012). This physically based BRDF was developed at Disney Animation Studios, and was first deployed for the production of the movie *Wreck-it Ralph*[13]. The development process of Burley and his team consisted in comparing analytical BRDFs with real-world light reflection data, previously measured by the Mitsubishi Electric Research Laboratories (MERL) and collected in the *MERL BRDF Database*[14].

The output of the Disney BRDF is the result of blending between different reflection modes. Other than the *diffuse* and *specular* lobes, there's also a *clearcoat* lobe. The clearcoat lobe can be used to form an additional layer of reflection on top of the base material, giving it a "glossy" look.

3.5.1 *Parameters*

The Disney BRDF is actually *not* physically correct; that is, not completely. It is a "principled" model, rather than a strictly physical one, meaning that it's physically *plausible*, but not *exact*. This is because it was essential

13 In *Wreck-it Ralph*, the Disney "principled" BRDF was used for virtually any material other than hair (which is a bit more tricky to render).(Burley 2012)

14 The *MERL BRDF Database* can be found at https://www.merl.com/research/downloads/BRDF/ (last accessed: 20.03.2025)

for the model to be "art directable", even at the cost of sacrificing some physical rules (Burley 2012). After all, in animation, reflection models are meant to be tools for artists, so they must allow proper creative expression.

During development of the BRDF, one of the requirements was to keep the number of parameters as low as possible. This resulted in eleven of them, the first one being a vector that defines the surface color, while the others are scalars that go from 0 to 1 (in their "plausible" range). They are described as follows.

- *baseColor*: this is simply the surface color.
- *subsurface*: when different from zero, the BRDF uses an approximation for subsurface scattering to control the shape of the diffuse reflection, making it less uniform.
- *metallic*: this is the "metallic-ness" of the material (0 = dielectric, 1 = metallic). It blends linearly blend between the two reflection models, where the metallic model has no diffuse component and its reflections are tinted towards the base color, and the dielectric model has a diffuse component and achromatic reflections.
- *specular*: the incident specular amount. It corresponds to what we earlier called the *base reflectance* F_0, and is used in lieu of an explicit index of refraction.
- *specularTint*: this is a concession for artistic control that tints the incident specular towards the base color. The "grazing specular", that is, F_{90}, is still achromatic.
- *roughness*: the surface roughness, which influences both diffuse and specular response.
- *anisotropic*: the degree of anisotropy for specular highlights. (0 = isotropic, 1 = maximally anisotropic.)
- *sheen*: this is an additional grazing component, primarily intended for cloth.
- *sheenTint*: it's the amount of tint in the sheen towards the base color.
- *clearcoat*: this activates the clearcoat lobe and regulates its strength.

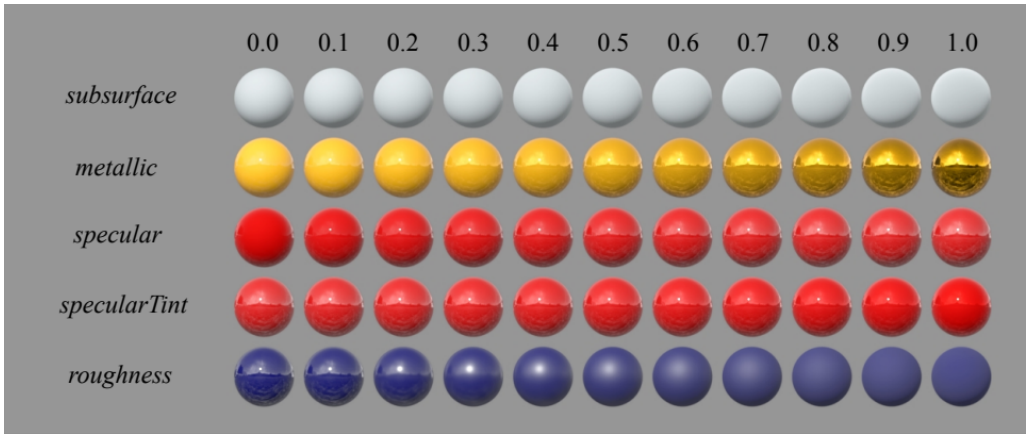

Fig. 42: "Examples of the effect of our BRDF parameters. Each parameter is varied across the row from zero to one with the other parameters held constant." (1) (Burley 2012)

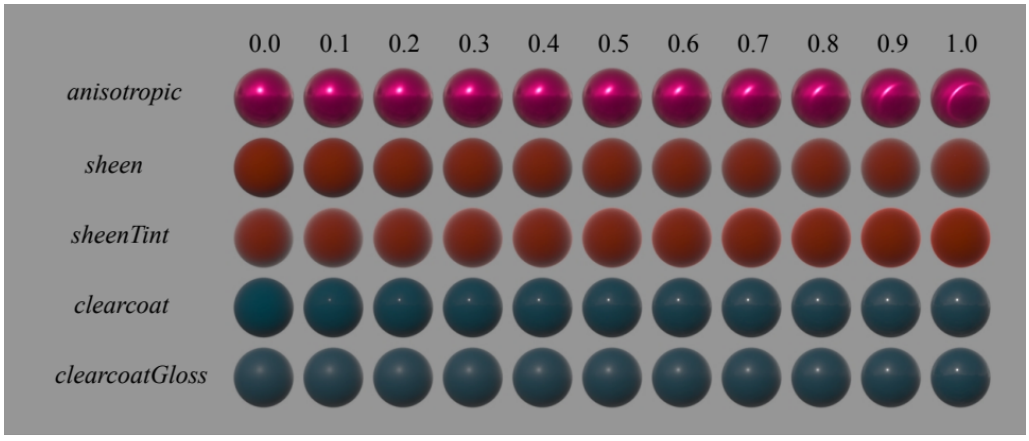

Fig. 43: "Examples of the effect of our BRDF parameters. Each parameter is varied across the row from zero to one with the other parameters held constant." (2) (Burley 2012)

- *clearcoatGloss*: controls the clearcoat's glossiness (0 gives a "satin" appearance, while 1 a "gloss" appearance).

Let's now analyze why these parameters are needed and how they are used.

3.5.2 *Diffuse Term*

Here and in the next sub-sections, we'll express the BRDF function (3.11) as

$$f(\vec{L}, \vec{V}) = \texttt{diffuse} + \frac{F(\theta_D)G(\theta_L, \theta_V)D(\theta_H)}{4\cos\theta_L \cos\theta_V}$$

Where we need to decide on a $\texttt{diffuse}$ term. θ_L and θ_V are the angles formed by the $\vec{L}$ and $\vec{V}$ vectors with the surface normal, θ_H is the angle between the normal and the half vector $\vec{H}$, and θ_D is the "difference" angle between $\vec{L}$ and the half vector (or, symmetrically, between $\vec{V}$ and $\vec{H}$).

Earlier, we introduced the *Lambertian diffuse model* (see section 3.3.3). This model assumes that refracted light gets scattered enough inside a surface to lose all its directionality when being re-emitted. For this reason, light at a surface point is re-emitted in all directions uniformly.

The Lambertian model is widely used in rendering, as it is a good approximation for various types of surfaces. However, very few materials exhibit *true* Lambertian response. It can be observed that many materials show a drop in *grazing retroreflection* (that is, retroreflection at grazing angles), while many others show a peak (Burley 2012). *Retroreflection* is a type of reflection phenomena where the material scatters light primarily back along the incident direction (Pharr, Jakob, and Humphreys 2023).

Note in Figure 44 that, at *edges*, *smooth surfaces* tend to be in *shadow*, while *rough surfaces* show *reflection peaks*. Grazing shadows for smooth surfaces are actually a result of the Fresnel equations, since, at grazing angles, all light is reflected and none is reserved to diffuse reflection (making the diffuse contribution zero). On the other hand, the reflective peaks in the edges of rough surfaces aren't usually considered by diffuse models.

This is why at Disney it was decided to develop a new, empirically based model for diffuse retroreflection. The model transitions from the classic diffuse Fresnel shadow for smooth surfaces into an added highlight for rough surfaces, as the *roughness* parameter grows in value.

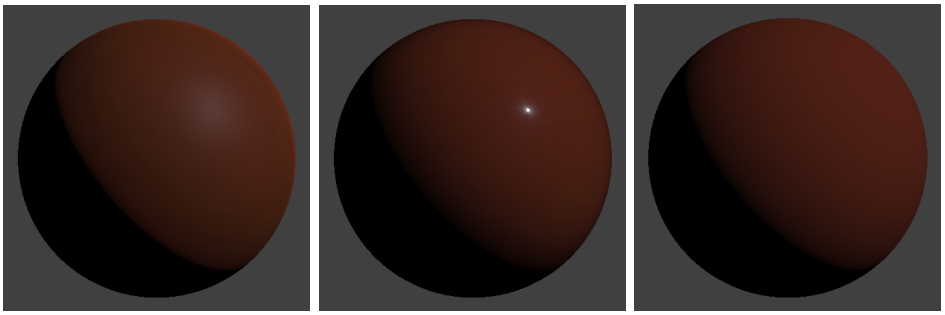

Fig. 44: "Point light response of *red-plastic*, *specular-red-plastic*, and *Lambert diffuse*. (Burley 2012)

The base Disney diffuse model is defined as

$$f_d = \frac{baseColor}{\pi}(1 + (F_{D90} - 1)(1 - \cos\theta_L)^5)(1 + (F_{D90} - 1)(1 - \cos\theta_V)^5) \tag{3.15}$$

Where

$$F_{D90} = 0.5 + 2roughness\ \cos^2\theta_D \tag{3.16}$$

In (3.16), we can see that a Fresnel factor multiplies the base Lambertian model. There is not much information about this formula, other than the fact that it was derived from the Fresnel factor(Where refraction due to the Fresnel law is considered twice, once on the way in and once on the way out of the surface.) $(1 - F(\theta_L))(1 - F(\theta_D))$ using Schlick's approximation (3.14).

In (3.16) the grazing retroreflection response is modified to go to a specific value determined from roughness, rather than zero. According to Burley,

> "This produces a diffuse Fresnel shadow that reduces the incident diffuse reflectance by 0.5 at grazing angles for smooth surfaces and increases the response by up to 2.5 for rough surfaces."
> (Burley 2012)

The base Disney diffuse (3.15) is blended together with another model, which is a rough approximation of subsurface scattering at small scattering path lengths, inspired by the *Hanrahan-Krueger subsurface BRDF*.

3.5.3 *Specular D*

In the Disney BRDF, the normal distribution function $D()$ follows the *Generalized-Trowbridge-Reitz distribution* (*GTR*):

$$D_{GTR}(\theta_H) = c/(\alpha^2 \cos^2\theta_H + \sin^2\theta_H)^\gamma$$

Where c is a "scaling constant" and α is a *roughness* parameter, derived as $\alpha = \text{roughness}^2$. This squaring operation was found to produce a more linear change in *perceived* roughness. The GTR model is a generalization of the *Trowbridge-Reitz distribution* (*TR*), where $\gamma = 2$. This is also equivalent to the so-called *GGX distribution* (Burley 2012).

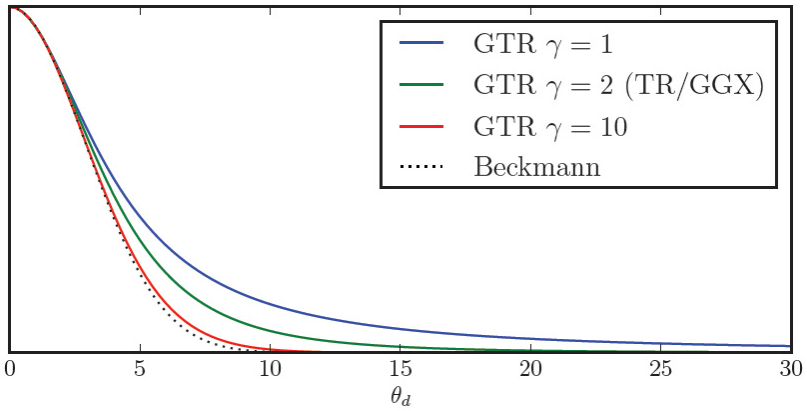

Fig. 45: "GTR distribution curves vs θ_h for various γ values." (Burley 2012)

Both the primary specular and the clearcoat lobe use GTR, with exponents $\gamma = 2$ and $\gamma = 1$ respectively. Here are the complete function definitions.

$$D_{GTR_1}(\theta_H) = \frac{\alpha^2 - 1}{\pi \log \alpha^2} \frac{1}{(1 + (\alpha^2 - 1)\cos^2\theta_H)}$$

$$D_{GTR_2}(\theta_H) = \frac{\alpha^2}{\pi} \frac{1}{(1 + (\alpha^2 - 1)\cos^2\theta_H)}$$

We can also derive *anisotropic forms*. In the case of D_{GTR_2}, this is

$$D_{GTR_{2aniso}}(\theta_H) = \frac{1}{\pi} \frac{1}{\alpha_x \alpha_y} \frac{1}{(\sin^2\theta_H(\cos^2\phi_H/\alpha_x^2 + \sin^2\phi_H/\alpha_y^2) + \cos^2\theta_H)^2}$$

Where $\phi \epsilon [0, 2\pi]$ is the *azimuthal angle*[15], and the two roughness parameters α_x and α_y are calculated from α as follows.

$$\begin{aligned} aspect &= \sqrt{1 - 0.9 anisotropic} \\ \alpha_x &= roughness^2 / aspect \\ \alpha_y &= roughness^2 \cdot aspect \end{aligned}$$

The 0.9 factor is used to limit the aspect ratio to $10 : 1$.

The primary lobe may be anisotropic, while the secondary lobe is always isotropic.

The Disney BRDF doesn't describe a material with an explicit index of refraction, and instead uses the base reflectance F_0 (defined in section 3.4.1), which corresponds to the *specular* parameter. This parameter is remapped linearly to the range $[0.0, 0.08]$, which corresponds to values in the range $[1.0, 1.8]$ for the IOR's real part. Most common materials can be found in this range. However, the *specular* parameter can also be pushed beyond 1 to reach higher IORs.

The clearcoat layer uses a fixed IOR of 1.5, representative of *polyurethane*. Artists can scale its overall strength using the *clearcoat* parameter.

3.5.4 *Specular F*

For the Fresnel reflectance $F()$, the Schlick approximation was deemed sufficient:

$$F_{Schlick} = F_0 + (1 - F_0)(1 - \cos\theta_D)^5$$

F_0 is achromatic for dielectric and tinted for metals. The response always becomes achromatic at grazing incidence, as all light is reflected.

3.5.5 *Specular G*

In microfacet models, sometimes we can apply a derivation method to get a shadowing function $G()$ directly from the microfacet distribution, $D()$. This process is called *Smith G derivation*, and was also used in the case of the Disney BRDF, where the $G()$ term derives from the GGX distribution (section 3.5.3).

Before being passed to $G()$, the *roughness* parameter is scaled from $[0, 1]$ to $[0.5, 1]$. This is done because it has been observed that small roughness values result in exaggerated reflection gains.

15 The *azimuthal angle* is the angle of rotation of $\vec{H}$ around $\vec{N}$.

Remember from earlier that the roughness is squared. This is done after the scaling, resulting in a roughness parameter α_g for $G()$ that is $\alpha_g = (0.5 + roughness/2)^2$.

For the clearcoat specular layer, the GGX $G()$ is used simply with a fixed roughness of 0.25.

3.5.6 *Sheen*

The MERL BRDF database contains measures indicating that materials made of *fabric* exhibit *tinted specular response* at *grazing angles*. This makes sense, since cloth often has *transmissive fibers* which can pick up the material color also near object silhouettes. Another interesting observation is that, for a fixed *roughness* value, fabric seems to have a stronger Fresnel peak than other kinds of materials. The qualities we just mentioned are collectively referred to as the *sheen* of the fabric.

Hence, another lobe was added to the Disney BRDF to show extra grazing reflectance for fabric. Since this extra reflectance is very Fresnel-like, it is modeled with the Schlick-Fresnel "shape" $sheen \cdot (1 - \cos\theta_D)^5$. The sheen can also be tinted towards the base color with the use of the *sheenTint* parameter.

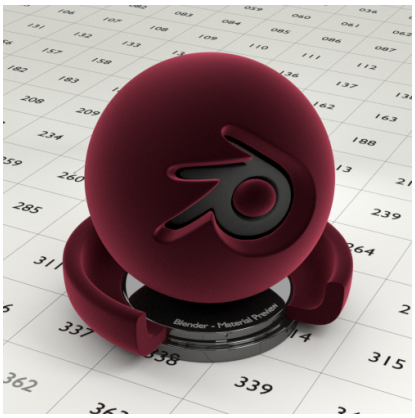

Fig. 46: Example of a *sheen* effect in Blender. Source: *Sheen BSDF*. Blender 4.3 Manual (license URL: https://creativecommons.org/licenses/by-sa/4.0/)

3.6 GLOBAL ILLUMINATION

A more advanced form of ray tracing is the *path tracing* light transport algorithm[16]. Path tracing was introduced in 1986 by Kajiya, and serves as a *Monte Carlo* solution to the *rendering equation*, which was also described first in Kajiya's paper (Kajiya 1986). The rendering equation is, effectively, the equation that every realistic rendering application is trying to solve. As we'll see, this is not an easy task, as proper (that is, realistic) solutions need to take into account *global illumination*.

In *global illumination* algorithms, the colors of surfaces are influenced not only by light sources, but also by other *objects*. This derives from the fact that, in the real world, light is reflected many times. If we think about it, this makes perfect sense. After reflection, new light is emitted by a surface, and this light will be incident to other surfaces in the scene, to then be reflected again. To put it simply, we can say that light "bounces" between objects many times. At each reflection, some portion of light is absorbed, making bounces less visually significant as their number grows. However, the first few ones might influence significantly the colors of surfaces. For instance, if we have a bright light shining on a red object, the reflected light could very well induce a reddish tint on nearby objects in the scene (Pharr, Jakob, and Humphreys 2023).

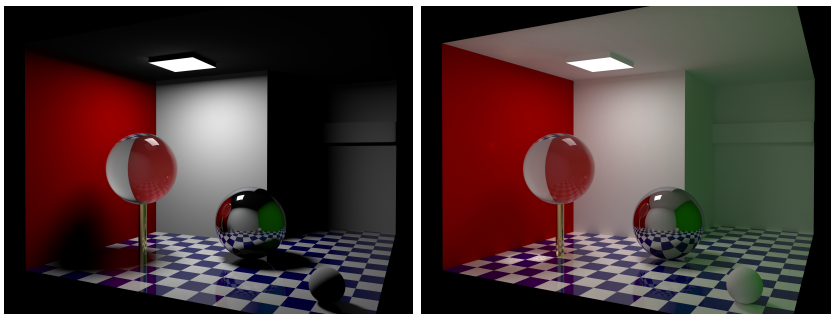

Fig. 47: Rendering with and without global illumination (global illumination can be seen in the image to the right). Example of a *sheen* effect in Blender. Source: Wikimedia Commons. (license URL: `https://creativecommons.org/licenses/by-sa/4.0/deed.en`)

In the following sub-sections, we'll see how the global illumination problem can be expressed through the rendering equation. We'll also mention a couple concepts related to Monte Carlo techniques, to under-

16 Used, for example, in the `pbrt` renderer. (Pharr, Jakob, and Humphreys 2023)

stand how the rendering equation can be (approximately) solved with their use.

3.6.1 *The Rendering Equation*

We've already seen a special case of the rendering equation, that is, the *reflection equation*:

$$L_o(\tilde{p}, \vec{\omega_o}) = \int_{H^2(\vec{N})} f_r(\tilde{p}, \vec{\omega_o}, \vec{\omega_i}) L_i(\tilde{p}, \vec{\omega_i}) cos\theta_i d\omega_i$$

Where, as we know, f_r is the bidirectional reflectance distribution function (BRDF). If we want to simulate proper light transport, this equation is not enough, as it doesn't take into account other ways that light could "move around" in a scene. For example, we know from Chapter2 that light doesn't only get *reflected*, but also *refracted* (or *transmitted*). Transmission can be modeled through a surface's *Bidirectional Transmittance Distribution Function* (*BTDF*) f_t, which we can define in a manner similar to the BRDF f_r (Pharr, Jakob, and Humphreys 2023). For convenience, the BRDF and the BTDF can be considered *together* in a function f, called the *Bidirectional Scattering Distribution Function* (*BSDF*). This function considers general "scattering" of light, which could include also subsurface scattering (that is, the *BSSDF*, mentioned in section ...).

The *rendering equation* (also called *light transport equation*) is related to the BSDF f. In fact, It uses f to express the distribution of radiance in a scene (Pharr, Jakob, and Humphreys 2023).

The rendering equation is based on the principle of *energy balance*:

> "The difference between the amount of energy going out of a system and the amount of energy coming in must also be equal to the difference between energy emitted and energy absorbed." (Pharr, Jakob, and Humphreys 2023)

Simply by enforcing this principle at a surface, we get:

$$L_o(\tilde{p}, \vec{\omega_o}) = L_e(\tilde{p}, \vec{\omega_o}) + \int_{H^2(\vec{N})} f(\tilde{p}, \vec{\omega_o}, \vec{\omega_i}) L_i(\tilde{p}, \vec{\omega_i}) cos\theta_i d\omega_i \quad (3.17)$$

Where L_e is the radiance emitted by the surface. This is integral equation is what we call *the rendering equation*.

3.6.2 *Monte Carlo Integration*

The rendering equation 3.17 is, in general, *impossible* to solve analytically. The problem lies in the integral over the hemisphere H^2. In fact, it requires us to sum the (weighted) incoming radiance contributions coming in from *all* (that is, *infinite*) directions above the surface point $\tilde{p}$.

However, even though we can't solve it in an analytical sense, *approximate solutions* to the rendering can be found. We can estimate the value of its integral by considering only the contributions of a *few*, random directions above $\tilde{p}$. Randomness is at the base of *Monte Carlo estimators*.

Monte Carlo estimators can be built to approximate the values of arbitrary integrals (Pharr, Jakob, and Humphreys 2023). Let's say that we want to compute

$$\int_a^b f(x)dx \tag{3.18}$$

And we have a supply of independent random variables $X_i \epsilon [a, b]$. Then, if the random variables are drawn from a probability distribution function $p(x)$, a Monte Carlo estimator of (3.18) can be defined as

$$F_n = \frac{1}{n}\sum_{i=1}^{n}\frac{f(X_i)}{p(X_i)}$$

It can be proven that the estimator F_n converges to the right answer, that is, the value of (3.18), with a convergence rate that's proportional to its variance, $\sigma[F_n]$ (Pharr, Jakob, and Humphreys 2023). Because the variance corresponds to the squared error, the error of a Monte Carlo estimator goes down at a rate of $O(n^{-1/2})$, for a number of samples n.

The main reason why Monte Carlo integration is preferred to other numerical methods, such as the *quadrature techniques*, is that its convergence rate is independent of the dimension of the integral. This makes it the only practical numerical integration algorithm for high-dimensional integrals, such as the ones found in path tracing (Pharr, Jakob, and Humphreys 2023).

Path tracing follows the path of light through multiple "bounces", resulting in nested integrals over hemispheres. However, instead solving them directly, it's more convenient to transform these nested integrals into non-nested ones, defined over a high-dimensional "path space", whose points are the geometric paths of light (Gortler 2012).

We now end this chapter by mentioning a simple sampling method for Monte Carlo integration over a hemisphere, taken from (Pharr, Jakob, and Humphreys 2023).

In the most general case of hemisphere sampling, we generate random directions ω over the hemisphere H^2 that are *uniformly distributed*. Mathematically, this means that their probability distribution function is constant:

$$p(\omega) = c$$

To derive the constant c, we can exploit the fact that probability density functions must integrate to one over their domain:

$$\int_{H^2} p(\omega) d\omega = 1 => c \int_{H^2} d\omega = 1 => c = \frac{1}{2\pi}$$

Hence, we have $p(\omega) = \frac{1}{2\pi}$. If we express ω in spherical coordinates $\theta \epsilon [0, \pi/2]$ and $\phi \epsilon [0, 2\pi]$, we get

$$p(\theta, \phi) = \frac{\sin\theta}{2\pi}$$

This is due to the fact that $d\omega = \sin\theta \; d\theta \; d\phi$, and therefore:

$$p(\theta, \phi) \; d\theta \; d\phi = p(\omega) \; d\omega$$
$$p(\theta, \phi) = \sin\theta \; p(\omega)$$

Next, we find the marginal probability distributions $p(\theta)$ and $p(\phi)$:

$$p(\theta) = \int_0^{2\pi} p(\theta, \phi) d\phi = \int_0^{2\pi} \frac{\sin\theta}{2\pi} d\phi = \sin\theta$$
$$p(\phi|\theta) = \frac{p(\theta, \phi)}{p(\theta)} = \frac{1}{2\pi}$$

Finally, we use the *inversion method*. This method maps uniform samples from $[0, 1)$ to a given 1D probability distribution by inverting the distribution's cumulative distribution function (CDF). Writing the CDFs, we get:

$$P(\theta) = \int_0^{\theta} \sin\theta' d\theta' = 1 - \cos\theta$$
$$P(\phi) = \int_0^{\phi} \frac{1}{2\pi} d\phi' = \frac{\phi}{2\pi}$$

Then we consider $P(\theta)$ and $P(\phi)$ as two uniformly distributed numbers ξ_1, ξ_2, and solve for θ and ϕ:

$$\xi_1 = 1 - \cos\theta \Rightarrow \theta = \cos^{-1}(1 - \xi_1)$$
$$\xi_2 = \frac{\phi}{2\pi} \Rightarrow \phi = 2\pi\xi_2$$

Since ξ_1 is uniformly distributed, we can replace it with $1 - \xi_1$:

$$\theta = \cos^{-1}\xi_1$$

We can now get our sampling direction by converting θ and ϕ to cartesian coordinates:

$$x = \sin\theta\cos\phi = \cos(2\pi\xi_2)\sqrt{1-\xi_1^2}$$
$$y = \sin\theta\sin\phi = \sin(2\pi\xi_2)\sqrt{1-\xi_1^2}$$
$$z = \cos\theta = \xi_1$$

4

THE BOXOFSUNLIGHT RENDERER

5

RESULTS

6

CONCLUSIONS AND FUTURE WORK

BIBLIOGRAPHY

Akenine-Möller, Tomas et al. (2018). *Real-Time Rendering, Fourth Edition. Online chapter: Real-Time Ray Tracing, version 1.4*. URL: https://www.realtimerendering.com/Real-Time_Rendering_4th-Real-Time_Ray_Tracing.pdf. (accessed: 19.03.2025).

Burley, Brent (2012). *Physically-based shading at Disney*. URL: https://blog.selfshadow.com/publications/s2012-shading-course/burley/s2012_pbs_disney_brdf_notes_v3.pdf. (accessed: 22.02.2025).

Cook, R. L. and K. E. Torrance (Jan. 1982). "A Reflectance Model for Computer Graphics". In: *ACM Trans. Graph.* 1.1, pp. 7–24. ISSN: 0730-0301. DOI: 10.1145/357290.357293. URL: https://doi.org/10.1145/357290.357293.

Feynman, Richard P., Robert B. Leighton, and Matthew L. Sands (2011). *The Feynman lectures on physics*. New Millennium. New York, NY: Basic Books.

Glassner, Andrew S. (1995). *Principles of Digital Image Synthesis*. San Francisco: Morgan Kaufmann.

– (2019). *An Introduction to Ray Tracing*. License: https://creativecommons.org/licenses/by/4.0/. URL: https://www.realtimerendering.com/raytracing/An-Introduction-to-Ray-Tracing-The-Morgan-Kaufmann-Series-in-Computer-Graphics-.pdf. (accessed: 12.03.2025).

Gortler, Steven J. (2012). *Foundations of 3D Computer Graphics*. Cambridge, MA: MIT Press.

Hery, Christophe and Ryusuke Villemin (2013). *Physically Based Lighting at Pixar*. URL: https://graphics.pixar.com/library/PhysicallyBasedLighting/paper.pdf. (accessed: 23.02.2025).

Hoffman, Naty (2012). *Background: Physics and Math of Shading*. URL: https://blog.selfshadow.com/publications/s2012-shading-course/hoffman/s2012_pbs_physics_math_notes.pdf. (accessed: 12.03.2025).

Hughes, John F. et al. (2014). *Computer graphics: principles and practice*. 3rd edition. Upper Saddle River, NJ: Addison-Wesley.

Kajiya, James T. (Aug. 1986). "The rendering equation". In: *SIGGRAPH Comput. Graph.* 20.4, pp. 143–150. ISSN: 0097-8930. DOI: 10.1145/15886.15902. URL: https://doi.org/10.1145/15886.15902.

National Aeronautics and Space Administration, Science Mission Directorate (2010). *The Electromagnetic Spectrum*. URL: https://science.nasa.gov/ems. (accessed: 12.03.2025).

Pharr, Matt, Wenzel Jakob, and Greg Humphreys (2023). *Physically based rendering: From theory to implementation*. 4th edition. Cambridge, MA: MIT Press.

Shirley, Peter, Trevor David Black, and Steve Hollasch (Aug. 2024). *Ray Tracing in One Weekend*. URL: https://raytracing.github.io/books/RayTracingInOneWeekend.html. (accessed: 19.03.2025).

Stark, Glenn (Feb. 2025). *light. Encyclopedia Britannica*. URL: https://www.britannica.com/science/light. (accessed: 11.03.2025).

Whitted, Turner (June 1980). "An improved illumination model for shaded display". In: *Commun. ACM* 23.6, pp. 343–349. ISSN: 0001-0782. DOI: 10.1145/358876.358882. URL: https://doi.org/10.1145/358876.358882.