



**FACULTY
OF MATHEMATICS
AND PHYSICS**
Charles University

MASTER THESIS

Bc. Marcel Hruška

**A Methodical Approach to the
Evaluation of Appearance
Computations**

Department of Software and Computer Science Education

Supervisor of the master thesis: doc. Alexander Wilkie, Dr.

Study programme: Computer Science

Study branch: Computer Graphics and Game
Development

Prague 2020

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In date
Author's signature

Dedication.

Title: A Methodical Approach to the Evaluation of Appearance Computations

Author: Bc. Marcel Hruška

Department: Department of Software and Computer Science Education

Supervisor: doc. Alexander Wilkie, Dr., Department of Software and Computer Science Education

Abstract: Abstract.

Keywords: key words

Contents

Introduction	3
1 Rendering and Color Science	4
1.1 Light	4
1.1.1 Color	5
1.1.2 Conversions to tristimulus color spaces	6
1.1.3 Photometry and Radiometry	6
1.2 Physically based rendering	7
1.2.1 Digital scene	9
1.2.2 BRDF	10
1.2.3 Global Illumination	11
1.2.4 Monte Carlo integration	12
1.2.5 Light transport algorithms	13
1.3 Spectral Rendering	16
1.3.1 Color representation	16
1.3.2 Advantages	17
1.3.3 Disadvantages	18
2 Appearance Computations	19
2.1 Reflectance	19
2.1.1 Fresnel equations	19
2.1.2 Microfacet theory	20
2.2 Polarization	21
2.2.1 Polarization in rendering	24
2.3 Dispersion	26
2.4 Iridescence	27
2.4.1 Iridescence in rendering	28
2.5 Fluorescence	29
3 Benchmark	30
3.1 Framework	30
3.2 Supported Spectral Renderers	30
3.2.1 Mitsuba2	30
3.2.2 ART	30
3.3 Scenes	30
3.3.1 GGX Reflectance	30
3.3.2 Spectral accuracy	30
3.3.3 Polarization	30
3.3.4 Fluorescence	30
3.3.5 Iridescence	30
3.4 Common Data	30
3.5 Usage	30
3.6 Open source contributions	30
3.6.1 GGX for ART	30
3.6.2 Iridescence for Mitsuba2	30

3.6.3	Multi-channel EXR support for jeri.io	30
3.7	Future Work	30
Conclusion		31
Bibliography		32

Introduction

Goals

Thesis organization

1. Rendering and Color Science

This chapter serves as an introduction to the computer graphics and the color science. We briefly overview basic aspects of these fields, mainly to familiarize the reader with some of the fundamental processes, their backgrounds and usages. We also establish the terminology, such as *rendering* or *RGB color space*, that will be used throughout the thesis frequently. A significant part of the following sections is based on Wyszecki and Stiles [25], Wilkie [23], Nimier-David et al. [15] and Pharr et al. [18].

First, we discuss the mechanisms behind the light and colors and then we look into the process of the physically based image reproduction.

1.1 Light

According to the definition by Barbrow [2], the light is "any radiation capable of causing a visual sensation directly". In other words, the visible light is an electromagnetic radiation that is perceivable by the human eye and allows us to see the objects around us.

As all electromagnetic radiations, the light also propagates in form of waves. The oscillation direction of these waves does not change the color of the light but it may interact differently with the reflective/refractive objects as it passes through them. A representation of such wave propagation is displayed in Figure 1.1.

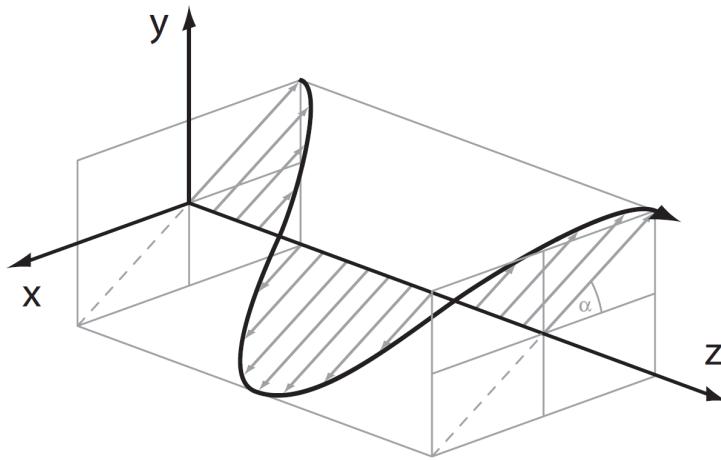


Figure 1.1: A propagation of wave [23]

Normally, by the term light we mean the *visible light* (also called the *white light*) which consists of multiple waves of unique frequencies (wavelengths). There are no exact boundaries to the visible spectrum as distinct human eyes might perceive light slightly differently but the lower boundary is estimated between 360 and 400nm and the upper boundary from 760 to 830nm [20]. Above this range there is the infrared light and below the ultraviolet. An explanatory image of the known electromagnetic wavelengths can be found in Figure 1.2.

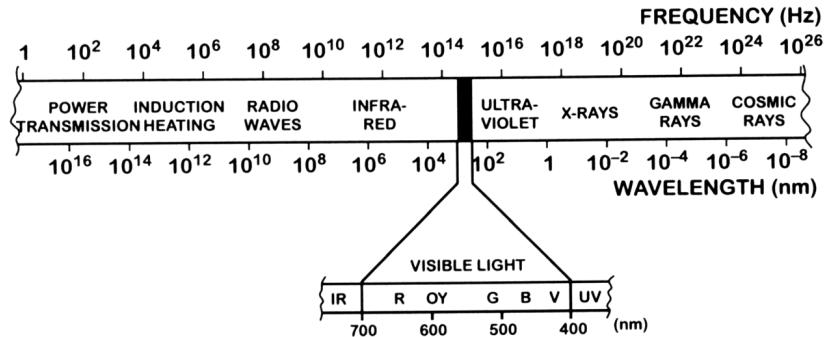


Figure 1.2: An image displaying various wavelengths [23]

1.1.1 Color

While we observe an illuminated object, three different signals are sent from the eye sensors (rods and cones) to the brain, each representing a red, a green or a blue wavelength. When put together inside the brain, they form a sensation of the final color.

To categorize the colors, we formed several reproducible representations of them called the *color spaces*. A natural decision was to create an *RGB color space* as it directly correlates with the signals sent from the human eye's rods and cones. Note that multiple variations of the RGB color space exist such as *sRGB* or *Adobe RGB*. An illustrative comparison between several color gamuts is shown in Figure 1.3.

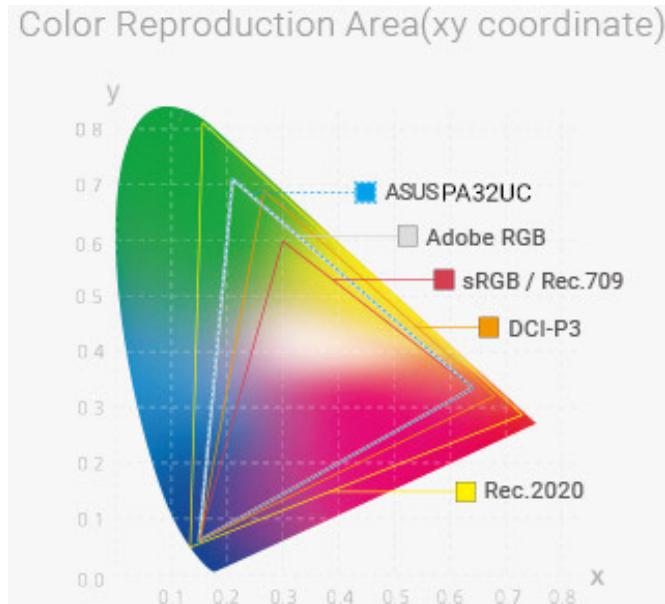


Figure 1.3: An illustrative comparison between five RGB gamuts by Asus ¹

In 1913, the Commission internationale de l'éclairage (International Commission on Illumination), shortly CIE, was formed as an authority that defines almost everything that concerns colors and their perception. In 1931, they conducted

¹<https://www.asus.com/Microsite/ProArtMonitor/experience-truecolor.html>

the color matching experiments to obtain the three color matching functions that would convert the color stimuli perceived in our eyes to the *CIE RGB* color space. As these functions had a negative component, a new imaginary color space was created called *CIE XYZ*. These conversions are further described in subsection 1.1.2.

CIE also defined *CIE L*a*b** color space, standard illuminants D65 and D50 and many others.

1.1.2 Conversions to tristimulus color spaces

The conversion from the spectrum to a RGB color space:

1. Compute the tristimulus value XYZ using the CIE color matching functions (shown in Figure 1.4)

$$\begin{aligned} X &= \int P(\lambda) \bar{x}(\lambda) d\lambda \\ Y &= \int P(\lambda) \bar{y}(\lambda) d\lambda \\ Z &= \int P(\lambda) \bar{z}(\lambda) d\lambda \end{aligned}$$

, where $P(\lambda)$ is the spectral power distribution and $\bar{x}(\lambda)$, $\bar{y}(\lambda)$ and $\bar{z}(\lambda)$ are the color matching functions.

2. Convert the XYZ to the desired RGB color space using a transformation matrix. Depending on the specific RGB color space, the matrix differs — an example of CIE XYZ to sRGB conversion is demonstrated in item 2.

$$\begin{aligned} r &= 3.240X - 0.969Y + 0.55Z \\ g &= -1.537X + 1.875Y - 0.204Z \\ b &= -0.498 + 0.041Y + 1.057Z \end{aligned}$$

3. (Optional) As the resulting r,g,b values may be negative, a gamut mapping might be necessary.

1.1.3 Photometry and Radiometry

Two different science fields were developed to quantify the light – the *photometry* and the *radiometry*. The radiometry recognizes the light as an electromagnetic radiation while the photometry focuses more on the human perception of the light. Despite the distinct purposes, their quantities are often easily convertible.

Radiometric Quantity	Photometric Equivalent
Spectral radiant energy [J]	Luminous energy [<i>Lumen – second</i>]
Radiant flux [W]	Luminous flux [<i>Lumen</i>]
Irradiance [$W.m^{-2}$]	Illuminance [$Lumen.m^{-2}$]
Radiant intensity [$W.sr^{-1}$]	Luminous intensity [$candela = Lumen.sr^{-1}$]
Radiance [$W.sr^{-1}.m^{-2}$]	Luminance [$candela.m^{-2}$]

²<https://scipython.com/blog/converting-a-spectrum-to-a-colour/>

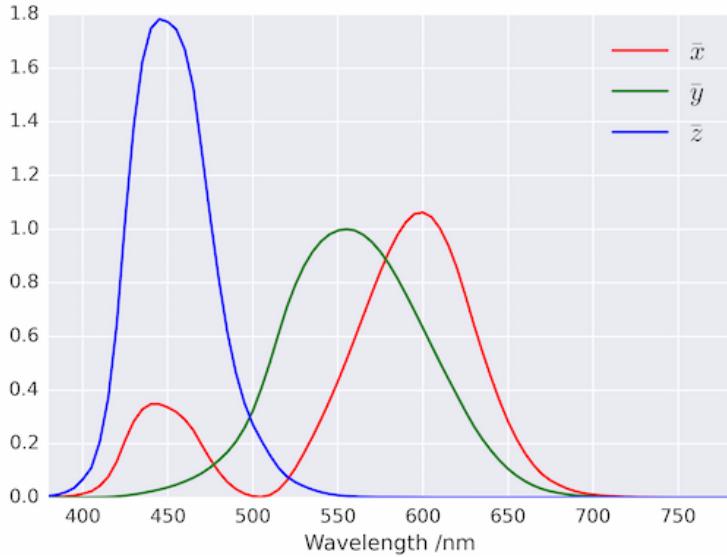


Figure 1.4: Color matching functions plotted in Python²

A brief description of each of them:

Spectral radiant energy Amount of the light energy at some wavelength

Radiant flux Amount of the light energy with respect to time

Irradiance Flux at a specific point (space)

Radiant intensity Flux in a direction (sr (steradian) is a unit of the solid angle
— surface on a unit sphere, whole sphere has 4π steradians)

Radiance Spatial and directional flux

The relationship between these quantities is described by the *spectral efficiency function*. It states how efficiently a human eye reacts to different wavelengths, i.e. we can detect some wavelengths more easily than others. As we can see in Figure 1.5, the scotopic (night) perception peaks at around 507nm and the photopic (day) at 555nm.

1.2 Physically based rendering

One of the ultimate goals of the computer graphics is the ability to reproduce visually plausible and physically coherent images that should be indistinguishable from a photograph based on a description of a scene. Such process is called the *photo realistic rendering*. In this thesis, we abbreviate the term and call it simply the *rendering* as the non-photo realistic one does not concern us.

Depending on the implementation, the renderer simulates various phenomena commonly seen in nature such as light reflections, refractions, shadows, etc. Providing a powerful hardware, modern renderers adapt various physical models (or their approximations) of light transport or material properties to provide accurate photo realistic results. In reality, the renderers are so capable that the

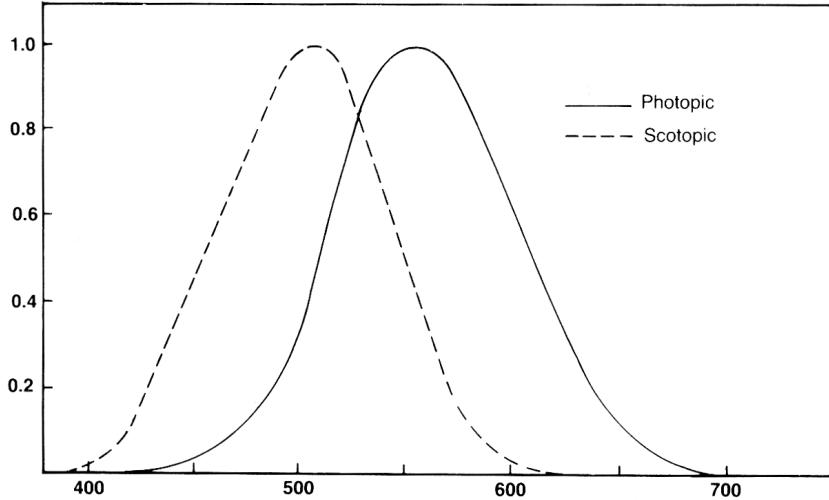


Figure 1.5: Relative luminous efficiency function [23]

rendered images are almost identical to the real life photos. An example can be seen in Figure 1.6.



Figure 1.6: An image generated with the Corona Renderer³

The main idea is similar for every renderer:

1. A 3D digital scene is described by the objects it contains
2. A light simulation algorithm runs for every visible pixel from the viewer
3. Upon object interaction, the shading of the intersected point is computed
4. As the algorithm terminates, a picture ("photograph") of the scene called the *render* is created

³<https://corona-renderer.com/gallery>

Please see figure Figure 1.7 for a simple demonstration of this workflow.

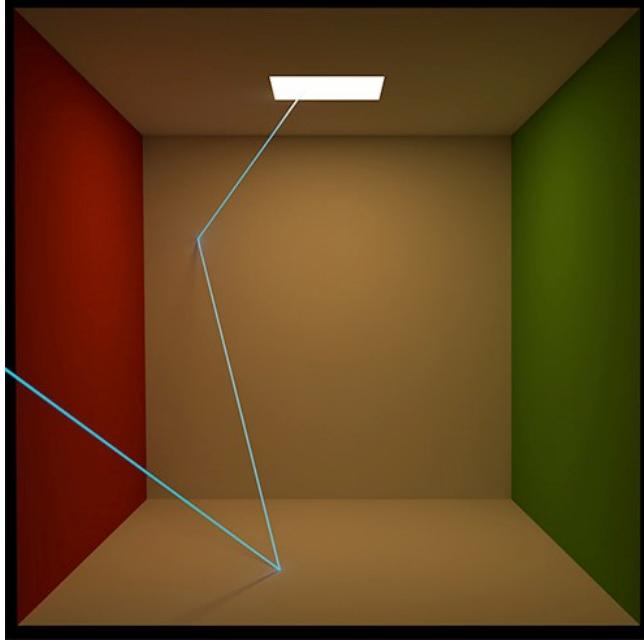


Figure 1.7: A visualization of a light transport algorithm (path tracer) [1]

1.2.1 Digital scene

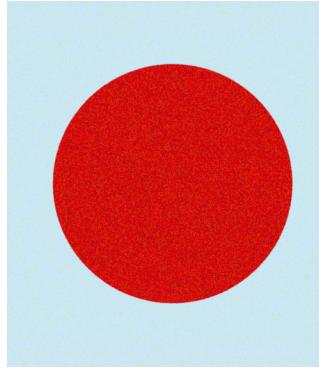
Basic elements of a digital scene are roughly the same for each renderer.

Camera A camera (or a sensor) in a digital scene works in the same manner as in real life — it records a picture. Generally, you may define the coordinate position and the viewing vectors but also the properties such as focal distance or the type of the film.

Light source The scene needs to be illuminated by one or multiple sources in order to be visible. The common kinds of lights are point light, area light, spot light or environment (constant) lighting.

Objects The actual visible contents of the scene are objects. Almost all rendering systems offer a choice to either use their precomputed basic geometry such as spheres or triangles or to include a mesh geometry described in an external file (usually created by an external modeling software). These objects must state their material properties so that the algorithm may correctly interact with them, e.g. diffuse vs. reflective material.

Unfortunately, as each renderer may have a very unique implementation details, the formats of the scenes are vastly different. For example, mitsuba uses XML but PBRT has its own specific format. An example of a simple scene for Mitsuba2 can be found in Figure 1.8.



```

<scene version="2.0.0">
  <!-- Light transport algorithm -->
  <integrator type="path"/>

  <!-- Camera looking at the sphere -->
  <sensor type="perspective">
    <transform name="to_world">
      <lookat origin="0,-6,0" target="0,0,0" up="0,0,1"/>
    </transform>
  </sensor>

  <!-- Red sphere in the middle -->
  <shape type="sphere">
    <bsdf type="diffuse">
      <rgb name="reflectance" value="1.0,0.0,0.0"/>
    </bsdf>
  </shape>

  <!-- Light blue light all around the scene-->
  <emitter type="constant">
    <rgb name="radiance" value="0.6,0.8,0.9"/>
  </emitter>
</scene>

```

Figure 1.8: A simple scene rendered with Mitsuba2 (left) along with its scene description (right)

1.2.2 BRDF

The fundamental part of the rendering process is its implementation of the light transport simulation. This and the following sections will describe the physics theory and the models behind the light transport. Then we take a look at the specific algorithms.

The materials are described by *Bidirectional Distribution Reflectance Function*, shortly *BRDF* [14]. It looks as follows:

$$f_r(\omega_i, \omega_o) = \frac{dL_o(\omega_o)}{L_i(\omega_i)\cos\theta_i d\omega_i} \quad (1.1)$$

This function is given the incoming vector ω_i , the outgoing vector ω_o and it states how much radiance is reflected from the direction ω_i ($L_i(\omega_i)$) to the direction ω_o ($L_o(\omega_o)$) for the specific material. An image interpretation of the function is in Figure 1.9. As it is a distribution function, we can also reformulate it's meaning as a probability density that a defined amount of light energy gets reflected from ω_i to ω_o .

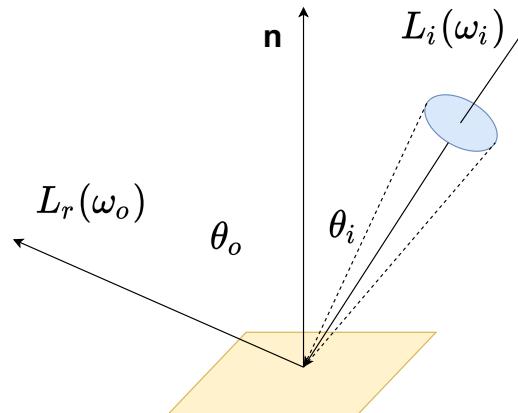


Figure 1.9: Bidirectional Distribution Reflectance Function

In the simplest cases, the BRDF states how reflective the surface of an object

is. The renders of diffuse, rough glossy and mirror materials are compared in Figure 1.10.

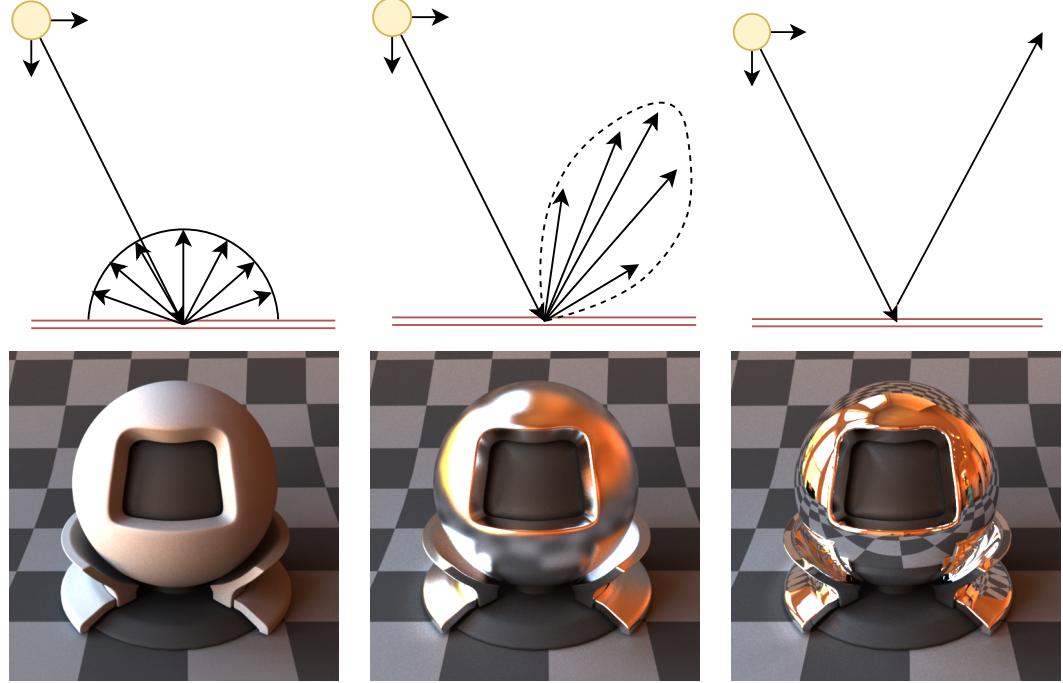


Figure 1.10: A preview of diffuse (left), glossy (middle) and mirror (right) materials rendered in Mitsuba2 along with their illustrative BRDF visualizations

Physically based BRDFs must fulfill several properties 8:

Heimholtz reciprocity The amount of reflected energy from the incoming direction to the outgoing direction is equal to the amount of energy in the reversed directions ($f_r(\omega_i, \omega_o) = f_r(\omega_o, \omega_i)$).

Energy conservation The amount of reflected energy cannot be larger than all received energy.

Positivity BRDF is always positive ($f_r(\omega_i, \omega_o) \geq 0$).

Note that BRDF concerns only opaque surfaces. There exist multiple distribution functions that describe behavior of other materials, for example:

BTDF Describes light transmission

BSDF Combination of BTDF and BRDF (e.g. glass, water)

BSSRDF Considers scattering of the light under the surface as well (skin)

1.2.3 Global Illumination

With the BRDF defined, we can now formulate an equation that evaluates the global illumination of a scene — illumination of each point from all light sources. It is generally called the *rendering equation* [12]:

$$L_o(x, \omega_o) = L_e(x, \omega_o) + L_r(x, \omega_o) \quad (1.2)$$

Let's break it down first:

x is the currently computed point in the scene.

L_o is the outgoing radiance.

L_e is the emitted radiance of the point x as x can be on a light source.

L_r is also called the *reflectance equation* and it states the total amount of the reflected radiance for all contributions of the incident radiance. Hence, it is an integral over the upper hemisphere over x that look as follows:

$$L_r(x, \omega_0) = \int_{\Omega} f_r(x, \omega_o, \omega_i) L_i(x, \omega_i) \cos \theta_i d\omega_i \quad (1.3)$$

, where

$f_r(x, \omega_o, \omega_i)$ is the BRDF of x as defined in Equation 1.1.

$L_i(x, \omega_i)$ is the incoming radiance from a light source.

An image interpretation of the reflectance equation can be seen in Figure 1.11.

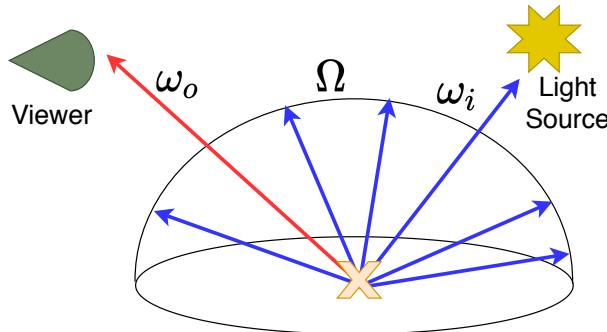


Figure 1.11: Reflectance Equation

As a matter of fact, each light transport algorithm tries to solve some of the formulations of the rendering equation.

Interestingly, the light transport is recursive in nature. As we can see from the rendering equation, to compute the outgoing radiance at a certain point x , we need to know all the contributed incoming radiances. These do not necessarily have to originate at a light source — the incoming radiance may come from another, non-emitting point y in the scene as a result of the rendering equation computed at the point y .

1.2.4 Monte Carlo integration

Before we proceed to the actual algorithms that evaluate the rendering equation, we briefly introduce a method that is used to approximate the definite integral part of the equation — *Monte Carlo integration* [6].

Formally, for a multidimensional definite integral

$$I = \int_{\Omega} g(x) dx \quad (1.4)$$

Monte Carlo (MC) estimates I as

$$\langle I \rangle = \frac{1}{N} \sum_{k=1}^N \frac{g(\xi_k)}{p(\xi_k)}; \xi_k \sim p(x) \quad (1.5)$$

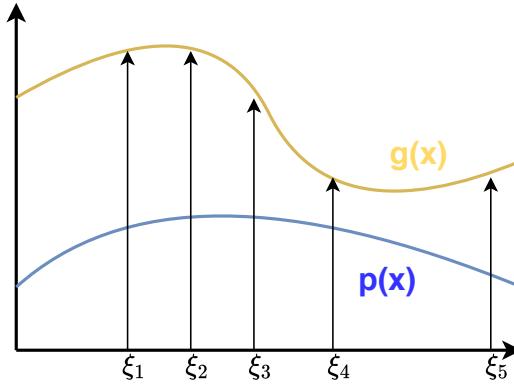


Figure 1.12: Monte Carlo method 2D visualization

In other words, Monte Carlo is a non-deterministic method that sums N randomly chosen samples ξ_k , computes their values $g(\xi_k)$ and averages them. To reduce variance, an importance sampling is introduced by drawing samples from a distribution $p(x)$ that is chosen for each specific problem to approximate the former $g(x)$ function. In reality, the importance sampling ensures that if we pick some samples twice as much, we decrease their weight to half.

There exist other methods that are used to approximate integrals such as deterministic quadrature or Markov Chain Monte Carlo (MCMC).

1.2.5 Light transport algorithms

Path tracing

Over the years, a large number of various light transport algorithms and their variations have been developed, where each has its own benefits. The one that we will mention the most in this thesis is called the *path tracing*. Its core idea is simple:

1. For each pixel in the image plane, shoot a primary ray r from camera into the scene.
2. If r hits a non-emitting object at point x :
 - 2.1. Compute BRDF at x .
 - 2.2. Generate a new random direction ω . Ideally, the distribution of the generated direction should be proportional to the BRDF — e.g. diffuse BRDF would generate a direction uniformly over a hemisphere while glossy BRDF would prioritize samples from the reflectance lobe (look at Figure 1.10).
 - 2.3. Add the BRDF value to the final color of the pixel.

2.4. Check for a terminating condition — there exist several options, usually a combination of them is applied:

Maximum depth User specified maximum number of recursions.

Russian Roulette Randomly choose if the ray survives, with each consecutive ray the chance lowers.

BRDF-proportional Depending on the surface material, we decide whether the ray survives or not. For example, reflective or refractive surfaces need a lot more recursions as they propagate the light further to the scene than diffuse surfaces.

- 2.5. In case the termination was not successful, bounce – shoot a secondary ray r from the point x in the direction ω and continue from step 2.
3. If r hits an emitting object (light source), add it's emission L_e to the final color of the pixel
4. If no scene geometry is hit, terminate the algorithm and add the color of the surrounding light (if there is any).

The bouncing of the light in the scene nicely correlates with the recursive nature of the rendering equation. Even though the path tracing is a slow algorithm (e.g. not suitable for real-time rendering in games), it's variations can be extremely accurate, even indistinguishable from a real photograph.

MIS In the algorithm described above, the direct illumination computation of each scene intersection is dependent only on the BRDF of the intersected surface and consequent walk to the light source. However, such scenario that the light source is hit at the end of every walk is greatly dependent on the number of samples and the maximum allowed depth of the recursion. Consequently, this creates variance which can be easily improved the integration of the *multiple importance sampling* (MIS). Generally, it involves a combination of multiple sampling techniques but in our case it is a combination of the BRDF proportional sampling and the light source sampling — in each step of the path tracing, every light source that is visible from the intersected point contributes to it's value. Both sampling methods are, of course, weighted to avoid an over-illumination.

Volumes Another aspect that needs to be accounted for in the rendering process are volumetric objects such as fogs or smokes. Normally, there are two ways that a volumetric object may effect the light passing through it. The volume is either attenuating the light by absorbing it or scattering to different directions. Or the volume can also strengthen it by emitting light (e.g. flame) or scattering light from different directions to the current one.

A single walk of a path tracer capable of volume tracing and MIS sampling is visualized in Figure 1.13.

Other methods

As the path tracing is of our main concern in this thesis due to its physically based and unbiased properties, some of the other global illumination techniques are described only briefly here:

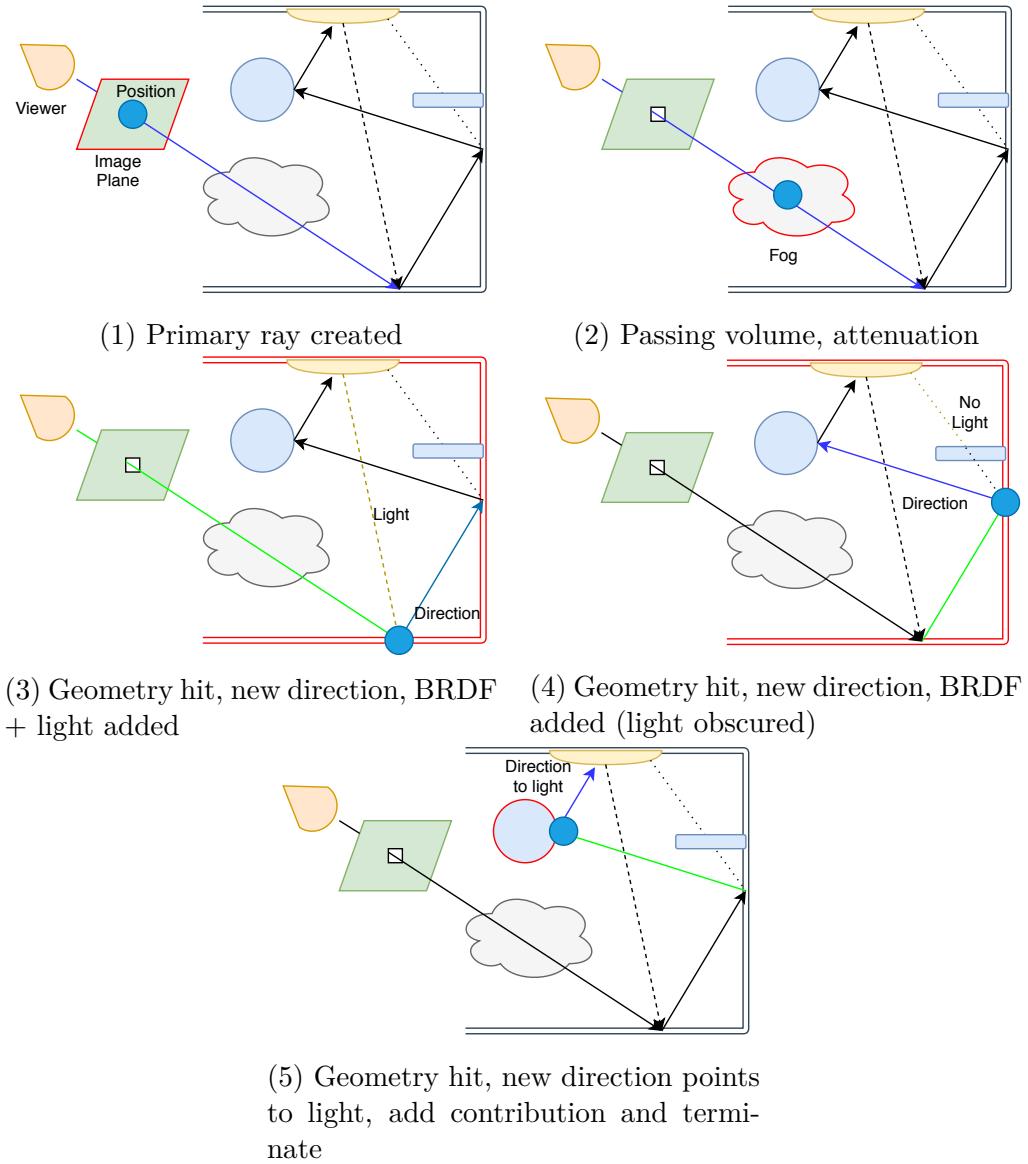


Figure 1.13: A visualization of a single walk in path tracer

Ray tracing [9] Similar to the path tracing but there are no bounces from the surfaces, simulates only reflections, refractions, scattering etc. Capable of realtime rendering these days.

Photon mapping [11] Two rays are traced independently — from the camera and from a light source until termination, then the radiance is computed based on their final positions. Faster at some scenarios but biased (does not have to converge to a correct solution).

Radiosity [19] Uses the finite element method instead of Monte Carlo. View independent, the light is traced from the source and bounced (possibly) to the viewer. Good for precomputations.

1.3 Spectral Rendering

So far, we've considered the colors to be internally represented by a tristimulus color space during the rendering process. For the explanation purposes, let's consider it to be RGB color space — the objects are defined in RGB, the path tracing step colors are RGB and the output color is RGB. In a large number of scenarios, this workflow is sufficient as we are capable of simulating a majority of the common aspects (e.g. optics) while keeping the rendering simple and robust. Unfortunately, the RGB color space is only a fraction of the visible gamut and does not contain any information about the light as an electromagnetic radiation. Consequently, we are loosing a significant amount of information, causing the colors to be at times inaccurate and some phenomena completely impossible to render.

Therefore, a new approach to the rendering has been introduced that internally represents the colors as a spectrum distribution function instead of a tristimulus color space — the *spectral rendering*. The core idea is to track and sample several wavelengths at once for each step of the path tracing and to perform the integration all over them. We also generalize the BSDF to account for the wavelengths: $f_r(\lambda, \omega_i, \omega_o)$.

Most of the phenomena evaluated in this thesis are a direct consequence of the light's nature as an electromagnetic radiation, hence the primary focus is placed on the spectral rendering. The following sections are largely based on Wilkie and Purgathofer [24].

1.3.1 Color representation

If the spectral rendering is desired, there is a need to store the spectral distribution function. While several techniques are feasible, it is often a matter of a simple trade-off between the precision and the performance. For example, sampling wavelengths uniformly each 10nm would yield significantly more accurate results than sampling only four wavelengths in total. However, such approach might become unbearable in terms of speed and memory. Moreover, the spectral functions are mostly quite smooth. Some examples of these functions can be seen in Figure 1.14.

A typical approach is to sample at larger ranges (10 or more nanometers) and combine it with the basis functions [17]. More approaches and their details are briefly explained in Wilkie and Purgathofer [24].



Figure 1.14: Spectral curves measured for different colors[10].

1.3.2 Advantages

The spectral rendering presents the ability to reproduce the colors in a more photorealistic way. We might not see it at first, but the colors produced by a conventional RGB renderer tend to be slightly over-saturated as they do not account for the spectral characteristics which might attenuate the final color. A comparison between the RGB and the spectral render of the same scene by Mitsuba2 is shown in Figure 1.15.

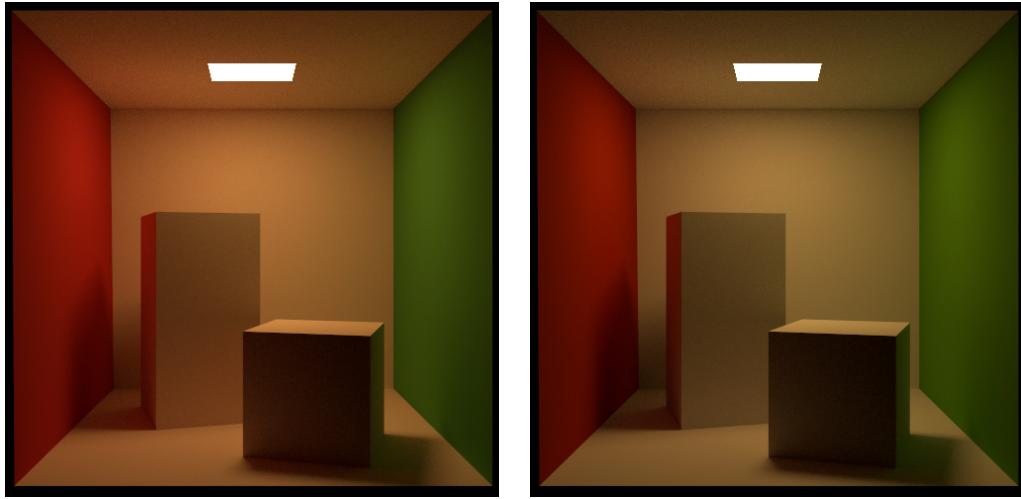


Figure 1.15: A comparison between the RGB render (left) and the Spectral render (right) of the same scene by Mitsuba2.

Still, the biggest improvement is the possibility to reproduce some of the natural phenomena for which the tracing of multiple wavelengths is an absolute necessity. Namely, those are:

Fluorescence Absorption and re-emission of a different color

Dispersion Splitting of the white light into its wavelength components via refraction

Polarisation Change of the oscillation direction of a light wave

Iridescence Thin layer constructive/destructive interference

They are explained in detail in chapter 2.

1.3.3 Disadvantages

On the other side, the need to numerically integrate over multiple wavelengths introduces a problem of the chromatic noise. Even though the performance is not a key factor, it obviously worsens as well.

Moreover, we are still unable to properly display spectral images on the current monitors. The only viable way is to store the distinct spectral bands as separate images. Clearly, it is quite inconvenient to have multiple results instead of one. Therefore, the rendering needs to have a well-done spectrum to final RGB conversion as the final image is still being displayed on an RGB monitor. Fortunately, the conversion is well-defined and fairly simple to implement.

Due to the fact that the RGB gamut is only a subset of the visible spectrum, the situation gets significantly more complicated for the reversed conversion. As there exist infinitely many spectra for one RGB value, several techniques were proposed to convert the tristimulus to the spectral domain — commonly called the *spectral upsampling*. For those who may be interested in the details of the current development to the spectral upsampling, refer to the article by Jakob and Hanika [10] which proposes a solution that is capable of converting full sRGB gamut with zero error.

There exist several reasons to integrate the spectral upsampling, mainly because the spectral values are a lot harder to obtain and to use. You need a specific device (spectrometer) that would measure the color values under a specific light and then use regularly distributed samples from it as an input. Because of the reproducibility of the RGB color space and its legacy usage (lots of existing textures are already defined in RGB), it is a lot more convenient to input the values of textures as an RGB value, convert them internally to the spectral domain and convert them to the desired color space for the output image.

2. Appearance Computations

So far, we've covered the fundamental basics of the light and the rendering process to be able to comprehend the more advanced techniques practiced in the computer graphics. As we've mentioned before, our primary goal is to evaluate the computational accuracy of several specific appearance sensations. Even though they are quite common in every day life, their integration to the modern renderers is, to this date, rare. In this chapter, we discuss these phenomena individually — their manifestations in the nature, the physics behind them and finally, their computations in the rendering process.

2.1 Reflectance

The reflective surfaces are a surprisingly common sighting. As the perfectly diffuse materials basically do not exist in the nature, a large set of the materials that surround us are considered glossy. In subsection 1.2.2, we defined a bidirectional reflectance distribution function that defines the reflective properties of a material.

2.1.1 Fresnel equations

It is necessary to know the basics of the geometry optics to be able to properly define a reflectance model. First of all, the *Snell's law* [18]

$$\eta_i \sin \theta_i = \eta_t \sin \theta_t \quad (2.1)$$

states that the incoming angle θ_i (angle between the surface normal and the incoming direction) times the *index of refraction* of the entering medium η_i must be equal to their transmitted counterparts. In other words, knowing the indices of refraction of the entering and the leaving media and the incoming direction, we can compute the transmitted direction.

The index of refraction (IOR) varies from material to material (e.g. IOR of glass is 1.5) and it essentially states how fast the light travels through the specific material.

However, this gives us only the direction of the refracted light. In most cases, we also need to know the ratio between the reflected and the refracted light. Depending on the polarization of the light (further explained in section 2.2), the *Fresnel equations* take the two following forms [18]:

$$r_s = \frac{\eta_t \cos \theta_i - \eta_i \cos \theta_t}{\eta_t \cos \theta_i + \eta_i \cos \theta_t}$$
$$r_p = \frac{\eta_i \cos \theta_i - \eta_t \cos \theta_t}{\eta_i \cos \theta_i + \eta_t \cos \theta_t}$$

From these, we can compute the *Fresnel reflectance* for an unpolarized light:

$$F_r = \frac{1}{2}(r_s^2 + r_p^2) \quad (2.2)$$

The transmitted energy is equal to $1 - F_R$ due to the energy conservation law.

Note that the previous computations describe only a subset of the materials called the *dielectrics*, which are the materials that do not conduct electricity and are capable of transmitting light, such as glass, water, diamond, etc. The second large group consists of the *conductors* which are basically all metals/materials with opaque surfaces. The light is actually transmitted into the conductor, however, due to its physical properties, it is quickly absorbed. There exists a third group of the *semiconductors* which are very rarely considered in the physically based rendering, therefore we skip them. A comparison between a dielectric and a conductor is shown in Figure 2.1.



Figure 2.1: A preview of a dielectric (diamond, left) a conductor (aluminium, right) rendered in Mitsuba2 [1]

While this is a matter of simply computing the Fresnel equations, the practice is more complicated as most of the commonly seen materials are not perfectly smooth. Slight imperfections, either on purpose or due to the manufacturing errors, can be found on a large majority of all materials which makes them rough.

2.1.2 Microfacet theory

With that in mind, the *microfacet theory* was developed by Cook and Torrance [7] to address this aspect and provide a theoretical representation of the rough surfaces.

The main idea is that a rough surface consists of the *microfacets* – a collection of very small surfaces distributed statistically throughout the whole underlying *macrosurface*. We compute the illumination (BRDF) for each of these microfacets (commonly considered to be a perfect mirror) and their aggregate behavior states the final scattering. An example of such distribution is shown in Figure 2.2.

As these microfacet computations are local, we need to consider the possibility that they might obscure each other. Three main aspects are accounted for:

Masking Microfacet is not visible from the viewer

Shadowing Microfacet is not reachable from the light source

Interrecflection Bounces between the microfacets

Many variations to the Cook-Torrance model have been developed, such as it's predecessor Torrance-Sparrow [21] or Oren-Nayar [16] for diffuse reflectance.

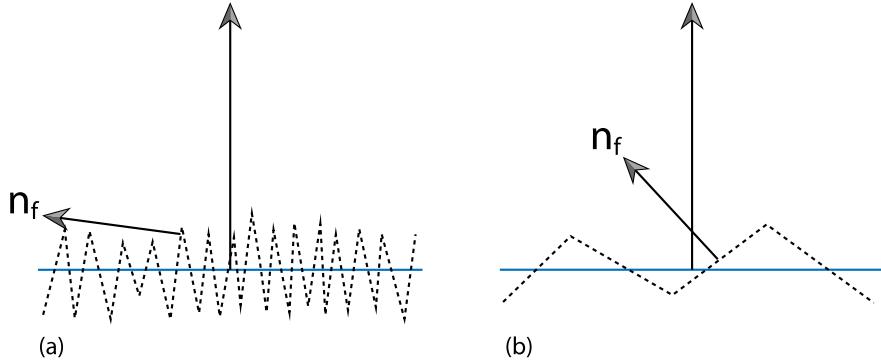


Figure 2.2: A demonstration of a very rough (left) and a relatively smooth (right) microfacet distribution [18]

In this thesis, we focus on the distribution functions of the microfacets as they are ultimately the deciding factor of the rough surface look. A nice comparison of the three commonly used microfacet distributions — *Phong*, *Beckmann* and *GGX* — along with their distribution functions, masking functions and sampling equations can be found in the Walter et al. [22]. As the exact formulations of those three methods are not a necessity for this thesis, we provide only a brief overview for each of them and an illustrative comparison between the *GGX* and the *Beckmann* of the same roughness in Figure 2.3.

Phong Even though the Phong distribution is purely empirical (not physically based), it is still quite popular choice for the microfacet distribution as it is simple to implement and provides sufficient results.

Beckmann The Beckmann distribution [3] is already physically based and for a long time has been considered the best solution to the rough surfaces as it is based on the Gaussian roughness. However, with the parameters set appropriately, it still provides the results very similar to the Phong distribution.

GGX The GGX distribution [22] was introduced as an improvement over the Beckmann’s solution for some cases. It maintains stronger tails, thus better shadowing and is based on the measured data of the real rough materials.

2.2 Polarization

Similarly to all kinds of the electromagnetic radiation, the light also propagates through space as a wave. The oscillation direction of this wave neither defines nor modifies the color of the light but it makes the light behave differently upon an interaction with certain materials. Figure 2.4 explains the different directions of a wave.

In the light’s natural state (sun, common light bulb), the directions of it’s oscillation are arbitrary — such light is called *unpolarized*. The *polarized* light maintains a restricted direction of oscillation and it is a result of a *polarization* process. Note that the light is often only partially polarized as the restriction of the direction does not have to be perfect and allows some variations.

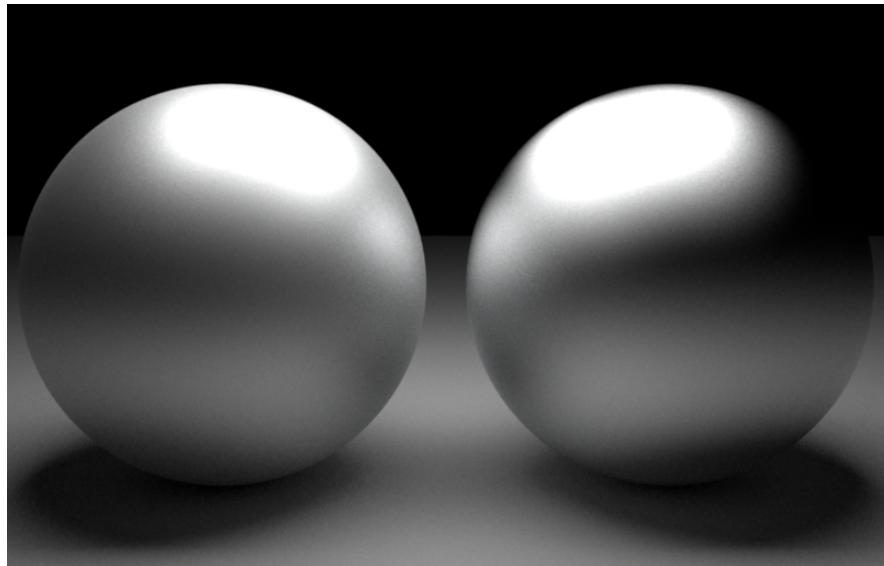


Figure 2.3: A rough aluminium sphere with the GGX distribution (left) compared to its Beckmann equivalent (right) rendered in Mitsuba2

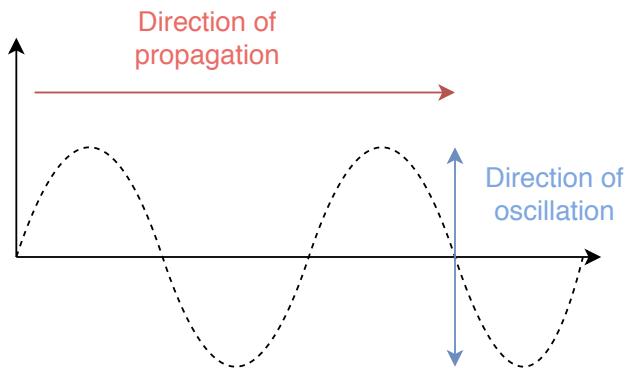


Figure 2.4: Demonstration of the direction of oscillation and the direction propagation

Depending on the shape of their electric fields, we distinguish three types of polarization which are demonstrated in Figure 2.5.

To create a linearly polarized light, one may put a dielectric object in the direction of propagation of an unpolarized light. Due to the optical properties of the dielectrics, the reflected and the transmitted light will be polarized proportionally to the angle of the incidence. The angle at which the reflected light is perfectly polarized is called the *Brewster's angle* [5]. It is computed by the following formula:

$$\theta = \arctan\left(\frac{n_2}{n_1}\right) \quad (2.3)$$

, where n_1 is the IOR of the incident medium and n_2 of the transmitted medium.

In this case, we distinguish two types of the linearly polarized light depending on their orientation to the incident plane. The reflected light is called *p-polarized*

¹<http://hyperphysics.phy-astr.gsu.edu/hbase/phyopt/polclas.html>

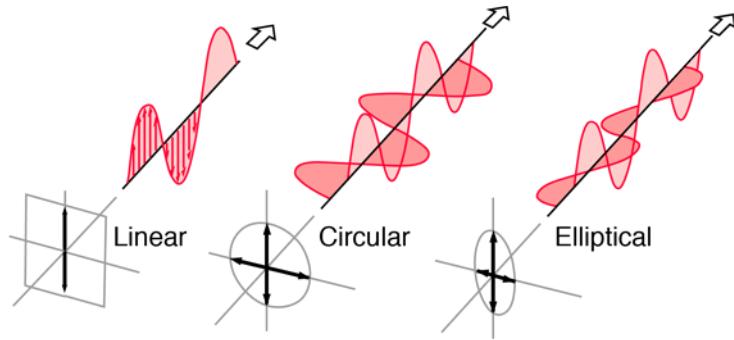


Figure 2.5: An illustrative demonstration of the different types of polarization¹

as its oscillation is parallel to the plane of incidence. The transmitted light is called *s-polarized* as its oscillation is perpendicular (from the German word) to the plane of incidence. Both are shown in Figure 2.6.

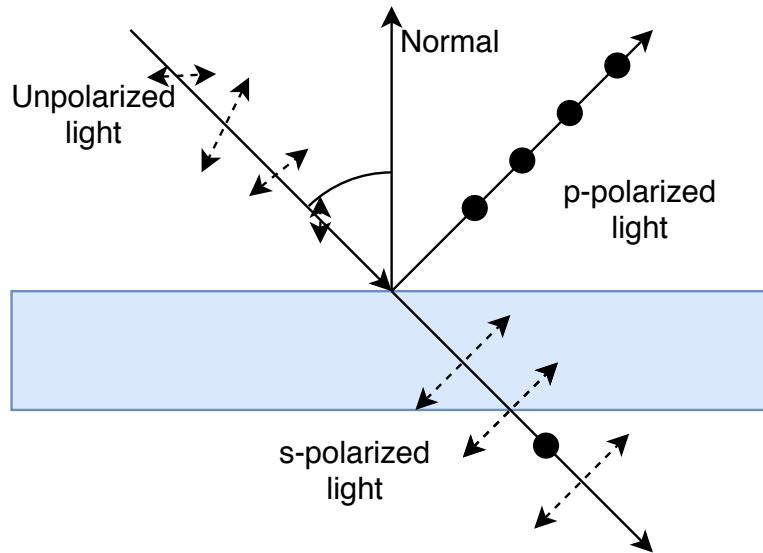


Figure 2.6: Perfectly p-polarized reflected and partially s-polarized light refracted from a dielectric interface at the Brewster's angle

The principle of Brewster's angle is used in a material called the *polarizer*. As the name suggests, it polarizes the light, restricting it's direction of oscillation accordingly the specifics of the polarizer. If the light that passes through the polarizer is of the opposite (perpendicular) direction to the polarizer's transmission orientation, it won't let it through and no light is visible. Figure 2.7 illustrates the effects of a polarizer.

This property is frequently incorporated in the sunglasses or camera filters to reduce the glare of the sun reflected from a horizontal surface. The reflected p-polarized light goes through a polarizer with a perpendicularly oriented transmission axis which consequently eliminates the incoming light. The effect of a polarizing filter is shown in Figure 2.8.

¹<https://www.apioptics.com/about-api/resources/visible-light-linear-polarizer/>

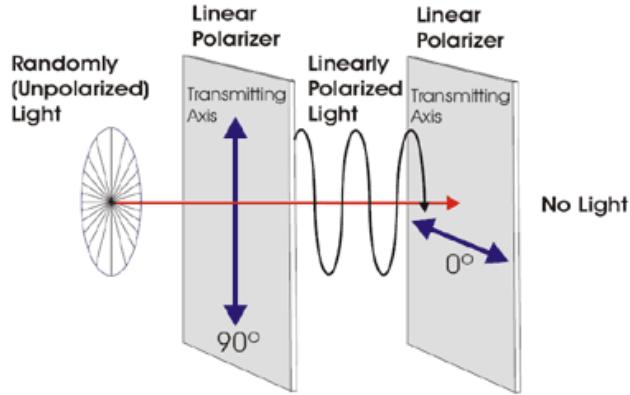


Figure 2.7: Unpolarized light passes through a vertical polarizer → linearly vertically polarized light passes through a horizontal polarizer → no light²



Figure 2.8: Polarizing filter by Nikon³

As the polarization is a vast topic, we do not need to go into further detail for the purposes of this thesis. If the reader wishes to learn more, please refer to a scientific literature, for example Kliger and Lewis [13]

2.2.1 Polarization in rendering

The integration of the polarization in the rendering process is quite rare as only a few scenarios display the effects of the polarization and one must implement quite complex behavior of the radiation waves to the spectral rendering. However, renderers such as Mitsuba2 or ART fully track the polarization state of light when needed. The implementation covered by this section is already in Mitsuba2 [1].

The polarization state is represented by a *Stokes vector* — a 4-dimensional quantity that fully parameterizes the elliptical shape of the light's electric field for each wavelength separately. The information stored in the Stokes vectors is explained in Figure 2.9.

³<https://www.nikonusa.com/en/learn-and-explore/a/tips-and-techniques/polarizing-filters-add-pow-to-pictures.html>

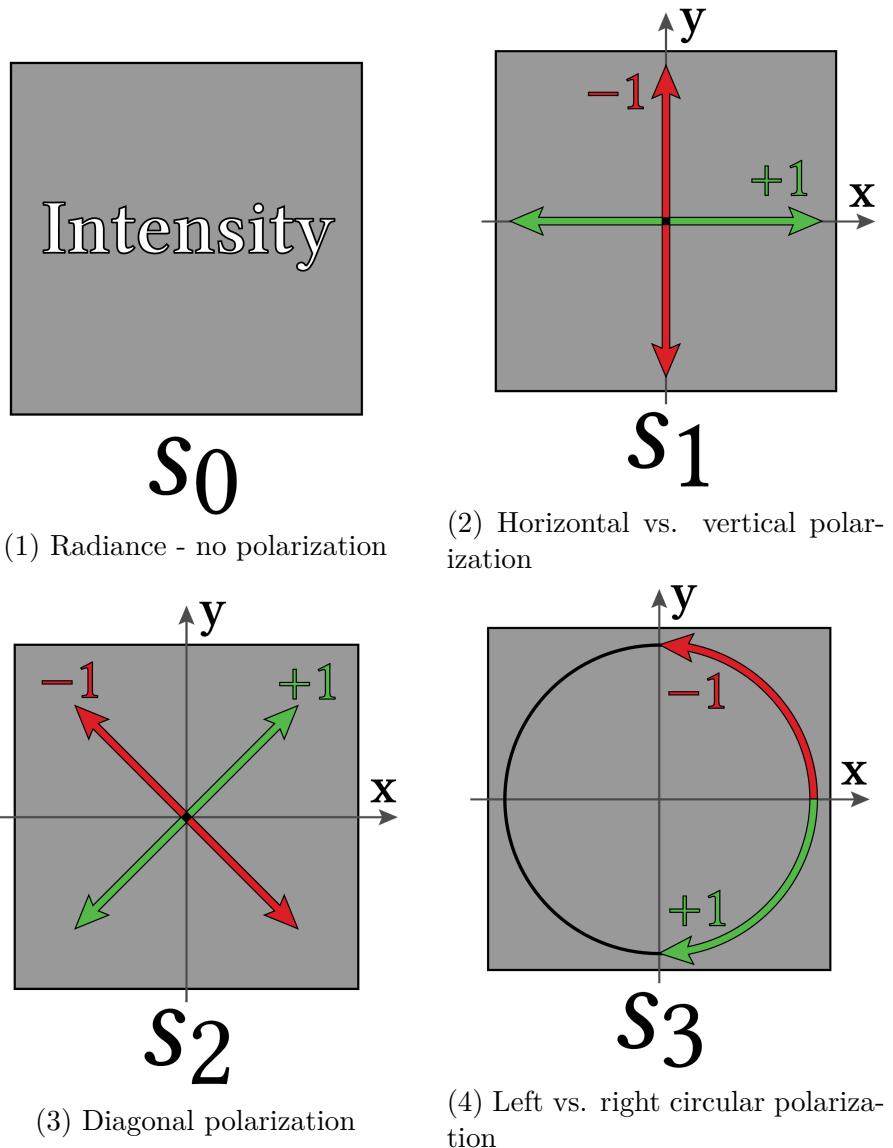


Figure 2.9: Different information carried by the Stokes vector

As we have the polarization states properly represented and we can track them throughout the rendering process, the next step is to determine the affects to these states upon a surface interaction. A transformation between the incoming state and the outgoing state is represented by the *Mueller matrix* $M \in \mathbb{R}^{4x4}$. Due to the adjustments to all interactions in the light transport, we also generalize the BSDF $f_r(\lambda, \omega_i, \omega_o)$ to the polarized pBSDF $M(\lambda, \omega_i, \omega_o)$.

If the reader is curious about the complications this implementation brings and their solutions, he may want to look into the details of Mitsuba2 in Nimier-David et al. [15]. Nevertheless, they are not crucial for the purposes of this thesis and we purposely skip them.

2.3 Dispersion

Generally, the IOR of a dielectric at least slightly varies for different wavelengths(e.g. glass has IOR between 1.5 and 1.6). This causes that the polychromatic light is split into its spectral components upon an intersection with such materials. As each wavelength is slightly shifted, a rainbow effect can be perceived — this phenomenon is called the *dispersion*. It can be frequently seen on rainy days with some sun rays in form of a rainbow or through a dispersive prism 2.10.

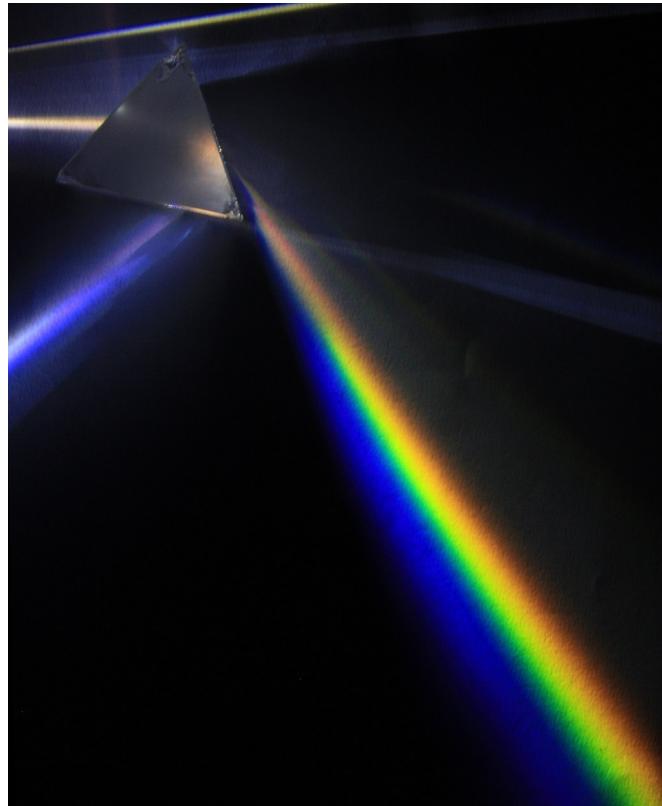


Figure 2.10: Photograph of a dispersive prism⁴

In the computer graphics, even though it is possible to simulate the dispersion in the tristimulus rendering, it is insufficient and the obvious choice would be to

⁴https://en.wikipedia.org/wiki/Dispersive_prism



Figure 2.11: Structural iridescence of the peacock feathers (left) and the thin-film light interference in a soap bubble (right)⁵

use the spectral rendering as it already contains most of the information about the tracking of the wavelengths. After that, the only fundamental thing is the representation of the varying IOR – the currently largely used method is called *Sellmeier approximation* [24].

2.4 Iridescence

It is quite common that some objects in the nature exhibit an interesting behavior where the hue of their surface gradually changes with the viewing angle and the illumination angle, such as butterfly wings, soap bubbles, oil etc. This phenomenon is called *iridescence* or *goniochromism* and is caused by an interference between the light waves, their consequent scattering depending on their wavelength which produces a rapid change in colors [4].

We distinguish two main types of the iridescence:

Microscopic structures Reflections from structures of a size similar to the wavelength (e.g. peacock feathers)

Thin film Light interaction with a thin film of a size similar to the wavelength (e.g. soap bubble)

An example of both can be seen in Figure 2.11.

In this thesis, we focus on the thin-film interference as it is nicely described as a physical process and it is already incorporated in Mitsuba [4]. From now on, by iridescence we mean the thin-film interference until told otherwise.

Let's preview a light interaction at the soap bubble in Figure 2.12. The light strikes at the surface of the film, based on the angle it can be either reflected or transmitted. The transmitted light very quickly strikes the bottom boundary of the soap bubble (as it is very thin) and again can be reflected and/or refracted. As the film is a few hundreds of nanometers thick, this repeats a lot and, as you can see, the light transmitted from the upper boundary can easily interfere with the light reflected from the lower boundary.

An obvious observation is that the iridescence is also dependent on the thickness of the interacting layer - as the thickness increases, the transmission of the

⁵<https://en.wikipedia.org/wiki/Iridescence>

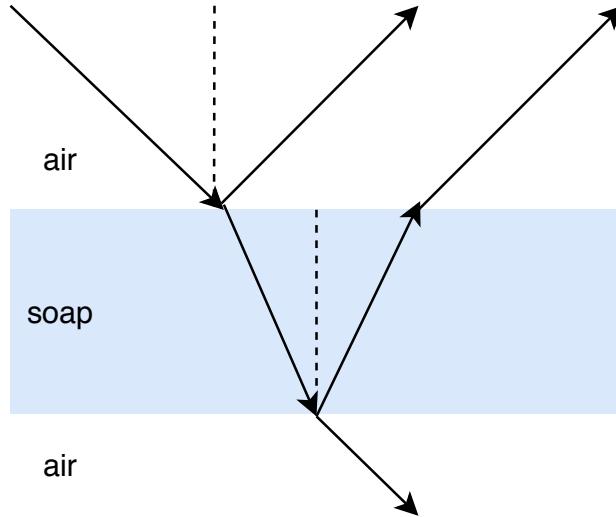


Figure 2.12: A cross-section of a light interaction with the soap bubble

light takes longer time which consequently causes a lot less interferences. The difference between two variously thick films is displayed in Figure 2.13.

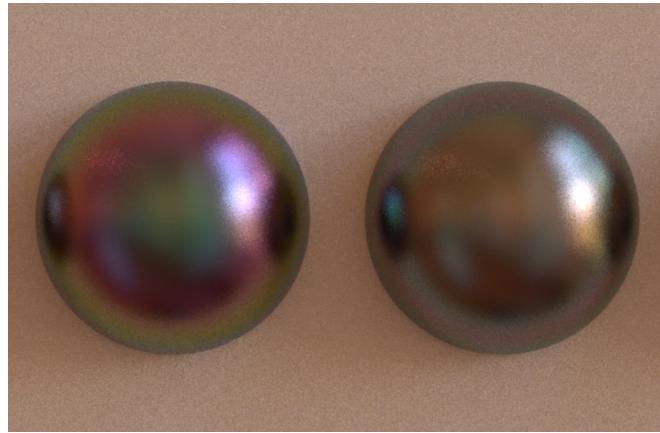


Figure 2.13: Two identical rough conductors with differently thick film layers on top of them: 550nm (left) vs. 1500nm (right) rendered in Mitsuba2

2.4.1 Iridescence in rendering

Based on the Belcour and Barla [4], we overview the computational process of the iridescence caused by a thin film on top of a rough material. We purposely avoid the exact formulations of the equations as these would be unnecessarily complicated to explain as it is sufficient to comprehend the basics in order to evaluate the correctness of the computation. For more details, the interested reader is referred to the Belcour and Barla [4].

Essentially, this procedure computes an iridescence term of the thin film layer that is plugged into the BSDF of the rough base surface:

1. Compute the reflected and the transmitted values of the Fresnel equations for the IOR of the film and the IOR of the exterior

2. Compute the optical paths differences between the primary and the secondary light paths
3. Evaluate the Fresnel phase shift
4. Determine the term by using the Airy summation for the parallel and perpendicular polarization

2.5 Fluorescence

Curiously, certain materials or substances in the nature change their colors with no respect to the illumination color. Such material are called *fluorescent* and, in fact, they are capable of absorbing the electromagnetic radiation of specific wavelengths and emitting back a different one. The most eye-catching fluorescence is caused by the absorption the ultraviolet light which is invisible to the human eye.

3. Benchmark

3.1 Framework

3.2 Supported Spectral Renderers

3.2.1 Mitsuba2

3.2.2 ART

3.3 Scenes

3.3.1 GGX Reflectance

3.3.2 Spectral accuracy

3.3.3 Polarization

3.3.4 Fluorescence

3.3.5 Iridescence

3.4 Common Data

3.5 Usage

3.6 Open source contributions

3.6.1 GGX for ART

3.6.2 Iridescence for Mitsuba2

3.6.3 Multi-channel EXR support for jeri.io

3.7 Future Work

Conclusion

Bibliography

- [1] Mitsuba2 docs. <https://mitsuba2.readthedocs.io/en/latest/>. Accessed: 2020-6-7.
- [2] LE Barbow. International lighting vocabulary. *Journal of the SMPTE*, 73(4):331–332, 1964.
- [3] Petr Beckmann and Andre Spizzichino. The scattering of electromagnetic waves from rough surfaces. *ah*, 1987.
- [4] Laurent Belcour and Pascal Barla. A practical extension to microfacet theory for the modeling of varying iridescence. *ACM Transactions on Graphics (TOG)*, 36(4):1–14, 2017.
- [5] David Brewster. On the laws which regulate the polarisation of light by reflexion from transparent bodies. *Philosophical Transactions of the Royal Society of London*, 105:125–159, 1815.
- [6] Russel E Caflisch et al. Monte carlo and quasi-monte carlo methods. *Acta numerica*, 1998:1–49, 1998.
- [7] Robert L Cook and Kenneth E. Torrance. A reflectance model for computer graphics. *ACM Transactions on Graphics (ToG)*, 1(1):7–24, 1982.
- [8] Bernardt Duvenhage, Kadi Bouatouch, and Derrick G Kourie. Numerical verification of bidirectional reflectance distribution functions for physical plausibility. In *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference*, pages 200–208, 2013.
- [9] Andrew S Glassner. *An introduction to ray tracing*. Elsevier, 1989.
- [10] Wenzel Jakob and Johannes Hanika. A low-dimensional function space for efficient spectral upsampling. In *Computer Graphics Forum*, volume 38, pages 147–155. Wiley Online Library, 2019.
- [11] Henrik Wann Jensen. *Realistic image synthesis using photon mapping*. AK Peters/CRC Press, 2001.
- [12] James T Kajiya. The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 143–150, 1986.
- [13] David S Kliger and James W Lewis. *Polarized light in optics and spectroscopy*. Elsevier, 2012.
- [14] Fred E Nicodemus. Directional reflectance and emissivity of an opaque surface. *Applied optics*, 4(7):767–775, 1965.
- [15] Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. Mitsuba 2: A retargetable forward and inverse renderer. *ACM Transactions on Graphics (TOG)*, 38(6):1–17, 2019.

- [16] Michael Oren and Shree K Nayar. Generalization of lambert's reflectance model. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 239–246, 1994.
- [17] Mark S Peercy. Linear color representations for full speed spectral rendering. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 191–198, 1993.
- [18] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016.
- [19] Francois X Sillion and Claude Peuch. Radiosity & global illumination. 1994.
- [20] DH Sliney. What is light? the visible spectrum and beyond. *Eye*, 30(2):222–229, 2016.
- [21] Kenneth E. Torrance and Ephraim M. Sparrow. Theory for off-specular reflection from roughened surfaces. 1967.
- [22] Bruce Walter, Stephen R Marschner, Hongsong Li, and Kenneth E Torrance. Microfacet models for refraction through rough surfaces. *Rendering techniques*, 2007:18th, 2007.
- [23] Wilkie. Cgg course: Introduction to the color science. <https://cgg.mff.cuni.cz/~wilkie/Website/NPGR025.html>. Accessed: 2020-11-7.
- [24] Kate Devlin1 Alan Chalmers1 Alexander Wilkie and Werner Purgathofer. Tone reproduction and physically based spectral rendering. Eurographics, 2002.
- [25] Gunter Wyszecki and Walter Stanley Stiles. *Color science*, volume 8. Wiley New York, 1982.