Tampere University

Osman Yilmaz

Spectral Ray Tracing for Generation of Spatial Color Constancy Training Data

Master of Science Thesis
Faculty of Information Technology and Communication Sciences
Examiners: Prof. Moncef Gabbouj
Dr. Jarno Nikkanen
October 2022

ABSTRACT

Osman Yilmaz: Spectral Ray Tracing for Generation of Spatial Color Constancy Training Data
Master of Science Thesis
Tampere University
Signal Processing and Machine Learning
October 2022

Computational color constancy is a fundamental step in digital cameras that estimates the chromaticity of illumination. Most of automatic white balance (AWB) algorithms that perform computational color constancy assume that there is a single illuminant in the scene. This widely-known assumption is frequently violated in the real world. It could be argued that the main reason for the assumption of single illuminant comes from the limited amount of available mixed illuminant datasets and the laborious annotation process. Annotation of mixed illuminated images is orders of magnitude more laborious compared to a single illuminant case, due to the spatial complexity that requires pixel-wise ground truth illumination chromaticity in various ratios of existing illuminants.

Spectral ray tracing is a 3D rendering method to create physically realistic images and animations using the spectral representations of materials and light sources rather than a trichromatic representation such as red-green-blue (RGB). In this thesis, this physically correct image signal generation method is used in creation of spatially varying mixed illuminated image dataset with pixel-wise ground truth illumination chromaticity. In complex 3D scenes, materials are defined based on a database of real world spectral reflectance measurements and light sources are defined based on the spectral power distribution definitions that have been released by the International Commission on Illumination (CIE). Rendering is done by using Blender Cycles rendering engine in the visible spectrum wavelengths from 395nm to 705nm with 5nm equal bins resulting in 63 channel full-spectrum image. The resulting full-spectrum images can be turned into the raw response of any camera as long as the spectral sensitivity of the camera module is known. This is a big advantage of spectral ray tracing since color constancy is mostly camera module-dependent. Pixel-wise white balance gain is calculated through the linear average of illuminant chromaticities depending on their contribution to the mixed illuminated raw image. The raw image signal and pixel-wise white balance gain are fundamentally needed in spatial color constancy dataset. This study implements an image generation pipeline that starts from the spectral definitions of illuminants and materials and ends with an sRGB image created from a 3D scene.

6 different 3D Blender scenes are created, each having 7 different virtual cameras located throughout the scene. 406 single illuminated and 1015 spatially varying mixed illuminated images are created including their pixel-wise ground truth illumination chromaticity. Created dataset can be used to improve mixed illumination color constancy algorithms and paves the way for further research and testing in the field.

Keywords: spectral ray tracing, mixed-illumination, illumination estimation, computational color constancy, spatial white balance, full-spectrum image

PREFACE

This thesis was done for Xiaomi Finland Oy as a final step of my M.Sc. studies at Tampere University. I would like to thank for the given opportunity to work and write a thesis in parallel. Big gratitude undoubtedly goes to my supervisors, Prof. Moncef Gabbouj and Dr. Jarno Nikkanen, for guidance during my thesis work. I truly appreciated your inputs on the thesis and path to a successful thesis project. Thank you Janne Kotka, manager in Xiaomi Finland Oy, as well as the AWB team, for your ideas and thoughts on the topic. I am grateful to all Xiaomi employees who gave input for the work and participated in the development of supportive tools, thank you. The journey I started in sRGB went back to the physics of light and turned back to sRGB. It undoubtedly was a colorful one.

I would like to thank my friends; Taner, Alper, Asim, Mehmet, Atakan, Duygu, Baran, Pragya, Umit, Gani, Emin, Avishek, Mert, Baran, Egemen, Ozcereyan, Burak, Elif, Sandeep for making this journey unforgettable with all the wonderful times we shared together. Although there were miles between us with some of you, the support was from the heart and was felt.

Beloved Zsofia Knetl, I deeply appreciate your support throughout this journey. You were there for me, each and every time. I will never forget it.

My deepest gratitude is to my parents, Hasan and Nursen Yilmaz, and my brother Tayyib. Their unwavering love, support, and sacrifice have always been a source of motivation for me. This, of course, includes my bigger family in Ayas, Mersin. Without them, I would not be the person I am today.

Tampere, 19th October 2022

Osman Yilmaz

CONTENTS

1. Introduction 1
1.1 Objective of the thesis 3
1.2 Structure of the thesis 4
2. Theoretical Background 5
2.1 Physically based rendering 5
2.1.1 Radiometry 5
2.1.2 Light transportation 6
2.1.3 Material surfaces 8
2.1.4 Rendering equation 10
2.1.5 Spectral ray tracing 12
2.2 Colors as spectra in an image 14
2.2.1 Light sources 16
2.2.2 Human visual system 20
2.2.3 Color in digital cameras 23
2.3 Computational color constancy algorithms 30
2.3.1 Gray world 31
2.3.2 Max-RGB 31
2.3.3 Shades of gray 32
2.3.4 Gray search algorithms 32
2.3.5 LSMI U-net 33
2.3.6 Mixed illumination white balance 34
3. Proposed Method 35
3.1 Spectral image generation 36
3.2 Ground truth white balance gain map calculation 39
4. Implementation 42
4.1 Spectral ray tracing with Blender Cycles 42
4.2 Guided filter 44
4.3 Auto exposure control 45
5. Validation and Testing 49
5.1 Validation of spectral ray tracing 49
5.2 Testing dataset images 53
6. Conclusion and Future Work 56
References 58
Appendix A: Validation results of spectral ray tracing 63

Appendix B: DE2000 grid 69

LIST OF FIGURES

1.1 Visual comparison of taking a photograph in real life and rendering a scene in computer graphics. . . . 1
1.2 A simple mixed illuminated scene to visualize the problem statement. . . . 3

2.1 A simple scene showing the possible behaviour of a ray after the interaction with a material surface or volume. . . . 7
2.2 Example micro-structure of 2 different surfaces. . . . 9
2.3 The color triangle. . . . 15
2.4 The electromagnetic spectrum and radiation types with a special care on visible spectrum. . . . 16
2.5 Relative spectral power distribution of some of the light sources in the CIE library. . . . 17
2.6 Relative response of the color matching functions x, y and z. . . . 18
2.7 CIE 1931 chromaticity diagram according to 2 degree standard observer with planckian locus and illuminants of Figure 2.5. . . . 19
2.8 Schematic cross section of the human eye, obtained from [36]. . . . 21
2.9 Relative spectral sensitivity of the rod (scotopic) and the cone (photopic). . 22
2.10 Relative spectral sensitivities of the L M and S cones of the HVS. . . . 22
2.11 A camera cross section and an example Bayer pattern. . . . 24
2.12 Relative camera module spectral sensitivity of the Canon 5DSR, Sony IMX135 and Nikon D810. . . . 25
2.13 A set of camera algorithms run on ISP, obtained from [7]. . . . 26
2.14 Example Canon 5DSR raw image from Intel-TAU dataset [18]. . . . 27
2.15 CCM applied version of the Figure 2.14 (b). . . . 28
2.16 An example spatially varying multi illuminant scene. . . . 29
2.17 An example mixed illuminated image from the LSMI dataset with corresponding illuminant coefficient map. . . . 33

3.1 Used pipeline to generate images via spectral ray tracing. . . . 36
3.2 Spectral image generation of a color checker under D65 illuminant. . . . 38
3.3 Example gain map of a mixed illuminated image. . . . 40

4.1 Relative R/G B/G error with and without the guided filter. . . . 45
4.2 Edge aware smoothing effect of the guided filter. . . . 46
4.3 AEC algorithm applied on a high contrast scene and the change in histogram. 47
4.4 Example mixed illuminated scenes generated with spectral ray tracing method. 48

5.1 Conducted spectral measurements in the lab that is necessary for the validation experiment. 50
5.2 Reference laboratory image and generated image with spectral ray tracing. 51
5.3 Predictions of the learning-based algorithms. 55

B.1 DE2000 grid error visualization for the prediction of Afifi et al. in Figure 5.3. 70
B.2 DE2000 grid error visualization for the prediction of LSMI U-net in Figure 5.3. 71

LIST OF TABLES

5.1 Comparison of reference color patches to generated color patches under illumination D65. 52
5.2 DE2000 average for each scene as well as number of color patches between the error limits. 53

A.1 Comparison of reference color patches to generated color patches under illumination U30. 63
A.2 Comparison of reference color patches to generated color patches under illumination CWF. 64
A.3 Comparison of reference color patches to generated color patches under illumination TL84. 65
A.4 Comparison of reference color patches to generated color patches under illumination A. 66
A.5 Comparison of reference color patches to generated color patches under illumination H2. 67

LIST OF SYMBOLS AND ABBREVIATIONS

AEC	Automatic Exposure Control
AF	Automatic Focus
AI	Artificial Intelligence
API	Application Programming Interface
AWB	Automatic White Balance
BRDF	Bidirectional Reflectance Distribution Function
BSDF	Bidirectional Scattering Distribution Function
BTDF	Bidirectional Transmittance Distribution Function
CCC	Computational Color Constancy
CCD	Charge-Coupled Device
CCM	Color Correction Matrix
CCT	Correlated Color Temperature
CFA	Color Filter Array
CIE	International Commission on Illumination
CMOS	Complementary Metal Oxide Semiconductor
DSLR	Digital Single-Lens Reflex
GBCE	Global Brightness and Contrast Enhancement
HSV	Hue Saturation Value
HVS	Human Visual System
ISP	Image Signal Processor
LSMI	Large Scale Multi Illuminant
PBR	Physically Based Rendering
RGB	Red Green Blue
SPD	Spectral Power Distribution
WP	White Point

1. INTRODUCTION

Rendering is the process of generating a 2D visualization of the described 3D scene. It is comparable to taking a picture, where the described 3D scene is the real world and the visualized 2D output is the photograph [1]. In photography, the small hole of camera with lens captures the light in the environment bouncing off objects in the scene and reflecting it onto film or sensor, inverted. In computer graphics rendering, the center of virtual camera represents the lens, and the virtual screen represents the sensor or film. The light passes from the center of the virtual camera, interacts with the objects in the environment, and reflects it on the virtual screen with no inversion, as shown in Figure 1.1.

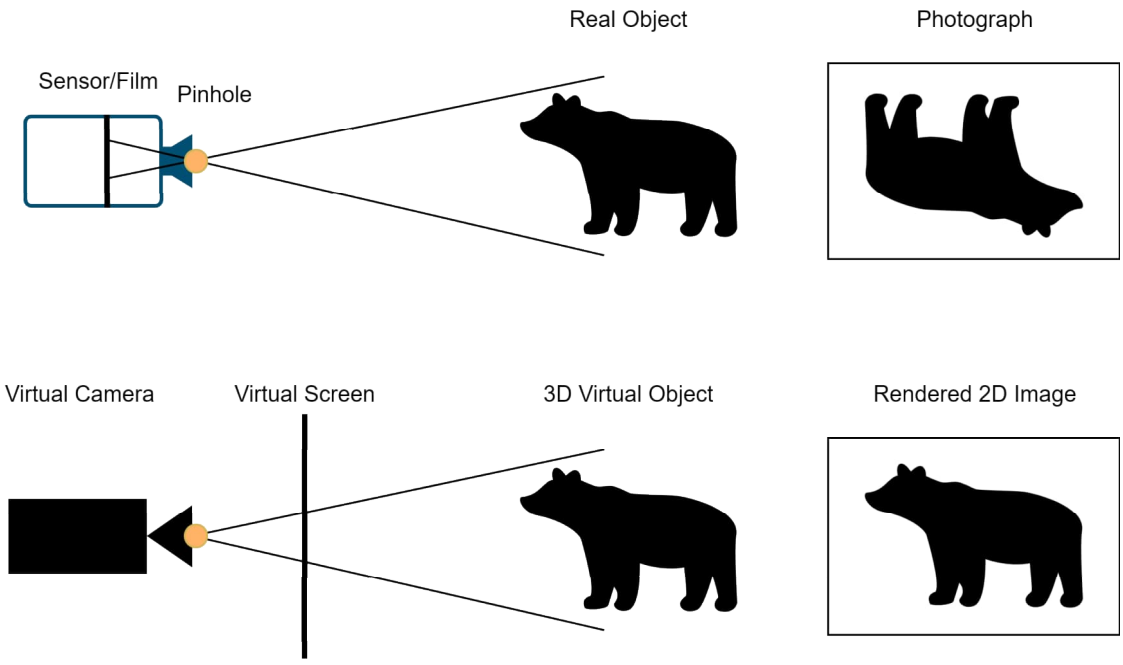


Figure 1.1. *Visual comparison of taking a photograph in real life and rendering a scene in computer graphics.*

Ray tracing is one of the fundamental techniques of photo-realistic rendering in computer graphics. As its name explains, the ray-tracing rendering technique sends rays for each pixel in the virtual image plane and simulates the intersections of that ray while it is going through or bouncing through objects in the scene to determine the pixel value. The pixel value is calculated by shading computation of intersection point, surface normal, and

other characteristics of objects in the scene [2], [1]. Although ray tracing algorithms create photo-realistic scenes with objects, shade, and intersections, it is insufficient for accurate color reproduction. It is due to modeling light, material, and all other color characteristics of objects as trichromatic values such as RGB. However, this trichromatic definition of color is missing some physical realities of the real world, such as chromatic light dispersion, metamerism, fluorescence, polarization, and colored scattering [3],[4]. Spectral ray tracing solves it by defining the color as a spectral distribution in the visible spectrum. Once the material surface is defined as spectral reflectance and light with spectral power distribution, generated image mimics the physical realities of the world [1]. The downside is that spectral ray tracing is computationally expensive and time-consuming. In this thesis work, offline rendering will be enough as long as the created pixel values project the physically correct real world.

Color constancy is an ability of the Human Visual System (HVS) that allows the color of the same object to appear relatively constant regardless of the scene's illumination. Computational color constancy (CCC) is a fundamental step in today's digital cameras that estimates the chromaticity of the illumination in the scene and prevents the color casts by eliminating the effects of illumination on color reproduction [5], [6]. Other color reproductions and color correction steps in digital photography are built on top of the response from color constancy algorithms. Therefore, the failure of the color constancy algorithm leads to undesirable color casts as well as unrealistic color reproduction [7]. For instance, in a fully color constant system, an achromatic white or gray object is expected to be observed as the same color under cool clear blue sky or warm candle flame illumination. Color constancy algorithms compensate for the addition of illumination on that achromatic surface and make them appear as if it was taken under neutral illumination. The term color constancy algorithm is used interchangeably with automatic white balancing (AWB) algorithm in this thesis work.

The majority of the white balance algorithms that perform computational color constancy assume that there is a single illuminant in the scene [8], [9], [10], [11], [12], [13], [14], [15]. This is a widely known assumption that is frequently violated in the real world [16], [17]. A simple example scene can be a room with windows where indoor room light and outdoor light coming through windows mix. The assumption of single illuminant comes from the limited amount of mixed illuminant datasets and annotation process that is orders of magnitude more laborious compared to a single illuminant case. The laborious annotation process is due to gradual or sharp illumination transitions and the necessity of pixel-wise annotation. White balance algorithms that assume a single illuminant in the scene try to find an undefined balance between illuminants when they are tested on a mixed-illuminant scene. But in a fully color constant system, pixel-wise white point prediction is required to compensate for the color casts caused by the illuminants. Pixel-wise white point prediction is also important for color reproduction of mixed illumination to be controlled in a

well-defined manner, instead of it happening in an undefined and uncontrolled manner. A simple example showcasing the described problem of estimating a single illuminant and proposed ground truth can be seen in Figure 1.2

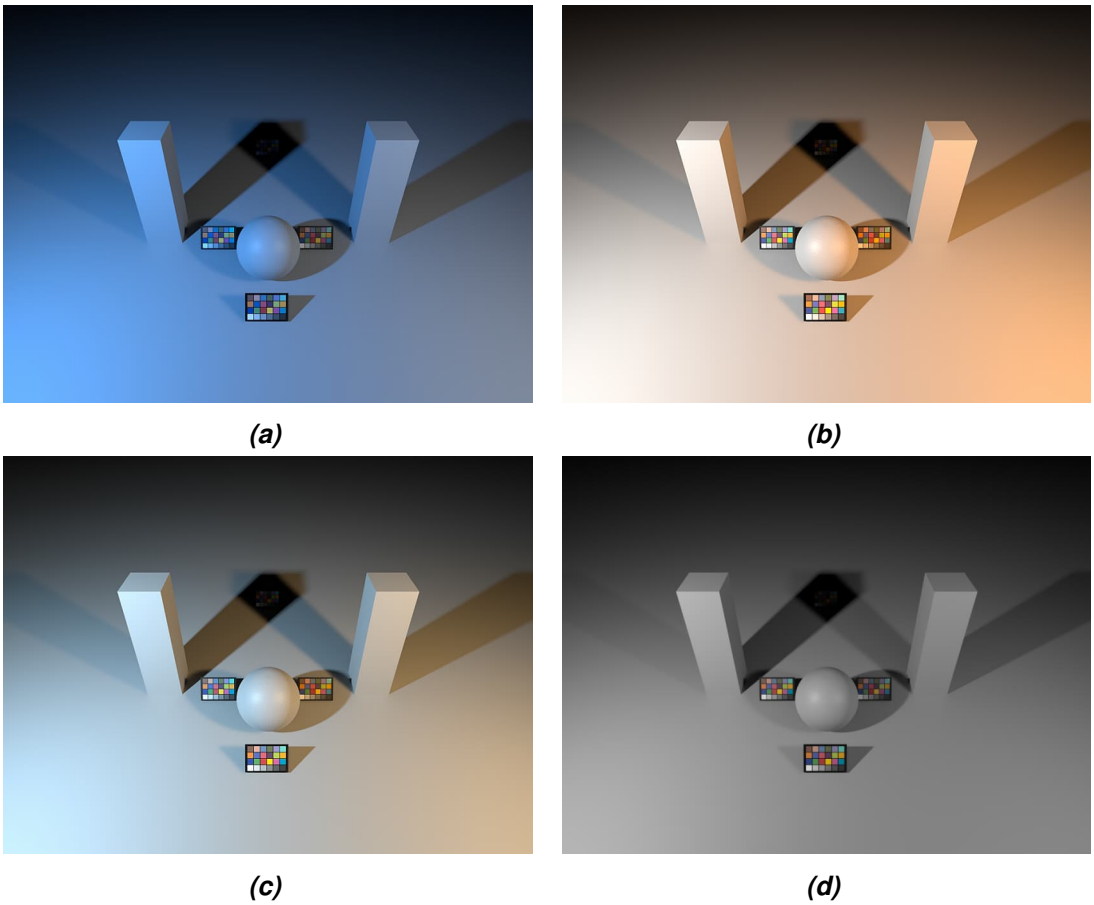


Figure 1.2. *A simple mixed illuminated scene to visualize the problem statement. The ground surface, rectangular prism, and sphere are defined as near achromatic gray objects, and 4 Color Checker Classics are placed in the scene. There are 2 light sources, a filtered tungsten halogen "D65" on the left and incandescent "A" on the right. (a) is an sRGB image gained with the white point of incandescent "A" illumination for the whole scene. (b) is an sRGB image gained with the white point of illumination "D65" for the whole scene. (c) is an sRGB image gained with the weighted average of 2 white points used for the whole scene. (d) is an sRGB image gained by pixel-wise white point calculated from the linear combination of 2 light sources.*

1.1 Objective of the thesis

The objective of this thesis is to create a spatial color constancy training dataset with spectral ray tracing so that algorithms can be developed to solve mixed illumination color constancy. It aims to create photo-realistic raw images with 2 illuminations in the scene and the corresponding ground-truth white balance map. An image generation pipeline is proposed in this thesis that can be used for creating a spatial color constancy dataset. Basically, the main objectives are:

1. Do spectral ray tracing using the Blender Cycles rendering engine where material surfaces and illuminants are defined by their spectral representations.
2. Create full spectrum images and convert them to be the raw response of any camera as long as the camera sensor spectral response is known. This thesis uses Canon 5DSR, Sony IMX135, and Nikon D810 since their camera sensor spectral response is publicly available [18].
3. Normalize the image brightness by pushing the histogram in order to compensate the dim areas in the image.
4. Apply domain transform filter to do edge-aware smoothing while improving pixel quality to eliminate noise caused by relatively low number of ray samples per-pixel in rendering.
5. Calculate ground truth white balance gain for each pixel. Calculate color correction matrix (CCM) for each pixel to convert the image from camera RGB to device-independent sRGB color space.
6. Validate spectral ray tracing technique by comparing a laboratory scene to its spectrally created equivalent through measured spectral data.
7. Test a few images from the created dataset on publicly available pixel-wise white balance algorithms.

1.2 Structure of the thesis

This thesis is organized as follows. Chapter 2 covers the physically based rendering, color understanding from the spectral perspective, and computational color constancy algorithms. Chapter 3 explains the spectral image generation method as well as ground truth white balance gain map calculation. Chapter 4 describes the implementation of the method with the help of the Blender Cycles rendering engine and supplementary algorithms used in the image generation pipeline. It is followed by the validation of the proposed method and testing of the generated images on open source algorithms in Chapter 5. Finally, Chapter 6 concludes the main findings as well as a discussion on the possible future studies.

2. THEORETICAL BACKGROUND

This chapter explains the general context of the main topics in this thesis work. First, Section 2.1 introduces the physically based rendering approach, its equation, and spectral ray tracing implementation details. Then, Section 2.2 explains colors in an image from the spectral point of view. Next, Section 2.3 gives an overview of some of the color constancy algorithms and their working principles.

2.1 Physically based rendering

Physically based rendering (PBR) is a widely used technique to create photo-realistic images where an efficient light transportation simulation is performed. A common and simple algorithm that performs realistic light transportation is called ray tracing. Ray tracing keeps track of rays of light while it is encountering objects in a realistic manner so that reflections, detailed shadows, and ambient occlusions can be simulated [19] [20] [21]. The term photo-realistic or indistinguishable from reality is not well-defined and differs between observes, and context [2]. In this thesis work, it is used as the physically correct color reproduction of the real world.

The virtual scene that the physically based renderer is trying to realistically simulate consists of several objects and phenomenons such as cameras, different geometries, materials, light sources, ray-object intersections, indirect light transport, surface reflectance, transmittance, scattering, and ray propagation [1]. Light transportation in these complex scenes is modeled by the intersection with materials at various angles. Each light ray going through these interactions in high-dimensional space has certain properties like hit points, throughput, radiance, and path length. This complex behavior of each light ray for each pixel on the virtual scene comes down to numerical approximations, statistical and stochastic samplings, variance, and bias control in order to work with the computing power available today [22].

2.1.1 Radiometry

Radiometry focuses on the ideas and mathematical tools to define electromagnetic radiation and radiometric domains widely used in computer graphics due to light being a part of

the electromagnetic spectrum. The light transport is built on the radiometry formulations and units. [1], [23]. Important radiometric quantities are radiant power, irradiance, and radiance. *Radiant power* is the total energy passing through a finite surface or volume in space as $S \subset \mathbb{R}^3$ as per unit time and can be formulated as

$$\Phi(t) = \frac{dQ(t)}{dt} \tag{2.1}$$

where Q is radiant energy in joules $[J]$, radiant power is measured in watts as $[W = J.s^{-1}]$. *Irradiance* is radiant power per unit surface area and formulated as

$$E(p) = \frac{d\Phi(p)}{dA(p)} \tag{2.2}$$

with unit $[W.m^{-2}]$. Irradiance is always defined for a point p on the surface S and generally defines incoming radiance on the surface only. If a light ray is scattering through a surface or transmitting through it, it is defined as *radiant exitance*. *Radiance* as the most important quantity for light transport and PBR, defines the irradiance and radiant exitance according to solid light ray angles and formulated as

$$L(p, w) = \frac{d^2\Phi(p, w)}{dA(p)d\sigma_p^{\perp}(w)} \tag{2.3}$$

where $d\sigma(w)$ is the solid angle around w that is perpendicular to p. Radiance calculates the flux density per unit time passing through a small surface per unit solid angle and uses $[W.m^2.sr^{-1}]$ as unit [23] [1].

Out of all the radiometric quantities explained here, radiance will be the one that is widely used in the rest of the thesis when explaining light transportation and PBR. This is due to radiance being a principle quantity in radiometry, and many other terms can be computed with integrals over unit spheres and directions. In addition, radiance stays constant along light ray transporting through empty space, which makes it suitable for ray tracing computations [1].

2.1.2 Light transportation

In PBR, light is equivalent to a ray, and it carries information related to the objects in the scene [22]. Transported information by the ray can be a wavelength as in spectral rendering or a color representation in a color-space, like red-green-blue (RGB) or Hue Saturation Value (HSV) in a trichromatic manner. In addition, the ray travels in the scene by interacting with the existing surface or volume. The material surface specialties determine the behavior of the ray after an interaction. A simple scenario of a light ray and its

possible behavior is shown in Figure 2.1.

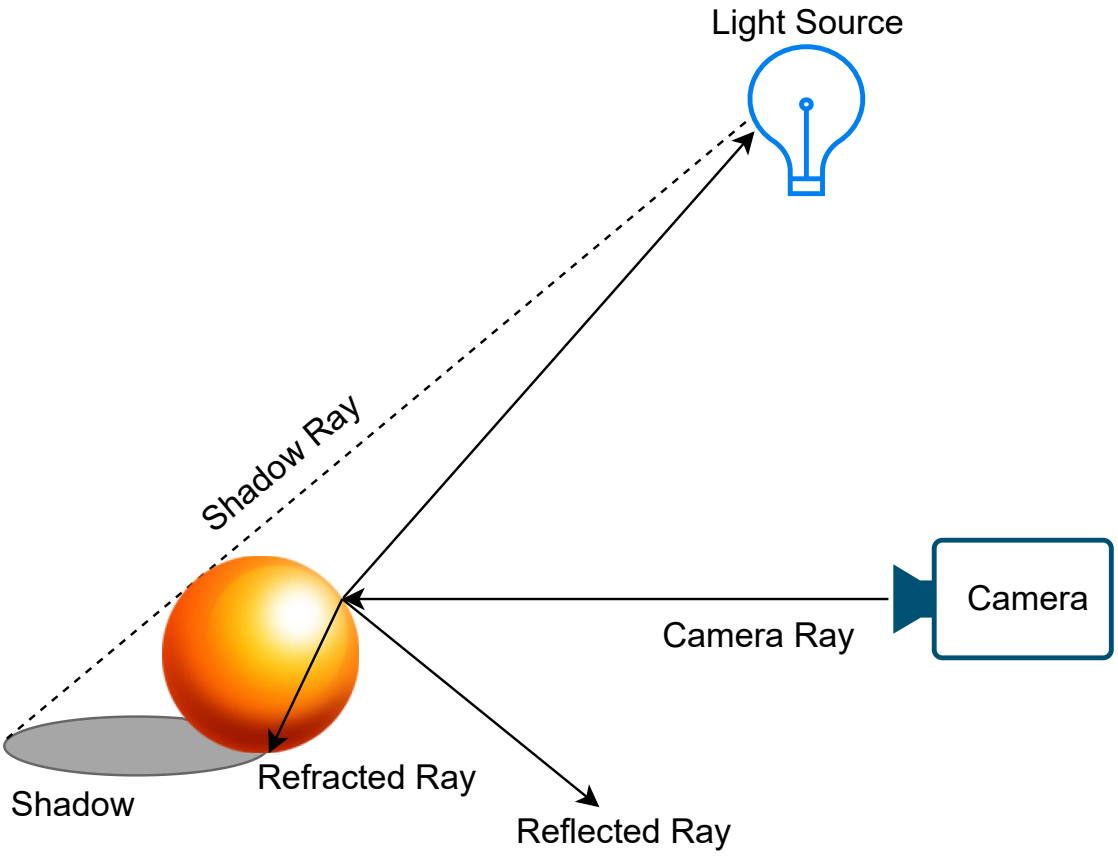


Figure 2.1. *A simple scene showing the possible behaviour of a ray after the interaction with a material surface or volume.*

The behavior of a light ray, when interacted with material surfaces or volume in a 3D scene, differs in a variety of ways. Depending on the material and its surface specifications, a light ray can be partly absorbed, partly reflected, and even partly strayed. In addition to the material and surface structure, the direction of the light ray and incident angle affect the behavior of the ray after the interaction. Example behaviors of a light ray after an interaction with a material surface can be reflection, refraction, transmission, absorption, emission etc. [1], [2], [24]. For instance, a green leaf partially absorbs and partially reflects the incoming light ray while a glass refracts the light. In addition to these behaviors, the intensity of the outgoing light ray differs with the incident angle and material surface. The Fresnel effect explains that if a light ray hits an opaque surface with a low incident angle, the majority of the ray is almost perfectly reflected, while it would normally reflect with diffuse angles in a uniform way with no specific direction [1]. In Figure 2.1, a light ray is sent from the camera, reflected and refracted on the surface of the ball, and the rays that reach the light source will define the color of that specific pixel. In rendering applications, multiple rays need to be sent for the same pixel in order to have a reliable and less noisy image.

When a light ray is incident on a material surface or volume, that surface scatters the light and reflects some of it. Modeling of this reflection needs to define the spectral distribution of the reflected ray and its outgoing direction distribution. These distributions are calculated with bidirectional reflectance distribution function (BRDF) and bidirectional transmittance distribution function (BTDF). BRDF formulates the radiance of the light ray coming from the light source and leaving the surface towards the camera by considering the incident angle of the incoming and outgoing light. BTDF formulates the transmitted light from a material surface or volume in a similar manner as in BRDF. [1], [24], [22], [25]. These distribution functions can be calculated by measurements on real material surfaces and used in rendering [1]. In addition to the measurements, artificial specialties can be added to the distribution functions of materials such as mirror reflections, subsurface scattering, glossiness, etc. [22], [26].

Bidirectionality in BRDF and BTDF is the indication of having the same pixel value on the virtual screen of the rendering output, both following a single light path from the virtual camera to the light source or vice versa. Bidirectionality characteristic of distribution functions is significant from the point of evaluating rendering equation due to being able to follow through the path of light in a reserved order [1], [26]. However, from the convergence point of view, following a path of ray from the virtual screen to the light source is more beneficial. This is due to the possibility of a light ray leaving the light source, scattering around with objects in the 3D scene but not ending up in the virtual screen, therefore not contributing to the final color of the virtual pixel [1], [22].

2.1.3 Material surfaces

The BRDF and BTDF introduced earlier focus on only part of the rendering equation since they describe how a light ray is reflected or transmitted on a surface. In addition to the scattering of the light ray on the surface, the renderer needs to know which BRDFs and BTDFs exist at that specific location on that surface and their parameters [1]. Material surface parameters such as metalness and roughness directly affect the scattering and absorption of the light ray when it interacts with a material surface or volume. Nevertheless, for a precise prediction of a ray of light when it hits a material surface, there need to be complex distribution functions to determine the statistical behavior of the light ray [26].

Photo-realistic PBR needs precise surface representation to generate the effect of the specific material characteristics of the substance and the geometry of the surface. While the substance of the material defines the reflected and refracted amount of hitting light rays, the geometry of the material surface defines the direction of the reflected and refracted light rays. Any surface geometry is not perfectly flat, considering the atoms of the surface geometry. These non-visible to the naked eye surface irregularities are called *microgeometry* [22]. Microgeometry affects the surface normal and changes the reflec-

tion and refraction direction of the light ray. A single virtual pixel consists of aggregated several reflected light rays in the direction of each surface microgeometry.

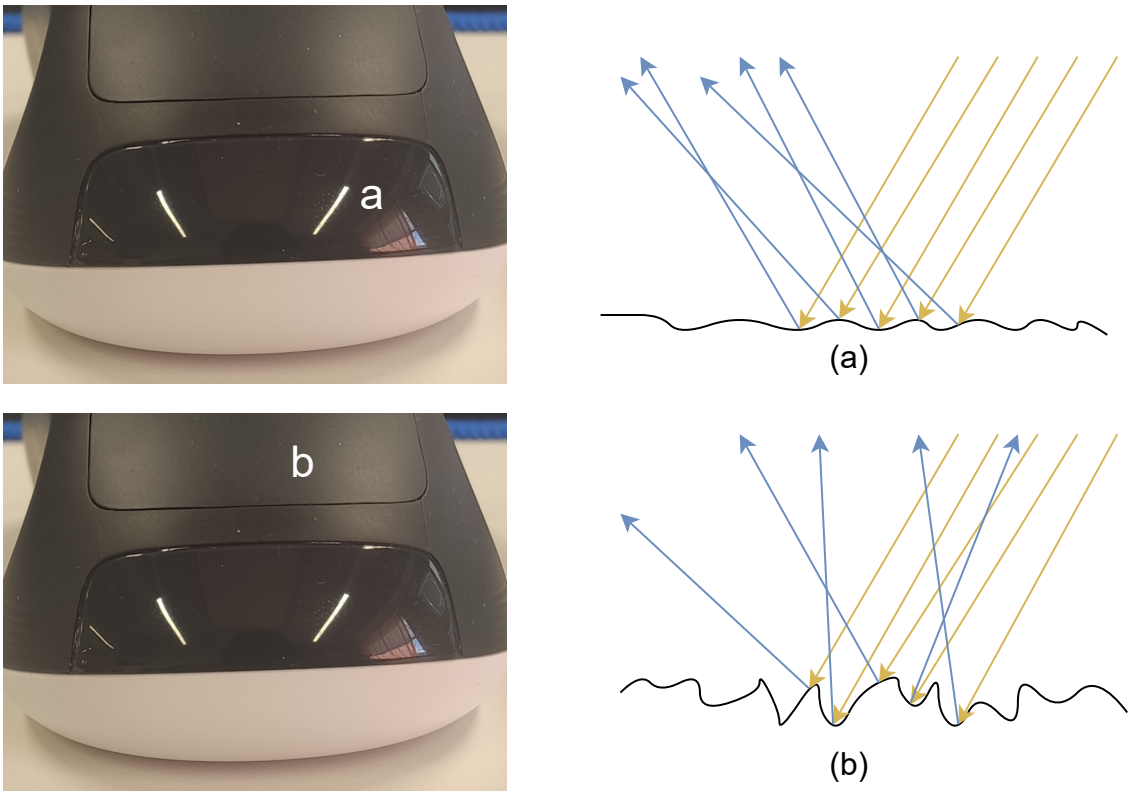


Figure 2.2. *Example micro-structure of 2 different surfaces. On the left is a photo of a computer mouse, and on the right is the microscopic surface structure of 2 different locations on the mouse. The surface point defined with "a" has a slightly rough microgeometry and the light rays hitting the surface are somewhat reflected in the same directions. This causes the reflection of fluorescent light bulbs and window to be reflected. On the other hand, the surface point "b" has a much rougher micro-structure; thus, light rays are reflected in different directions. This spread of light rays after the hit causes blurrier reflections. The example is adapted from [22].*

Material surfaces or volumes have complex characteristics that are defined with a combination of BRDFs and BTDFs. Bidirectional scattering distribution function (BSDF) handles this complexity with handling both BRDF and BTDF depending on the surface characteristics [1], [22]. The scattering function handles light reflection on the hemisphere to the direction of the surface normal. On the contrary, transmission is handled with light distribution inside the surface of the hemisphere but to the opposite side of the surface normal. These intrinsically different two distribution functions ease the process of light transportation through different surfaces and volumes. In addition to the surfaces that are defined by using both of them, it is still possible to use a single distribution function for a surface. For instance, an opaque material surface defined with BSDF will exclude the BTDF since it does not let any light transmit through its surface.

2.1.4 Rendering equation

PBR approach started to get more attention in the computer graphics industry around the 1980s with the paper of Whitted et al. on ray tracing, where he simulated the global lighting effects [19]. The accurate light transportation approach of Whitted generated significantly different images with unseen realistic details. It could simulate the contribution of directly incoming light rays while ignoring most of the indirect light rays, therefore able to perform perfect reflection and refraction. Kajiya and Immel et al. improved Whitted's ray tracing, defined the rendering equation with independent papers, and introduced path tracing [27], [28]. Light transport integral equation of Kajiya et al. with Monte Carlo integral opened the way to simulate realistic optical phenomenons [27]. Taking into consideration the various direct and indirect light sources for each point on the surface, it can render scenes with complex lighting environments. Considering light paths on a surface point of all possible directions in a 3D scene requires integral over the whole scene surface. The rendering equation is as follows:

$$L_o(p, w_o) = L_e(p, w_o) + \int_{S^2} f(p, w_o, w_i) L_i(p, w_i) |\cos \theta_i| dw_i \tag{2.4}$$

where:

p	is the point on the surface microgeometry where light scatters
w_i	is the incident light direction
w_o	is the outgoing light direction
$L_o(p, w_o)$	is the outgoing light radiance to direction w_o
$L_e(p, w_o)$	is the emitted light radiance to direction w_o
$L_i(p, w_i)$	is the incoming light radiance from direction w_i
$f(p, w_o, w_i)$	is the BSDF of incident light radiance from all directions around point p
θ_i	is the angle between incoming light and surface normal
S^2	is unit sphere of both inside and outside of the surface centered at the origin

The rendering equation can be supplemented with a simple visual representation in Figure 2.1. Evaluation point p is the intersection point of light ray on the ball, and the L_o outgoing light radiance to the virtual camera pixel is calculated with the BSDF and incoming light radiance L_i from the light source.

In order to better analyze the participating components of the rendering equation, it is important to understand the recursive characteristic of $L_i(p, w_i)$. The evaluation of incoming radiance at point p is connected with an outgoing radiance at another point p' in the scene due to evaluating light coming from every direction in the unit sphere. It leads to $L_i(p, w_i) = L_o(p', -w_i)$, incoming radiance from direction w_i equal to the outgoing

light radiance from direction $-w_i$. In a recursive way, $L_o(p', -w_i)$ now becomes the next rendering equation that needs to be solved in the same manner. By considering this, the rendering equation can also be written as:

$$L_o(p, w_o) = L_e(p, w_o) + \int_{S^2} f(p, w_o, w_i) L_o(p', -w_i) |\cos \theta_i| dw_i \tag{2.5}$$

The energy balance of the 3D scene is the key factor in evolving the equation. The key difference of Equation 2.5 is that there is now only one parameter of interest, which is the outgoing radiance from points on material surfaces. It is still not easy to solve since it exists on both sides of the rendering equation due to the recursive structure of the scattering of light. The recursion stops at one point of L_o, where, for all the directions of w, there is no interaction point on a surface. In that condition, only emitting radiance L_e needs to be evaluated. But, since the integral goes over all possible directions on the unit sphere, it is mostly the case that there exists an intersection point that causes continuity in recursive behavior. This generally leads to the analytical evaluation of the rendering equation to be not possible [1],[26].

One way to analytically solve the recursive equations is the Monte Carlo method. Monte Carlo methods were discovered after World War II and named after the city of Monaco, famous for its casinos. They are used for solving repeated random sampling equations to have a numerical result, like the rendering Equation 2.4. For example, consider the integral

$$J = \int_{\Omega} f(x) d\mu(x) \tag{2.6}$$

where Ω is an n-dimensional unit hypercube, $f : \Omega \to \mathbb{R}$ is a function with a real numerical output, and μ is a measurement unit on Ω such as length, volume, area or angle. Evaluation of this integral using random sampling is done by independently sampling N points over the Ω domain like $X_1, ..., X_N$ with a *probability density function* p. It results as

$$\widehat{J} = \frac{1}{N} \sum_{i=1}^{N} \frac{F(X_i)}{p(X_i)} \tag{2.7}$$

where $\widehat{J}$ being a variable depending on the number of chosen points on Ω domain [23], [29]. Monte Carlo integration is simple and powerful over multi-dimensional integrals of unbound functions since it converges in the rate of $O(N^{-1/2})$. The nature of the Monte Carlo approach to random sampling handles infinite-dimensional equations, which is the case in rendering equations since the light coming from all directions on the unit sphere

needs to be evaluated [23]. The adoption of Monte Carlo on rendering equation is shown in Equation 2.8.

$$L_o(p, w_o) = L_e(p, w_o) + \frac{1}{N}\sum_{j=1}^{N}\frac{f(p, w_o, w_j)L_i(p, w_j)|\cos\theta_j|)}{p(w_j)} \tag{2.8}$$

The convergence of the rendering equation will contribute to the virtual pixel value on the virtual screen for that specific 3D scene. If $\overline{X} = X_0, ..., X_{N-1}$ is all the transport paths from light source to the virtual camera, the radiance I entering a single virtual pixel k can be written as in Equation 2.9.

$$\widehat{I_k} = \frac{1}{N}\sum_{j=1}^{N}\frac{L(\overline{X}_j)}{p(\overline{X}_j)} \tag{2.9}$$

This path-space integral solution to the rendering equation has a more simplified approach to understanding the addition of each light path to the virtual screen pixel value [23].

2.1.5 Spectral ray tracing

PBR aims to generate photo-realistic images with the Monte Carlo light transport algorithm as in equation 2.9. The majority of the render engines use trichromatic representations for modeling light sources, material surfaces, and all other necessary color definitions. Trichromatic representation is mainly done with RGB values due to its design for displays. However, for a physically true representation of color appearance, the trichromatic approach is not sufficient [30]. Advanced light phenomenons like chromatic light dispersion, metamerism, fluorescence, polarization, and colored scattering are not possible to simulate with trichromatic rendering. However, spectral rendering has the ability to solve these problems by representing scene material surfaces and illumination in the whole visible spectrum of light but with extra computational expense [3],[4], [31], [32].

Trichromatic rendering with RGB or HSV has no justification for producing a physically correct output image signal. The wide usage comes from the hardware requirements of real-time rendering and effective screening of rendered scenes on a monitor, as well as a simple color-picking ability by users. RGB primaries are not spectrally defined but rather in terms of CIE 1931 xy chromaticity coordinate [33]. It creates a problem with the final color appearance due to metamerism. Metamerism is a phenomenon of two objects with matching color appearances under one illumination but not matching once the illumination changes. This wavelength-dependent phenomenon is more visible with spiky and not smooth illuminant spectra. Considering light with smooth illuminant spectra on smooth

surface reflectance, RGB rendering does not cause considerable inaccuracy. On the other hand, metamerism occurs almost always in fluorescent light sources with spiky illumination spectra. In addition, considering the complex scenes with several different light sources scattering through objects, even light with smooth illuminant spectra goes colorful [32], [34], [35].

Another visually distinguishable difference resulting from rendering in a trichromatic manner is chromatic light dispersion. When a near achromatic light goes through a dispersive prism, it splits into wavelength-dependent colors with different angles. In trichromatic rendering, only 3 colors are visible after the dispersive prism. Differently, all spectral colors are visible like a rainbow with spectral rendering due to the availability of the whole visible spectrum. Chromatic light dispersion is visible if the scene has objects behind a glass and light needs to go through it. With a little blurring of the glass, resulted scene can be convincing even with trichromatic rendering, but for actual physical reality, spectral rendering is needed [32].

Rendering equation evaluation with spectral ray tracing technique requires the illumination and surface reflectance to be defined with a spectral wavelength λ. In the case of the material surface, *spectral reflectance* of the material surface $S(\lambda)$ needs to be defined. Spectral reflectance is the ratio of spectral radiance reflected by a surface to the received amount. Spectral radiance L_λ can be formulated with the help of Equation 2.3,

$$L_\lambda(p, w, \lambda) = \frac{d^3\Phi(p, w, \lambda)}{dA(p)d\sigma_p^\perp(w)d\lambda} \tag{2.10}$$

where $L_\lambda = dL/d\lambda$. Spectral radiance is a fundamental radiometric quantity that many other quantities can be obtained from and its unit is $[W.m^{-2}.sr^{-1}.nm^{-1}]$ [23]. The nanometer in the unit is used to explain wavelengths for spectral variables. The illumination in the scene for a spectral ray tracing needs to be defined with relative *spectral power distribution* (SPD). Spectral power can be formulated from Equation 2.1 in a similar manner to Equation 2.10 as

$$\Phi(\lambda) = \frac{d\Phi}{d\lambda} \tag{2.11}$$

With spectral reflectance and SPD being part of the rendering equation, the spectral rendering equation can be derived with the help of Equation 2.4 as

$$L_o(p, w_o, \lambda) = L_e(p, w_o, \lambda) + \int_\Lambda \int_{S^2} f(p, w_o, w_i, \lambda) L_o(p, w_i, \lambda) |\cos\theta_i| dw_i d\lambda \tag{2.12}$$

where Λ denotes the additional spectral domain of wavelengths. Priorly, it is shown that Veach et al. further improved the rendering equation to path-space integral solution as in Equation 2.9 [23]. The same solution can be applied to the spectral path-space integral problem. The spectral path-space integral problem can be formulated as in Equation 2.13

$$I_k = \int_{\Lambda} \int_{S^2} L_k(\overline{X}, \lambda) d\mu(\overline{X}) d\lambda \tag{2.13}$$

where path-space of all possible light transport paths $\overline{X}$ carrying the wavelength-dependent information λ to contribute to the final virtual pixel value k [23]. Monte Carlo solution of path-space integral for all possible light paths is

$$\widehat{I_k} = \frac{1}{N} \sum_{j=1}^{N} \frac{L_k(\overline{X}_j, \lambda_j)}{p(\overline{X}_j, \lambda_j)} \tag{2.14}$$

This equation will be the fundamental basis of doing psychically based rendering in this thesis. It renders the scene 1 wavelength at a time which causes heavy computational burden [34]. If the whole visible spectrum is represented with $\lambda^n = [\lambda^{380}, \lambda^{385}, ..., \lambda^{780}]$, Monte Carlo path tracing to reach a full spectrum image can be written as in Equation 2.15.

$$\widehat{I_k}(\lambda^n) = \frac{1}{N} \sum_{j=1}^{N} \frac{L_k(\overline{X}_j, \lambda_j^n)}{p(\overline{X}_j, \lambda_j^n)} \tag{2.15}$$

Here, the sampling of the wavelength is 5nm, but this can differ depending on the application area. For example, It can be linearly sampled with 7nm per bin as in *Mitsuba* [3]. Nevertheless, the important point in selecting the bin size is not to miss the spiky areas of certain material spectral reflectances and some illuminant SPDs, such as fluorescent. This thesis uses the range of 395nm until 705nm with 5nm equal bins for the spectral representations in the visible spectrum.

2.2 Colors as spectra in an image

Definition of color is not straightforward and requires further explanation of the condition and observer. Color is an attribute of visual perception consisting of any combination of chromatic and achromatic content according to *International Lighting Vocabulary* [5]. Color perception can be impacted by a spectral distribution of light, reflectance of material, and combination of both under a certain condition. It is certain that the definition of color requires its spectral representation, for example, in the visible spectrum if it is considered for the human visual system. The description of color in this thesis is based on a

color triangle, as shown in Figure 2.3. Color appears as a result of the interaction of an object surface, light source, and an observer visual system, such as an eye and a brain. Initially, electromagnetic energy is needed to start a sensory vision response. This energy will be transformed with the physical and chemical characteristics of an object once interacted and will be later imaged by the photo-receptors of the human visual system to create the perception of color. The bidirectional arrow between the visual system and light source is due to the light source itself having a color appearance. The other bidirectional arrow between the visual system and the object indicates the possible variations in color appearance of an object due to the previous knowledge of the visual system regarding the object and environmental surroundings where the appearance occurs [5],[36], [6].

Figure 2.3. *The color triangle. Colors are perceived thanks to the interaction of a light source, volume or material surface and a visual system like human visual system. Obtained from [5].*

The triangle of color in Figure 2.3 requires spectral representation of each party in order to properly represent the physical reality. The object surfaces are represented with spectral reflectance, and it is explained in Section 2.1.5 with its use case in PBR. Light sources are represented with their relative SPD and human visual system is represented with spectral sensitivity of the eye [5], [36], [7]. These will be further explained to better understand color as a spectral representation. In addition, color in digital cameras will be the match between the PBR and physically based color production.

2.2.1 Light sources

Light sources are a fundamental part of the color triangle due to providing electromagnetic energy for vision to happen. Electromagnetic energy exists in different wavelengths and energy amounts which creates the whole electromagnetic spectrum. The human observable part of the electromagnetic spectrum is called the visible spectrum, and it is described by its wavelength in nanometer. Figure 2.4 shows the radiation types that are part of the electromagnetic spectrum with special care for the visible spectrum. Radiation that the human visual system is able to see is called *light*. Therefore the visible spectrum can also be named visible light. The narrow structure of the visible light is due to the relative sensitivity of the human eye consisting between 380nm to 780nm. The hue human eye recognizes in the visible spectrum starting from the shorter wavelengths is illustrated by violet, blue, green, yellow, orange, and red, respectively. It is important to note that hue like magenta, which is the mixture of both ends of the visible spectrum, blue and red, is not possible to see in the visible spectrum [36].

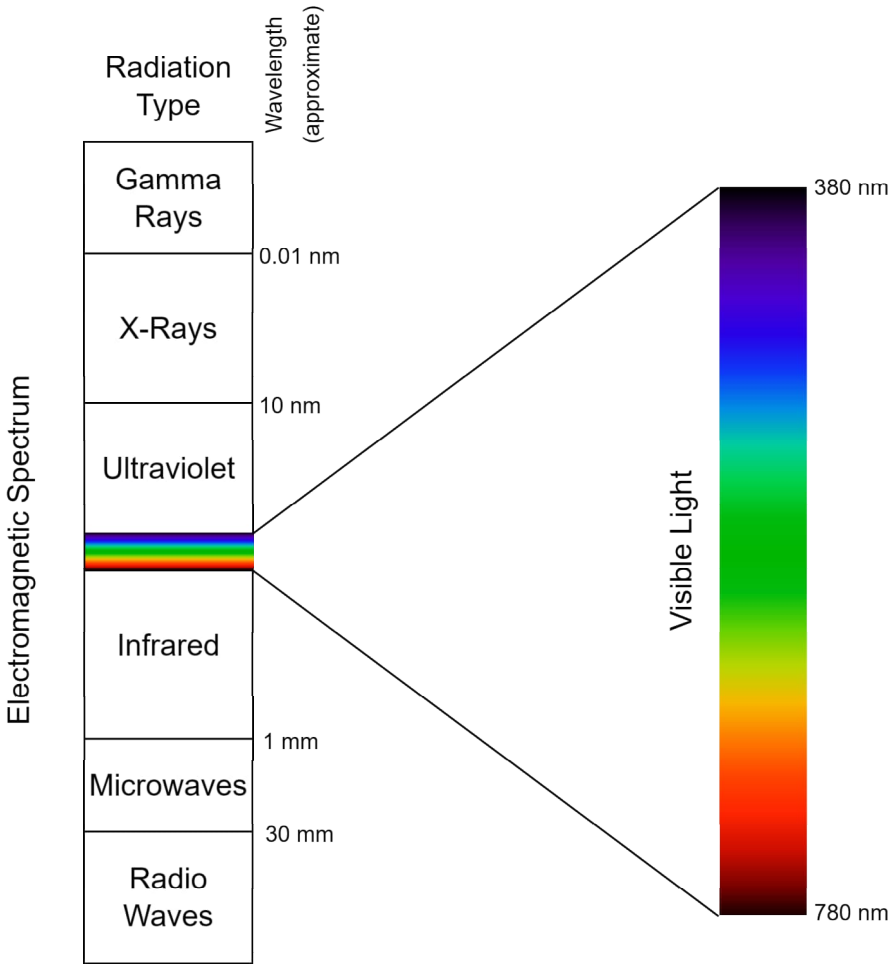


Figure 2.4. *The electromagnetic spectrum and radiation types with a special care on visible spectrum.*

Sources of light emit visible energy at different wavelengths, except monochromator.

Such example light sources can be sun, sky, fluorescent tubes, and solid-state lamps. Due to emitting visible energy at different wavelengths, light sources are defined by their spectral power distribution (SPD) throughout the electromagnetic spectrum. Since power levels of light sources may differ dramatically, SPDs are usually normalized at 560nm and called relative SPD without a unit. Figure 2.5 illustrates relative SPDs of 4 different light sources. It is evident to say that the distribution of energy varies for different light sources. However, it is important to note that relative SPDs do not represent the intensity of the light source but rather their color properties [5].

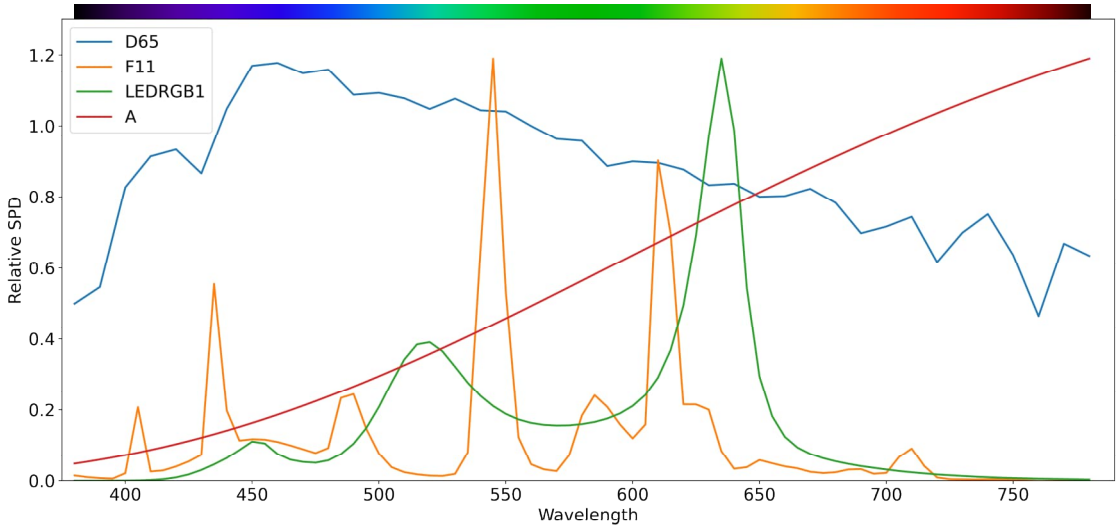


Figure 2.5. *Relative spectral power distribution of some of the light sources in the CIE library.*

Some typical light sources are standardized by International Commission on Illumination (CIE) to describe color [33]. These standards or statistical representations of spectral distributions of light sources are called *illuminants* by CIE. In addition to the SPD, these standard illuminants can be represented by CIE 1931 xy chromaticity coordinate. These coordinates are calculated from CIE 2 degree standard observer tri-stimulus values. X, Y and, Z values are obtained according to standard illuminant E as in Equation 2.16. λ refers to the value at the specific wavelength, $\Delta\lambda$ is the sampled spectrum step size, and k is a normalization factor. $\overline{x}_\lambda$,$\overline{y}_\lambda$ and, $\overline{z}_\lambda$ are representations of color vision of a normal person in the visible spectrum according to 1931 CIE 2 degree standard observer as in Figure 2.6. Once the X, Y, and Z values are calculated with equal energy illuminant E, CIE 1931 xy chromaticity values can be calculated as in Equation 2.17 [33].

$$
\begin{aligned}
X &= k \sum_\lambda E_\lambda \overline{x}_\lambda \Delta\lambda \\
Y &= k \sum_\lambda E_\lambda \overline{y}_\lambda \Delta\lambda \qquad k = \frac{100}{\sum_\lambda E_\lambda \overline{y}_\lambda \Delta\lambda} \\
Z &= k \sum_\lambda E_\lambda \overline{z}_\lambda \Delta\lambda
\end{aligned}
\tag{2.16}
$$

$$x = \frac{X}{X+Y+Z} \quad y = \frac{Y}{X+Y+Z} \tag{2.17}$$

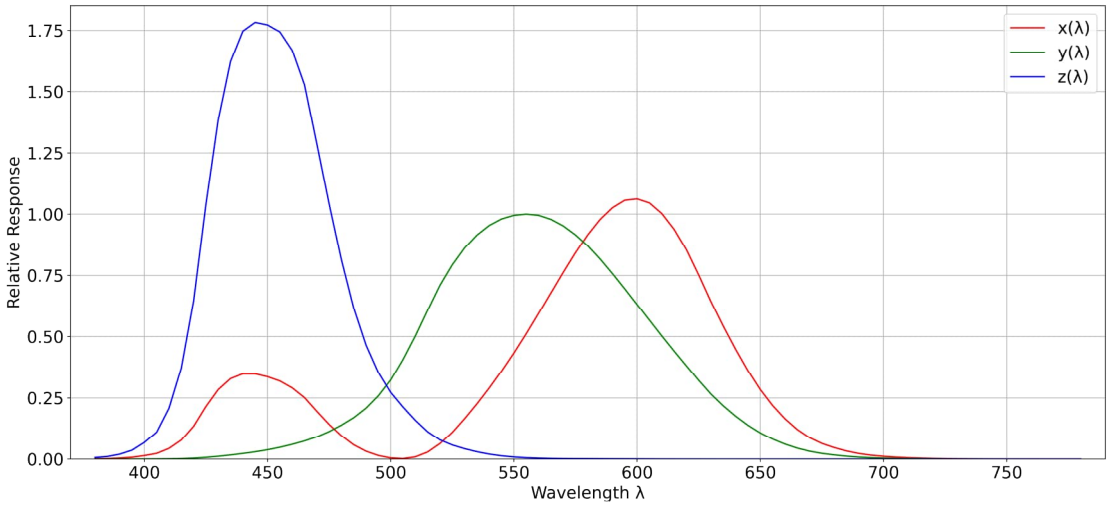


Figure 2.6. *Relative response of the color matching functions x, y and z.*

Another important radiometric quantity that can be used in the definition of a light source is called *color temperature*. It is based on the theoretical *black-body radiator*, or *Planckian radiator* light source, which is a perfect emitter of energy, and the only dependent factor is the temperature. Emitted energy in a Planckian radiator increases as the temperature increases. Planck's equation defines the SPD of a black-body radiator depending on the temperature in *Kelvin* (K). Therefore, if the temperature is known, the SPD of the theoretical Planckian light source can be known as well as its color. However, due to the black-body radiator mainly being a theoretical representation in the laboratories, color temperature is not well generalizing to the real-world light sources. A more generalizing term based on color temperature but applicable to real-world light sources is called *correlated color temperature* (CCT). CCT of a light source is the closest color temperature representation of the specific light source to the black-body radiator since they will have almost the same color as the black-body source. Examples of well-known light sources with their location in the 1931 xy chromaticity plane and Planckian locus are shown in Figure 2.7. For instance, incandescent light source A with CCT at 2856K is a Planckian radiator. Typical fluorescent tube like F11 is at 4000K, average daylight D65 is at 6500K, and mixing of red, green, and blue led light LED-RGB1 is at 2840K.

There are several different groups of light sources that are standardized by the CIE. One of the fundamental light sources of everyday life is daylight. Daylight can be either direct sunlight or sunlight that scatters below the horizon through the atmosphere. CIE defined a methodology to calculate daylight SPDs dependent on the correlated color temperature between 4000K and 25000K. The function defines the daylight locus, and these standard illuminants are called CIE D-illuminants. Most known examples like D50 at 5000K and

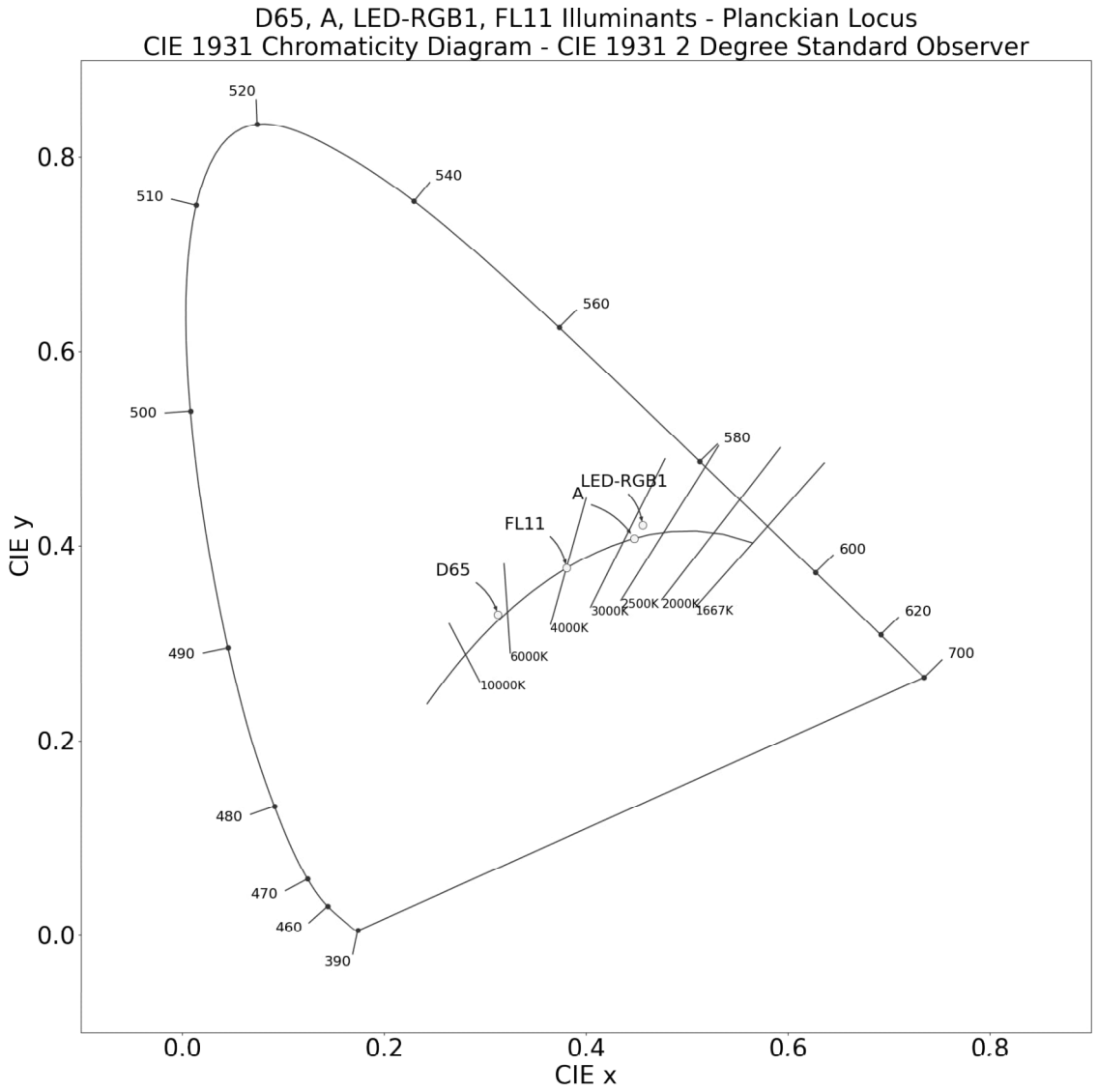


Figure 2.7. *CIE 1931 chromaticity diagram according to 2 degree standard observer with planckian locus and illuminants of Figure 2.5.*

D65 at 6500K defines average daylight SPD while D75 is north sky daylight with 7500K [33], [37]. CIE A illuminant is a Planckian radiator at 2856K, and it is widely used in experiments and research if it is related to incandescent illumination. CIE F illuminants determine the SPDs of several fluorescent light sources. The F-series consist of common illuminants from F1 to F6, broadband illuminants from F7 to F9, and three-band illuminants from F10 to F12. CIE E is an equal energy illuminant where relative SPD of all wavelengths are the same. CIE LED illuminants consist of several types of solid-state lamps such as LED-B1 to LED-B5 phosphor-based solid-state lamps, LED-BH1 hybrid type solid-state light, LED-RGB1, and LED-RGB2 mixing of red, green, and blue solid-state lamps and LED-V1 and LED-V2 violet-pumped solid state lamps. These standard illuminants are widely used in colorimetry and color appearance research as they will be used in this thesis also [36], [6], [33]. 32 different CIE illuminants are utilized between

2700K and 25000K in this thesis. Some of the relative SPDs of defined illuminants are shown in Figure 2.5 as an example representation, while more can be reached through CIE [33].

2.2.2 Human visual system

The last piece of color triangle is the Human visual system (HVS). HVS is a complex structure that combines physiology, optics, chemical signals, detectors, and neural processing of the brain to turn received electromagnetic energy into a particular color. This transition of the physical stimulus of energy to color is highly correlated with the anatomy of the human eye.

A schematic representation of human eye with key-labeled structures for vision to happen is shown in Figure 2.8 [36]. *Cornea* is a vital part of image formation and plays an important role in the refraction capability of the human eye due to its curved shape. Hence, a refractive error in the eye is handled with laser surgery to reshape the cornea. The liquid filling the eyeball between the cornea and lens is called *aqueous humor*. The *iris* muscle controls the pupil size in order to adjust the amount of light that enters the eye. Pigment of iris also defines the eye color of each individual. The *lens* is a flexible and layered structure that modulates focusing depending on the distance with the help of muscles around it. In the case of focusing on a nearby object, the lens changes in a hilly shape to increase the optical ability and gets flatter if the focus is to the distance. Lens also acts as a yellow filter, and its yellowness increases as the age get older. The *vitreous humor* is a liquid that fills behind the lens, and both vitreous and aqueous humor affects the image quality formed by the eye. Once the light energy that forms an image is through the front of the eye, it reaches *retina*. The retina is a thin layer located at the back of the eye, and it contains photosensitive cells and initial image signal processing and transmission circuitry. These cells are connected to the central nervous system and, therefore, part of the brain. The *fovea* is part of the retina, and it is a vital area for the best spatial and color vision. The two degrees of visual angle is covered by the fovea. The *macula* is a yellow filter that protects fovea from intense short-wavelength energy. Most of the diversity in color vision between individuals is due to the various optical densities of the lens, and the macula [36], [5].

There are two different classes of retinal photosensitive cells called *rods* and *cones*. They are named after their shapes since rods are long and slender while cones are canonical. But a more significant difference between them is their visual function. Rods are good at low luminance, while cones are good at higher luminance levels. Capability of the human eye to have vision in diverse sets of luminance levels comes from the ability to transit between rod and cone vision. In low-level luminance environments where only rods are active in vision is called *scotopic vision*. In environments with high luminance levels,

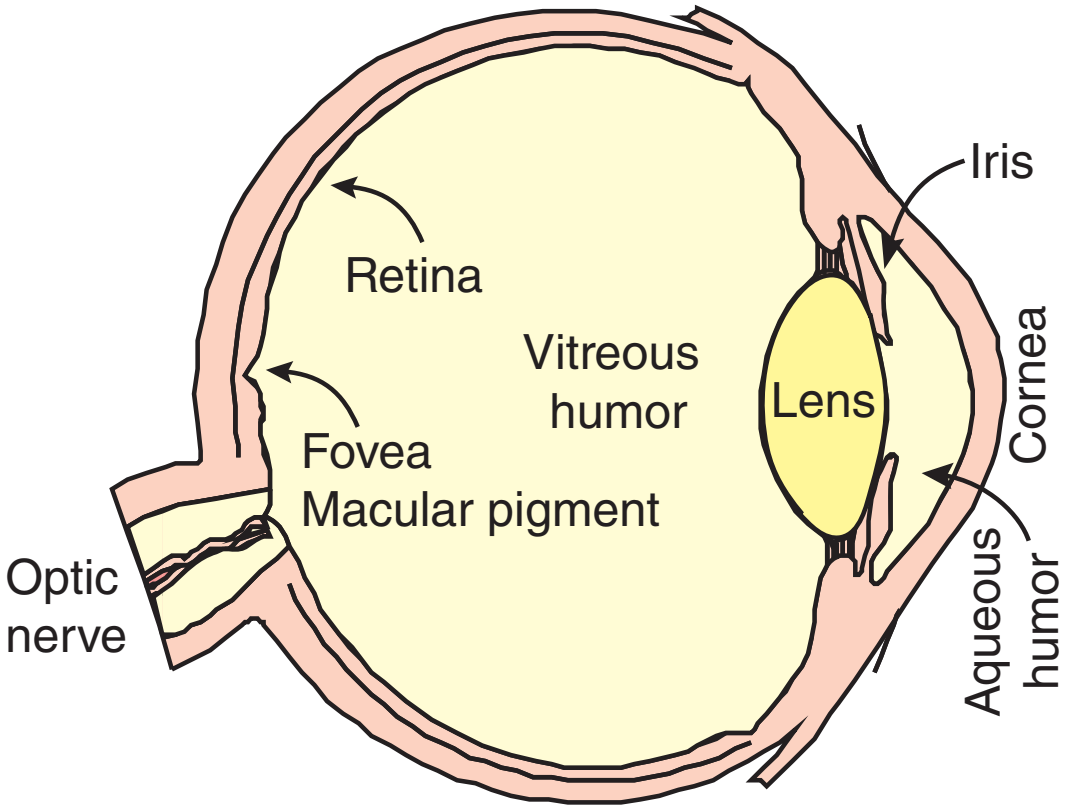


Figure 2.8. *Schematic cross section of the human eye, obtained from [36].*

rods get saturated, and cones create color vision alone, and it is called *photopic vision*. It is possible that in mid-level luminance environments, both rods and cones support the vision, and it is called *mesopic vision*. The relative spectral sensitivity of scotopic and photopic vision is shown in Figure 2.9. There is only 1 type of rod receptor, and therefore scotopic vision relative spectral sensitivity is the same as spectral sensitivity of rod photoreceptor. Its peak spectral response is around 510nm and it is same as CIE spectral luminous efficiency function [5],[36],[33].

Color vision in HVS is enabled by cone photoreceptors, and there are 3 types of cones, with each having a different spectral sensitivity response. These cones are called L, M, and S, referring to the long, medium, and short wavelength sensitive cones, respectively. The relative spectral sensitivity of the 3 types of cone cells is shown in Figure 2.10. The photopic vision relative spectral sensitivity in Figure 2.9 is the representation of the combination of 3 types of cone photoreceptors. It is evident to see that cone spectral sensitivities overlap, and it improves color discrimination. In the color vision of HVS, cone photo-receptors reduce the entire spectrum of incident light interacting with the object spectral reflectances into three signals L, M, and S. This trichromaticy can be formulated as in Equation 2.18 considering a Lambertian surface reflectance where geometrical factors like view angle do not matter. The $\Phi(\lambda)$ refers to relative SPD of the illuminant, $S(\lambda)$

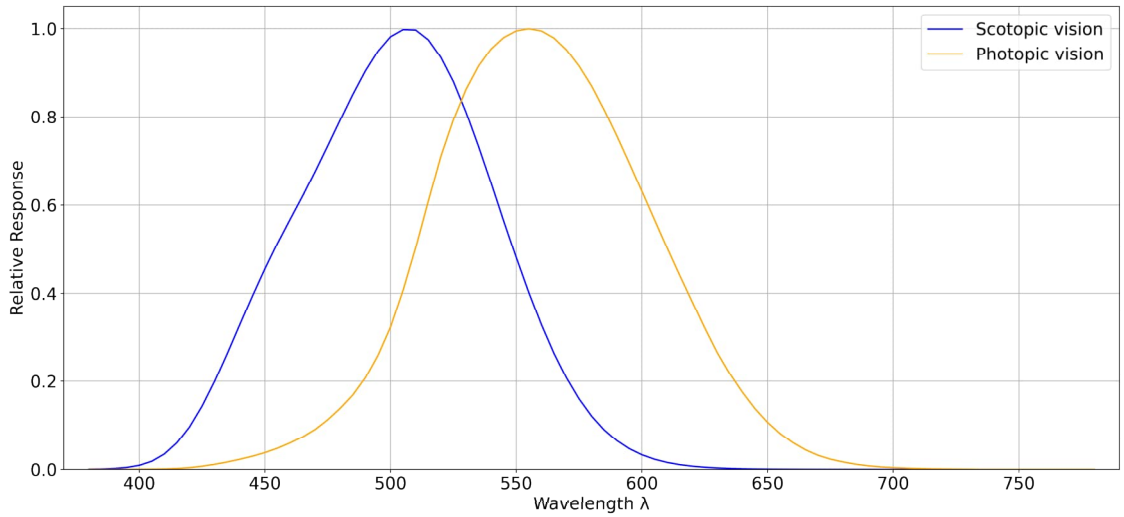


Figure 2.9. *Relative spectral sensitivity of the rod (scotopic) and the cone (photopic).*

refers to spectral reflectance of the material surface and $l(\lambda), m(\lambda)$ and $s(\lambda)$ refers to spectral sensitivity of the cone photoreceptors. It is possible for metamerism to occur due to trichromaticity as explained in Section 2.1.5 [5], [6], [36].

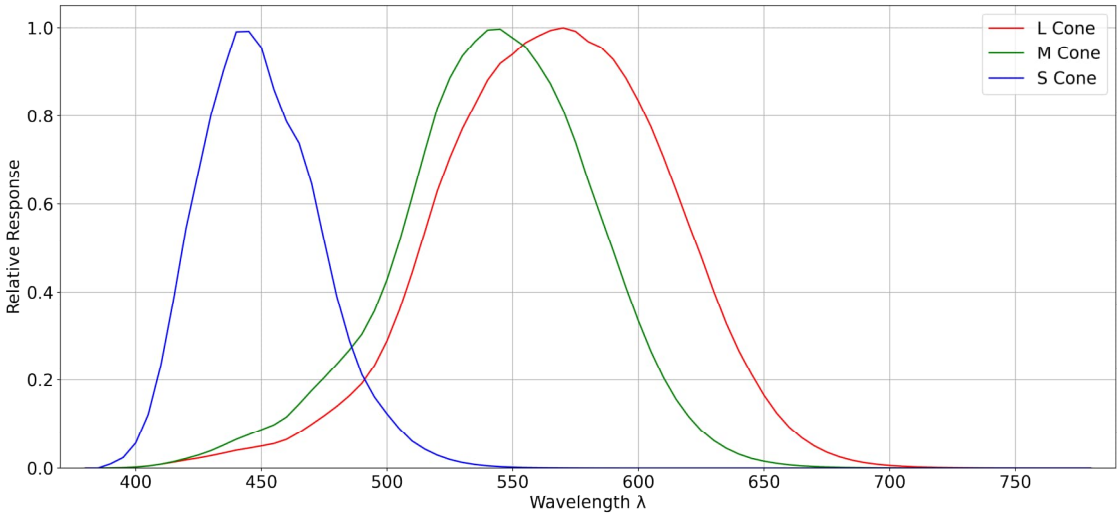


Figure 2.10. *Relative spectral sensitivities of the L M and S cones of the HVS.*

$$L = \int_{\Lambda} \Phi(\lambda) S(\lambda) l(\lambda) d\lambda \qquad M = \int_{\Lambda} \Phi(\lambda) S(\lambda) m(\lambda) d\lambda \qquad S = \int_{\Lambda} \Phi(\lambda) S(\lambda) s(\lambda) d\lambda \tag{2.18}$$

The main subject of this thesis, *color constancy* is originated from the HVS. In everyday life, there are various light sources that the human eye encounters, and some of them are explained in Section 2.2.1. Although the illuminance level and color of the light change throughout the day, the color of the skin and watch, for instance, stay relatively constant. This is due to the color constancy ability of the human eye because it can discount the

effect of a variety of illuminations and adapt to the changes in the scene. This ability of the HVS is thanks to the sensory and cognitive capability of the human eye and brain [5], [36]. However, HVS is not fully color constant, and it can be observed with various color evaluations [38]. For instance, surfaces near wood fire or candlelight appear to have a yellowish cast. This and several other adaptations of HVS is further researched by *color appearance models* [5].

2.2.3 Color in digital cameras

In a simple manner, the color triangle shown in Figure 2.3 can be completed with a camera as well as a human brain and a visual system. Color production and pixel creation in digital cameras are complex like HVS and require various mechanisms to work together. Image creation in digital cameras uses optics, camera sensor, and several algorithms that run on a processor such as Image Signal Processor (ISP) in order to transform the sensor output signal into visually plausible images. There are structural similarities between the HVS and digital camera, such as the cornea and lens in HVS are like a camera lens to do the image focusing. Additionally, there are similarities between the retina and image sensor of the camera, but there are also significant differences in the detail of the systems [5]. The basics of camera operation and image pixel color creation are explained in this section in order to complete the background information necessary for computational color constancy. Then, the ISP structure with a focus on computational color constancy is explained.

Digital camera components include a lens, aperture, actuators, single or multiple filters, sensor, and an ISP. A camera module cross-section without the ISP is shown in Figure 2.11 with an example Bayer pattern color filter array (CFA). The light from the 3D environment, after interacting with the objects in the scene, goes through the optic lens of the camera. The Infra-red cutoff filter stops the incoming lights above the infra-red region as well as below the ultra-violet region so that light from the visible spectrum passes through. Approximately, the above infra-red region starts between 700nm to 780nm, and below ultra-violet starts from 380nm [5]. The aim of removing these wavelengths is due to consumer photography not needing them as well as photographing the scene according to the consumer preference, which is the HVS. Additionally, part of the dynamic range of the pixel would be consumed by those invisible wavelengths, leaving less for the visible wavelengths. Microlens array helps the camera to collect light from the scene more efficiently. Gathered light from the visible spectrum goes through a CFA to implement different color components of the sensor. An example CFA is shown in Figure 2.11 with a 2x2 Bayer pattern of red green and blue [39]. Raw Bayer image creation is completed by the camera sensor. There are 2 fundamental camera sensors that are used in digital cameras nowadays, Complementary Metal Oxide Semiconductor (CMOS) and Charge-

Coupled Device (CCD) [40], [41]. There are advantageous use cases for each camera sensor depending on the field of application, but CMOS sensors are widely used in digital cameras [42].

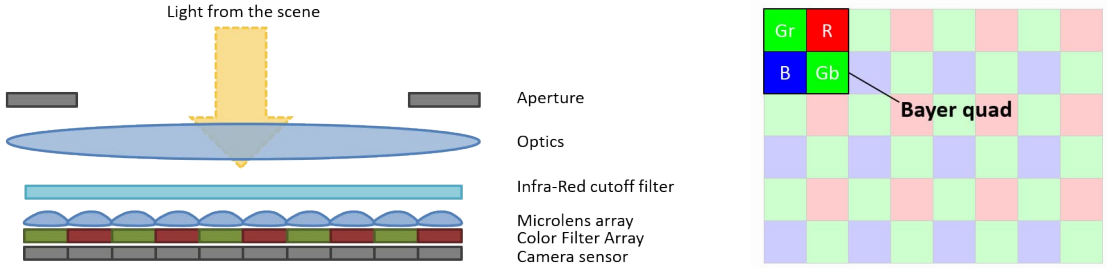


Figure 2.11. *A camera cross section and an example Bayer pattern. (a) visualizes an example camera module cross section and (b) is an example 2x2 Bayer pattern color filter array (CFA). Both are obtained from [7].*

The spectral response of different color channels of a camera module is identified by its infra-red cutoff filter, single or multiple color filters, and the camera sensor. These can cause even large variations in camera module sensor responses, and a simple example of it can be seen in Figure 2.12. The example camera sensors shown in Figure 2.12 are Canon 5DSR, Sony IMX135, and Nikon D810. The legend coloring visualizes the red, green, and blue channels of the camera module as green is the average from green-red and green-blue channels. Canon 5DSR and Nikon D810 are Digital Single-Lens Reflex (DSLR) camera modules, and Sony IMX135 is a considerably small camera module mainly used in mobile phones, and all of them have CMOS sensors. Their camera sensor spectral sensitivities are open source and, therefore, used in the rest of this thesis [18]. The effect of the infra-red cutoff filter can be observed in all camera modules since no wavelength approximately after 700nm and before 380nm is included in pixel color production. Shown spectral sensitivities are somewhat similar to the cone photoreceptor spectral sensitivity of HVS in Figure 2.10. A fundamental difference between the two is that the photoreceptors of the HVS are adaptable to complex environments by changes in the spectral sensitivity, but the camera sensor spectral sensitivity is an absolute measure staying constant. Due to camera sensor spectral sensitivity staying constant throughout different environments, several algorithms run on ISP aim to solve the problem.

A raw color image can be formed with a known camera sensor spectral sensitivity similarly to Equation 2.18 in HVS. Equation 2.19 shows how the image signal I can be acquired for each color channel k for pixel (x, y), provided the SPD of the illuminant on that pixel $\Phi(\lambda, x, y)$, spectral reflectance of surface $S(\lambda, x, y)$ and camera sensor spectral sensitivity of the color channel $C_k(\lambda)$ through the visible spectrum Λ. The equation assumes lambertian surface reflectance where the geometrical factors are eliminated, as it was mentioned before. This equation gives an overall understanding of how an image

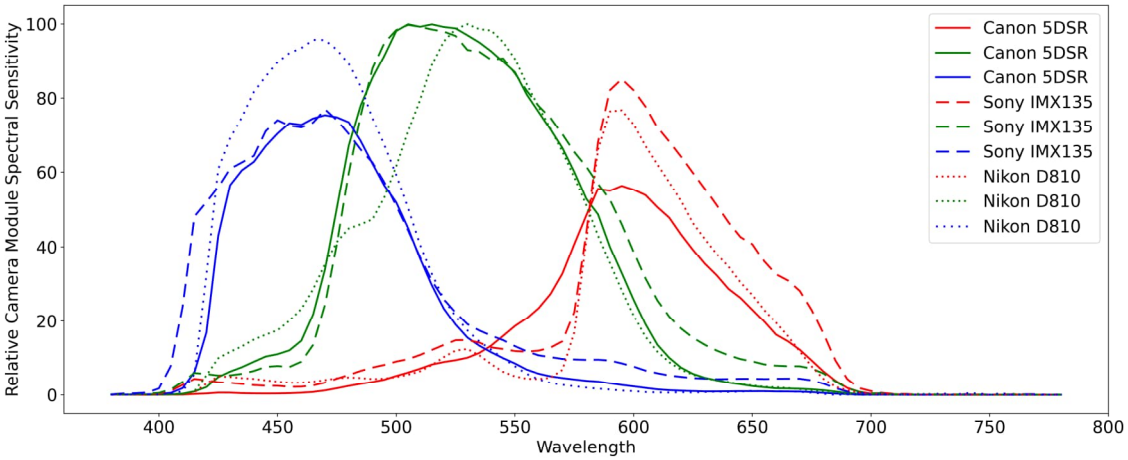


Figure 2.12. *Relative camera module spectral sensitivity of the Canon 5DSR, Sony IMX135 and Nikon D810. Data is obtained from [18] and colors specified in legend represents the color channels of the camera module. Green channel is shown as the average of green-red and green-blue.*

is formed in a digital camera, but it is not physically accurate for all real-life situations [7]. Nevertheless, it is important to understand that C, Φ, and S directly influence the output image signal. It proves the color inconstancy of the digital cameras since any change in illuminant Φ will cause a change in pixel color values for the same surface and camera sensor and therefore creates the need of a color constancy algorithm in the ISP.

$$I_k(x,y) = \int_\Lambda C_k(\lambda)\Phi(\lambda,x,y)S(\lambda,x,y)d\lambda \tag{2.19}$$

Various algorithms of image processing are needed to compensate for the inability of the digital cameras compared to HVS as well as produce consumer-preferred final image output. These algorithms run in harmony with each other, and their order of design may vary, but an example set of algorithms run on ISP is shown in Figure 2.13 [7]. The design and order of the algorithms vary due to requirements from the system, memory and processing power restrictions, and several other trade-offs between speed and accuracy. This set of camera algorithms consists of pixel in, pixel and other data out as well as several analysis algorithms that coexist to produce consumer-preferred image output. An important set of analysis algorithms are 3A and Global Brightness and Contrast Enhancement (GBCE). 3A algorithms are Automatic White Balance (AWB), Automatic Exposure Control (AEC), and Automatic Focus (AF). Computational color constancy algorithms are also called AWB algorithms in the digital imaging and academic community due to the goal of having a white area stay white under different illuminations. There are several pre-processing algorithms that are needed to be run in advance for AWB analysis to perform well. This is due to the correcting of non-linearities of the raw data so that AWB analysis performs better. AWB analysis estimates the chromaticity of the illumination in the scene and discounts its effect on the surfaces, making them color constant. Once

the AWB is completed, the rest of the set of algorithms are implemented so that device-independent color space image can be outputted. But if the AWB analysis is incorrect, the color inconstancy from the illumination source causes visible color casts at the output image [7].

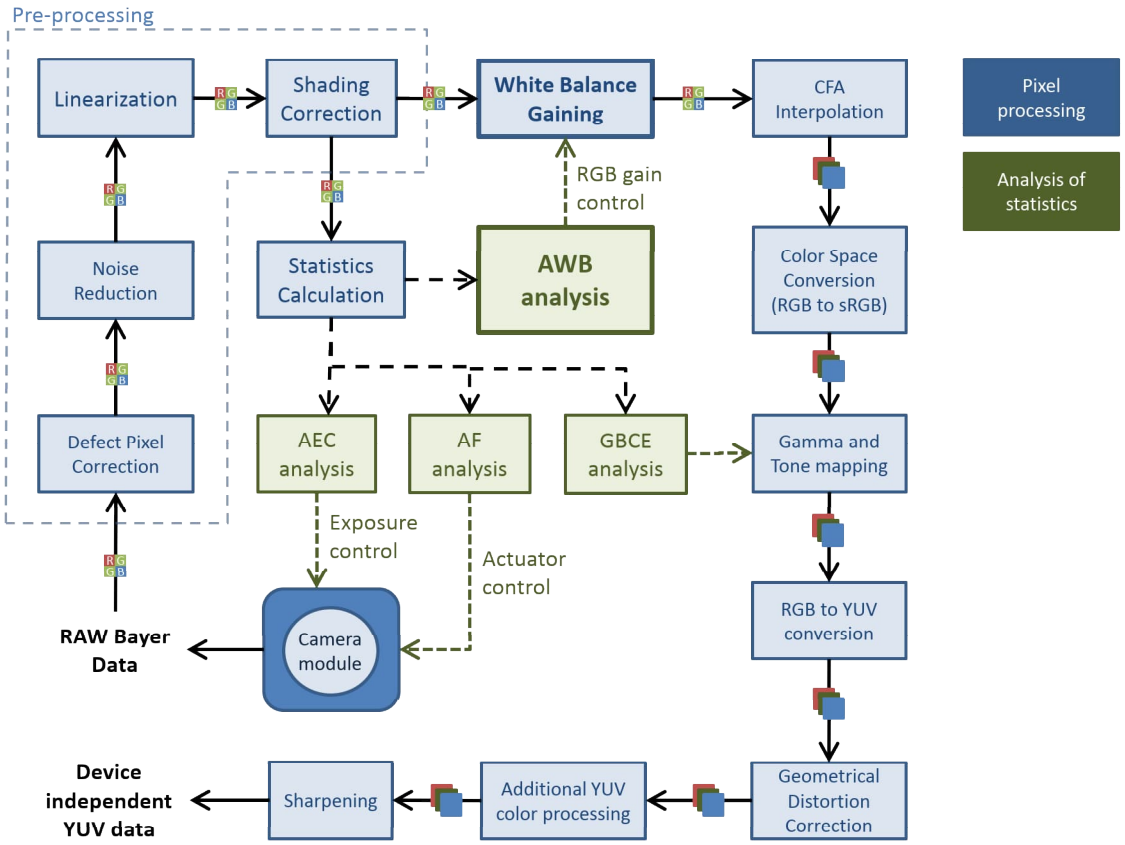


Figure 2.13. *A set of camera algorithms run on ISP, obtained from [7].*

Given an image scene of a certain camera module, AWB algorithms predict the chromaticity of the illumination in the scene. The goal is to transform the input image taken under an unknown illuminant to have colors as if the image is taken under a canonical illuminant. This transformation can be done by using a diagonal model or *von Kries* coefficient rule [43]. The diagonal model can be formulated for each pixel location (x, y) as

$$I^c(x,y) = G^{u,c}(x,y) \cdot I^u(x,y), \qquad G^{u,c} = \begin{bmatrix} w_R^c/w_R^u & 0 & 0 \\ 0 & w_G^c/w_G^u & 0 \\ 0 & 0 & w_B^c/w_B^u \end{bmatrix} \tag{2.20}$$

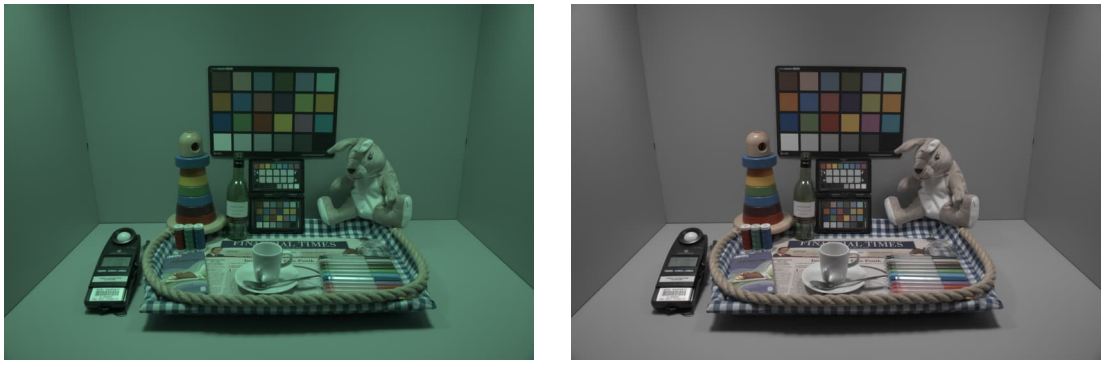
where I^c is image signal under canonical illumination, $G^{u,c}$ is the diagonal 3x3 white balance gain matrix of $w^u = [w_R^u, w_G^u, w_B^u]$ and $w^c = [w_R^c, w_G^c, w_B^c]$, and I^u is the im-

age signal under an unknown illuminant, for the spatial pixel location (x, y). Here, w^u represents the linearized camera module response of an achromatic surface under an unknown illumination, and w^c represents the same achromatic surface under canonical illumination. If all pixel colors of the image signal I^u can be transformed from unknown illumination to canonical illumination image signal I^c, it is evident to say that the diagonal model, or von Kries coefficient rule holds true [7]. So the diagonal gain matrix is the addition of the chromaticity of the illuminant, which needs to be discounted by the AWB analysis. Therefore it is required to be predicted. Let us define the diagonal matrix in Equation 2.20 as $diag(gain_R, gain_G, gain_B)$, this three white balancing gain can also be written as

$$c = [R/G, B/G] \tag{2.21}$$

where c is the chromaticity of illuminant and R, G, and B refer to the pre-processed raw values of red, green, and blue for that specific pixel for the used camera sensor. This two-channel representation holds valid because the chromaticity of the illuminant is independent of the brightness [7]. As a result, the correlation between the white balance gains and chromaticity of the illumination c can be defined as

$$c_{wp} = [gain_G/gain_R, gain_G/gain_B] \tag{2.22}$$



(a) ***(b)***

Figure 2.14. *Example Canon 5DSR raw image from Intel-TAU dataset [18]. (a) is the raw image and (b) is the white balance gain applied image. Both images are gamma applied for visualization purposes.*

Once the correct White Point (WP) is found and applied to the raw image, achromatic objects like gray patches of the color checker should appear achromatic, and other colors should be closer to how they appear in HVS. An example laboratory image taken with Canon 5DSR from the Intel-TAU dataset is shown in Figure 2.14 [18]. Figure 2.14 (a) is the raw preprocessed sensor response ready to be used in the AWB algorithm, and the image (b) is after the white balance gain is applied. However, with only white balancing,

chromatic colors are still under-saturated and not converted from sensor color space to the target color space, which is sRGB [44]. The conversion from sensor color RGB to target color sRGB is done via *Color Correction Matrix* (CCM). CCM is applied after the white balance gains as a 3x3 matrix shown in Equation 2.23. An example CCM is given in Equation 2.24 for the image shown in Figure 2.14. CCM applied version of the Figure 2.14 is shown in Figure 2.15. This CCM is only for the camera module that captured the image according to D65 illumination. It should be noted that any error in the predicted WP that causes color cast for the scene will be further amplified by the CCM [45].

$$
\begin{aligned}
I^c(x,y) &= CCM(x,y) \cdot G^{u,c}(x,y) \cdot I^u(x,y) \\
CCM(x,y) &= \begin{bmatrix} RinR & GinR & BinR \\ RinG & GinG & BinG \\ RinB & GinB & BinB \end{bmatrix}
\end{aligned}
\tag{2.23}
$$

$$
CCM_{RGB2sRGB,D65} = \begin{bmatrix} 1.991 & -1.186 & 0.194 \\ -0.231 & 1.788 & -0.557 \\ -0.019 & -0.502 & 1.521 \end{bmatrix}
\tag{2.24}
$$

Figure 2.15. *CCM applied version of the Figure 2.14 (b). Gamma is applied for visualization purposes.*

All AWB algorithms aim to find the correct c_{wp} which represents the chromaticity of the illumination for the specific camera module. It can be understood from the Equation 2.19 that for the same surface, depending on the change in the illumination, or the camera module, the output image signal varies. This variation will result in different c_{wp} vectors for the same scene. It is important to say that the variation may become too much to eliminate since, in Figure 2.12, there are considerable differences between the camera module spectral sensitivities. Some of the AWB algorithms are independent of camera module response and utilize image signal statistics [8], [9] and [10]. Nevertheless, in the mass production of low-cost products like mobile phone or common digital cameras, these camera module differences and even sample-to-sample differences are important to address for the final image quality [7], [46]. The hardship of having a camera module-dependent algorithm is that it requires a new dataset for each camera module which is not feasible from a financial point of view.

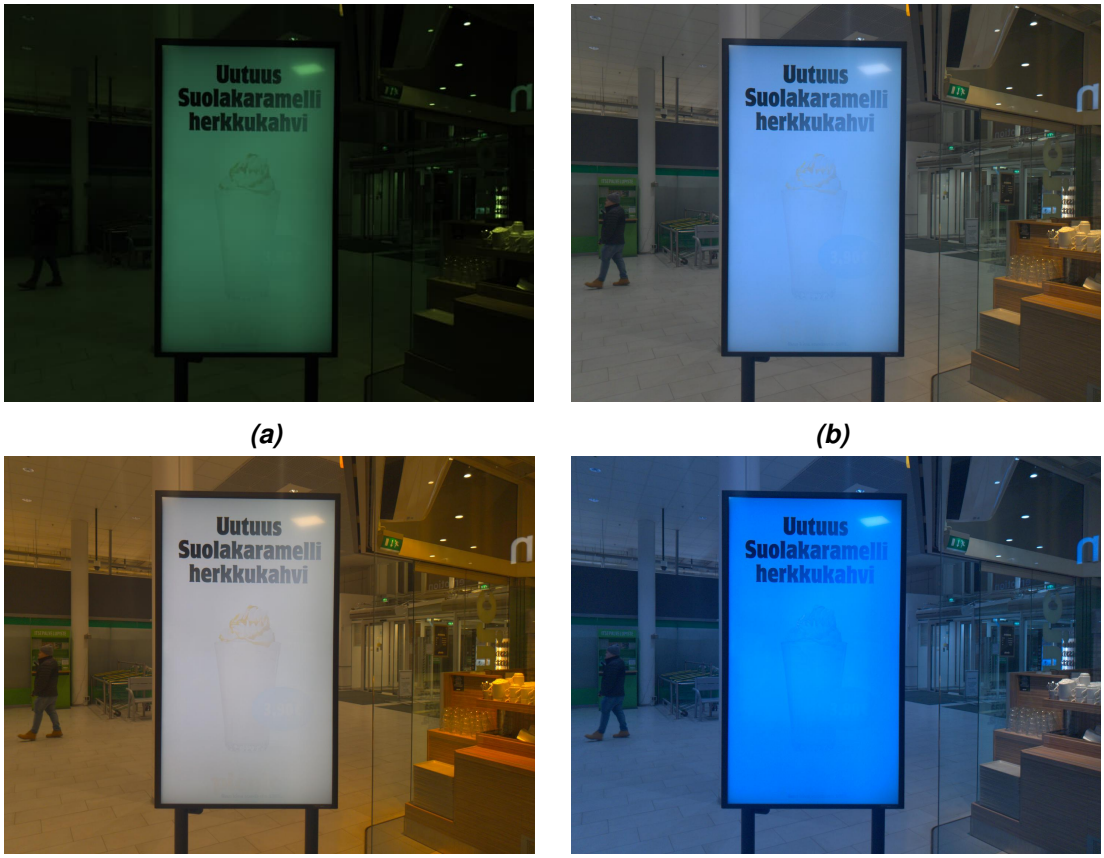


Figure 2.16. *An example spatially varying multi illuminant scene. (a) is the raw image, (b) is gamma applied sRGB image by using the WP of the ceiling illuminant of background mall, (c) is the gamma applied sRGB image by using the WP of the self-illuminant advertisement board and (d) is the gamma applied sRGB image by using the WP of the coffee shop ceiling light.*

A very crucial point to address in this thesis is that most of the AWB algorithms are built on the idea that there is only one dominant illuminant in the scene and therefore their aim is

to find the chromaticity of that illuminant [8], [9], [10], [11], [12], [13], [14], [15]. But in real life, where different light sources exist together, this is a frequently violated assumption [16], [17]. The fundamental reason for the single illuminant estimation is the difficulty of annotating spatially varying multiple illuminant scenes. Because, rather than a single c_{wp} for the whole scene, mixed illuminated scenes require c_{wp} for each pixel. That is why, Equations 2.19, 2.20 and 2.23 indicates the spatial (x, y) pixel location of the image signal. An example scene is shown in Figure 2.16 to visualize how laborious it can be to annotate spatially varying mixed illuminant scenes without causing color casts. It can be observed that with WP of each illuminant used for the whole scene, there are unrealistic color appearances. There are mixed illumination datasets created either by synthesis [16], [47], synthetically [17] or in controlled environments [48],[49] and [50]. But, especially data-hungry learning-based methods require a much bigger dataset for reliable output. In this thesis, Section 3 explains how to produce physically realistic spatially varying mixed illuminated scenes with their ground truth white point map for each pixel.

2.3 Computational color constancy algorithms

The goal of the computational color constancy algorithms is to estimate the chromaticity of the illuminant and turn the image into how it would look under a canonical illuminant. It is one of the fundamental steps in an ISP pipeline with a dependency on the camera module used to capture the image. They aim to predict the c_{wp} white balance gain as in Equation 2.22. But the input images are required to be preprocessed in order to eliminate the non-idealities of the raw image that helps for more reliable prediction. Lens shading correction, and black level correction are examples of these preprocessing steps[7]. Once the raw image is prepared, it can be used in an AWB algorithm as input.

The goal of the AWB algorithms is not to make the image fully color constant but rather to provide correct information regarding the chromaticity of the illuminant so that several color appearance modeling can be applied accordingly [5]. Therefore, wrong white balancing could further lead to failure in the color appearance modelings. It is explained previously in Section 2.2.3 that after the white balancing gain is applied, camera sensor RGB is transformed to device-independent sRGB color space with a CCM matrix, which amplifies the white balance error further in the final color appearance [44], [45].

In this section, a general overview of 6 different AWB algorithms is explained. 4 of them are statistical-based methods where image statistics are used in the white balance gain prediction. These algorithms in Section 2.3.1 - 2.3.3 are camera module independent while Section 2.3.4 is dependent to the used camera module. These statistical methods are used mainly for single illuminant estimation for the whole image. The other 2 are learning-based methods where the algorithm is trained priorly with images that have ground truth annotation. By learning from the shown examples, they can make predictions

on unseen images through correlation. The algorithm in Section 2.3.5 is camera module dependent while the one explained in Section 2.3.6 is not. Both of these learning-based algorithms are for mixed illuminated scenes that make pixel-wise predictions. Evaluation of these learning-based algorithms are provided in Section 5.2 with error metric defined in Appendix B.

2.3.1 Gray world

The basic gray world algorithm is built on top of the idea that the average reflectance of surfaces in an image scene is achromatic [8]. Simply put, the average pixel values of the color channels red, green, and blue are equivalent to each other. The gray world algorithm is commonly used as a base reference for new color constancy algorithms and datasets. It is due to its simple implementation, low computational cost, and easily predictable behavior. The white balance gain calculation for the gray world can be seen below in Equation 2.25 where I_R, I_G, and I_B are red, green, and blue color channels of the image scene, respectively. The diagonal matrix representation is based on the white balance gain in Section 2.2.3.

$$G_{GW} = \begin{bmatrix} \frac{mean(I_G)}{mean(I_R)} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \frac{mean(I_G)}{mean(I_B)} \end{bmatrix} \tag{2.25}$$

The accurate prediction of illumination chromaticity by the gray world algorithm depends on what the scene contains. If it consists of mostly achromatic objects or if the scene is colorful in a way that colors are represented somewhat equally, the performance of the algorithm is expected to be good. However, scenes with one channel of color covering the majority of the image scene lead to the average scene not to be gray, the performance of the algorithm decreases. For instance, a large red object in the scene would cause the color error towards cyan [46].

2.3.2 Max-RGB

Max-RGB or Scale-By-Max is another simple algorithm that assumes illumination chromaticity is the maximum response in each color channel. The algorithm is a limited version of Retinex theory, and it is dependent on the dynamic range of the used camera sensor [51]. White balance gain calculation with Max-RGB algorithm can be seen in Equation 2.26 below where I_R, I_G, and I_B are red, green, and blue color channels of the image scene, respectively.

$$G_{max-RGB} = \begin{bmatrix} \frac{max(I_G)}{max(I_R)} & 0 & 0 \\ & 1 & 0 \\ 0 & 0 & \frac{max(I_G)}{max(I_B)} \end{bmatrix} \tag{2.26}$$

The Max-RGB algorithm performs well if brightest color channel values of an image scene belong to an achromatic object or specular reflection. But the prediction quality is dependent on matte reflectances, the number of surfaces, existing specularities in the scene that causes maximum reflectance to be greater than pure white, and chromatic object belonging to diffuse reflection [51].

2.3.3 Shades of gray

Shades of gray algorithm exploit the Gray World and Max-RGB algorithms by making them two instantiations of *Minkowski norm* with $p = 1$ and $p = \infty$, respectively [12]. The white balance gain calculation with the Shades of Gray algorithm can be seen below in Equation 2.27 where c is the color channels like red, green, or blue, while m and n represents the dimensions of the image I.

$$G_{max-RGB} = \begin{bmatrix} \frac{psum_G}{psum_R} & 0 & 0 \\ & 1 & 0 \\ 0 & 0 & \frac{psum_G}{psum_B} \end{bmatrix} \tag{2.27}$$

$$I_c(x, y) : psum_c = \sqrt[p]{\frac{1}{mn} \sum I_c^p(x, y)}$$

Shades of Gray algorithm combines the good and bad properties of Gray World and Max-RGB by giving different weights by the parameter p while keeping the simplicity of both. Through repetitive experiments by the author, it is found that $p = 6$ performs the most accurately for a trial of $1 < p < \infty$ on the tested datasets [12].

2.3.4 Gray search algorithms

Gray surfaces reflect all wavelengths of illuminants equally, and that is why they are called achromatic. If the SPD of illumination is reflected on this achromatic surface, it turns to represent the chromaticity of the specific illuminant. With methods of searching and finding these gray surfaces, a white balance prediction for the scene can be made. Because the found achromatic area represents the chromaticity of the illuminant. The algorithm

in [52] searches for gray surfaces in YUV coordinate while [53] exploits the strength and simplicity of the Gray World algorithm to search for gray regions, and it is camera dependent. The algorithm in [54] searches for gray surfaces in the scene iteratively with the help of statistical algorithms.

2.3.5 LSMI U-net

This learning-based algorithm is based on the dataset shared by the paper itself called Large Scale Multi Illuminant (LSMI). It is a dataset of mixed illuminated real-life images with their ground truth pixel-wise white balance gain map [16]. The dataset consists of several Color Checkers placed around the scene to calculate the white point of the illuminants from the achromatic color patches. Images in the dataset either have 2 or 3 illuminants, and the image is taken from the same camera angle by turning off light sources one by one. This technique is used for calculating their contribution to the mixed illuminated image through the green color channel of the raw image. They claim this weighting approach helps approximate the coefficient map of illuminants. An example image from the dataset with its corresponding illuminant coefficient map can be seen in Figure 2.17 below. The coefficients and the chromaticity of illuminant calculated from the achromatic patches of the Color Checker result in a pixel-wise white balance gain map. This open-source dataset consists of 2762 unique scenes captured with 3 different camera modules in different amounts, which leads to 7486 total images. Although the dataset is contributing to mixed illuminated white balance research, some of the ground truth calculations fail due to not including reflected indirect illumination.

Figure 2.17. *An example mixed illuminated image from the LSMI dataset with corresponding illuminant coefficient map.*

The authors used the dataset they created for pixel-wise white balance neural network training with a U-net [55], and HDRnet [56] models. For HDRnet, 256x256 and 128x128 raw RGB images are used as high and low-resolution image inputs. The default hyperparameters are used as in the original paper in [56] with mean square error and cosine

similarity as the loss function. It is expected from the network to learn the chromaticity of each pixel in the image. For U-net, 256x256 image size is used with a 7 levels of encoder-decoder structure. Differently from HDRnet, images are converted to l, u, v color space before they are given into the network, as l representing luminance and two chrominance channels u and v.The network only uses two chrominance channels as input and learns two chrominance channels of the ground truth white-balanced image. Therefore mean square errors between the chrominance vectors are used as loss function. 3 different versions of the U-net are trained for each camera module Samsung Galaxy Note 20 Ultra, Nikon D810 Sony α9, and the neural network models are publicly available.

2.3.6 Mixed illumination white balance

This learning-based algorithm does not perform illumination chromaticity estimation but rather renders the wrong white-balanced sRGB image to its corrected version. This is done through estimating weights of predefined white balance settings [57]. They adopted GridNet architecture consisting of 6 columns and 4 rows [58],[59]. The network takes an initial camera module output sRGB image with mixed illumination and creates smaller representations of the initial image with different white balance settings. These white balance settings are explained as tungsten (incandescent), fluorescent, daylight, cloudy, and shade, referring to 2850K, 3800K, 5500K, 6500K, and 7500K, respectively. Given these small different white balance settings applied 384x384 sRGB images, the proposed method maps the weights of each image to the ground truth final sRGB image. The blended version of these corresponding weights results in the predicted sRGB output. GridNet network in this study used reconstruction loss as the loss function.

The dataset used in the training of this neural network is Rendered WB dataset [60] that consists of approximately 65000 sRGB images that same image exist in several different white balance settings with common CCT values. Each of these images also has a corresponding ground truth white-balanced sRGB version. 9200 ground truth images, each with its different white balance setting applied version, are used in the training of this network. The authors also created 150 mixed illuminated synthetic images to be used in the testing of the network. The synthetic dataset consists of images rendered with at least 2 different illuminants, and the ground truth image is rendered using the 5500K illuminant for all light sources in the scene. It is reported that these scenes are created in Autodesk 3Ds Max, and photo-realistic rendering is completed using Vray render engine [57].

3. PROPOSED METHOD

The main research goal of the present study is to create a spatial color constancy dataset. This is achieved by creating physically realistic camera sensor invariant mixed illuminated scenes. Image generation is made with 3D spectral rendering in a controlled environment so that the ground truth white point map can be calculated. This is important because the pixel-wise annotation made by humans would be orders of magnitude more laborious and not necessarily correct. Spatial color constancy dataset creation is achieved with the pipeline shown in Figure 3.1. It starts with spectral ray tracing, the proposed method in this thesis to overcome the hardship of the laborious annotation process. Spectral ray tracing requires material surfaces to be defined with their spectral reflectance and light sources with their relative spectral power distributions. Rather than the RGB color space, defining them with their spectral representations throughout the whole visible spectrum results in the physical realism of the real-world image signal creation. Another advantage of this method of image signal creation is that created images can be converted to any camera's response as long as the relative camera sensor spectral response is known. Thanks to defining each illuminant in the scene, the pixel-wise white balance gain map can be calculated for the specified camera sensor in order to have a fully color constant image. The pipeline includes further steps to transform the image from the specified camera color space to the sRGB color space. The physical realism of the generated image signal is vital since a learning-based auto white balance algorithm that solves pixel-wise color constancy trained with the created dataset images would be tested in real life. Therefore compatibility between the two will affect the predicted output image signal quality. Additionally, if this learning-based algorithm needs to be re-trained for a different camera sensor in the future, the created dataset has the advantage to be transformed to that specific sensor image response.

The rest of this chapter details the spectral ray tracing methodology and its use in color constancy for mixed illuminated scenes. Section 3.1 introduces the spectral image generation in order to generate physically accurate images. Then Section 3.2 defines how these spectral images can be used in the pixel-wise white balance gain map calculation.

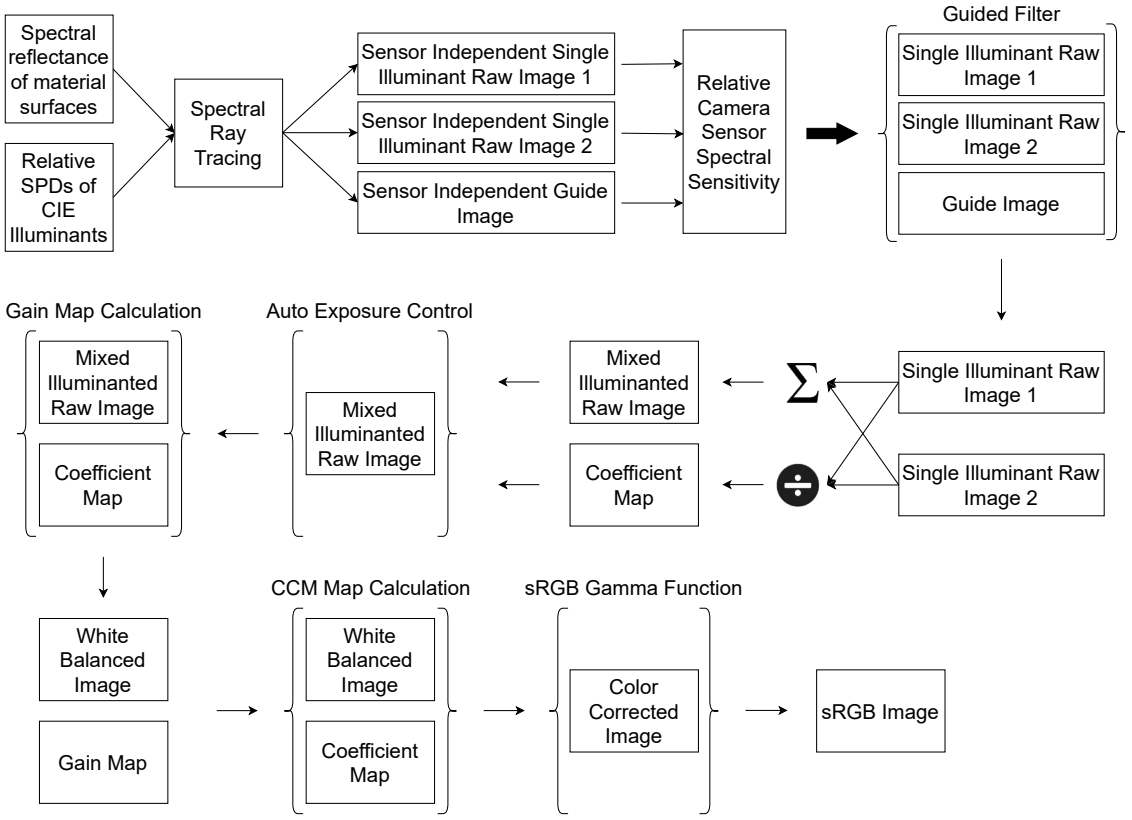


Figure 3.1. *Used pipeline to generate images via spectral ray tracing.*

3.1 Spectral image generation

Ray tracing is widely used in rendering 3D scenes into 2D images or animations. Color definition of materials and light sources in ray tracing is mainly made in RGB or HSV color space. This causes metamerism, and it is an unwanted phenomenon if the goal is to match an image signal to the physical realism of the world. However, with spectral ray tracing for image generation, both physically correct image generation can be accomplished, and metamerism can be handled. Spectral image generation requires the material surfaces to be defined with spectral reflectance and illumination to be defined with its relative SPD. The output full-spectrum image signal consists of a stacked combination of all the spectral images, which are the output of spectral rendering for each wavelength. In Section 2.1.5, Equation 2.15 explains the method that is used in this thesis for full-spectrum image generation. The only difference is that spectral representations of material surfaces and illuminants are made between 395nm and 705nm with 5nm equally sized bins. Due to infra-red cut-off filter of the used camera modules, a narrower visible spectrum gives approximately the same image signal with a lower rendering time. The equation of spectral ray tracing for the visible spectrum $\lambda^n = [\lambda^{395}, \lambda^{400}, ..., \lambda^{705}]$ can be written as

$$\widehat{I_k}(\lambda^n) = \frac{1}{N}\sum_{j=1}^{N}\frac{L_k(\overline{X}_j, \lambda_j^n)}{p(\overline{X}_j, \lambda_j^n)} \tag{3.1}$$

where j is the number of sent ray samples per pixel and $\overline{X}_j$ is all possible light paths that contribute to the pixel value k for the wavelength λ_j^n. One way to do spectral ray tracing is with equally binned wavelength representation within the visible spectrum. The bin size may vary depending on the application and the available computing resources, but the important point is to be able to include especially the spiky material or illuminant spectral representations such as illumination F11 in Figure 2.5. In this study, physical realism is more important than computing time since images need to be generated only once. Therefore, 5nm equal bins are used, which leads to 63 spectral channels. The image signal $\widehat{I}(\lambda^{395,..,705})$ for all pixels and spectral channels stacked as a tensor results in a camera invariant full-spectrum 63 channels image.

Spectral ray tracing requires a wavelength value to define the pixel value rather than a trichromatic definition. Firstly, a fired ray from the camera hits a surface or volume to calculate a pixel value, considering there is an existing illumination in the 3D scene. As mentioned before in Section 2.1.5, material surfaces are defined with their spectral reflectances. However, in order to create a variety of scenes, a rich library of material spectral reflectance is required. *In situ* database has a variety of scenes from the real world with spectral radiance measurements of some of the surfaces in the scene with a spectroradiometer covering the visible spectrum between 380*nm* and 780*nm* [61]. Besides the spectral radiance, a white tile corrected version of the surfaces is also shared in the dataset, which is used in spectral ray tracing material surface definition. The dataset has over 1800 real-life surfaces that can be used to create complex spectral images with a variety of colors. So, by designing a 3D scene where materials are defined with a preferred spectral reflectance from the In Situ database, incoming rays can hit a spectral surface.

Each fired ray from the virtual camera needs to end up with a light source in the scene after the surface interactions to contribute to the pixel value. As is the case with material surfaces, light sources in the scene are required to be defined with their spectral representations so that spectral image pixels can be generated. Illuminations are defined with their relative spectral power distributions and can be measured with an illuminance meter. A rich library of illuminants is necessary to create spatially varying mixed illuminated scenes. Therefore, 32 CIE standardized illuminants consisting of D series, F series LED series, and illuminant A are used in spectral image generation. Illuminants with a CCT of little over 2700K until 25000K provide a wide range of selection to create spatially varying mixed illuminated scenes. With each light source in a 3D scene defined with its relative SPD, the incoming light ray can contribute to the final pixel value of the specific

wavelength. Figure 3.2 visualizes a spectral image generation of a color checker scene with each surface defined with spectral reflectance and illuminant with relative SPD in the visible spectrum. As a result, spectral images of each wavelength band can be obtained. A significant advantage of this full-spectrum image is that it can be transformed to be the raw response of any camera module as long as the relative camera sensor spectral sensitivity is known.

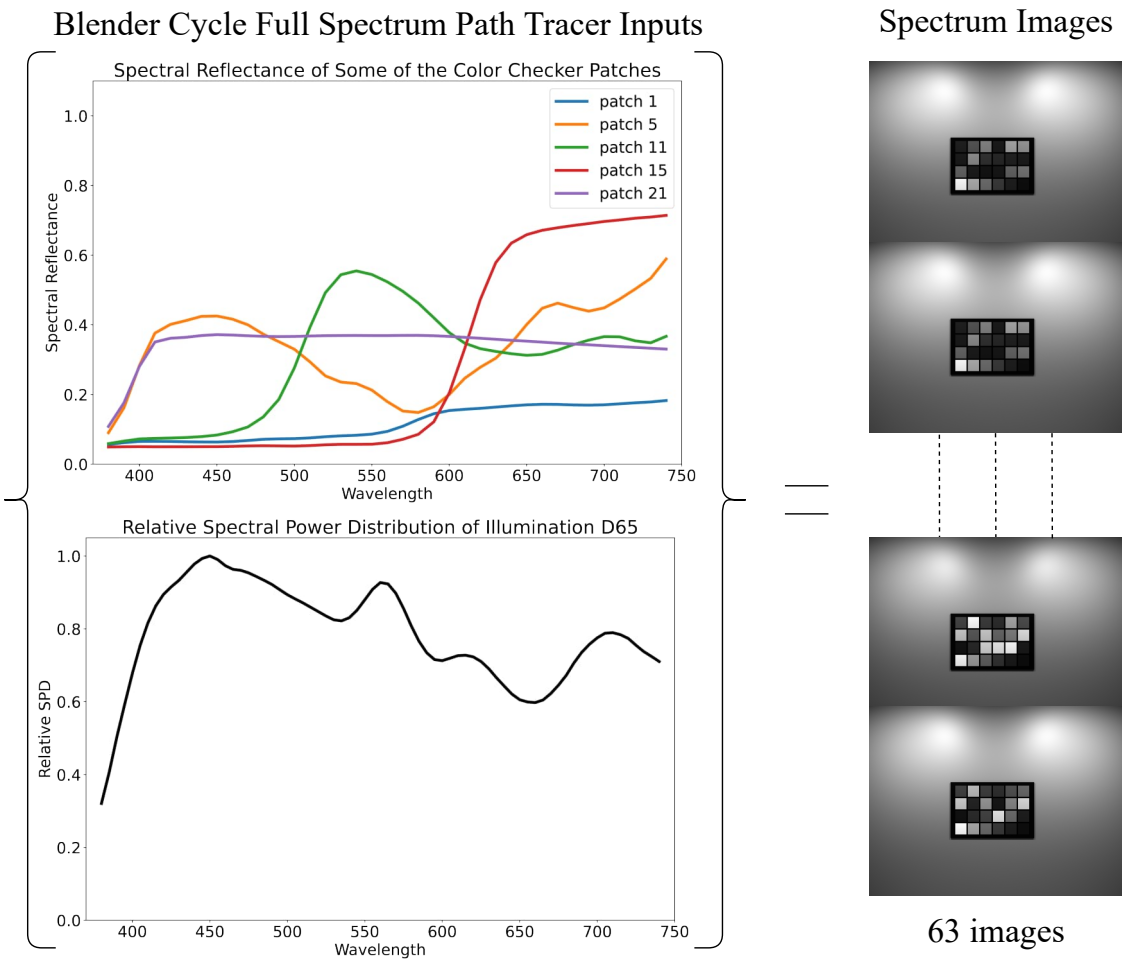


Figure 3.2. *Spectral image generation of a color checker under D65 illuminant.*

Spectral image $\widehat{I}$ consists of predefined material and illuminants. In this study, mixed illuminant spectral image generation is accomplished by summing single illuminant images. To create scenes that have only 1 type of illuminant, light sources in the scene are grouped, and rendering is completed while only one group of illuminants is on. Basically, turning on light sources of the group "light 1" and turning off group "light 2" in order to create scenes that are made up with only a single illuminant type. The same process is repeated for the group "light 2" so that their sum will result in the mixed illuminated scene. Each light group represents a chosen illuminant from the CIE library. Grouping is done by naming the light sources, including the emission materials. The additive method of creating spatial scenes relies on using the same random seed for the ray generator for

the same camera angle. Although this approach depends on the number of fired rays per pixel, it will be shown in Section 4 that it holds fairly true for physically accurate image signal creation.

3.2 Ground truth white balance gain map calculation

In controlled environments of mixed illuminated scenes, it is possible to get the pixel-wise white balance gain through calculation [48],[49] and [50]. This is valid for spectrally generated images also, due to illuminants being predefined and known. There are 3 indicators that are needed in order to calculate the gain map of a mixed illuminated scene; pixel-wise weight map of the illuminants, relative SPD of the illuminants, and relative camera sensor spectral response. As mentioned in Section 3.1, full-spectrum images can be transformed to be the raw response of any camera module as long as the relative camera sensor spectral response is known. It is a vital part of white balance gain calculation because the used camera module directly affects the output image signal, as can be seen in Equation 2.19. This is why pixel-wise white balance gain is sensor-dependent, and gain can be fairly different with a different camera module. In this thesis, 3 different camera modules are used due to their relative response being publicly available [18]. Used camera modules are Canon 5DSR, Sony IMX135, and Nikon D810.

Pixel-wise white balance gain map of a spatially varying mixed illuminated scene is shown below in Equation 3.2 and Equation 3.3. Firstly, full-spectrum images should be converted to match the response of a camera module. Due to predefining camera module and illuminant in the scene, the color of the illuminant for that camera sensor can be calculated as in Equation 3.2. This is also the white point of a single illuminant raw image for that camera sensor. Secondly and most importantly, the weight map of the illuminations in a pixel-wise manner needs to be calculated. Since mixed illuminated images are generated by summing single illuminant scenes, the weight map is calculated by the G color channel contribution of the raw single illuminant images to the mixed illuminated scene. This is adapted from the [16] with a suitable way for spectral images. The biggest difference of spectral ray tracing as an advantage though, any light source including the sky can be turned off. The weight is calculated for each pixel and varies between $[0, 1]$ where in any pixel, the sum of weights is maximum 1. Since the white balance of a mixed illuminated scene pixel is the linear average of the used illuminants with respect to their weights, white balance gain can be calculated linearly. This is repeated for the whole raw image signal, and a pixel-wise gain map can be acquired as in Equation 3.3. Pixel-wise multiplication of the wb_{gains} with the mixed illuminated raw image results in a white-balanced image. A mixed illuminated raw image, white balance gain map, and white balanced image can be seen in Figure 3.3 as an example.

$$\begin{aligned} \Phi_{RGB}(j,k) &= \int_{\lambda=395}^{\lambda=705} C(k,\lambda)\cdot\Phi(j,\lambda) \\ \Phi_{RGB}(j,k) &= \Phi_{RGB}(j,k)/\Phi_{RGB}(j,G) \end{aligned} \tag{3.2}$$

$$\begin{aligned} weight(j,x,y) &= \frac{I(j,G,x,y)}{\sum_{j=1}^{N} I(j,G,x,y)} \\ \Phi_{RGB,final}(k,x,y) &= \sum_{j=1}^{N} \Phi_{RGB}(j,k)\cdot weight(j,x,y) \\ \Phi_{RGB,\max} &= \max(\Phi_{RGB,final}(:,x,y)) \\ wb_{gains}(x,y) &= \begin{bmatrix} \Phi_{RGB,\max}/\Phi_{RGB,final}(R,x,y) \\ \Phi_{RGB,\max}/\Phi_{RGB,final}(G,x,y) \\ \Phi_{RGB,\max}/\Phi_{RGB,final}(B,x,y) \end{bmatrix} \end{aligned} \tag{3.3}$$

where:

λ	is the wavelength in nanometers
k	is color channels like $[R,G,B]$
j	is the index of the used illuminant in the scene
$C(k,\lambda)$	is the camera module spectral response for the color channel k
$\Phi(j,\lambda)$	is the relative SPD of the illuminant j
$\Phi_{RGB}(j,k)$	is the RGB values of the illuminant j
$I(j,G,x,y)$	is the green channel of image signal I that has illuminant j
$weight(j,x,y)$	is the weight of illuminant j on pixel (x,y)
$\Phi_{RGB,final}(k,x,y)$	is the summed color channel values of the existing illuminants
$wb_{gains}(x,y)$	is the white balance gain for the pixel (x,y)

(a) ***(b)*** ***(c)***

Figure 3.3. *Example gain map of a mixed illuminated image. (a) is the mixed illuminated spectrally generated raw image, (b) is the calculated white balance gain map, and (c) is the white balanced image after the gain is applied to the raw image. Both raw image and white balanced image are gamma applied for visualization purposes.*

After applying the white balance gain, the image is still under-saturated and not converted

to the sRGB target color space. This conversion is done for example, for single illuminant image in Figure 2.15. CCM is dependent on the illuminant in the scene and the camera module that captured the image. For spatially varying mixed illuminated scenes, a CCM map can be calculated in a similar manner to gain map. CCM matrices can also be linearly interpolated depending on the illuminants used. Therefore, calculating a CCM for each pixel can be done with the weight map used in the white balance gain map calculation. Multiplication of white balanced image with the calculated CCM map results in sRGB target color space output.

4. IMPLEMENTATION

This chapter explains the spectral ray tracing implementation with Blender Cycles render engine with its Python Application Programming Interface (API) as well as supplementary algorithms that are needed from raw image creation until sRGB output. Figure 3.1 is the fundamental pipeline used in this thesis to create a spectrally generated, physically accurate image from the beginning of defining the material and illuminant spectra until the sRGB image output. Each step of the pipeline is explained in this study.

4.1 Spectral ray tracing with Blender Cycles

This thesis uses Blender Cycles rendering engine to do spectral ray tracing. Blender is an open-source 3D content creation platform owned by a nonprofit organization called Blender Foundation [2]. Blender Cycles is an unbiased patch-tracing render engine that sends rays from the camera and traces them until they reach the light source. It supports CPU, GPU, or clusters of GPU to render scenes, which makes it flexible to use by a great number of people [2]. Blender Cycles is flexible to the definition of trichromatic values that give the color of light sources, and materials [17]. Therefore, wavelength values can be given as spectral color rather than R, G, and B. The only difference is that rendering needs to be repeated several times to cover the whole visible spectrum. In this thesis, it is repeated 21 times to cover 395nm until 705nm with 5nm bins, given 3 wavelengths at a time. It brings a big computational expense to generate one image, but doing it once is enough to use in color constancy dataset creation. Also, thanks to summing single illuminant images to create the mixed illuminated scene, several combinations of single illuminant images can be created. In addition, in this study, physical realism is more important than rendering time since average mixed illuminated scene creation takes between 7 to 10 minutes, depending on the number of meshes in the scene, with an Nvidia GeForce RTX 3090. It decreases 1.5 to 3 minutes on average if a cluster of 4 GPUs is used. Both experiments are for 720x960 pixel image size, as is the size used in all created images.

It is experimented that there are several changes needed in the Blender Cycles settings in order to perform physically realistic image generation. Most importantly, the used color space should be changed to "Non-color" since spectral values do not represent any color

space possible in Blender. Additionally, there are various settings that stop the render engine from sending more rays because it will contribute considerably less to the final pixel value. However, it is experimented that even these small contributions affect physical realism. Therefore these settings such as "Noise Threshold", "Light Threshold", "Clamping", and "Caustics" are turned off. These are experimented with assigning a single wavelength value to all 3 color channels and expecting the same histogram from color channels of the output image. The addition of Blender changes the pixel value while all 3 color channels are defined with the same wavelength. Settings that caused differences in the histogram of the color channels are turned off since it is expected that the histograms should be the same. Only indirect clamping is turned on in order to avoid the fireflies effect. Additionally, most of the rendering platforms have denoising capability in order to have less noisy and more appealing output at the end of the rendering. But it is studied that especially artificial intelligence (AI) based denoising algorithms change the raw response away from the physically realistic pixel values in order to enhance visual appearance. Therefore, no denoising algorithm of the Cycles rendering engine is used. The output image is set to be a 16-bit floating-point in OpenEXR "HALF" format [62].

Completing the Blender Cycles settings compatible with spectral ray tracing, there are 2 steps left to start rendering. Firstly, all material names should be changed with their compatible representative from the In situ database. Measured surfaces are recorded even in several forms in In situ database, which makes it easier to choose one. For instance, there are several options to assign a leaf or grass with different colors and shades. Secondly, illuminants and emission materials in the scene are grouped with respect to their changed names. Light groups are called "world", "light 1", "light 2", "light 3" etc. Assigning each light source into any group is done by changing the name of the light source. This grouping of light sources is helpful to render single illuminant scenes automatically with Blender python API. Because with a group of light sources sharing a common name, all those lights can be turned on or off automatically. The renaming is eased with a Blender add-on specifically designed to rename material and light sources in the 3D scene and save time.

Aside from defining a single material for a surface, textures can be used to create visually appealing scenes. Due to defining everything in the scene in its spectral form, textures should be replaced with spectral textures. Simple spectral textures are created using material reflectances from the In Situ database. This can be achieved by replacing a pixel color with a chosen spectral reflectance and saving each three spectral reflectance values as an image. For example, if the aim is to create a sky spectral texture, an RGB sky texture image can be used as a reference. White cloud pixels can be replaced with cloud spectral reflectance, and blue sky pixels can be replaced with blue sky spectral reflectance. The new spectral texture image size is $width \times height \times 63$ and can be saved as 21 3-channel images, each with 3 channels of spectral reflectance from 395nm

until 705nm. In each rendering, the next spectral image is used as a texture image, which makes the scene use the whole spectral representation. For instance, some material surfaces are defined with spectral textures in Figure 3.3.

Once the scene is prepared with the steps explained above, it can automatically render several full spectrum images at once by automatically turning on and off the light sources with a python script. In addition, several camera positions can be defined, and rendering can be done for several camera angles at once. The developed python script that runs on the Blender python API renders the single illuminant images and repeats it for every camera angle. In this thesis, 7 different camera positions are defined for each scene, which causes different white balance gain maps. Single illuminant images are rendered with 500 rays per-pixel, while guide images are rendered with 5000 rays per-pixel. 500 rays used images are noisy since no denoising algorithm is used, but this is solved with a guided filter approach by using the 5000 rays image. The details and the reason for a guided filter will be explained below in Section 4.2

4.2 Guided filter

Spectral ray tracing is a computationally expensive process, depending on the number of rays traced for each pixel and the number of total meshes in the scene, rendering time varies. In this study, 500 rays per pixel are used in single illuminant images, but since no denoising algorithm is used, images are noisy. The completed study showed that denoising algorithms make a considerable change in the R/G B/G of the raw image signal, which makes them unreliable if the aim is to create physically realistic images. Therefore, a new idea is proposed with an edge-preserving guided filter. The same scene from the same camera angle is rendered once when multiple light sources are on, but rather than 500 rays per pixel, it is rendered with 5000 rays, which causes the output image to be less noisy and have more reliable pixel values.

As a guided filter, interpolated convolution technique of the domain transform filter is used [63]. There are two things a guided filter performs, it increases the physical reality of the pixel without longer rendering times, and it does denoising the image while keeping the edges preserved. The effect of increased physical reality is calculated with $Relative_{R/G,B/G}$ difference between the 2 raw image signal. It is shown in Equation 4.1 where $L2$ is the euclidean distance, c is the same pixel both in reference image ref and evaluated image x. $Relative_{R/G,B/G}$ error is able to show several color errors, including color shading error [45]. For example, it is expected that there is no $Relative_{R/G,B/G}$ error between the rendering of a mixed illuminated scene and summing single illuminants. This is tested with and without the guided filter and reported in Figure 4.1. On the left side, the $Relative_{R/G,B/G}$ error between the sum of two 500 rays used single illuminant images and the same scene rendered with mixed illuminated 5000 is shown. It is reported

that the mean error is around 1.8% while the 99.3 percentile is around 9.43% for all pixels. On the right side, the same experiment is done after the guided filter is applied with 5000 rays used as the guide image on the single illuminant images. The median error decreases to 0.9% with 99.3 percentile is around 4.89%. A big advantage of guided filter is that it is independent of the used illuminant of the guided image. Therefore, rendering only one guide image can be used for all possible mixed illumination combinations. For instance, the guide image used in the experiment shown in Figure 4.1 is different from the illuminants used in 500 rays images. It is understood that if all the rendering were done with 5000 rays, the pixel values would be more reliable due to less noise, but it can relatively be done with a guided filter and only rendering one guide image of 5000 rays. This leads to keeping the total rendering time considerably shorter. It is understood that areas that have relatively high $Relative_{R/G,B/G}$ difference are either dark areas of the image or glossy surfaces. From the denoising point of view, Figure 4.2 shows a cross-section of a spectrally generated image, and denoising while preserving the edges thanks to the guided filter can be observed. On the left side, there is a mixed illuminated sRGB gamma applied image without the guided filter, and on the right, the same image with a guided filter is applied.

$$Relative_{R/G,B/G} = \frac{L2(c_{R/G,B/G,ref}, c_{R/G,B/G,x})}{L2(c_{R/G,B/G,ref}, \vec{origin})} \times 100 \tag{4.1}$$

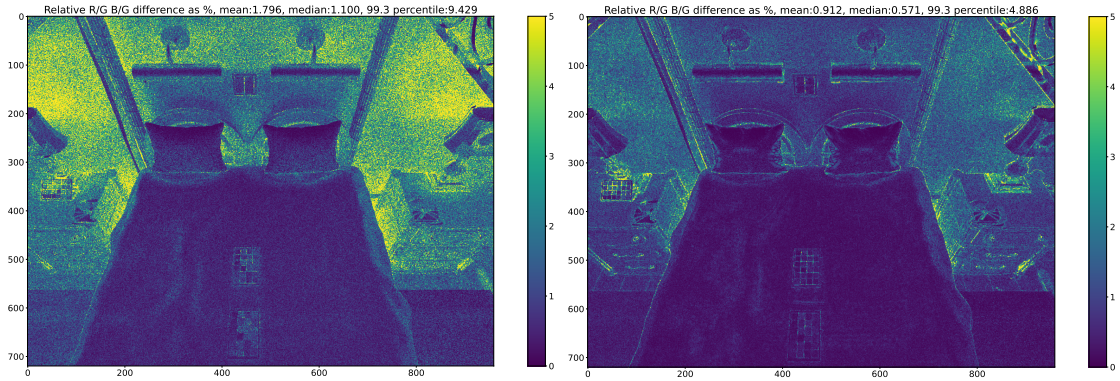
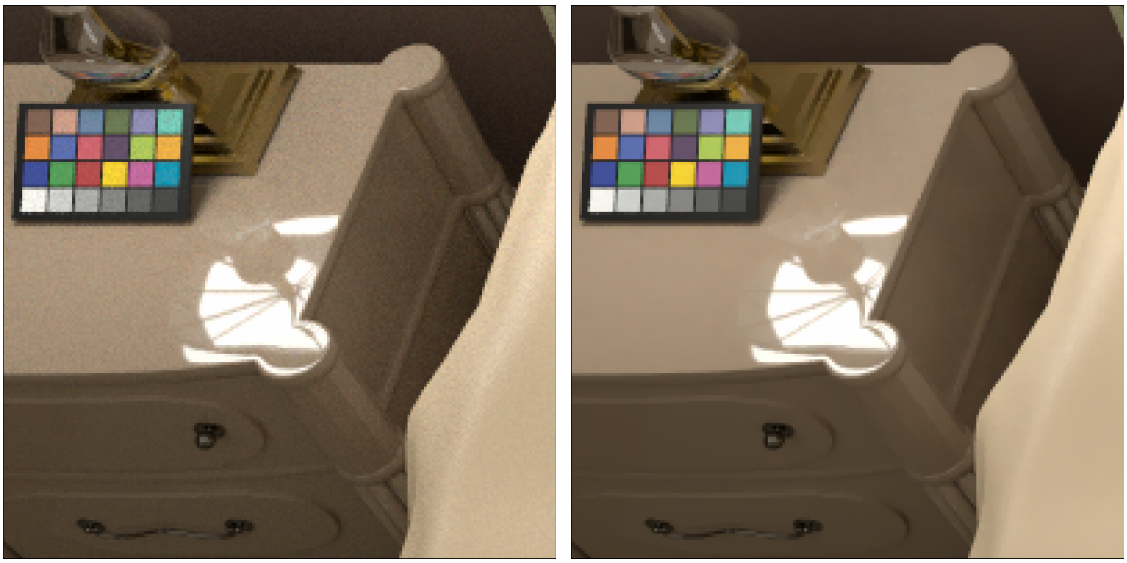


Figure 4.1. *Relative R/G B/G error with and without the guided filter. On the left side, the difference is shown between the sum of two 500 rays used single illuminant images and the same scene rendered with mixed illuminated 5000 rays. On the right side, the same experiment is done after the guided filter is applied with 5000 rays used guide image on the single illuminant images.*

4.3 Auto exposure control

Every scene is unique, material surfaces and used illuminants vary as well as the strength of the illumination. Since only the ray tracing engine of the Blender Cycles is used through



(a) *(b)*

Figure 4.2. *Edge aware smoothing effect of the guided filter. The left is a noisy image without the guided filter applied, while the right is when the guided filter is applied. It is an sRGB gamma applied image of Canon 5DSR.*

python API, the output spectrum and full spectrum images are float without a limit on the maximum pixel value. Especially pixels representing the visible light source have considerably higher values than the rest of the scene. Therefore, keeping the image signal linear by normalizing with respect to the highest pixel value does not work and makes the image very dim. This is why a linear Auto Exposure Control (AEC) method that pushes the histogram with certain histogram statistics is used [64]. It uses the Equation 4.2 to define histogram statistics and calculate a gain value to linearly change the whole image histogram. According to the histogram of all color components of the raw image, M_{cur} refers to the mean of the histogram, H_{cur} is the bright end of the histogram, H_{tgt} is the target level of the bright end of the histogram and $[M_{min}, M_{max}]$ refer to the range which M_{cur} is allowed to be once the gain is applied. Therefore gain is the ratio of H_{tgt} to the M_{cur}. It is explained in the paper that in spite of the raw image, white balance and CCM applied image can be used as well due to a better representation of the final image signal [7]. As a result of the statistics calculation, a gain value is found to multiply the whole image. Figure 12 shows an example histogram change for a high contrast scene before and after AEC is applied.

$$M_{tgt} = \min(\max(M_{cur} \cdot \frac{H_{tgt}}{H_{cur}}, M_{min}), M_{max}) \tag{4.2}$$

Two different versions of the AEC algorithm are used in this study. In one of them, a constant M_{min} value is chosen, and in the second one, an adaptive M_{min} is selected depending on the percentage of the saturated pixels over the whole image. Since light

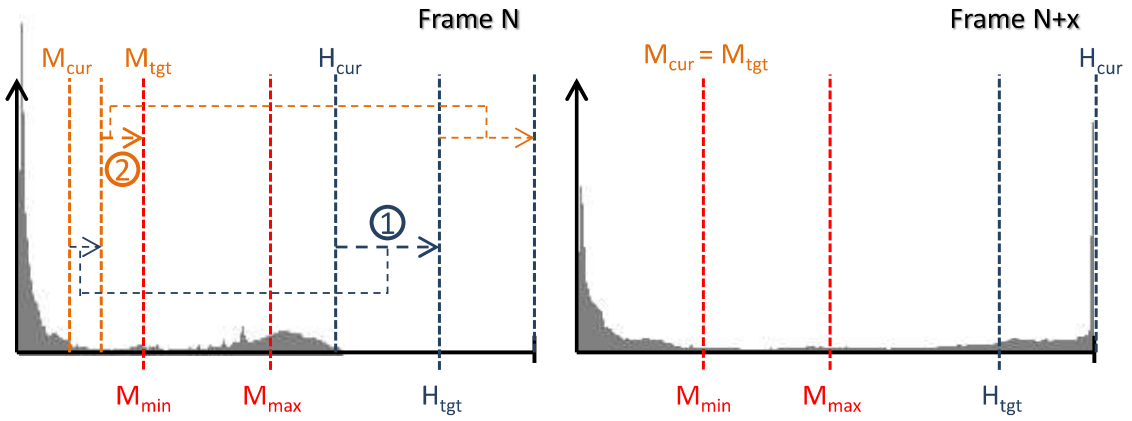


Figure 4.3. *AEC algorithm applied on a high contrast scene and the change in histogram. Obtained from [7].*

source pixels have much higher values, they are likely to saturate, but if the saturation area is relatively constant with different M_{min} values, a higher M_{min} can be used to make the image look brighter. Adaptive M_{min} is calculated with the highest gradient change point of a M_{min} with respect to the percentage of saturated pixel area over the whole image. This adaptive AEC is used only for visualization purposes since some scenes were not getting bright enough even when sRGB gamma is applied. Since the calculated gain is applied to the whole image, the linearity of the image signal is not changing. This is important for the physical realism of the image signal. Thanks to the AEC algorithm, the floating image can be turned to use the camera module raw response pixel values that vary between 0 and $2^{bit\_depth} - 1$ depending on the bit depth of the camera module.

As a result of implementing the whole pipeline, as shown in Figure 3.1, full-spectrum single illuminant images, spatially varying mixed illuminated images, corresponding white balance gain map, and output gamma applied sRGB image can be acquired. The advantage of spectral ray tracing for mixed illuminated color constancy that is shown in this thesis is that the method can be applied for an unlimited number of scenes as long as its 3D design is done. This repetition helps to create training data for learning-based algorithms. Another advantage is that generated full spectrum images can be transformed to the raw response of any camera module as long as the relative camera sensor spectral sensitivity is known. Therefore, the same dataset becomes available for different camera module based color constancy algorithm design. Figure 4.4 shows some example scenes from the dataset. On the left column, the gamma applied raw image of Sony IMX135 is shown, the middle column is the pixel-wise white balance gain map, and the right column is the sRGB image with gamma applied for visualization purposes.

In this study, 6 different 3D Blender scenes are created for mixed illumination scene creation. In each scene, 7 different virtual camera positions are predefined for spectral rendering. Since used illuminants will vary for the same camera position, the dataset does not become repetitive. This is due to raw pixel values varying for different illumination. In

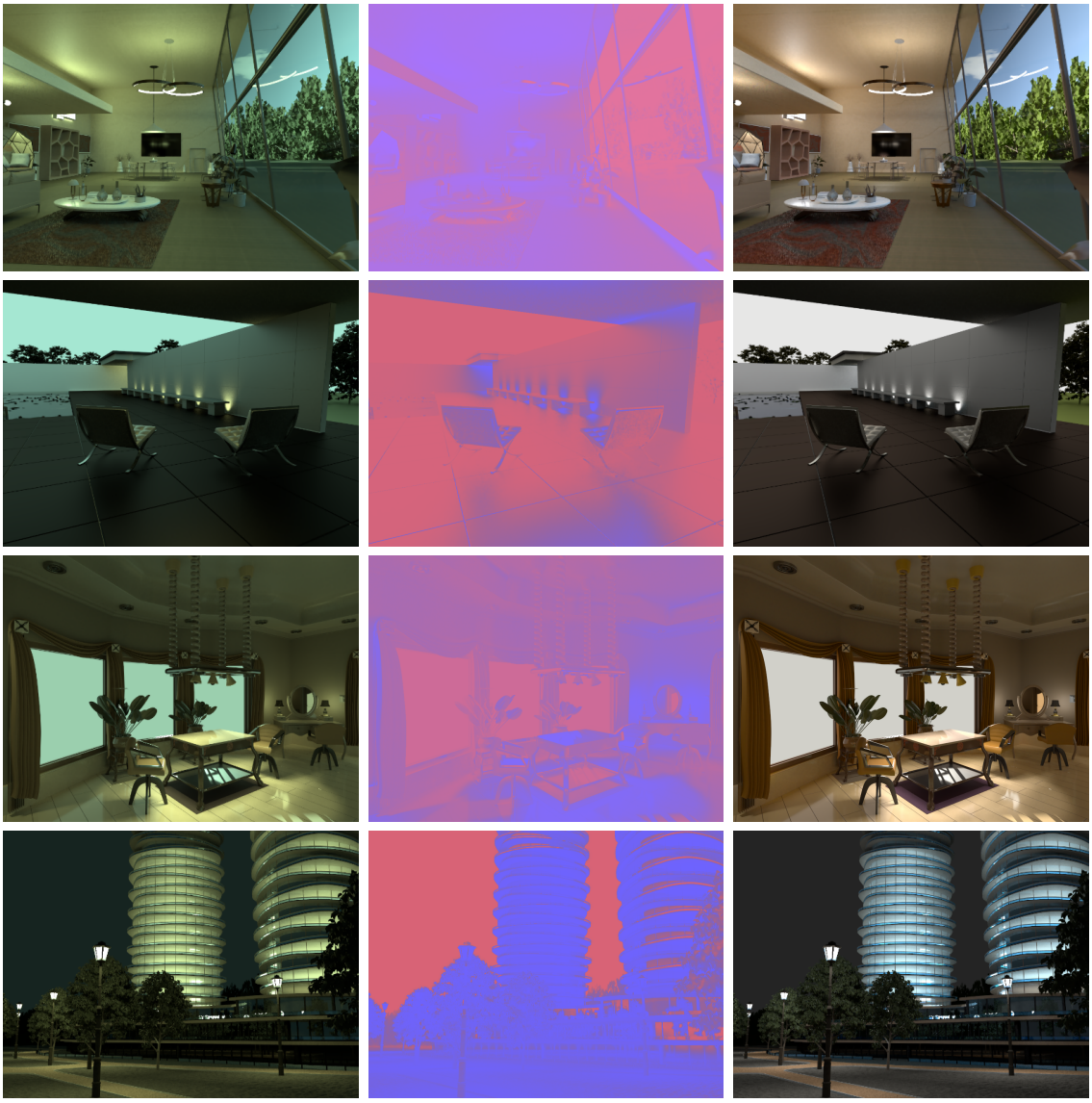


Figure 4.4. *Example mixed illuminated scenes generated with spectral ray tracing method. Left column is the gamma applied raw image of Sony IMX135, middle column is the pixel-wise white balance gain map and right column is the sRGB image with gamma applied for visualization purposes.*

total, 406 single illuminant images are generated. With a combination of them from the same virtual camera position in the same scene in an additive way, 1015 spatially varying mixed illuminated images with their corresponding ground truth white balance gain map are created. These 1015 images can be converted into raw responses of different camera sensors. The dataset is not only enough for a pixel-wise learning-based algorithm design, but the scalability of the spectral ray tracing method opens the way to generate more images.

5. VALIDATION AND TESTING

In this chapter, validation of the spectral ray tracing method as well as testing of the generated images with some publicly available algorithms is shown. Firstly, in Section 5.1, validation of the spectral ray tracing method is explained by comparing an image from real life to its compatible match that is generated with spectral ray tracing. The color reproduction of spectral ray tracing is researched. Secondly in Section 5.2 some of the spatially varying mixed illuminated images are tested on algorithms explained in Section 2.3 that can perform pixel-wise prediction.

5.1 Validation of spectral ray tracing

The idea behind the validation of spectral ray tracing is to measure the physical realism of the method and quantitatively report it. The experiment is done by comparing 2 images. The first one is an image of the Color Checker Classic[1] inside a SpectraLight QC[2] light booth taken with a Canon 5DSR camera. The second image is the 3D representation of the same scene in Blender rendered with spectral ray tracing. The measurements from the laboratory provide the inputs necessary for spectral ray tracing. In Figure 5.1, the left image shows the SPD measurement made by Konica Minolta CL-70F. The light booth offers 6 different light sources, and measurements of SPD of each light are done with a 5 minute waiting time in between for the stabilization of illumination. The waiting time for D65 was 15 minutes. On the right side, the spectral reflectance measurement of each patch of Color Checker Classic is shown. The measurement is made by Konica Minolta CM-26dG and repeated for each patch gently. Spectral reflectance measurements also included gray surfaces of the light booth. In addition to these, relative camera sensor spectral sensitivity is needed for Canon 5DSR. Since there is a sample-to-sample difference between the cameras of the same model, the spectral response of the camera that took the images in the lab is used in the experiment, and the sample-to-sample difference is taken into account.

6 images are taken in the light booth in complete darkness with the help of the camera timer. 6 different available light sources in the light booth are D65, U30, A, CWD, TL84,

[1] https://www.xrite.com/categories/calibration-profiling/colorchecker-classic
[2] https://www.xrite.com/categories/light-booths/spectralight-qc

(a) *(b)*

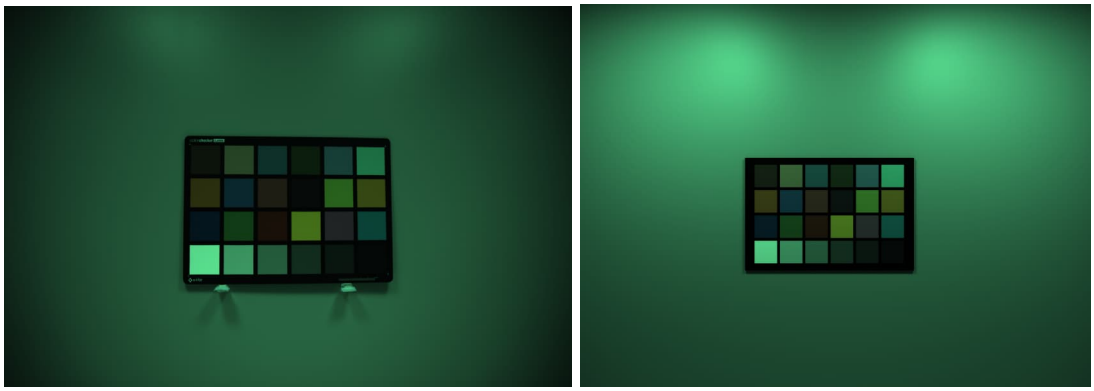
Figure 5.1. *Conducted spectral measurements in the lab that is necessary for the validation experiment. On the left, the spectral power distribution measurement of 6 available light sources with Konica Minolta CL-70F is shown. On the right, spectral reflectance measurement of all patches of Color Checker Classic with Konica Minolta CM-26dG is shown.*

and H2. They cover a good range of CCT values. Therefore, the comparison can give good coverage of understanding under different illuminants. These images taken by the camera in the lab are applied pre-processing steps that are needed before the white balance gain is applied because non-idealities in the raw data should be addressed [7]. These applied steps are black level correction and lens shading correction. Black level correction is the removed offset from the raw camera data due to being completely dark thanks to the dark current as a result of thermal energy [7]. Lens shading correction is fixing the effect of the radial shape of the lens, which causes artificial vignetting and non-uniform color and intensity distribution. With the help of these two steps, the raw image is ready to be used in the white balance step.

In comparison, 6 images are generated with corresponding relative SPDs of illuminants according to the measurement. Figure 3.2 shows the spectrum images of the laboratory scene for illuminant D65. Once it is completed, generated full-spectrum images are turned into the raw response of Canon 5DSR with the relative camera sensor spectral sensitivity of the camera in the laboratory. Figure 5.2 shows the raw images that are comparable. On the left, a reference raw image taken in the lab under illumination D65 with Canon 5DSR camera is shown. While on the right, generated image with the spectral ray tracing method for the same illuminant and same camera module is shown. If spectral ray tracing does create physical realism, the raw responses of color patches are expected to be the same. The comparison of each color checker patch is made with 3 different metrics. These are $Error_{angular}$, $Relative_{R/G,B/G}$ [45] and ΔE_{00} [65]. $Error_{angular}$ and $Relative_{R/G,B/G}$ are calculated on raw image color responses while ΔE_{00} is calculated between the 2 sRGB colors through their Lab color space representatives. $Relative_{R/G,B/G}$ calculation is shown in Equation 4.1 before in Section 4.2 with unit as percentage, $Error_{angular}$ with a unit of angular degree is shown in Equation 5.1 below where $\vec{c}_{R/G,B/G,ref}$ represents

color patch raw values of the reference image and $\vec{c}_{R/G,B/G,gen}$ represents color patch raw values of the generated image. In addition to the raw image signal comparison, images are applied the same white balance gain, same CCM, and gamma correction in order to compare the final color appearance difference with ΔE_{00} error. However, it is known that the brightness of the color affects the ΔE_{00} error, and equal brightness is not aimed by the spectral ray tracing since inputs are relative. Therefore, brightness is equalized after CCM and before gamma correction. Brightness equalization is done by equalizing the green channel average of all color checker patches as a group. After a single brightness gain is calculated and applied, gamma correction is performed.

$$Error_{angular} = cos^{-1}(\vec{c}_{R/G,B/G,ref}, \vec{c}_{R/G,B/G,gen}) \tag{5.1}$$



(a) ***(b)***

Figure 5.2. *Reference laboratory image and generated image with spectral ray tracing. On the left, Canon 5DSR raw image from the laboratory under illumination D65 is shown. On the right side, generated image with spectral ray tracing method for the same illuminant.*

In the comparison of the reference laboratory image and generated image of the Color Checker Classic, each color patch is individually evaluated. This leads to 24 different error values for each color metric. The numbering of the patches is in the order from 1 to 24 as the first one is the left upper corner while the last one is the black patch on the bottom right corner. The number increases going towards the right and continues from the left side in the next row. For instance, the first near achromatic white patch is represented with the number 19. With these prior definitions for the error metric and color patch number, results are shown below in the Table 5.1 for the scene illuminated by the D65.

Results are evaluated mainly with ΔE_{00} because it relates to the actual perceived color difference. It is researched that ΔE_{00} error until 3 is most likely to be acceptable for the average person, while errors over 9 are most likely to cause not acceptable color reproduction [45]. Looking at Table 5.1 for D65 and rest of other 5 illuminants in Appendix A,

Table 5.1. *Comparison of reference color patches to generated color patches under illumination D65.*

Patch Number	ΔE_{00}	$Relative_{R/G,B/G}\,[\%]$	$Error_{angular}\,[^{\circ}]$
1	5.52	4.95	1.78
2	5.06	1.87	0.66
3	6.11	2.57	0.72
4	3.98	5.45	1.28
5	4.38	1.35	0.20
6	4.19	2.25	0.49
7	2.47	4.57	1.58
8	3.11	5.75	1.47
9	2.86	8.69	2.84
10	5.73	10.10	2.87
11	1.83	4.97	0.92
12	2.19	3.59	1.13
13	3.42	9.92	2.47
14	3.03	7.77	1.38
15	1.99	15.47	5.20
16	2.67	5.49	1.44
17	2.12	7.89	2.12
18	0.82	3.32	1.05
19	5.13	1.45	0.497
20	4.57	1.44	0.50
21	4.57	1.50	0.50
22	2.45	1.67	0.59
23	1.57	1.35	0.52
24	0.76	0.92	0.37

no color patch has error over 9. Table 5.2 shows the mean ΔE_{00} error for all scenes of possible illuminants as well as how many of them belong to each error group. Conducted study for the source of errors showed that rather than doing a single brightness adjustment for all color patches, brightness adjustment for each row of Color Checker Classic showed decreased ΔE_{00}. It is observed that most of the high angular error patches are colors with high spectral reflectance values around the longer wavelengths of the visible spectrum. Part of the error can be due to residual color shading. In addition, a part of the error could be coming from used measurement devices such as an illuminant meter for SPD and a spectrophotometer for spectral reflectance. The manufacturer of the Color Checker Classic does not share any original spectral reflectance data, but found spectral

Table 5.2. *DE2000 average for each scene as well as number of color patches between the error limits.*

Illuminant	Mean ΔE_{00}	Low (≤ 3)	Medium $(3 < 9)$	High (≥ 9)
D65	3.39	11	13	0
U30	3.02	11	13	0
CWF	2.47	20	4	0
TL84	2.72	16	8	0
A	2.46	18	6	0
H2	2.65	18	6	0

reflectance measurements from other resources showed that there are small variations in the representation of each patch. Also, simulation accuracy for the longest wavelengths might be degraded and further studies of this topic might be addressed in future research. All in all, it is observed that as long as the whole scene does not consist of relatively high error colors only, the physical realism is an acceptable level compared to the real world.

5.2 Testing dataset images

Although the dataset is created as a color constancy training dataset, creating a learning-based approach is not in the scope of this thesis. But it is beneficial to observe the performance of publicly available color constancy algorithms on some of the dataset images. This is why the color constancy algorithms explained in Section 2.3 are utilized in testing. But it is observed that it is not fair to test statistical methods on mixed illuminated images since they are developed for single illuminant chromaticity estimation for the whole image. In some test approaches, these statistical approaches are used in a patch-wise manner where the image is divided into equal patches, and the algorithm makes predictions for each patch individually [16], [66]. But this results in considerable color casts even between the patches since the algorithm is not meant to create a correlation between them. Therefore, two learning-based algorithms that make pixel-wise predictions are utilized in the testing of some of the dataset images created by the spectral ray tracing method explained in this paper. These algorithms are mixed illumination correction by Afifi et al. [57] and U-net of LSMI [16]. All testing images are Nikon D810 responses since it is the only common camera module between this study and LSMI. The algorithm explained in [57] works with sRGB data, that is why sRGB output of Nikon D810 is used as input. It is accepted that there can be sample-to-sample differences between Nikon D810 camera modules used in this research and LSMI [16].

Mixed illumination pixel-wise white balancing is solved as a color rendering problem of the camera output sRGB image [57]. The algorithm details is explained in Section 2.3.6. Thanks to the implementation of the algorithm being openly available, testing was done

by following the steps explained by the authors. All possible white balance settings are used since images created by spectral ray tracing are quite wide from the warm end of the Planckian locus until the cool end. Ensembling and edge-aware smoothing is enabled, therefore, 384x384 is used as the input size since it is recommended by the authors. Patch size of 64 is used since it is reported that it resulted in less error on the mixed illuminated synthetic dataset. The input sRGB images for this network needs to be wrongly white-balanced mixed illuminated camera module output. Therefore, from the dataset created by this thesis, a weighted average of the white points of existing illuminants is calculated and used for the whole image, which causes color casts. In addition, a weighted average of CCM matrices are calculated, and a single CCM is applied to reach the sRGB image. Later, gamma correction is applied since camera module output would be gamma applied. It is expected that these color casts are eliminated or decreased by the network. The wrongly white-balanced image, which is also the input for this algorithm, can be seen in Figure 5.3 in the first column, while the network predictions are shown in the second column.

Secondly, U-net of LSMI is used in the testing of the images due to being publicly available [16]. The algorithm and dataset details are explained in Section 2.3.5. The algorithm has 3 different neural network models, each trained for a different camera sensor. Since Nikon D810 is also utilized in this thesis, testing is done on the Nikon D810 model. Testing required a 14-bit Nikon D810 raw image, ground truth white-balanced image, and coefficient map. Required files are created and given as input to the network. Testing is completed with 256x256 image size since the authors developed the algorithm in that way. Testing is done in the l, u, v color space since U-net is trained on it. White balance gain map prediction is saved from the network output, and it is up-scaled to 720x960 with a bi-cubic interpolation so that it can be gained on the input raw image directly. Once the image is white balanced, the same CCM and gamma correction as previously explained is applied to output an sRGB image. The resulted sRGB images can be seen in Figure 5.3 on the third column.

The ground truth sRGB image is reached by applying the correct white balance gain map on the raw Nikon D810 image, and then the same CCM matrix and gamma function is used as in other methods for equal comparison. As the error metric, ΔE_{00_grid} is used, which is a modified version of the ΔE_{00}, developed and used internally by Xiaomi. In the Section 5.1, since color patches are compared individually, ΔE_{00} error between patch colors were enough. But, once the comparison is for the whole image, a grid-based approach is used. This is achieved in Xiaomi by dividing the image into 12x16 grids, where the average sRGB color of each grid is compared with its equivalent. Once the ΔE_{00} for each grid is calculated between the grid colors, worst 25% is averaged for defining a single error value for the whole image. The worst 25% is selected in order to give more weight to the color artifacts if they exist in the scene. Calculated worst 25%

error values compared to the ground truth sRGB image can be seen below in the Figure 5.3 on the right below of each image. The details of the ΔE_{00_grid} is explained in the Appendix B.

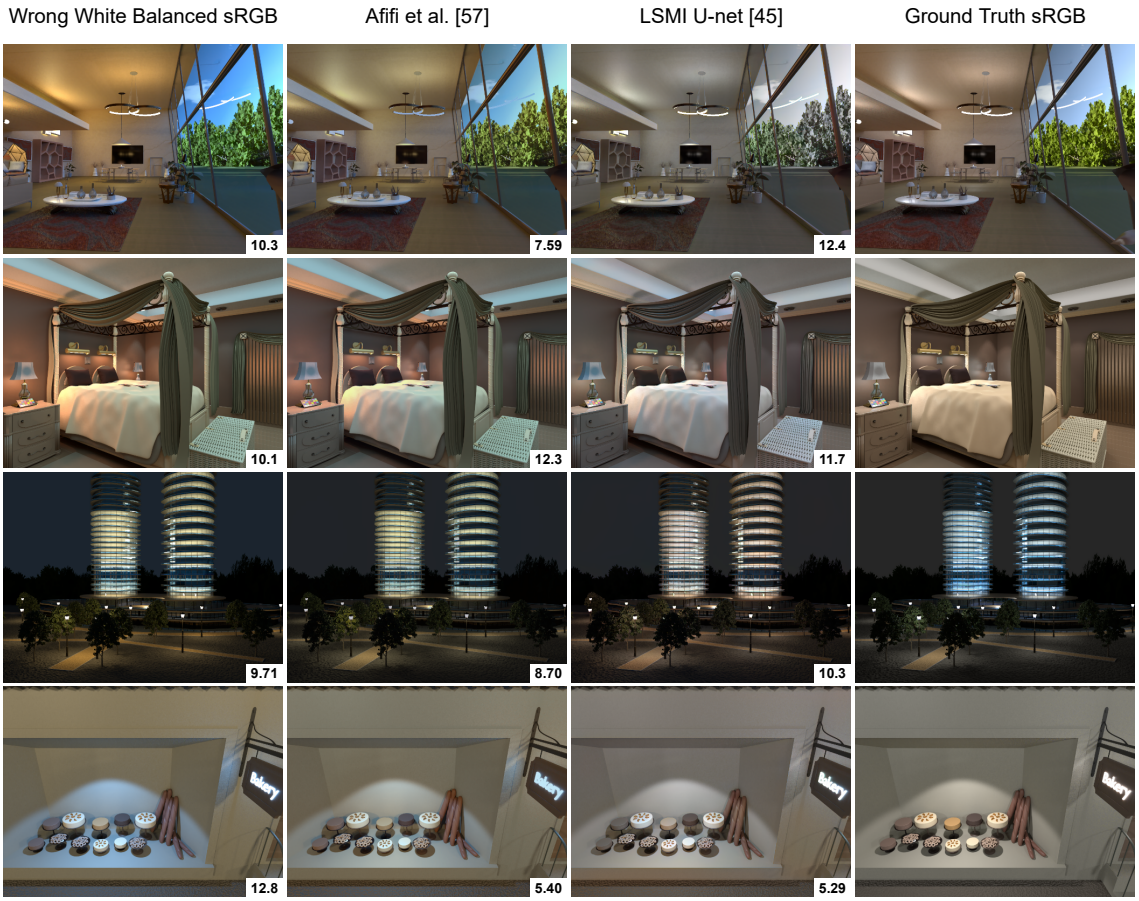


Figure 5.3. *Predictions of the learning-based algorithms. The left column is the sRGB image with a single white balance gain applied to the whole image, the second column is the prediction by the algorithm proposed by Afifi et al. [57], the third column is the prediction by the LSMI U-net [16], and the last column is the ground truth correctly white balanced image. All sRGB images are gamma applied.*

Results indicate that the color casts caused by the wrong white balancing can be observed. Especially in the bedroom scene, the ceiling has a bluish color cast, and the light nearby the bed has reddish. In addition, the color of the bed linen is wrong in both of the predictions. The method proposed by Afifi et al. performed relatively well for the living room and bakery scenes, while LSMI U-net only performed relatively good in the bakery scene. This relative goodness indicates a medium error, ranging between 3 and 9 ΔE_{00} error [45]. No prediction has less than 3 ΔE_{00} error, which gives the understanding that the color difference as a result of these algorithms would be either somewhat acceptable or not acceptable by an average observer. The interpretation of the results indicates the necessity of the dataset created by this thesis. Although more extensive testing of the networks is required, it can be said that the improvement channels are still open with more data.

6. CONCLUSION AND FUTURE WORK

This thesis proposes a training dataset creation method for computational color constancy in spatially varying mixed illuminated images. The orders of magnitude more laborious annotation process requirement for mixed illuminated scenes compared to single illuminated scenes was the main drawback of the algorithm development in the field. Although the number of datasets is growing, the improvement and contribution channels are still open. In this study, spectral ray tracing for physically realistic image generation is proposed as a spatial color constancy dataset creation method. Physical realism is aimed since a learning-based algorithm developed using the dataset created with this method will be used and tested on real-life images and maybe even deployed on a consumer camera. This physical realism is achieved by creating a 3D scene in Blender and defining each material and illuminant with its spectral representation rather than a trichromatic one. Material spectral reflectances are taken from In Situ database, while illuminant SPDs are chosen from the CIE light sources library. Rendering the scene in its spectral form rather than the trichromatic form helps reaching closer to physical realism of the real world. The full spectrum images generated by Blender Cycles rendering engine can be turned to any camera's raw response as long as the camera module spectral response is known. Due to predefining illuminants in the scene and camera module used in the creation of the raw image, a pixel-wise white balance gain can be calculated. The raw camera response and the gain map are fundamentally needed for spatial color constancy training data.

The evaluation of spectral ray tracing in the real world is done by creating the same controlled scene in both environments. A color checker scene in a controlled laboratory environment was created both in real life and in 3D. Measurements taken in the laboratory are given as inputs to the spectral ray tracing, and synthetic images are generated, real images are captured with Canon 5DSR. It is shown that no color patch in the Color Checker Classic has high ΔE_{00} error, and the average of all errors is most likely to be acceptable for an average observer. It is researched that errors may come from the measurement device, the camera's infra-red cut-off filter response, or residual color shading. Because high angular error values mainly come from the color patches that have higher values around the end of the spectrum that represent reddish colors. Overall, it is understood that the colors produced by the spectral ray tracing method are at an acceptable level for physical realism.

In this thesis, 6 different 3D Blender scenes are created, with each having 7 different virtual camera positions inside. 406 single illuminant images under different illuminations are rendered in total. Due to Blender using the same random seed for sending rays, mixed illuminated images are acquired by summation of the same camera angle single illuminant images. This allows combining single illuminant images with being used multiple times. In total, 1015 spatially varying mixed illuminated images are generated with their ground truth white balance gain map in 3 different camera modules, Canon 5DSR, Sony IMX135, and Nikon D810.

For further development, the reason of relatively bigger errors when rendering colors with high values in the longer wavelengths of the visible spectrum can be researched as well as how to reduce those errors. Additionally, it can be researched if different sensors rather than Canon 5DSR show similar error results or not for the Color Checker evaluation scene. Also, rendering is completed 3 wavelengths at a time in this thesis, causing to repeat the rendering 21 times. The rendering algorithm can be improved to do the whole spectral rendering at once, resulting shorter rendering times. In this thesis, the created dataset opens the way for learning-based pixel-wise white balance algorithm development to solve unrealistic color casts in mixed illuminated scenes due to a single white point gained on the whole image. Also, it can be used as additional data by turning the created full-spectrum images into any wanted camera sensor's response with a simple multiplication. In addition, there is research on camera module independent color constancy. Since the dataset can be transformed to various camera module responses, the created dataset can be used either in the training or testing of such algorithms. In each future implementation, if more data is needed, it can be generated with spectral ray tracing.

REFERENCES

[1] M. Pharr, W. Jakob, and G. Humphreys, *Physically Based Rendering: From Theory to Implementation*. Morgan Kaufmann, 2016.

[2] J. Peddie, *Ray Tracing: A Tool for All*. Springer International Publishing, 2019.

[3] M. van de Ruit and E. Eisemann, "A multi-pass method for accelerated spectral sampling," *Computer Graphics Forum*, vol. 40, no. 7, pp. 141–148, 2021.

[4] A. Wilkie, S. Nawaz, M. Droske, A. Weidlich, and J. Hanika, "Hero wavelength spectral sampling," *Computer Graphics Forum*, vol. 33, no. 4, pp. 123–131, 2014.

[5] M. D. Fairchild, *Color appearance models*, 3rd ed., ser. The wiley-IS&T series in imaging science and technology. Chichester, West Sussex, U.K: Wiley, 2013.

[6] G. Wyszecki and W. S. (S. Stiles, *Color science*, eng, 2nd ed., Wiley classics library ed., ser. Wiley classics library. New York: John Wiley & Sons, 2000.

[7] J. Nikkanen, "Computational color constancy in mobile imaging," Ph.D. dissertation, Tampere University of Technology, 2013.

[8] G. Buchsbaum, "A spatial processor model for object colour perception," *Journal of the Franklin Institute*, vol. 310, no. 1, pp. 1–26, 1980.

[9] J. van de Weijer and T. Gevers, "Color constancy based on the grey-edge hypothesis," in *IEEE International Conference on Image Processing 2005*, vol. 2, 2005, pp. II–722.

[10] H.-K. Lam, O. Au, and C.-W. Wong, "Automatic white balancing using adjacent channels adjustment in rgb domain," in *2004 IEEE International Conference on Multimedia and Expo (ICME) (IEEE Cat. No.04TH8763)*, vol. 2, 2004, 979–982 Vol.2.

[11] J. T. Barron, "Convolutional color constancy," *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 379–387, 2015.

[12] G. Finlayson and E. Trezzi, "Shades of gray and colour constancy," Proceedings of the 12th Color Imaging Conference, Jan. 2004, pp. 37–41.

[13] J. T. Barron and Y. Tsai, "Fast fourier color constancy," *CoRR*, vol. abs/1611.07596, 2016.

[14] Y. Hu, B. Wang, and S. Lin, "Fc^4: Fully convolutional color constancy with confidence-weighted pooling," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 330–339.

[15] J. Nikkanen, "Color constancy by characterization of illumination chromaticity," eng, *Optical Engineering*, vol. 50, no. 5, pp. 057 204–057 204, 2011.

[16] D. Kim, J. Kim, S. Nam, *et al.*, “Large scale multi-illuminant (lsmi) dataset for developing white balance algorithm under mixed illumination,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2410–2419.

[17] X. Hao and B. Funt, “A multi-illuminant synthetic image test set,” *Color Research and Application*, vol. 45, no. 6, pp. 1055–1066, Aug. 2020.

[18] F. Laakom, J. Raitoharju, J. Nikkanen, A. Iosifidis, and M. Gabbouj, “Intel-tau: A color constancy dataset,” *IEEE Access*, vol. 9, pp. 39 560–39 567, Mar. 2021.

[19] T. Whitted, “An improved illumination model for shaded display,” *Commun. ACM*, vol. 23, pp. 343–349, 1980.

[20] H. Landis, “Production-ready global illumination,” in *Proceedings of the RenderMan in Production, SIGGRAPH Course*, New York, USA: ACM, 2002, pp. 87–102.

[21] S. Zhukov, A. Iones, and G. Kronin, “An ambient light illumination model,” in *Rendering Techniques ’98*, Vienna: Springer Vienna, 1998, pp. 45–55.

[22] T. Akenine-Mller, E. Haines, and N. Hoffman, *Real-Time Rendering, Fourth Edition*, 4th. USA: A. K. Peters, Ltd., 2018.

[23] E. Veach, “Robust monte carlo methods for light transport simulation,” AAI9837162, Ph.D. dissertation, Stanford, CA, USA, 1998.

[24] S. Marschner and P. Shirley, *Fundamentals of Computer Graphics, Fourth Edition*, 4th. USA: A. K. Peters, Ltd., 2016.

[25] P. Shirley and R. K. Morley, *Realistic Ray Tracing*, 2nd ed. USA: A. K. Peters, Ltd., 2003.

[26] J. F. Hughes, A. van Dam, M. McGuire, *et al.*, *Computer Graphics: Principles and Practice*, 3rd ed. Upper Saddle River, NJ: Addison-Wesley, 2013.

[27] J. T. Kajiya, “The rendering equation,” in *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH ’86, New York, NY, USA: Association for Computing Machinery, 1986, pp. 143–150.

[28] D. S. Immel, M. F. Cohen, and D. P. Greenberg, “A radiosity method for non-diffuse environments,” in *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH ’86, New York, NY, USA: Association for Computing Machinery, 1986, pp. 133–142.

[29] M. Romaniuk and P. Nowak, *Monte Carlo methods: theory, algorithms and applications to selected financial problems*. Institute of Computer Science Polish Academy of Sciences, Dec. 2015.

[30] C. F. Borges, “Trichromatic approximation for computer graphics illumination models,” *SIGGRAPH Comput. Graph.*, vol. 25, no. 4, pp. 101–104, Jul. 1991.

[31] V. Petitjean, P. Bauszat, and E. Eisemann, “Spectral gradient sampling for path tracing,” in *Computer Graphics Forum (Proceedings of EGSR)*, 2018.

[32] M. Radziszewski and W. Alda, “An improved technique for full spectral rendering,” *Journal of WSCG*, vol. 17, Jan. 2009.

[33] “Cie (international commission on illumination) homepage”. (), [Online]. Available: https://cie.co.at/ (visited on 05/08/2022).

[34] G. F. Evans and M. D. McCool, “Stratified wavelength clusters for efficient spectral monte carlo rendering,” in *Proceedings of the Graphics Interface 1999 Conference, June 2-4, 1999, Kingston, Ontario, Canada*, Jun. 1999, pp. 42–49.

[35] G. Johnson and M. Fairchild, “Full-spectral color calculations in realistic image synthesis,” *IEEE Computer Graphics and Applications*, vol. 19, no. 4, pp. 47–53, 1999.

[36] R. S. Berns, *Billmeyer and Saltzman’s principles of color technology*, eng, Fourth edition. Hoboken, NJ: Wiley, 2019.

[37] J. Hernández-Andrés, R. L. Lee, and J. Romero, “Calculating correlated color temperatures across the entire gamut of daylight and skylight chromaticities,” *Appl. Opt.*, vol. 38, no. 27, pp. 5703–5709, Sep. 1999.

[38] L. T. Maloney and J. A. Schirillo, “Color constancy, lightness constancy, and the articulation hypothesis,” English (US), *Perception*, vol. 31, no. 2, pp. 135–139, 2002.

[39] B. E. Bayer, “Color imaging array,” US Patent 3 971 065, 1975.

[40] D. Litwiller, “Cmos vs. ccd: Maturing technologies, maturing markets,” *Photonics Spectra*, vol. 39, pp. 54–60, Aug. 2005.

[41] J. R. Janesick, *Scientific Charge-coupled Devices*, ser. Press Monographs. Society of Photo Optical, 2001.

[42] L. Zhang, Y. Jin, L. Lin, J. Li, and Y. Du, “The comparison of ccd and cmos image sensors,” in *2008 International Conference on Optical Instruments and Technology: Advanced Sensor Technologies and Applications*, International Society for Optics and Photonics, vol. 7157, SPIE, 2009, pp. 231–235.

[43] J. von Kries, “Influence of adaptation on the effects produced by luminous stimuli,” *Source of Color Science*, vol. 9, pp. 109–119, 1970.

[44] M. Stokes, M. Anderson, S. Chandrasekar, and R. Motta. “A standard default color space for the internet - srgb,” (1996), [Online]. Available: https://www.w3.org/Graphics/Color/sRGB (visited on 05/26/2022).

[45] J. Nikkanen, T. Gerasimow, and L. Kong, “Subjective effects of white-balancing errors in digital photography,” *Optical Engineering*, vol. 47, no. 11, pp. 1–15, 2008.

[46] A. S. Baslamisli, “Camera sensor invariant auto white balance agorithm weighting,” M.S. thesis, Tampere University of Technology, Tampere, Finland, 2016.

[47] L. Murmann, M. Gharbi, M. Aittala, and F. Durand, “A dataset of multi-illumination images in the wild,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2019.

[48] M. Bleier, C. Riess, S. Beigpour, *et al.*, “Color constancy and non-uniform illumination: Can existing algorithms work?” In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, 2011, pp. 774–781.

[49] S. Beigpour, C. Riess, J. Weijer, and E. Angelopoulou, “Multi-illuminant estimation with conditional random fields,” *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, Jan. 2014.

[50] A. Gijsenij, R. Lu, and T. Gevers, “Color constancy for multiple light sources,” *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, vol. 21, pp. 697–707, Aug. 2011.

[51] K. Barnard, V. Cardei, and B. Funt, “A comparison of computational color constancy algorithms. i: Methodology and experiments with synthesized data,” *IEEE Transactions on Image Processing*, vol. 11, no. 9, pp. 972–984, 2002.

[52] J.-y. Huo, Y.-l. Chang, J. Wang, and X.-x. Wei, “Robust automatic white balance algorithm using gray color points in images,” *IEEE Transactions on Consumer Electronics*, vol. 52, no. 2, pp. 541–546, 2006.

[53] W. Xiong, B. Funt, L. Shi, *et al.*, “Automatic white balancing via gray surface identification,” Proceedings of the Fifteenth IST Color Imaging Conference, Albuquerque, Nov. 2007.

[54] B. Li, D. Xu, W. Xiong, and S. Feng, “Color constancy using achromatic surface,” *Color Research & Application*, vol. 35, no. 4, pp. 304–312, 2010.

[55] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” *CoRR*, vol. abs/1505.04597, 2015.

[56] M. Gharbi, J. Chen, J. T. Barron, S. W. Hasinoff, and F. Durand, “Deep bilateral learning for real-time image enhancement,” *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, p. 118, 2017.

[57] M. Afifi, M. A. Brubaker, and M. S. Brown, “Auto white-balance correction for mixed-illuminant scenes,” *CoRR*, vol. abs/2109.08750, 2021.

[58] D. Fourure, R. Emonet, É. Fromont, D. Muselet, A. Trémeau, and C. Wolf, “Residual conv-deconv grid network for semantic segmentation,” *CoRR*, vol. abs/1707.07958, 2017.

[59] S. Niklaus and F. Liu, “Context-aware synthesis for video frame interpolation,” *CoRR*, vol. abs/1803.10967, 2018.

[60] M. Afifi, B. Price, S. Cohen, and M. S. Brown, “When color constancy goes wrong: Correcting improperly white-balanced images,” in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 1535–1544.

[61] D. Wüller, “In situ measured spectral radiation of natural objects,” in *Color Imaging Conference*, 2009.

[62] F. Kainz, R. Bogart, and D. Hess, “Openexr image file format,” *ACM SIGGRAPH ’03 Technical Sketches*, 2003.

[63] E. S. L. Gastal and M. M. Oliveira, “Domain transform for edge-aware image and video processing,” *ACM TOG*, vol. 30, no. 4, 69:1–69:12, 2011, Proceedings of SIGGRAPH 2011.

[64] J. Nikkanen and K. Ossi, “Method and system in digital imaging for adjusting exposure and a corresponding device,” US Patent 7 474 847, 2009.

[65] “Publication cie 142-2001: Improvement to industrial colour-difference evaluation,” CIE (International Commission on Illumination), Tech. Rep. ISBN 3901906088, 2001.

[66] A. Gijsenij, R. Lu, and T. Gevers, “Color constancy for multiple light sources,” *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, vol. 21, pp. 697–707, Aug. 2011.

APPENDIX A: VALIDATION RESULTS OF SPECTRAL RAY TRACING

In Section 5.1, validation of spectral ray tracing experiment is conducted. The aim was to measure the color reproduction accuracy between the real-life image and the image generated with spectral ray tracing under 6 different illuminants. The comparison of each color checker patch is shown for illuminant D65 in Section 5.1. Errors of the 5 other scenes under different illuminants are shown below.

Table A.1. *Comparison of reference color patches to generated color patches under illumination U30.*

Patch Number	ΔE_{00}	$Relative_{R/G,B/G}\,[\%]$	$Error_{angular}\,[^\circ]$
1	4.08	7.80	2.49
2	4.00	6.38	2.03
3	3.89	0.89	0.29
4	2.68	2.49	0.73
5	3.05	3.46	1.19
6	3.02	2.38	0.62
7	2.63	8.56	2.54
8	1.63	2.94	0.96
9	2.46	12.37	3.49
10	3.64	7.98	2.43
11	2.46	2.01	0.61
12	2.52	4.67	1.45
13	2.07	7.43	2.53
14	2.86	4.82	1.06
15	3.12	17.14	4.56
16	3.09	3.72	1.17

Table A.1 Continued: *Comparison of reference color patches to generated color patches under illumination U30.*

17	3.08	14.01	4.20
18	1.14	5.16	1.46
19	4.79	3.26	1.05
20	4.80	3.29	1.05
21	4.57	3.42	1.09
22	3.02	3.90	1.23
23	2.26	3.94	1.23
24	1.77	3.70	1.10

Table A.2. *Comparison of reference color patches to generated color patches under illumination CWF.*

Patch Number	ΔE_{00}	$Relative_{R/G,B/G}\,[\%]$	$Error_{angular}\,[^\circ]$
1	3.18	5.16	1.75
2	2.57	3.21	1.11
3	3.00	0.74	0.22
4	2.33	4.32	1.15
5	1.11	1.17	0.40
6	2.35	3.21	0.79
7	2.10	4.31	1.46
8	1.74	4.40	1.49
9	2.33	9.23	3.07
10	4.79	7.29	2.18
11	2.60	3.17	0.82
12	2.75	2.48	0.86
13	2.96	9.50	3.12
14	2.48	5.92	1.29
15	2.83	14.51	4.69
16	2.69	2.94	0.95

Table A.2 Continued: *Comparison of reference color patches to generated color patches under illumination CWF.*

17	2.62	9.24	2.89
18	0.89	4.83	1.59
19	3.27	2.24	0.70
20	3.17	2.27	0.71
21	2.79	2.33	0.73
22	1.72	2.44	0.78
23	1.30	2.26	0.73
24	1.74	1.55	0.47

Table A.3. *Comparison of reference color patches to generated color patches under illumination TL84.*

Patch Number	ΔE_{00}	$Relative_{R/G,B/G}\,[\%]$	$Error_{angular}\,[^\circ]$
1	3.69	7.41	2.48
2	3.22	5.54	1.86
3	3.47	0.70	0.21
4	2.35	4.70	1.21
5	2.09	2.60	0.94
6	2.32	3.28	0.81
7	2.18	8.03	2.58
8	1.81	3.64	1.16
9	2.44	12.37	4.00
10	4.15	7.25	2.17
11	2.50	3.25	0.83
12	2.61	3.84	1.25
13	2.64	8.33	2.67
14	2.63	6.78	1.39
15	3.05	18.25	5.64
16	2.67	3.70	1.18

Table A.3 Continued: *Comparison of reference color patches to generated color patches under illumination TL84.*

17	2.88	12.96	4.25
18	1.30	3.57	1.16
19	3.76	2.90	0.94
20	3.81	2.96	0.96
21	3.53	3.04	0.98
22	2.33	3.39	1.10
23	1.83	3.22	1.04
24	1.90	3.02	0.91

Table A.4. *Comparison of reference color patches to generated color patches under illumination A.*

Patch Number	ΔE_{00}	$Relative_{R/G,B/G}\,[\%]$	$Error_{angular}\,[^\circ]$
1	2.44	5.29	1.77
2	1.93	3.02	1.04
3	1.60	1.56	0.45
4	2.48	3.00	0.92
5	0.75	1.15	0.41
6	2.71	3.46	0.84
7	2.23	4.46	1.38
8	1.46	3.62	1.33
9	2.23	9.89	2.69
10	4.43	10.57	3.24
11	3.01	2.44	0.71
12	3.13	2.75	0.95
13	3.00	9.51	3.45
14	2.34	6.58	1.47
15	3.02	15.99	3.68
16	3.43	2.11	0.81

Table A.4 Continued: *Comparison of reference color patches to generated color patches under illumination A.*

17	2.97	11.90	3.33
18	2.07	9.46	2.77
19	3.21	1.45	0.50
20	2.95	1.46	0.51
21	2.37	1.44	0.51
22	2.00	1.52	0.54
23	1.51	1.36	0.48
24	1.72	0.79	0.26

Table A.5. *Comparison of reference color patches to generated color patches under illumination H2.*

Patch Number	ΔE_{00}	$Relative_{R/G,B/G}\,[\%]$	$Error_{angular}\,[^\circ]$
1	2.99	5.21	1.62
2	2.15	3.39	1.06
3	1.98	1.32	0.38
4	2.93	2.09	0.69
5	1.56	2.03	0.66
6	2.91	3.07	0.78
7	2.28	4.61	1.28
8	1.58	3.03	1.14
9	2.38	9.90	2.34
10	4.34	11.21	3.25
11	3.05	1.55	0.51
12	2.88	2.94	0.91
13	2.88	9.11	3.47
14	2.65	6.64	1.60
15	3.16	15.42	3.02
16	3.80	1.90	0.68

Table A.5 Continued: *Comparison of reference color patches to generated color patches under illumination H2.*

17	2.86	11.93	2.91
18	1.88	12.26	3.34
19	3.93	1.28	0.50
20	3.58	1.23	0.48
21	2.99	1.31	0.50
22	1.90	1.51	0.55
23	1.30	1.43	0.50
24	1.58	1.31	0.42

APPENDIX B: DE2000 GRID

ΔE_{00} is a metric developed by the CIE Technical Committee to calculate the differences between two perceived colors in regards to human visual judgment [65]. The formula of the metric is based on the CIELAB color space, and calculation is done on the L, a, b equivalent of the perceived colors. This metric is used to compare individual color patches of the Color Checker Classic in Section 5.1. However, if the aim is to compare two images, a pixel-to-pixel color comparison with ΔE_{00} is not preferred due to noise. Therefore, a modified version of the ΔE_{00} error metric is developed and used in Xiaomi. The core idea relies on understanding the areas of the image where the wrong white balance gain prediction caused the most perceived color difference. This is why the image is divided into a 12x16 grid, and the average sRGB color of each grid cell is compared with its equivalent using ΔE_{00}. This modified version of the original metric is called ΔE_{00_grid}. When calculating a single error value for the whole scene, an average of the worst 25% of the grid errors is used. This is due to giving more weight to possible color casts if they exist in the scene. Because the mean error may be at an acceptable level while there are visible color differences in just some areas of the image. The grid errors for two scenes shown in Figure 5.3 are visualized below in Figure B.1 and Figure B.2 to better understand the metric for AWB errors that utilizes ΔE_{00} for this particular use case.

The bluish and reddish color casts on the ceiling as well as the wrong color rendering of the bed linen are calculated as the biggest error grids between the prediction from Afifi et al. and the ground truth, as can be seen in Figure B.1 (a). This is different from the prediction of LSMI U-net, where most of the error comes from the bed linen and the box in front of the bed as in Figure B.2 (a). The idea of using the average of the worst 25% rather than the average of the whole grid can be understood from the skyscraper scene. In both predictions, the main high error source is the skyscraper, which causes visible color differences compared to the ground truth. It would not be a reliable idea to use 3.84 or 4.45 as ΔE_{00} error between the images. Therefore, for grid-based usage of the ΔE_{00}, worst 25% provides a healthier comparison to understand the color casts caused by the wrong white balancing.

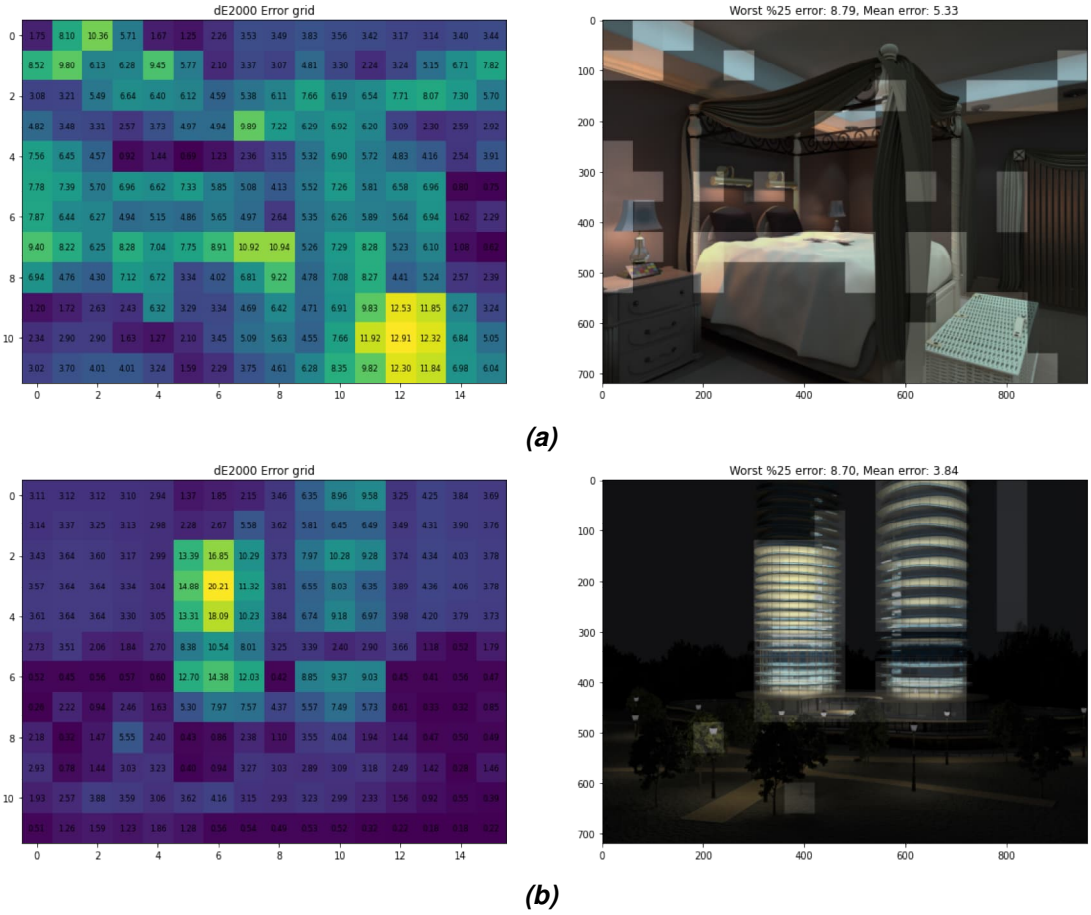


Figure B.1. *DE2000 grid error visualization for the prediction of Afifi et al. in Figure 5.3. (a) is the bedroom scene with DE2000 error for each grid shown on the left side, and worst 25% error grids are visualized brightly on the image on the right side. (b) is the same metric visualization applied to the skyscraper scene. The ground truth used in the comparison is shown in Figure 5.3.*

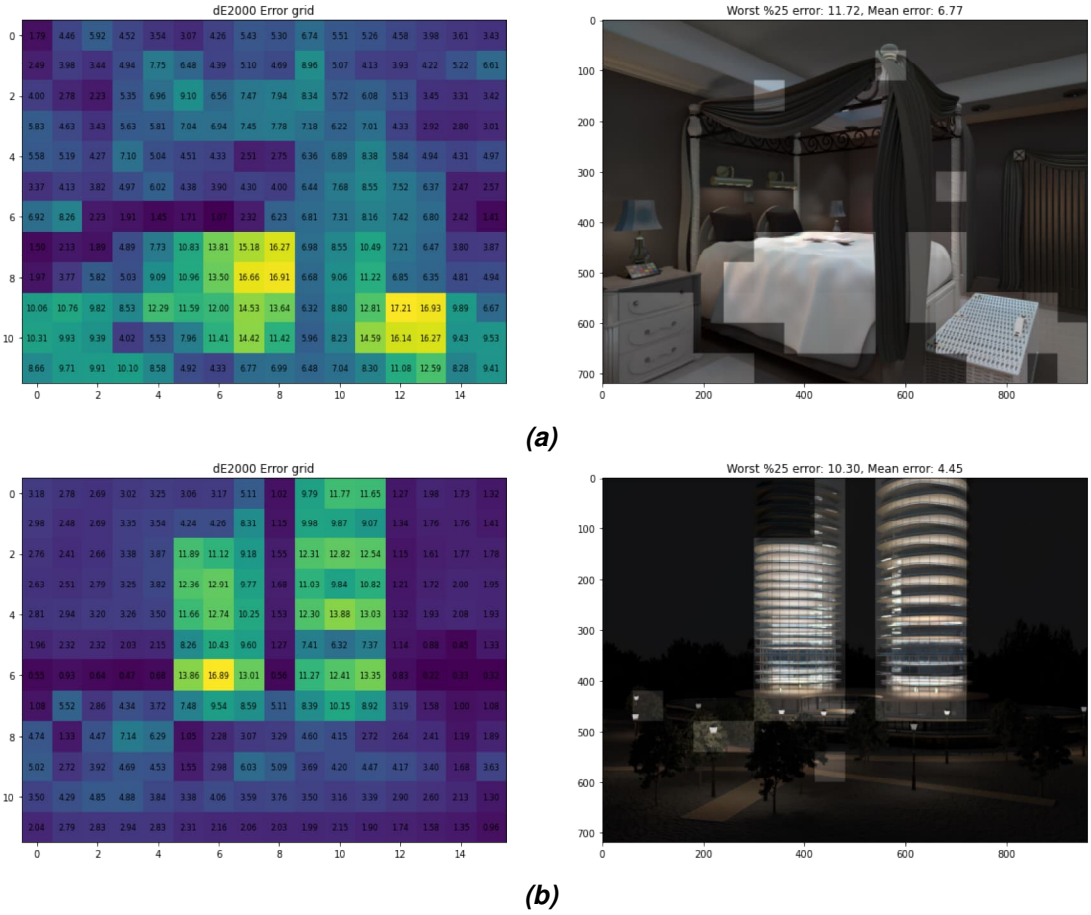


Figure B.2. *DE2000 grid error visualization for the prediction of LSMI U-net in Figure 5.3. (a) is the bedroom scene with DE2000 error for each grid shown on the left side, and worst 25% error grids are visualized brightly on the image on the right side. (b) is the same metric visualization applied to the skyscraper scene. The ground truth used in the comparison is shown in Figure 5.3.*