BRDF Blending in Path Tracing with Multiple Importance Sampling

Ian Parzival Legmaier

MASTERARBEIT

eingereicht am
Fachhochschul-Masterstudiengang

Interactive Media

in Hagenberg

im Juni 2018

Declaration

I hereby declare and confirm that this thesis is entirely the result of my own original work. Where other sources of information have been used, they have been indicated as such and properly acknowledged. I further declare that this or similar work has not been submitted for credit elsewhere.

Hagenberg, June 25, 2018

Ian Parzival Legmaier

Contents

Abstract

Multiple importance sampling (MIS) provides a robust solution to combine several sampling techniques. Especially in the area of *Monte Carlo* based rendering, *multiple importance sampling* is an established strategy and part of many professional rendering solutions. This thesis aims to utilize *multiple importance sampling* in a different context by blending an arbitrary amount of *bidirectional reflectance distribution function's* (BRDF) into a single complex surface representation. To evaluate the feasibility of this approach, the *multiple importance sampling* BRDF blending concept is integrated into an own implemented *Monte Carlo path tracing* solution. The results demonstrate that BRDF blending via *multiple importance sampling* is possible and achieves results comparable to existing rendering solutions. While leaving a low computational footprint, the novel combination approach described in this thesis does not introduce additional noise in the final image.

Kurzfassung

Multiple importance sampling (MIS) bietet eine robuste Lösung um eine beliebige Anzahl von Sampling-Strategien zu kombinieren. Insbesondere im Bereich der *Monte Carlo* basierten Bildsynthese hat sich *multiple importance sampling*, als fester Bestandteil von vielen professionellen Rendersystemen, etabliert. Das Ziel dieser Arbeit ist es, *multiple importance sampling* in einem alterierten Kontext zu verwenden, welches die Kombination einer beliebigen Anzahl von *bidirektionalen Reflektanzverteilungsfunktionen* (BRDF) ermöglicht. Um die Machbarkeit dieser Vorgehensweise zu evaluieren, wurde das MIS Kombinationsprinzip für BRDFs in einem eigens implementierten *Monte Carlo path tracer* integriert. Die Resultate demonstrieren die erfolgreiche Vermengung von BRDFs mittels *multiple importance sampling*, auch im Vergleich zu exisiterenden Render-Systemen, werden überzeugende Resultate produziert. Desweiteren führt diese neuartige Kombinationsstrategie zu einem minimal höheren Rechenaufwand. Das finale Endergebnis leidet außerdem nicht unter zusätzlichen Bildartefakten in Form von Rauschen.

Chapter 1

Introduction

The combination of an arbitrary amount of already existing surface representations is a very powerful approach to create a broad range of divergent materials. This thesis concentrates on a new approach to combine different *bidirectional reflectance distribution functions* (BRDF) via the *multiple importance sampling* technique. An introduction to the general problem statement is given and the contribution is described in this chapter. Furthermore, an overview of the thesis structure is provided.

1.1 Motivation and problem statement

Combining multiple simple surface representations in order to receive a single complex material is very effective to cover a wide range of different materials. *Multiple importance sampling* (MIS) provides a possible new approach to combine multiple surface representations in the context of *Monte Carlo path tracing*.

The research in this thesis focuses on combining BRDFs (bidirectional reflectance distribution functions) with *multiple importance sampling* (MIS). A BRDF provides a model to describe surface properties of a rendered object, it summarizes the necessary parameters to describe a material [7]. Standard BRDFs like diffuse, glossy or mirror reflection types are very common and can be found in various *Path Tracing* solutions due to their relatively simple implementation. However, often more complex reflection models are required to simulate materials which occur in the real world.

The common use case of *multiple importance sampling* is to merge various importance sampling strategies, the intended usage of MIS is to combine *direct light sampling* and BRDF sampling [21]. Therefore this thesis aims to apply *multiple importance sampling* in a different context in order to blend an arbitrary amount of BRDFs to a single complex model. In this thesis the general possibility of the blending approach via MIS is evaluated and also implemented into an own *Monte Carlo path tracing* solution. *Multiple importance sampling* is typically not used to blend different BRDF models, therefore this approach can lead to a new strategy to combine multiple surface representations to obtain a single more sophisticated material.

1.2 Contribution

The main contribution of this thesis is a *bidirectional reflectance distribution function* blending approach which utilizes *multiple importance sampling* in combination with heuristics to calculate the blending weight for each BRDF. This BRDF combination strategy provides the capability to merge an arbitrary amount of surface representations. Furthermore the whole combination system offers an easy integration into existing *Monte Carlo Path tracing* solutions.

1.3 Goals

The most important goal for this thesis is to successfully adapt *multiple importance sampling* in the context of BRDF combination. This refers not only to the mathematical point of view, which addresses the correct adaption of the *multi-sample estimator* of MIS, also reasonable visual results of the combined surface models are essential to fulfill this main goal.

Based on the main goal of an operative BRDF blending system, an additional objective is to support the combination of an arbitrary amount of *bidirectional reflectance distribution functions* in order to eliminate possible constraints during the material blending process. Furthermore, this combination approach should not induce additional artifacts in form of noise into the final render outcome. A further objective is to provide an easy integration of the *multiple importance sampling* blending approach for BRDFs into *Monte Carlo* based rendering systems.

1.4 Thesis structure

As a starting point this thesis introduces *multiple importance sampling* and common usages of this technique in Chapter 2. Furthermore, an overview of existing BRDF combination techniques are provided. Subsequently, an overview of *path tracing* and *Monte Carlo integration* is given in Chapter 3 in order to provide the essential knowledge for the implementation in Chapter 4. The general implementation in Chapter 4 covers the thesis path tracer, the corresponding systems, as well as the BRDF representation and implementation. Chapter 5 explains in detail how MIS was adapted for BRDF blending and gives insight into the whole combination system. The results of the BRDF blending system via *multiple importance sampling* are shown in Chapter 6 and furthermore discussed in Chapter 7.

Chapter 2

State of the Art

Combining multiple simple surface representations into a single complex one is an effective approach to cover a wide range of different materials. Not only physically plausible surface representations can be crafted with this strategy, also non existing exotic materials for an artistic usage can be created.

Multiple importance sampling (MIS) is used in this thesis to combine arbitrary bidirectional reflectance distribution functions (BRDF). This chapter introduces how *multiple importance sampling* is used in the intended way and also gives an insight in other MIS applications. Also other approaches for material and BRDF blending are given in this chapter to clarify the distinction to the approach of this thesis which blends BRDFs with *multiple importance sampling*.

2.1 Multiple importance sampling

Multiple importance sampling (MIS) was introduced by *Erich Veach* and *Leonidas J. Guibas* a new *Monte Carlo* rendering technique. This section introduces MIS based on the results of *Veach* and *Guibas* in [22] which are described more elaborately in [21], the following subsections summarize their work except explicitly stated otherwise. Furthermore, also the intended application of the *multiple importance sampling* approach is illustrated.

2.1.1 Overview

In the area of photo realistic rendering, *Monte Carlo integration* is mostly used in order to solve the complex lighting integrals by a random sampling approach. By relying on estimations to solve integrals the quality of the estimate is depending on the chosen sample count. A major drawback of *Monte Carlo integration* is therefore the dependency on the quantity of calculated samples. A high sampling count leads to a higher quality image, whereas a low sampling count leads to a noisy image due to the tendentially high variance of the sampled results.

2.1.2 Reason to use several sampling techniques

As described in Section 2.1.1, an integral which is sampled with *Monte Carlo integration* can have a high variance result. This is due to often non optimal properties of rendering integrals which are noticeable as singularities or very large values over small parts of the given domain. In order to solve this problem, it is needed to use multiple sampling techniques for different domains of the given rendering integral. Usually each sampling technique is especially designed to provide low variance results for a specific domain. A simple ray tracer may for example divide the rendering equation to the domains *direct lighting*, *glossy reflections* and *ideal specular* and implement sampling strategies for each of the given domain, which will yield to low variance results for each domain.

2.1.3 Focus of multiple importance sampling

Multiple importance sampling uses the in Section 2.1.2 described idea of multiple sampling techniques for specific domains and extends it to a technique which is able to make *Monte Carlo integration* more robust by using more than one sampling method to evaluate the same integral. Beside the combination itself, *Veach* also introduced several heuristics in [21] on how to combine several sampling strategies for a single integral.

Multi-sample estimator

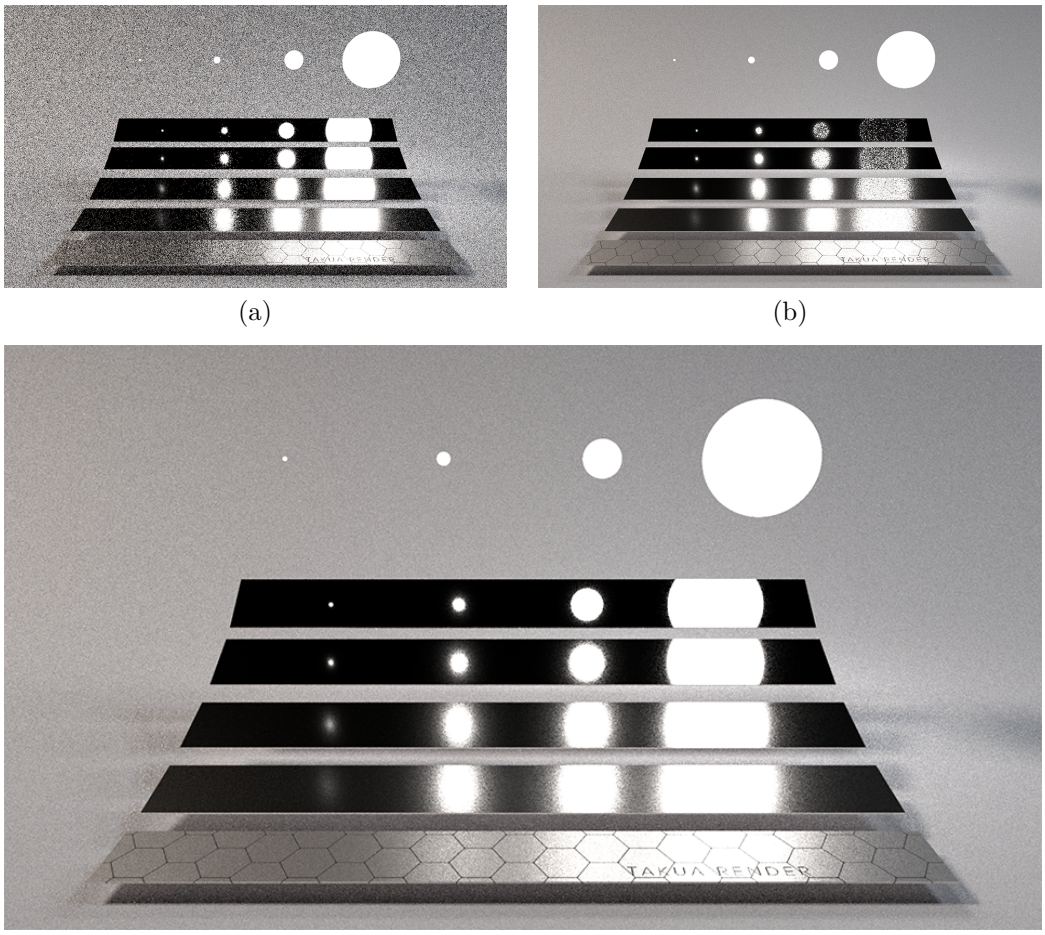
The *multi-sample estimator* provides a solution to sample different domains of a given integral. The probability density functions (pdf) $p_0, \ldots, p_{n-1}$ yields the information on which part of a given distribution to sample. The *multi-sample estimator* is given by

$$F = \sum_{i=0}^{m} \left[\frac{1}{N_i} \sum_{j=0}^{N_i} w_i(X_{i,j}) \frac{f(X_{i,j})}{p_i(X_{i,j})} \right], \tag{2.1}$$

where $f(X_{i,j})$ defines the function to sample, which is in the field of rendering the rendering equation with a corresponding bidirectional reflectance distribution function (BRDF). The probability density function is given by $p_i(X_{i,j})$. Also each *multi-sample estimator* has a set of weighting functions $w_0, \ldots, w_{n-1}$. A random sample is defined by $X_{i,j}$. Furthermore, m describes the amount of sampling techniques while N_i provides the information of the sampling count for the sampling technique with the index i. The result F refers to the approximated result of a given original function $f(x)$.

Principle of weighting heuristics

The *multi-sample estimator* described in Section 2.1.3 is mainly driven by the weighting heuristic w_i in order to combine multiple sampling techniques in the right proportion. To choose the optimal sampling technique, the weighting function relies on the given probability density function p_i for the integrand f. For each pdf $p_0, \ldots, p_{n-1}$ where the probability is high, it is attempted to evaluate f with the most optimal sampling strategy. A commonly used weighting heuristic, the *balance heuristic*, calculates the weightings based on the dominance of the given probability density function p_i.



(a) (b)

(c)

Figure 2.1: This Figure shows a *multiple importance sampling* test scene which is rendered with the *Takua Renderer* by *Yining Karl Li* [41] with 64 samples per pixel. *BRDF sampling* (a), *Light sampling* (b), *Combined result* (c). Each rendering contains the same 5 planes with different glossiness values. Also 4 light sources with the same emission value are placed with different radii.

2.1.4 Application of MIS

In this section the intended application of *multiple importance sampling* is introduced. In [21] *Veach* presents the problem of rendering glossy surfaces, the two common strategies to sample such a surface and it is also shown that each sampling method performs well in certain situations, but behaves non optimal in others. These sampling strategies are called *BRDF sampling* and *direct light sampling*. It is shown that *multiple importance sampling* can be used to combine these strategies to obtain a single sampling strategy which performs well in any situation.

BRDF sampling

A more detailed explanation of the BRDF sampling concept can be found in Chapter 4. This section gives only an overview of BRDF sampling in order to grasp the context within the *multiple importance sampling* approach. To sample the BRDF of a surface a random incident ray ω_i is generated in a hemisphere which is aligned to the normal N of the surface hit point p of the view ray ω_o. This approach is usually performed recursively until a light source is hit or a maximum depth is reached. Due to this random approach, the probability to hit a light source is depending on the size of a particular light [18]. Therefore BRDF sampling performs bad with small lights and handles bigger light sources better. In Figure 2.1 (a) an example of a BRDF sampled image is viewed.

Direct light sampling

In order to sample an area light source, a random point p_L is chosen on the surface of the light, this is done by calculating the incident ray as $\omega_i = \|p_L - p\|$ [16]. As shown in Figure 2.1 (b), direct light sampling performs very well in sampling small light sources. Bigger light sources are the weak point of this sampling strategy and result in a high variance render output.

Combining BRDF and direct light sampling

As seen in Figure 2.1 (a) and Figure 2.1 (b) BRDF sampling performs not well on glossy surfaces with small light sources, while direct light sampling performs in a non optimal way when big light sources are used. Figure 2.1 (c) shows the merged *multiple importance sampling* result which combines both strategies. It can be seen that the positive properties of both sampling strategies are combined in order to achieve a final low variance result.

2.2 Multiple importance sampling for participating media

Multiple importance sampling is designed to generally combine multiple sampling techniques. Therefore, it is not unexpected to find usages of this method in another context. This section provides an overview of the application of MIS for rendering participating media. The content summarized in this section is based on previous research by *Christopher Kulla* and *Marcos Fajardo* in [11].

2.2.1 Overview

Alternative importance sampling techniques for rendering homogeneous and heterogeneous participating media for arbitrary light sources were introduced by *Kulla* and *Fajardo*. Several sampling methods were developed for special situations which were combinable by *multiple importance sampling*.

2.2.2 Sampling homogeneous media

Kulla and *Fajardo* introduced *equi angular sampling*, beside the older sampling strategy *distance sampling*, for homogeneous participating media. However, *equi angular sam-*

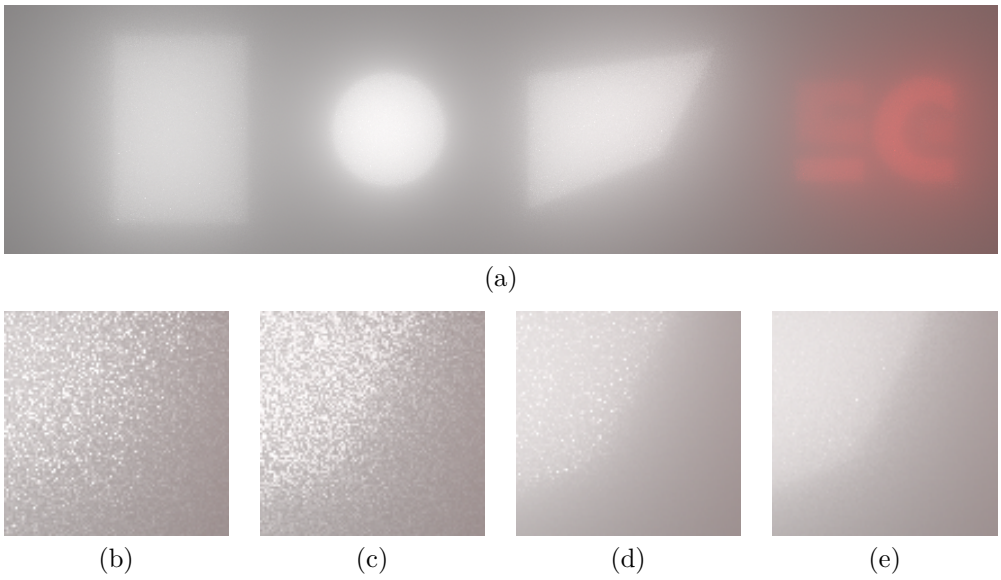


Figure 2.2: This figure shows a high quality target image of homogeneous participating media from [11] in combination with various area lights in (a). Furthermore, example renderings with several sampling techniques are shown. *Distance sampling* without MIS (b), *distance sampling* with MIS (c), *equi angular sampling* without MIS (d), *equi angular sampling* with MIS (e).

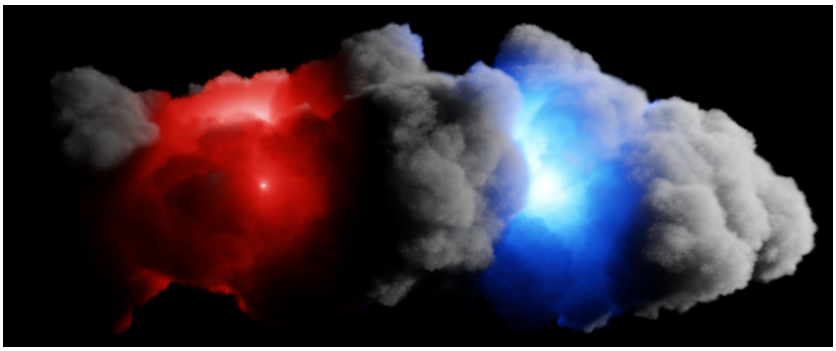
pling provides non optimal results, due to the fact that this strategy prefers mostly samples close to the viewer, which can lead to high variance artifacts.

Figure 2.2 shows the results of *distance sampling* and *equi angular sampling* with 256 paths per pixel. It is clearly evident that *equi angular sampling*, especially with MIS, provides the result with the highest quality. However, *multiple importance sampling* is, in the context of homogeneous media, not used to combine *direct sampling* and *equi angular sampling*. It is used to combine *area sampling* (see Section 2.1.4) and the sampling of the *phase function* of the participating media.

2.2.3 Sampling heterogeneous media

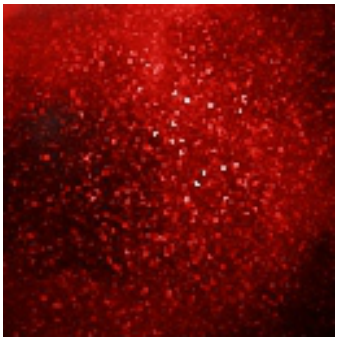
Unlike homogeneous volumes, a heterogeneous participating media features varying properties on its light path, which leads to non uniform visual effects like clouds or smoke [4]. An example of heterogeneous media is shown in Figure 2.3.

As stated in [11], a homogeneous region can also be calculated by defining bounds. This fact is used for heterogeneous media by computing N number of small homogeneous regions, this regions are generated via *ray marching*. Due to problems like sharp changes at the surface of heterogeneous participating media or empty spaces between volumes, different sampling techniques are required in order to focus the sampling on the visually important regions. Therefore, it is suitable to use different sampling techniques for various situations, which are *equi angular sampling* (see Figure 2.2 (b)) and *discrete density sampling* (see Figure 2.2 (c)). Both sampling techniques are only optimal in distinct situations. As a consequence *multiple importance sampling* provides a solution



(a)

(b)



(c)

(d)

Figure 2.3: The final MIS combination result of [11] is shown in (a). Also close ups of *equi angular sampling* (b), *discrete density sampling* and the final merged outcome (c) are displayed. Every image is computed with 16 samples per pixel.

to combine the strengths of both sampling methods to provide a better visual result without increasing the sampling count. The combined *multiple importance sampling* result can be seen in Figure 2.2 (d).

2.3 Material mixing

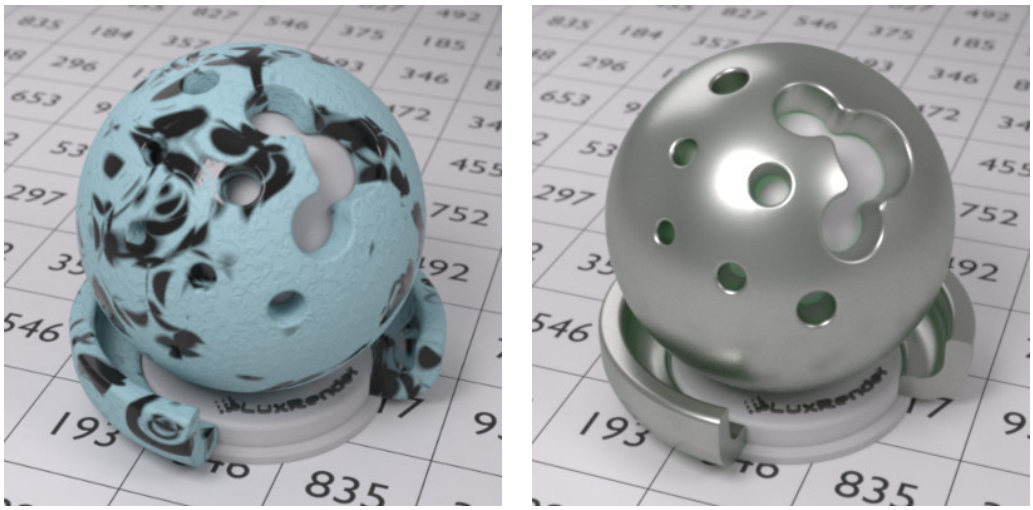
This section gives an overview of the *mix material* which is described in the book *Physically Based Rendering* (PBRT) by *Matt Pharr*, *Wenzel Jakob* and *Greg Humphreys*.

2.3.1 The mix material

This section summarizes the properties and approach of the *mix material*, which is introduced in [14], to define the own thesis approach better in contrast to the *mix material* of PBRT.

Functional principle

The *mix material* m_c is able to combine two other materials m_1 and m_2 based on a scalar blend weight w. The blend weight w simply defines the intensity of the materials



(a) (b)

Figure 2.4: Both images are rendered with *LuxRender* and a *mix material* [37]. The combination of a glossy black BRDF and a diffuse material via a weight map is shown in (a). Furthermore, (b) shows the mixture of a silver BRDF and green rough glass.

m_1 and m_2 during the shading process. The weight is given as a texture, therefore it is necessary that the rendered geometry must have UV coordinates to be able to apply meaningful weights. This texture can be handcrafted or procedurally generated to achieve interesting visual blending results. The results of both materials m_1 and m_2 are calculated and blended based on the weighting value $w_{u,v}$ of the shaded point p.

Lux Render, which is based on PBRT, implements this type of material combination [36]. Figure 2.4 shows an example of the *mix material* approach. The first material m_1 has a black and glossy appearance, whereas the second material m_2 has a matt visual aspect. The weight texture is generated procedurally which leads to the blending result seen in Figure 2.4 (a).

2.4 BRDF layering

A central element of rendering in the field of computer graphics is the concept of light scattering on surfaces, which is often formulated as *bidirectional reflectance distribution function* (BRDF). Most of the time simple diffuse or micro-surface based material representations are sufficient to cover a high amount of surfaces. However, surfaces exist which are not replicable with standard BRDFs for example: coating over car paint, glazed ceramic or skin. In order to represent this kind of complex materials, an approach can be used which layers or stacks multiple single BRDFs on top of each other, also taking effects like total inner reflection and absorption between thin layers into account. This special type of layered BRDFs are described in [8] by *Jakob* and in [25] by *Weidlich*. Based on the results of *Weidlich* this section provides an overview of layering methods, except stated otherwise.

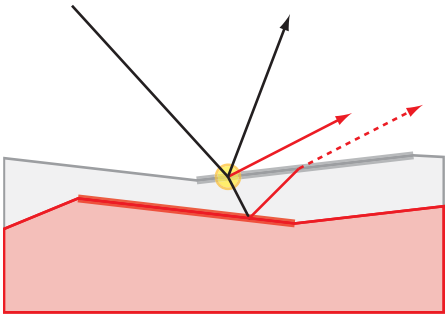


Figure 2.5: This Figure displays the simplified layering model used in [25] by *Weidlich*. The two surface layers are represented by the grey and red area. An example ray interaction is also visualized by red and black arrows. A more detailed description can be viewed in Section 2.4.1.

2.4.1 Layering concepts

The basic concept of BRDF layering can be defined by using single surface representations, like perfect diffuse or specular material types, and layer them on top of each other to obtain a more sophisticated BRDF. In order to apply this concept with reasonable performance, several simplifications are often assumed like omitting the light scattering between layers to save costly sub surface computations or the limitation of the layer thickness. It is also possible to precompute BRDFs and phase functions (for light scattering between layers) as a *Fourier basis* and store the results for later rendering [8]. This enables the simulation of complex layered BRDFs in a reasonable time without sacrificing physical correctness by relying on simplifications. In [25] *Weidlich* describes a layering simplification which includes the following two major steps:

1. In order to save intersection tests, rays are always assumed to leave through the entering micro-facet.
2. Exiting rays from the lower layer levels will always cast out from the original entry point of the higher level [27]. This is visualized in Figure 2.5 by the yellow circle, the dashed red line displays the physically correct exit point.

2.4.2 Problems of layering

As described in Chapter 3, *Monte Carlo* based rendering systems rely on BRDFs with suitable probability density functions (pdf), in order to achieve better convergence rates during the rendering process. Due to this fact it is often not practical to determine a single pdf for a complex layered material [10]. A solution is to sample the BRDF of each layer independently at first and choose the most dominant pdf during the ray intersection process. Also the layering method proposed by *Weidlich* does not consider any kind of inner reflection between material layers, which leads to wrong albedo values, even for simple layering scenarios, additionally the stated technique is not able to properly evaluate the results of multiple rough borders, which needs full consideration of total inner reflection rays.

(a) (b) (c)

Figure 2.6: These images show results of layered materials from [25]. A metal material combined with a glossy layer on top is illustrated in (a) as an example for car paint. In (b) a material with red spray paint as the bottom layer and glassy glaze on top is shown. Furthermore, (c) displays a multi layered material which represents wall paint.

2.4.3 Application of layered materials

Due to the simplicity of a layering based system, it easy to cover a wide variety of different real surface appearances. With already two basic BRDF types, diffuse and glossy, it is possible to create a lot of different materials. Some of them can be viewed in Figure 2.6. In order to particularize potential applications of layer based materials, an overview of material examples are given based on the results of [25] by *Weidlich*.

Glossy paint: This combined BRDF is based on a diffuse layer on the bottom and a glossy coating on top. This type of material can be used for a ceramic glazing or any other type of glossy coating based appearance.

Frosted paint: This material combination consists of a rough BRDF on top and another rough, but painted surface at the bottom. An example can be seen in Figure 2.6 (b) which shows a gloss reducing spray coating on top of red paint.

Foil: In order to simulate metallic based foils, two glossy BRDFs can be layered on top of each other. The bottom layer can be colored to achieve the effect of a particular metal type like bronze or gold.

Car paint: Since the real appearance of a car body is defined by multiple layers of different materials, a multi layer BRDF system is, regarding the concept, suitable to simulate those kind of complex surface representations. Car paint can be simulated by using a more rough surface type at the bottom layer and a glossy material type at the top. An example of this combined BRDF can seen in Figure 2.6 (a).

Multi layer: Not all material types can be simulated by only using two BRDF layers. Figure 2.6 (c) shows an example of a complex wall paint material.

2.5 Time-Varying BRDF

This section concentrates on the results of [19] by *Sun* who presents a BRDF concept which is extended by a parameter τ in order to change the visual effect over time, this time based effects contain a blending between several BRDFs. In particular those blending results are of importance for this thesis. First, this section provides an overview of the time based BRDF approach, additionally more insights are given about particular effects like the representation of drying surfaces or dust accumulation which are based on the results of *Sun*, except stated otherwise.

2.5.1 Overview

Real world surfaces typically change the visual appearance over time, either due to general aging processes, drying effects or dust accumulation. Since BRDFs usually portray the state of a material at a single point of time, *Sun* introduced a concept to model surfaces with an additional time parameter τ in order to simulate surface changing effects over a given time span. Furthermore, real world BRDF data of drying paint, the accumulation of dust on surfaces and the drying process of general wet surfaces are obtained with a custom made BRDF acquisition system, based on this time based BRDF data base, the properties are fitted to parameters of existing BRDF models to be able to compute the time based surface information with a regular rendering system.

2.5.2 Drying paint

In general *Sun* observed that wet paint results in a high specularity on the specific surface, however due to the drying effect the material becomes more diffuse. In order to model the time based BRDF of drying paint the *Oren-Nayar* [13] reflection model is used for diffuse surfaces and the *Torrance-Sparrow* [20] model for glossy and specular materials. The diffuse model ρ_d and the specular model ρ_s are treated strictly separate and are therefore combined with a simple addition, similar to the *Phong reflection model* used for real time applications [15].

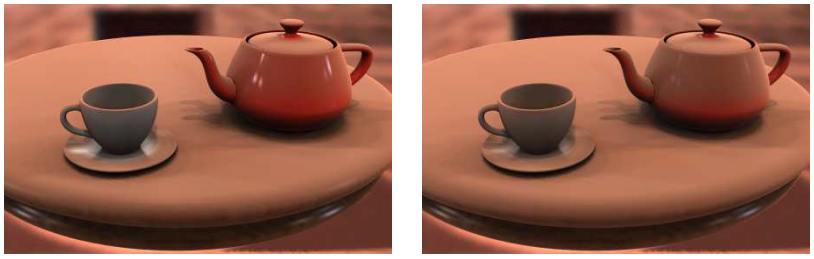
2.5.3 Drying of wet surfaces

Wet materials lose color contrast and general color saturation over time, also specular highlights are disappearing quickly as the drying process proceeds. The diffuse color of the dried surface shows linear color variations, and therefore *Sun* combines the color properties of the dry and wet BRDF via linear interpolation. As base BRDFs, the *Oren-Nayar* and the *Torrance-Sparrow* reflection models are used for the diffuse and glossy material type.

2.5.4 Dust accumulation

The blending between reflection types, in order to achieve the visual effect of dust accumulation on surfaces, is described in this section based on the results of [19] by *Sun*. For the surface representation of the dust layer, the reflection function of *Blinn* for simulation dusty surfaces [2] is used. The BRDF of the base layer can either be the *Oren-Nayar* or the *Torrance-Sparrow* model. The dust and base layer are combined via simple

(a) $\tau = 0.0$ (b) $\tau = 0.19$



(c) $\tau = 0.72$ (d) $\tau = 1.28$

Figure 2.7: Example results of dust accumulation from [19], where τ denotes the point in time for the *Time-Varying BRDF* are displayed in (a–d). The dust source is placed directly above the scene.

linear interpolation based on the BRDF data base values of *Sun* and environmental data during the render process. The dust accumulation itself is based on various properties like the position of the dust source or the contact with other objects in the environment. The general amount of dust on a surface is regulated by $d = N_p \cdot U$, which is the dot product of the normal vector N of a given surface point p and the up vector U of the environment, also general dust occlusion is calculated on every surface point p in order to shade areas correctly which are inside the *dust shadow*. An example of the dust accumulation effect over a given time frame can be seen in Figure 2.7.

Chapter 3

Monte Carlo Path Tracing

Monte Carlo based *Path Tracing* provides a state of the art solution for solving the rendering equation with statistical methods. This chapter gives an overview of *Path Tracing*, the basics of *Monte Carlo integration* and how these techniques are connected. Also variance reduction methods which were utilized in the thesis project are introduced.

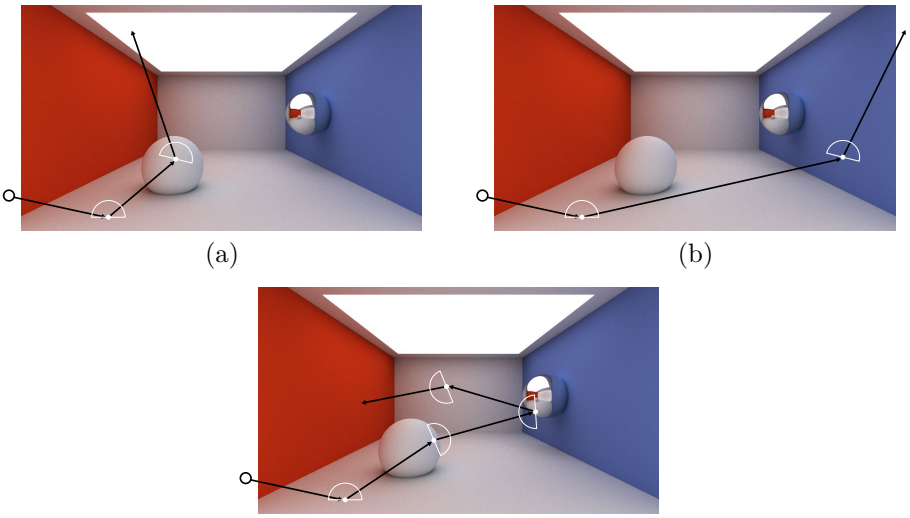
3.1 Path Tracing

Path Tracing is a ray tracing based derivative which was introduced in [9] by *Kajiya* as an algorithm to solve the full rendering equation (see Section 3.2) via *Monte Carlo integration*. *Path Tracing* is categorized as a ray tracing based technique, however classic *Whitted* [26] style ray tracing only captures local surface effects like reflections or shadows while *Path Tracing* is additionally able to model global light interaction effects such as color bleeding (global illumination) and caustics.

3.1.1 Basic principle

This section provides an introduction to *Path Tracing* which is based on [14]. Naive *Path Tracing* can be described as a light transport algorithm which computes direct and indirect lighting of an environment with a brute-force approach. Like in classical ray tracing, a ray is generated outgoing from the camera. If a surface is hit, a ray tracer calculates the radiance based on local object properties, exceptions are reflective and refractive surfaces. Nevertheless, in *Path Tracing* new rays are generated for each surface hit until the path reaches a light source, the sky or a defined maximum depth limit of the light path as shown in Figure 3.1. For each surface hit point p_i a new incident ray ω_i is created by sampling a random direction of the hemisphere which is aligned to the surface normal N_i of the point p_i. The way a new ray is sampled, or which directions are preferred in the context of probability, is defined by the corresponding BRDF of the surface point p_i. The algorithm can be broken down into the following rough steps:

1. Cast ray from the camera into the scene.
2. Check for the criteria shown in Figure 3.1.
3. Sample a new ray ω_i based on the hemisphere of hit point p_i.
4. Repeat from step 2.



(a) (b) (c)

Figure 3.1: These examples show different criteria for a light path to end. For each image the starting ray, or primary ray, stays the same and simulates possible light paths for the calculation of a single pixel computed with 3 samples. The white hemispheres are visualizing the sampling space of the corresponding generated incident ray ω_i. Each rendered image is produced by the path tracer of the thesis project. *Light hit* (a), *sky hit* (b), *max depth reached* (c).

3.2 Rendering equation

The rendering equation, introduced in [9] by *Kajiya*, describes the scattering of light in the environment. This comprehensive equation provides the total amount of photon transport, which can be described as the reflected light at the point p. The following equation provides a more formal description of the rendering equation which is defined as

$$L(p, \omega_o) = L_e(p, \omega_o) + \int_\Omega L_i(p, \omega_i) \cdot f_r(p, \omega_o, \omega_i) \cdot \cos\theta \, d\omega_i. \tag{3.1}$$

The result $L(P, \omega_o)$ of the rendering equation is, as already mentioned, the amount of light reflected at the point p towards the outgoing direction ω_o, which is the light approaching the viewer. The function $L_e(p, \omega_o)$ outside of the integral defines the emitted light at the given point p to the direction ω_o, which is especially important for light emitting surfaces (see Section 4.2.4). The integral is needed in order to obtain the light coming from all directions above the hemisphere which is aligned to the surface normal N at the point p. Ω denotes that the integration domain is the hemisphere oriented to N. $L_i(p, \omega_i)$ describes the incoming light from the direction ω_i to the point p. This is the recursive part of the rendering equation which is also responsible for effects like global illumination. This recursion is implemented by computing the rendering equation with the given values p and ω_i. The BRDF (*bidirectional reflectance distribution function*) is represented by $f_r(p, \omega_o, \omega_i)$ where p is again the surface point. Furthermore, ω_o and ω_i are the outgoing and incoming directions. The BRDF part varies depending on the used reflection model and is often the target to optimize via *importance sampling* (see

Section 3.3.2). The result of the BRDF is attenuated by $\cos\theta$ which can be calculated by utilizing the dot product between the incoming light ω_i and the surface normal N.

3.3 Monte Carlo integration

Complex integrals, which can be found for example in the rendering equation (see Equation 3.1), are often not analytically solvable, therefore numerical methods provide a solution to compute such integrals in a reasonable time. *Monte Carlo integration*, described in [28] by *Wojciech*, is such a numerical integration method which is able to deliver results with a good convergence rate without depending on the dimensionality of the given integral, it is a random based approach to evaluate an integral, this is done by taking random samples of the given integral and average the results, which provides a solution which is statistically very close to the real outcome. The core aspects, which are the *Monte Carlo estimator* and *importance sampling*, are summarized in this section as described by *Wojciech*. Additional resources are stated explicitly.

3.3.1 Monte Carlo estimator

A more formal description of *Monte Carlo integration* is defined by the *Monte Carlo estimator*. Given an arbitrary integral $\int f(x)\,dx$ the expected value is calculated by

$$E[F_N] = \frac{1}{N}\sum_{i=1}^{N}\frac{f(X_i)}{pdf(X_i)}, \tag{3.2}$$

where $E[F_N]$ defines the expected value of the integral for a given function $f(x)$ and N represents the amount of samples taken in order to calculate the expected result. The random variable X_i is used as an input for the given function $f(x)$ and the probability density function (pdf) (see Section 3.3.2). By utilizing the pdf the *Monte Carlo estimator* given in Equation 3.2 is also called *Monte Carlo estimator* with *importance sampling* (see Section 3.3.2).

By applying the *Monte Carlo estimator* to the integral part of the rendering equation mentioned in Equation 3.1, the integration result with the *Monte Carlo estimator* and *importance sampling* can be achieved as

$$E[\int_{\Omega} L_i(p,\omega_i)\cdot f_r(p,\omega_o,\omega_i)\cdot\cos\theta\,d\omega_i] = \frac{1}{N}\sum_{i=1}^{N}\frac{L_i(p,\omega_i)\cdot f_r(p,\omega_o,\omega_i)\cdot\cos\theta}{pdf(\omega_i)}. \tag{3.3}$$

3.3.2 Importance sampling

Importance sampling is an approach to reduce the variance of *Monte Carlo integration* in relation to the *Monte Carlo* method without any variance reduction techniques and the same amount of samples N [28]. Equation 3.2 shows already the *Monte Carlo estimator* with *importance sampling* which is achieved by the divison of the probability density function (see Section 3.3.2).

The concept of *importance sampling*, in the context of *Monte Carlo* based rendering, is to concentrate the ray generation on areas which provide the most energy, the

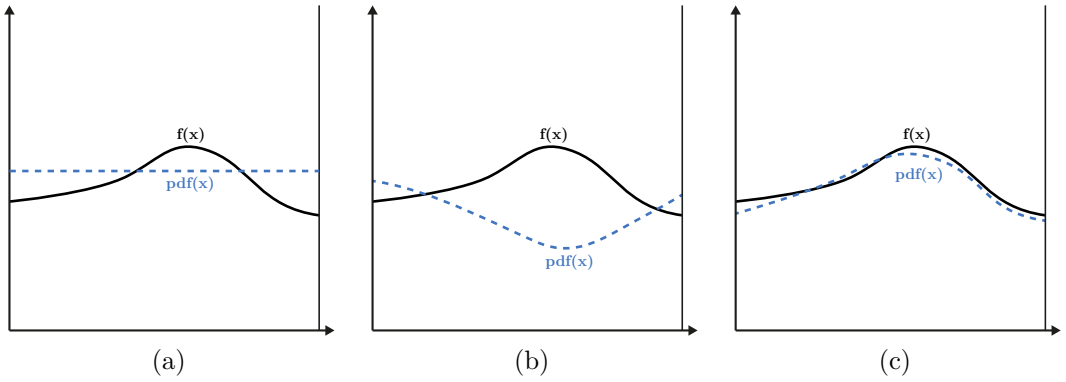


Figure 3.2: This figure describes the different relational cases between the probability density function and a function $f(x)$. Every image displays the same function $f(x)$, only the pdf changes in order to visualize the shape resemblance described in Section 3.3.2. *Uniform* pdf (a), *mismatched* pdf (b), *similar* pdf (c).

information about this essential areas is provided by the probability density function [17]. In the application of *Path Tracing* the rendering equation results in higher values if a light source was hit by a ray.

Probability density function (pdf)

The probability density function (pdf) defines the probability of a random variable $X_i \in [a, b]$ obtaining a specific value within its domain $[a, b]$ [6]. A pdf is always non negative which can be defined as

$$\forall x, pdf(x) \geqslant 0. \tag{3.4}$$

Furthermore the integration of a probability density function always results in 1 for its domain $[a, b]$ which can be stated as

$$\int_a^b pdf(x)\, dx = 1. \tag{3.5}$$

As explained in Section 3.3.2 the pdf is used for *importance sampling* to increase the efficiency of the *Monte Carlo estimator*. The performance of the variance reduction is significantly dependent on the shape of the probability density function $pdf(x)$ in relation to the original function $f(x)$ [28]. Ideally the shape of the pdf resembles a shape which is similar to the original function in order to achieve considerable variance reduction (see Figure 3.2 (c)), as a consequence a mismatched pdf (see Figure 3.2 (b)), which does not approximate the shape of the function $f(x)$ in any way, increases the overall variance [1]. The different relation cases between the probability density function $pdf(x)$ and the original function $f(x)$ are shown in Figure 3.2.

3.4 Stratification

As mentioned in Section 3.3, *Monte Carlo integration* is based on the idea to randomly sample the given integral in order to approximate the final result statistically with a

number of samples N. Random sampling has the drawback that an even distribution of those samples is not guaranteed [34]. With typical random generators the resulting samples tend to clump in the given domain, this problem was directly encountered during the path tracer implementation. As a consequence of clumping, similar parts of the rendering equation are evaluated, which leads ultimately to higher variance. A more detailed explanation and examples are provided in Section 4.1.3.

The goal of stratification is to distribute random samples in a more optimal way to reduce clumping, which leads eventually to less variance [23]. Therefore, this type of technique is categorized as a variance reduction method. Stratification can be roughly divided in the areas of *stratified sampling* (see Section 3.4.1) which is also utilized in the thesis implementation and *Quasi Monte Carlo* techniques which are described in Section 3.4.2. Figure 3.3 displays a comparison between a complete random approach and the *stratified sampling* method on the basis of a 2D image plane.

3.4.1 Stratified sampling

Stratified sampling is the utilized stratification technique used in the thesis project. A more detailed implementation approach of *stratified sampling* is given in Chapter 4, this section gives a general overview of the concept.

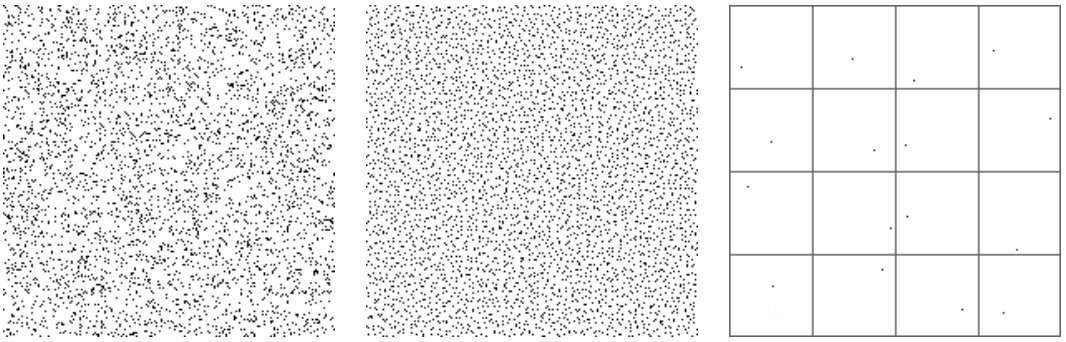
For *stratified sampling* the domain S is split up into multiple sectors S_i with the quantity of n, whereby each region must not overlap with any other. A region is also named as *stratum* in the context of *stratified sampling* [38]. This can be defined as

$$\bigcup_{i=1}^{n} S_i = S_1 \cup S_2 \cup \ldots \cup S_{i-1} = S, \tag{3.6}$$

where S defines the whole domain which consists of multiple *strata* $S_1, S_2, \ldots, S_{i-1}$ with the quantity of n. Figure 3.3 (c) shows a visualization of the *strata* on a 2D image plane. With *stratified sampling* in combination with *Monte Carlo integration*, a sample N_i is not taken from the whole domain S, the idea is that each sample N_i is now associated with a *stratum* S_i which means also the sampling is performed in the subdomain of S_i as described in Section 4.1.3. In the context of *Path Tracing* this sampling technique is often used to subdivide each cell of the virtual camera grid for the primary ray generation [12]. However, also during the process of sampling the hemisphere, which is part the BRDF sampling, *stratified sampling* is used [34]. For the thesis project stratification is exclusively used for sampling the hemisphere for BRDFs.

3.4.2 Quasi Monte Carlo

As stated in [39], by replacing the random component of the *Monte Carlo integration* with deterministic algorithms which produce uniformly distributed point sets, also called *low-discrepancy* point sets, the standard *Monte Carlo* approach is usually called *Quasi Monte Carlo*. *Low-discrepancy* sets perform better than randomly based stratification techniques in context of the resulting uniformity of the samples, however for high dimensional integrals like the rendering equation (see Equation 3.1), non trivial adjustments must be made in order to function in a multi-dimensional context. *Stratified sampling*, which is described in Section 3.4.1, integrates easier into the rendering equation. Due to



(a) (b) (c)

Figure 3.3: In (a) $N = 4096$ randomly sampled points on a 2D image plane are shown. The same amount of samples with the *stratified sampling* approach is displayed in (b). A visualization of the *strata* with $N = 16$ samples is shown in (c). It can be observed that *stratified sampling* (b) produces considerably more uniform sample output than naive random sampling (a).

this *stratified sampling* was the preferred choice for the thesis project. Well known *low-discrepancy* sets are for example the *Hammersley-Sequence* [33] and the *Sobol-Sequence* which is utilized in the *Cycles* [32] renderer of *Blender*.

3.5 Bidirectional reflectance distribution function

This section provides insights to the corresponding terminology of the *bidirectional reflectance distribution function* (BRDF) which are necessary for this thesis. Therefore this section gives insights to radiometric terms, the general definition of the BRDF concept and its usage for *Path Tracing* in particular to context of this thesis.

3.5.1 Irradiance and radiance

In order to describe the radiometric quantities, this sections aims to provide a basic understanding of the terms *irradiance*, *radiance* and *radiant flux* based on [18]. *Radiant flux* Φ is energy $Q = [J]$ (Joule) over a given period of time, the *radiant flux* Φ is measured in watts ($[J/s] = [W]$). The *irradiance* quantity E adds an area component A to the *radiant flux* Φ and therefore measures the energy per time over a given area which is defined as

$$E = \frac{\Phi}{A} \, [W/m^2]. \tag{3.7}$$

Verbally this equation can be described as the arriving or exiting *radiant flux* respective to given surface A. The *radiance* L measures *irradiance* E in terms of solid angles which is quantified as $[\frac{W}{m^2 \cdot sr}]$ where sr represents *steradiant*. So the *radiance* L is the *radiant flux* Φ per area and per solid angle.

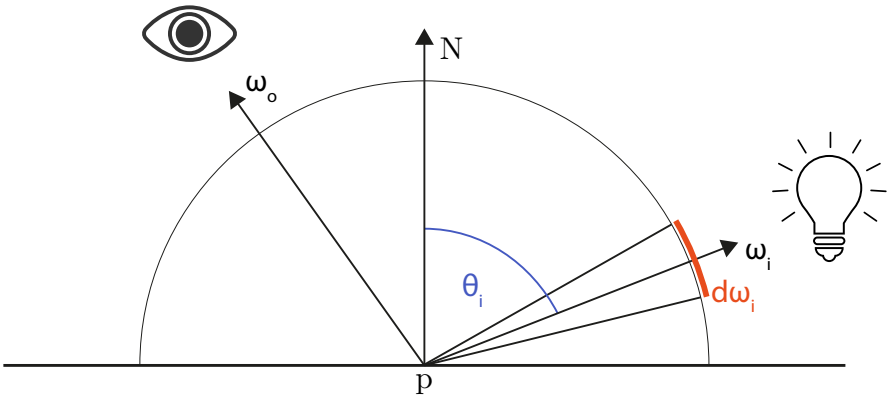


Figure 3.4: This illustration displays a simplified 2D version of the BRDF concept which is explained in Section 3.5.2. It visualizes the incoming light direction ω_i in the center of the differential solid angle $d\omega_i$ for the incident light L_i and the outgoing direction ω_o. Furthermore, the surface point p, its surface normal N and the angle θ_i between N and ω_i is shown.

3.5.2 The BRDF

This section provides an overview of the BRDF (*bidirectional reflectance distribution function*) concept based on the definition of [18]. The *bidirectional reflectance distribution function* describes how an incident direction ω_i is reflected on a surface point p which results in an outgoing direction ω_o. However, this is only a superficial explanation of the BRDF concept.

Figure 3.4 illustrates a more precise model of the BRDF approach. Given the incident direction ω_i with the incoming differential irradiance $dE(p, \omega_i)$ and the outgoing direction w_o with the differential radiance dL_0, a BRDF is defined by the ratio between them which is described as

$$f_r(p, w_o, w_i) = \frac{dL_o(p, \omega_o)}{dE(p, \omega_i)}, \tag{3.8}$$

where $f_r(p, w_o, w_i)$ is representing the *bidirectional reflectance distribution function*, $dL_o(p, \omega_o)$ is the outgoing radiance and $dE(p, \omega_i)$ the incoming irradiance. To get a better grasp on the differential solid angle $d\omega_i$ for the *irradiance*, it can be thought of the same quantity of light L_i arriving on each position of the differential solid angle $d\omega_i$ wich can be stated as $L_i \cdot d\omega_i$. This is however defined in terms of differential solid angle and not useful for the actual surface point p in Figure 3.4, due to this the incoming light L_i is projected to the surface which changes the irradiance term of the BRDF Equation 3.8 to

$$f_r(p, \omega_o, \omega_i) = \frac{dL_o(p, \omega_o)}{L_i \cdot \cos\theta_i \, \omega_i}, \tag{3.9}$$

where $L_i \cdot \cos\theta_i \, \omega_i$ is a reformulation of the differential *irradiance* $dE(p, \omega_i)$ and θ_i represents the angle between the surface normal N and the incident light direction ω_i which can be seen in Figure 3.4. In ray tracing based applications the surface point p is hit from the direction ω_o, the view direction, to calculate the reflected radiance $L_o(p, \omega_o)$

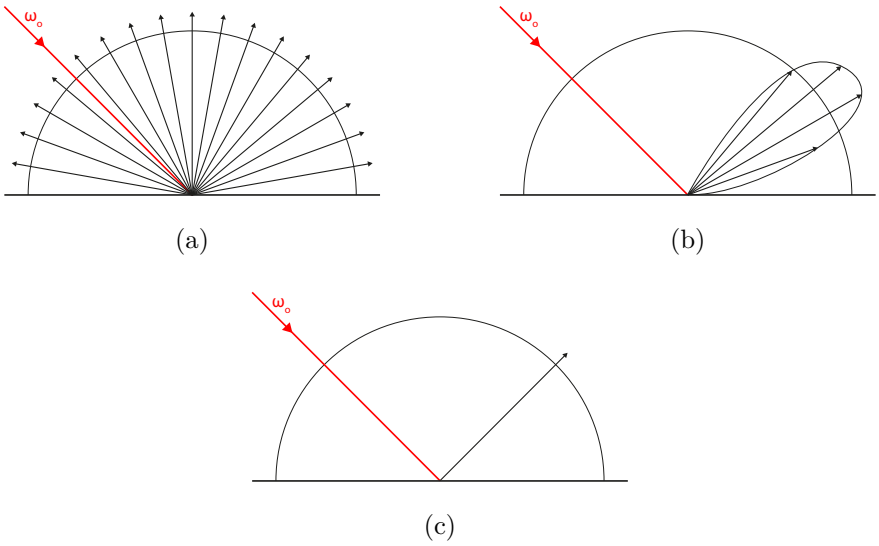


Figure 3.5: This figure visualizes three common BRDF sampling strategies: *diffuse* (a), *glossy* (b) and *mirror* (c). The red arrow resembles the direction ω_o which serves as an input value to calculate the sampled direction ω_i which are represented by the black arrows.

(see rendering Equation 3.1). This is why only a single ray for the outgoing direction ω_o is shown in Figure 3.4.

BRDF properties

This section provides an overview of the most important properties of *bidirectional reflectance distribution functions* based on the results of [7]. A BRDF is **reciprocal** regarding the incoming direction ω_i and the outgoing direction ω_o which can be formally described as

$$f_r(p, \omega_o, \omega_i) = f_r(p, \omega_i, \omega_o). \tag{3.10}$$

Furthermore, a BRDF must be **energy conserving** which means that the energy for the outgoing light is less or equal to the power of the incoming light. This is described as

$$\int_\Omega f_r(p, \omega_o, \omega_i) \cos\theta \, d\omega_i \leqslant 1, \tag{3.11}$$

where the integral describes the total energy for every direction ω_i over the hemisphere Ω of the surface point p. BRDFs where the *energy conserving* property is not fulfilled are emissive materials which emit light on their own, due to that more light energy can be reflected in comparison to the incoming light energy.

3.5.3 BRDF in Path Tracing

This section provides a short overview of the practical usage of BRDFs in *Path Tracing* which is based on the implementation of the thesis project. A detailed explanation regarding the implementations of BRDFs is given in Section 4.2.

In *Path Tracing* not only the computation of the term $f_r(p, \omega_o, \omega_i)$ is required for a BRDF component, also the generation (see Figure 3.5) of a new ray ω_i for the term $L_i(p, \omega_i)$ of the rendering equation (see Equation 3.1) is handled by the BRDF. A new ray is sampled within the hemisphere Ω (see Section 4.2.2) which is aligned to the surface normal N of the point p (see Figure 3.4). The strategy for sampling a new ray ω_i within the hemisphere Ω is derived from the corresponding probability density function, which specifies the visual look of the rendered surface point. As a consequence depending on the sampling strategy a different surface appearance is computed. Three common surface representations and the according simplified sampling strategy are visualized in Figure 3.5. As an example the *diffuse sampling* strategy shown in Figure 3.5 (a) calculates randomly new rays ω_i based in the input ray ω_o which are uniformly distributed on the whole hemisphere Ω. The *glossy sampling* strategy in Figure 3.5 (b) shows also randomly sampled direction w_i within a more directed area, which results in glossy reflection in the final rendered image. Figure 3.5 (c) simply shows the sampling of a new direction ω_i by reflecting the incoming direction ω_o based on the surface normal N which leads to mirror reflections.

Chapter 4

Implementation

Multiple importance sampling is an approach which is especially created for *Monte Carlo* based rendering techniques like *Path Tracing* (see Section 2.1). In order to achieve applicable results by combining multiple BRDFs via *multiple importance sampling*, which is the problem statement of this thesis, a fully functional *Monte Carlo* based path tracer is necessary. Therefore this chapter provides insights into the major application components of the thesis project. Furthermore, a general introduction to *Monte Carlo Path Tracing* is given and also major optimizations and tricky problem solutions are elucidated.

Since *bidirectional reflectance distribution functions* (BRDF) and especially their combination via *multiple importance sampling* are the core subject of this thesis, the representation and implementation of BRDFs for *Path Tracing* is described in detail.

4.1 Application design

This section concentrates on the major components of the thesis project, which is a fully functional *Monte Carlo* based path tracing renderer. The most relevant parts in context of the thesis statement are described which are mainly general rendering and *Path Tracing* components. Furthermore, the *Monte Carlo* concept and the BRDF interaction in the whole application structure is characterized. Figure 4.1 shows an overview regarding the major components of the thesis project.

4.1.1 Tracing engine

The *tracing engine* serves as the central computing start point for the rendering of a synthesized image I. It is designed to manage its *tracer* (see Section 4.1.2) by passing the information of which pixel to render. In its most simple form, this is implemented by iterating over a given rectangular image region R. However, depending on the implementation purpose this procedure can vary slightly.

For a pure ray tracing based application, a plain iteration over the image region R is perfectly sufficient. A *Path Tracing* based implementation of the *tracing engine* requires a different approach due to the *Monte Carlo integration* (see Section 3.3), an example of this strategy is shown in Algorithm 4.1.

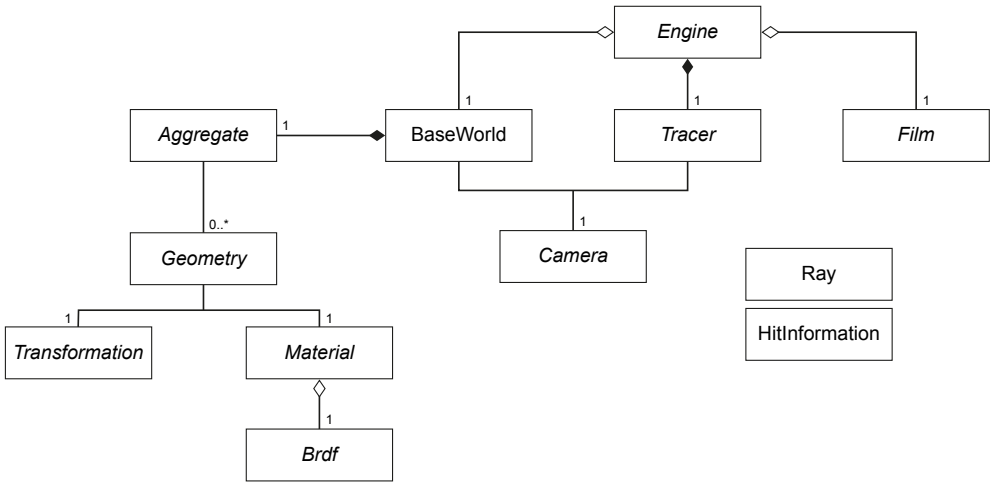


Figure 4.1: This figure shows an overview of the thesis project architecture as a simplified class diagram to visualize the relations between the major components of the system.

This design allowes also an easy integration of a multi threaded implementation of the *tracing engine* by managing a pool of *tracing engine* worker threads which are processing a queue of render regions $R_q = R_0, R_1, \ldots, R_{n-1}$. The computation of all regions R_q ultimately yields in the synthesized image I. The *tracing engine* can be seen in Figure 4.1 as the *engine* component.

4.1.2 Tracer

This section provides a detailed insight of the *tracing component* which performs the actual core computations of *path tracing*. First the needed data for the *tracer* is presented. Furthermore, the implemented algorithm for *path tracing* is explained.

Required data

In order to compute the result for a given pixel various components are required to accomplish this task. To trace a given ray r_i the scene information, encapsulated in the *world component* which is explained in Section 4.1.4, is needed. Furthermore, a virtual *camera component* is necessary to compute primary rays based on the position I_p on the image plane I and the camera properties of the given camera implementation. Additionally, the *tracer* holds an instance of the *film component* (see Section 4.1.6) which basically stores the computation results.

As an optimization, in the context of variance reduction, the *tracing component* also stores an instance of the *sampling strategy* which is used to perform *stratified sampling* (see Section 4.1.3) during the BRDF calculation. It is also ensured that only a single instance of the *sampling strategy* exists, to ensure correct sampling over multiple threads.

Algorithm 4.1: This pseudo code of the *path tracing engine* describes the process of computing a rectangular image region R with a tracer T for a given sample count S.

```
1: global variables
2:     I, Image to save tracing computation
3:     T, Tracer for image computation
4:     S, Sample count per pixel
5: end global variables
6:
7:  COMPUTE(R)
   Computes an area defined by the rectangle R for the image I with the tracer T.
8:     for y ← min y ∈ R, y < max y ∈ R do
9:         for x ← min x ∈ R, x < max x ∈ R do
10:            for i ← 0, i < S do
11:                I ← T(x, y, I)
12:                i ← i + 1
13:            end for
14:            I(x, y) ← I(x, y)/S
15:            x ← x + 1
16:        end for
17:        y ← y + 1
18:    end for
19: end
```

Tracing process

This section provides detailed insights into the implemented tracing procedure of the thesis project path tracer. The major computations steps are outlined in Algorithm 4.2.

Generating primary ray: In the first step the *tracer* is called by the *tracing engine* (see Algorithm 4.1 line 14) with the given image point I_p. The image position I_p is used to calculate the first ray (*primary ray*) r_0 which is generated by the camera component. The *primary ray* r_0 is used to call the main *tracing method* outlined in Algorithm 4.2 with $depth = 0$.

End criteria and world intersection: Within the *tracing method* two end criteria exist. The first one simply checks the current depth d against a configurable maximum depth $d_{\max}$ and simply returns a black color if $d > d_{\max}$. If $d_{\max}$ is not yet reached the next end criterion is analyzed which needs the scene intersection information for the given ray r_i. If no geometry is hit by the ray r_i, the *tracing method* returns the color of the sky, which can be a constant value or a procedural generated environment.

BRDF calculation: During this step the term $f_r(p, \omega_o, \omega_i)$ of the rendering equation (see Equation 3.1) is calculated, which is the BRDF of the given surface hit point p. This means a new ray r_{i+1} is sampled based on the surface hit information and the given ray r. Also $\cos\theta$ between the surface normal N and the sampled ray r_{i+1} is computed,

Algorithm 4.2: This algorithm shows a simplified outline of the implemented tracing procedure used in the thesis *path tracing* implementation.

```
1: global variables
2:     black, Black color vector
3:     sky, Sky color
4:     maxDepth, Maximum tracing depth
5: end global variables
6:
7: TRACE(r_i, depth)
   Performs the tracing procedure for the given ray r_i and the current recursion depth.
8:     if depth > maxDepth then
9:         return black
10:    end if
11:    hit ← intersect world for r_i
12:    if hit is valid then
13:        r_{i+1} ← sample new ray from Brdf of hit
14:        indirectLight ← TRACE(r_{i+1}, depth + 1)
15:        hitColor ← calculate color
16:        return hitColor
17:    else
18:        return sky
19:    end if
20: end
```

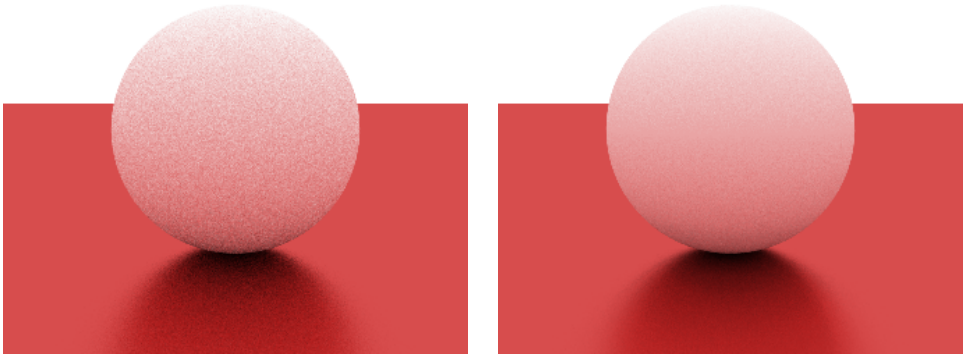
furthermore the BRDF's corresponding probability density function (see Section 3.3.2) is calculated in this step. All this data is needed for the indirect lighting calculation and for general *Monte Carlo integration*. Since the approach of sampling a *bidirectional reflectance distribution function* in *Path Tracing* is a comprehensive topic, Section 4.2 explains this in further detail.

Indirect lighting: In order to calculate *indirect lighting* the term $L_i(p, \omega_i)$ of the rendering equation (see Equation 3.1) for the surface hit point p and the incoming lighting direction $\omega_i = r_{i+1}$ must be evaluated. This is achieved by calling the *tracing function* (see Algorithm 4.2) with the ray r_{i+1} which is sampled by the corresponding BRDF and the incremented depth.

Color composition: In the last step after the recursive *indirect lighting* evaluation, the rest of the rendering equation (see Equation 3.1) is computed and the result is saved into a three dimensional color vector $C_{x,y} = (r, g, b)$.

Further processing

The in Section 4.1.2 described *tracing process* pictures the computational process of calculating a single sample for a given pixel I_p for the image plane I. Usually this whole tracing procedure is performed for the amount of samples N per pixel. For each sample



(a) (b)

Figure 4.2: Both images are rendered with the thesis project path tracer with a sample count of $N = 64$ samples per pixel. The left image (a) is rendered with no stratification and shows a noticeable amount of noise in comparison to the right image (b) which is rendered with the stratified approach.

the result is cumulated and ultimately divided by the number of samples N as defined by the *Monte Carlo estimator* in Section 3.3.

4.1.3 Stratified sampling

As already mentioned in Section 3.4, stratification is a concept to generate better chosen samples which reduces the variance of the *Monte Carlo estimator*, as a consequence this results in less perceptible noise in the final rendered image. This section gives detailed insights into the implemented *stratified sampling* (see Section 3.4.1) approach of the master thesis project.

Use case

Stratified sampling is used in the hemisphere sampling process (see Section 4.2.2) during the BRDF calculation step. The stratified sampler produces better distributed samples on the hemisphere which leads to less variance and therefore to a lower noise level in the final path traced image. Figure 4.2 shows a comparison of rendering outputs with and without stratified sampling, both images were computed with the thesis project renderer.

Generating stratified samples

The implemented *stratified sampling* approach is designed to output even distributed random samples in the domain of a *unit square* where each side has the length of 1. This concept was chosen due to the fact that the hemisphere sampling procedure without stratification calculates hemisphere points based on two uniform random variables ζ_0 and ζ_1 in the domain of $\zeta_0, \zeta_1 \in [0, 1]$. By designing the *stratified sampling* algorithm to output the same range, both strategies can simply be swapped if necessary.

Initialization phase: During the initialization of the stratified sampler, several values can be pre-calculated, these values and their role are visualized in Figure 4.3 (b). The whole domain of the *unit square* is divided in areas, also called *strata* (plural for stratum). For later processing the side length of a stratum $S_{i,j}$ is calculated by

$$s_L = \frac{1.0}{\sqrt{SPP}}, \tag{4.1}$$

where s_L represents the stratum length (see Figure 4.3 (b)) and SPP describes the samples per pixel. Since stratified samples are always generated outgoing from the center of a stratum S_i, the half of the stratum length s_L is also pre-calculated by

$$s_H = \frac{s_L}{2.0}, \tag{4.2}$$

where s_H denotes the half of the stratum length s_L.

Generating indices: Each stratum $S_{i,j}$ (see Figure 4.3 (a)) is bound to a *Monte Carlo* sample N_i. It must be avoided to use a stratum multiple times, this would nullify the idea of stratification and simply provide random sampling. Due to this problem a stratum index $k = \langle i, j \rangle$ is generated for each N_i and saved into an index array $indices = k_0, k_1, \ldots, k_{i-1}$. The index array is also shuffled afterwards to reduce artifacts produces by the the high dimensional rendering equation and possible multiple used stratified samples. Another countermeasure for this problem is performed during the sample calculation step.

Sample calculation: The first step of *stratified sampling* requires to retrieve an index k_i from the index array based on the current *iteration state* of the specific sample. The *iteration state* provides information about the current sample N_i, which is the pixel position I_p and the recursion depth d of the lighting calculation. The index k_i for a given state is calculated as

$$k_i = (N_i + d + I_{px} + I_{py}) \bmod SPP. \tag{4.3}$$

This ensures, for any recursion depth, a different permutation of the sample indices. This approach solves the elimination of artifacts which would occur due to the dimensionality problem of the rendering equation. After the index calculation the actual stratified sample $U_i = \langle u_x, u_y \rangle$ can be calculated as

$$U_i = (k_i \cdot s_L) + s_h + \zeta(-s_h, s_h), \tag{4.4}$$

where U_i is a two dimensional vector which represents a stratified sample. Furthermore, $\zeta(a, b)$ defines a function which returns a random floating point number between the bounds $[a, b]$.

4.1.4 World

The *world* component is responsible for the management of all environmental objects which can be geometry, light sources or cameras. Geometry is delegated to the *aggregation* component to guarantee access to fast ray-scene intersections. Therefore, the main purpose during rendering is to process ray intersection queries and deliver the results.

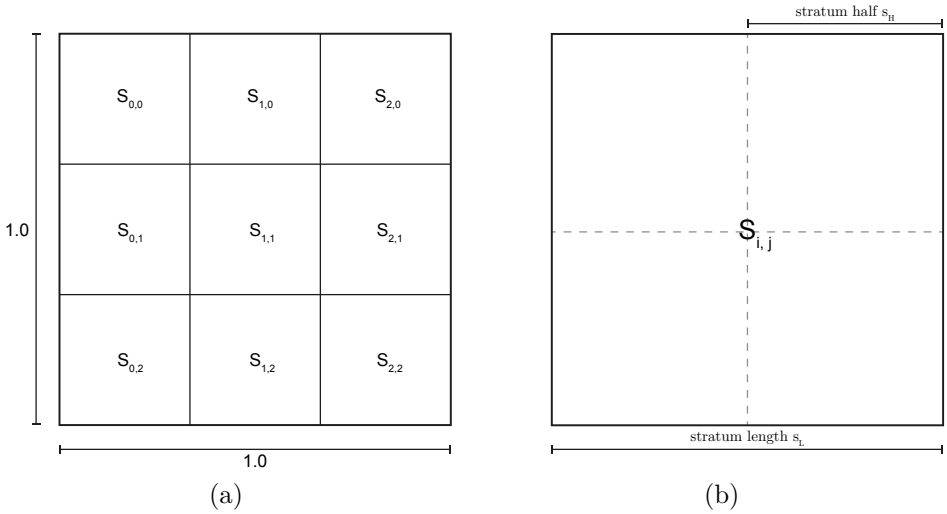


Figure 4.3: In (a) the domain of a unit square, divided in 9 strata $S_{i,j}$, is shown. Furthermore, (b) illustrates a single stratum $S_{i,j}$ and some of its properties which are explained in Section 4.1.3.

4.1.5 Scene aggregation and collision acceleration

The purpose of scene aggregation is to accelerate collision tests of a given ray r and all geometric objects within the environment. A naive way to check for ray-scene collisions is to iterate over all objects g_i, perform intersection tests and return the nearest result. This is however only suitable for a limited amount of objects. Due to this constraint the naive aggregation concept was mainly used during early development stages. Also currently this method is used if the amount of objects within the scene is smaller than a configurable value.

If the scene contains a lot of geometric objects, which is very easily achieved by rendering triangle based meshes, a naive intersection check against each triangle would result in high computation times. Therefore an *Octree* implementation as a true acceleration structure was also developed to speed up collision tests significantly, especially for triangle based meshes. The *Octree* of the thesis project consists of nodes which store a bounding box to describe each node spatially, the data of the node which contains a collection of geometric objects and references to eight child nodes.

The *Octree* construction starts with the root node which covers the whole scene. Each node and its following children are expanded until conditions regarding the maximum depth limit or the minimum of data within a node are valid. The traversal always checks the bounding box of the node first. If the given node is hit and is a leaf, a detailed geometry intersection is computed with the containing data of the node. If the node is not a leaf, the children are checked in a distance sorted manner. This means that the nodes with the smallest intersection distance are checked first in order to omit the later checks if the ray hits one of the nearer nodes.

4.1.6 Film

The *film* component represents a virtual version of a camera film. It stores the state of the processing and final image data I_r and is also the data source for saving the rendered image on the file system. The storage of the film component can be formally described as

$$I_r = \begin{bmatrix} C_{0,0} & \cdots & C_{x,0} \\ \vdots & \ddots & \vdots \\ C_{0,y} & \cdots & C_{x,y} \end{bmatrix}, \tag{4.5}$$

where I_r is a two dimensional matrix which represents the image state. I_r contains three dimensional color vectors $C_{x,y} = (r, g, b)$ to store the respective computed value of each pixel. The dimensions of I_r are configurable within the thesis project application.

4.1.7 Ray

The *ray* component represents light rays in the thesis project path tracer. A ray implements the *parametric line equation* as

$$p = r_0 + t \cdot d, \tag{4.6}$$

where t describes the distance from the origin position r_0 and d represents the direction of the line. The *ray* component is heavily used for intersection tests, which are designed to calculate the distance t. If $t < 0$ further processing can be omitted since the intersection is behind the ray origin r_0 and therefore in a non visible space. However, for valid distances t the hit point p can be easily calculated with the given direction d. Supported intersection routines in the thesis project are ray-plane, ray-box, ray-sphere and ray-triangle collision tests. Due to the support of ray-triangle intersections, also ray-mesh based intersections are supported.

4.1.8 Hit information

The application component *hit information* is created for every ray collision interaction and contains data of a potential collision. The most important data is the actual hit point p and the distance from the ray origin r_o to the surface hit point p in world space. Also geometry based information is stored in order to retrieve data like the surface normal for the hit point p or material based information.

The *hit information* component also provides utility methods to confirm if a potential collision is valid, this is especially used extensively during the central *Path Tracing* routine described in Section 4.1.2.

4.2 BRDF

The goal of this thesis is to combine multiple *bidirectional reflectance distribution functions* via *multiple importance sampling*, therefore this section provides detailed information about the BRDF implementation and representation in the thesis project. Furthermore, the whole process of sampling the BRDF in the *path tracing* context is illustrated and also the connection to the probability density function (see Section 3.3.2) is shown in an example. General theoretical information about BRDFs are covered in Section 3.5.

4.2.1 BRDF representation

As seen in Figure 4.1, the BRDF is part of a *material* component. While the material itself describes surface representation specific meta data like the color or the emissive strength, the BRDF describes in which way a new ray r_{i+1} is generated during the tracing process described in Section 4.1.2. The following *Java* method signature describes the main interface of the BRDF component.

```
Sample sampleRay(Ray incoming, HitInformation hitInfo, IterationInfo iterationInfo,
    SequenceSampler sampler);
```

The `incoming` ray represents the view ray ω_o of the rendering Equation 3.1 in world space. The hit information holds the necessary geometric information like the surface normal N or the hit point p. Furthermore, the `IterationInfo` and `SequenceSampler` parameters are required for stratified sampling explained in Section 4.1.3. A `Sample` is returned as a result which contains the new incident ray ω_i in *world space* and the corresponding probability density function value during the sampling process. Depending on the specific implementation of the BRDF interface, the reflective appearance a surface is defined. A simplified visualization of different BRDF sampling strategies can be viewed in Figure 3.5.

4.2.2 BRDF sampling

This section provides insights on the implemented BRDF sampling approach which is performed during the tracing routine explained in Section 4.1.2. In general the BRDF calculation is responsible for calculating a new incident ray ω_i based on the input parameters described in Section 4.2.1. A step by step explanation independent of the specific BRDF implementation (diffuse, glossy, etc.) is given in this section.

Local coordinate system

The first step in order to sample a BRDF is to create a *local coordinate system*, or *orthonormal basis*, which is usually aligned to the normal N of a given surface hit point p. In this thesis the local coordinate system of the BRDF is also referred as the *hemisphere space*.

The coordinate system of the *hemisphere space* is computed by utilizing a robust orthonormal basis algorithm proposed in [5] by *Duff*. Figure 4.4 (b) shows an illustration of the local *hemisphere space* where N_t (tangent), N and N_b (bitangent) are representing the counterparts of the respective world x-, y- and z-axis.

Generate random sequence

The step described in this section serves as an initialization phase for the hemisphere sampling explained in Section 4.2.2. The goal of this step is to generate two random values $\zeta_0, \zeta_1 \in [0, 1]$. A simple random generator can be used at this point, however in order to achieve a more uniform distribution of random values *stratification* (see Section 3.4) is applied for the random number generation of ζ_0 and ζ_1 in the thesis project.

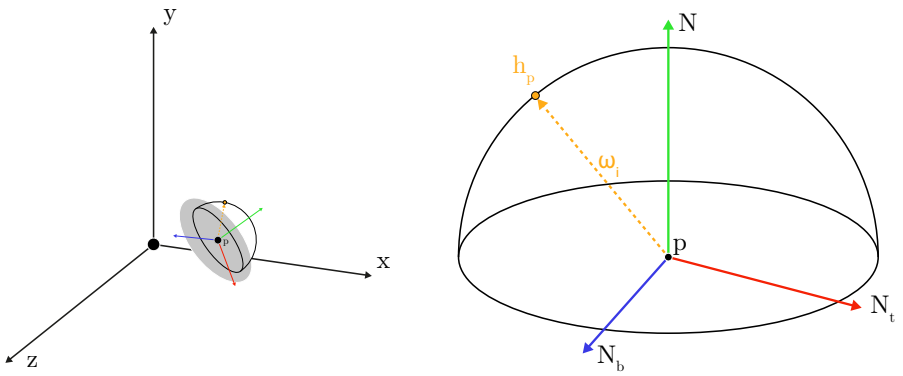


Figure 4.4: In (a) a BRDF hemisphere is shown which is aligned on a grey surface in *world space*. A more detailed variant of the BRDF hemisphere in the local hemisphere space is illustrated in (b). It shows the local coordinate system with the normal N as up-vector as well as N_t and N_b as *tangent* and *bitangent*.

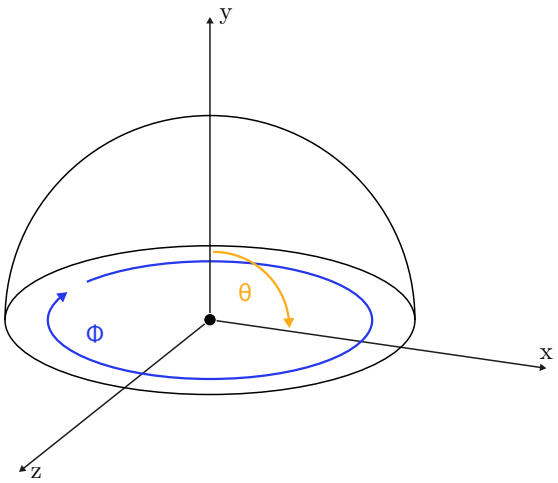


Figure 4.5: This figure illustrates the components of spherical coordinates on a unit hemisphere. In the context of hemispheres θ has a range of $[0, \pi/2]$ while ϕ is given in the range of $[0, 2\pi]$.

Sampling the hemisphere

The goal of this step is to sample points on a hemisphere Ω in terms of *solid angles* and with respect to the given probability density function (pdf) of the respective BRDF. This section outlines the overall steps in order to achieve this task, a more detailed example is given in Section 4.2.3 in which the derivation process for sampling the hemisphere for a diffuse BRDF is shown.

Since the calculation with solid angles is often not practical, spherical coordinates in the context of a unit hemisphere are utilized. Figure 4.5 illustrates a unit hemisphere with $\phi \in [0, 2\pi]$ and $\theta \in [0, \pi/2]$. The transformation to *cartesian* coordinates in respect

to the hemisphere coordinate system of the thesis project is stated as

$$\begin{aligned} x &= \sin\theta\ \cos\phi, \\ y &= \cos\theta, \\ z &= \sin\theta\ \sin\phi, \end{aligned} \tag{4.7}$$

where no radius r is given since a unit hemisphere is assumed. The x, y, z components form the cartesian coordinates while ϕ and θ represent spherical coordinates (see Figure 4.5). As already stated the goal is to achieve a uniform sampling distribution on the hemisphere in respect to solid angles and a given pdf. The end result is a two dimensional function with two random variables $\zeta_0 \in [0,1]$ and $\zeta_1 \in [0,1]$ as input parameters which returns an arbitrary uniform sampled point on a unit hemisphere. The steps to attain this routine can be outlined as the following:

1. Transform the given pdf to a 2D joint probability distribution with spherical coordinates.
2. Separate the 2D joint probability distribution into two 1D (pdf) functions.
3. Apply the 1D inversion technique (see Section 4.2.3).
4. Formulate the transformation function based on previous results.

These steps outline roughly the procedure to achieve a transformation function which receives two random variables ζ_0 and ζ_1 and outputs an arbitrary point on a unit hemisphere. A detailed example to uniformly sample the hemisphere for diffuse reflection is given in Section 4.2.3.

Transform to world space

Since the computed point on the hemisphere h_p is still in the local hemisphere space, h_p is transformed to world space as

$$h'_p = \begin{bmatrix} h_{px} \\ h_{py} \\ h_{pz} \end{bmatrix} \cdot \begin{bmatrix} N_{bx} & N_x & N_{tx} \\ N_{by} & N_y & N_{ty} \\ N_{bz} & N_z & N_{tz} \end{bmatrix}, \tag{4.8}$$

where h'_p is the sampled hemisphere point in *world space*. Furthermore, the vectors $N = (N_x, N_y, N_z)$, $N_t = (N_{tx}, N_{ty}, N_{tz})$ and $N_b = (N_{bx}, N_{by}, N_{bz})$ form the the orthonormal basis which was calculated in the first step described in Section 4.2.2.

Construct sample

After the sampling and transformation steps the final returned *sample* is created which contains the newly constructed incident ray ω_i and the respective probability density function value of the used BRDF. The direction d and the origin point r_0 of the resulting incoming ray ω_i is calculated as

$$\begin{aligned} d &= h'_p, \\ r_0 &= p + (h'_p \cdot bias), \end{aligned} \tag{4.9}$$

where p is the surface hit point and $bias = 0.0001$ to offset the origin r_0 of the new ray ω_i in order to prevent self collision with the geometry during the next recursive tracing step.

4.2.3 Example: Diffuse BRDF

In order to illustrate the outlined description for hemisphere sampling in Section 4.2.2, a detailed step by step approach is given as an example to derivate a sampling function for a uniform sampling distribution on the hemisphere Ω, based on the approach of [14] and [40] which both address this particular example. A uniform distribution results visually in a matt diffuse surface.

Step 1: Acquire the pdf

In the first step the probability density function for the diffuse BRDF must be acquired. Since a diffuse reflection type (see Figure 3.5 (a)) implies that each direction within the hemisphere has the same probability it can be assumed that the probability density function $pdf(\omega) = c$ with respect to solid angle is constant. In order to deduce the pdf, Equation 3.5 can be utilized which states that a given pdf integrates to 1 for its domain. Based on this assumptions, the probability density function $pdf(\omega)$ for the diffuse BRDF can be expressed as

$$\int_0^{2\pi} pdf(\omega)\, d\omega = 1, \tag{4.10}$$

where the domain of $[0, 2\pi]$ is given in solid angle, in which 2π represents the solid angle of the whole hemisphere Ω. The final solution of $pdf(\omega)$, after solving the integral in Equation 4.10, is

$$pdf(\omega) = \frac{1}{2\pi}. \tag{4.11}$$

In order to generate random points on the hemisphere Ω, it is easier to work with spherical coordinates (θ, ϕ), which are mentioned in Section 4.2.2, instead of solid angle. This is achieved by substituting all solid angle components of Equation 4.10 by elements of spherical coordinates. The probability density function $pdf(\omega)\, d\omega$ can be formulated in terms of spherical coordinates θ and ϕ as

$$pdf(\omega)\, d\omega = pdf(\theta, \phi)\, d\theta\, d\phi. \tag{4.12}$$

Furthermore, a differential solid angle $d\omega$ can be expressed with spherical coordinates θ and ϕ as

$$d\omega = \sin\theta\, d\theta\, d\phi. \tag{4.13}$$

Equations 4.12 and 4.13 are necessary for the solid angle substitution and to obtain the 2D joint probability distribution $pdf(\theta, \phi)$ in the context of spherical coordinates. Therefore the differential solid angle $d\omega$ in Equation 4.12 can be substituted by Equation 4.13 which results in

$$pdf(\omega)\ \sin\theta\, d\theta\, d\phi = pdf(\theta, \phi)\, d\theta\, d\phi. \tag{4.14}$$

As seen in Equation 4.14, the differential components $d\theta$ and $d\phi$ can be cancelled out of the equation. Furthermore, $pdf(\omega)$ can be replaced by the already calculated solution of Equation 4.10 which results in

$$pdf(\theta, \phi) = \frac{\sin\theta}{2\pi}, \tag{4.15}$$

where $pdf(\theta, \phi)$ represents the 2D joint probability distribution in respect of spherical coordinates. This combined probability density function $pdf(\theta, \phi)$ is used to attain the single *pdf's* for the θ and ϕ components.

Step 2: Separating the joint probability distribution

In the first step, which is described in Section 4.2.3, the joint probability distribution $pdf(\theta, \phi)$ was determined. However, the independent *pdf's* of θ and ϕ must be acquired in order to apply the later *1D inversion technique* in Section 4.2.3.

The joint probability distribution $pdf(\theta, \phi)$, in the case of sampling the hemisphere uniformly (diffuse reflection), is separable by utilizing the concept of *marginal density* and *conditional density*. Nonetheless, the marginal density is able to deliver the pdf of a single component of a joint distribution function by integrating over the unwanted variable within the domain of the joint distribution function. In order to retrieve $pdf(\theta)$ the marginal density applied as

$$pdf(\theta) = \int_0^{2\pi} pdf(\theta, \phi)\, d\phi, \tag{4.16}$$

where the single pdf for θ is expressed by integrating the joint distribution function over the unwanted variable ϕ. After solving the integral in Equation 4.16, the final result for the separated probability density function $pdf(\theta)$ is

$$pdf(\theta) = \sin\theta. \tag{4.17}$$

To calculate the second pdf for ϕ, the *conditional density* is utilized which calculates the desired pdf through dividing the joint distribution function by an already given singular pdf. Therefore, the pdf for ϕ results in

$$pdf(\phi|\theta) = \frac{pdf(\theta, \phi)}{pdf(\theta)} = \frac{1}{2\pi}, \tag{4.18}$$

where $pdf(\phi|\theta)$ is the probability density function $pdf(\phi)$ given $pdf(\theta)$.

Step 3: Apply the 1D inversion technique

As stated in [14], the *1D inversion technique* starts by using the previously calculated independent density functions $pdf(\theta)$ and $pdf(\phi)$ and integrate them from 0 to θ and respectively ϕ which results in $pdf'(\theta)$ and $pdf'(\phi)$. For $pdf'(\theta)$ this results in

$$pdf'(\theta) = \int_0^{\theta} pdf(\theta)\, d\theta = 1 - \cos\theta. \tag{4.19}$$

To obtain the result for $pdf'(\phi)$ the same approach is applied which results in

$$pdf'(\phi) = \int_0^{\phi} pdf(\phi)\, d\phi = \frac{\phi}{2\pi}. \tag{4.20}$$

In the next step of the *1D inversion technique*, each result of $pdf'(\theta)$ and $pdf'(\phi)$ are put into an equation with random variables $\zeta_0, \zeta_1 \in [0, 1]$ and are rearranged in order

to solve the corresponding equation by θ and ϕ. For θ the equation results in

$$\begin{aligned} \zeta_0 &= 1 - \cos\theta, \\ \theta &= \cos^{-1}(1 - \zeta_0). \end{aligned} \tag{4.21}$$

The same concept is applied for ϕ which is defined as

$$\begin{aligned} \zeta_1 &= \frac{\phi}{2\pi}, \\ \phi &= 2\pi \cdot \zeta_1. \end{aligned} \tag{4.22}$$

Step 4: Transformation function

The meaning of the Equations 4.21 and 4.22 is that θ and ϕ can be uniformly sampled by simply providing random numbers ζ_0 and ζ_1 in the range of $[0, 1]$. However, in order to obtain applicable values for hemisphere sampling, the spherical coordinates (θ, ϕ) must be converted to the cartesian system by utilizing Equation 4.7. The calculation of θ and ϕ, as well as the transformation to the cartesian system is shown by the following *Java* code snippet:

```
1 public static Vector3d uniformSampleHemisphere(double zeta0, double zeta1) {
2     double sinTheta = Math.sqrt(1.0 - (zeta0 * zeta0));
3     double phi = TWO_TIMES_PI * zeta1;
4     double x = sinTheta * Math.cos(phi);
5     double z = sinTheta * Math.sin(phi);
6     return new Vector3d(x, zeta0, z);
7 }
```

Note that in line 2 the calculation is based on following equation:

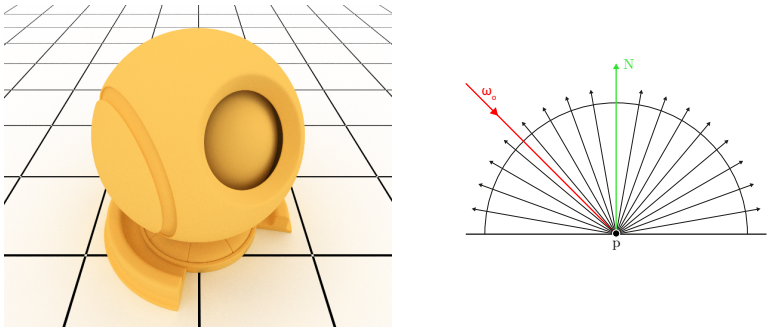
$$\begin{aligned} \sin^2\theta + \cos^2\theta &= 1, \\ \sin\theta &= \sqrt{1 - \cos^2\theta}, \\ \sin\theta &= \sqrt{1 - \zeta_0 \cdot \zeta_0}. \end{aligned} \tag{4.23}$$

The result $\sin\theta$ represents the calculation in line 2. The substitution in Equation 4.23 of $\cos^2\theta$ only uses ζ_0 from Equation 4.21 instead of the full term $1 - \zeta_0$ as a minor optimization step since it does not alter the probability of the chosen random number in the domain of $[0, 1]$.

Also note that the return statement in line 6 provides only the random variable ζ_0 for the y-component of the 3D vector Also note that the return statement in line, although Equation 4.7 states that $y = \cos\theta$. However in Equation 4.21 it is described that $\cos\theta = 1 - \zeta_0$, which can again be simplified by only using the random variable ζ_0.

4.2.4 Implemented BRDF types

Several types of *bidirectional reflectance distribution functions* were implemented in the thesis project. This section provides an overview of the BRDFs which are incorporated in the thesis project. However, this covers only directly implemented BRDFs and not BRDFs which are generated through *multiple importance sampling*. The outcomes of



(a) (b)

Figure 4.6: In (a) an object rendered with a diffuse BRDF is shown which is created with the thesis path tracer and 256 samples per pixel. Furthermore, (b) illustrates the sampling concept for diffuse surface types with the ray view ray ω_o and the normal N of the hit point p.

combined BRDFs via MIS are presented in Chapter 5 and in Chapter 6. Furthermore, a detailed general explanation on how to sample BRDFs is already provided in Section 4.2.2 and therefore this section is dedicated to the variations of the general approach which occur in the different BRDF implementations.

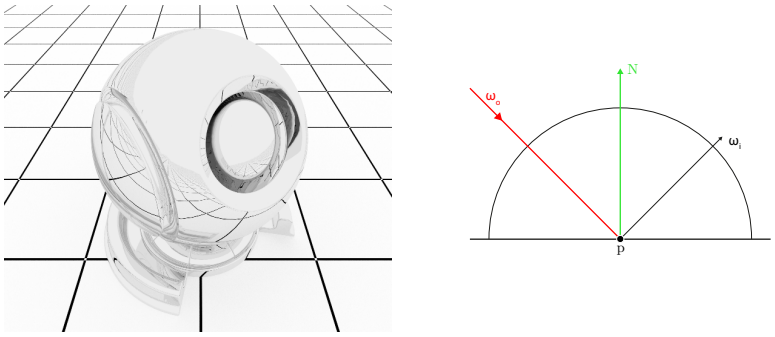
Diffuse BRDF

As illustrated in Figure 4.6 (b) for diffuse surface types, the hemisphere Ω, which is aligned to the normal N of the hit point p, is sampled uniformly. This means that each direction for the generated ray ω_i within the hemisphere has an equal probability. As a consequence, for the point p the resulting light is always the same, regardless of the view ray ω_o. Therefore the ray ω_o has no impact on the sampling of ω_i which can be also described as *camera independent*.

The result of an equally sampled hemisphere is very matte surface as shown in the rendering in Figure 4.6 (a) which was computed with the thesis path tracer. In the thesis project the diffuse material can be visually adjusted only by a color parameter.

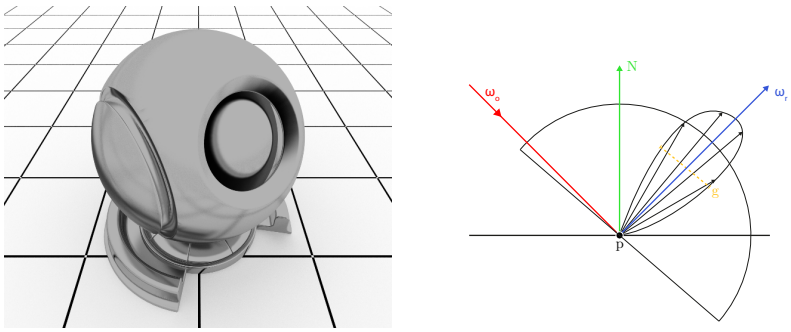
Mirror BRDF

The mirror BRDF is a very simple reflectance function since it skips the hemisphere sampling step (see Section 4.2.2) completely due to the fact that the incident ray ω_i can be easily calculated directly based on the view ray ω_o and the surface normal N. This is also illustrated in Figure 4.7 (b). Therefore, the new ray ω_i can be calculated by reflecting ω_o at the given surface normal N, as a consequence this BRDF is view dependant. The resulting visual appearance provides a surface representation which reflects its environment similar to a mirror as it can be seen in Figure 4.7 (a), where the grid lines of the ground can be seen as reflections on the surface of the centered object in the scene. Furthermore, the material of the mirror reflection type supports color tinting to achieve the desired visual look.



(a) (b)

Figure 4.7: The outcome in (a) shows thesis path tracer result with an applied mirror BRDF, which was rendered with 256 samples per pixel. Additionally, (b) illustrates the sampling of the incident ray ω_i on the hemisphere based on the view ray ω_o and the surface normal N.



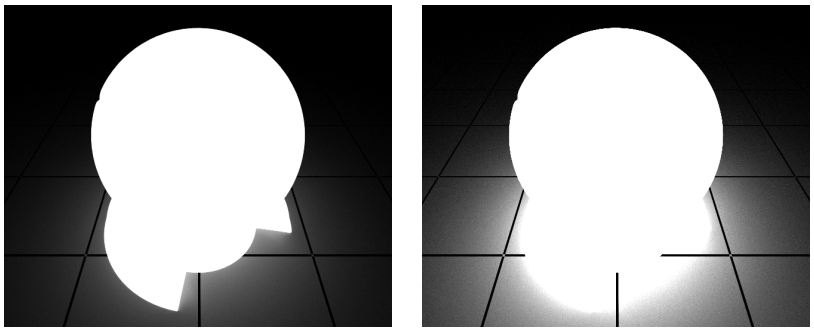
(a) (b)

Figure 4.8: Example (a) displays a computed image (256 samples per pixel) of the thesis path tracer with an applied glossy BRDF. Moreover, (b) shows the hemisphere sampling procedure for glossy reflections, where g represents the amount of glossiness of the BRDF and ω_r shows the reflected ray of ω_o.

Glossy BRDF

A glossy reflection type can be seen as a more sophisticated version of the mirror BRDF stated in Section 4.2.4. For glossy reflections the view vector ω_o is also reflected which can be seen in Figure 4.8 (b) as ω_r. The hemisphere sampling step for glossy BRDFs does not orient the hemisphere Ω around the surface normal N, the hemisphere is aligned to the reflection ray ω_r which is also illustrated in Figure 4.8 (b).

The sampling of the hemisphere is not performed over the whole hemisphere uniformly, a so called *cosine power distribution* is computed which enables to concentrate the sampling on a specific area of the hemisphere in respect to the solid angle. This is controlled via the glossiness exponent g which specifies the expansion of the cosine lobe visualized in Figure 4.8 (b). Higher values of g result in smaller lobes and therefore sharper reflection, while lower values are producing a wide sampling which leads to blurry reflections of the environment.



(a) (b)

Figure 4.9: Both renderings (a) and (b) show examples of the emission based BRDF with a sample count of 256 per pixel. Both images are computed with the thesis renderer and visualize different emission values for L_e.

The glossy reflection type of the thesis project supports the control of the glossiness g and the color of the surface. The glossiness g can be used to visualize nearly matte blurry reflections or mirror like reflections with a high glossiness value. Figure 4.8 (a) displays an rendering example of a glossy reflection surface type computed with the thesis renderer showing blurry reflections on the lower side of the centered object.

Emissive BRDF

An emissive surface actively casts light into the scene environment. As a consequence, nearby objects of a light emission based surfaces will be illuminated. In the case of the BRDF in the thesis path tracer a simple constant L_e, which refers to the emission term of the rendering equation (see Equation 3.1), defines the strength of the emission. Due to the fact that emissive surfaces are handled during the tracing process described in Section 4.1.2 the BRDF is independent of the given ray ω_o and serves simply as a data container. Figure 4.9 shows two different examples of a geometric objects utilizing the emissive BRDF to illuminate a dark scene.

Chapter 5

Multiple importance sampling for BRDFs

Multiple importance sampling was introduced in Section 2.1 and provides a solution to combine multiple sampling techniques in the context of *Monte Carlo integration*. *Bidirectional reflectance distribution functions*, as described in Section 4.2, can be also interpreted as sampling technique, at least in the context of hemisphere sampling during the path tracing procedure.

Based on the idea of picturing the BRDF as sampling technique, this thesis attempts to use *multiple importance sampling* to combine arbitrary BRDFs. The approach for the BRDF blending via MIS is provided in this chapter which starts by outlining encountered problems during the prototyping phase of this BRDF combination technique. Furthermore, necessary rearrangements of the MIS equation are discussed, also detailed insights into the actual implementation are given. Moreover, examples are provided which are rendered with the MIS approach for BRDFs.

5.1 Initial problems

In order to achieve a better understanding for the encountered problems during the initial phase of the implementation of MIS for BRDF blending, this section revisits shortly the *Monte Carlo* and MIS estimator which is followed by a short example. The specific problem is also explained based on the given example of this section.

5.1.1 Monto Carlo and MIS estimator revisited

As already stated in Section 3.3.1, the expected value $E[F_N]$ for an arbitrary integral $\int f(x)\,dx$ can be calculated with the *Monte Carlo estimator* which is defined as

$$E[F_N] = \frac{1}{N}\sum_{i=1}^{N}\frac{f(X_i)}{pdf(X_i)}, \tag{5.1}$$

where N denotes the amount of samples for the *Monte Carlo* calculation and $pdf(X_i)$ represents the suitable probability density function for the function $f(x)$.

The *multi-sample estimator*, as already stated in Section 2.1.3, is designed to combine different sampling techniques and especially defined to fuse an arbitrary amount of

functions which are solved via the *Monte Carlo estimator*. This appears more clearly by observing the *multi-sample estimator*

$$F = \sum_{i=0}^{m} \left[\frac{1}{N_i} \sum_{j=0}^{N_i} w_i(X_{i,j}) \frac{f(X_{i,j})}{pdf_i(X_{i,j})} \right], \tag{5.2}$$

where F represents the expected result for an arbitrary amount of sampling functions m. The inner summation is a *Monte Carlo estimator* modified with an weighting function $w_i(X_{i,j})$. The outer sum simply adds up weighted *Monte Carlo estimator* results which represent the outcomes of the respective sampling technique.

5.1.2 MIS estimator example

The example given in this section utilizes Equations 5.1 and 5.2 from Section 5.1.1 and applies two placeholder functions $f(x)$ and $g(x)$ to the equations. This walk through may seem obvious, but the overall problem of using the original MIS equation for BRDFs is better clarified as a consequence. For two sampling strategies $f(x)$ and $g(x)$ the *multi-sample estimator* results in

$$F = \frac{1}{N_0} \sum_{j=0}^{N_0} w_0(X_{0,j}) \cdot \frac{f(X_{0,j})}{pdf_f(X_{0,j})} + \frac{1}{N_1} \sum_{j=0}^{N_1} w_1(X_{1,j}) \cdot \frac{g(X_{1,j})}{pdf_g(X_{1,j})}, \tag{5.3}$$

where the outer sum of the *multi-sample estimator* in Equation 5.2 is written explicitly. Note that each addend is a *Monte Carlo estimator* for the respective function $f(x)$ and $g(x)$ and the respective sampling counts N_0 and N_1.

5.1.3 Problem for BRDF blending

As observable in Equation 5.3, for two given sampling strategies $f(x)$ and $g(x)$ the *multi-sample estimator* results in two weighted *Monte Carlo estimators*. This means that the whole tracing process (see Section 4.1.2) would be executed twice in this case. For the intended scenario of blending *direct light sampling* and *BRDF sampling* together (see Section 2.1.4), this seems perfectly suitable since the amount of sampling strategies is very low. If the same concept is used to blend BRDFs together, each BRDF, which is treated as a sampling strategy, causes another pass of the whole tracing process.

By assuming the same sampling count N for every sampling strategy, the disadvantage of multiple *Monte Carlo estimator* runs can be eliminated by developing different *tracer* implementations, where each *tracer* resembles a *Monte Carlo integrator* with a specialized purpose. This is done in [21] by *Veach* where a *tracer* component was implemented for *direct light sampling* and *BRDF sampling*.

However, in the context of BRDF blending the effort to develop a whole *tracer* for each BRDF is not practical. The initial idea was to somehow fit the BRDF combination feature into the current tracing pipeline of the thesis project which offers a single *tracer* component to render the scene. This restriction also ensures an easy integration into existing rendering pipelines like the in the thesis project itself. Due to this constraint a notion emerged to apply the BRDF blending via *multiple importance sampling* on a deeper level, which is the BRDF sampling (see Section 4.2.2) step during the tracing process.

5.2 Adapting the MIS equation

As already mentioned in Section 5.1.3, the goal which resulted from the initial problematic attempt for BRDF blending was to integrate the BRDF combination via *multiple importance sampling* into the existing BRDF sampling procedure. In order to accomplish this task, a number of conditions for the *multi-sample estimator* in Equation 5.2 are defined which are described in this section.

5.2.1 MIS conditions for BRDFs

In order to integrate MIS for BRDFs within the BRDF sampling process of the thesis path tracer, a number of conditions emerged while experimenting with the *multi-sample estimator* and during the prototyping phase of the BRDF blending feature. This section describes this set of conditions and the impact on the original *multi-sample estimator* which is used ultimately in a modified form.

Limitation of BRDFs to combine

The first condition is to limit the amount of combined BRDFs to exactly two at a given time. Equation 5.3 in Section 5.1.2 already expresses this situation with two sampling techniques, or in this case BRDFs, $f(x)$ and $g(x)$. However, this limitation does not mean that only a total amount of two BRDFs can be mixed. In order to combine an arbitrary amount of BRDFs a nesting approach is possible, for instance the BRDF $g(x)$ can again contain a whole *multi-sample estimator* with two other BRDFs $h(x)$ and $i(x)$.

Sampling count for each BRDF

A *bidirectional reflectance distribution function* is sampled within the tracing process which is calculated N times as stated in the *Monte Carlo estimator* in Equation 5.1. As a consequence a BRDF is sampled exactly one time during a single tracing pass N_i. Due to this interpretation, Equation 5.3 for the two BRDFs $f(x)$ and $g(x)$ can be rearranged by assuming that the sampling count $N_0 = 1$ and $N_1 = 1$ for both BRDFs. Therefore, the simplified equation can be formulated as

$$F = w_0(X_0) \cdot \frac{f(X_0)}{pdf_f(X_0)} + w_1(X_1) \cdot \frac{g(X_1)}{pdf_g(X_1)}, \tag{5.4}$$

where F is the result the two BRDFs $f(X_0)$ and $g(X_1)$ which are weighted with $w_0(X_0)$ and $w_1(X_1)$. Note that the summation and the division by the sampling count of each sampling technique are eliminated from the original Equation 5.3.

Probabilistic interpretation

As stated in Section 3.5.3, the visual appearance of a surface in *Path Tracing* is primarily influenced by the way how new rays are sampled in the hemisphere which is defined by a BRDF. With this in mind, the simplified Equation 5.4 for combining two BRDFs via MIS states that the weighted sum of both BRDFs results in the combined BRDF. However, in the context of generated rays, a meaningful summation of two independent rays is hard to find and several attempts led to visually wrong results.

However, by interpreting the summation of the two BRDFs in Equation 5.4 by choosing one of the BRDFs $f(x)$ and $g(x)$ with their respective weight, visually correct results were achieved. The weightings $w_0(x)$ and $w_1(x)$ from Equation 5.4 are now serving the purpose of choosing probabilities where the conditions $w_0(x), w_1(x) \in [0, 1]$ and $w_0(x) + w_1(x) \leqslant 1$ must be satisfied.

5.3 Implementing MIS for BRDFs

This section provides information about the implementation of MIS for BRDFs into the render pipeline of the thesis project. First, a short overview is given about general implementation decisions for sampling the BRDF via MIS. Furthermore, details about the sampling execution flow and the actual implementation are given. Additionally the developed heuristics for the *choosing weightings* are described.

5.3.1 Overview

Since the purpose of BRDFs in *Path Tracing* is to calculate a new ray ω_i based on the given view ray ω_o, the BRDF component is part of a *material* component which is illustrated in Figure 4.1. A material holds information like the color or the emission value of a surface, which is not necessarily a direct part of the BRDF. However, since it should be possible to combine all properties of a surface, the sampling process on which BRDF to choose happens in the *material* component level.

This special kind of material is implemented in this thesis as a so called *MIS material* which handles the whole sampling process. The *MIS material* contains two materials m_0 and m_1 which are combined and a *heuristic* for the weighting calculation (see Section 5.3.3). As a sampling result the *MIS material* provides one of the given materials based on the provided heuristic.

5.3.2 General flow of execution

In order to understand where the BRDF combination via MIS fits into the application structure of the thesis project, the general execution flow is described step by step in this section. Figure 5.1 illustrates the material sampling procedure starting from the tracing function.

As described in Section 4.1.2, the BRDF sampling process and all material specific values are calculated and used during the tracing process if a valid surface hit occurs. However, with the new BRDF blending system it must be ensured that a material with all its properties and corresponding BRDF is chosen once during a tracing pass.

The choosing or sampling of a material is performed by executing the method `sampleMaterial()` (see Figure 5.1) of the surface hit material which is provided by the hit information of the scene intersection during the tracing process.

In the next step the *MIS material* is responsible to choose one of the containing materials m_0 and m_1. This is done by generating a random number $\zeta \in [0, 1]$. If $\zeta = 0.5$ the resulting material is simply returned in a random way. Otherwise, the heuristic calculation is triggered which provides probability values for each material m_0 and m_1. Based on the calculated probabilities w_0 and w_1 the respective material is returned.

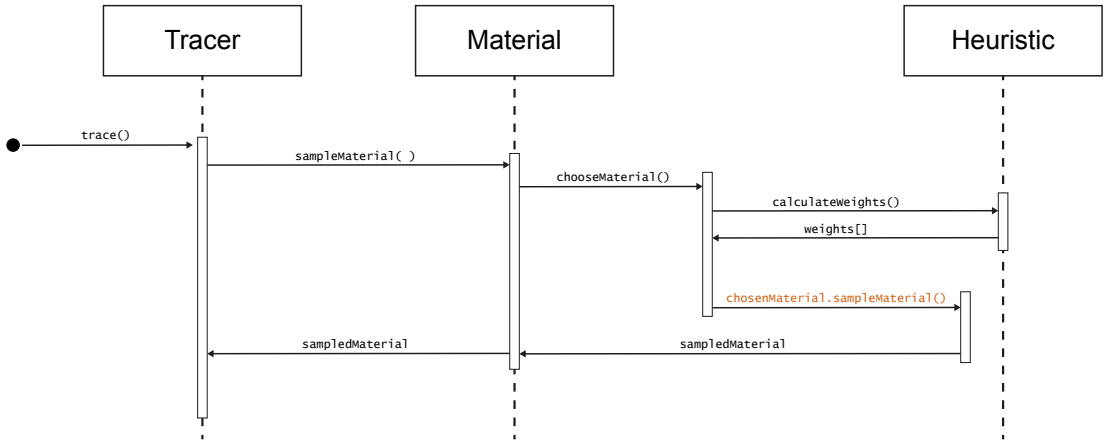


Figure 5.1: This sequence diagram shows a simplified sequence which is processed when a material is sampled during a tracing pass. The illustration shows the flow of the calculation and the involved components. The orange method call indicates that the sampling process is performed for the chosen material.

However, since one of the materials m_0 and m_1 can potentially be nested *MIS materials*, it is necessary to execute the sampling procedure of the chosen material. This is highlighted in Figure 5.1 as an orange method call. Regular material implementations simply return themselves while the specialized *MIS material* implementation performs a probability based sampling.

5.3.3 Heuristics

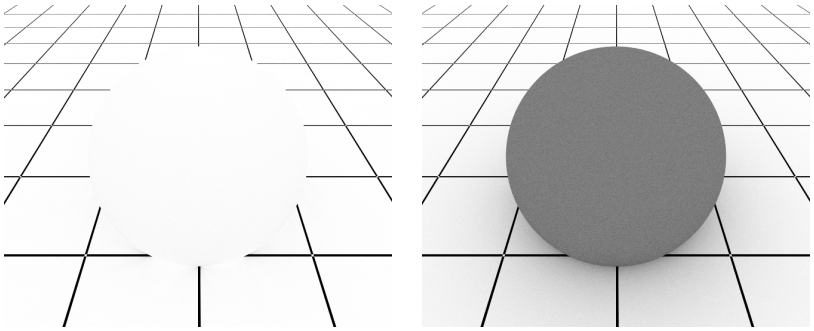
As explained in Section 2.1, *Veach* described the weighting functions for *multiple importance sampling* as *heuristics*, the same name is applied to the probability functions $w_0(x)$ and $w_1(x)$ in this thesis, as they have a similar purpose despite the slightly different context. This section provides an overview of all implemented heuristics with visual examples to attain a better understanding for the actual use case.

Heuristic interface

In order to calculate the probability values for *multiple importance sampling* within a specific heuristic implementation, the general interface is defined by passing the view ray ω_o and the hit information, both in world space, into the calculation method. The interface in *Java* is defined as follows:

```
double[] calculateWeights(Ray viewRay, HitInformation hit);
```

The return value is an array of double values which represent the weightings w_0 and w_1 respectively for the materials m_0 and m_1. As already mentioned in Section 5.2.1, the properties $w_0, w_1 \in [0, 1]$ and $w_0 + w_1 \leqslant 1$ must be satisfied.



(a) $w_0 = 1.0$, $w_1 = 0.0$ (b) $w_0 = 0.5$, $w_1 = 0.5$

Figure 5.2: The renderings (a) and (b) are created with the thesis path tracer with 400 samples per pixel. The images are visualizing an applied constant heuristic between two diffuse materials m_0 with a white color and m_1 with a black surface.

Constant heuristic

The *constant heuristic* is a very simple approach to calculate the probability weightings for two materials m_0 and m_1. As input, the *constant heuristic* receives the desired probability w_0 for the first material m_0 as an input value during the initialization. The second weighting can be easily calculated by

$$w_1 = 1 - w_0. \tag{5.5}$$

Figure 5.2 shows an example of the implemented *constant heuristic* were m_0 resembles a matte perfect white material and m_1 a diffuse black surface. As illustrated in Figure 5.2 (a), the weighting $w_0 = 1.0$ enables a 100% probability for material m_0 and the result is therefore a perfectly white sphere.

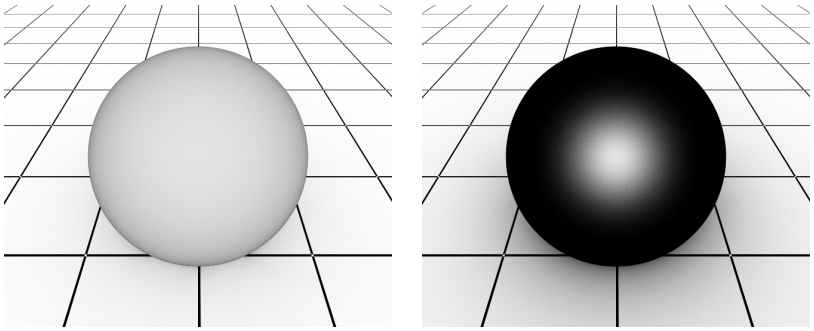
In the second example shown in Figure 5.2 (b), both materials m_0 and m_1 have an equal probability where $m_0 = m_1 = 0.5$ which results in an equal mix of their properties, which in this case only differ by the surface color. As a consequence of a mix of black and white surface colors, the final color in Figure 5.2 (b) results in a middle grey ($r = 0.5, g = 0.5, b = 0.5$).

Facing ratio heuristic

The *facing ratio heuristic* is view dependent an relies on the view ray ω_o and the surface hit normal N. The weights w_0 and w_1 are defined as

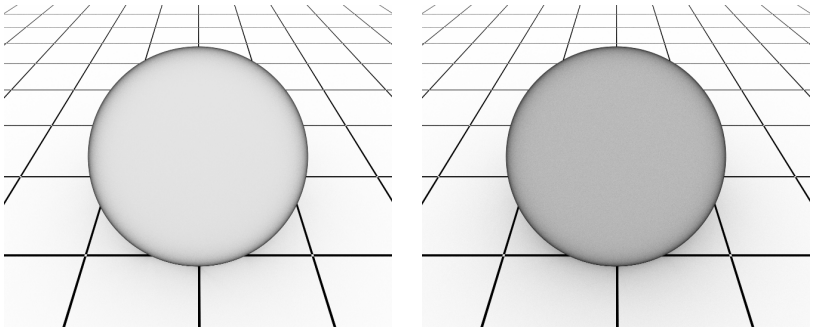
$$\begin{aligned} w_0 &= \begin{cases} 0 & \text{if } \mid N \cdot \omega_o \mid^{\beta} < 0, \\ 1 & \text{if } \mid N \cdot \omega_o \mid^{\beta} > 1, \\ \mid N \cdot \omega_o \mid^{\beta} & \text{otherwise,} \end{cases} \\ w_1 &= 1 - w_0, \end{aligned} \tag{5.6}$$

where N is the surface normal of a hit point p and ω_o represents the view ray. The facing power β controls the strength of the effect which can be seen in a visualized form in Figure 5.3. The main contribution for the weighting calculation is the dot product



(a) $\beta = 0.2$ (b) $\beta = 10.0$

Figure 5.3: Both outcomes (a) and (b) are computed with 900 samples per pixel by the thesis path tracer. The *facing ratio heuristic* is illustrated in both examples with a different facing power β. The first material m_0 is a 90% white color while m_1 is black, both a rendered with a diffuse BRDF.



(a) $\eta = 1.0$ (b) $\eta = 2.5$

Figure 5.4: The results in (a) and (b) are rendered with 900 samples per pixel in the thesis renderer. Both images show the *Fresnel heuristic* with different values for the *index of refraction* (η). Both spheres are rendered with a diffuse BRDF, and the blending happens between a 90% white color an a black surface.

between N and ω_o which results in the highest probability for w_0 if N and ω_o are parallel. The lowest probability for w_0 originates if the normal N and the view ray ω_o are perpendicular.

The *facing ratio heuristic* is visualized in Figure 5.3 (a) with $\beta = 0.2$ and Figure 5.3 (b) with $\beta = 10.0$. Both images are rendered with a diffuse BRDF in the thesis path tracer. As it can be seen in Figure 5.3 the *facing ratio heuristic* can be used to blend two materials m_0 and m_1 based the ratio between the surface normal N and the view ray ω_o while using a controlling value β to adjust the strength of the effect.

Fresnel heuristic

The *Fresnel heuristic* (see Figure 5.4) is similar to the *facing ratio heuristic* described in Section 5.3.3, as a consequence the *Fresnel heuristic* depends on the view ray ω_o and the surface normal N. In the context of rendering surfaces, the *Fresnel equation*

is commonly used to compute how much a surface reflects or refracts incoming light based on the index of refraction η. In case of performing *multiple importance sampling* for BRDFs, the *Fresnel equation* is used to calculate the probabilities w_0 and w_1 for the materials m_0 and m_1. Figure 5.4 visualizes the heuristic. The result of the *Fresnel heuristic* can be interpreted as a mask for the materials m_0 and m_1.

In order to compute the *Fresnel equation* the approximation of *Schlick* is utilized as stated in [35]. *Schlick's* approximation is defined as

$$\begin{aligned} R &= R(0) + (1 - R(0)) \cdot (1 - cos(\theta))^5, \\ R(0) &= \left(\frac{\eta_0 - \eta_1}{\eta_0 + \eta 1} \right)^2, \end{aligned} \tag{5.7}$$

where $R \in [0, 1]$ the amount of reflection for a surface. However, in case of the heuristic, R describes the probability of the material m_1. The reflectance at 0° is defined by $R(0)$ which consists of a calculation between two index of refraction values η_0 and η_1 for two different mediums [35]. In case of the heuristic, it is assumed that a given light ray always passes from air to another medium, as a consequence $\eta_0 = 1$. Based on *Schlick's* approximation defined in Equation 5.7, the probability weightings w_0 and w_1 can be calculated as

$$\begin{aligned} w_0 &= 1 - R, \\ w_1 &= R. \end{aligned} \tag{5.8}$$

Certainly the calculation for w_0 and w_1 can be swapped which results in an inverted result. However, this constellation is chosen in order to have a consistent result compared to the *facing ratio heuristic* described in Section 5.3.3. An example of the *Fresnel heuristic* is shown in Figure 5.4 (a) with $\eta = 1.0$ and Figure 5.4 (b) with $\eta = 2.5$. The solution is similar to the *facing ratio heuristic* shown in Figure 5.3. However, the *Fresnel heuristic* has a higher focus on areas where $N \cdot \omega_o$ results in small values which is the sphere border in the given example in Figure 5.4. Also a greater η has different effects than raising the facing power of the *facing heuristic* as it can be seen in Figure 5.4 (b).

Reverse heuristic

The *reverse heuristic* is intended to be a utility heuristic to reverse the effect of any other given heuristics. As a consequence, the *reverse heuristic* receives an arbitrary heuristic h_i and calculates the reversed weightings w_0' and w_1' as

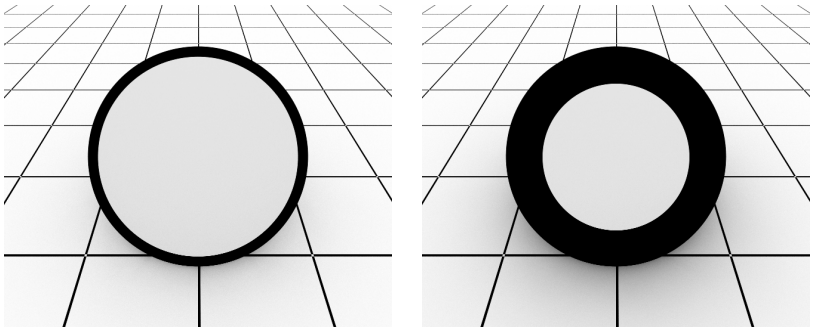
$$w_0' = 1 - w_0, \tag{5.9}$$

$$w_1' = 1 - w_1, \tag{5.10}$$

where w_0 and w_1 are the original weightings of the heuristic h_i. As a result, the visual impact concludes in an inverse result of the given heuristic h_i.

Threshold heuristic

The *threshold heuristic* is another utility heuristic which is designed to modify the original weightings w_0 and w_1 of an arbitrary heuristic h_i based on a given threshold τ.



(a) $\tau = 0.01$ (b) $\tau = 0.2$

Figure 5.5: The renderings (a) and (b) are both computed with the thesis path tracer with 400 samples per pixel. Both examples illustrate the effects of different *threshold heuristic* values τ in combination with the *facing heuristic* with $\beta = 5$.

The thresholding is defined as

$$w_0' = \begin{cases} 1, & \text{if } w_0 \geqslant \tau, \\ 0, & \text{if } w_0 < \tau, \end{cases} \qquad (5.11)$$
$$w_1' = 1 - w_0',$$

where w_0' and w_1' represent the new calculated weightings based on the threshold τ. An example is shown in Figure 5.5 where a *facing heuristic* with $\beta = 5$ is limited by a *threshold heuristic* with two different values for τ. As shown in Figure 5.5 (b) the white material m_0 has a 100% probability to be sampled if its weighting w_0 is above 20% ($\tau = 0.2$), while a value below 20% leads to sample only the black material m_1 is shown in Figure 5.5 (b).

Chapter 6

Evaluation

The approach of utilizing the *multiple importance sampling* concept for blending multiple BRDFs provides in theory a great possibility to create complex surface representations based on multiple simple materials. This chapter evaluates the results of the strategy to combine an arbitrary amount of BRDFs via MIS as introduced in Chapter 5.

6.1 Evaluation methods

In order to evaluate the implementation of *multiple importance sampling* for BRDFs, several methods are defined to examine the results of the provided implementation described in Chapter 5. Therefore, this section illustrates the chosen evaluation methods which concentrate on general rendering comparisons between the thesis renderer and the *Cycles* renderer of *Blender*. Furthermore, internal comparisons regarding the noise level and the computing time between regular materials and MIS based materials are given. The evaluation renderings are all computed with a resolution of 650×540, the CPU of the testing machine is an Intel Core i7-4790K with 4×4.00 GHz with 16 GB RAM.

6.1.1 Blender result comparison

The goal of this evaluation method is to verify the capabilities of the implemented BRDF combination approach via MIS in comparison to an existing rendering solution which also provides a way to combine materials. Blender is chosen for this comparison, since it also provides a mechanism to blend multiple BRDFs and due to the possibility to craft surface representations easily with an node based editor.

6.1.2 Noise comparison

This evaluation method recreates several simple BRDFs which are mentioned in Section 4.2.4 with a combination of two materials. The goal is to clarify if the *multiple importance sampling* approach for BRDFs introduces visible noise in the render result.

6.1.3 Render time comparison

Similar to the evaluation strategy in Section 6.1.2, this method aims to assess the computation time regarding a combined BRDF and a directly implemented BRDF of the thesis renderer.

6.2 Evaluation results

This section provides the results of the evaluation methods described in Section 6.1. The general procedure of the information retrieval and the used data and parameters are illustrated in this chapter. An interpretation and discussion of the evaluation results is provided in Chapter 7.

6.2.1 Blender result comparison

In order to verify the capabilities of the BRDF combination approach of this thesis, several material types, which can be created by combining simple BRDFs, are chosen to be recreated with the thesis BRDF blending strategy. The rendering results are compared with the *Cycles* renderer of *Blender* which enables to recreate the materials which are designed in the thesis renderer.

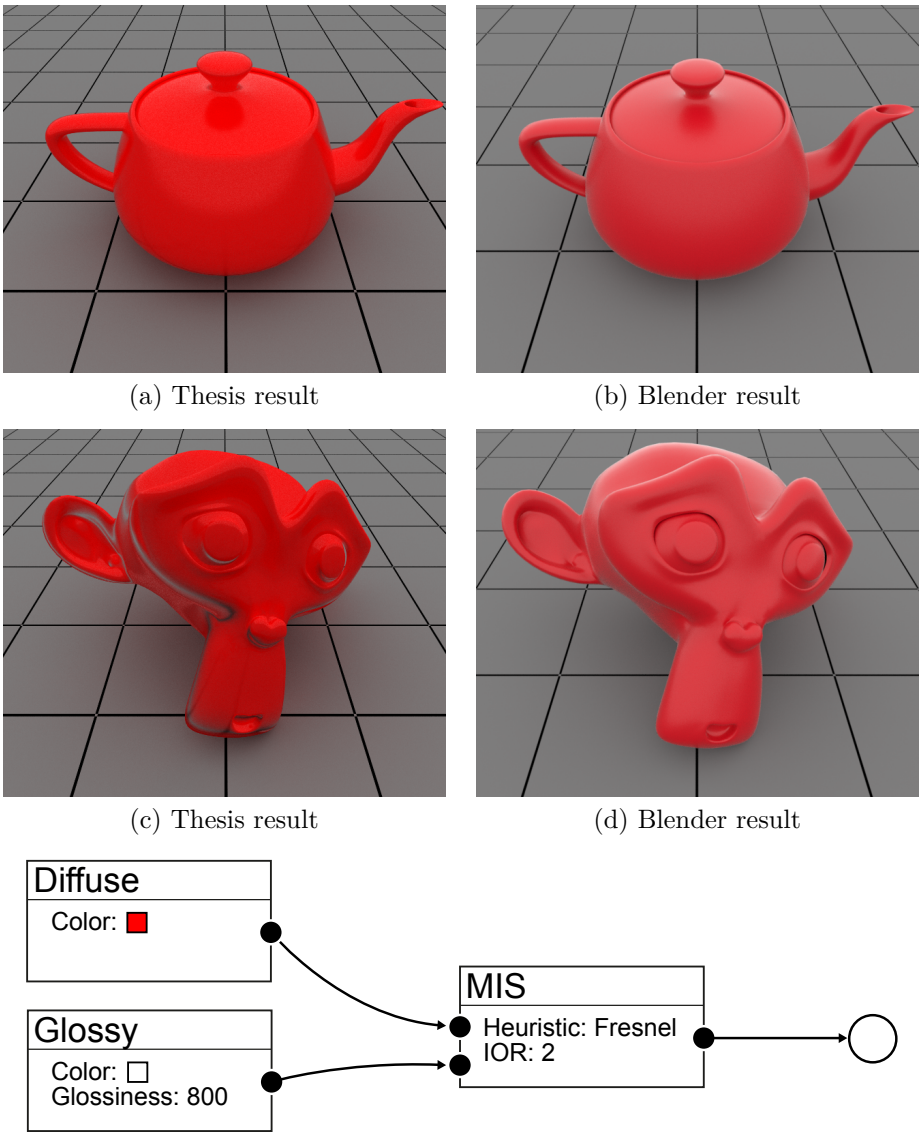
In order to achieve significant comparison results, it is attempted to equalize the most important global render settings in *Blender* and the thesis path tracer. During the whole comparison the samples per pixel (SPP) is set to 256 while the maximum tracing depth $d_{\max} = 5$, furthermore *stratification* is enabled in both renderers. However, *Blender* represents a professional solution for highly realistic rendering, therefore a lot of properties are given which are not implemented in the thesis tracer. This applies also to the complex *color management* feature of *Blender* where the exposure is set to 1 and the *gamma correction* to 2.2 in order to match the thesis results as close as possible, regarding the color management.

Note that the visual appearance is considered in this comparison and not the computation time, since *Blender* offers a highly optimized BVH (bounding volume hierarchy) [29] for ray-scene intersection which outperforms the *octree* implementation of the thesis.

Plastic material

A glossy plastic like material is created by utilizing a diffuse and glossy component (see Figure 6.1 (e)) with a high glossy value, which is 800 in the thesis renderer. Both components are combined via *multiple importance sampling* by using the *Fresnel heuristic* with a refraction index $\eta = 2.0$. The *Fresnel heuristic* enables to assign the glossy and diffuse BRDF corresponding to the viewing direction ω_o.

In *Blender* the plastic material is built as close as possible to the approach of the thesis which is shown in Figure 6.1 (e). However, an exact recreation of the BRDF properties is not achievable since *Blender* uses a different model for glossy reflections where the glossiness value ranges between 0 and 1. The node editor of *Blender* also provides a *Fresnel* component which can be adapted in order to be functional for the *Blender* internal BRDF blending mechanism.



(a) Thesis result (b) Blender result

(c) Thesis result (d) Blender result

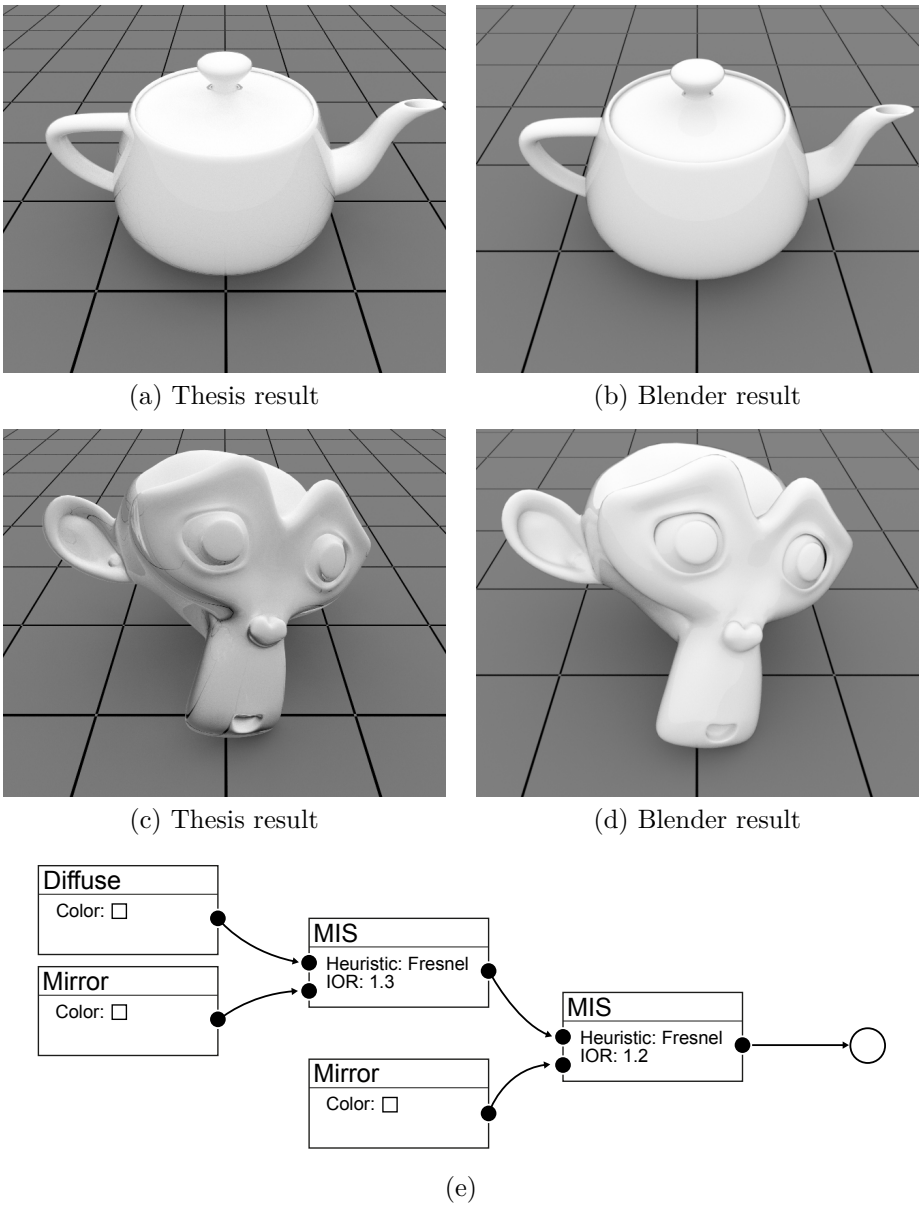
(e)

Figure 6.1: The plastic material results of the thesis path tracer are shown in (a) and (c) and the *Blender* renderings are displayed in (b) and (d) with 256 samples per pixel and a resolution of 650×540. Also the creation of this material is shown in (e).

The rendered results of the plastic material are shown in Figure 6.1. The results of the thesis path tracer are shown in Figures 6.1 (a) and 6.1 (c), while the images produced by *Blender* are displayed in Figures 6.1 (b) and 6.1 (d).

Ceramic material

As illustrated in Figure 6.2 (e), a coated glossy ceramic surface is created by utilizing two layers of material blending. The first layer combines a diffuse BRDF and a mirror



(a) Thesis result (b) Blender result

(c) Thesis result (d) Blender result

(e)

Figure 6.2: The results of the ceramic material rendered in the thesis project are shown in (a) and (c) and the *Blender* results are be seen in (b) and (d) with a resolution of 650×540 and 256 samples per pixel. Additionally the material creation is shown in (e).

BRDF with a surface color of white for both materials. The blending is performed with a *Fresnel heuristic* and an index of refraction $\eta_0 = 1.3$. The result of the first layer is blended again with a white mirror material. For the second combination layer the *Fresnel heuristic* is used once again with a slightly different value $\eta_1 = 1.2$ for the index of refraction.

In *Blender* the graph shown in Figure 6.2 (e) is recreated with the same values whenever possible. The mirror material which is used in the thesis renderer for example is

mimicked in *Blender* by utilizing a glossy material with the highest glossiness value 1 and by using the so called *sharp distribution* type which is developed to simulate mirror like reflections [31]. The ceramic material results of the render comparisons between the thesis renderer and *Blender* are shown in Figure 6.2. The thesis renderer results are shown in Figures 6.2 (a) and 6.2 (c) while the *Blender* results are displayed in Figures 6.2 (b) and 6.2 (d).

Iridescent material

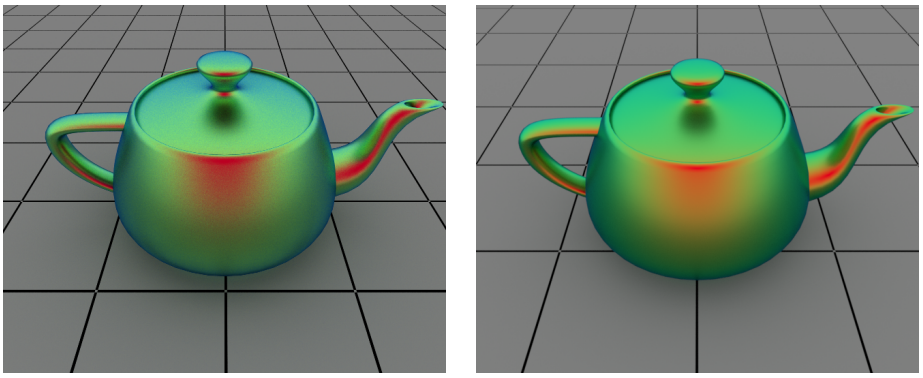
Iridescent surfaces can be often observed on soap bubbles, metal surfaces or on glimmering side of a CD. This effect can be recreated via material blending and view dependent weighting heuristics as it is shown in Figure 6.3 (e). In order to create an iridescent surface with the thesis system, a two layer combination network, as shown in Figure 6.3 (e), is created. The first layer contains two glossy materials with a red and green surface color. The combination is controlled via the *facing ratio heuristic* with $\beta_0 = 8.0$. The result of the first combination layer is blended with another glossy material and a *facing ratio heuristic* with $\beta = 0.7$. The glossiness value for all materials is set to a 40.

The version of this material, which is created in *Blender*, operates on the same color values. However, since the glossiness in Blender operates between a range of 0 and 1, this value is adjusted manually in order to achieve a similar visual output like in the thesis renderer. Also *Blender* does not offer a component which is equal to the *facing ratio heuristic* used in the thesis renderer. The *Blender* node system only provides a *facing node* with a blend value which was used to compensate the *facing ratio heuristic* of the thesis renderer. The rendered result of the iridescent material is shown in Figure 6.3. The thesis results are displayed in Figures 6.3 (a) and 6.3 (c), additionally Figures 6.3 (b) and 6.3 (d) illustrate the results of *Blender*.

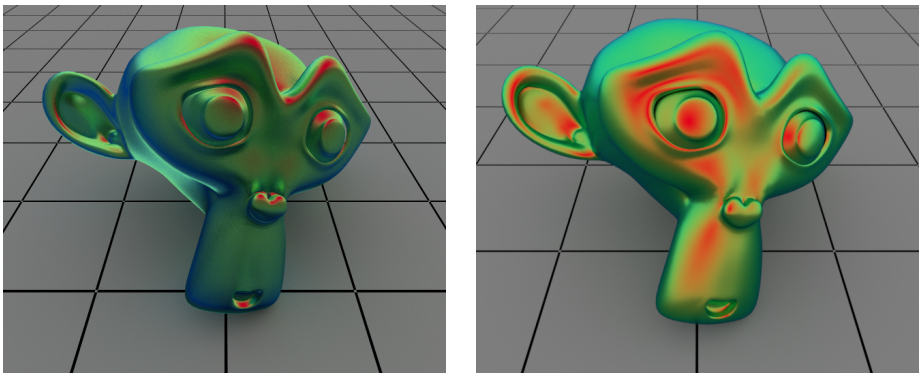
Toon material

Toon shading or cell shading is categorized as a non photo realistic rendering technique and is often associated by rendering objects with outlines or a small amount of shading levels which results in a cartoon like appearance [3].

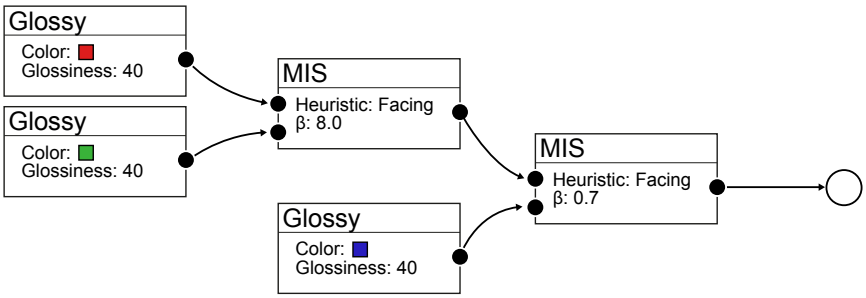
The thesis renderer can achieve this material by thresholding view dependent heuristics as shown in Figure 6.4 (e). The created toon material consists of two blending layers as seen in Figure 6.4 (e) and plain diffuse materials. The first layer combines the black outline material with the diffuse red main color by utilizing a thresholded *Fresnel heuristic* with $\eta = 1.1$. The threshold $\tau_0 = 0.8$ specifies that the black surface material acquires a 100% probability if the underlying *Fresnel heuristic* returns a value above τ. Otherwise the red main material is used. The next layer simply uses slightly different values with $\tau = 0.9$ combined with a *facing ratio heuristic* with $\beta = 0.45$. The second layer combines the result of the first layer and a light orange diffuse material. In *Blender* the toon material is created in a similar way illustrated in Figure 6.4 (e). However, since no *threshold heuristic* like in the thesis renderer is provided in *Blender* a similar behaviour is created with conditional statements. The results are shown in Figure 6.4. The thesis renderer results are shown in Figures 6.4 (a) and 6.4 (c), the outcomes of *Blender* are displayed in Figures 6.4 (b) and 6.4 (d).



(a) Thesis result (b) Blender result



(c) Thesis result (d) Blender result



(e)

Figure 6.3: The results of the iridescent material, rendered in the thesis project, are shown in (a) and (c), additionally the *Blender* outcomes are displayed in (b) and (d) with 256 samples per pixel and a image resolution of 650×540. Furthermore, the creation of this material is shown in (e).

6.2.2 Noise comparison

The goal of this evaluation method is to compare BRDFs which are created via *multiple importance sampling* with simple BRDFs which are directly implemented in the thesis path tracer. The comparison is performed with results of the thesis renderer in different scenarios where the original target material m_0 is always a diffuse material with a surface color of $c_d = (0.8, 0.8, 0.8)$. In two different cases, this simple material is recreated via

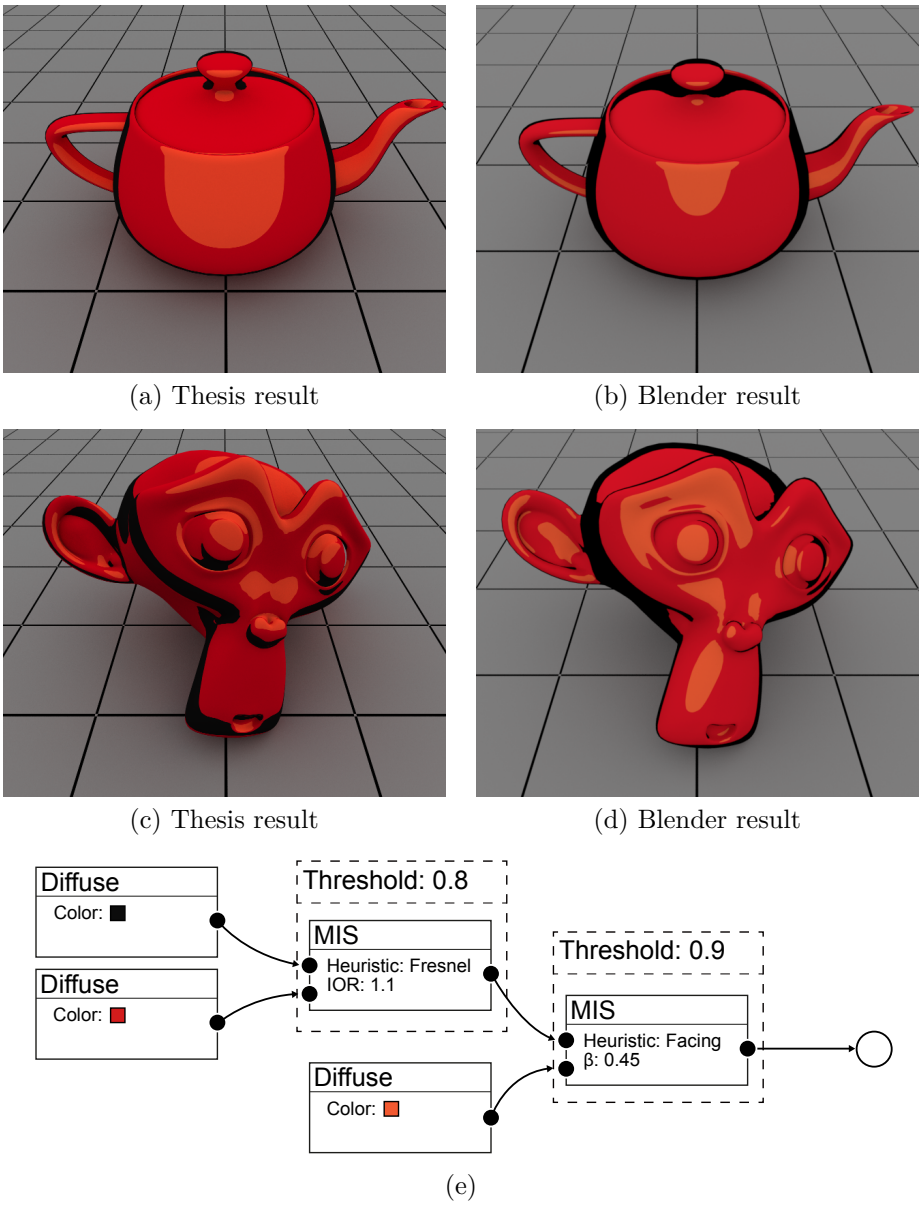


Figure 6.4: The toon material results of the thesis renderer are shown in (a) and (c), furthermore the *Blender* results are displayed in (b) and (d) with a resolution of 650×540 and 256 samples per pixel. In addition, the creation of this material is shown in (e).

MIS for BRDFs with two different colored materials and also with two equally colored materials. The focus of this comparison is to investigate if MIS for BRDFs introduces noise in the final rendering with different constellations. In order to measure the noise differences, visual comparison results are provided. Furthermore, the *mean-squared-error* (MSE) is utilized to provide a numeric metric for measuring the noise of rendered results in comparison to a high quality reference image, which is produced with a samples per pixel count of 1024. As described for 1D signals in [24], the *mean-squared-error* for

images can be defined as

$$MSE = \frac{1}{m \cdot n} \sum_{i=1}^{m} \sum_{j=1}^{n} (I(i,j) - I'(i,j))^2, \tag{6.1}$$

where m and n specify the dimensions of the original image I and the approximated image I'.

Blending different colors

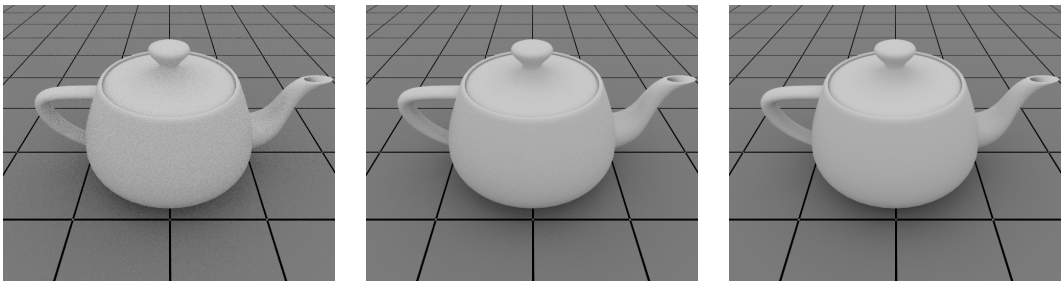
This comparison aims to recreate the target diffuse material m_0 with the specified color $c_d = (0.8, 0.8, 0.8)$ by blending two other materials m_1 and m_2 via MIS and a *constant heuristic* (see Section 5.3.3) with the weighting $w_0 = 0.8$ for the first material m_1. The material m_1 has a white color with $c_1 = (1, 1, 1)$ while material m_2 is defined with a black surface color $c_2 = (0, 0, 0)$. Via the *constant heuristic* and the weighting $w_0 = 0.8$, the average outcome should be the same like the destination material m_0. The results of this comparison are shown in Figure 6.5 with different samples per pixel configurations. Also the outcomes of the target material m_0 without *multiple importance sampling* and the recreations of m_0 are displayed in Figure 6.5. Furthermore, Figure 6.6 shows the *mean-squared-error* on the y-axis for the examples with and without *multiple importance sampling* shown in Figure 6.5. On the contrary the x-axis shows the different sampling configurations. The values of Figure 6.6 are created by using a high quality version of the teapot rendering shown in Figure 6.5 as a reference image I which was rendered with a sample per pixel count of 1024.

Blending same colors

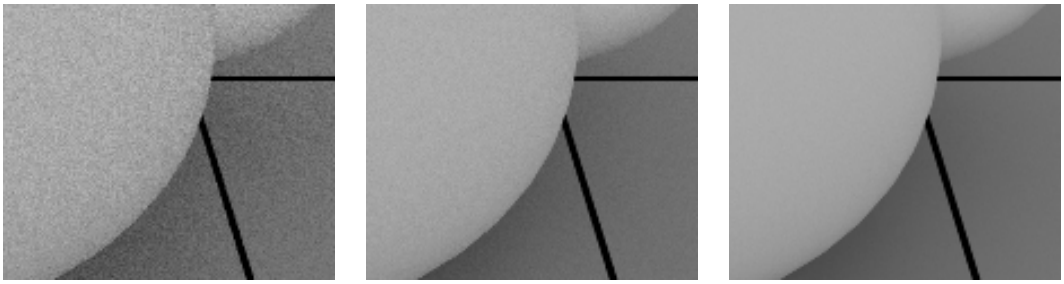
Similar to Section 6.2.2 this comparison aims to mimic a target diffuse material m_0 with the color $c_d = (0.8, 0.8, 0.8)$ by combining two materials m_1 and m_2. However, this comparison differs by using the same color c_d for both materials m_1 and m_2 and combine them with a *constant heuristic* with $w_0 = 0.5$. This usage of the same color demonstrates if the blending approach via MIS introduces noise in general or if the produced noise is dependant on the specific material properties like the color. The results of this comparison are shown in Figure 6.8 with different sampling configurations. Furthermore, the rendering outcomes of the target material m_0 and the combined materials are shown. Additionally the *mean-squared-error* is shown on the y-axis of Figure 6.7 for the shown examples in Figure 6.8 with and without MIS. Furthermore, the different sampling configurations are shown on the x-axis.

6.2.3 Render time comparison

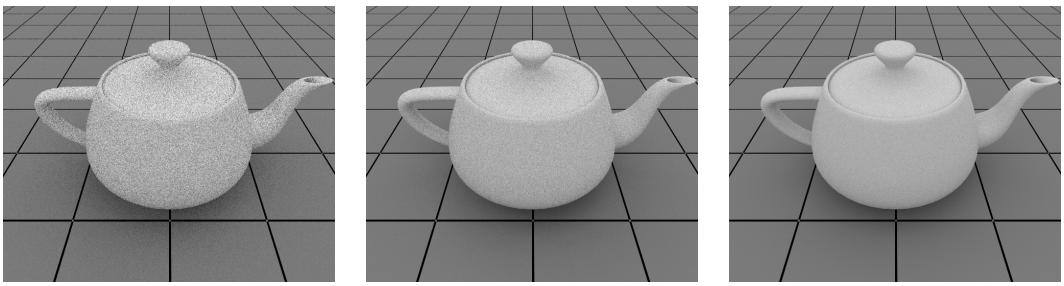
The render time comparison is performed by path tracing a simple scene which contains a plane and a sphere as geometric objects as showed in Figure 6.10. The sky has a constant white color, also *stratified sampling* is enabled for each rendered image. The time comparison is performed with different sampling configuration which are 16, 64, 256, 400, 600 and 1024 samples per pixel with and without enabled *multiple importance sampling* for BRDFs. The visual rendering results are shown in Figure 6.10. A detailed



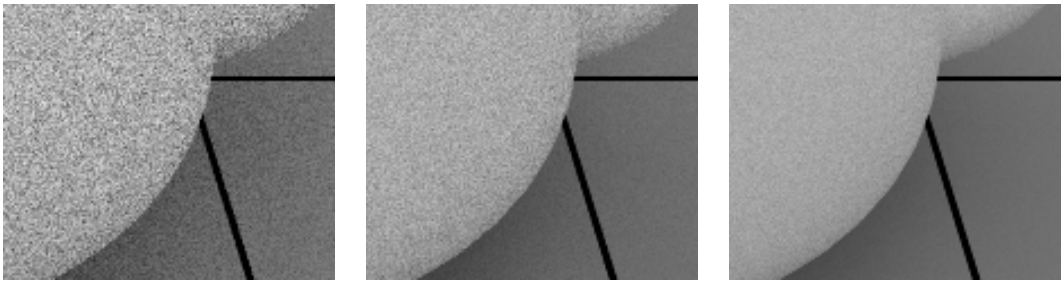
(a) No MIS, 16 SPP (b) No MIS, 64 SPP (c) No MIS, 256 SPP



(d) No MIS, 16 SPP (e) No MIS, 64 SPP (f) No MIS, 256 SPP



(g) MIS, 16 SPP (h) MIS, 64 SPP (i) MIS, 256 SPP



(j) MIS, 16 SPP (k) MIS, 64 SPP (l) MIS, 256 SPP

Figure 6.5: The examples in this figure illustrate example renderings of the thesis path tracer with a simple diffuse material m_0 (a–f) and a material m_c (g–l), which is created by combining two BRDFs via MIS, in order to recreate m_0. Material m_c uses different colored BRDFs to achieve the mimicked output. The render results are provided with different samples per pixel (SPP) configurations and a resolution of 650×540.

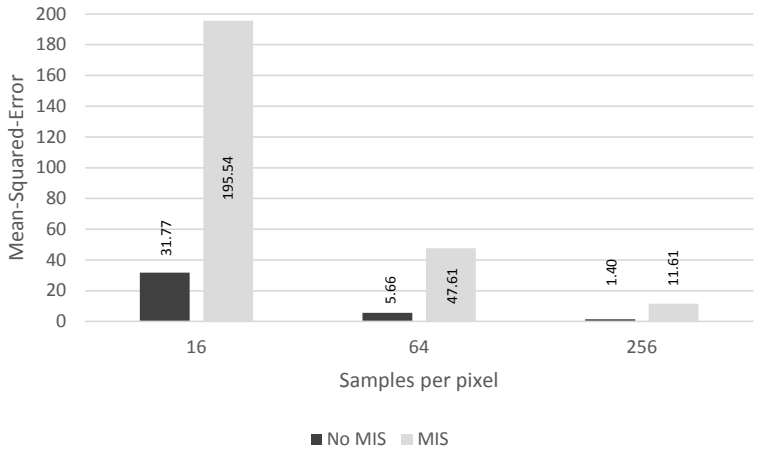


Figure 6.6: On the y-axis this diagram shows the *mean-squared-error* of the results shown in Figure 6.5 for different colors in comparison to a high quality reference image with a sampling count of 1024 samples per pixel. The different sampling configurations are illustrated on the x-axis.

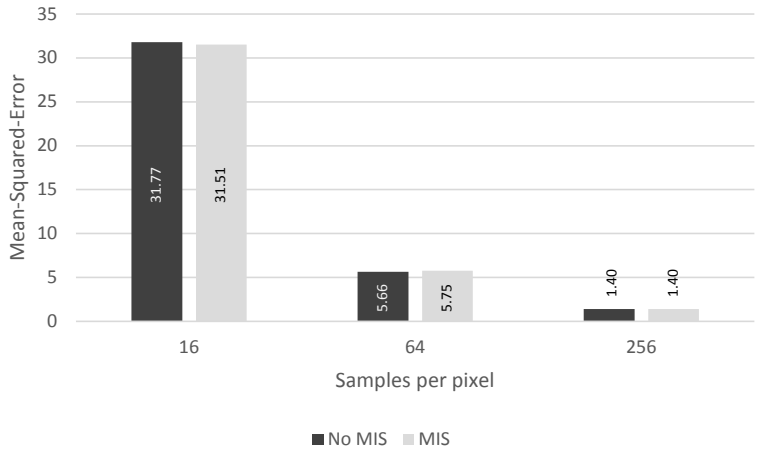
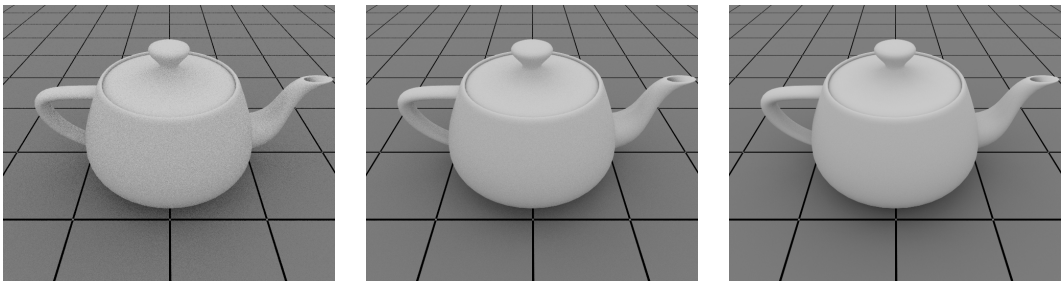
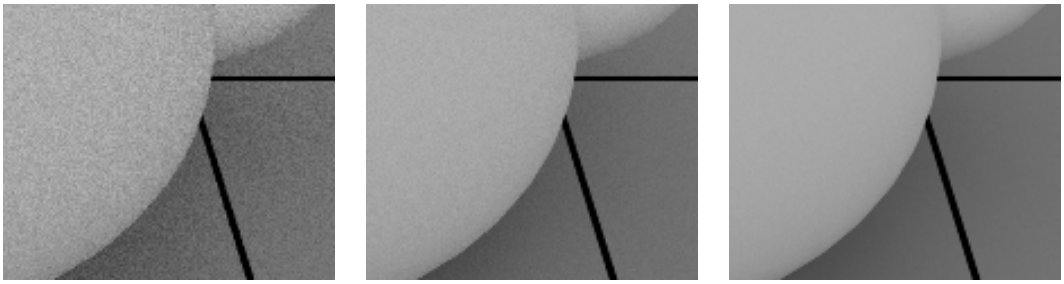


Figure 6.7: This diagram shows the *mean-squared-error* on the y-axis of the results shown in Figure 6.8 for the same colors in comparison to a high quality reference image with a sampling count of 1024 samples per pixel. The x-axis displays the different sampling configurations.

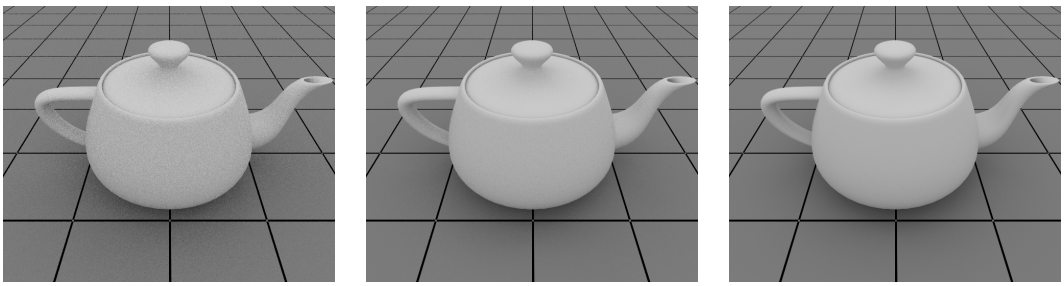
render time based comparison is shown in Figure 6.9, which displays the computation times of different sampling configurations with and without *multiple importance sampling* for BRDFs. The y-axis of Figure 6.9 displays the rendering time in milliseconds, whereas the samples per pixel is illustrated on the x-axis.



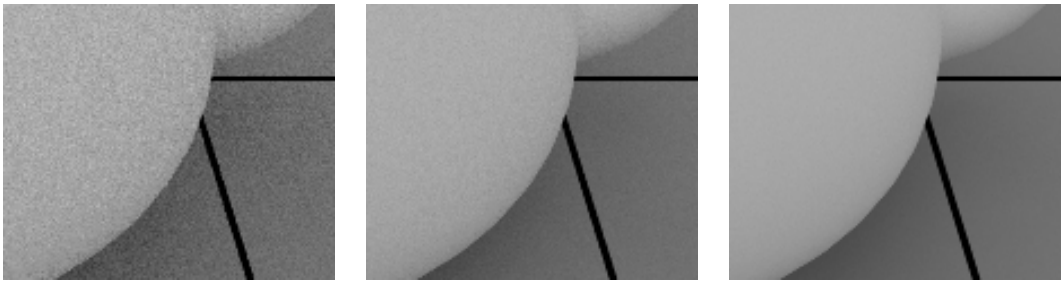
(a) No MIS, 16 SPP (b) No MIS, 64 SPP (c) No MIS, 256 SPP



(d) No MIS, 16 SPP (e) No MIS, 64 SPP (f) No MIS, 256 SPP



(g) MIS, 16 SPP (h) MIS, 64 SPP (i) MIS, 256 SPP



(j) MIS, 16 SPP (k) MIS, 64 SPP (l) MIS, 256 SPP

Figure 6.8: In different example renderings, a simple diffuse material m_0 without MIS (a–f) and a material m_c with MIS (g–l) is shown. Material m_c is produced by the combination of two BRDFs to recreate m_0. The recreated material m_c is built by using BRDFs with the same color to achieve the mimicked output. The rendered results have a resolution of 650×540 and varying samples per pixel configurations.

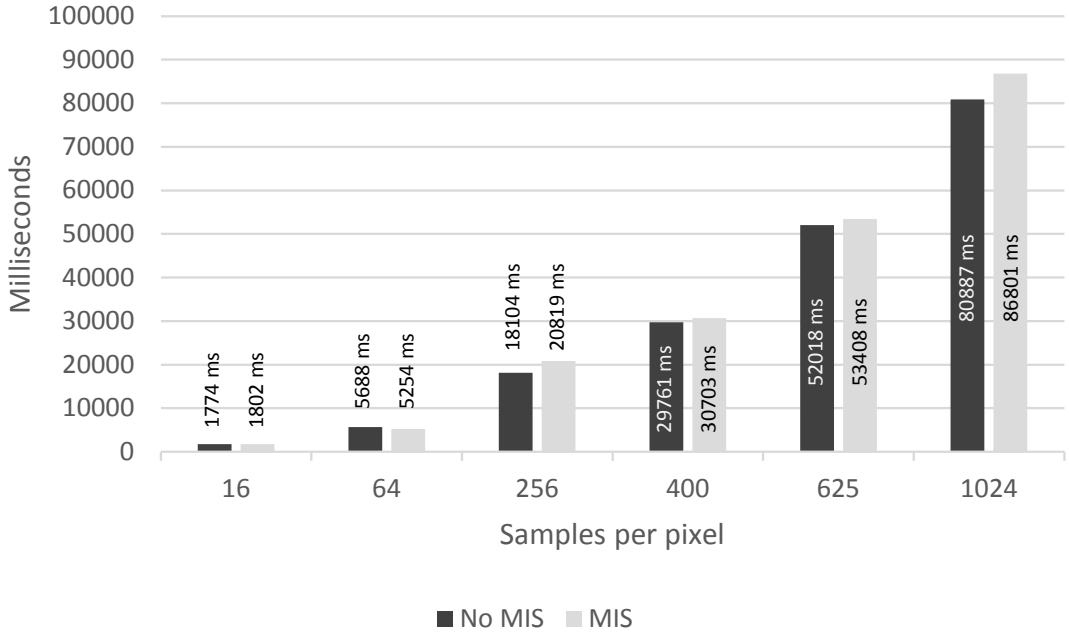
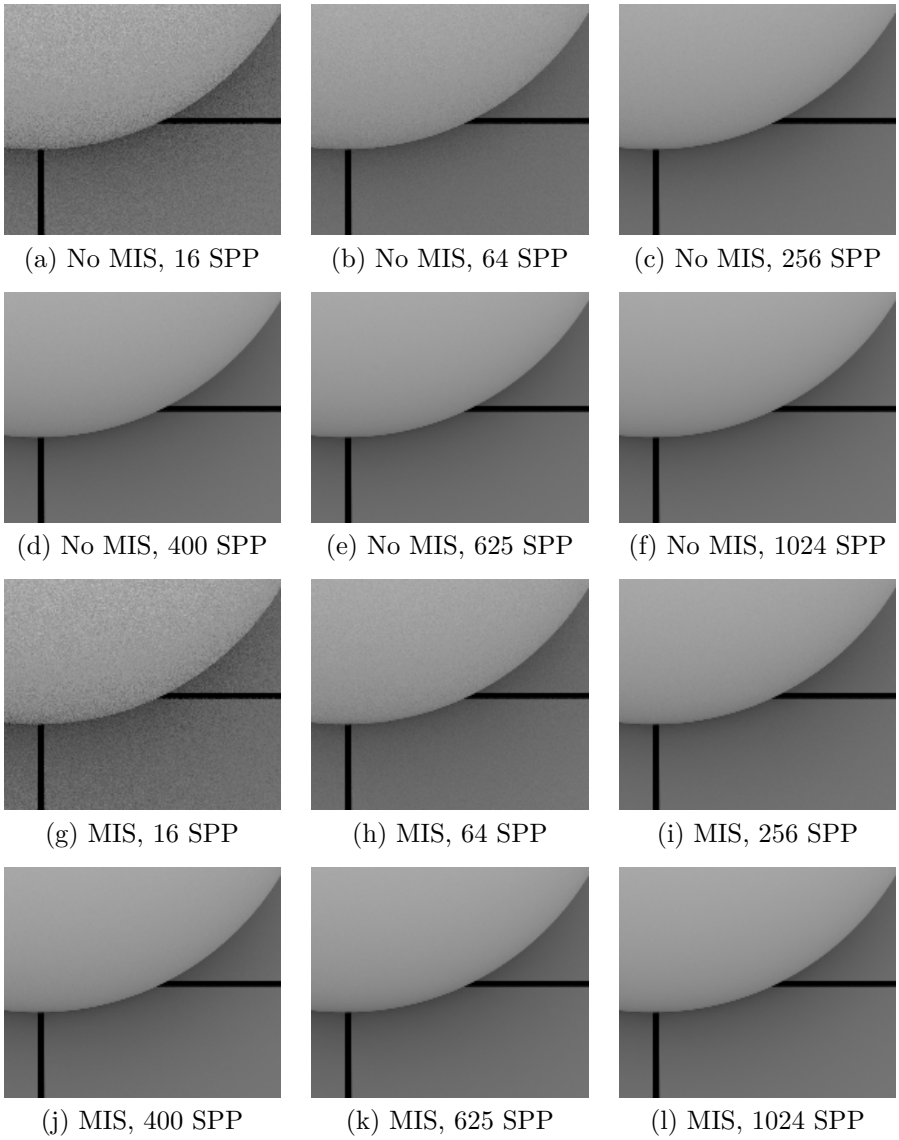


Figure 6.9: This Figure shows the computation times with and without MIS of the calculated images from Figure 6.10 in a bar chart. Each chart category on the x-axis displays the respective sampling configuration. Furthermore, the y-axis represents the render time in milliseconds.



(a) No MIS, 16 SPP (b) No MIS, 64 SPP (c) No MIS, 256 SPP

(d) No MIS, 400 SPP (e) No MIS, 625 SPP (f) No MIS, 1024 SPP

(g) MIS, 16 SPP (h) MIS, 64 SPP (i) MIS, 256 SPP

(j) MIS, 400 SPP (k) MIS, 625 SPP (l) MIS, 1024 SPP

Figure 6.10: Multiple rendering results of the thesis path tracer with varying sampling configurations (SPP) and a resolution of 650×540 are shown in this figure. Furthermore, the images display the outcome with (g–l) and without (a–f) *multiple importance sampling* for BRDFs.

Chapter 7

Discussion and Limitations

The BRDF blending approach via the *multiple importance sampling* technique is evaluated in Chapter 6. The results of each evaluation method are discussed and interpreted in this chapter. Furthermore, subsequent limitations of the BRDF blending method are described.

7.1 Blender result comparison

This section discusses the evaluation results provided in Section 6.2.1, which presents a rendering comparison between the thesis BRDF blending method and the BRDF combination strategy of *Blender*.

It must be noted that rendering parametrization between *Blender* and the thesis path tracer is tried to be as equal as possible. However, due to the fact that *Blender* is a very feature-rich application regarding 3D rendering, not all implemented algorithms of the master thesis offer the same versatility of *Blender*. As a consequence visual differences can be caused due to the fact that several capabilities of the thesis BRDF blending system are emulated or approximated in *Blender*. However, since the comparison with *Blender* shows the general practicality of the thesis BRDF blending approach, an exact outcome of both rendering systems is not necessary.

7.1.1 Plastic

As stated in Section 6.2.1, the red plastic material shown in Figure 6.1 is created by blending a red diffuse material m_0 with a glossy white material m_1. The blending is performed by a *Fresnel heuristic* with an index of refraction $\eta = 2$.

As shown in Figures 6.1 (a) and 6.1 (c) the thesis results occurrence of the white glossy material can be mostly observed on surface areas where the angle between the view direction ω_o and the surface normal N is high. Therefore, reflections are more visible on outer edges of the given objects. Due to the *Fresnel heuristic* the red diffuse material m_0 has a higher sampling probability in the opposite case. However, due to a rather high index of refraction value $\eta = 2$, material m_1 also receives a smaller probability on areas the angle between N and ω_0 is small which leads to a general subtle glossy effect for the whole object which can be seen in Figure 6.1 (a).

The results displayed in Figures 6.1 (b) and 6.1 (d) which are computed by *Blender* show an overall higher amount of the white glossy material m_1. This different behaviour is caused by an different implementation of the *Fresnel* component of *Blender* compared to the thesis approach. As stated in [30], *Blender* utilizes the exact *Fresnel* equation, while in the thesis renderer *Schlick's* approximation of the *Fresnel* equation is implemented as described in Section 5.3.3. However, apart from the different *Fresnel* implementation a similar material blending result can be achieved with the thesis BRDF combination system.

7.1.2 Ceramic

The evaluation results of Section 6.2.1 provide coated ceramic like material which is composed by a two layer BRDF blending approach shown in Figure 6.2 (e). A white diffuse material m_0 is used as the main visual contributor which is blended with a mirror material m_1 with a *Fresnel heuristic* with $\eta_0 = 1.3$. The result of the combination is again blended with another mirror material m_2, also with a *Fresnel heuristic* with $\eta_1 = 1.2$.

The result of the thesis renderer shown in Figure 6.2 (a) is visually very similar to the *Blender* result in Figure 6.2 (b) for the teapot model. Differences can be seen with the second object showed in Figures 6.2 (c) and 6.2 (d). The chimpanzee head, which is a standard *Blender* model, shows a much brighter appearance in the *Blender* results (see Figure 6.2 (d)). Also the reflections at the object borders are more direct and visible in the thesis result shown in Figure 6.2 (c). Since the *Fresnel heuristic* is used in the thesis renderer and the standard *Fresnel* node is utilized in Blender, those visual deviations can again be derived from the different *Fresnel* implementations of both applications, as already stated in Section 7.1.1.

7.1.3 Iridescent

The material blending results for iridescent surface are provided in Section 6.2.1, which shows an iridescent glossy material produced by utilizing a red material m_0, a green material m_1 and a blue material m_2 which are all defined as glossy material with the same glossiness.

As shown in Figure 6.3 (e) the iridescent material is created in two layers and by using the *facing ratio heuristic* with a *facing power* $\beta_0 = 8.0$ for the first layer and $\beta_1 = 0.7$ for the second layer.

In *Blender* a similar facing node is utilized in order to mimic the behaviour of the thesis *facing ratio heuristic*. As a consequence, the results of the thesis renderer and *Blender* appear different as illustrated in Figure 6.3. However, despite small visual differences the overall effect can be easily created with the proposed *multiple importance sampling* technique for BRDFs.

7.1.4 Toon

Section 6.2.1 provides the results for non photo realistic stylized rendering output, which is referred as *toon material* in this thesis. The *toon material* is created by utilizing a black diffuse material m_0 for outline highlighting, a red diffuse material m_1 for the main color

and a light orange diffuse material m_2 for shading highlights. The hard edged shading style is created by thresholding the view dependent heuristics shown in Figure 6.4 (e). As it can be observed in Figure 6.4 (a), it is possible to achieve a similar appearance with the BRDF blending system of the thesis renderer as the *Blender* result shown in Figure 6.4 (b). Also the second example shown in Figures 6.4 (c) and 6.4 (d) illustrates that the same combined *toon material* achieves consistent effects on a different model. The differences to the *Blender* results are attributable to differently or non existent heuristic equivalents in *Blender*, such as the *Fresnel heuristic*, the facing *ratio heuristic* and the *threshold heuristic*.

7.2 Noise comparison

This section discusses the evaluation results shown in Section 6.2.2. The goal of the noise comparison is to recreate a given target material m_0 with two other materials m_1 and m_2 by using *multiple importance sampling* for BRDFs. The aim is to determine if the MIS blending technique for BRDFs causes visible noise in the finished endering output. As stated in Section 6.2.2 the target material m_0 is defined with the color $c_d = (0.8, 0.8, 0.8)$ as a diffuse BRDF.

7.2.1 Different colors

As described in Section 6.2.2 the target material m_0 is recreated by utilizing two diffuse materials with the color $c_1 = (1, 1, 1)$ and $c_2 = (0, 0, 0)$. Furthermore, a *constant heuristic* with a weighting $w_0 = 0.8$ is used to obtain the target color c_d.

The results with different sampling configurations are displayed in Figures 6.5 with enabled and disabled *multiple importance sampling* for BRDFs. It can be seen that the noise level in the cropped Figures 6.5 (d), 6.5 (e) and 6.5 (f) without *multiple importance sampling* is much lower in comparison to the recreated material shown in Figures 6.5 (j), 6.5 (k) and 6.5 (l). Also with a high sampling count the MIS outcome with different colors results in clearly visible noise as seen in Figure 6.5 (l).

Despite the visible noise in this example, the rendering result with *multiple importance sampling* yields in average the target color c_d. Naturally the recreation is more exact if more samples are used for the calculation. However, as a result of this trial it becomes apparent that the chosen colors c_1 (white) and c_2 (black) are causing the noisy image. Since the *Monte Carlo estimator* (see Equation 3.2) can be superficially interpreted by averaging the sum of samples of a function $f(x)$, this approach performs better if the overall variance $V[f(x)]$ of all samples is small. Since the colors c_1 and c_2 are chosen randomly with a probability of 80% for c_1, the high contrast between these colors induces higher variance and therefore noise in the image as shown in Figures 6.5 (j), 6.5 (k) and 6.5 (l). Despite the clearly visible noise in this trial, this does not infer that the *multiple importance sampling* approach itself produces noisy results. As seen in Figure 6.5 the blending setup itself causes the noise in the end result, which is in this case produced by high contrast colors. Therefore, it becomes apparent that the material composing process itself can be responsible for results with visible noise. Section 7.2.2 discusses the important circumstance if the *multiple importance sampling* approach itself causes noise artifacts. Even though the noise difference between the MIS

renderings and the regular BRDFs shown in Figure 6.5 are clearly visually perceptible, the diagram in Figure 6.6 clarifies the comparison differences even more with a *mean-squared-error* metric. The higher the *mean-squared-error*, the more noise is contained in the rendered image in comparison to a high quality rendering which is computed with 1024 samples per pixel. For different colors the diagram in Figure 6.6 clearly shows that this *multiple importance sampling* blending example results in a notably higher *mean-squared-error* than the regular BRDF. This insight continues for each sampling configuration shown in Figure 6.6.

7.2.2 Same colors

As already explained in Section 6.2.2, a target diffuse material m_0 with the defined color $c_d = (0.8, 0.8, 0.8)$ is recreated by combining two other diffuse materials m_1 and m_2 also with the same color c_d. A *constant heuristic* with $w_0 = 0.5$ is utilized for this example. Since only one color c_d is used, an exact recreation of m_0 should be guaranteed. However, the goal of this example is to determine if the combination approach via *multiple importance sampling* itself produces any noise during the rendering process.

The visual results are shown in Figure 6.8 for different sampling configurations and display a comparison between BRDF blending via MIS and the regular calculated BRDF. Visually the cropped Figures 6.8 (j), 6.8 (k) and 6.8 (l) with *multiple importance sampling* show no noticeable differences to the Figures 6.8 (d), 6.8 (e) and 6.8 (f) which are rendered without MIS. Also the diagram in Figure 6.7 illustrates that the *mean-squared-error* for every sampling configurations is nearly the same for *multiple importance sampling* and the regular BRDF. The equality of the results for this examples shows that BRDF blending with *multiple importance sampling* does not introduce additional noise into the final rendered output.

7.3 Render time comparison

In Section 6.2.3 the render time for BRDF blending via MIS is measured in comparison to a material which is computed without MIS. For *multiple importance sampling* a simple *constant heuristic* is used. The example scene and the visual results of the render time comparison can be seen in Figure 6.10.

The measured render time results can be seen in Figure 6.9. It is apparent that for each sampling configuration shown in Figure 6.9 the MIS approach requires slightly more time in order to produce the final image. For 1024 samples per pixel the MIS approach is approximately 6 seconds slower compared to the regular material without *multiple importance sampling*.

However, a minimal computation overhead was expected since additional steps have to be made in order to choose a material with the *multiple importance sampling* approach. Additionally, more complex heuristics can also increase the render time noticeable, depending on the decision routine within the heuristic.

7.4 Limitations

This section explains limitations of the BRDF combination approach via *multiple importance sampling*, which are referring to other material combination techniques mentioned in Chapter 2 and general encountered limits during the evaluation process.

7.4.1 No real material layering

In Section 2.4 a BRDF layering concept based on the research in [25] by *Weidlich* is described. BRDFs are combined by layering them on top of each other in order to achieve combined results.

The approach described in this thesis does not support the layering of BRDFs, a given surface point p is associated to only one BRDF for a given point in time. However, a range of layering effects can be simulated by using the decision heuristics explained in Section 5.3.3. Recreations of layered BRDFs can be seen in Section 6.2.1 which shows plastic, ceramic and iridescent material types with a simulated glossy layer on top of the surface.

7.4.2 Noise output

As described in Section 7.2, the combination approach via *multiple importance sampling* itself does not introduce additional noise into the final image. However, as discussed in Section 7.2.1, unfavorable chosen material parameters of the combine surface representation may lead to noise due to the strong difference of the given parameters. The example in Section 6.2.2 illustrates this by blending two materials with high contrast colors.

Chapter 8

Conclusion and Future Work

This chapter provides a final conclusion of the achieved BRDF blending in *Path Tracing* with *multiple importance sampling*. Furthermore, an outlook of potential topics for future work is given.

8.1 Conclusion

This thesis introduced a technique to utilize *multiple importance sampling* (MIS) in order to combine an arbitrary amount of given BRDFs into a single complex one. Since *multiple importance sapmling* is usually used in another context (see Section 2.1), MIS is adapted in Chapter 5 for the purpose of BRDF blending. Furthermore, several heuristics were introduced in Section 5.3.3 to calculate weightings during the combination process of *multiple importance sampled* BRDFs. Additionally to the MIS blending system, a fully functional path tracer was developed in order to test the BRDF combination system and to demonstrate the easy incorporable blending system into an existing render system. As discussed in Chapter 7, the blending system is able to recreate results of the well established *Cycles* path tracer of *Blender* and offers therefore a valid solution to combine multiple BRDFs. Furthermore, *multiple importance sampling* for BRDF combination does not induce additional noise to the final rendering. Also only a minor loss of performance was measured as explained in Section 7.3. Altogether BRDF blending with *multiple importance sampling* offers a fast an easy incorporable solution to create complex surface representations based on arbitrary BRDFs. For a variety of surface representations the combination system eliminates the need to implement complex BRDFs directly, they can simply be put together based on multiple simple materials.

8.2 Future work

This section provides insights into potential topics for future research for BRDF blending via *multiple importance sampling*.

8.2.1 Texture based heuristic

The introduced heuristics in Section 5.3.3 compute weightings based on surface parameters like the normal N or the view ray w_o. Another possible approach is to store the weightings in a texture which enables to create 3D model specific weight maps. The actual weighting value can be retrieved via the UV-coordinates of the hit point p. Furthermore, noise algorithms (perlin noise, worley noise, etc.) can be utilized to generate weight maps based on noise in order to achieve interesting visual results.

8.2.2 Visual material editor

Although the combination approach via *multiple importance sampling* provides reasonable visual results as seen in Section 6.2.1, it can be inconvenient to build complex materials through code, especially if the desired surface representation consists of several combination layers. Due to this a visual editor, similar to the node editor of *Blender*, would be a convenient solution to create and store materials for later usage.

Appendix A

DVD Contents

Format: DVD+RW, Single Layer, 4.7 GB

A.1 PDF-Files

Path: /

Legmaier2018.pdf . . .	Master thesis

A.2 Project-Files

Path: /Project

ThesisPathTracer.zip . .	Full *Java* source code as *Eclipse* project

A.3 Example Renderings

Path: /Renderings

Renderings.zip	A collection of example renderings computed with the thesis path tracer

References

Literature

[1] Thomas Bashford-Rogers, Kurt Debattista, and Alan Chalmers. "Importance Driven Environment Map Sampling". *IEEE Transactions on Visualization and Computer Graphics* (2014), pp. 907–918 (cit. on p. 17).

[2] James F. Blinn. "Light Reflection Functions for Simulation of Clouds and Dusty Surfaces". In: *Proceedings of the 9th Annual Conference on Computer Graphics and Interactive Techniques – SIGGRAPH '82*. 1982, pp. 21–29 (cit. on p. 12).

[3] Philippe Decaudin. *Cartoon Looking Rendering of 3D Scenes*. Research Report. INRIA, 1996 (cit. on p. 53).

[4] Cyril Delalandre et al. "Single Scattering in Heterogenous Participating Media". In: *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques – SIGGRAPH '10*. 2010, 14:1–14:1 (cit. on p. 7).

[5] Tom Duff et al. "Building an Orthonormal Basis, Revisited". *Journal of Computer Graphics Techniques (JCGT)* (2017), pp. 1–8 (cit. on p. 31).

[6] Philip Dutré. "Global Illumination Compendium". *Computer Graphics, Department of Computer Science, Katholieke Universiteit Leuven* (2003) (cit. on p. 17).

[7] Bernardt Duvenhage, Kadi Bouatouch, and Derrick Kourie. "Numerical Verification of Bidirectional Reflectance Distribution Functions for Physical Plausibility". In: *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference – SAICSIT '13*. 2013, pp. 200–208 (cit. on pp. 1, 21).

[8] Wenzel Jakob et al. "A Comprehensive Framework for Rendering Layered Materials". *ACM Transactions on Graphics* (2014), 118:1–118:14 (cit. on pp. 9, 10).

[9] James T. Kajiya. "The Rendering Equation". In: *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques – SIGGRAPH '86*. 1986, pp. 143–150 (cit. on pp. 14, 15).

[10] Csaba Kelemen and Laszlo Szirmay-Kalos. "A Microfacet Based Coupled Specular-Matte BRDF Model with Importance Sampling". In: *Eurographics Short Presentations*. 2001, pp. 25–34 (cit. on p. 10).

[11] Christopher Kulla and Marcos Fajardo. "Importance Sampling Techniques for Path Tracing in Participating Media". *Computer Graphics Forum* (2012), pp. 1519–1528 (cit. on pp. 6–8).

[12] Don P. Mitchell. “Consequences of Stratified Sampling in Graphics”. In: *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques – SIGGRAPH ’96*. 1996, pp. 277–280 (cit. on p. 18).

[13] Michael Oren and Shree K. Nayar. “Generalization of Lambert’s Reflectance Model”. In: *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques – SIGGRAPH ’94*. 1994, pp. 239–246 (cit. on p. 12).

[14] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically Based Rendering: From Theory to Implementation (3rd ed.)* Morgan Kaufmann Publishers Inc., 2016 (cit. on pp. 8, 14, 34, 35).

[15] Bui Tuong Phong. “Illumination for Computer Generated Pictures”. *Communications of the ACM* (1975), pp. 311–317 (cit. on p. 12).

[16] Peter Shirley, Changyaw Wang, and Kurt Zimmerman. “Monte Carlo Techniques for Direct Lighting Calculations”. *ACM Transactions on Graphics* (1996), pp. 1–36 (cit. on p. 6).

[17] Joshua Steinhurst and Anselmo Lastra. “Global Importance Sampling of Glossy Surfaces Using the Photon Map”. In: *Proceedings of the 2006 IEEE Symposium on Interactive Ray Tracing*. 2006, pp. 133–138 (cit. on p. 17).

[18] Kevin Suffern. *Ray Tracing from the Ground Up*. A. K. Peters, Ltd., 2007 (cit. on pp. 6, 19, 20).

[19] Bo Sun et al. “Time-varying BRDFs”. In: *Proceedings of the 2nd Eurographics Conference on Natural Phenomena – NPH’06*. 2006, pp. 15–23 (cit. on pp. 12, 13).

[20] Kenneth Torrance and Ephraim Sparrow. “Theory for Off-specular Reflection from Roughened Surfaces”. *Journal of the Optical Society of America* (1967), pp. 32–41 (cit. on p. 12).

[21] Eric Veach. “Robust Monte Carlo Methods for Light Transport Simulation”. PhD thesis. Stanford University, 1998 (cit. on pp. 1, 3–5, 41).

[22] Eric Veach and Leonidas J. Guibas. “Optimally Combining Sampling Techniques for Monte Carlo Rendering”. In: *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques – SIGGRAPH ’95*. 1995, pp. 419–428 (cit. on p. 3).

[23] C. M. Wang et al. “Ellipse Sampling for Monte Carlo Applications”. *Electronics Letters* (2004), pp. 21–22 (cit. on p. 18).

[24] Zhou Wang and Alan Bovik. “Mean squared error: Love it or leave it? A new look at Signal Fidelity Measures”. *IEEE Signal Processing Magazine* (2009), pp. 98–117 (cit. on p. 55).

[25] Andrea Weidlich and Alexander Wilkie. “Arbitrarily Layered Micro-facet Surfaces”. In: *Proceedings of the 5th International Conference on Computer Graphics and Interactive Techniques in Australia and Southeast Asia – GRAPHITE ’07*. 2007, pp. 171–178 (cit. on pp. 9–11, 66).

[26] Turner Whitted. “An Improved Illumination Model for Shaded Display”. *Communications of the ACM* (1980), pp. 343–349 (cit. on p. 14).

[27] Alexander Wilkie et al. “A Reflectance Model for Diffuse Fluorescent Surfaces”. In: *Proceedings of the 4th International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia – GRAPHITE '06*. 2006, pp. 321–331 (cit. on p. 10).

[28] Jarosz Wojciech. “Efficient Monte Carlo Methods for Light Transport in Scattering Media”. PhD thesis. UC San Diego, 2008 (cit. on pp. 16, 17).

Online sources

[29] Blender Team. *Cycles BVH acceleration structure*. Blender Foundation. 2018. URL: https://wiki.blender.org/index.php/Dev:Source/Render/Cycles/BVH (visited on 05/20/2018) (cit. on p. 50).

[30] Blender Team. *Cycles Fresnel Node*. Blender Foundation. 2014. URL: https://developer.blender.org/diffusion/C/browse/master/src/kernel/shaders/node_fresnel.h (visited on 05/26/2018) (cit. on p. 63).

[31] Blender Team. *Cycles Glossy BSDF*. Blender Foundation. 2018. URL: https://docs.blender.org/manual/en/dev/render/cycles/nodes/types/shaders/glossy.html (visited on 05/21/2018) (cit. on p. 53).

[32] Blender Team. *Cycles Integrator Documentation*. Blender Foundation. 2018. URL: https://docs.blender.org/manual/en/dev/render/cycles/settings/scene/render/integrator.html (visited on 04/19/2018) (cit. on p. 19).

[33] Holger Dammertz. *Hammersley Points on the Hemisphere*. Nov. 2012. URL: http://holger.dammertz.org/stuff/notes_HammersleyOnHemisphere.html (visited on 04/05/2018) (cit. on p. 19).

[34] Rory Driscoll. *Better Sampling*. Jan. 2009. URL: http://www.rorydriscoll.com/2009/01/07/better-sampling/ (visited on 04/19/2018) (cit. on p. 18).

[35] Sébastien Lagarde. *Memo on Fresnel Equations*. Apr. 2013. URL: https://seblagarde.wordpress.com/2013/04/29/memo-on-fresnel-equations/ (visited on 05/12/2018) (cit. on p. 47).

[36] LuxRender Team. *Lux Core Render - Relation to Physically Based Rendering: From Theory to Implementation*. LuxRender. 2012. URL: https://luxcorerender.org/aboutus/ (visited on 06/17/2018) (cit. on p. 9).

[37] LuxRender Team. *LuxRender Materials Mix*. LuxRender. 2012. URL: http://www.luxrender.net/wiki/LuxRender_Materials_Mix (visited on 04/01/2018) (cit. on p. 9).

[38] Pixar. *RenderMan Stratified Sampling*. Pixar Animation Studios. 2012. URL: https://renderman.pixar.com/resources/RenderMan_20/stratification.html (visited on 06/17/2018) (cit. on p. 18).

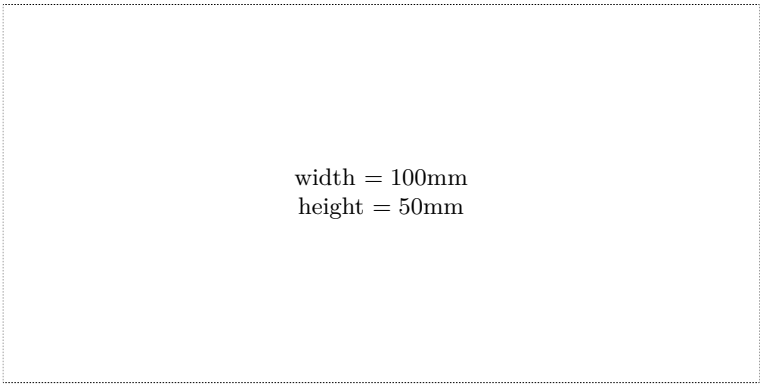
[39] Scratchapixel Team. *A Quick Introduction to Quasi Monte Carlo*. 2015. URL: www.scratchapixel.com/lessons/mathematics-physics-for-computer-graphics/monte-carlo-methods-in-practice/introduction-quasi-monte-carlo (visited on 06/18/2018) (cit. on p. 18).

[40] Scratchapixel Team. *Global Illumination and Path Tracing*. 2015. URL: https://www.scratchapixel.com/lessons/3d-basic-rendering/global-illumination-path-tracing/global-illumination-path-tracing-practical-implementation (visited on 04/12/2018) (cit. on p. 34).

[41] Karl Li Yining. *Multiple Importance Sampling*. Feb. 2015. URL: http://blog.yiningkarlli.com/2015/02/multiple-importance-sampling.html (visited on 03/31/2018) (cit. on p. 5).

Check Final Print Size

— Check final print size! —



— Remove this page after printing! —