Markov Chain Monte Carlo Reflectometry

Submitted in partial fulfillment of the requirements for

the degree of

Master of Science

in

Information Networking

Yuan Meng

B.Eng., Computer Science and Technology, University of Electronic Science and Techonology of China

Carnegie Mellon University
Pittsburgh, PA

April, 2024

Acknowledgements

First, I extend my deepest gratitude to my thesis advisor, Professor Gkioulekas, for his support and guidance throughout the research.

I am also immensely grateful to the reader of this project, Professor Sankaranarayanan, for his constructive feedback and valuable suggestions.

Lastly, I appreciate all the participants in my study, who generously shared their time and experiences, making this research possible.

The project is self-funded without funding support from any agency.

Abstract

This project explores the use of Markov chain Monte Carlo (MCMC) algorithm to directly draw samples from an unknown BRDF. MCMC can produces samples from any function, so long as we can evaluate it. In the case of BRDF acquisition, this means being able to measure the BRDF at any given incident and outgoing directions—this is indeed the main building block of any reflectometry procedure. This will enable our reflectometry procedure to adapt to the unknown BRDF and measure it at random locations distributed according to the BRDF, without the need to first acquire it. The effectiveness of any MCMC sampling algorithm critically depends on the proposal distribution we use. We developed proposals that facilitate reflectometry, by exploring a few research directions: First, we explored proposals that take advantage of physical and empirical properties of real-world BRDFs—reciprocity, isotropy, bivariate symmetry, and so on. Second, we explored proposals that mimic existing physics-based analytical BRDF models—microfacet, PCA, mixture models. Third, we explored controlled MCMC algorithms, which adapt proposals to previously drawn samples as the MCMC iteration proceeds. Fourth, we explored the state-of-the-art technique of estimating density using normalizing flows and how we can use it as proposal and pdf/BRDF of the material.

Table of Contents

Acknowledgements **ii**

Abstract **iii**

List of Figures **vii**

1 Introduction **1**

2 Background **3**

2.1 Data-driven BRDF and its acquisition 3

2.2 MCMC 4

2.3 Microfacet theory and Distribution of Visible Normal(VNDF) 5

2.4 Generative Models, Normalizing Flows and Neural Importance Sampling 6

3 MCMC Proposals **8**

3.1 Spherical Gaussian Proposal 8

3.2 Optimization for reciprocity and isotropic materials 9

3.2.1 Symmetry from Reciprocity 9

3.2.2 Isotropic Material and bilateral symmetry 10

3.3 Simple Primary Sample Space Proposal 10

3.4 VNDF Sampler in Primary Sample Space 11

3.5 Normalizing Flows Sampler in Primary Sample Space 12

3.5.1 Latent Space of Normalizing Flows 12

3.5.2 Normalizing Flow Architecture 12

3.5.3 Train Normalizing Flows Using KL Divergence 14

3.5.4 Space to Importance Sampling in 14

3.5.5 Pretraining Normalizing Flow . . . 15
3.5.6 Adaptive Training During MCMC . . . 16
3.5.7 Controlled MCMC . . . 16
3.6 Validation of convergence . . . 17

4 Post-acquisition Processing **19**
4.1 Interpolation and Fit to Analytic Models . . . 20
4.2 Generating Grids for Renderer . . . 21
4.2.1 Determining Incident Elevation . . . 21
4.2.2 PDF grid generation . . . 22
4.2.3 BRDF grid generation . . . 23
4.3 Interpolation Using Normalizing Flows . . . 24

5 Results **27**
5.1 Simple Primary Sample Space . . . 27
5.1.1 Rough Materials . . . 27
5.1.2 More Specular Materials . . . 28
5.1.3 VNDF proposal . . . 29
5.2 Normalizing Flows Primary Sample Space . . . 29
5.2.1 Comparison of pdf . . . 30
5.2.2 Successful Rendering for Single Channel . . . 30
5.2.3 Failed Rendering for other channels . . . 31
5.2.4 Rough Materials . . . 31

6 Analysis **34**
6.1 Samples Distribution . . . 34
6.1.1 Problem of Perturbing θ . . . 35
6.2 Brighter BRDF in Fitted Interpolation . . . 36
6.2.1 The Most Importance Sampled Area in a 2D (θ, ϕ) Grid . . . 36
6.2.2 Higher BRDF Value in the Most Importance Sampled Part . . . 37
6.3 Noisy Grazing Angles . . . 38
6.4 Bad PDF Causing Information Lost . . . 39

6.4.1 Not able to importance sample three channels at the same time 40
6.5 Normalizing Flows 40
6.5.1 More accuracy in BRDF fitting 40
6.5.2 Instability during MCMC 42
6.5.3 Interpolation Failure in the area where no sample exists 42

7 Discussions **45**
7.1 Conclusion 45
7.2 Future Improvements 45
7.2.1 Better Fitting Models and Algorithm 45
7.2.2 Enforce Physics-based Constraints in Normalizing Flows . . . 46
7.2.3 Tuning Normalizing Flows 46
7.2.4 Separate the Sampling Process of Incident and Outgoing Directions 47
7.2.5 Combination of two methods 47
7.2.6 Better Designed Experiments 47

Bibliography **49**

List of Figures

Figure 3.1 Convergence Test 18

Figure 5.1 Rough purple material comparison 28
Figure 5.2 Rough cc_ibiza material comparison 28
Figure 5.3 White material comparison 29
Figure 5.4 chm orange material comparison 29
Figure 5.5 ilm blue material comparison 30
Figure 5.6 pdf comparison 31
Figure 5.7 Grayscale red channel of chm_orange comparison 31
Figure 5.8 NF failure on green and blue channel 32
Figure 5.9 Rough cc_ibiza material comparison(normalizing flows) 33

Figure 6.1 Samples distribution comparison with different proposals 35
Figure 6.2 A 2D pdf grid for chm_orange 37
Figure 6.3 Comparison of most importance sampled part 38
Figure 6.4 Comparison on importance sampling weight 39
Figure 6.5 Non-uniform BRDF result for ilm_blue 40
Figure 6.6 BRDF fitting result using normalizing flows 41
Figure 6.7 Last 50 cols of BRDF fitting result using normalizing flows . . 43

Symbols

ω_i	the incident vector.
ω_o	the outgoing vector.
ω_h	the half vector.
ω_d	the difference vector.
$\lvert\omega_1 \cdot \omega_2\rvert$	the dot product of vector ω_1 and $vector\omega_2$. If they are both unit vector, this equals the cosine of the angle between ω_1 and ω_2

Abbreviations

NF	Normalizing flows
BRDF	Bidirectional redlectance distribution function
MCMC	Markov chain Monte Carlo

1

Introduction

The bidirectional reflectance distribution function (BRDF) is a compact representation for the way different materials interact with light. It is a key component for the photorealistic modeling of real-world materials, for example by physics-based rendering systems. There has been extensive research in optics, computer science, and computer graphics on modeling and measuring real-world BRDFs. In particular, data-driven BRDFs are directly acquired by measuring how light reflects off of samples of real-world materials. Well-acquired data-driven BRDFs can represent material details and types analytic BRDFs cannot express. Unfortunately, acquiring BRDFs of read-world material, a task we call reflectometry, is challenging: it requires measurements over a 4D domain at sufficiently high resolution. This results in impractically high acquisition times. To alleviate this issue, we can draw inspiration from the Monte Carlo rendering process, where we use only importance-sampled values of the BRDF, instead of the entire BRDF, to form an accurate Monte Carlo estimate of an image. Likewise, during acquisition, we can focus measurements on the part of a real BRDF where its value is high. This will make the acquisition process faster, while still providing sufficient information for rendering. Of course,

such an approach requires having access to an importance sampling process for the unknown BRDF. i.e. we want to importance sampling the cosine weighted BRDF

$$pdf(\omega_i, \omega_o) = \frac{BRDF(\omega_i, \omega_o) * \cos\theta_o}{c} \tag{1.1}$$

where c is the normalization factor:

$$c = \int_\Omega \int_\Omega BRDF(\omega_i, \omega_o) * \cos\theta_o d\omega_i d\omega_o \tag{1.2}$$

We will focus on using Markov chain Monte Carlo(MCMC) to do this importance sampling task. Markov chain Monte Carlo theoretically guarantees that we can importance sampling from any distribution as long as we can evaluate the unnormalized version of that distribution. In our case, this means being able to evaluate the cosine weighted BRDF given an incident and an outgoing angle, which is exactly what we want to do for the reflectometry task. However, the effectiveness of MCMC heavily depends on the design of proposal distributions for producing a new sample given previous sample. After reviewing the background and related work of data-driven BRDF, MCMC and other importance sampling techniques in Section 2, we will describe the proposals we came up based on physical properties of BRDF, physics-based analytic BRDF models, controlled MCMC theory, primary sample space MCMC and normalizing flows in Section 3. We will then describe how we could use the samples we have to render through interpolation in Section 4. In Section 5, we present the results and analyze them.

2

Background

2.1 Data-driven BRDF and its acquisition

Data-driven BRDF is representing BRDF using data acquired from real-world materials. Given the BRDF values of various pairs of incident and outgoing angles, one need to be able to represent it in a rendering software that will evaluate the BRDF from any incident and outgoing angles. Various different kinds of methods are proposed for this job, including fitting the real-life BRDF into analytic models using numerical optimization[1][2], constructing combinations using existing BRDF databases[3][4], adaptively parameterize the BRDF[5],or simply densely sample the material so that the resolution is high and the error is low[6]. Our methods try to combine the adaptive parameterization and numerical optimizations. It is possible to also use existing BRDF database to improve ther result. For acquisition, MERL[6] simply densely samples the BRDF at an resolution of 1°, while the EPFL database[5] uses the knowledge of normal distribution function to do an adaptive acquisition. Our acquisition process is controlled by MCMC, and is a non-deterministic one, which is different from current techniques

2.2 MCMC

Markov chain Monte Carlo is first introduced in graphics in the context of light transport[7]]. The most popular MCMC algorithm is the Metropolis-Hasting algorithm, where a new sampled is generated from the proposal and is either accepted or rejected based on a acceptance probability

$$A(x', x) = \min\left(1, \frac{P(x')}{P(x)} \frac{g(x|x')}{g(x'|x)}\right) \tag{2.1}$$

where P is the unnormalized distribution we want to sample from and $g(x|x')$ is the transition probability from state x to state x'. The transition probability is related to the proposal. If the proposal is symmetric, for example, a Gaussian, then the transition probability cancels out and the acceptance probability is simplified to

$$A(x', x) = \min\left(1, \frac{P(x')}{P(x)}\right) \tag{2.2}$$

Most MCMC proposals are a combination of local exploration and global exploration. The purpose of local exploration is that we assume that if we are at a location that the distribution is high, we might still have a high probability if we only move a little bit from the current location, given the distribution is continuous. However, we will still want a way to jump out of the location area so that we can explore the whole domain. A combination of these two are essential for a good proposal.

A simpler space to do the perturbation was proposed by Kelemen[8]. If we have some arbitrary sampler that takes in random variables in $[0, 1]^n$, and we use this sampler as the proposal of MCMC. We can simply perturb the random variables using n-dimensional Gaussian while still maintaining the complexity of the sampler itself. The transition probability still cancels out in this case. But the Jacobian from unit random variable space to the space we are sampling needs to be applied in the computation of acceptance probability. If we denote x the samples in primary

sample space $[0,1]^N$, and y the samples in the space we want to actually importance sample, we have

$$A(x', x) = \min(1, \frac{P(y')/pdf(y')}{P(y)/pdf(y)} \frac{g(x|x')}{g(x'|x)}) \tag{2.3}$$

$$= \min(1, \frac{P(y')/(pdf(x')|\frac{dx'}{dy'}|)}{P(y)/(pdf(x)|\frac{dx}{dy}|)}) \tag{2.4}$$

$$= \min(1, \frac{P(y')|\frac{dx}{dy}|}{P(y)|\frac{dx'}{dy'}|}) \tag{2.5}$$

Thus, if we can have a good importance sampler, we can easily integrate it into an MCMC algorithm.

2.3 Microfacet theory and Distribution of Visible Normal(VNDF)

Microfacet theory[9][10] is a geometric optics model. It represents a material as a collection of small specular microfacets. With different distribution of those facts, which is the distribution of microfacet normals ,we can represent both specular and rough materials. Analytic models based on microfacet theory will have dedicated distribution of normals, which we denote as $D(\omega_m)$. The state of the art importance sampling technique based on microfacet theory is the distribution of visible normals[11][12]. Given a fixed incident direction, we can sample a half vector and then compute the outgoing direction. The general form for VNDF is

$$D_{\omega_i}(\omega_m) = \frac{G_1(\omega_i, \omega_m)|\omega_i \cdot \omega_m| D(\omega_m)}{|\omega_i \cdot \omega_g|} \tag{2.6}$$

where G_1, the masking function, for GGX[10] and Beckmann[13] models has analytic solutions and $D(\omega_m)$ is the normal distribution function. The normal distribution

function of anisotropic GGX is

$$D(\omega_m) = \frac{1}{\pi\alpha_x\alpha_y \cos^4\theta_m(1+\tan^2\theta_m(\frac{\cos^2\phi_m}{\alpha_x^2}+\frac{\sin^2\phi_m}{\alpha_y^2}))^2} \tag{2.7}$$

If we can find a good fit of the real-life material to an analytic microfacet model, we might be able to use the VNDF as a proposal and as the pdf during rendering.

2.4 Generative Models, Normalizing Flows and Neural Importance Sampling

Various neural networks have been proposed to learn generative models. Generative models are a type of machine learning model that are designed to generate new data samples that resemble the training data. These models capture the probability distribution of the data, enabling them to transform random samples from a latent space to new instances that are similar to, but distinct from, the examples they were trained on.

A successful application of generative model in graphics is Neural Importance Sampling[14][15]. It utilizes a special kind of generative model, called normalizing flows[16][17], which are capable of both evaluating and sampling from a probability density function, to learn the desired distribution and sample from. Normalizing flows consist of a few coupling layers. The coupling layer takes a Nd vector and partitions its dimensions into two groups. One group is unchanged and is used to parameterize the transformation of the second group. This unchanged group is usually fed into a neural network, and the outputs of the neural network will be used to transform the second group. Through these neural networks, normalizing flows have the ability to express complex probability density. Because of this design, the neural network will not affect the computation of the Jacobian determinant, enabling us to have a tractable Jacobian for sampling and estimating density. A normalizing

flows take in an N-d random variables and by running through few coupling layers, it will generate N-d random variables that follows the learned pdf. It can also compute the pdf of any given samples. Various coupling transforms,neural network architectures and permutation strategies between coupling layers have been proposed but they all follow what's discussed above.

Because of these layered architecture and easy-to-compute Jacobian, normalizing flows are capable of doing both evaluating and sampling. Running it forward is training/evaluating while running it backward will simply be sampling.

Neural Importance Sampling is successfully applied in generating light paths more efficiently by learning where the path integral is high. Both pretraining it with large amounts of data and training and sampling in an interleaved manner works.

It is also used as the pdf of neural-network based BRDFs to make importance sampling them in a renderer possible[18][19].

Normalizing flows are also used in representing BRDF in the context of inverse rendering[20]. In this case, the normalizing flows are pretrained using 100 MERL materials in a multidimensional latent space(this latent space is not the latent space where random variables are in) and a new material can be represented by optimizing it in this MERL latent space.

These applications cover the two use cases we want: importance sampling during acquisition and importance sampling the acquired material during rendering.

Since normalizing flow is simply a very complex sampler with neural networks and carefully designed coupling transform functions, we might be able to integrate it into a MCMC algorithm and achieve adaptive sampling by training and sampling in an interleaved manner.

3

MCMC Proposals

In this section we will discuss the MCMC proposals we have come up with, from simple, non-adaptive proposals to complex, adaptive proposals.

3.1 Spherical Gaussian Proposal

The simplest proposal working directly on directions are spherical Gaussian(which is the 3D case of von-Mises Fisher distribution). The 3D von-Mises Fisher distribution is

$$f_{vMF}(\omega) = \frac{\kappa}{4\pi \sinh \kappa} \exp\left(\kappa \mu^T \omega\right) \tag{3.1}$$

where κ controls how concentrated the distribution is to the mean direction μ. Directly perturbing in the direction(solid angle) space is ideally better than perturbing it in spherical space since it can explore the domain more evenly. the spherical Gaussian could work both in ω_i, ω_o space and ω_h, ω_d space

3.2 Optimization for reciprocity and isotropic materials

For any material, we can operate in a 4D space $[\theta_i, \phi_i, \theta_o, \phi_o]$ which is directly transformed from solid angle $[\omega_i, \omega_o]$. Another BRDF parameterization is the half-diff space[21], where $\hat{\omega_h}$ is the normalized version of half vector $\omega_h = \frac{\omega_i}{||\omega_i||} + \frac{\omega_o}{||\omega_o||}$ and $\hat{\omega_d}$ is the normalized difference vector which is just the incident direction in a frame of reference where $\hat{\omega_h}$ is the north pole(Z axis). In our implementation, we compute the X axis of this new coordinate system as $(\sin(\theta_h + \frac{\pi}{2}) * \cos\phi_h, \cos(\theta_h + \frac{\pi}{2}) * \cos\phi_h, \cos(\theta_h + \frac{\pi}{2}))$. And we compute the Y axis as the cross product of X and Z axis $\vec{Y} = \vec{X} \times \vec{Z}$. Thus, we will have ω_h and ω_d, we can transform them into spherical coordinates $[\theta_h, \phi_h, \theta_d, \phi_d]$. Good properties about the half-diff parameterization are that:

- It directly parametrizes BRDF with respect to θ_h, which is one of the most important distribution in microfacet theory.
- There are a few symmetric properties that can reduce the dimension and range of the domain for isotropic material.

3.2.1 Symmetry from Reciprocity

The Helmholtz Reciprocity holds true for any BRDF. So we have $BRDF(\omega_i, \omega_o) = BRDF(\omega_o, \omega_i)$, in the half-diff space this simply becomes $\phi_d = \phi_d + \pi$. However, the Helmholtz Reciprocity is true for BRDF and what we want to evaluate is the cosine weighted BRDF, the reciprocity does not hold if there is an additional multiplication of cosine term. So we can not use this reciprocity during our MCMC acquisition process, but we still can use the reciprocity to generate 2x samples after acquisition

3.2.2 Isotropic Material and bilateral symmetry

For isotropic materials, the ϕ_h dimension of the half-diff space can be optimized out. Also, there exists the bilateral symmetry[22] $\phi_d = \pi - \phi_d$. This symmetry is not affected by the cosine term so we could use it during the MCMC process. Thus, for isotropic material, we can operate the MCMC on a 3D domain $\theta_h \subset [0, \frac{\pi}{2}), \theta_d \subset [0, \frac{\pi}{2}), \phi_d \subset [0, \pi)$ However, if we reduce the domain to 3D, we lose the ability to perturb the direction.

3.3 Simple Primary Sample Space Proposal

A simple primary sample space proposal is that, we form three samplers for $\theta_h, \theta_d, \phi_d$, where

$$u \sim U(0,1)$$

$$\theta_h = \arcsin(u_1) \leftrightarrow pdf(\theta_h) = \cos(\theta_h)$$

$$\theta_d = u_2 * \frac{\pi}{2} \leftrightarrow pdf(\theta_d) = \frac{2}{\pi}$$

$$\phi_d = u_3 * \pi \leftrightarrow pdf(\phi_d) = \frac{1}{\pi}$$

note that θ_h is not uniformly sampled as we want to focus more samples to the location where θ_h is small.

For the primary sample space perturbation, we follow what was suggested in the original paper. We use a Gaussian with standard deviation σ to perturb the random variables, we also do random global step(uniformly sample all the random variables) with a probability p_{global}. The actual proposal for primary sample space MCMC is pretty simple, but if we can have carefully designed sampler, it will still work well.

The overall pdf term of this sampler for primary sample space MCMC is

$$\begin{aligned} pdf(\omega_i, \omega_o) &= pdf(\theta_h)|\frac{d\theta_h d\phi_h}{d\omega_h}| * pdf(\theta_d)pdf(\phi_d)|\frac{d\theta_d d\phi_d}{d\omega_d}| * |\frac{d\omega_h d\omega_d}{d\omega_i d\omega_o}| \\ &= pdf(\theta_h) * pdf(\theta_d) * pdf(\phi_d) * \frac{1}{\sin\theta_h} * \frac{1}{\sin\theta_d} * \frac{1}{4 * \cos\theta_d} \\ &= \frac{\cos(\theta_h)}{2\pi^2 \sin(\theta_h)\sin(\theta_d)\cos(\theta_d)} \end{aligned} \tag{3.2}$$

3.4 VNDF Sampler in Primary Sample Space

Based on the primary sample space technique, we can come up with more complicated proposals. The VNDF importance sampler[11][23] could be used as long as we have a good roughness fit to the material we want to measure. Ideally this proposal will be much more efficient than the previous simple one. In this section, we only talk about how proposal works, the fitting problem is discussed in later section. Since VNDF requires a given ω_i, and there is actually no good empirical and theoretical knowledge on how the BRDF is distributed for different ω_i. The BRDF under microfacet theory is

$$BRDF(\omega_i, \omega_o) = \frac{F(\omega_i, \omega_h)G_2(\omega_i, \omega_o, \omega_h)D(\omega_h)}{4|\omega_i \cdot \omega_g||\omega_o \cdot \omega_g|} \tag{3.3}$$

When evaluating cosine weighted BRDF, the term $|\omega_o \cdot \omega_h|$ cancels out. Ignoring the Fresnel term and G_2 term as they will not make big influence on the BRDF, we will see that, when $\cos(\theta_i)$ is small, the cosine weighted BRDF might be larger. And this is true to many real life materials that follow microfacet theory. So actually we might want to come up with ways to importance sampling this trend. For now, the

incident direction is sampled uniformly

$$\theta_i = \arccos(u_1)$$

$$\phi_i = 2\pi u_2$$

$$pdf(\omega_i) = \frac{1}{2\pi}$$

Then the ω_h is importance sampled from VNDF $pdf(\omega_h) = D_{\omega_i}(\omega_h)$, please refer to the original VNDF paper on how to do this. The overall pdf of this sampler for primary sample space MCMC is

$$pdf(\omega_i, \omega_o) = pdf(\omega_i) * pdf(\omega_h)|\frac{d\omega_h}{d\omega_o}| = pdf(\omega_i) * D_{\omega_i}(\omega_h) * \frac{1}{4|\omega_o \cdot \omega_h|} \quad (3.4)$$

3.5 Normalizing Flows Sampler in Primary Sample Space

3.5.1 Latent Space of Normalizing Flows

Many normalizing flows use a Gaussian latent space which is unbounded. However, in our case, the spherical coordinates we want to sample are bounded. As a result, we choose to use the uniform latent space in $[0, 1]^N$ hypercube, the same as NIS[14]. In addition, the uniform latent space can be directly applied to the current framework of primary sample space MCMC. According to our experiments in training some analytic BRDFs, normalizing flows generally work better in the half-diff space than the $\theta_i, \phi_i, \theta_o, \phi_o$ space. So, we could simply convert the ND uniform random variables uniformly to the spherical coordinates we need. Since this conversion is a simple scaling, the Jacobian term will cancel out everywhere.

$$\theta_h = u_1 * \frac{\pi}{2}, \theta_d = u_2 * \frac{\pi}{2}, \phi_d = u_3 * \pi$$

3.5.2 Normalizing Flow Architecture

In general, normalizing flows with more layers and more complex coupling functions can represent more complex density functions. In our case, we use the state-of-

the-art piecewise-quadratic warp[14][24] that is bounded in $[0, 1]$ as the coupling transform. The piecewise-quadratic coupling transforms admit a piecewise-linear PDF, modeled using $K+1$ vertices. For each input, the neural network is responsible for outputting the vertical coordinates of those vertices and the horizontal differences between neighboring vertices(bin widths). Different applications use different neural network structures. We found no significant difference between different structures in our experiments. The we use the neural network structure provided by neural spline flows[24] and the normflows[25] library. It contains two blocks of residual network, each block with two linear layers with 128 hidden channels. The output of the last layer is random shuffled as the input to the next layer.

In conclusion, in the evaluating/training process, we have a vector of spherical coordinates $(\frac{\theta_h}{\frac{\pi}{2}}, \frac{\theta_d}{\frac{\pi}{2}}, \frac{\phi_d}{\pi})$ which is scaled to the unit interval $[0, 1]^3$. We run the normalizing flows forward and we will get a vector (u_1, u_2, u_3) which is the random variables used to sample these spherical coordinates and a log probability p as the output of the normalizing flows. In the sampling process, we run the normalizing flows backward with random vector (u_1, u_2, u_3) to the normalizing flows and get $(\frac{\theta_h}{\frac{\pi}{2}}, \frac{\theta_d}{\frac{\pi}{2}}, \frac{\phi_d}{\pi})$ and a log probability p as the output.

The log probability p we get in these two processes are the pdf of the scaled spherical coordinates

$$\exp(p) = pdf(u_{\theta_h}, u_{\theta_d}, u_{\phi_d}) = \frac{pdf(\theta_h, \theta_d, \phi_d)}{\frac{\pi}{2} * \frac{\pi}{2} * \pi} \tag{3.5}$$

We don't have the separate pdf of each spherical coordinates, but we will have the product of them. Note that the second step in the above pdf equation is to count for the scaling of spherical coordinates. But as we mentioned in previous section, these Jacobian terms will be cancelled out everywhere in the MCMC process so it can be safely ignored in the implementation.

3.5.3 Train Normalizing Flows Using KL Divergence

The general way to train a normalizing flows is to use the max log likelihood of all the samples[26][27]. This training procedure assumes the samples are directly drawn from the distribution we want to learn. In fact, this is a special case of training it with Kullback–Leibler(KL) divergence between source and the target distributions.

$$D_{KL}(p||q) = \int p(x) \log \frac{p(x)}{q(x;\theta)} dx \tag{3.6}$$

The gradient of KL divergence with respect to neural network parameters θ is

$$\nabla_\theta D_{KL}(p||q) = -\nabla_\theta \int p(x) \log \left(q(x;\theta) dx\right. \tag{3.7}$$

$$= E[-\frac{p(X)}{h(X)} \nabla_\theta \log q(X;\theta] \tag{3.8}$$

where the expectation is over $X \sim h(x)$. Mathematically $h(x)$ could be any distribution we are sampling from. If $h(x) = p(x)$, the expected value over the gradient will be simplified to only the log likelihood. Then the job becomes maximizing the log likelihood. But $h(x)$ could be other distributions that we are sampling from.

In our case, we can sample from the target distribution because our normalizing flows is in the framework of MCMC. Note that we must only use accepted samples in MCMC because only accepted samples follow the target distribution.

Others suggest using a reverse KL divergence loss[26] in the case that we can not draw samples directly from the target distribution, we have not explored this yet.

3.5.4 Space to Importance Sampling in

The normalizing flows are only able to learn a distribution in spherical coordinate$(\theta_h, \theta_d, \phi_d)$ space, but we eventually want to importance sampling BRDF in ω_i, ω_o space. As a result, the samples from our MCMC algorithm follows the target distribution

$$pdf(\omega_i, \omega_o) = \frac{BRDF(\omega_i, \omega_o) * \cos(\theta_o)}{c} \tag{3.9}$$

After applying the Jacobian, the pdf in spherical space becomes

$$pdf(\theta_h, \theta_d, \phi_d) = \frac{BRDF(\omega_i, \omega_o) * \cos(\theta_o)}{c} * |\frac{d\omega_i d\omega_o}{d\omega_h d\omega_d}| * |\frac{d\omega_d}{d\theta_d d\phi_d}| * |\frac{d\omega_h}{d\theta_h d\phi_h}| \quad (3.10)$$

It seems like this pdf does not satisfy the requirement of being able to importance sampling the target distribution so that one can train it with max log likelihood. However, if we think of this in another way, what we actually did is importance sampling in the space of $(\theta_h, \theta_d, \phi_d)$ with respect to

$$BRDF * cos * |\frac{d\omega_i d\omega_o}{d\theta_h d\theta_d d\phi_d}| \quad (3.11)$$

And by simply training it with max log likelihood, the normalizing flows learn the distribution

$$pdf(\theta_h, \theta_d, \phi_d) = \frac{BRDF * cos * |\frac{d\omega_i d\omega_o}{d\theta_h d\theta_d d\phi_d}|}{c} \quad (3.12)$$

After converting it back to ω_i, ω_o space, we still get what we want

$$pdf(\omega_i, \omega_o) = pdf(\theta_h, \theta_d, \phi_d) * |\frac{d\theta_h \theta_d \phi_d}{d\omega_i \omega_o}| = \frac{BRDF * cos}{c} \quad (3.13)$$

This observation raises another question: can we simply do both MCMC and normalizing flow training in the spherical space$(\theta_h, \theta_d, \phi_d)$ and get rid of all those Jacobian terms? This issue will be discussed later.

3.5.5 Pretraining Normalizing Flow

Although normalizing flows can learn the target distribution on the fly, the initialization of the flows is random. Also, we will want the flows to generate decent proposals from the beginning as we have limited sample budget. Thus we need to pretrain the model.

Pretrained normalizing flows were used in the iBRDF[20] to express BRDF where the normalizing flows will first learn a 16D latent space(embedding code) of 100

MERL materials. And the new BRDF is optimized to find its own 16D embedding code in the space of MERL materials

Currently, we simply want the initial normalizing flows to have knowledge about the approximate roughness of the new material. Thus, we train a few normalizing flows, each with a simple GGX material of different roughness. After this pretrain process, we confirmed that the normalizing flows almost learn a perfect GGX BRDF distribution, which can be used as the initial state of the flows.

3.5.6 Adaptive Training During MCMC

During the MCMC process, we need to update our normalizing flows to fit the real materials. We train the flows using small batches of accepted samples, and generate new proposals using the updated normalizing flows. The problem of optimizing normalizing flows is that one should not do stochastic gradient descent on a batch of samples for a large amount of epochs. Doing this is equivalent to telling normalizing flows that all the target distribution will only generate that single batch of samples. Without other batches of samples, the flows will be wrongly optimized to only focusing on this single batch of samples. This implies that to train good normalizing flows, one generally needs large amount of samples.

3.5.7 Controlled MCMC

Since we are training the normalizing flows on the fly, we are essentially updating our MCMC proposals periodically. This falls in a special kind of Markov chain Monte Carlo algorithm called controlled MCMC/adaptive MCMC[28]. Researches on controlled MCMC[28] point out that the Markov chain will not converge theoretically if one just update the proposal without any constraints. These constraints need to be followed in the context of adaptive MCMC rendering[29] to obtain unbiased results. There are two requirements for a controlled MCMC algorithm to converge:

- Diminishing Adaptation: The amount of adaptation must decrease as the number of iterations increases. This condition ensures that changes to the proposal distribution become smaller over time, allowing the algorithm to satisfy the necessary conditions for a traditional MCMC algorithm to converge. Essentially, the chain's transition kernel must converge to a limiting kernel.

- Bounded Convergence of Adaptations: The adaptations must not be too large or unbounded; they should be controlled to ensure that the Markov chain remains ergodic.

While the second requirement is typically satisfied, the first one requires us to reduce the learning rate of the optimizer, or reduce the frequency of updating normalizing flows, to make the MCMC converge theoretically.

However, since the Markov chain in our application will run very limited amount of iterations and we do not require the samples to be perfectly unbiased, we did not include this requirement in our MCMC algorithm. We do enforce this during the validation process in the next section.

3.6 Validation of convergence

To validate that all of the proposals mentioned in this section and the MCMC algorithm converge, we will simply try to compute an arbitrary integral using the samples we get from MCMC and compare it with ground truth. Assume we want to compute the integral

$$\int_{\Omega}\int_{\Omega} BRDF(\omega_i, \omega_o) * \cos(\theta_o) * |\omega_d \cdot \omega_g| * |\omega_i \cdot \omega_o| d\omega_i d\omega_o$$

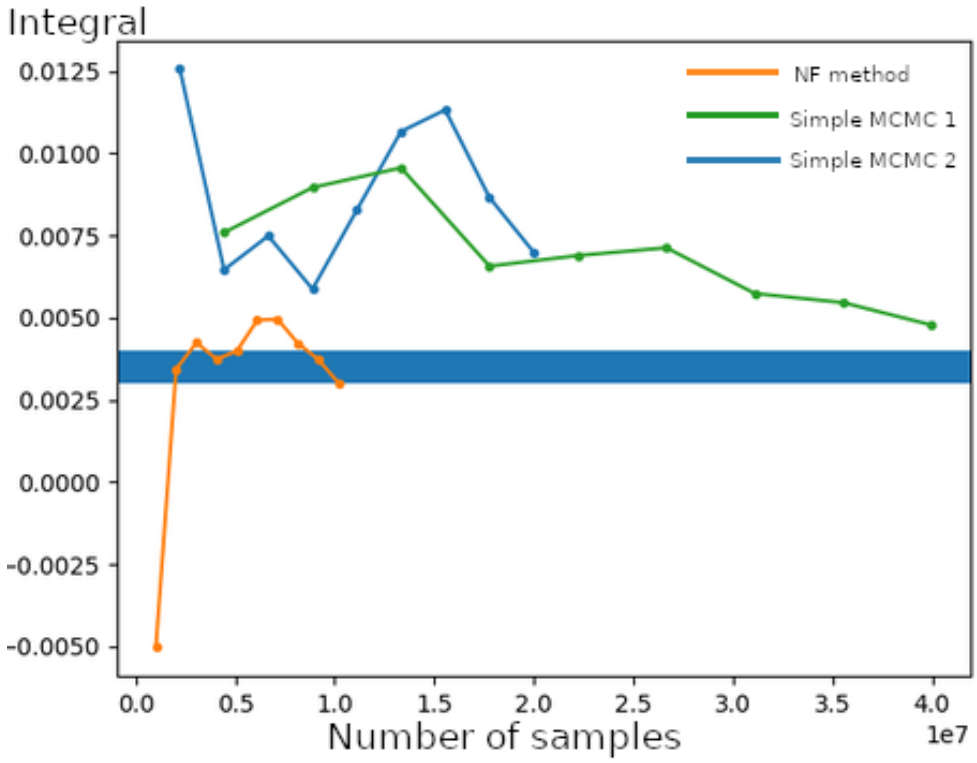


Figure 3.1: Convergence Test

and MCMC samples are generated from the target distribution, as it should be, the Monte Carlo estimator of this integral using MCMC samples will be

$$\frac{1}{N} * \frac{BRDF(\omega_i, \omega_o) * \cos(\theta_o) * |\omega_d \cdot \omega_g| * |\omega_i \cdot \omega_o|}{\frac{BRDF(\omega_i, \omega_o) * \cos(\theta_o)}{c}} = \frac{1}{N} * |\omega_d \cdot \omega_g| * |\omega_i \cdot \omega_o| \quad (3.14)$$

The experiments show that the MCMC using all proposals converge to the same value, we also observed that the adaptive normalizing flows converge much faster than the simple primary sample space method. This simple test shows the potential strong power of adaptive MCMC using normalizing flows.

4

Post-acquisition Processing

This section describes what we do after we generate the samples by importance sampling the cosine weighted BRDF. In general, we want to come up with a way to evaluate the BRDF at every incident and outgoing angles using our samples, and we want to come up with a good pdf to enable any renderer to do efficient importance sampling. Since the samples we have from MCMC acquisition is not structured, we can not easily make our samples into a BRDF that is usable in a renderer. Thus, a good post processing procedure to interpolate the BRDF is as important as the acquisition process itself. Note that, only the accepted samples in the MCMC process follow the target distribution, this means we will have repeated samples in the accepted samples. However, we also want to utilize the rejected samples since we actually does not care if these samples follow the target distribution or not. Thus, all the fitting and optimization processes in this section use all samples, including the rejected ones.

4.1 Interpolation and Fit to Analytic Models

The simplest way to generate data in our case is through interpolation. We utilize the idea of importance sampling weight and fitted BRDF to make the process more stable. First, we will discuss about fitting measured BRDF to analytic BRDF

Fitting measured BRDF to analytic BRDF is a common way to compress measured BRDF. For example, the BRDF can be fitted to the mixture of a few GGX BRDFs and a Lambertian BRDF[1]. These techniques are often used to compress measured BRDF for densely sampled BRDF as they have large amounts of BRDF samples and will take up a lot of disk space. Also, these methods are not guaranteed to work for every material. Because of these reasons, we want to only use the pdf from the fitted analytic model, while keep using our original BRDF values. However, this will restrict us to only being able to use the models with analytic pdf, such as a single lobe of GGX, which will reduce the ability to fit. For the fitting process, we compute the cosine weighted root mean square error[30]

$$E = \sqrt{\frac{\sum \left(M(\omega_i, \omega_o)\cos\theta_o - A(\omega_i, \omega_o, p)\cos\theta_o\right)^2}{n}} \tag{4.1}$$

where M is the acquired BRDF and A is the analytic fitted BRDF.

For the following example, we will assume we fit this material successfully to a single lobe of GGX with roughness α. We also tried to fit the acquired data to some more advanced models like the ABC model[2], but the optimization failed for many materials, so we keep using GGX while it might not be the best fit.

Once we have this fitted material, though it might be not accurate, we will know an approximate pdf at any outgoing direction given an incident direction using the distribution of visible normal.

$$pdf(\omega_o|\omega_i) = D_{\omega_i}(\omega_h)\left|\frac{d\omega_h}{d\omega_o}\right| \tag{4.2}$$

For every BRDF samples we have, we can compute the importance sampling weight that is used in Monte Carlo estimation as if this sample is sampled from the pdf above

$$weight(\omega_i, \omega_o) = \frac{BRDF(\omega_i, \omega_o) * \cos\theta_o}{pdf(\omega_o|\omega_i)} \tag{4.3}$$

The quality of interpolation depends heavily on the data. if the data is smooth/flat, then the result will generally be better. Theoretically, a perfect importance sampling method(the pdf is exactly proportional to the cosine weighted BRDF itself) will flatten the BRDF to a constant weight. However, for a measured material, we need to sample from some fitted analytical material, or from some generated tabulated pdf. The pdf in this case will **not** be perfect. But a good pdf will still flatten the BRDF and result in a better interpolation for pdf.

4.2 Generating Grids for Renderer

Once we have a way to interpolate BRDF, we can try to generate data structures that can be used in renderer. The idea is to generate a few 2D grids, each grid correspond to a fixed ω_i. The reason to do this is that the renderer only evaluates the BRDF and samples the direction given an ω_i, thus we need to convert our full BRDF data in ω_i, ω_o space to some BRDF grids that has a fixed ω_i, so that it can be used in a renderer.

4.2.1 Determining Incident Elevation

An alternative to compute the VNDF, as mentioned in the original VNDF paper and the adaptive paper, is

$$D_{\omega_i}(\omega_h)) = \frac{|\omega_i \cdot \omega_h| D(\omega_h)}{\int_\Omega |\omega_i \cdot \omega_h| D(\omega_h) d\omega_h} \tag{4.4}$$

The denominator is a normalization factor $\sigma(\theta_i)$. The larger the $\sigma(\theta_i)$ is, the

larger the integral of distribution of visible normal is given this θ_i. As a result, we will want to focus more on the θ_i that has a large distribution of visible normal in total. Then We can determine the discrete θ_i by computing this σ term. Sadly there is no analytical solution to sigma so it can only be computed using Monte Carlo and stored by a 1D tabulated distribution. Once we have the tabulated distribution of σ, we sample it using random numbers with uniform interval.

Currently this is implemented in GGX material. If we do not have a fit, we will not be able to determine θ_i non-uniformly

4.2.2 PDF grid generation

For each θ_i we generate in the previous part, we will have a 2D VNDF grid for it. v(the row index) will determine ϕ_h and u(the column index) will determine θ_h separately:

$$\phi_h = 2\pi v$$

$$\theta_h = \frac{\pi}{2} u^2$$

So we will have a unique ω_h at each discrete point in the VNDF grid and the grid will cover the whole hemisphere. Since we know the NDF, We can compute the VNDF using either

$$D_{\omega_i}(\omega_h) = \frac{G1(\omega_i, \omega_h)|\omega_i \cdot \omega_h| D(\omega_h)}{|\omega_i \cdot \omega_h|} \tag{4.5}$$

or

$$D_{\omega_i}(\omega_h) = \frac{|\omega_i \cdot \omega_h| D(\omega_h)}{\int_\Omega |\omega_i \cdot \omega_h| D(\omega_h) d\omega_h} \tag{4.6}$$

The latter might be more accurate if we run Monte Carlo to compute the σ term. Moreover, because we are sampling from a grid, we can not simply store the VNDF there, we should apply the Jacobian from solid angle to spherical coordinate, and

from spherical coordinate to (v, u) space so that the 2D tabulated table will sample it correctly.

$$p(v,u) = D_{\omega_i}(\omega_h) * |\frac{d\omega_h}{d\theta_h d\phi_h}| * |\frac{d\theta_h}{du}| * |\frac{d\phi_h}{dv}|$$
$$= D_{\omega_i}(\omega_h) * \sin\theta_h * u\pi * 2\pi$$

Here we are computing the VNDF assuming $\phi_i = 0$, this will result in some rotation on ϕ_h in rendering code in the next section. Basically this means what we have in the grid is $\Delta\phi$.

Note that, the use of pdf grid is to do efficient sampling, theoretically the rendering result should be the same no matter what pdf we use if we run the renderer for a long enough time.

4.2.3 BRDF grid generation

Now we will have the weight $w(\theta_h, \theta_d, \phi_d)$ in 3D space, we generate the interpolation data structure to interpolate this weight given any $\theta_h, \theta_d, \phi_d$ parameters. We will generate BRDF following these steps:

1. use (v, u) values with uniform interval, compute ω_h the same way in VNDF grid
2. compute ω_o, ω_d from the ω_h we get from the above sampling procedure and the given ω_i
3. compute $\theta_h, \theta_d, \phi_d$ from ω_h, ω_d
4. reduce ϕ_d domain: if $\phi_d > \pi$ then $\phi_d = 2\pi - \phi_d$. Test shows $f_r(\phi_d)\cos\theta_{o1} = f_r(2\pi - \phi_d)\cos\theta_{o2}$(Is this reciprocity?)
5. get the interpolated weight of corresponding parameter $(\theta_h, \theta_d, \phi_d)$ from the interpolator

6. store this weight to the (v, u) location at the BRDF grid

7. For location that can not be interpolated because it's out of range, run a Poisson equation solver(I don't know if the solver is 100% correct as it is a third-party library) to fill the data

We implement a Mitsuba plugin, based on the source code of the EPFL material Mitsuba plugin, to render images using the data structures mentioned above, by doing a bi-linear interpolation between two slices.

4.3 Interpolation Using Normalizing Flows

The normalizing flows are used for density estimation, so it will satisfy the property of a probability density function

$$\int_0^1 \int_0^1 \int_0^1 pdf(u_{\theta_h}, u_{\theta_d}, u_{\phi_d}) du_{\theta_h} du_{\theta_d} du_{\phi_d} = 1 \tag{4.7}$$

By optimizing the normalizing flows, the learned pdf at locations where the target distribution is low will be automatically compressed because the normalizing flows must integrates to 1. Thus, the best situation is that the gradient descent optimization of normalizing flows will automatically learn the distribution at places even if we have no samples. Following this idea, we propose a second stage optimization of normalizing flows after acquisition.

Normalizing flows are optimized during acquisition using KL divergence in hope of learning the target distribution. Here, the optimization serves a different purpose.

Note that the distribution we want to learn(assuming we are importance sampling in ω_i, ω_o space is

$$pdf(\omega_i, \omega_o) = pdf(\theta_h, \theta_d, \phi_d) * |\frac{d\theta_h \theta_d \phi_d}{d\omega_i \omega_o}| = \frac{BRDF * cos}{c} \tag{4.8}$$

And the output of normalizing flow is simply $pdf(\theta_h, \theta_d, \phi_d)$. So we need a step of conversion, to convert the half-diff space pdf to ω_i, ω_o space pdf which is proportional to the cosine weighted BRDF. For the second stage, our idea is still fitting the normalizing flows to the BRDF we have. Note that, in the case of normalizing flows, the cos-weighted BRDF could be rewritten as

$$BRDF(\omega_i, \omega_o) * \cos\theta_o = pdf(\omega_i, \omega_o) * c \tag{4.9}$$

where c is an currently unknown constant(normalization factor)

$$c = \int_\Omega \int_\Omega BRDF(\omega_i, \omega_o) \cos\theta_o d\omega_i d\omega_o \tag{4.10}$$

We will try to compute a relative loss between $pdf * c$, which is the estimated cosine weighted BRDF by the normalizing flow, and the ground truth cosine weighted BRDF for each channel.

$$loss = \sqrt{\frac{1}{N}\sum(\frac{BRDF(\omega_i^n, \omega_o^n) * \cos\theta_o^n - pdf(\omega_i^n, \omega_o^n) * c}{BRDF(\omega_i^n, \omega_o^n) * \cos\theta_o^n})^2} \tag{4.11}$$

However, the problem is that we don't know this normalization factor c, doing gradient descent on both the neural network parameters and c generally does not work. We have come up with an interleaved method to update c and neural network parameters. When neural network parameters are fixed, the derivative of $\sum(BRDF * cos - pdf * c)^2$ with respect to c is

$$\sum(2c * pdf(\omega_i^n, \omega_o^n) - 2 * pdf(\omega_i^n, \omega_o^n) * BRDF(\omega_i^n, \omega_o^n) * \cos\theta_o^n) \tag{4.12}$$

Let the derivative equal zero, then the loss will be at its minimum:

$$\sum 2c * pdf(\omega_i^n, \omega_o^n)^2 = \sum 2 * pdf(\omega_i^n, \omega_o^n) * BRDF(\omega_i^n, \omega_o^n) * \cos\theta_o^n \tag{4.13}$$

$$c\sum pdf(\omega_i^n, \omega_o^n)^2 = \sum pdf(\omega_i^n, \omega_o^n) * BRDF(\omega_i^n, \omega_o^n) * \cos\theta_o^n \tag{4.14}$$

$$c = \frac{\sum pdf(\omega_i^n, \omega_o^n) * BRDF(\omega_i^n, \omega_o^n) * \cos\theta_o^n}{\sum pdf(\omega_i^n, \omega_o^n)^2} \tag{4.15}$$

After updating the neural network for a few iterations using the loss, we compute the constant deterministically. Due to the instability of gradient descent optimizer such as Adam[31], we might update c when the neural network parameters are optimized to a state that is far from ground truth. A simple solution is deferring this constant update until the iteration where the loss is the smallest.

By keeping doing this until the loss is smaller than certain threshold, it is possible for us to obtain a reasonable pair of neural network parameters and normalization factor c that will closely match the samples.

Now, we can generate BRDF grid directly using normalizing flows by computing $pdf(\omega_i, \omega_o) * c$. Since we don't have a fitted pdf in this case, we simply let the unnormalized pdf of ω_o given an ω_i be $pdf(\omega_o|\omega_i) = BRDF(\omega_i, \omega_o) * \cos\theta_o$, then we can generate a pdf grid in half vector space by applying Jacobian the same way in last section.

5

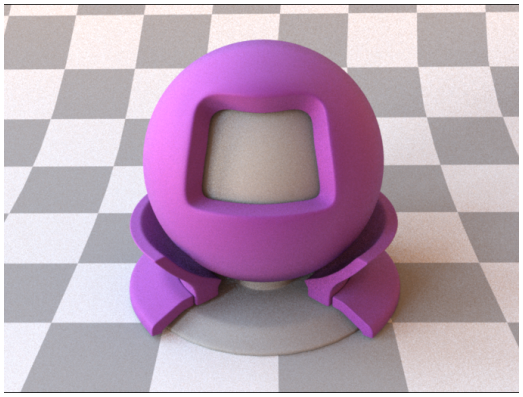
Results

This chapter shows the rendering result of different methods we have. For non normalizing flows method, we do post processing as chapter 4.1 and 4.2. For normalizing flows, the post processing is described in 4.3. All images are rendered using Mitsuba 0.6[32] using simple path tracer with next event estimation. The samples per pixel are kept the same for all images. In short, the simple method works well for rough materials, but faces more and more serious problems in more specular materials. The normalizing flows have the ability to express some channels of specular materials well, but it might fail for other channels.

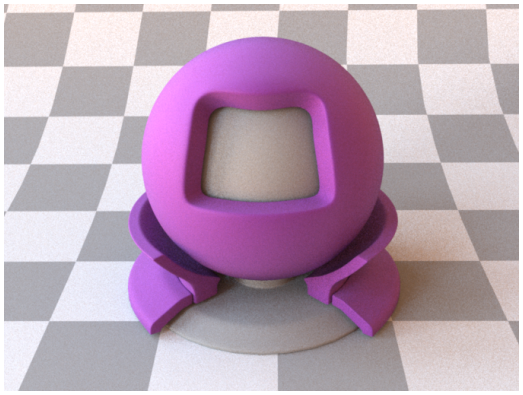
5.1 Simple Primary Sample Space

5.1.1 Rough Materials

The acquisition and post-processing pipeline work well in rough material, it will give us a nearly identical BRDF compared to the ground truth

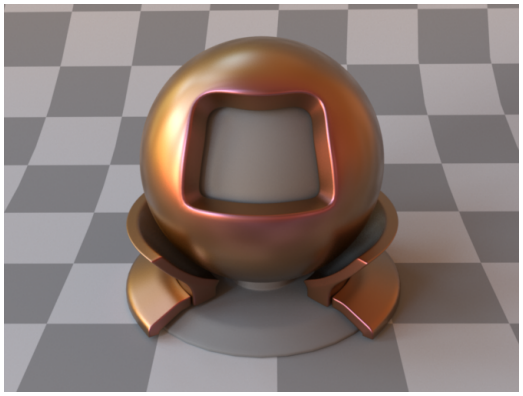


(a) Ground truth purple material

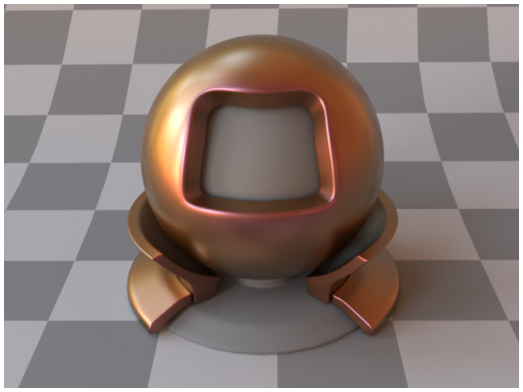


(b) Simple PSS purple material

Figure 5.1: Rough purple material comparison



(a) Ground truth cc_ibiza material

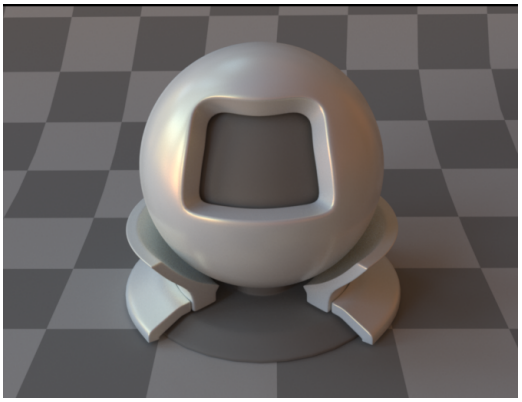


(b) Simple PSS cc_ibiza material

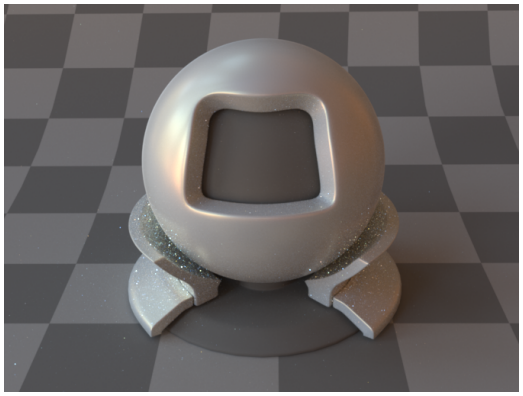
Figure 5.2: Rough cc_ibiza material comparison

5.1.2 More Specular Materials

As the material becomes more and more specular, we start to observe two problems: the noise becomes noticeable at grazing angles and the material is generally brighter than the ground truth. We can start to notice some noise for the white material, this material is more specular than the previous two. The noise and brighter problem are most noticeable in the chm_orange material. We can also easily observe the brighter problem in ilm_blue material, but because the BRDF itself has a black color, the noises at grazing angles are not observable.

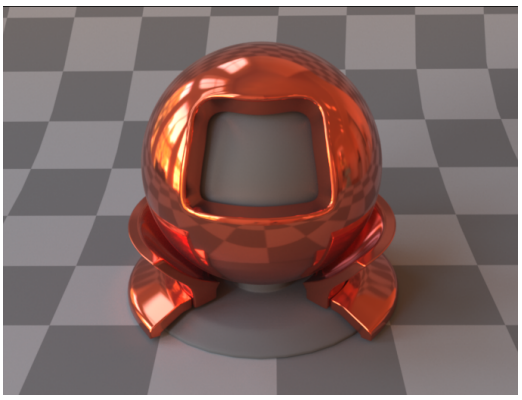


(a) Ground truth white material

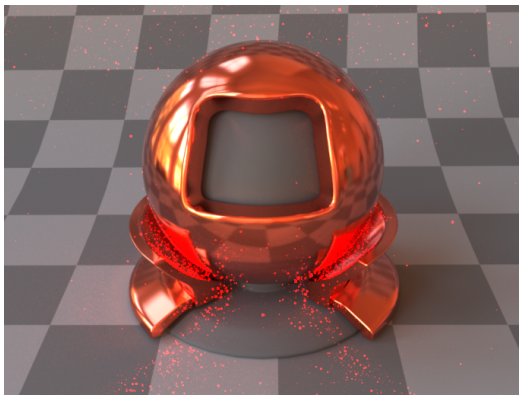


(b) Simple PSS white material

Figure 5.3: White material comparison



(a) Ground truth chm orange material



(b) Simple PSS chm orange material

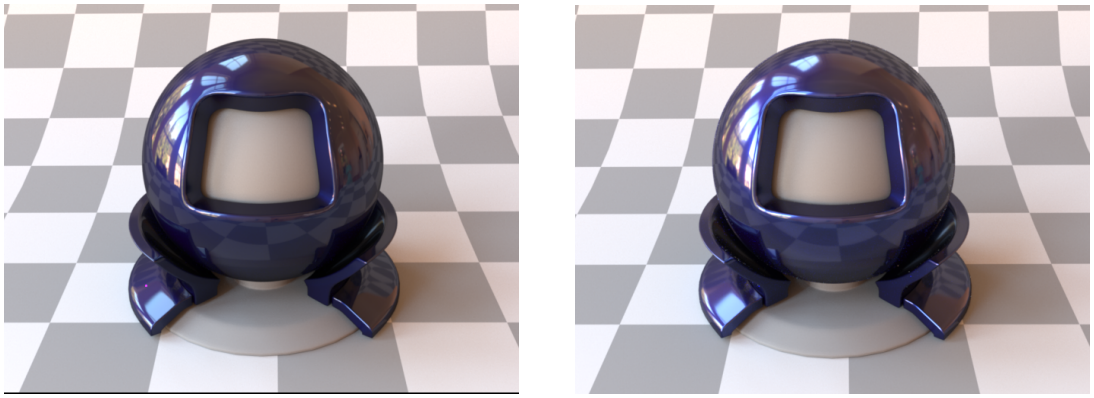
Figure 5.4: chm orange material comparison

5.1.3 VNDF proposal

The results from using a VNDF proposal whose α is close to the fitted roughness are similar to that of the simple primary sample space proposal, this suggest that the main problem might be the post-processing and interpolation part of our pipeline.

5.2 Normalizing Flows Primary Sample Space

As we switch to normalizing flows, we find that normalizing flows have the potential to reconstruct a specular material that failed previously. But if suffers from ther



(a) Ground truth ilm blue material (b) Simple PSS ilm blue material

Figure 5.5: ilm blue material comparison

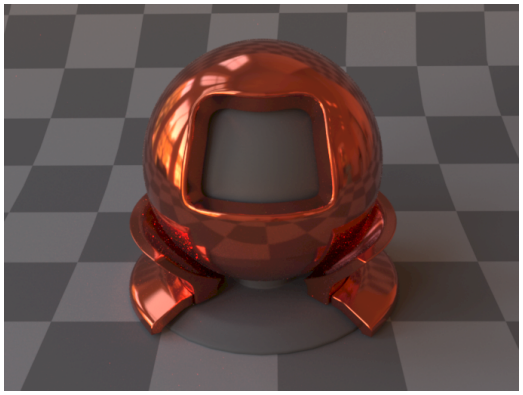
problems.

5.2.1 Comparison of pdf

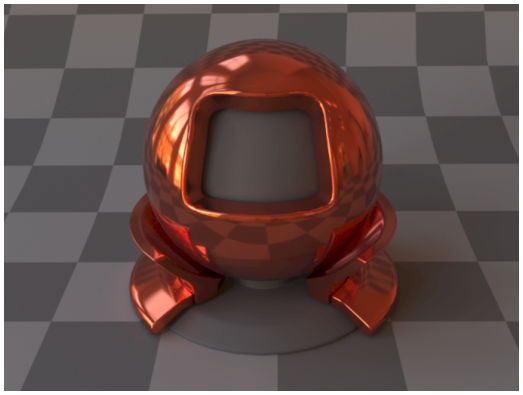
Here, we test the pdf of the two methods on chm_orange material. We generate the BRDF grid from ground truth BRDF, and only use our own pdf. The comparison is made between VNDF pdf mentioned in 4.1(BRDF is interpolated as weight) and normalizing flows pdf mentioned in 4.3. There are noticeably less noises when we use the red channel of our normalizing flows as the pdf, comparing to using a fitted VNDF as the pdf.

5.2.2 Successful Rendering for Single Channel

The reason why we only use the red channel of the chm_orange material in the pdf comparison is that, normalizing flows are only able to reconstruct the red channel in our experiments. We render a gray-scale red channel of the material for comparison, we can see that the two images are almost identical.

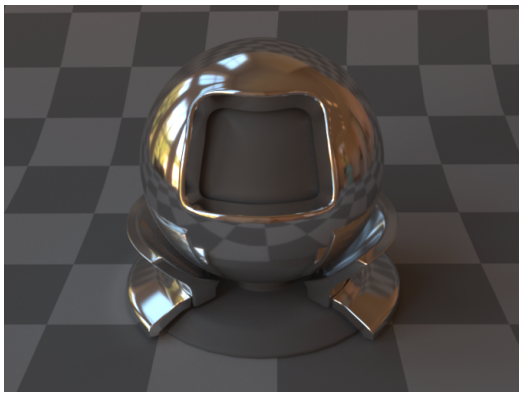


(a) Ground truth BRDF + VNDF pdf

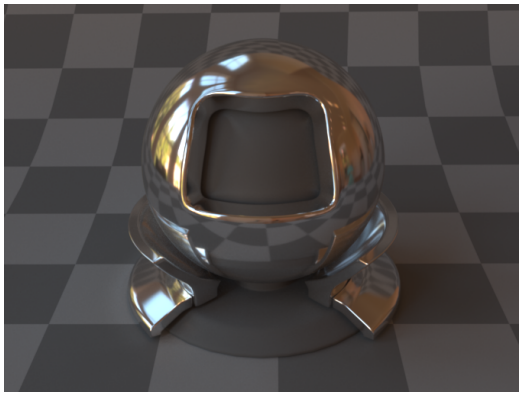


(b) Gronud truth BRDF + NF pdf

Figure 5.6: pdf comparison



(a) Ground truth gray scale red channel



(b) NF gray scale red channel

Figure 5.7: Grayscale red channel of chm_orange comparison

5.2.3 Failed Rendering for other channels

However, if we render all three channels, still using the 'good' red channel pdf. the result will look like:

5.2.4 Rough Materials

In general, the Normalizing flows are able to reconstruct rough materials, but it still faces the problems of potential failure to fit some channels. For example, we had an example of a noticeable fitting failure in blue channel, causing the material to look

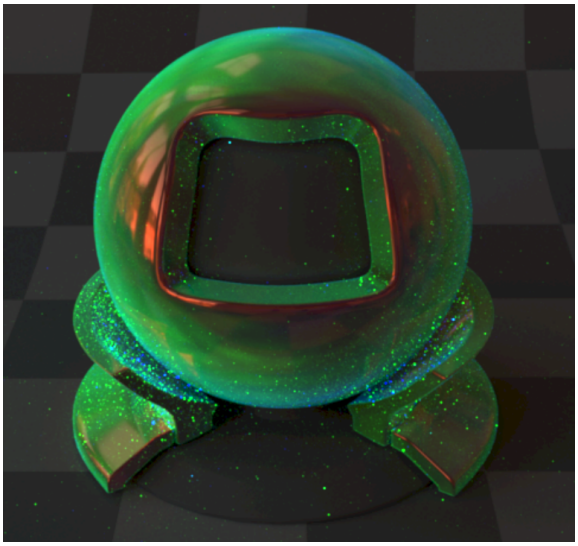
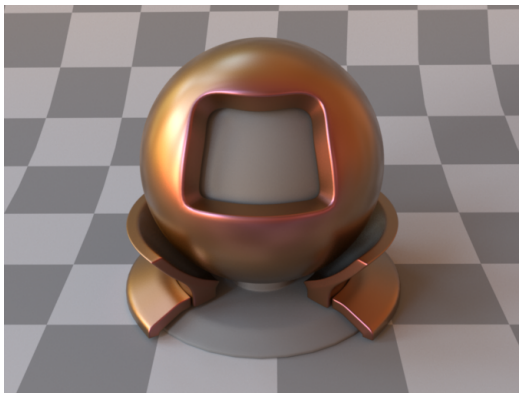
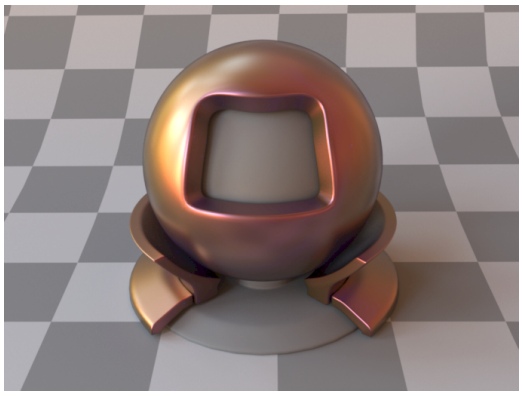


Figure 5.8: NF failure on green and blue channel

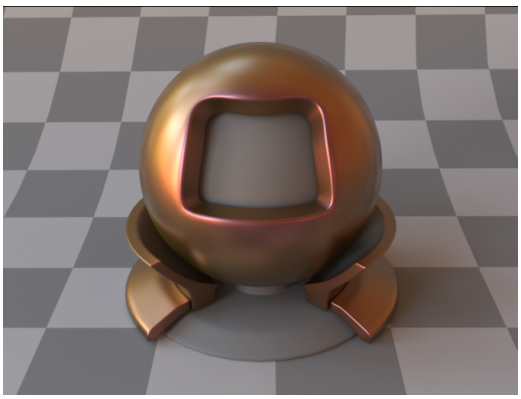
more hazy purple. This failed image use a normalizing flow that is optimized by mean square error during post-acquisition stage and expresses the BRDF in $(\theta_h, \theta_d, \phi_d)$ space.



(a) Ground truth cc_ibiza material



(b) NF cc_ibiza material(blue channel faliure)



(c) NF cc_ibiza material

Figure 5.9: Rough cc_ibiza material comparison(normalizing flows)

6

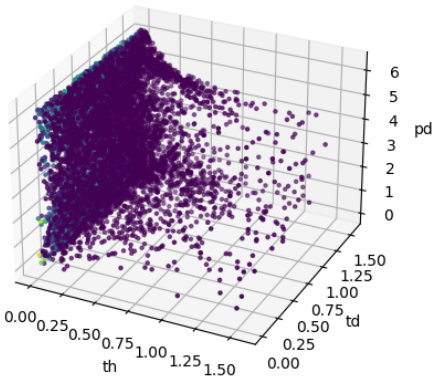
Analysis

In this section we will identify the problems shown in the results, and we will analyze the cause of the problems. All the results in this section are from chm_orange material

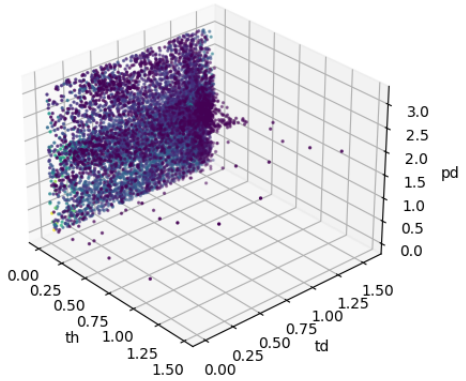
6.1 Samples Distribution

First, since we have a few proposals, we want to compare the actual sample distributions from MCMC, All graphs are produced using 10000 Samples, including rejected samples in MCMC. Here are the samples we have drawn for chm_orange material, using three different proposals: simple PSS proposal, VNDF proposal, pretrained normalizing flows proposal:

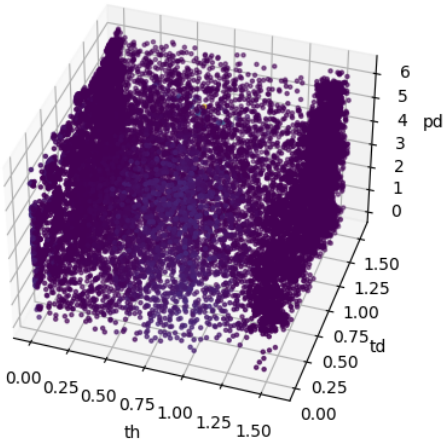
The normalizing flows use a pretrained GGX with roughness 0.1, the VNDF use GGX with roughness 0.05. We ca see that, the normalizing flows has the most samples concentrate on where θ_h is small, even if it starts from a rougher pretrained model than the non-adaptive VNDF model. The simple PSS almost has samples every where in the 3D $(\theta_h, \theta_d, \phi_d)$ space



(a) VNDF MCMC distribution



(b) NF MCMC distribution



(c) Simple PSS MCMC distribution

Figure 6.1: Samples distribution comparison with different proposals

6.1.1 Problem of Perturbing θ

Since we are showing all samples, including rejected ones, in the visualization, the samples distribution does not represent the target distribution. However, it will still contain the information on generally how samples are distributed and how well the MCMC is importance sampling the BRDF.

If we look at the simple MCMC visualization, we can see that there are more samples focusing on θ_h is small, but there are also many samples located where

$\theta_h \approx \frac{\pi}{2}$, even more than samples in the middle.

This is because, the primary sample space MCMC is perturbing a random variable in $[0, 1]$ using a Gaussian, and it use a circular perturbation to ensure the symmetry of this proposal. This means, when u_{θ_h} is close to zero, it is possible to simple perturb it to some value close to 1 because of this circular implementation in perturbation. This is actually not ideal especially for a specular material, as the BRDF where at locations where θ_h is large will be very small.

6.2 Brighter BRDF in Fitted Interpolation

6.2.1 The Most Importance Sampled Area in a 2D (θ, ϕ) Grid

In a renderer, when ω_i is fixed, the common way to importance sample is sample ω_h or ω_o. No matter what, the change of variable from solid angle to spherical coordinate involves a Jacobian term

$$\sin\theta d\theta d\phi = d\omega \tag{6.1}$$

Now, assuming we have a 2D BRDF grid in spherical coordinate system θ, ϕ. The actual importance sampling PDF for any of the point on the grid should be

$$pdf(\theta, \phi) = pdf(\omega) * \sin\theta = BRDF_{\perp} * \sin\theta \tag{6.2}$$

The multiplication of $\sin\theta$ will make pdf at the locations that are very close to $\theta = 0$ very small. As a result, the most importance sampled part in a 2D BRDF grid is not the part where BRDF is the highest, but the part where $BRDf_{\perp} * \sin\theta$ is the highest.

For example, here is a visualization of a slice of the pdf of chm_orange generated using the method above, the parameterization of θ, ϕ follows what is described in section 4. We only show the first thirty columns since other columns are nearly zero We can clearly see that it is not the first few columns, who should have the highest cosine weighted BRDF, that has the highest pdf.

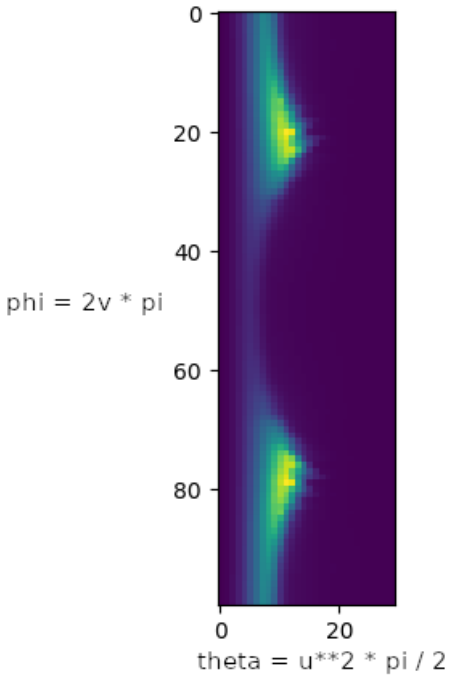


Figure 6.2: A 2D pdf grid for chm_orange

So we argue that the BRDF value of these parts are even more important than the θ-close-to-zero area.

6.2.2 Higher BRDF Value in the Most Importance Sampled Part

With this in mind, we investigate the BRDF grid generated from fitting VNDF. Although the BRDF has a close match with the close-to-zero area, it is generally a few times larger than ground truth in this most importance sampled area. We visualize the BRDF by selecting the most importance columns. The upper half of the image is the ground truth BRDF and the lower half is the fitted BRDF. We can see that it is generally higher. This is the result that why the renderer is brighter in the fitted VNDF method

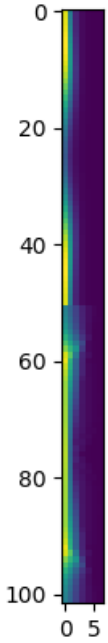


Figure 6.3: Comparison of most importance sampled part

6.3 Noisy Grazing Angles

In the renderer, a pixel value is computed by doing Monte Carlo Estimation

$$I = \frac{1}{N} \sum \frac{T(x)}{p(x)} \tag{6.3}$$

where T is the path throughput and p is the pdf of this path. So, what really matters in this quotient which we refer as importance sampling weight.

We have visualize a few slices of importance sampling weight using the fitted VNDF method. The first figure is a non-grazing angle case, and the second figure is a grazing angle case. In both figures we can see that the area where θ_h is very large has high weight. This means the pdf itself is not perfect. But for grazing angle case, the weight in the area where θ_h is most importance sampled is low, leaving only the above mentioned area to have high weight. I believe that the grazing angle noises

(a) weight where θ_i is small (b) weight where θ_i is large

Figure 6.4: Comparison on importance sampling weight

come from this. And the experiment result that using normalizing flow pdf reduces grazing angle noises also supports this guess.

6.4 Bad PDF Causing Information Lost

We have another example showing that the fitted pdf is not good enough. If we change the step 1 of generating BRDF in 4.2.3 to

- use (v, u) values with uniform interval to sample from VNDF grid, we will get another (v, u) pair following the vndf distribution and we convert it to ω_h using the inverse of the rule mentioned in VNDF grid generation

i.e. we generate a non-uniform BRDF grid according to our fitted VNDF. The BRDF grid will have more values in the area where the VNDF pdf is high.

The rendering result of reference material Figure 5.5a will become black

This generally means that our fitted pdf failed to capture all the information of the pdf. For this material, the non-uniform BRDF grid generated using pdf fails to capture the blue color which is slightly off the specular direction.

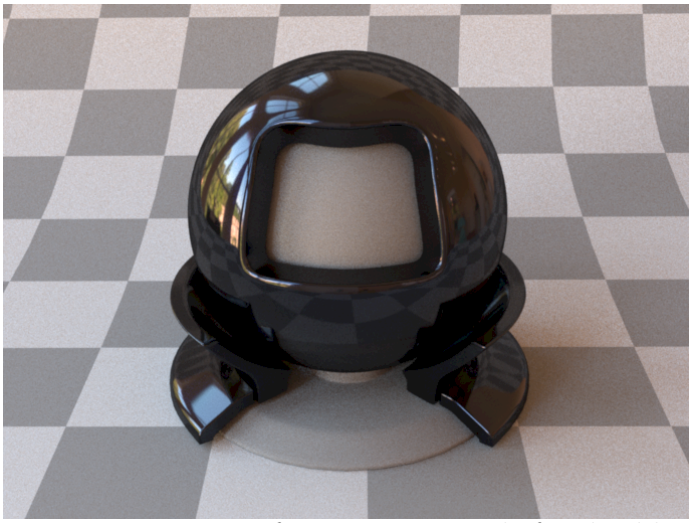


Figure 6.5: Non-uniform BRDF result for ilm_blue

6.4.1 Not able to importance sample three channels at the same time

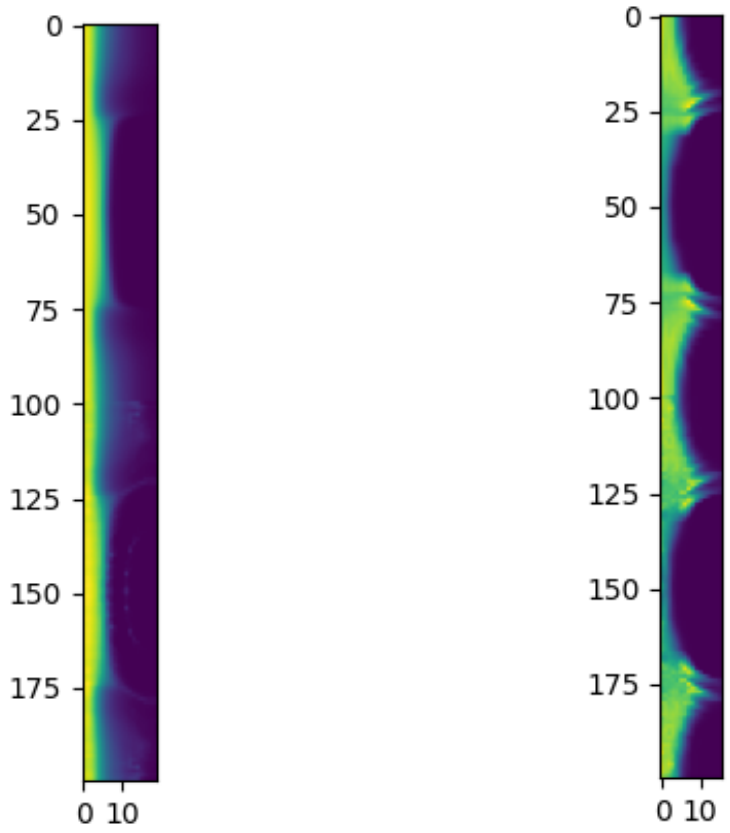
Another reason is that, in MCMC, we are only able to importance sampling according to one channel, we might be able to capture the information of that channel but it is possible that other channels are different from the one we are importance sampling.

6.5 Normalizing Flows

After switching to normalizing flows, we have had some improvements. We successfully reconstruct some channel, but we still face a few problems preventing us from getting a good rendered image.

6.5.1 More accuracy in BRDF fitting

In general, the post-acquisition optimization process for normalizing flows will give us a much more accurate fit to the BRDF. Here we show two BRDF slices at near grazing angles. The left one is fitted in half-diff space. The right one is fitted in ω_i, ω_o space and the first three columns are cropped. The upper half of both figures are ground truth, the lower half are the fitted BRDF. We only show the first 20 columns as the others are close to zero. We think that the normalizing flows fit will solve



(a) BRDF fitted by NF in half-diff space (b) BRDF fitted by NF in ω_i, ω_o space

Figure 6.6: BRDF fitting result using normalizing flows

the brighter problem in previous method as it will give us good fit at the previous problematic *most importance sampled* location.

Two spaces to importance sampling in

As we discussed in the previous chapter, there are two spaces that normalizing flows can learn and importance sample in, the $(\theta_h, \theta_d, \phi_d)$ space and the (ω_i, ω_o) space. We compared the BRDF optimization result of these two spaces. The $(\theta_h, \theta_d, \phi_d)$ space will give us more accurate fit for θ_h that is close to zero, which means the first few

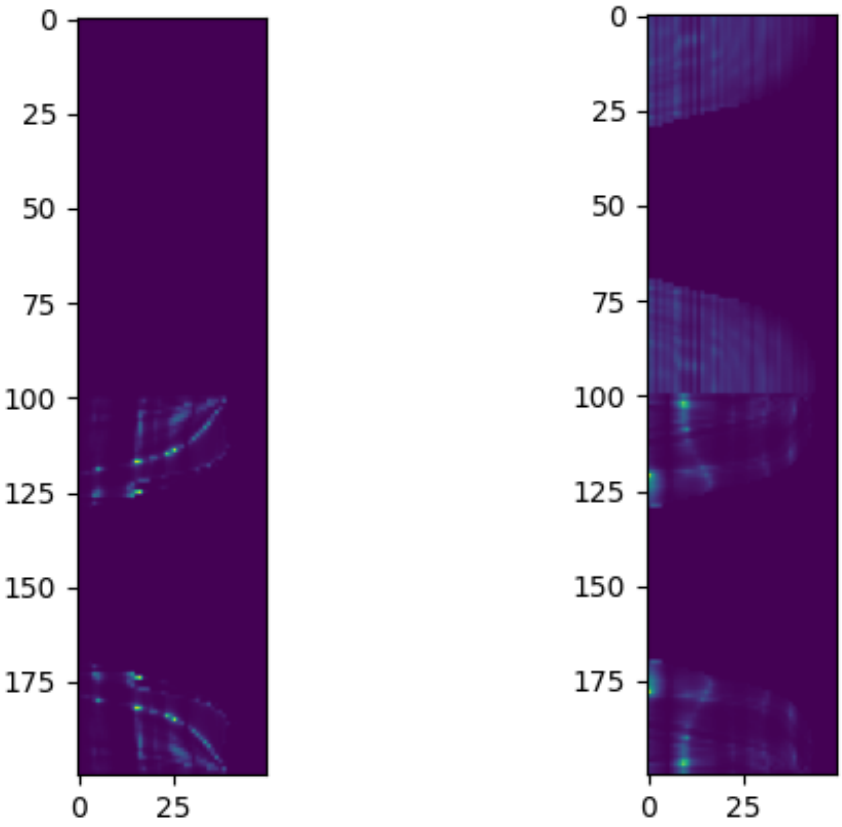
columns in our 2D BRDF grid. The (ω_i, ω_o) space will give us a more accurate fit in the most importance sampled part described in 6.2 while it will not have good fit for the first few columns simply because it does not have samples there. But this is expected because of the multiplication of $\sin(\theta)$ as Jacobian. We think that we should focus more on the most importance sampled part, but the current (ω_i, ω_o) space method using equation 4.8 to compute BRDF suffers from the problem of being divided by a denominator that is close to zero. This will make the BRDF value of any location that has either θ_h or θ_d close to zero very large. Though we do not find those values impact the rendering result, because they will have almost zero chance to be importance sampled in a renderer.

6.5.2 Instability during MCMC

One problem we noticed is that the adaptive MCMC chain is not stable during the MCMC process. We monitor the max likelihood of the samples and the acceptance rate. Sometimes the batch of samples we generate will have a very small likelihood. This means that we are somehow trapped in a location where BRDF and pdf are both small. Thus it is very easy for new samples to be rejected because the factor $\frac{BRDf}{pdf}$, which is used to compute acceptance probability of this location, is not small. After the normalizing flows being trained by those 'bad' samples, it produces worse samples. Sometimes the flows are able to jump out of this and return to normal operation. Sometimes this will act like a chain reaction and eventually break the flows and the chain.

6.5.3 Interpolation Failure in the area where no sample exists

Although the normalizing flows generally gives a better BRDF approximation than our previous method, it might fail for some channels. We think this is because the lack of samples in the area where BRDF is small. If we look at the sample distributions



(a) Last 50 cols of BRDF(green channel) (b) Last 50 cols of BRDF(red channel)

Figure 6.7: Last 50 cols of BRDF fitting result using normalizing flows

at the beginning of this section, we will see that normalizing flows are the '*most importance sampled*' method. This means the flows will have nearly no information at the locations where there are no samples. Although we've mentioned that these locations might be automatically optimized because normalizing flows must integrate to 1, it is still possible that they are not optimized well. We visualize the last 50 columns(where θ_h is large) of our BRDF slice and compare it to the ground truth.

These two are relative figures. The color depends on the global max/min value. We can still see that although there are some spikes in red channel BRDF, the range

of fitted red channel is close to ground truth(they both have visible colors in the comparison graph). But for the green channel, the fitted channel is so high that the ground truth values are not visible in the relative graph. This is one reason that causes the failure of rendering green and, similarly, blue channels.

7

Discussions

In this section we will briefly conclude the results and discuss what are the possible future improvements.

7.1 Conclusion

Our methods work well for rough materials, but face serious problems for specular materials. The use of normalizing flows has the potential to solve these problems but further studies are needed. Currently the biggest problem is not in the acquisition part but the post-acquisition processing pipeline.

7.2 Future Improvements

Now we discuss the directions that we will try to improve the current pipeline

7.2.1 Better Fitting Models and Algorithm

We have concluded that the VNDF pdf is not good enough to represent many materials. We might want to find a better model, though we tried the ABC model and it does not work for many materials. There are a lot of researches on the optimization

metrics, models. We might want to explore more to see if there is a good one that suits our application

7.2.2 Enforce Physics-based Constraints in Normalizing Flows

Normalizing flows are just neural networks, so it can be optimized to any state. It does not have to follow any existing physics-based constraints for BRDF(e.g. the microfacet model). Currently, our approach is to pretrain the normalizing flows with a GGX microfacet material. This is applying a physics-based constraint at the beginning, but after that the normalizing flows will be optimized to any possible state. It is still an open question on how to do this.

More Complex Pretraining

Like the iBRDF[20] and other works using neural network to express BRDF[18], one way to enforce physics-based constraints is to pretrain the flows on a collections of other materials, and only optimizing the 'embedded code' of the new material. That means the parameters inside the neural network will not be optimized, only the input(embedded code) will be optimized to find a complex combination of those pretrained materials. Thus, it guarantees the normalization flows keep the knowledge of all pretrained physics-based materials. However, this approach might limit the expressiveness of the flows, as the new BRDF will only be a combination of the pretrained one. Especially if we want to expand our method to anisotropic material.

7.2.3 Tuning Normalizing Flows

The NIS paper suggest to encode the inputs of normalizing flows using one-blob-encoding[14]. It is also possible to encode directions using spherical harmonics[33][34]. Normalizing flows are also able to receive auxiliary inputs that might be helpful for it to learn, we might want to explore what could be the auxiliary inputs in our case.

7.2.4 Separate the Sampling Process of Incident and Outgoing Directions

There is always a mismatch between how rendering software works and how our pipeline acquire samples. The rendering software always works in 2D domain, the ω_i is always given while our pipeline works in 4D(3D if isotropic) domain, perturbing both ω_i and ω_o. This mismatch is also the main reason that we need to come up with good interpolation method. We might try to also fix ω_i, do perturbation on ω_h or ω_o and generate slices directly to compare the results.

Conditional Normalizing Flows

We briefly investigate if there is conditional normalizing flows[35] that enable one to fix some parameter and sample others. But we did not find solid works on this. If conditional normalizing flows are possible, we can do this incident/outgoing direction separation easily. Otherwise, we need to use a brand new normalizing flows for each ω_i

7.2.5 Combination of two methods

We have two kinds of methods, the normalizing flows one and the fitting one. The normalizing flows can achieve accurate result in some channel but fail others, the fitting one will not have 'catastrophic' failure on certain channels. What can we do to combine those two to take the advantages of both methods.

7.2.6 Better Designed Experiments

Since we are doing two things: the acquisition and the post-acquisition processing, it is hard for us to directly compare our methods. To test if our samples are good, we need to complete post-acquisition processing to see the rendering result. But the result might be bad simply because our post-acquisition processing part is bad. For example, we can not compare the normalizing flows acquisition and the VNDF

acquisition process because they are different not only in the acquisition part but also in the post-acquisition part. A better designed experiment pipeline that controls variables so that we can directly compare certain method is needed if we want to investigate deeper.

Bibliography

[1] A. Ngan, F. Durand, and W. Matusik, "Experimental analysis of BRDF models," in *Proceedings of the Sixteenth Eurographics conference on Rendering Techniques*, ser. EGSR '05. Goslar, DEU: Eurographics Association, Jun. 2005, pp. 117–126, [Accessed 2024-04-27].

[2] J. Löw, J. Kronander, A. Ynnerman, and J. Unger, "BRDF models for accurate and efficient rendering of glossy surfaces," *ACM Transactions on Graphics*, vol. 31, no. 1, pp. 9:1–9:14, Feb. 2012. [Online]. Available: https://doi.org/10.1145/2077341.2077350. [Accessed 2024-04-27].

[3] T. Sun, H. W. Jensen, and R. Ramamoorthi, "Connecting measured BRDFs to analytic BRDFs by data-driven diffuse-specular separation," *ACM Transactions on Graphics*, vol. 37, no. 6, pp. 273:1–273:15, Dec. 2018. [Online]. Available: https://doi.org/10.1145/3272127.3275026. [Accessed 2024-04-27].

[4] J. B. Nielsen, H. W. Jensen, and R. Ramamoorthi, "On optimal, minimal BRDF sampling for reflectance acquisition," *ACM Transactions on Graphics*, vol. 34, no. 6, pp. 186:1–186:11, Nov. 2015. [Online]. Available: https://doi.org/10.1145/2816795.2818085. [Accessed 2024-04-27].

[5] J. Dupuy and W. Jakob, "An adaptive parameterization for efficient material acquisition and rendering," *ACM Transactions on Graphics*, vol. 37, no. 6, pp. 274:1–274:14, Dec. 2018. [Online]. Available: https://doi.org/10.1145/3272127.3275059. [Accessed 2024-04-27].

[6] W. Matusik, H. Pfister, M. Brand, and L. McMillan, "A data-driven reflectance model," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 759–769, Jul. 2003. [Online]. Available: https://dl.acm.org/doi/10.1145/882262.882343. [Accessed 2024-04-27].

[7] E. Veach and L. J. Guibas, "Metropolis light transport," in *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '97. USA: ACM Press/Addison-Wesley Publishing Co., Aug. 1997,

pp. 65–76. [Online]. Available: https://dl.acm.org/doi/10.1145/258734.258775. [Accessed 2024-04-27].

[8] C. Kelemen, L. Szirmay-Kalos, G. Antal, and F. Csonka, "A Simple and Robust Mutation Strategy for the Metropolis Light Transport Algorithm," *Computer Graphics Forum*, vol. 21, no. 3, pp. 531–540, 2002, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/1467-8659.t01-1-00703. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/1467-8659.t01-1-00703. [Accessed 2024-04-27].

[9] R. L. Cook and K. E. Torrance, "A Reflectance Model for Computer Graphics," *ACM Transactions on Graphics*, vol. 1, no. 1, pp. 7–24, Jan. 1982. [Online]. Available: https://dl.acm.org/doi/10.1145/357290.357293. [Accessed 2024-04-27].

[10] B. Walter, S. R. Marschner, H. Li, and K. E. Torrance, "Microfacet models for refraction through rough surfaces," in *Proceedings of the 18th Eurographics conference on Rendering Techniques*, ser. EGSR'07. Goslar, DEU: Eurographics Association, Jun. 2007, pp. 195–206, [Accessed 2024-04-27].

[11] E. Heitz and E. d'Eon, "Importance Sampling Microfacet-Based BSDFs using the Distribution of Visible Normals," *Computer Graphics Forum*, vol. 33, no. 4, pp. 103–112, 2014, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.12417. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12417. [Accessed 2024-04-27].

[12] E. Heitz, "Understanding the masking-shadowing function in microfacet-based brdfs," *Journal of Computer Graphics Techniques (JCGT)*, vol. 3, no. 2, pp. 48–107, June 2014. [Online]. Available: http://jcgt.org/published/0003/02/03/.

[13] P. Beckmann and A. Spizzichino, *The scattering of electromagnetic waves from rough surfaces*, Jan. 1987, publication Title: Norwood ADS Bibcode: 1987ah...book.....B. [Online]. Available: https://ui.adsabs.harvard.edu/abs/1987ah...book.....B. [Accessed 2024-04-27].

[14] T. Müller, B. Mcwilliams, F. Rousselle, M. Gross, and J. Novák, "Neural Importance Sampling," *ACM Transactions on Graphics*, vol. 38, no. 5, pp. 145:1–145:19, Oct. 2019. [Online]. Available: https://doi.org/10.1145/3341156. [Accessed 2024-04-27].

[15] Q. Zheng and M. Zwicker, "Learning to Importance Sample in Primary Sample Space," *Computer Graphics Forum*, vol. 38, no. 2, pp. 169–179, 2019, _eprint:

https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13628. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13628. [Accessed 2024-04-27].

[16] L. Dinh, D. Krueger, and Y. Bengio, "NICE: Non-linear Independent Components Estimation," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1410.8516. [Accessed 2024-04-27].

[17] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using Real NVP," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. [Online]. Available: https://openreview.net/forum?id=HkpbnH9lx. [Accessed 2024-04-27].

[18] C. Zheng, R. Zheng, R. Wang, S. Zhao, and H. Bao, "A Compact Representation of Measured BRDFs Using Neural Processes," *ACM Transactions on Graphics*, vol. 41, no. 2, pp. 14:1–14:15, Nov. 2021. [Online]. Available: https://doi.org/10.1145/3490385. [Accessed 2024-04-27].

[19] B. Xu, L. Wu, M. Hasan, F. Luan, I. Georgiev, Z. Xu, and R. Ramamoorthi, "NeuSample: Importance Sampling for Neural Materials," in *ACM SIGGRAPH 2023 Conference Proceedings*, ser. SIGGRAPH '23. New York, NY, USA: Association for Computing Machinery, Jul. 2023, pp. 1–10. [Online]. Available: https://dl.acm.org/doi/10.1145/3588432.3591524. [Accessed 2024-04-27].

[20] Z. Chen, S. Nobuhara, and K. Nishino, "Invertible Neural BRDF for Object Inverse Rendering," in *Computer Vision – ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham: Springer International Publishing, 2020, pp. 767–783.

[21] S. M. Rusinkiewicz, "A New Change of Variables for Efficient BRDF Representation," in *Rendering Techniques '98*, G. Drettakis and N. Max, Eds. Vienna: Springer, 1998, pp. 11–22.

[22] F. Romeiro and T. Zickler, "Inferring Reflectance under Real world Illumination," 2010, accepted: 2016-01-28T18:25:55Z. [Online]. Available: https://dash.harvard.edu/handle/1/24947961. [Accessed 2024-04-27].

[23] E. Heitz, "Sampling the ggx distribution of visible normals," *Journal of Computer Graphics Techniques (JCGT)*, vol. 7, no. 4, pp. 1–13, November 2018. [Online]. Available: http://jcgt.org/published/0007/04/01/.

[24] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, "Neural Spline Flows," in *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://papers.nips.cc/paper_files/paper/2019/hash/7ac71d433f282034e088473244df8c02-Abstract.html. [Accessed 2024-04-27].

[25] V. Stimper, D. Liu, A. Campbell, V. Berenz, L. Ryll, B. Schölkopf, and J. M. Hernández-Lobato, "normflows: A PyTorch Package for Normalizing Flows," *Journal of Open Source Software*, vol. 8, no. 86, p. 5361, Jun. 2023. [Online]. Available: https://joss.theoj.org/papers/10.21105/joss.05361. [Accessed 2024-04-27].

[26] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, "Normalizing flows for probabilistic modeling and inference," *The Journal of Machine Learning Research*, vol. 22, no. 1, pp. 57:2617–57:2680, Jan. 2021.

[27] I. Kobyzev, S. J. Prince, and M. A. Brubaker, "Normalizing Flows: An Introduction and Review of Current Methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 11, pp. 3964–3979, Nov. 2021, conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence. [Online]. Available: https://ieeexplore.ieee.org/document/9089305. [Accessed 2024-04-25].

[28] G. O. Roberts and J. S. Rosenthal, "Coupling and Ergodicity of Adaptive Markov Chain Monte Carlo Algorithms," *Journal of Applied Probability*, vol. 44, no. 2, pp. 458–475, 2007, publisher: Applied Probability Trust. [Online]. Available: https://www.jstor.org/stable/27595854. [Accessed 2024-04-27].

[29] F. Luan, S. Zhao, K. Bala, and I. Gkioulekas, "Langevin monte carlo rendering with gradient-based adaptation," *ACM Transactions on Graphics*, vol. 39, no. 4, pp. 140:140:1–140:140:16, Aug. 2020. [Online]. Available: https://doi.org/10.1145/3386569.3392382. [Accessed 2024-04-27].

[30] A. Fores, J. Ferwerda, and J. Gu, "Toward a perceptually based metric for BRDF modeling," in *Color and Imaging Conference*, vol. 20, pp. 142–148, ISSN: 2166-9635 Issue: 1 Journal Abbreviation: cic. [Online]. Available: https://library.imaging.org/cic/articles/20/1/art00025. [Accessed 2023-11-08].

[31] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and

Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6980. [Accessed 2024-04-27].

[32] W. Jakob, "Mitsuba renderer," 2010, http://www.mitsuba-renderer.org.

[33] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Transactions on Graphics*, vol. 41, no. 4, pp. 102:1–102:15, Jul. 2022. [Online]. Available: https://dl.acm.org/doi/10.1145/3528223.3530127. [Accessed 2024-04-27].

[34] D. Verbin, P. Hedman, B. Mildenhall, T. Zickler, J. T. Barron, and P. P. Srinivasan, "Ref-NeRF: Structured View-Dependent Appearance for Neural Radiance Fields," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2022, pp. 5481–5490, iSSN: 2575-7075. [Online]. Available: https://ieeexplore.ieee.org/document/9879796. [Accessed 2024-04-27].

[35] C. Winkler, D. Worrall, E. Hoogeboom, and M. Welling, "Learning Likelihoods with Conditional Normalizing Flows," Nov. 2023, arXiv:1912.00042 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1912.00042. [Accessed 2024-04-27].