Diffraction Shader

Michael Single

14. November 2013

Inhaltsverzeichnis

1 Introduction

1.1 Motivation

effect of diffraction, stam, genf, rendering snake skin

Introducation bla In phicss/bioligy

The purpose of this thesis is to render realtime the effect of diffraction on different snakes skins in a photorealistic manor. In oder to achieve this purpose we will rely J. Stam's formulation of a BRDF which basically describes the effect of diffraction on a given surface assuming one knows the hightfield on this surface. In our case, those heightfields are small patches of the nanostracture of the snake skin provided by GENEVA taken by MIKROSKOP. In his Paper, J. Stam assuming distribution on his heightfields whereas we require a an explicit provided hightfield of the surface or at least a small patch. Therefore, this work can be considered as an extension of J. Stam's derivations for the case one is provided by a explicit height field on a quasiperiodic structure. Since one goal of this work is to render in realtime, we have to perform also precomputations which will require us to slightly modify Stam's main derivation.

1.2 Related Work

see papaer listing

1.3 Thesis Outline

describe what is which chapter

2 Theoretical Background

Explain that this thesis has deep theoretical background, some derivations show derivation roadmap

2.1 The Effect Of Diffraction

2.2 BRDF - Spectral Rendering

2.3 Stams derivation

In his Paper Diffraction Shader, Jos Stam derives a an BRDF modeling the effect of diffraction for various analytical anistropic reflaction models using the scalar Kirchof theory and the theory of random processes. By emplyong the so called wave theory of diffraction [source 5 in stams paper] in which a wave is assumed to be a complex valued scalar. It's noteworthy, that stam's BRDF formulation does not take into account the polarization of the light. Nevertheless, light sources like sunlight and light bulbs are unpilarizaed. In our simulations we will always assume we have given i directional light source, i.e. sunlight. Hence, we can use stam's model for our derivations

A further assumption in Stam's Paper is, the emanated waves from the source are stationary - sunlight once again. Which implies the wave is a superposition of independent monochromatic waves. This implies that each wave is associated to a definite wavelangth lambda.

Mention Helmolth equation, which has the solution $k = \frac{2\pi}{\lambda}$ which is the wavenumber

Stams starts his derviations by above's assumptions and by applying the Kirchhoff integral, which descirbes the reflected field and the Huygen's principle, which states, when somebody knows the wavefront at a given moment, the wave at a later time can be deducted by considering each point on the first wave as the source of a new disturbance.

$$\psi_2 = \frac{ike^{iKR}}{4\pi R}(F\mathbf{v} - \mathbf{p}) \cdot \int_S \hat{\mathbf{n}} e^{ik\mathbf{v}\cdot\mathbf{s}d\mathbf{s}} \tag{1}$$

In optics, when dealing with scattered waves, one does use differential scattering cross-section rather than a BRDF which has the following identitiy:

$$\sigma^0 = 4\pi \lim_{R\to\infty} R^2 \frac{\langle|\psi_2|^2\rangle}{\langle|\psi_1|^2\rangle} \tag{2}$$

Relationship between the BRDF and the scattering cross section is the follwing:

$$BRDF = \frac{1}{4\pi}\frac{1}{A}\frac{\sigma^0}{cos(\theta_1)cos(\theta_2)} \tag{3}$$

Wheras $\theta_1 a$ and θ_2 are the angles that the vectors $\hat{k_1}$ and $\hat{k_2}$ make with the vertical direction.

ADD FIGURE for k1, k2

where R is the disance from the center of the patch to the receiving point x_p, $\hat{\mathbf{n}}$ is the normal of the surface at s and the vectors:

$$\mathbf{v} = \hat{\mathbf{k_1}} - \hat{\mathbf{k_1}} = (u, v, w)$$

$$\mathbf{p} = \hat{\mathbf{k_1}} + \hat{\mathbf{k_1}}$$

During his derivations, Stam provides a analytical representation for the Kirchhoff integral by using his assumptions. He restricts himself to the reflaction of waves from height fields $h(x, y)$ with the assumption that the surface is defined as an elevation over the (x,y) plane using the surface plane approximation. Which will lead him to the follwoing identity for the Kirchhoff integral

$$\mathbf{I}(ku, kv) = \int\int \frac{1}{ikw}(-p_x, -p_y, ikwp) \tag{4}$$

wheras

$$p(x, y) = e^{ikwh(x,y)} \tag{5}$$

We the observation that the integral is a Fourier transform by $-iku$ and $-ikv$ which will lead us to his final derivation, using the identity of BRDF, and computing the limes:

$$BRDF = \frac{k^2F^2G}{4\pi^2Aw^2}\langle|P(ku, kv)|^2\rangle \tag{6}$$

Where

$$G = \frac{(1 - \hat{\mathbf{k_1}} \cdot \hat{\mathbf{k_2}})^2}{cos(\theta_1)cos(\theta_2)} \tag{7}$$

and P(x,y) is the Fourier transform of the function p(x,y) from above. This identitiy for the BRDF is the starting point for our derivations.

2.4 Taylor Series Approximation

Based on J. Stam's Paper about Diffraction shaders we will show that there is an approximation of his equation (5), $\mathbf{p(x,y)}$, for a explicitly given heightfield $\mathbf{h(x,y)}$. This approximation is achieved by using Taylor-Series and using this identity we will further be able to approximate the Fourier-Transformation of p(x,y), denoted as $\mathbf{P(u,v)}$. Finally we will give an error bound for this approximation.

2.4.1 Taylor Series of p

Given $p(x,y) = e^{ikwh(x,y)}$form Stam's Paper where h(x,y) is here a given heightfield. Also given the definition $e^y = 1 + y + \frac{y^2}{2!} + \frac{y^3}{3!} + ... = \sum_{n=0}^{\infty} \frac{y^n}{n!}$where y can be real or even complex valued - note this identity can either be derieved by power series or by Taylor-Series(using the derivatives of the exp-function and developing the Taylor-Series around the point a=0). Let us now set $y = ikwh(x,y)$where i is the imaginary number. For simplification, let us denote h(x,y) as h. It follows by our previous stated identities: $e^y = 1 + (ikwh) + \frac{1}{2!}(ikwh)^2 + \frac{1}{3!}(ikwh)^3 + ... = \sum_{n=0}^{\infty} \frac{(ikwh)^n}{n!}$. Hence it holds $p(x,y) = \sum_{n=0}^{\infty} \frac{(ikwh(x,y))^n}{n!}$.

2.4.2 Fourier Transformation of function p

Let us now compute the Fourier Transformation of p(x,y) form above:$\mathcal{F}\left\{p\right\}(u,v) = \mathcal{F}\left\{\sum_{n=0}^{\infty} \frac{(ikwh(x,y))^n}{n!}.\right\} =^{\mathcal{F}\, lin\, Operator} \sum_{n=0}^{\infty} \mathcal{F}\left\{\frac{(ikwh(x,y))^n}{n!}\right\} = \sum_{n=0}^{\infty} \frac{(ikwh)^n}{n!} \mathcal{F}\left\{h(x,y)^n\right\}$. Hence it follows: $P(\alpha,\beta) = \sum_{n=0}^{\infty} \frac{(ikwh)^n}{n!} \mathcal{F}\left\{h^n\right\}(\alpha,\beta)$.

NB: $\mathcal{F}\left\{h^n\right\}(u,v)$ denotes the two dimensional Fourier Transformation of p(x,y) and can be nummerically computed by the two dimensional **DFT** or rather by the two dimensional **FFT** over h(x,y).

2.4.3 Approximation of function P

Next we are going to look for an $N \in \mathbb{N}$s.t. $\sum_{n=0}^{N} \frac{(ikwh)^n}{n!} \mathcal{F}\{h^n\}(\alpha, \beta) \approx P(\alpha, \beta)$. is a good approximation. We have to prove two things:

1. Show that there exist such an $N \in \mathbb{N}$s.t the approximation holds true.
2. Find a value for B s.t. this approximation is below a certain error bound, for example machine precision ϵ.

2.4.4 Proof Sketch of 1.

By the **ratio test** (see [**1**]) we can show that the series $\sum_{n=0}^{N} \frac{(ikwh)^n}{n!} \mathcal{F}\{h^n\}(\alpha, \beta)$ converges absolutely:

Proof: Consider $\sum_{k=0}^{\infty} \frac{y^n}{n!}$where $a_k = \frac{y^k}{k!}$. By the definition of the ratio test for series it follows: $\forall y : limsup_{k \to \infty} |\frac{a_{k+1}}{a_k}| = limsup_{k \to \infty} \frac{y}{k+1} = 0$

Thus this series converges absolutely, no matter what value we will pick for y.

2.4.5 Part 2: Find such an N

Let $f(x) = e^x$. We can formulate its Taylor-Series, stated above. Let $P_n(x)$denote the n-th Taylor-Polinomial, $P_n(x) = \sum_{k=0}^{n} \frac{f^{(k)}(a)}{k!}(x-a)^k$,where a is our developing point (here, in this case a=0). We can define the error of the n-th Taylor-Polinomial to be $E_n(x) = f(x) - P_n(x)$. That error is the actual value minus the Taylor polinomial. It holds true: $|E_n(x)| = |f(x) - P_n(x)|$. By using the Lagrangien Error Bound - (see source [**2**]) it follows: $|E_n(x)| \leq \frac{M}{(n+1)!}|x-a|^{n+1}$ with a=0, where **M** is some value satisfying $|f^{(n+1)}(x)| \leq M$ on the interval $I = [a, x]$.Since we are interested in an upper bound of the error and since **a** is known, we can reformulate the interval as $I = [0, x_{max}]$, where $x_{max} = |i| * k_{max} * w_{max} * h_{max}$, since we are interested in computing an error bound for $e^{ikwh(x,y)}$. From Stam's Paper about diffraction shader we know some paramters for the length, width and height for a given sample patch, i.e. heightfield h(x,y) and when using those parameters are able to find a explicit number for x_{max}.

Facts we are using from Stam's Paper:

- Height of bump: 0.15micro meters

- Width of a bump: 0.5micro meters
- Length of a bump: 1micro meters
- $k = \frac{2\pi}{\lambda}$ is the wavenumber and $\lambda \in [\lambda_{min,} \lambda_{max}]$its wavelength hence $k_{max} = \frac{2\pi}{\lambda_{min}}$
- w is a component of the vector $\vec{v} = \vec{k_1} - \vec{k_2} = (u, v, w)$, where $\vec{k_1}$ and $\vec{k_2}$ are **normalized** direction vectors and this each component can have a value in range [-2, 2].
- for simplification, assume$[\lambda_{min,} \lambda_{max}] = [400nm, 700nm]$.

Hence $x_{max} = |i| * k_{max} * w_{max} * h_{max} = k_{max} * w_{max} * h_{max} = 2 * (\frac{2\pi}{4*10^{-7}m}) * 1.5 * 10^{-7} = 1.5\pi$and it follows for our intervall $I = [0, 1.5\pi]$. Next we are going to find the value for M. Since the exponential function is monoton growing (on the interval I) and and the derivative of the **exp** function is the exp function itself, we can find such an M: $M = e^{x_{max}} = exp(1.5\pi)$and $|f^{(n+1)}(x)| \leq M$ holds. With $|E_n(x_{max})| \leq \frac{M}{(n+1)!}|x_{max} - a|^{n+1} = \frac{exp(1.5\pi)*(1.5\pi)^{n+1}}{(n+1)!}$we now can find a value of n for a given bound, i.e. we can find an value of $N \in \mathbb{N}$ s.t. $\frac{exp(1.5\pi)*(1.5\pi)^{N+1}}{(N+1)!} \leq \epsilon$. With Octave/Matlab we can see:

- if N=20 then $\epsilon \approx 2.9950 * 10^{-4}$
- if N=25 then $\epsilon \approx 8.8150 * 10^{-8}$
- if N=30 then $\epsilon \approx 1.0050 * 10^{-11}$

2.4.6 Conclusion

With this approach we have that$\sum_{n=0}^{25} \frac{(ikwh)^n}{n!} \mathcal{F}\{h^n\}(\alpha, \beta)$is an approximation of P(u,v) with error$\epsilon \approx 8.8150 * 10^{-8}$. This means we can precompute 25 Fourier Transformations (for example via FFT2) and then sum them up in order to approximate P(u,v) and $\epsilon \approx 8.8150 * 10^{-8}$. This approach will allow us to speed up our shader. Furthermore we see that when we just take 5 more iterations, we will reduce the error bound to the dimension of 10^{-11}.

NB: More explanation about how the shader itself works during our discussion.

2.4.7 Sources

- [**1**] http://en.wikipedia.org/wiki/Ratio_test
- [**2**] http://math.jasonbhill.com/courses/fall-2010-math-2300-005/lectures/taylor-polynomial-error-bounds

2.5 Our derivations

EXPLAIN: Why do we want a formulation for $L_\lambda(w_r)$ in some words. what does it represent?

Definition of $BRDF(w_i, w_r) := f_r(w_i, w_r) = \frac{dL_r(w_r)}{dE_i(w_i)} = \frac{dL_r(w_r)}{L_i(w_i)cos(\theta_i)dw_i}$ Hence, we can dervie the following expression:

$$\begin{aligned} f_r(w_i, w_r) &= \frac{dL_r(w_r)}{L_i(w_i)cos(\theta_i)dw_i} \\ => f_r(w_i, w_r)L_i(w_i)cos(\theta_i)dw_i &= dL_r(w_r) \\ => \int_\Omega f_r(w_i, w_r)L_i(w_i)cos(\theta_i)dw_i &= \int_\Omega dL_r(w_r) \\ => L_r(w_r) &= \int_\Omega f_r(w_i, w_r)L_i(w_i)cos(\theta_i)dw_i \end{aligned}$$

We assume, that our incident light is a directional light source like sunlight and therefore its radiance is given as $L_\lambda(w) = I(\lambda)\delta(w - w_i)$ where $I(\lambda)$ is the intensity of the relative spectral power for the wavelength λ. Thus we get for our the brdf formulation:

$$L_\lambda(w_r) = \int_\Omega BRDF_\lambda(w_i, w_r)L_\lambda(w_i)cos(\theta_i)dw_i \tag{8}$$

$$= BRDF_\lambda(w_i, w_r)I(\lambda)cos(\theta_i) \tag{9}$$

where w_i is the solid angle for the incoming light, θ_i is the angle of incidence, w_r is the solid angle for the reflected light, λ wavelength, Ω is the hemisphre we of integration for the incomming light. Radiance reflected by given surface in given direction: $L_\lambda(w_i)$ is the incomming radiance, $L_\lambda(w_r)$ is the reflected radiance

For the $BRDF(w_i, w_r)$ we are going to use the formulation dervied by Stam described above which looks like this using the fact that wavenumber $k = \frac{2\pi}{\lambda}$:

$$
\begin{aligned}
BRDF(w_i, w_r) &= \frac{k^2F^2G}{4\pi^2Aw^2}\langle|P(ku,kv)|^2\rangle \\
&= \frac{k^2F^2(1-\hat{\mathbf{k_1}}\cdot\hat{\mathbf{k_2}})}{cos(\theta_1)cos(\theta_2)4\pi^2Aw^2}\langle|P(ku,kv)|^2\rangle \\
&= \frac{4\pi^2F^2(1-\hat{\mathbf{k_1}}\cdot\hat{\mathbf{k_2}})}{cos(\theta_1)cos(\theta_2)4\pi^2A\lambda^2w^2}\langle|P(ku,kv)|^2\rangle \\
&= \frac{F(w_i,w_r)^2(1-\hat{\mathbf{k_1}}\cdot\hat{\mathbf{k_2}})}{cos(\theta_1)cos(\theta_2)A\lambda^2w^2}\langle|P(ku,kv)|^2\rangle
\end{aligned}
$$

where $\hat{\mathbf{k_t}}$ represents a unit vector whose spherical coordinates are given by the solid angle t. Since we are going to integrate over a sphere Ω we can write the component $w = (cos(\theta_i) + cos(\theta_r))$ SHOW WHY WE ARE ALLOWED TO WRITE IT LIKE THIS => SPHERICAL COORDINATES DIFFERENCE $(k1 - k2) = (u, v, w)$ and so on. this our identity for $L_r(w_r)$ will lead us to the following identiy using our identity :

$$
\begin{aligned}
L_\lambda(w_r) = &\frac{F(w_i,w_r)^2(1-\hat{\mathbf{k_1}}\cdot\hat{\mathbf{k_2}})^2}{A\lambda^2cos(\theta_i)cos(\theta_r)(cos(\theta_i)+cos(\theta_r))^2}\langle\left|P_{cont}(\frac{2\pi u}{\lambda},\frac{2\pi v}{\lambda})\right|^2\rangle cos(\theta_i)I(\lambda) \\
&= I(\lambda)\frac{F(w_i,w_r)^2(1-\hat{\mathbf{k_1}}\cdot\hat{\mathbf{k_2}})^2}{\lambda^2A(cos(\theta_i)+cos(\theta_r))^2cos(\theta_r)}\langle\left|P_{cont}(\frac{2\pi u}{\lambda},\frac{2\pi v}{\lambda})\right|^2\rangle \\
&= I(\lambda)\frac{F(w_i,w_r)^2(1-\hat{\mathbf{k_1}}\cdot\hat{\mathbf{k_2}})^2}{\lambda^2A(cos(\theta_i)+cos(\theta_r))^2cos(\theta_r)}\langle\left|T_0^2P_{dtft}(\frac{2\pi u}{\lambda},\frac{2\pi v}{\lambda})\right|^2\rangle
\end{aligned}
$$

P_{cont} is the continious inverse Fourier transform for the taylor seies of our hight-field representing the nano structure, i.e. $P(k,l) = \mathcal{F}^{-1}\{p\}(k,l)$ and P_{dtft} is the dicrete-time inverse Fourier Transform for the same problem domain and T_0 the sampling distance for the discretization pf $p(x,y)$ assuming equal and uniform sampling in both dimensions x, y.

2.6 Relative BRDF

reason why relative brdf: In order to scale the reflactiance such that we are able to texture. convex combination reflectance with texture. Scale illumination.

Let us examine what $L_\lambda(w_r)$ will be for $w_r = w_0 := (0,0,*)$ i.e. specular reflection case, denoted as $L_\lambda^{spec}(w_0)$. When we know the expression for $L_\lambda^{spec}(w_0)$ we would be able to compute the relative reflected radiance for our problem by simply dividing $L_\lambda(w_r)$ by $L_\lambda^{spec}(w_0)$, denoted as

$$\rho_\lambda(w_i, w_r) = \frac{L_\lambda(w_r)}{L_\lambda^{spec}(w_0)} \tag{10}$$

But first, let us derive the following expression:

$$\begin{aligned}
L_\lambda^{spec}(w_0) &= I(\lambda)\frac{F(w_0,w_0)^2(1 - \begin{pmatrix}0\\0\\1\end{pmatrix} \cdot \begin{pmatrix}0\\0\\-1\end{pmatrix})^2}{\lambda^2 A(cos(0)+cos(0))^2 cos(0)} \langle \left| T_0^2 P_{dtft}(0,0)\right|^2 \rangle \\
&= I(\lambda)\frac{F(w_0,w_0)^2(1+1)^2}{\lambda^2 A(1+1)^2 1}\left|T_0^2 N_{sample}\right|^2 \\
&= I(\lambda)\frac{F(w_0,w_0)^2}{\lambda^2 A}\left|T_0^2 N_{sample}\right|^2
\end{aligned}$$

Where $N_{samples}$ is the number of samples of the dtft.

Thus, we can plug our last derived expression into the definition for the relative reflectance radiance in the direction w_r and will get:

$$\begin{aligned}
\rho_\lambda(w_i, w_r) &= \frac{L_\lambda(w_r)}{L_\lambda^{spec}(w_0)} \\
&= \frac{I(\lambda)\frac{F(w_i,w_r)^2(1-\hat{\mathbf{k_1}}\cdot\hat{\mathbf{k_2}})^2}{\lambda^2 A(cos(\theta_i)+cos(\theta_r))^2 cos(\theta_r)}\langle \left|T_0^2 P_{dtft}(\frac{2\pi u}{\lambda}, \frac{2\pi v}{\lambda})\right|^2\rangle}{I(\lambda)\frac{F(w_0,w_0)^2}{\lambda^2 A}\left|T_0^2 N_{sample}\right|^2} \\
&= \frac{F^2(w_i,w_r)(1-\hat{\mathbf{k_1}}\cdot\hat{\mathbf{k_2}})^2}{F^2(w_0,w_0)(cos(\theta_i)+cos(\theta_r))^2 cos(\theta_r)}\langle \left|\frac{P_{dtft}(\frac{2\pi u}{\lambda}, \frac{2\pi v}{\lambda})}{N_{samples}}\right|^2\rangle
\end{aligned}$$

for simplification and a better overview, let us introduce the following expression, the so called gain factor

$$C(w_i, w_r) = \frac{F^2(w_i,w_r)(1-\hat{\mathbf{k_1}}\cdot\hat{\mathbf{k_2}})^2}{F^2(w_0,w_0)(cos(\theta_i)+cos(\theta_r))^2 cos(\theta_r) N_{samples}^2} \tag{11}$$

Using this substitute, we will end up with the following expression for the relative reflectance radiance

$$\rho_\lambda(w_i, w_r) = C(w_i, w_r)\langle\left|P_{dtft}(\frac{2\pi u}{\lambda}, \frac{2\pi v}{\lambda})\right|^2\rangle \tag{12}$$

using the previous definition for the relative reflectance radiance $\rho_\lambda(w_i, w_r) = \frac{L_\lambda(w_r)}{L_\lambda^{spec}(w_0)}$ which we can rearrange to the expression

$$L_\lambda(w_r) = \rho_\lambda(w_i, w_r) L_\lambda^{spec}(w_0) \tag{13}$$

Let us choose $L_\lambda^{spec}(w_0) = S(\lambda)$ such that is has the same pforifle as the relative spectral power distribuation of CIE Standard Illuminant $D65$. Further, when integration over λ for a specular surface we should get CIE_XYZ values corresponding to the white point for $D65$

the corresponding tristimulus values using CIE colormatching functions for the CIE_XYZ values look like:

SEE HOW THIS DEFINITION DIFFERS FROM THE WIKIDEF AND HOW WE COULD END UP WITH A SIMILAR DEFINITION.

$$X = \int_\lambda L_\lambda(w_r)\overline{x}(\lambda)d\lambda \tag{14}$$

$$Y = \int_\lambda L_\lambda(w_r)\overline{y}(\lambda)d\lambda \tag{15}$$

$$Z = \int_\lambda L_\lambda(w_r)\overline{z}(\lambda)d\lambda \tag{16}$$

where $\overline{x}$, $\overline{y}$, $\overline{z}$ are the color matching functions

Using our last finding for $L_\lambda(w_r)$ and the definition for the tristimulus values we can actually derive an expression for computing the colors for our brdf model. Since X, Y, Z are defined similarly, it satisfies to derive an explicit expression for just one tristimulus term, for example X. The other two will look the same, except the we have to replace all X with Y or Z respectively. Therefore, we get:

$$
\begin{aligned}
X &= \int_\lambda L_\lambda(w_r)\overline{x}(\lambda)d\lambda \\
&= \int_\lambda \rho_\lambda(w_i,w_r)L_\lambda^{spec}(w_0)\overline{x}(\lambda)d\lambda \\
&= \int_\lambda \rho_\lambda(w_i,w_r)S(\lambda)\overline{x}(\lambda)d\lambda \\
&= \int_\lambda C(w_i,w_r)\langle\left|P_{dtft}(\frac{2\pi u}{\lambda},\frac{2\pi v}{\lambda})\right|^2\rangle S(\lambda)\overline{x}(\lambda)d\lambda \\
&= C(w_i,w_r)\int_\lambda \langle\left|P_{dtft}(\frac{2\pi u}{\lambda},\frac{2\pi v}{\lambda})\right|^2\rangle S(\lambda)\overline{x}(\lambda)d\lambda \\
&= C(w_i,w_r)\int_\lambda \langle\left|P_{dtft}(\frac{2\pi u}{\lambda},\frac{2\pi v}{\lambda})\right|^2\rangle S_x(\lambda)d\lambda
\end{aligned}
$$

Where we used the definition $S_x(\lambda)\overline{x}(\lambda)$ in the last step.

2.7 Taylour approximation for BRDF

Using $P_{dtft} = \mathcal{F}^{-1}\{p\}(u,v)$ definied in the section of the taylor approximationwe get for the tristumulus value X, we will get:

$$
\begin{aligned}
X &= C(w_i,w_r)\int_\lambda \langle\left|P_{dtft}(\frac{2\pi u}{\lambda},\frac{2\pi v}{\lambda})\right|^2\rangle S_x(\lambda)d\lambda \\
&= C(w_i,w_r)\int_\lambda \left|\sum_{n=0}^{N}\frac{(wk)^n}{n!}\mathcal{F}^{-1}\{i^n h^n\}(\frac{2\pi u}{\lambda},\frac{2\pi v}{\lambda})\right|^2 S_x(\lambda)d\lambda
\end{aligned}
$$

2.8 Sampling: Gaussian Window

Let $window_g$ denote the gaussian window with $4\sigma_s$ μm where $\sigma_f = \frac{1}{2\pi\sigma_s}$ let us further substitute $\mathbf{t}(\mathbf{x},\mathbf{y}) = i^n h(x,y)^n$

$$
\mathcal{F}_{dtft}^{-1}\{\mathbf{t}\}(u,v) = \mathcal{F}_{fft}^{-1}\{\mathbf{t}\}(u,v)window_g(\sigma_f) \tag{17}
$$

Therefore we can deduce the following expression from this:

$$
\begin{aligned}
\mathcal{F}^{-1}_{dtft}\{\mathbf{t}\}(u,v) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F^{-1}_{fft}\{\mathbf{t}\}(w_u,w_v)\phi(u-w_u,v-w_v)dw_u dw_v \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\sum_i\sum_j F^{-1}_{fft}\{\mathbf{t}\}(w_u,w_v)\delta(w_u-w_i,w_v-w_j)\phi(u-w_u,v-w_v)dw_u dw_v \\
&= \sum_i\sum_j\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F^{-1}_{fft}\{\mathbf{t}\}(w_u,w_v)\delta(w_u-w_i,w_v-w_j)\phi(u-w_u,v-w_v)dw_u dw_v \\
&= \sum_i\sum_j F^{-1}_{fft}\{\mathbf{t}\}(w_u,w_v)\phi(u-w_u,v-w_v)
\end{aligned}
$$

where $\phi(x,y) = \pi e^{-\frac{x^2+y^2}{2\sigma_f^2}}$

2.9 Aplitude smooting

Let us consider the so called 1-dimensional Box-function with length T which is defined as the following: ADD AN IMAGE OF BOXFUNCTION

$Box(x) = \begin{cases} 1 & \text{if } x \leq T \\ 0 & \text{if } else \end{cases}$

We assume, that our given heighfield can be represented as a 2-dimensional box-function. Note that we can use any explicit given constrainted 2-dimensional function and will get some identities like we get from the box-function.

Further we are assuming that we can model the overall surface be assuming this heighfield being distributed in a periodic manor. Therfore, the whole surface can be represented like this $f(x) = \sum_{n=0}^{N} Box(x+nT_1, y+mT_2)$ assuming the given heighfield has the dimensions T_1 by T_2. But let us first consider the 1-dimensional Box-function case before deriving an identity for the Fourier transform of our 2-dimensional Box-function, i.e. the fourier transform of our heighfield.

Note: A function f periodic with periode T means: $\forall x \in \mathcal{R} : Box(x) = Box(x+T)$

A so called bump can be represented by our 1-dimensional Box-function. We assume periodicity which is equaivalent to: $f(x) = \sum_{n=0}^{N} Box(x+nT)$

We are insterested in the 1-dimensional inverse Fourier transform of the 1-dimensional Box-function:

$$\begin{aligned}\mathcal{F}^{-1}\{f\}(w) &= \int f(x)e^{iwx}dx \\ &= \int_{-\infty}^{\infty}\sum_{n=0}^{N} Box(x+nT)e^{iwx}dx \\ &= \sum_{n=0}^{N}\int_{-\infty}^{\infty} Box(x+nT)e^{iwx}dx\end{aligned}$$

Next, apply the following substituation $x + nT = y$ which will lead us to:

$$\begin{aligned}x &= y - nT \\ dx &= dy\end{aligned}$$

Plugging this substituation back to the equation from above we will get

$$\begin{aligned}\mathcal{F}^{-1}\{f\}(w) &= \int f(x)e^{iwx}dx \\ &= \sum_{n=0}^{N}\int_{-\infty}^{\infty} Box(y)e^{iw(y-nT)}dy \\ &= \sum_{n=0}^{N} e^{-iwnT}\int_{-\infty}^{\infty} Box(y)e^{iwy}dy \\ &= \sum_{n=0}^{N} e^{-iwnT}\mathcal{F}\{f\}(w) \\ &= \mathcal{F}^{-1}\{f\}(w)\sum_{n=0}^{N} e^{-iwnT}\end{aligned}$$

We used the fact that the term e^{-iwnT} is a constant when integrating along dy and the identity for the inverse Fourier transform of the Box function. Next, let us consider $\sum_{n=0}^{N} e^{-uwnT}$ further:

$$\begin{aligned}\sum_{n=0}^{N} e^{-uwnT} &= \sum_{n=0}^{N}(e^{-uwT})^{n} \\ &= \frac{1-e^{iwT(N+1)}}{1-e^{-iwT}}\end{aligned}$$

We recognize the geometric series identity for the left-handside of this equation. Since our series is bounded we can derive our right-handside.

Since e^{-ix} is a complex number and every complex number can be written in its polar form, i.e. $e^{-ix} = cos(x) + isin(x)$ we can go even further, using the trigonometric idententities that $cos(-x) = cos(x)$ and $sin(-x) = -sin(x)$:

$$\frac{1 - e^{iwT(N+1)}}{1 - e^{-iwT}} = \frac{1 - cos(wT(N+1)) + isin(wT(N+1))}{1 - cos(wT) + isin(wT)}$$

Which is still a complex number $(p + iq)$. Every complex number can be written as a fraction of two complex numbers. This means that the complex number $(p + iq)$ can be written as $(p + iq) = \frac{(a+ib)}{(c+id)}$ for any $(a + ib), (c + id) \neq 0$. For our case, let us use the follwoing substituations:

$$a := 1 - cos(wT(N+1)) \qquad b = sin(wT(N+1)) \tag{18}$$
$$c = 1 - cos(wT) \qquad d = sin(wT) \tag{19}$$

hence it follows $\frac{1-e^{iwT(N+1)}}{1-e^{-iwT}} = \frac{(a+ib)}{(c+id)}$. By rearanging the terms it follows $(a + ib) = (c + id)(p + iq)$ and multiplying the right handside out we get the follwing system of equations:

$$(cp - dq) = a \tag{20}$$
$$(dp + cq) = b \tag{21}$$

Which gives lead us we some further math (trick: mult first eq. by c and 2nd by d, then adding them together. using distributivity and we have the identity for p for example, similar for q) to

$$p = \frac{(ac + bd)}{c^2 + d^2} \tag{22}$$
$$q = \frac{(bc + ad)}{c^2 + d^2} \tag{23}$$

Putting our substituation for a, b, c, d back into the current representatio for p and q and using some trigonometric identites, this we then get:

$$p = \frac{1}{2} + \frac{1}{2}\left(\frac{cos(wTN) - cos(wT(N+1))}{1 - cos(wT)}\right) \tag{24}$$

$$q = \frac{sin(wT(N+1)) - sin(wTN) - sin(wT)}{2(1 - cos(wT))} \tag{25}$$

Since we have seen, that $\sum_{n=0}^{N} e^{-uwnT}$ is a complex number and can be written as $(p + iq)$ and we know now the explicit identity for those p and q we get for the 1-dimensional Fourier transform of the 1-dimensional Box-function the following final identity:

$$\begin{aligned} \mathcal{F}^{-1}\{f\}(w) &= \mathcal{F}^{-1}\{f\}(w) \sum_{n=0}^{N} e^{-iwnT} \\ &= (p + iq)\mathcal{F}^{-1}\{Box\}(w) \end{aligned}$$

In oder to derive next a identity for the Fourier transform for our 2-dim heighfield, we can proceed similarly, the only fact which changes is, that we are now in a 2-dimensional domain, i.e. we are about to compute a two-dimensional Fourier transform: Let us again us again a Box-function, this time a 2-dimensional Box-function $Box(x, y)$ just for the sake of convenience.

$$
\begin{aligned}
\mathcal{F}^{-1}\{f\}(w_1,w_2) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\sum_{n_2=0}^{N_1}\sum_{n_2=0}^{N_2} Box(x_1+n_1T_1, x_2+n_2T_2)e^{iw(x_1+x_2)}dx_1dx_2 \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\sum_{n_2=0}^{N_1}\sum_{n_2=0}^{N_2} Box(y_1,y_2)e^{iw((y_1-n_1T_1)+(y_2+n_2T_2))}dx_1dx_2 \\
&= \sum_{n_2=0}^{N_1}\sum_{n_2=0}^{N_2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} Box(y_1,y_2)e^{iw(y_1+y_2)}e^{-iw(n_1T_1+n_2T_2)}dy_1dy_2 \\
&= \sum_{n_2=0}^{N_1}\sum_{n_2=0}^{N_2} e^{-iw(n_1T_1+n_2T_2)}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} Box(y_1,y_2)e^{iw(y_1+y_2)}dy_1dy_2 \\
&= \left(\sum_{n_2=0}^{N_1}\sum_{n_2=0}^{N_2} e^{-iw(n_1T_1+n_2T_2)}\right)\mathcal{F}^{-1}\{Box\}(w_1,w_2) \\
&= \left(\sum_{n_2=0}^{N_1} e^{-iwn_1T_1}\right)\left(\sum_{n_2=0}^{N_2} e^{-iwn_2T_2}\right)\mathcal{F}^{-1}\{Box\}(w_1,w_2) \\
&= (p_1+iq_1)(p_2+iq_2)\mathcal{F}^{-1}\{Box\}(w_1,w_2) \\
&= ((p_1p_2-q_1q_2)+i(p_1p_2+q_1q_2))\mathcal{F}^{-1}\{Box\}(w_1,w_2) \\
&= (p+iq)\mathcal{F}^{-1}\{Box\}(w_1,w_2)
\end{aligned}
$$

Where we define $p := (p_1p_2 - q_1q_2)$ and $q := (p_1p_2 + q_1q_2)$. For this identity we used green's integration rule which allowed us to split the double integral to the product of two single integrations. Also, we used the definition of the 2-dimensional inverse Fourer transform of the Box-function. We applied the same substituation like we did in for the 1 dimensional case, but this time twice, once for each variable seperately. The last step, substituting with p and q will be useful later in the implementation. The insight should be, that the product of two complex numbers is again a complex number. We will have to compute the absolute value of $\mathcal{F}^{-1}\{f\}(w_1,w_2)$ which will then be equal $(p^2+q^2)^{\frac{1}{2}}\left|\mathcal{F}^{-1}\{Box\}(w_1,w_2)\right|$

2.10 Final Expression

As the last step of our series of derivations, we plug all our findings together to one big equation in order to compute the colors in the CIE_XYZ colorspace:

For a given heigh-field $h(x,y)$, representing a small patch of the nanostructure of our surface, the resulting CIE_XYZ caused by the effect of diffraction can be computed like the following:

$$\begin{pmatrix} X \\ X \\ Z \end{pmatrix} = C(w_i, w_r) \int_\lambda \sum_r \sum_s (p(r,s;T)^2 + q(r,s;T)^2)^{\frac{1}{2}}$$
$$\sum_{n=0}^{N} \frac{(wk)^n}{n!} \left| F_{fft}^{-1}\{i^n h^n\}(\frac{2\pi u}{\lambda}, \frac{2\pi v}{\lambda}) \right|^2 \phi(u - w_r, v - w_s) \begin{pmatrix} S_x(\lambda) \\ S_y(\lambda) \\ S_z(\lambda) \end{pmatrix} d\lambda \tag{26}$$

where $(p(r,s;T)^2 + q(r,s;T)^2)^{\frac{1}{2}}$ represents the phaser, $\phi(x,y) = \pi e^{-\frac{x^2+y^2}{2\sigma_f^2}}$ is the gaussian window, $\begin{pmatrix} S_x(\lambda) \\ S_y(\lambda) \\ S_z(\lambda) \end{pmatrix}$.

3 Implementation

how to discretize from final derivation to computation? what do we have to precompute, what during runtime? how does the final algorithm look like explain shaders: vertex(geometriy, precomp) - and fragment-shader(in local space-tspace) how from cie_xyz to cie_rgb how gamma correction how texturing can we do better?

TODO: explain that there is the jrtr and the scene code - what are their responsibilities.

shader

In computergraphics, we are interested in rendering a given scene containing our 3d geometries by using so called shader programs. The purpose of such programs, which run directly on the gpu hardware device, is to compute the colorization and illumination of the objects living in our scene. This computation happens in several stages and depends on the provided input paramteters like the camera, light sources, objects material constants and the desired rendering effect one is interested in to model. The shader stages are also modeled as small little programs, the so called vertex-, geometry- and fragment-shaders. Those stages are applied within the rendering pipeline sequencially.

Our shaders are written in GLSL, developed for OpenGl. The decission for using OpenGl has been made since the underlying framework which is responsible for the precomputation of all scene date is based on a framework written in Java using JOGL in oder to communicate with the GPU and precompute all the relevant scene data. This framework, the so called jrtr framework has been developed as an exercise during the class computer graphics held by M. Zwicker which I attended in autumn 2012. The framework itself has been extended during this thesis quite a lot. Further, there are also some precomputations involved, perfomed in matlab. This is basically addressing all the required precomputations for the provided heigh-fields, refering to computation of the inverse two dimensional Fourier transformations which are further explained within this chapter.

It's noteworthy that all the vertices are processed within the vertex-shader, whereas the fragement shader's responsibility is to perfrom pixelwise rendering, using the input from the vertex shader. Just remember, fragements are determined by a triple of vertices. hence each pixel has assigned a trilinear interpolated value of all input parameters of its spanning vertices. Usually, all necessary transformations are applied vertex-wise, considering the vertex-shader as the precomputation stage for the later rendering within the rendering pipeline, in the fragment-shader. In the geometry shader, new vertices around a considered vertex can be created. this is useful for debugging - displaying normals graphically for example.

In this section we are going to explain how to get a fragment-shader from our findings for our BRDF formultion from the last section. this fragment-shader will render the effect of diffraction on our given geometry pixelwise. Therefore, the quality of diffraction depends on the number of pixels we are going to use for the rendering process and this is directly determined by the resolution of the canvas in which the rendered images are being displayed. But, before we can start formulating our fragment-shader we first have to write our vertex shader which does all the precomputations.

By the end of the day we will end up with two different shaders, one which basically samples the whole lambda space using a gaussian window. This shader will be modeling the effect of diffraction completely but will also be rather slow. The other shader will use a gaussian window too but will just use a few wavenumber for the sampling process. Furthermore, this shader will thread specularity seperatly as a special case which will be more like an approximation.

tell how we are going to sample - uniformly along lambda - explain drawback of this approach - explain possible solutions for this issue. maybe refer to reference shader or leave this for the disscusion part.

3.1 Setup

explain geometry computation explain light(source) setup explain factories explain camera setup explain how materials are stored explain how assigned to jrtr explain how passed to glsl shader - see computer graphics slides maybe show schematically the architecture

3.2 Precomputations in Matlab

explain matlab code explain shifts explain what will be outputed

3.3 jrtr Framework

explain how this will work

3.4 GLSL Diffraction Shader

start using the final findings from chapter 2 and substitute explain how all the components are computed and why they are computed like this.

Algorithm 1 Vertex diffraction shader

```
foreach $Vertex\, v \in Shape$ do
end for
```

Algorithm 2 Fragment diffraction shader

```
foreach $Pixel\ p \in Fragment$ do
    $BRDF_{XYZ}, BRDF_{RGB} = vec4(0.0)$
    $(u, v, w) = \hat{\mathbf{k_1}} - \hat{\mathbf{k_2}}$
    for $(\lambda = \lambda_{min};\ \lambda \leq \lambda_{max};\ \lambda = \lambda + \lambda_{step})$ do
        $k = \frac{2\pi}{\lambda}$
        $(w_u, w_v) = (ku, kv)$
        $w_{color} = (S_x(\lambda), S_y(\lambda), S_z(\lambda))$
        for $(r)$ do
            for $(s)$ do
                $coords = getLookUpCoord(r, s)$
                $P = taylorApprox(coords, k, w)$
                $w_{r,s} = gaussianWeight(dist)$
                $scale_{pq} = pqFactor(w_u, w_v)$
                $P* = scale_{pq}$
                $P_{abs} = |P|^2$
                $P_{abs}* = w_{r,s}$
                $BRDF_{XYZ}+ = vec4(P_{abs} * w_{color}, 0.0)$
            end for
        end for
    end for
    $BRDF_{XYZ} = BRDF_{XYZ} * C(\hat{\mathbf{k_1}}, \hat{\mathbf{k_2}}) * shadowF$
    $BRDF_{XYZ}.xyz = D_{65} * M_{XYZ-RGB} * BRDF_{XYZ}.xyz$
    $BRDF_{RGB}.xyz = D_{65} * M_{XYZ-RGB} * BRDF_{XYZ}.xyz$
    $BRDF_{RGB} = gammaCorrect(BRDF_{RGB})$
end for
```

4 Data Acquisition and Evaluation

what is this chapter about how is evaluation perfromed our shader

4.1 Diffraction Grating

list def from wiki

$$dsin(\theta_m) = m\lambda$$

$$d(sin(\theta_i) + sin(\theta_m)) = m\lambda$$

$$sin(\theta_m) = \left(\frac{m\lambda}{d} - sin(\theta_i)\right)$$

$$\forall \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathbb{R}^3 : \exists r \in [0, \infty)\exists\phi \in [0, 2\pi]\exists\theta \in [0, \pi] \text{ s.t.}$$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} rsin(\theta)cos(\phi) \\ rsin(\theta)sin(\phi) \\ rcos(\theta) \end{pmatrix}$$

4.2 Snake Skin Parameters

5 Results

differece of this shader compared to evaluation shader

6 Conclusion

explain why we did our derivations explain why our approach is a good idea explain how the straight foreward approach would behave compared to our approach, computing the fourier transformations straight away. explain what we achieved, summary say something about draw-backs and about limitations of current apporach say something about the ongoing paper

6.1 Further Work