Imperial College London

Imperial College London

Department of Computing

Advanced Computer Graphics Coursework

Authors:
Arvin Lin, CID: 01785173
Jeremy Xie, CID: 01742291

February 21, 2020

Fresnel Reflectance

In this part of the project, we were tasked to generate plots of Fresnel reflectance for a dielectric material. The detailed algorithm is given as follows:

Plot generating algorithm for Fresnel Reflectance

1. Here we are given an incidence angle θ_i and index of refractions η_i and η_t. According to Snell's Law, we have the following relationship :

$$\eta_i \sin\theta_i = \eta_t \sin\theta_t$$

Intuitively we can calculate the refraction angle as follows:

$$\begin{gathered}\eta_t \sin\theta_t = \eta_i \sin\theta_i \\ \sin\theta_t = \tfrac{\eta_i}{\eta_t} \sin\theta_i \\ \theta_t = \arcsin\left(\tfrac{\eta_i}{\eta_t} \sin\theta_i\right)\end{gathered}$$

With these four elements, we could calculate the Fresnel reflectance for parallel polarized light $R_\parallel$ and Fresnel reflectance for perpendicular polarized light $R_\perp$ following :

$$\mathrm{R}_\parallel = \left|\frac{\eta_t \cos\theta_i - \eta_i \cos\theta_t}{\eta_t \cos\theta_i + \eta_i \cos\theta_t}\right|^2$$

and

$$\mathrm{R}_\perp = \left|\frac{\eta_\mathrm{i} \cos\theta_i - \eta_\mathrm{t} \cos\theta_t}{\eta_\mathrm{i} \cos\theta_i + \eta_\mathrm{t} \cos\theta_t}\right|^2$$

We can then calculate the unpolarized reflectance as:

$$\mathrm{F_r} = \frac{1}{2}\left(\mathrm{R}_\parallel + \mathrm{R}_\perp\right)$$

and the Transmittance could be retrieved from :

$$\mathrm{T_r} = 1 - \mathrm{F_r}$$

2. We then calculate the Brewster angle θ_B and the Critical angle θ_C using the following formulas :

$$\begin{gathered}\tan\left(\theta_\mathrm{B}\right) = \left(n_2/n_1\right) \\ \theta_B = \arctan\left(\frac{\eta_2}{\eta_1}\right)\end{gathered}$$

and

$$\begin{gathered}\sin\left(\theta_C\right) = \left(n_2/n_1\right) \\ \theta_c = \arcsin\left(\frac{n_2}{n_1}\right)\end{gathered}$$

3. The Schlick's Approximation has the following formula where

$$\mathrm{F_r}(\cos\theta) = \mathrm{R}_0 + \left(1 - \mathrm{R}_0\right)\left(1 - \cos\theta\right)^5$$

Here the θ in the formula is our incidence angle θ_i

4. Here we present the plots we have generated regarding the following situation where η_t=1.45 and η_t=1.0 respectively. The latter case represents a ray exiting the material into air. We report Brewster angle for the former case where it is an air-material interface and Critical angle for the second case.

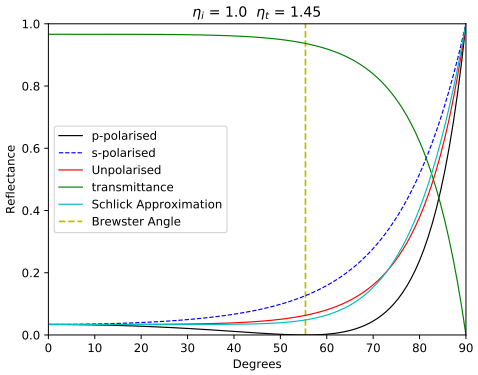

Figure 1: Brewster: 55.40771131249006

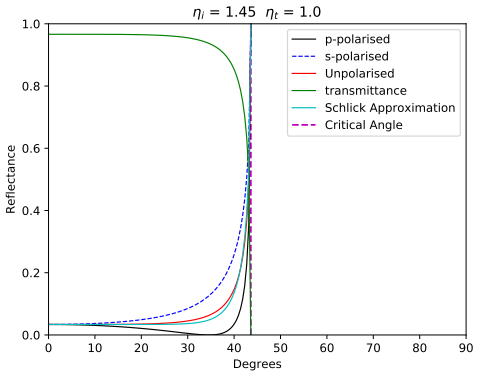

Figure 2: Critical: 43.60281897270362

Here as shown above, we have our η_i as 1.0 and η_t as 1.45, under this case, we have an upper bound of 90 degrees and we have the parallel polarized curve and perpendicular polarized curve as shown. Here the red curve represents the Unpolarized curve. It is worth to notice that our parallel polarized curve goes to zero at Brewster Angle of 55.40771131249006 as the vertical yellow dashed line.The cyan line represents the Schlick's Approximation.

The second case shows significantly different behaviour where the $\eta_i = 1.45$ and $\eta_t = 1.0$, this represents a ray exiting the material into air, under this circumstance, the upper bound is the Critical Angle itself with a degree of 43.60281897270362, the critical angle is plotted as a vertical purple dashed line. We have also plotted the curves of Transmittance for better observation for behaviours.

5. One advantage of suing Schlick's Approximation is that now we are able to acquire approximated reflectance without estimating index of refractions.

6. The function $index\_of\_refraction$ is a function that we have written to include the case against the complex entries. Here we do not need such a complicated function with over-complex entries.

Generate Monte Carlo samples according to an Environment Map

In this section of the assignment, we investigate the possibilities of acquiring Monte Carlo samples according to an Environment Map given to us. The detailed algorithm is stated as follows:

1. In the first step, we load the latitude-longitude format of the Environment Map of Grace Cathedral. Here the dimension of this image shall be $(512, 1024, 3)$

where 512 represents height and 1024 represents width and 3 represents 3 colour channels of R,G and B. we calculate the intensity map $I = \frac{R+G+B}{3}$. This sampling algorithm is based on the probability density function and cumulative density function of the Environment Map of each pixel. We create a copy of our image named F for operations on this instance.

2. The intensity of each pixel has been scaled by the solid angle term $\sin(\theta)$. For each pixel, we traverse the image through a nested for-loop with h represents the traversing index of height and w represents the traversing index of width. We calculate the intensity of each pixel in the following fashion :

$$I_{hw} = I_{hw} * \sin\left(\frac{h \cdot \pi}{height - 1}\right)$$

3. We then calculate the 1-dimensional CDF by dividing the sum of the intensities we have calculated above, this would convert the dimension to $(512,)$. This 1-dimensional CDF represents the sum likelihood of each row, this will be used for selecting a specific row later. Another blank 2-dimensional CDF is created using the height and width of the image.

4. Our goal here is to sample from the probability density function with a proposal distribution, here we have experimented with several distributions like Gaussian i.e.

$$X \sim \mathcal{N}(\mu, \sigma^2)$$

However, we have found that the most appropriate distribution to exercise here is the Uniform distribution with $a = 0$ and $b = 1$. We declare and initialise two indices for specifying as sampling window later, they are idx_row and idx_col respectively. An 1-dimensional cumulative distribution function will be generated for row selection named $sampling\_distribution\_row$ such that we traverse the through the height first, for any element h inside the 1-dimensional CDF that is larger or equal to the $sampling\_distribution\_row$, we make the idx_row to be this element. We generate another 1-dimensional CDF that is called $sampling\_distribution\_col$ for column selection. We break out of the loop once the conditions are met. It is worth to mention that

$$sampling\_distribution\_row \sim \mathcal{U}(0, 1)$$

$$sampling\_distribution\_col \sim \mathcal{U}(0, 1)$$

5. We have used a rather brute-force way to select a window of size 5-by-5 such that we listed every options in the for-loop, if our traversing index $window$ has its first entry ≥ 0 and ≤ 512 and its second entry ≥ 0 and ≤ 1024, we mark this area as **Blue**.

6. We then apply exponential scaling and gamma correction using functions we have written from the first assignment on the instance F. The images with different sample numbers are displayed below:

Figure 3: 64 Samples

Figure 4: 256 Samples

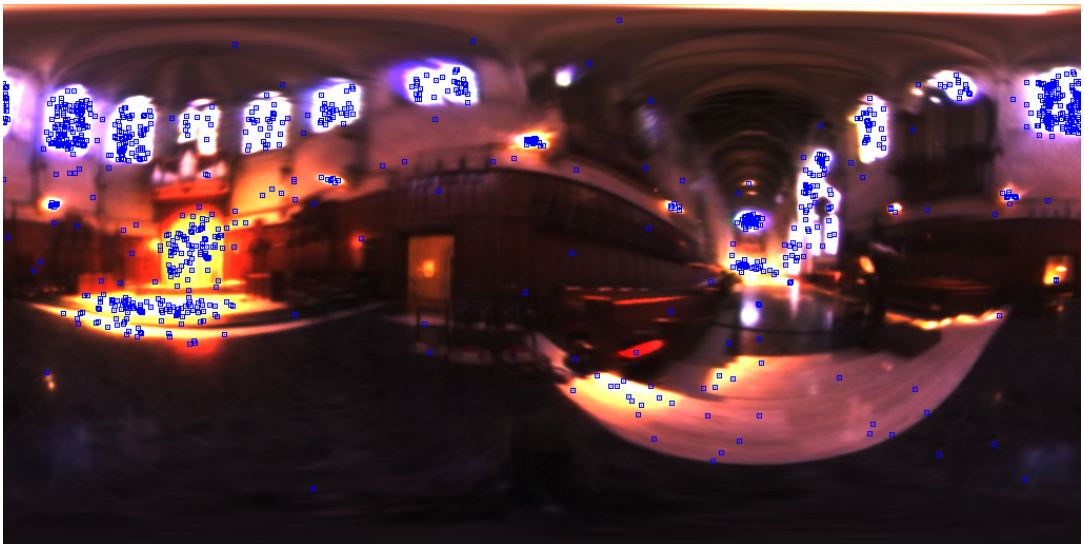

Figure 5: 1024 Samples

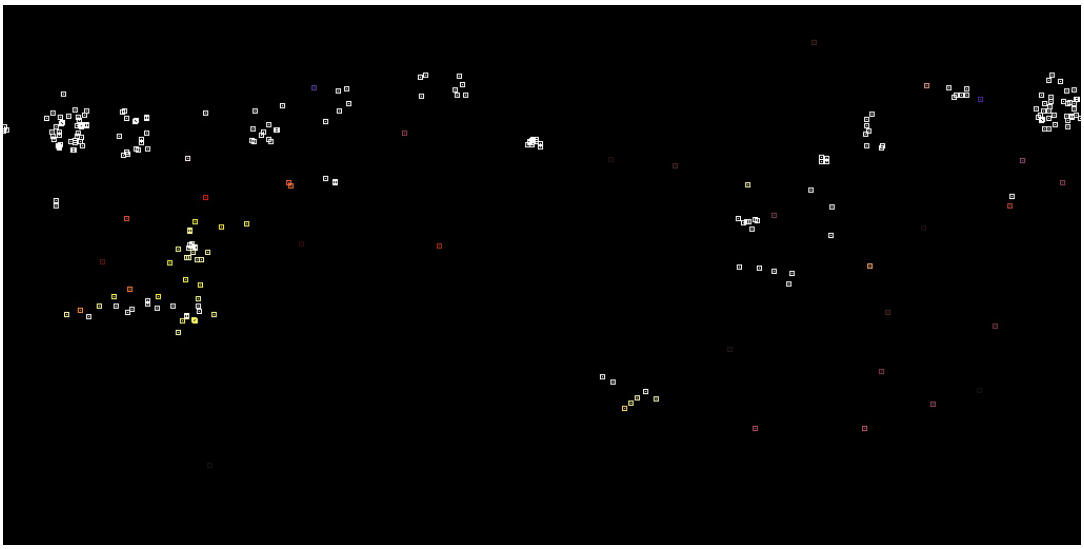

Figure 6: Environmental Map showing only samples with Sample Number equals to 256 with RGB values

We have created an empty Environmental Map using the height, width and color channels of the image that this empty map is used to store the RGB

values of the Grace Cathedral Environmental Map when the sample number equals 256.

7. It is safe to draw a conclusion based on the samples exhibited above that pixels with higher intensities are easier to be sampled where pixels in the middle part of the image are more likely to be included in the samples when they are compared to pixels at the edge of the image.

Median Cut on Environmental Map

In this section we present a method to perform Median Cut on a given environmental map. Below we describe the algorithm and present the result of the Median Cut method on the Grace Cathedral map.

Median Cut algorithm

Being given an environmental map we have to first prepare the energy map for performing cuts. Specifically given an RGB image we first compute the intensity map $I = \frac{R+G+B}{3}$, afterwards we scale the intensity map with the solid angle term $\sin\theta$ to obtain the energy map. The python implementation with Numpy is as follows:

```
# Given lat long environmental map: map_, with shape (512, 1024, 3)
I = np.average(map_, axis=2)  # I = (R+G+B)/3
# Now create the sin matrix for scaling
scale_sin = np.sin(np.linspace(0, np.pi, 512).reshape(-1, 1))
scale_sin_matrix = np.repeat(scale_sin, 1024, axis=1)
# Compute the Energy Map
E = I*scale_sin_matrix
```

Listing 1: Compute Energy Map from Environmental Map

With the energy map obtained, we can start performing the Median Cut to the energy map. The procedure is as follows:

1. Given an energy map, we find the median pixel by computing the cumulative sum along both axis, the pixel with a cumulative sum of a half of the total sum of energy is the median pixel. In short, we want to find pixel (i, j) such that:

$$\sum_{m=1}^{i}\sum_{n=1}^{1024} E(m,n) = \sum_{m=1}^{512}\sum_{n=1}^{1024} E(m,n)$$

$$\sum_{m=1}^{512}\sum_{n=1}^{j} E(m,n) = \sum_{m=1}^{512}\sum_{n=1}^{1024} E(m,n)$$

2. With the median pixel found, we now cut the image and energy map along the longer axis by the median pixel. This cuts the energy map in half, where each half have the same total energy.

3. To perform more than 1 cut, we call the above procedure recursively. As an illustration, we first find the median cut on the whole energy map, this cuts the whole map into two sub-maps, then we recursively find the median cut of

the two sub-maps and obtain 4 cuts, repeat this procedure until we get the desired partitions.

4. The recursive function would return the median pixel of the last energy map during the final step in the recursion, the returned median pixels at each stage are then concatenated to a full list representing the centroids of each partitions.

The following figures present the results for the Median Cut algorithm performed on the Grace Cathedral environmental map. Figure 7, 8, 14, and 15 shows the environmental map cut into 2, 4, 16, 64 regions respectively. Figure 11 shows the centroids representing the RGB radiance of the map if we partition the map into 64 partitions.

Figure 7: Median Cut, 1 Cut

Figure 8: Median Cut, 2 Cuts

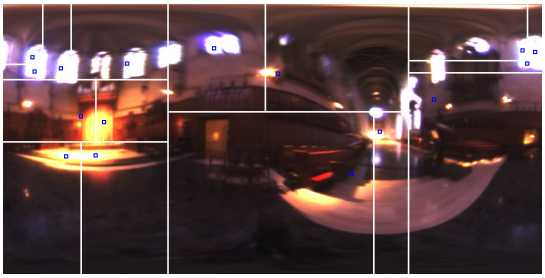

Figure 9: Median Cut, 4 Cuts

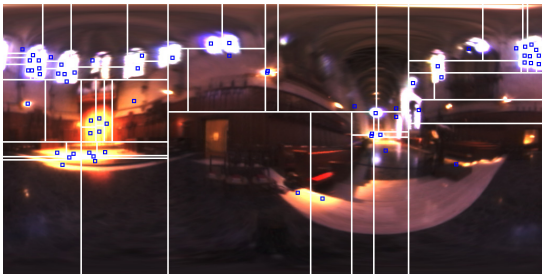

Figure 10: Median Cut, 6 Cuts

Figure 11: Centroids of RGB radiance for 64 partitions case

Rendering a sphere lit by Grace EM using PBRT

We are given an example scene file to render a diffuse sphere with albedo $\rho_d = 1.0$ in the Grace Cathedral lighting environment with 8, 16, 32 and 64 samples. With increased sample number, we have longer training time but less noise, we then extract the maximum from the image and apply linear scaling to the image by multiplying 255. We then save the images in pfm and ppm format.

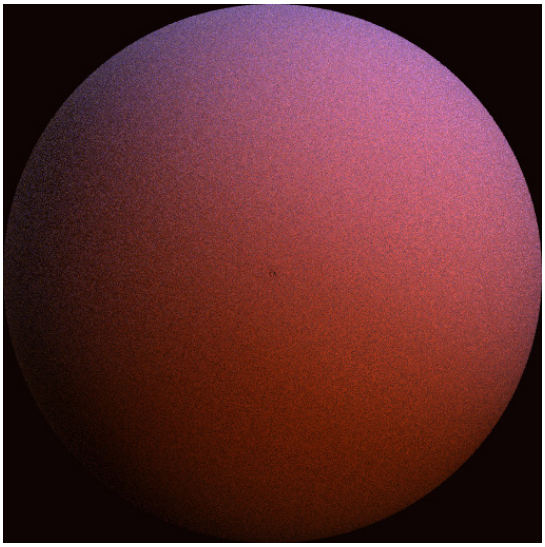

Figure 12: 8 samples

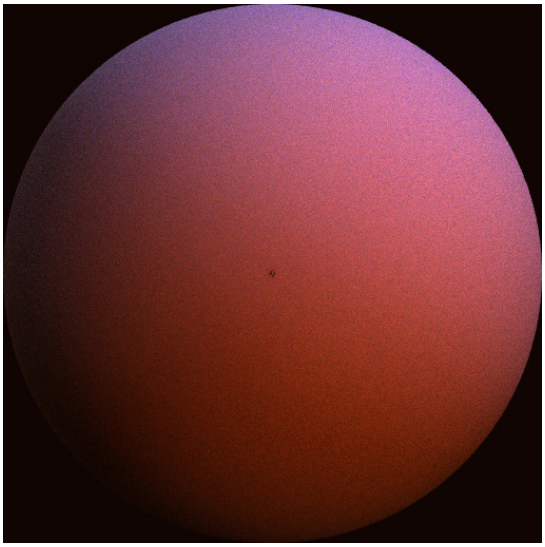

Figure 13: 16 samples

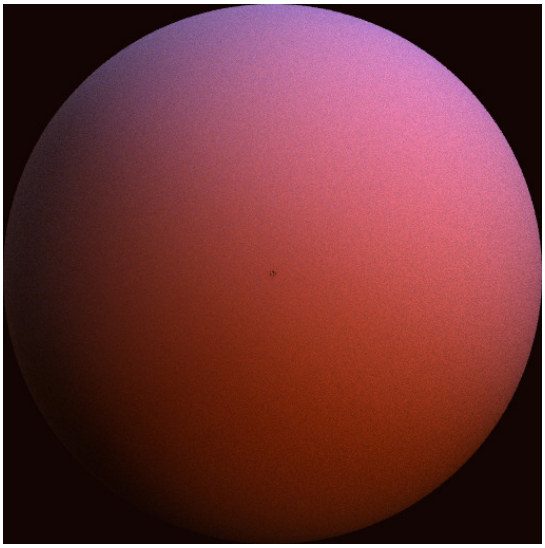

Figure 14: 32 samples

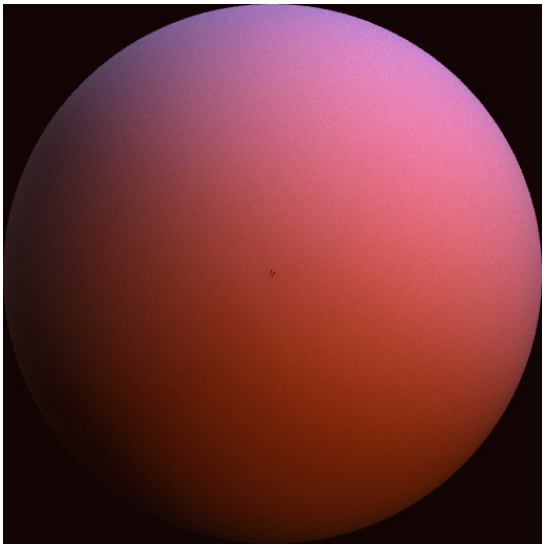

Figure 15: 64 samples

Statement of Contributions

Arvin is in charge of completing Part 3 and 4 while Jeremy finished Part 1 and 2. The report is a joint effort of both parties with equal contributions.