DH2323 Computer Graphics and Interaction Shader Rendering in OPENGL Project Report

1st Zehua Guo
zehuag@kth.se

2nd Hira Imtiaz
himtiaz@kth.se

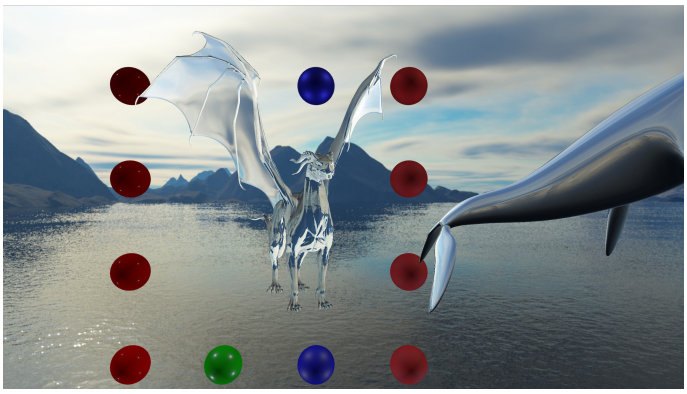

Fig. 1. Scene overview

***Abstract*—Physically-based rendering shading is an important element of computer graphics. It is used in creating the high quality and realistic images whose quality can be maintained by various lighting environment. This type of rendering shading has various applications which includes 3D animation, gemstones, films, games [2], fine scale human skin structure [10] and many more.**

***Index Terms*—rendering, PBR, reflection, refraction, skybox**

I. Introduction

The concept of Physically based rendering involves around many concepts that we have tried to implement. The main features revolve around diffusion and reflection, translucency and transparency, energy conservation and Fresnel equation. The main concepts of BRDF and Bling Phong model play an important role in the journey of creating a realistic and sophisticated imagery.

In this project, we have implemented a 3D scene with OpenGL environment [5] using C++ language. The main features of this project are Physically based rendering shader, reflection, refraction and skybox environment map. The project is based on the OPENGL tutorials from learnopengl.

We have taken an island as a background and over it, a 3D scene has been implemented. this scene consists of several spheres having different materials, a transparent dragon and a shiny whale. Surrounding the transparent dragon, there is a square of spheres using PBR rendering. We use RGB colors to make them look colorful. More specifically, the material of every sphere in each edge of the square is different so that we can observe the different rendering effects.

II. Related Work

Many talented people have worked in this area and their work and researches have helped us in the accomplishment of this project. The related work in the field of PBR are as follows:

1. Matt Pharr, Wenzel Jakob, and Grey Humphreys have written a book named as 'Physically Based Rendering: From Theory to Implementation' has described how to design a full featured rendering system capable of creating stunning imaginary. From the mathematical formulation to the practical implementation, this book has described PBR in a commendable manner [1]

2. Peter Kutz and Karl Li, current employees of Walt Disney Animation Studios have done a project that has some features related to this project. Their work and findings have been a source of inspiraion for us in our journey of our project.

3. Christiane Ulbricht, Alexander Wilkie and Werner Purgathofer have worked on PBR algorithms. They have demontrated in detail on how to practically implement rendering algorithms. Their findings has helped us in solving various problems that we encounter during implementation [3]

4. Learn OpenGL website have some amazing content related to PBR and everything has been described in detail which has helped us in understanding some basic concepts and its implementation in OpenGL environmen [4]

III. Technology Summary

A. Phong model

To introduce our PBR shader, we will first introduce Phong illumination model. Phong model consists of three key components: ambient light, diffuse light,specular highlight. We implemented a Phong shader to compare with our PBR shader. The key point is to add the three different lights together in the fragment shader to make up the object's final color.

B. Physically-based rendering shader(PBR)

A PBR shader tries to simulate the physically real illumination. It mainly contains three components, which are microfacet surface model,energy conserving and physically based BRDF. In this project, we use four direct light sources to light up the scene. With distributed light sources in four positions, we can observe the specular highlight in different directions.

1) Surface model: For the surface model, we have one parameter to measure the roughness of the surface. In fact, we all know that the rougher the surface, the more chaotic the direction of reflected light. In the evaluation part,we will adjust the value of roughness to see its affect. Besides the roughness, we have another parameter to measure how metallic the material is.Because the more metallic the material is, the more specular highlight it reflects.

As the figure below shows, from left to right, the roughness of the sphere gradually increases. We can see that as the roughness increases, the proportion of specular highlight gradually decreases.

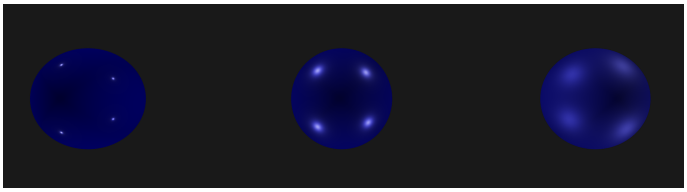

Fig. 2. Roughness

2) Energy conservation: The energy conservation mentioned above means outgoing light energy should never exceed the incoming light energy. We divide the light into two parts: reflection light and refraction light.

3) Cook-Torrance BRDF: The BRDF(Bidirectional reflectance distribution function) equation for our PBR shader contains both the diffuse and specular part.

$$f_r = k_d f_{lambert} + k_s f_{cook-torrance}$$

The first term is the $lambert$ diffuse light and the second term is the $cook-torrance$ specular light. k_d here is the ratio of incoming light energy that gets refracted with k_s being the ratio that gets reflected.

One important component of Cook-Torrance BRDF model is the Fresnel equation, which describes the ratio of light gets reflected over the light gets refracted. The result is corresponding to the angle between the light and the surface [8]. Because the Fresnel equation is based on continuous space, we use $Fresnel - Schlick$ approximation to achieve the performance.

$$F_{Schlick}(H, V, F_0) = F_0 + (1 - F_0)(1 - (H.V))^5$$

4) Evaluation : To evaluate our PBR shader, we compare it to the Phong illumination model. The Phong illumination contains three parts: ambient light, diffuse light, specular highlight [6]. Phong model works because it is simple enough to approximate common light scenes, but it is not physically based. The following graphs shows the result of Phong shader and PBR shader.Only one light source is used here. By comparing Fig3 and Fig4, we can find that PBR can create far more realistic scenes than Phong model. In this way, we can conclude that our PBR shader works well.

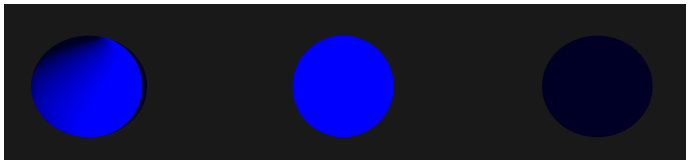

Fig. 3. Phong result

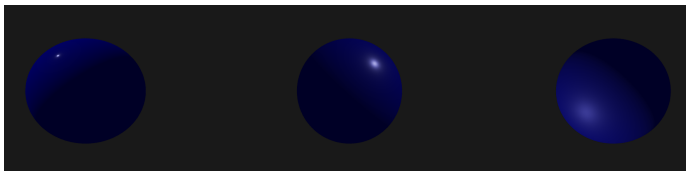

Fig. 4. PBR result

C. Refraction and Reflection Model

In this part, We have used ASSIMP(Open Asset Import Library) to load the 3D obj model to our scene. ASSIMP is a open-source model loader which is very light and flexible framework. It only has a header file which contains useful functions to load 3D objects with various formats.

The model we tried to load is a dragon with .obj format. We applied the refraction model in the fragment shader so that it can looks transparent and beautiful. To make it more cool, we also modified the model matrix corresponding to the vertex shader to rotate and transform it so as to get a better position for display.

We set the refraction index of air is 1.0 and the refraction index of the object material is 1.33. Therefore, we can directly use the predefined function $refract$ in GLSL language to compute the refraction light.

$$refract(vector, normal, 1.0/1.33)$$

Then we combine it into the fragment shader to make the dragon object look like transparent.

For the reflection model, we also tried to load a 3D whale object using the ASSIMP. We applied the reflection model to the whale so that it looks like a beautiful metallic 3D object reflecting with the background island.

To make it reflecting the environment, we need to understand the current spatial relationship. The skybox technology we used here can be seen as a environment map. The skybox wraps the whole scene into a cube with the island texture covered. If the light hits the object, the reflection light will hit a surface of the skybox.The color of the point where the reflection light hits this surface is the color we need to reflect.

Fig. 5. Refraction

To implement the reflection model, we also use the predefined function $reflect$ to compute the reflection color. The formula is embedded into the fragment shader of the whale object.

$$reflect(vector, normal));$$

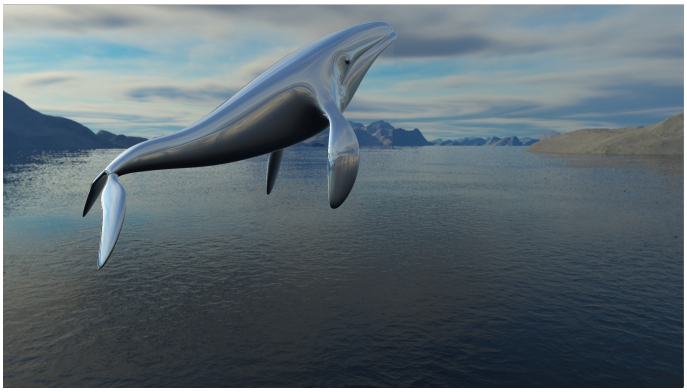

Fig. 6. Reflection

D. Skybox environment mapping

In game development, skybox is a common technique to make your game scene more interesting and larger than it actually is. We use 6 images to form a cube that encompasses the entire scene.

The skybox is a cube with the coordinate center at the camera position [7]. Since the volume of the skybox is 1, we need to scale down the global transformation matrix of the corresponding object to display our scene. The following 6 pictures make up the skybox we use.

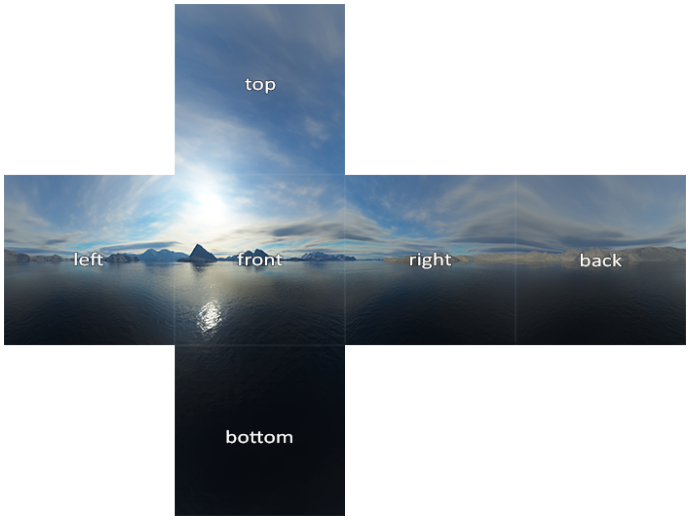

Fig. 7. Skybox

IV. Future Work

This project can be extended in various different ways depending upon the interest in a particular domain.

1. One of the ways to make it more presentable is to change the background and keep it simple but brighter. In this way one can add transparent spheres and work with different types of shading models to study the models in more details and to analyse the results practically.

2. Another way to extend this project is to add animation in the water to give it a more realistic look. In this way, one can learn the rendering and animation aspects of computer graphics in a single project. To achieve this goal, we still need more GPU resources to accelerate the computation process.

3. Besides, taking advantage of GPU to implement a Raytracer in this scene is also interesting and meaningful. With the Raytracer, we can add shadows and make the objects look more realistic and colorful. Currently, due to the limit of hardware, the rendering process always takes a long time and is not smooth enough. So we leave this Raytracer part for future work.

4. This project can be extended in a way that one can demonstrate the realistic view of an island by adding not only rendering and shading but also can give animation to various objects in an island. For example, the moving wind, animated animals or a whale even. Moving trees and displaying green area to give a complete picture of an island could be a part of this extension.

Contribution

From selecting a topic to writing our project report, it was an amazing journey. We both worked together and enjoyed it. We have learnt a lot from this project and hoping to go further in the area of computer graphics and vision. Zehua Guo was responsible for coding in OpenGL and Hira Imtiaz played her part in doing some research and writing the report alongside with the coding. We both are hoping to work together again in the near future.

REFERENCES

[1] Physically based rendering: from theory to implementation, url=https://www.pbr-book.org/, journal=Physically Based Rendering: From Theory to Implementation

[2] Mcauley, Stephen and Hill, Stephen and Hoffman, Naty and Gotanda, Yoshiharu and Smits, Brian and Burley, Brent and Martinez, Adam, title=Practical physically-based shading in film and game production, DOI=10.1145/2343483.2343493, journal=ACM SIGGRAPH 2012 Posters on - SIGGRAPH 12, year=2012

[3] Schlick Christophe, title=An Inexpensive BRDF Model for Physically-based Rendering, url=https://onlinelibrary.wiley.com, journal=Wiley Online Library , year=2003, month=Feb

[4] Learnopengl, title=Theory, url=https://learnopengl.com/PBR/Theory, journal=LearnOpenGL

[5] Hearn, Donald and Baker, M. Pauline. and Carithers, Warren R., place=Boston, title=Computer graphics with OpenGL, publisher=Prentice Hall, year=2010

[6] Tan, Ping, title=Phong Reflectance Model, DOI=10.1007/978-3-030-03243-2$_5$36 − 1, $journal = ComputerVision, year =$ 2020, $pages =$ 1–3

[7] Isidoro, John R. and Sander, Pedro V., title=Animated skybox rendering and lighting techniques, DOI=10.1145/1185657.1185827, journal=ACM SIGGRAPH 2006 Courses on - SIGGRAPH 06, year=2006

[8] Ghosh, Abhijeet, title=Cook-Torrance Model, DOI=10.1007/978-0-387-31439-6$_5$31, $journal = ComputerVision, year =$ 2014, $pages =$ 146–152

[9] Stephenson, Ian, title= Shading Models, DOI= 10.1007/978-1-4471-3800-6$_2$8, $journal = EssentialSeriesEssentialRenderManfast, year =$ 2003, $pages =$ 289–295

[10] Haro, Antonio and Guenter, Brian and Essa, Irfan, title=Real-time, Photo-Realistic, Physically Based Rendering of Fine Scale Human Skin Structure, DOI= 10.1007/978-3-7091-6242-2$_5$, $journal = EurographicsRenderingTechniques$2001, $year =$ 2001, $pages =$ 53–62