02562 Rendering – Introduction

Project Report

Jonatan Jais Kofoed Rasmussen
s183649@student.dtu.dk
18.12.24

Lab journal and Project website: https://budoboy07.github.io/Rendering/

IIII Introduction

This project is inspired by *"Project Initiator: Rendering with a Measured BRDF"*

Project website link + Worksheet & Exercises: https://budoboy07.github.io/Rendering/

This project, undertaken as part of the class *02562 Rendering - Introduction* at DTU, will explore how different materials interact with light using BRDF-based rendering techniques. For this, we have implemented a BRDF model using the WebGPU framework. As part of the project, the material-light interactions of five different materials were simulated, each with varying properties. The following materials were chosen: *Chrome*, *Gold*, *Ceramic*, *Emerald* and *Rubber*.

The purpose of the project is to implement a shader that uses measured material appearances to render realistic-looking materials. The project initiator worksheet asks for an implementation of measured BRDFs using the MERL database, but after many days of trial and error, I was not able to successfully render a WebGPU scene using binary material files from the MERL database. This is very troublesome, as the main learning objectives revolved around *measured* BRDFs. Although far from ideal, I chose to focus on the learning objective *"Render the appearance of a real-world material"* by using simplified physics-based rendering techniques from earlier in the course. As part of my results, five chosen materials are simulated via different configurations of properties such as *metallic*, *roughness*, and *specular* effects. This is not measured BRDF, but I decided that handing in a working (but simplified) scene was better than handing in a broken scene that couldn't be rendered at all. This report will cover the methodology and implementation of measured BRDF, and it will be discussed how such an implementation differs from the results I achieved, which is not the MERL BRDF implementation described in the project specifications.

IIII Methodology

The MERL BRDF database, based on measured data from real materials, contains binary files that describe how real materials reflect light at different angles. By interpreting these binary files, it is possible to look up exact reflection values for a material rather than calculating them. However, as mentioned in the introduction, the implementation I ended up going with instead focuses on analytical BRDF models based on physically-based rendering (PBR) principles.

In my code, I am using a simplified version of the Cook-Torrance BRDF model:

$$f_r(\omega_i, \omega_0) = k_d \cdot f_{Lambert} + k_s \cdot f_{CookTorrance}$$

Where:
k_d is the diffuse reflection coefficient (in my code, this is controlled by my metallic parameter).
k_s is the specular reflection coefficient.
$f_{Lambert}$ is the Lambert diffuse term (in my code, I use $diffuse = material.albedo \cdot\ NdotL$).
$f_{CookTorrance}$ is the specular term (in my code, I use $fresnel = F_0 + (1.0 - F_0) \cdot (1.0 - NdotV)^5$).

Note that this is only a practical approximation of BRDF that uses direct lighting calculation with environmental map reflections and material properties (material/roughness/specular). Since these are the key physical properties I am simulating with my rendering model, I will go over each of them in the implementation section.

IIII Implementation

The project scene was a continuation of the work done in week 9's worksheet, with the panoramic environment map (background texture) being swapped out. Originally I wanted to use five copies of the Newell teapot, but I was having trouble with the binary material files from the MERL database, so as part of my trouble-shooting I turned the teapots into balls: For simplicity, this ended up being the object I used for my final scene.

As previously mentioned, to simulate the appearance of various materials, I implemented three physical properties of materials, which I will now go over one by one:

Metallic:
Metallic surfaces reflect incoming light similar to a mirror. In our implementation, a high metallic value (close to 1.0) results in strong reflections and reduced diffuse scattering, while a low value (close to 0.0) allows for more diffuse scattering. In my implementation, this also impacts the Fresnel Effect, which is the reflectivity angle between the viewing direction and surface normal.

Roughness:
Metallic surfaces reflect incoming light similar to a mirror. In our implementation, a high metallic value Roughness simulates the microscopic surface variation of the material. A low roughness

value results in clear and sharp reflections (like polished chrome), while higher roughness values create a more diffused and spread-out reflection pattern (like brushed metal or matte surfaces).

Specular:
This controls the intensity of direct light reflections (the bright spots that appear on surfaces when light reflects directly toward the viewer). A higher specular value means stronger, more noticeable highlights, while a lower value results in more subtle highlights.

IIII Results

Below is an image of the final scene I ended up with. The material properties mentioned in the previous section have been combined to render the appearance of five different real-world materials. The materials are, going from left to right:

Chrome: High metallic, low roughness, high specular
Gold: Medium metallic, low roughness, medium specular
Ceramic: Low metallic, medium roughness, medium specular
Emerald: High metallic, low roughness, low specular
Rubber: Low metallic, high roughness, low specular

IIII Discussion and conclusion

In this project, the material-light interactions of five different materials (*Chrome*, *Gold*, *Ceramic*, *Emerald* and *Rubber*) was explored and documented. The appearance of these materials were simulated using a mix of the properties *metallic*, *roughness,* and *specular*. The methodology and model used (a simplified version of the Cook-Torrance BRDF model) has been described in the methodology section, and an image of the rendered scene is shown in the results section on the previous page. You can also render this scene in your own browser by running the provided project-related code.

Although the results were fine in their own context, they are not an implementation of measured BRDFs using the MERL database, which according to the project initiator worksheet was the purpose of the project. As mentioned in the introduction, I failed at implementing code that could parse and render the binary material files from the MERL database. WebGPU is capable of this, but I could simply not get it to work in my own scene. I decided that handing in a working (but simplified) scene was better than handing in a broken scene that couldn't be rendered at all, but I acknowledge that the scope and quality of the project suffered as a result of it. Throughout this report work, I tried to focus on the learning objective *"Render the appearance of a real-world material"* by using simplified physics-based rendering techniques from earlier in the course. I hope it has been clear how my implementation differs from the work that was specified in the project initiator. In terms of future work and improvements, it would be preferred to do proper rendering using measured BRDF. The implementation I chose instead focuses on analytical BRDF models based on physically-based rendering principles, which technically is less impressive, but I hope that my project work has ended up being acceptable in its current form.

Lab journal and Project website: https://budoboy07.github.io/Rendering/