Frank de Jong

160010@buas.nl

PBR

Workflow in Plutonium

Table of Contents

[Materials 2](#_Toc7079288)

[Metalness vs Specular 2](#_Toc7079289)

[Base debuggable material 2](#_Toc7079290)

[Base complex material 3](#_Toc7079291)

[Base BPR material 4](#_Toc7079292)

# Materials

The materials used by a shader will vary widely per model. We can either conform to very few materials and create default values to accommodate for the missing parts or we can create many specialized graphics pipelines for each material. The latter is faster if the graphics pipelines are created on load time, but the first approach is easier on the complexity as little to no switching will be required but the resource use will go up.

The current favored workflow will be to create many shaders for missing texture maps but keep the scalar and factor values global. This is done because scalar and factor values do not account for much resource space on the GPU, but texture maps do. This only creates a graphics pipeline swap whenever a material is used with different available texture maps.

## Metalness vs Specular

I’ve chosen to use a specular glossiness workflow over metal roughness because it gives the artist more control over their materials and because it has less artifacts.

The shaders will have about the same performance, but specular glossiness uses more memory as it has to store more textures. I however don’t see this as a problem as I tended to have enough memory to go around.

When a single model part goes from a metal to dielectric material (for instance when the paint is scratched off a metal surface). We can get artifacts in the metal roughness workflow where white lines will appear. This makes the surface look almost moldy when it’s far from it.

## Base debuggable material

A base object for materials is preferred as it gives us global debug information about a material. This will include the name of the material and whether meshes should be rendered that use the material.

## Base complex material

This is the first derived object from the base debuggable material. It defined parameters commonly found in either Blinn-Phong shading and PBR shading. This material contains the following parameters:

* Alpha mode (enum)
* Alpha cutoff (scalar)
* Double sided (Boolean)

The alpha mode specified which alpha blending should occur on meshes with this material. They are defined as:

* Opaque (no blending)
* Toggle (Fully opaque if < cutoff, otherwise; fully transparent)
* Blended (Apply Porter and Duff blending)

The double-sided property specifies whether this material should have back face culling disabled, this is for leaves and other flat shapes.

## Base BPR material

One of the directly derived objects from the base debuggable material is the Physically Based Rendering material. This material holds all the scalar and factor information of PBR materials. This includes:

* Albedo factor (RGBA)
* Glossiness (scalar)
* Specular factor (RGB)

The albedo factor can be used for a flat color refracted color on the material if no albedo map is present in a higher-level material. If a map is present, this functions as a filter for the albedo map.

The glossiness is chosen over roughness as it looks more natural to human when looking at the texture map. I also personally prefer the naming glossiness over roughness.

Because we must alter our BRDF functions slightly. This saves compute time in the shader at a high cost of memory.

According to the GLTF standard, the BRDF for roughness should be implemented [this way](https://github.com/KhronosGroup/glTF/tree/master/specification/2.0#appendix-b-brdf-implementation). For specular glossiness this should be done [this way](https://github.com/KhronosGroup/glTF/blob/master/extensions/2.0/Khronos/KHR_materials_pbrSpecularGlossiness/README.md).

We also use a specular albedo factor to allow for maximum control on our specular reflections. This comes at a cost of memory however because instead of a scalar value for mealiness we now store an RGB color. We must also make sure that the albedo and specular are mutually exclusive, meaning that a fragment must either be black in the albedo or specular albedo map.