1.1 Prep and Debug

• Your choice of shading space
I perform shading in world space, but I use normal in tangent space for lighting calculations.

• Your choice of descriptor set layout and descriptor bindings. Motivate this choice briefly.

- Descriptor Set 0, Binding 0: UScene uniform buffer block, used to store camera, projection, and projCamera matrices.

- Descriptor Set 1, Binding 0: albedo map texture sampler.
- Descriptor Set 1, Binding 1: metallic map texture sampler.
- Descriptor Set 1, Binding 2: roughness map texture sampler.
- Descriptor Set 1, Binding 3: normal map texture sampler.
- Descriptor Set 1, Binding 4: ambient occlusion map texture sampler.

- Descriptor Set 2, Binding 0: Ulight uniform buffer block, used to store position and colorfor lights.

By grouping the uniform buffer blocks and texture samplers into separate descriptor sets, it allows for more efficient binding of resources and easier management of resources at the application level. The use of separate bindings within each descriptor set allows for multiple resources to be bound to the same set and accessed by the shader.

• Describe uniform input data (e.g. UBOs and similar). Motivate your choices briefly.

1. UScene UBO block: This uniform buffer block stores camera, projection, and projCamera matrices, which are used to transform the 3D scene geometry from world space to clip space for rendering. By storing these matrices in a UBO, they can be updated less frequently and accessed efficiently by the shader.
2. ULight UBO block: This uniform buffer block stores the position and color of the light source in the scene. By storing the light source data in a UBO, it can be updated less frequently and accessed efficiently by the shader.

Using UBOs for input data allows the shader to efficiently access and update large amounts of data, such as matrices and light source data. Additionally, by grouping related data into separate UBO blocks, it allows for more efficient management of resources at the application level.

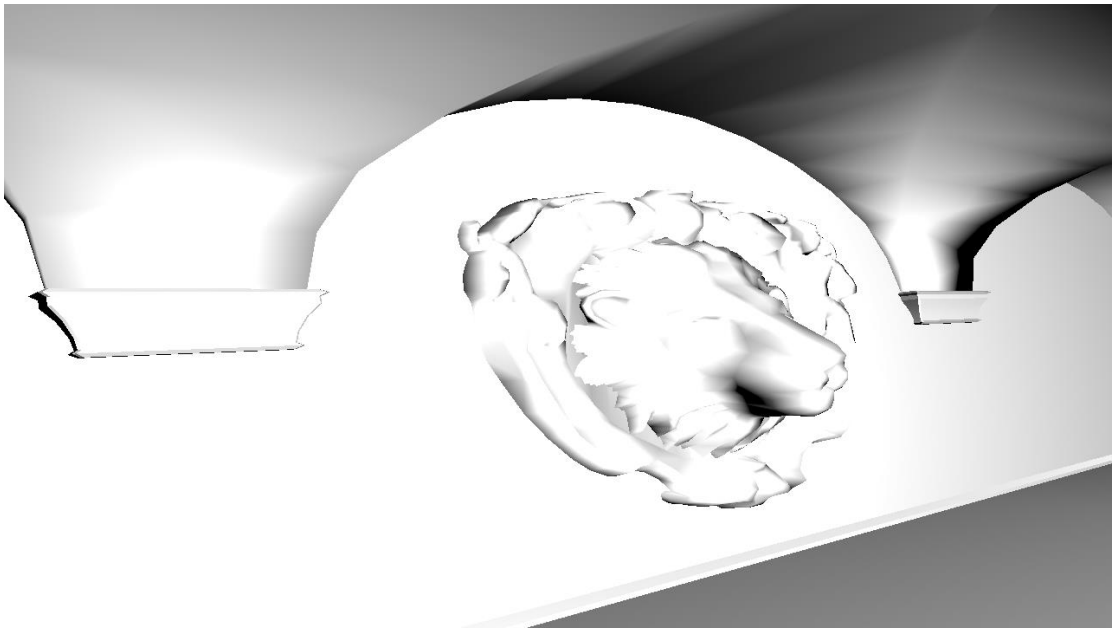
1.2 Lighting

include a few screenshots of your results. Include separate images that visualize the factors shown in Figure 4 and the diffuse contribution. Explain each contribution and discuss whether or not it behaves as you expect.

(a) Overview

(b) Lion Head

(c) D



(d) G

(e) F

(f) PBR specular term

The results of each term all have met my expect, but I do not stick to the equation teacher have given me, I modify a bit.

Term D : I use Trowbridge-Reitz GGX $NDF_{GGXTR}(n,h,a) = \frac{a^2}{\pi((n\cdot h)^2(a^2-1)+1)^2}$

a meaning roughness, but I also use $D(h) = \frac{a_p+2}{2\pi}(\boldsymbol{n}\cdot\boldsymbol{h})_+^{a_p}$,the visualized image is almost the same.

Term F : the equation to calculate F remains the same, but I modify the calculation of F0

```
vec3 F0 = mix(vec3(0.04), albedo, metallic);
```

Term G :I did not modify the equation and the result is just fine.
PBR specular term :

```
vec3 specular = F * Dh * Glv / ((4 * ndotl * ndotv ) + 0.0001) + Ldiffuse;
```

I add a fairly small ϵ to avoid d divisions with zero.
As to the final color,I multiply Lambient with 0.001 to strengthen the effect of metal reflections in the scene .

1.3 Alpha Masking

before

after

1.4 Normal mapping

before

After

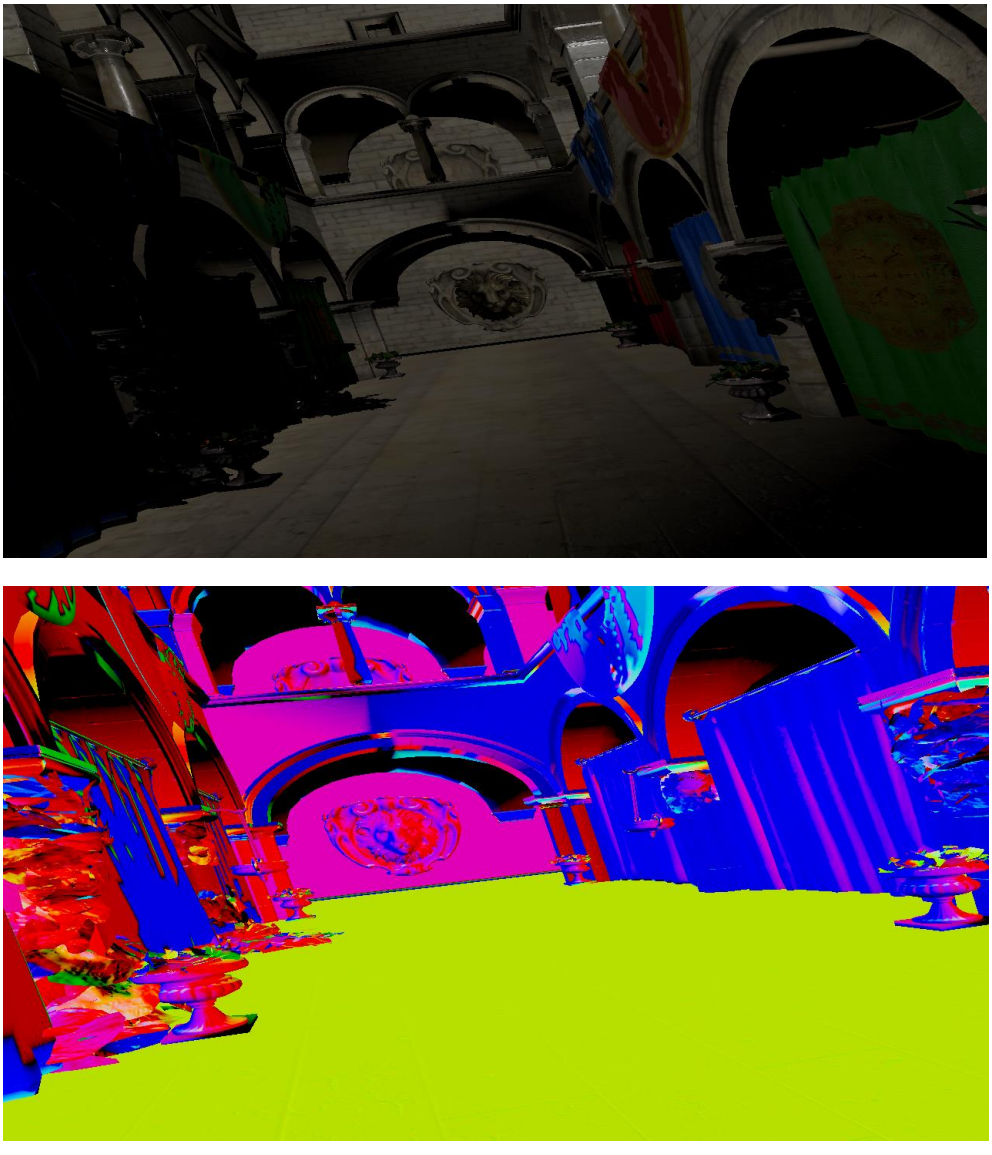
It is not always required,tgen produces a vec4 tangent because it includes a fourth component that indicates whether the TBN frame is mirrored. This is useful when working with mirrored textures or normal maps that have been flipped, as it ensures that the normal map is correctly aligned with the tangent space. However, in some cases, this fourth component may not be required, such as when using normal maps that are guaranteed to not be mirrored.

1.5 Mesh data optimizations

I chose to use R10G10B10A2 (VK_FORMAT_R32_UINT) encoding to store my quaternion. Before, I needed to use 3 normals and 4 tangents per vertex, but now I only need one unsigned integer to store this data, resulting in an 86% reduction in memory usage. However, it's important to note that this encoding may not always be worthwhile, as some parts of the model may lose details as a consequence.

To decode the TBN frame, I chose to do it in the vertex shader because it can reduce the amount of data that needs to be transferred and result in faster processing of the data, as it only needs to be done once per vertex.

While the use of R10G10B10A2 encoding can be useful for reducing memory usage, it can also lead to a loss of accuracy, as seen in the resulting image. Therefore, the decision to use this encoding format should be made with careful consideration of the specific requirements of the application and the desired level of precision and accuracy.



Color and normal visualization after encode and decode process