PONTIFÍCIA UNIVERSIDADE CATÓLICA
DO RIO DE JANEIRO

LUCAS SIMÕES DE ALMEIDA

MATRÍCULA - 1712101

PROJETO 3

RENDERIZAÇÃO DE UMA BOLA DE GOLFE

Relatório da Disciplina Computação INF1761, Segundo Semestre do ano de 2021.

Professor: Waldemar Celes

RIO DE JANEIRO

2021

Neste projeto, tivemos que renderizar uma bola de golfe, sendo que esta aplicação precisava atender a quatro requisitos obrigatórios:

- A esfera deve ser gerada através de um patch, usando tesselation shader
- Deve ser aplicado um mapa de normais para gerar a rugosidade da bola de golfe
- Deve ser aplicado uma textura de cor na bola, junto com a rugosidade
- Deve ser mapeado a reflexão de ambiente na bola, usando um Cube Map.

Após terminar a tarefa 3.3, a que criava um cube map e uma esfera com reflexão de ambiente, comecei a moldar o meu código do projeto 3 com o código usado na tarefa, pois já tinha grande parte do projeto feito só nisso, mas aí que entram algumas dificuldades, na tarefa 3.3, os shaders não estavam aptos a lidar com mapeamento de rugosidade, e nem sequer eram tesselation shaders.

Na tarefa 3.2, eu havia desenvolvida um tesselation shader, porém este programa de shader não conseguia processar mapas de normais, então eu tive que transformar diversas das informações do evaluation shader, em processamento de tangentes, segue a imagem abaixo:

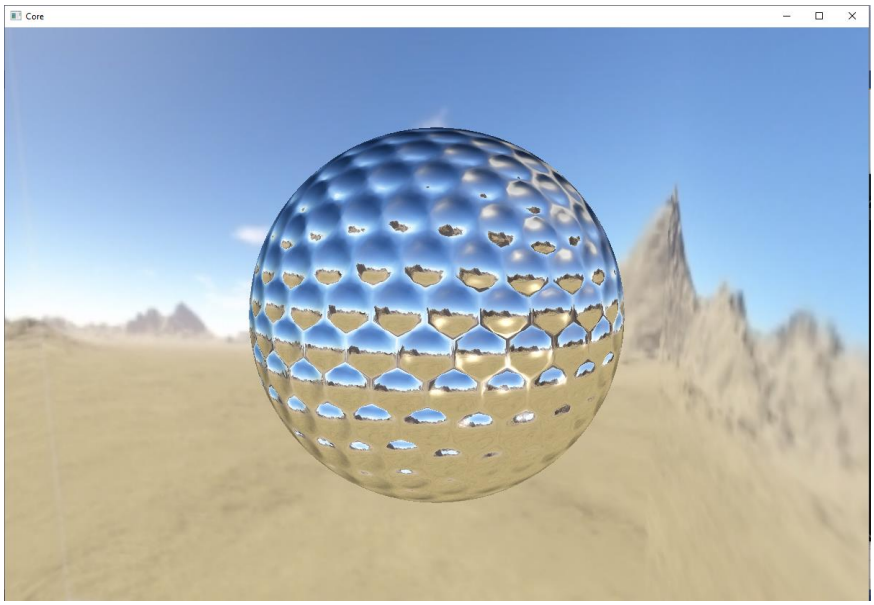
Imagem 1 -

```
uniform vec4 leye; // light pos in eye space
uniform mat4 mvp;
uniform mat4 inv;

patch in vec3 pos;
patch in float radius;

out data{
    vec3 veye;      //Bump -> vtan
    vec3 light;     //Bump ->ltan
    vec2 texcoord;
}v;

void main(void)
{

    float theta = 2*pi*gl_TessCoord.x;
    float phi = pi*gl_TessCoord.y;
    vec4 vpos;
    vpos.x = pos.x + radius*sin(theta)*sin(phi);
    vpos.y = pos.y +radius*cos(phi);
    vpos.z = pos.z + radius*cos(theta)*sin(phi);
    vpos.w = 1.0f;

    vec3 n;
    n.x=sin(theta)*sin(phi);
    n.y=cos(phi);
    n.z=cos(theta)*sin(phi);

    vec3 t;
    t.x = pos.x + (2 * pi * cos(2 * pi * theta) * sin(pi * phi)); //(2 * pi * radius * cos(2 * pi * theta) * sin(pi * phi))
    t.y = pos.y + 0;
    t.z = pos.z + (-2 * pi * sin(2 * pi * theta) * sin(pi * phi)); //(-2 * pi * radius * sin(2 * pi * theta) * sin(pi * phi))

    vec3 b = cross(n,t);

    mat3 base = transpose(mat3(t,b,n));
```

Estes cálculos eram feitos na main, mas agora estão sendo todos feitos diretamente no shader.

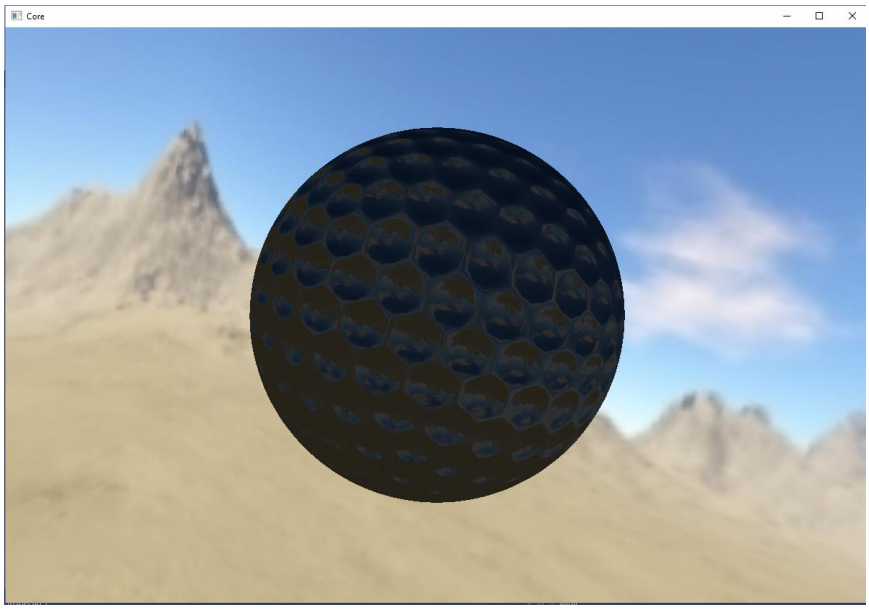
Agora que consegui obter as tangentes, e já possuo as normais, que são dadas pela imagem do mapeamento de rugosidade, temos como criar uma esfera com rugosidade, que reflete o ambiente do cubo e sendo gerada por um patch através de um tesselation shader.

Imagem 2 -



A maior dificuldade durante este projeto foi à adaptação dos shaders para processar corretamente o mapeamento rugosidade, sem falar dos cálculos de iluminação e reflexão, um problema que gostaria de destacar foi o seguinte: durante um bom tempo, na criação do projeto, a textura era refletida corretamente quando o objeto estava sendo iluminado, porém na parte escura do objeto, a textura era refletida invertida, segue abaixo uma imagem do evento:

Imagem 3 -



Gostaria de agradecer ao professor **Waldemar Celes**, pela oportunidade e por estar sempre disponível para esclarecer dúvidas pontuais.