Visualização de Informação - Guião Nº 3

Three.js - Introdução

Miguel Filipe Rodrigues Almeida de Matos Fazenda - N. 110877

Universidade de Aveiro

30 de janeiro de 2022

Resumo

Este relatório visa documentar o processo de realização do guião.

O guião dá continuidade às funcionalidades abordadas no guião anterior, sendo elas:

- texturas;
- interação entre texturas e iluminação;
- interação com o teclado;

Keywords
texturas, *iluminação*, *teclado*

1 Introdução

O **Three.js** é uma biblioteca e API cross-browser para **JavaScript** utilizada para **criar** e **visualizar** gráficos 3D animados num browser, com recurso a WebGL.

De maneira a facilitar a familiarização com a biblioteca, foram feitos vários exercícios com o intuito de aprender e solidificar os básicos da biblioteca.

2 Three.js

As ferramentas que permitiram a realização dos exercícios propostos neste guião foram:

- "TextureLoader", permite carregar imagens para serem itulizadas como texturas;
- geometrias, objetos que permitem definir o formato 3D;
- materiais, objetos que permitem definir estética do modelo 3D. Alguns tipos podem interagir com a luz;
- luzes, fontes de luz que permitem interagir com os materias dos objetos criados;
- Object3D, classe base de todas as geometrias que facilita o controlo sobre as mesmas de diversas maneiras;

3 Ferramentas e Bibliotecas

Ferramentas

- VS Code, editor de texto que fornece todas as funcionalidades necessárias para criar o código;
- browser, para podermos visualizar os exercícios;

Bibliotecas

- Three.js, biblioteca que permite criar os gráficos 3D;

4 Código

O código realizado é referente a 3 exercícios, cada um deles abordando um tema específico do guião.

4.1 1º Exercício

O 1º exercício consiste na aplicação de uma imagem a uma figura 3D. Este objetivo é atingido através de um "TextureLoader", ferramenta do Three.js que permite carregar imagens. A imagem depois é atribuida a um material(MeshBasicMaterial, MeshPhongMaterial, etc).

```
var texloader = new THREE.TextureLoader();
var tex=texloader.load("../img/lena.jpg");

const material = new THREE.MeshBasicMaterial( { color: 0xffffff, wireframe: false, map: tex} );
```

Figura 1: TextureLoader

O material depois é aplicado a uma geometria, uma PlaneGeometry neste contexto. As dimensões da imagem não se adaptam à superfície da geometria, pelo que se a geometria for muito grande, a imagem ficará esticada.

Figura 2: imagem normal

Figura 3: imagem esticada

4.2 2º Exercício

O segundo exercício aborda uma aplicação mais elaborada de imagens a meteriais. O objetivo é criar a geometria de um cubo e aplicar uma imagem diferente a cada face. De maneira a atingir este objetivo, é criado um array de materiais com a quantidade de faces da geometria alvo. Cada elemento do array é um material com uma imagem carregada através de um "TextureLoader".

```
const materials = [];
materials.push(new THREE.MeshBasicMaterial({ map: new THREE.TextureLoader().load('../img/Im1.jpg') }));
materials.push(new THREE.MeshBasicMaterial({ map: new THREE.TextureLoader().load('../img/Im2.jpg') }));
materials.push(new THREE.MeshBasicMaterial({ map: new THREE.TextureLoader().load('../img/Im3.jpg') }));
materials.push(new THREE.MeshBasicMaterial({ map: new THREE.TextureLoader().load('../img/Im4.jpg') }));
materials.push(new THREE.MeshBasicMaterial({ map: new THREE.TextureLoader().load('../img/Im5.jpg') }));
materials.push(new THREE.MeshBasicMaterial({ map: new THREE.TextureLoader().load('../img/Im6.jpg') }));

const cube = new THREE.Mesh( geometry, materials );
```

Figura 4: array de materiais

De maneira a verificar-mos os resultados, que podem ser vistos nas seguintes imagens, a geometria é vista através do controlo "OrbitControl", controlo que permite que uma câmera gire em torno da origem.

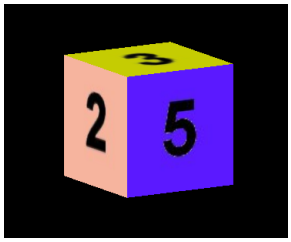

Figura 5: ponto de vista A

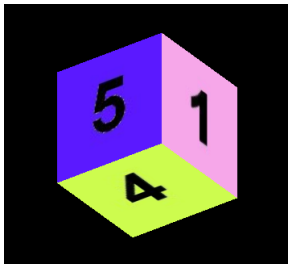

Figura 6: ponto de vista B

4.3 3º Exercício

O terceiro exercício visa combinar a interação entre imagens carregadas, com iluminação e interação com o teclado.

Face aos objetos criados para interação com o teclado, foram criados:

- SphereGeometry com coordenadas (0, 0, 0), caraterísticas (1, 32, 32) e material com uma imagem da terra;

- SphereGeometry com coordenadas $(\sqrt{178200},\ 0,\ -\sqrt{178200})$, caraterísticas$(1,\ 32,\ 32)$, rotação$(0.0089,\ \pi,\ 0)$ radianos e material com uma imagem da lua;
- OrtographicCamera e OrbitControl para obter vários pontos de vista das figuras;
- DirectionalLight com coordenadas (0, 0, 0) e direção para um "pivot", um Object3D com coordenadas(1, 0, 0) para direcionar a luz;

Face à interação com o teclado, esta é feita através de um "EventListener", função executada num dado evento, que permite realizar as seguintes interações:

- alterar a intencidade da luz através das teclas "+" e " -";
- ativar e desativar a luz ao pressionar a tecla "l";
- alterar a rotação de ambas as esferas, com centro na terra, através das teclas "seta direita"e "seta esquerda";
- alterar o ângulo feito entre as esferas, com centro na terra, e o eixo Z;

```
light_flag = true;
function onDocumentKeyDown(event){
    //get tge key code of the pressed key
    var key_code = event.which;
    console.log("tecla " + key_code);

    if (key_code == 76)
        if (light_flag){
            light_flag = false;
            scene.remove(directionalLight);
        }
        else{
            light_flag = true;
            scene.add(directionalLight);
        }

    // if (key_code == )  173-  171+
    if (key_code == 171)
        directionalLight.intensity += 0.1;

    if (key_code == 173)
        directionalLight.intensity -= 0.1;

    if (key_code == 37)  // left
        y_rotation -= 0.0025;

    if (key_code == 39)  // right
        y_rotation += 0.0025;

    if (key_code == 38)  // up
        sphere.rotation.z += 0.41;

    if (key_code == 40)  // down
        sphere.rotation.z -= 0.41;

}
```

Figura 7: definição do EventListener

A SphereGeometry que representa a lua tem como "parente" a SphereGeometry que representa a terra. Esta relação é possível porque as geometrias existentes são classes derivadas da "object3D". Desta maneira, as alterações aplicadas à terra relacionadas a movimentação, rotação e posição, também serão aplicadas à lua.

O resultado final, pode ser visto na seguinte imagem, embora recomenda-se a interação direta para experimentar a interação com o teclado.

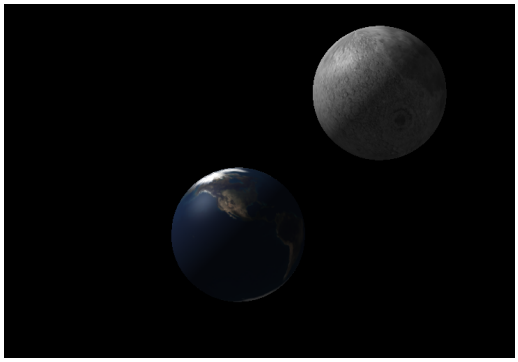

Figura 8: terra e lua

Conclusão

Após a realização destes exercícios podemos afirmar que foram adquiridas competências relacionadas com:

- aplicação de texturas baseadas em imagens a materiais;
- interação entre texturas e iluminação
- interação com o teclado;