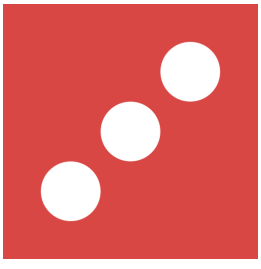
Relatório 4ª Entrega CG

Discente: Henrique Dias, 89455
Docente: Prof. João Brisson
13 de novembro de 2019

O trabalho foi desenvolvido em 6 diferentes classes, maioritariamente estendendo a classe `THREE.Object3D` da biblioteca em uso:

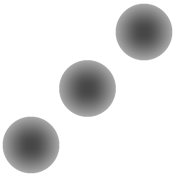
- `Mesh`, extende a classe `THREE.Mesh` e permite a utilização de dois tipos de materiais diferentes (`Basic` e `Phong`, ou seja, sem cálculo de luz e com cálculo de luz). Possui dois métodos:
  - `toggleLight`, para desligar ou ligar o cálculo da luz;
  - `toggleWireframe`, para alternar entre vista de arames ou não.
- `Ball`, recebe dois argumentos: raio e distância ao centro do tabuleiro. Possui um método para inverter a aceleração (`toggleAcceleration`) e um para tratar das animações (`animate`). Representa a bola.
- `Chess`, recebe como argumento o comprimento de um dos lados do tabuleiro e representa o tabuleiro de xadrez.
- `Die`, recebe como argumento o comprimento de um dos lados do dado e representa o dado. Possui um método para tratar das animações (`animate`).
- `PauseScene`, representa a cena de pausa com a sua própria câmara. É um *viewport* diferente usado somente quando o jogo está em pausa. Possui um método `resize` para tratar do evento de redimensionamento da janela.
- `MainScene`, representa a cena de jogo principal, onde tudo é construído. Possui também vários métodos para efetuar diversas ações:
  - `resize`, para redimensionar a cena de acordo com as mudanças existentes no tamanho da janela;
  - `animate`, para efetuar as animações a cada ciclo do requestAnimationFrame;
  - `toggleLight`, para ligar ou desligar o cálculo de luz;
  - `toggleDirectionalLight`, para ligar ou desligar a luz direcional;
  - `togglePointLight`, para ligar ou desligar a luz pontual;
  - `toggleWireframe`, para alternar entre vista de arames ou normal;
  - `toggleBall`, para parar ou continuar o movimento da bola;
  - `togglePause`, para entrar ou sair do modo de pausa.

Relativamente ao **dado**, fiz uma textura, um mapa de transparências e um mapa de alturas (*bump map*). O objetivo foi criar um dado vermelho, com bolas brancas, semi-transparente.



Textura

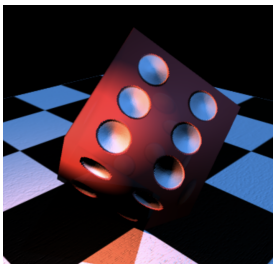
Mapa de Transparências



Mapa de Alturas

O mapa de transparências considera preto totalmente transparente e branco totalmente opaco. Neste caso, as bolas são opacas e o resto do dado é semi-transparente.

Relativamente ao mapa de alturas, quanto mais perto do preto puro, maior a altura. Coloquei altura nas bolas, sendo a altura maior quanto mais perto do centro do círculo.



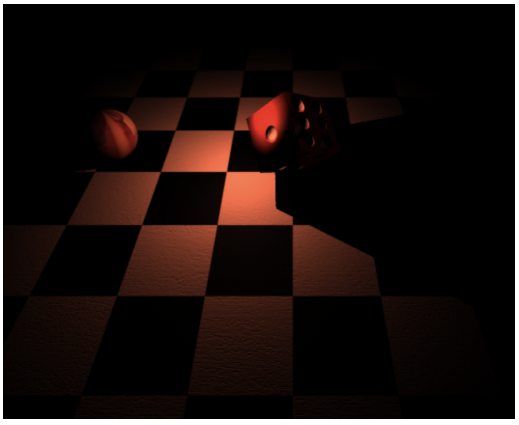
Para a rotação, rodei o dado em 45º nos eixos *x* e *z*, sendo a rotação sobre si mesmo feita segundo o eixo dos *yy* a cada ciclo do `requestAnimationFrame`.

Além disso, também coloquei um pouco de brilho no dado (`shininess`). O resultado do dado pode ser visto na imagem ao lado.

Relativamente à **bola,** decidi usar a Lena como textura. A bola gira em torno do dado, a uma determinada distância, sendo a posição calculada através da distância ao centro (dada como argumento inicialmente) e o coseno (para o eixo dos *xx*) e o seno (para os eixos dos *yy*) do ângulo dado pela aceleração, o valor do `Clock` to `THREE.js` e a velocidade atual.

A bola tem também uma componente especular e pode ser parada (ou recomeçado o movimento) recorrendo à tecla B. Quando se clica na tecla B, a aceleração é invertida, levando a que esta pare ou inicie movimento. O resultado da bola pode ser visto na foto do lado esquerdo, sem cálculo de luz.

O **tabuleiro de xadrez** foi criado da mesma forma: recorrendo a uma textura repetida 4 vezes e a um mapa de alturas para representar madeira.



A **cena principal** tem um tabuleiro de xadrez, uma bola e um dado. A luz direcional foi criada recorrendo à classe `THREE.DirectionalLight` tendo esta uma coloração mais azulada, podendo ser ligada ou desligada recorrendo à tecla D. Além disso, emite sombras.

Já a luz pontual, com uma aparência mais vermelha de forma a poder distinguir as duas, foi criada recorrendo à classe `THREE.PointLight`. É possível visualizar na imagem ao lado, a luz pontual ilumina parcialmente a cena, sendo possível ver o dado, a sua sombra, a bola e o tabuleiro de xadrez. Pode ser desligada ou ligada recorrendo à tecla P.

A câmara da cena principal é uma câmara de perspetiva que permite ver o tabuleiro inteiro. É, também, interativa, recorrendo aos `OrbitControls` do `THREE.js`.

A **cena da pausa** é constituída por um paralelepípedo texturizado com uma imagem que mostra um ecrã de pausa. Esta cena tem uma câmera ortogonal que permite visualizar toda a imagem.

Quando se coloca em pausa (tecla S), a cena principal é suspendida, ou seja, para-se todas as suas animações, e mostra-se a cena de pausa através da sua câmara ortogonal.

Ao fazer reset, clicando na tecla R, simplesmente é criada uma nova cena principal recorrendo à classe `MainScene`, o que faz um *reset* completo a todas as interações previamente feitas.

Quando a janela é redimensionada (evento `resize`), ativo uma *flag* que é utilizada na próxima iteração de `requestAnimationFrame`. Caso tenha sido redimensionada, chamo o método `scene.resize` que efetua os novos cálculos para as câmaras. Da mesma forma faço todas as ações relativamente a outras teclas (evento `keyup`).