Relatório 2ª Entrega CG

Discente: Henrique Dias, 89455
Docente: Prof. João Brisson
24 de outubro de 2019

O trabalho foi desenvolvido em 4 diferentes classes, maioritariamente estendendo a classe `THREE.Mesh` da biblioteca em uso:

- `Ball`, usava para as bolas. Todas possuem um vetor de direção e uma determinada velocidade, calculada aleatoriamente aquando do seu disparo.
- `Cannon`, usado para os três canhões.
- `Field`, para o recinto para onde são projetadas as bolas.
- `Wall`, para cada parede do recinto.

A cena global é composta, inicialmente, por um `Field`, três `Cannon` e um número aleatório entre 2 e 8 de `Ball`s. As bolas encontram-se inicialmente paradas.

Todas as rotações e translações de objetos são realizadas através de matrizes. Foram criadas quatro funções para construir uma matriz relativamente a cada tipo de movimento. Considerando o referencial do THREE.js e a um ângulo, estas foram as matrizes que usei para rotações:

cos(a)	0	sin(a)	0
0	1	0	0
-sin(a)	0	cos(a)	0
0	0	0	1

Rotação em Y

1	0	0	0
0	cos(a)	-sin(a)	0
0	sin(a)	cos(a)	0
0	0	0	1

Rotação em X

cos(a)	-sin(a)	0	0
sin(a)	cos(a)	0	0
0	0	1	0
0	0	0	1

Rotação em Z

Relativamente a uma translação x, y, z, temos a seguinte matriz:

1	0	0	x
0	1	0	y
0	0	1	z
0	0	0	1

Para a deteção de colisões **bola-parede**, utilizei Bounding Boxes. Quando uma bola colide com uma parede, reflete-se a direção da bola na direção da normal à parede.

Já nas colisões **bola-bola**, uso a forma da esfera, verificando se a distância ao quadrado entre duas esferas é inferior ou igual ao diâmetro da bola ao quadrado. Se for, existe colisão. Caso contrário, não há. Para resolver este tipo de colisões elásticas, troco a velocidade das bolas uma com a outra, bem como a sua direção.

A velocidade das bolas é decrementada continuamente num movimento retilineo uniformemente retardado.

No caso de uma bola sair do campo, esta entra em queda infinita, sendo removida da cena 5 segundos depois. As bolas disparadas pelos canhões que não acertam dentro do campo entram, também, em queda infinita.

Todas as animações são realizadas através de `requestAnimationFrame`, sendo que existe um objeto global que indica quais as teclas a ser pressionadas no momento e se a janela foi redimensionada. Estes valores são obtidos através dos eventos `keyup` e `resize`, respetivamente.

Relativamente às câmaras, os seus tamanhos (largura e altura) são calculados através do *aspect ratio* que pretendo neste caso (16 : 9) por forma a adaptar-se à maioria dos ecrãs, não tendo um valor fixo. Assim, em ecrãs que possuam um *aspect ratio* incompatível, os tamanhos da câmara ajustam-se ao mesmo, permitindo a correta visualização de toda a cena.

A terceira câmara (móvel) é colocada sempre atrás da última bola disparada (ou de uma outra bola aleatoriamente caso nenhuma tenha sido disparada ainda **ou** a bola disparada tenha caído para o infinito).