# Estruturas

Nesta última fase decidimos acrescentar algumas estruturas ao nosso código, bem como atualizar algumas das restantes por forma a corresponder corretamente aos objetivos definidos. De seguida vamos apresentar as alterações às estruturas existentes e o porquê de termos realizado essas alterações, no final apresentamos o diagrama de classes do nosso projeto.

## Luz

Por forma a introduzir iluminação no nosso projeto do sistema solar optamos por criar uma classe que corresponde a uma fonte de luz: a classe Luz. No entanto e visto que existem diferentes tipos de fontes de luz e com caraterísticas diferentes decidimos que esta classe Luz seria “abstrata” e que seria realizada em cada uma das suas subclasses: LuzPontual, LuzDirecional e LuzFoco.

No que diz respeito à classe Luz decidimos colocar algo que fosse comum a todas as luzes, ou seja a posição (tanto seja um ponto como um vetor no caso da luz direcional), as diferentes componentes ambiente, especular e difusa da luz que vai incidir nos objetos. Decidimos também ter um número entre 0 e 7, que corresponde às 8 luzes permitidas pelo OpenGL.

No que diz respeito às componentes que correspondem “à cor da luz” decidimos inicializar as mesmas com os valores por defeito para as luzes 1 a 7, sendo que caso seja a luz 0 estes valores são atualizados. Para além disso apenas consideramos estes valores para estas componentes, visto não deixarmos que o utilizador passe o valor dos mesmos.

class Luz {

private:

float posicao[4];

float ambiente[4] = {0,0,0, 1};;

float difusa[4] = {0,0,0, 1};;

float especular[4] = {0,0,0, 1};;

int numero;

public:

Luz();

Luz(int, float[4]);

GLfloat\* getPosicao();

GLfloat\* getAmbiente();

GLfloat\* getDifusa();

GLfloat\* getEspecular();

int getNumero();

virtual void desenhaLuz() = 0;

};

Luz::Luz(int nr, float pos[4]) {

if(nr == 0){

for(int i = 0; i < 3; i++){

difusa[i] = 1;

especular[i] = 1;

}

}

numero = (nr > 7 ? 7 : nr);

for(int i = 0 ; i < 4; i++){

posicao[i] = pos[i];

}

}

## Luz Pontual

No que diz respeito a uma luz pontual, como já foi referido anteriormente, o único parâmetro necessário a ser passado para além dos utilizados na classe Luz é o valor da atenuação. Aqui devemos referir que apenas permitimos atenuações lineares, visto que as constantes deixamos com o seu valor por defeito 1 e as quadráticas optamos também por não alterar.

private:

float atenuacao;

public:

LuzPontual();

LuzPontual(float[4],int,float);

void desenhaLuz();

};

A função desenhaLuz() apenas tem de desenhar cada componente da luz que definimos para esta classe.

void LuzPontual::desenhaLuz(){

int luzA = this->getNumero();

if(luzA>7){

luzA = 7;

}

// posicao

glLightfv(GL\_LIGHT0 +luzA, GL\_POSITION, this->getPosicao());

// cores

glLightfv(GL\_LIGHT0 +luzA, GL\_AMBIENT, this->getAmbiente());

glLightfv(GL\_LIGHT0 +luzA, GL\_DIFFUSE, this->getDifusa());

glLightfv(GL\_LIGHT0 +luzA, GL\_DIFFUSE, this->getEspecular());

//atenuacao

glLightf(GL\_LIGHT0 +luzA,GL\_LINEAR\_ATTENUATION,atenuacao);

}

## Luz Direcional

A classe LuzDirecional ainda é mais simples do que a anterior visto que não tem qualquer atributo para além dos identificados na sua superclasse.

private:

public:

LuzDirecional();

LuzDirecional(float[4],int);

void desenhaLuz();

};

Aqui só devemos referir que ao contrário dos outros casos, o valor das posições tem um valor 0 na última posição indicando que é um vetor.

O processo de desenho da luz é semelhante, sendo que apenas não tem a instrução da atenuação.

## Luz Foco

A classe LuzFoco corresponde a uma luz do tipo SPOT pelo que necessita de mais atributos dos presentas na sua superclasse. Como referimos anteriormente, necessita de um ângulo correspondente ao ângulo de propagação da mesma, de uma direção da propagação (um vetor), de uma atenuação (mais uma vez apenas definimos a linear) e de um expoente correspondente à intensidade de distribuição da luz.

class LuzFoco : public Luz {

private:

float angulo;

float direcao[3];

float atenuacao;

float expoente;

public:

LuzFoco();

LuzFoco(float[4], int,float,float[3],float,float);

void desenhaLuz();

};

Na função que “desenha” a luz apenas tem de acrescentar as componentes relativas à luz do tipo SPOT como mostramos de seguida:

glLightfv(GL\_LIGHT0 +luzA, GL\_SPOT\_DIRECTION, direcao);

glLightf(GL\_LIGHT0 +luzA, GL\_SPOT\_CUTOFF, angulo);

glLightf(GL\_LIGHT0 +luzA,GL\_SPOT\_EXPONENT, expoente);