



UNIVERSIDADE DO MINHO

LICENCIATURA EM CIÊNCIAS DA COMPUTAÇÃO

Computação Gráfica

Phase 4 - Normals and Textures Coordinates

Grupo Nº 1

| | | | |
|----------------|-----------------|-----------------|--------------|
| Bruna Carvalho | Carlos Beiramar | Daniel Ferreira | Ricardo Cruz |
| (a87982) | (a84628) | (a85670) | (a86789) |

7 de junho de 2021

Conteúdo

| | | |
|----------|---------------------------------------|-----------|
| 1 | Introdução | 3 |
| 2 | Alterações no <i>Generator</i> | 4 |
| 2.1 | Normais | 4 |
| 2.1.1 | Normais para o Plano | 5 |
| 2.1.2 | Normais para a Caixa | 5 |
| 2.1.2.1 | Topo | 5 |
| 2.1.2.2 | Base | 5 |
| 2.1.2.3 | Frente | 6 |
| 2.1.2.4 | Atrás | 6 |
| 2.1.2.5 | Esquerda | 6 |
| 2.1.2.6 | Direita | 6 |
| 2.1.3 | Normais da Esfera | 7 |
| 2.1.4 | Normais do Cone | 7 |
| 2.1.5 | Normais do Patch de Bezier | 8 |
| 3 | Alterações no <i>Engine</i> | 9 |
| 3.1 | Iluminação | 9 |
| 3.2 | Texturas | 10 |
| 4 | Sistema Solar | 11 |
| 5 | Conclusão | 12 |

Lista de Figuras

| | | |
|------|--|----|
| 2.1 | Função <i>normalize</i> | 4 |
| 2.2 | Normais para o plano | 5 |
| 2.3 | Normais para a face do topo | 5 |
| 2.4 | Normais para a face da base | 5 |
| 2.5 | Normais para a face da frente | 6 |
| 2.6 | Normais para a face atrás | 6 |
| 2.7 | Normais para a face da esquerda | 6 |
| 2.8 | Normais para a face da direita | 6 |
| 2.9 | Normal para os pontos da base do cone | 7 |
| 2.10 | Calculo normal para patch de Bezier | 8 |
| 3.1 | Lights | 9 |
| 3.2 | GLfloat | 10 |
| 3.3 | <i>Model</i> do Sol | 10 |
| 3.4 | Exemplo da implementação da textura no XML | 10 |
| 4.1 | Sistema Solar com as órbitas | 11 |
| 4.2 | Sistema Solar sem órbitas | 11 |

Capítulo 1

Introdução

O objetivo desta última fase passa por fazer alterações de modo a permitir que nossa aplicação aceite texturas e iluminação. Para isso, foi necessário fazer alterações no ***Generator*** e no ***Engine***.

Na aplicação *generator*, ao gerar os modelos, passou-se a guardar também em ficheiro as normais e texturas de cada ponto, de modo a permitir o uso de iluminação e texturas posteriormente.

No que diz respeito ao *engine*, foi necessário acrescentar código que permita ao ***OpenGL*** reconhecer a iluminação e as texturas, que foram acrescentadas no ficheiro ***XML***.

Nesta fase, recorreu-se também ao uso de 3 VBO's, nomeadamente para as coordenadas dos vértices, para as normais e para as coordenadas de textura. O resultado final desta fase, será apresentar uma "cena" de um sistema solar dinâmico (da fase anterior) com o acréscimo de iluminação e de texturas.

Capítulo 2

Alterações no *Generator*

2.1 Normais

Para auxiliar a normalização dos vetores, recorreu se à seguinte função utilizada também nas aulas práticas.

- Função *normalize*

```
void normalize(Vertex n) {  
    float l = sqrt(n.x * n.x + n.y * n.y + n.z * n.z);  
    n.x = n.x / l;  
    n.y = n.y / l;  
    n.z = n.z / l;  
}
```

Figura 2.1: Função *normalize*

2.1.1 Normais para o Plano

Como no nosso trabalho um plano está contido no plano XZ, as normais de um conjunto de pontos, que estão no mesmo lado do plano, são iguais para todos. Quanto às normais dos pontos do lado oposto são exatamente o simétrico das primeiras normais. Dessa forma as normais são as seguintes:

- Normais para o plano

```
Vertex norm1(0.0f, 1.0f, 0.0f);  
Vertex norm2(0.0f, -1.0f, 0.0f);
```

Figura 2.2: Normais para o plano

2.1.2 Normais para a Caixa

Para a caixa, vamos ter normais diferentes para cada face da mesma. De notar que, todos os pontos que pertençam a uma mesma face, terão as mesmas normais.

2.1.2.1 Topo

- Normais para a face do topo

```
Vertex norm2(0.0f, 1.0f, 0.0f);
```

Figura 2.3: Normais para a face do topo

2.1.2.2 Base

- Normais para a face da base

```
Vertex norm1(0.0f, -1.0f, 0.0f);
```

Figura 2.4: Normais para a face da base

2.1.2.3 Frente

- Normais para a face da frente

```
Vertex norm3(0.0f, 0.0f, 1.0f);  
Triangle[...]
```

Figura 2.5: Normais para a face da frente

2.1.2.4 Atrás

- Normais para a face atrás

```
Vertex norm4(0.0f, 0.0f, -1.0f);  
Triangle[...]
```

Figura 2.6: Normais para a face atrás

2.1.2.5 Esquerda

- Normais para a face da esquerda

```
Vertex norm5(-1.0f, 0.0f, 0.0f);  
Triangle[...]
```

Figura 2.7: Normais para a face da esquerda

2.1.2.6 Direita

- Normais para a face da direita

```
Vertex norm6(1.0f, 0.0f, 0.0f);  
Triangle[...]
```

Figura 2.8: Normais para a face da direita

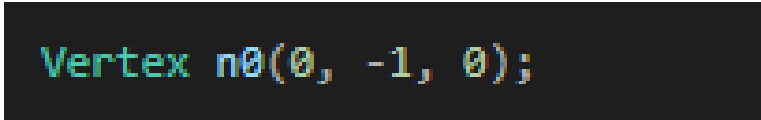
2.1.3 Normais da Esfera

Para determinar as normais dos vértices da esfera foi utilizada a função ***normalize*** anteriormente mencionada. Depois de calculadas as normais, a ordem pela qual são colocadas no ficheiro *.3d* vai de encontro à ordem pela qual os vértices são escritos.

2.1.4 Normais do Cone

O cálculo das normais do cone pode ser dividido em 2 partes. Numa primeira parte calcula-se as normais da base seguindo a mesma ideia das normais da base da caixa. Dessa forma, a normal de cada ponto da base será a mesma:

- Normal para os pontos da base do Cone



```
Vertex n0(0, -1, 0);
```

Figura 2.9: Normal para os pontos da base do cone

Para calcular as normais dos pontos do resto do cone seguiu-se a mesma ideia do que foi feito na esfera, recorrendo à função ***normalize*** aplicada a cada ponto do cone.

2.1.5 Normais do Patch de Bezier

Para calcular a normal de um ponto da superfície foi utilizado a seguinte expressão:

- Calculo normal para patch de Bezier

$$p(u, v) = [u^3 \quad u^2 \quad u \quad 1]M \begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix} M^T \begin{bmatrix} v^3 \\ v^2 \\ v \\ 1 \end{bmatrix}$$

$$\frac{\partial p(u, v)}{\partial u} = [3u^2 \quad 2u \quad 1 \quad 0]M \begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix} M^T V^T$$

$$\frac{\partial p(u, v)}{\partial v} = UM \begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix} M^T \begin{bmatrix} 3v^2 \\ 2v \\ 1 \\ 0 \end{bmatrix}$$

Normal
The normal vector at any point of the surface is defined as the normalized result of the cross product of the tangent vectors.

Figura 2.10: Calculo normal para patch de Bezier

Mas antes de se escrever o resultado no ficheiro, este ainda sofre a aplicação da função ***normalize*** também.

Capítulo 3

Alterações no *Engine*

No *Engine* foram feitas algumas alterações de modo a permitir a iluminação e aplicação de texturas ao modelo e/ou mudança do material através do comando *glMaterialfv*, recorrendo às coordenadas que geramos no *Generator*.

3.1 Iluminação

A partir da *tag lights*, o *Engine*, sabe qual o tipo de luz a ser implementada. Os filhos dessa *tag light* indicam o tipo, isto é, pode ser *POINT*, *DIRETIONAL* ou *SPOT* e, além de mais, indicam, também as coordenadas.

- **POINT** - os raios da luz são emitidos em todas as direções, através de um único ponto.
- **DIRETIONAL** - os raios de luz são todos paralelos entre si.
- **SPOT** - age de forma semelhante

O Sistema Solar, a luz vai ser proveniente do Sol, ou seja, o ponto (0,0,0). Com base neste motivo, esta é a posição dada no início do ficheiro *XML*:

```
<lights>
  <light type="POINT" posX="0.0" posY="0.0" posZ="0.0" att="0.0"/>
</lights>
```

Figura 3.1: Lights

Foi preciso incluir, também, no *Engine* uma luz difusa e uma luz ambiente que é aplicada a todos os elementos do modelo.

```
GLfloat amb[3] = { 0.001, 0.001, 0.001 };  
GLfloat diff[4] = { 1, 1, 1, 1.0 };
```

Figura 3.2: GLfloat

Além de mais, definiu-se os parâmetros RGB das cores das componentes ambiente (ambR, ambG, ambB), difusa (difR, difG, difB), especular (speR, speG, speB) e luz emissiva (emR, emG, emB). Em baixo será apresentado um exemplo de uma **model** em que são atribuídas as componentes da luz emissiva:

```
<model file="sphere.3d" texture="sol.jpg" emR="1" emG="1" emB="1"/>
```

Figura 3.3: *Model* do Sol

3.2 Texturas

As texturas estão presentes no ficheiro XML que é carregado pelo *Engine* para gerar o Sistema Solar. Assim, foi adicionado o elemento **textura** na *model*, exatamente como está representado no exemplo seguinte.

```
<model file="sphere.3d" texture="sol.jpg" emR="1" emG="1" emB="1"/>
```

Figura 3.4: Exemplo da implementação da textura no XML

Para além das texturas para os planetas, das luas, do Sol e do *teapot*, foi adicionada também uma **textura** para o *background* do Sistema Solar.

Capítulo 4

Sistema Solar

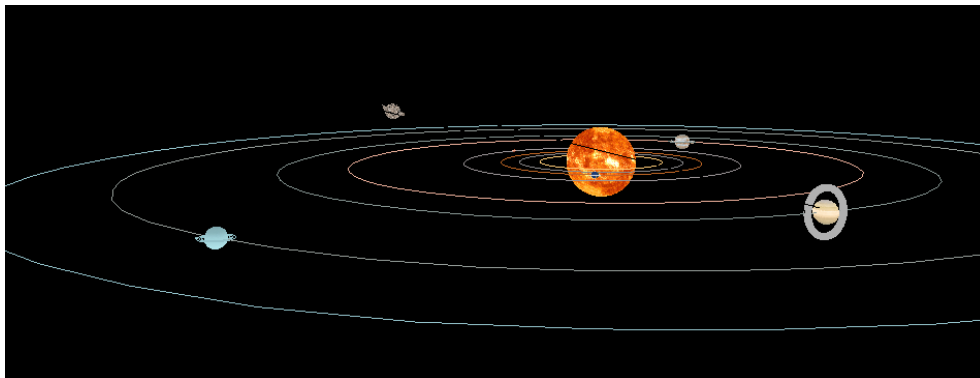


Figura 4.1: Sistema Solar com as órbitas

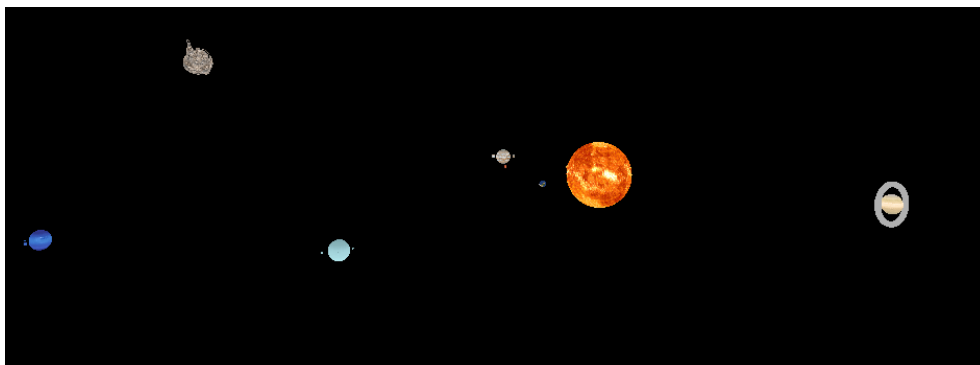


Figura 4.2: Sistema Solar sem órbitas

Capítulo 5

Conclusão

Nesta última fase do projeto, foram implementadas as respetivas texturas de cada um dos planetas e luas e, foi também implementada a luz. Após todas estas fases podemos afirmar que o trabalho tem vindo a ser melhorado de fase para fase, pondo assim em prática os conteúdos leccionados na UC de Computação Gráfica.

Concluimos assim o trabalho que, apesar de termos sentido dificuldade em alguns pontos específicos, achamos que cumpre a maioria dos critérios estabelecidos inicialmente.