Universidade do Minho
Escola de Engenharia

# Trabalho Prático - Grupo 32

# Fase 4 - Normais e Coordenadas de Textura

## Computação Gráfica

Bruno Filipe de Sousa Dias A89583
Luís Enes Sousa A89597
Pedro Miguel de Soveral Pacheco Barbosa A89529

30 de maio de 2021

# Conteúdo

# 1 Introdução

Nesta fase do trabalho prático, de forma a terminar a construção so Sistema Solar, tivemos que implementar iluminação e texturas.

Para tal, foi necessário definir as normais e as coordenadas de textura de cada vértice das figuras que desenhamos.

O cenário final desta fase é um modelo dinâmico do Sistema Solar, com iluminação e texturas.

# 2 Alterações na Estrutura do Projeto

Nesta secção, serão abordadas as alterações que foram efetuadas na estrutura do projeto, para alcançar os objetivos finais.

## 2.1 Generator

Nesta fase, o Generator foi alterado de forma a ser capaz de gerar normais e coordenadas de textura para os vértices de algumas primitivas geométricas, mais especificamente, a esfera e o toro.

### 2.1.1 *generator.cpp*

Neste ficheiro, alteramos a função que escreve os pontos num ficheiro, de forma a escrever, também, e caso estejam definidas, as normais e as coordenadas de textura.

A função que lê os *Bezier patches* de um ficheiro também foi alterada, de forma a calcular as normais dos pontos calculados. Para tal, usamos a função ***calculate_normal(p1, p2, p3)***, que recebe três pontos e calcula o vetor normal, normalizado, do plano formado por estes.

### 2.1.2 *primitives.cpp* e *primitives.h*

Neste ficheiro, alteramos as funções que geram os pontos para a esfera e para o toro.

Para calcular as normais da esfera, apenas temos que normalizar o próprio ponto, pois, sendo que a esfera está centrada na origem, este representa, também, o vetor normal à superfície.

Em relação às coordenadas de textura, decidimos incrementar o $x$ à medida que avançamos nas fatias e o $y$ à medida que avançamos nas camadas.

Em relação à geração do toro, decidimos atribuir o valor 0 de beta ao valor mínimo da coordenada $y$ (anteriormente este representava o máximo desta coordenada). Desta forma facilitamos a atribuição de texturas aos toros.

Para gerar as normais do toro, usamos um princípio semelhante ao da esfera, achando um ponto interior do toro e definindo um vetor desde esse ponto até ao ponto da superfície.

Quanto às coordenadas de textura, o nosso ângulo alpha é mapeado ao longo do $y$ e o ângulo beta do $x$.

## 2.2 Engine

### 2.2.1 engine.cpp

Neste ficheiro, foi adicionado um *vector* para guardar todas as luzes presentes no cenário. Também foi criada uma função para desenhar cada modelo, que testa se esse modelo tem normais e/ou texturas associadas.

### 2.2.2 *group.cpp* e *group.h*

Nestes ficheiros, as funções ***cross*** e ***normalize*** foram movidas para outro ficheiro. A maneira com a trajetória do teapot é desenhada também foi alterada, devido ao uso de iluminação.

### 2.2.3 *lights.cpp* e *lights.h*

Estes ficheiros foram adicionados ao projeto, de forma a processar os diferentes tipo de luz que podem estar presentes no cenário.

Nestes ficheiros, são declaradas três classes distintas que representam três tipos de luz diferentes (POINT, DIRECTIONAL e SPOT). Estas classes possuem atributos iguais, vindos de uma classe base, que guarda os valores ambiente, difuso e especular de cada luz.

No caso da luz POINT, é apenas necessário guardar a posição onde esta se encontra. Em relação à luz DIRECTIONAL, apenas guardamos a direção desta. Quanto à luz SPOT, é necessário guardar a sua posição, direção e ângulo de abertura.

Quanto à função ***apply***, presente nas três classes, esta aplica as componentes ambiente, difusa e especular da luz correspondente. Depois, dependendo do tipo de luz, aplica as caraterísticas necessárias.

### 2.2.4 *model.cpp* e *model.h*

O ficheiro *.h* foi alterado, de forma a guardar os índices das VBO's das normais e das coordenadas de textura e o índice da textura, do modelo em questão. Também foram adicionadas as caraterísticas do modelo em relação à iluminação.

No ficheiro *.cpp*, adicionado nesta fase, encontramos a função ***loadTexture*** que, recebendo um nome de um ficheiro, gera uma textura com o seu conteúdo e guarda-a no modelo correspondente.

### 2.2.5 *parser.cpp* e *parser.h*

Neste ficheiro, a função ***load3dFile*** foi alterada, sendo, agora, capaz de ler normais e coordenadas de textura do ficheiro *.3d* e gerar, caso necessário, as VBO's correspondentes.

De forma a organizar melhor o nosso código, adicionamos, também, uma função para fazer o *parsing* de um atributo de um elemento e várias funções para fazer o *parsing* dos atributos ambiente, difusa, especular e emissive, tanto de uma fonte de luz como de um modelos.

Em relação ao *parsing* de um modelo, agora é possível definir um ficheiro com texturas e as caraterísticas de iluminação do mesmo.

Quanto à iluminação, adicionamos uma função *parseXMLLightElement* que é responsável pelo *parsing* de um elemento *light*, do nosso ficheiro *.XML*.

### 2.3 Utils

#### 2.3.1 *float_vector.cpp* e *float_vector.h*

Estes ficheiros foram adicionados ao projeto, contendo algumas funções para operações entre vetores.

## 3 Alterações na Estrutura do Ficheiro *XML*

Nesta fase, e de forma a suportar as novas funcionalidades, tivemos que alterar, mais uma vez, a estrutura do ficheiro *XML*.

Surge então, o elemento *lights*, que, por sua vez, pode possuir vários elementos *light*.

Cada elemento *light* possui um atributo *type* que indica o tipo de luz descrito. Podem, ainda, ser especificadas as componentes ambiente, difusa e especular da luz. Caso contrário assumem valores padrão. Consoante o tipo de luz definido, podem surgir atributos como a posição, a direção ou o ângulo de abertura da luz.

No elemento *model* podemos encontrar o atributo *texture*, que indica o nome do ficheiro donde se pretende carregar as texturas. Adicionalmente, podemos encontrar atributos relacionados com a iluminação do modelo, nomeadamente a componente ambiente, difusa, especular e emissiva.

## 4 Alterações no Cenário Final

Em relação à fase anterior, há claras melhorias no aspeto geral, devido à introdução de iluminação e texturas.

Todos os nossos modelos possuem texturas próprias, à exceção do cometa e do anel de Urano. Quanto à iluminação, todos eles têm normais definidas, logo são capazes de produzir os efeitos corretos.

Nesta fase decidimos remover as órbitas do ficheiro *.XML* principal, pois causavam efeito de *flickering*. No entanto, criamos um ficheiro secundário com as mesmas, caso o utilizador pretenda usar.

Ainda em relação às órbitas, e de forma a reduzir o flickering e corrigir algumas órbitas quase impercetíveis, deicidimos criar um ficheiro *.3d* para cada órbita, em vez de aplicar o scale numa órbita modelo.

Decidimos, ainda, adicionar o movimento de rotação aos planetas e ao Sol, pois achamos que, com as texturas aplicadas, produz um efeito interessante e agradável.

De maneira a criar o efeito de haver um *background*, juntamos ao nosso cenário uma esfera com raio grande (de forma a englobar todo o Sistema Solar) e aplicamos uma textura de estrelas. Usando um scale com valores negativos, conseguimos "virar"a textura da esfera para "dentro", produzindo, então, a ilusão de haver um *background* com as estrelas.

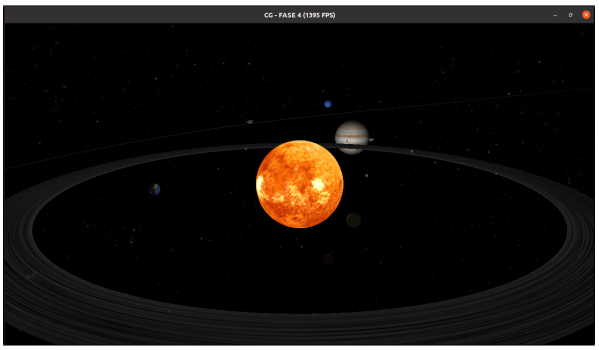

Figura 1: Cenário final - perto

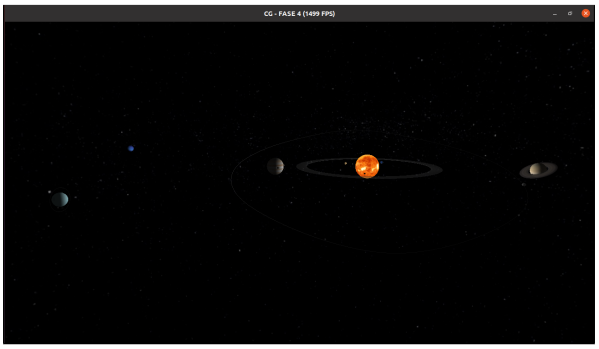

Figura 2: Cenário final - longe

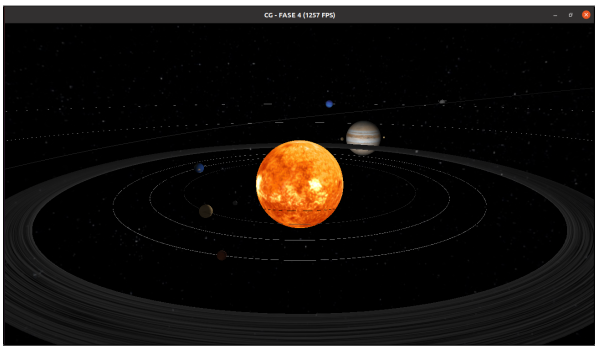

Figura 3: Cenário final com órbitas - perto

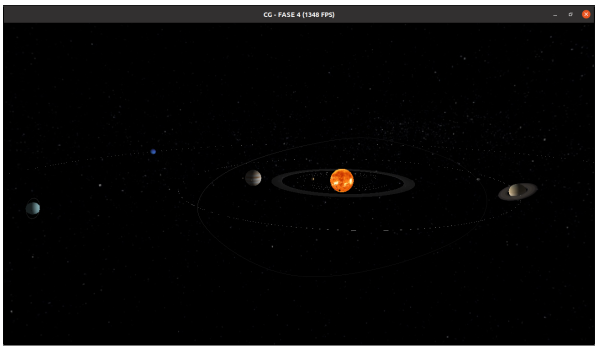

Figura 4: Cenário final com órbitas - longe

# 5 Conclusão

Concluída a quarta e última fase do projeto, achamos que superamos todos os desafios propostos no enunciado, tanto nesta como nas outras fases.

Nesta fase tivemos que gerar modelos com normais e coordenadas de textura associadas, de forma a podermos aplicar iluminação e texturas ao cenário final.

Em geral, sentimos a complexidade do trabalho a aumentar de fase para fase, mas também a nossa motivação, pois cada vez tínhamos um cenário mais interessante e agradavél de observar. Desta forma, podemos concluir que estamos satisfeitos com o resultado final e gostamos de poder pôr em prático os conceitos teóricas abordados nesta unidade curricular.

Em suma, consideramos que este projeto foi desafiante, mas ao mesmo tempo recompensador, pois obtivemos um cenário final muito apelativo.