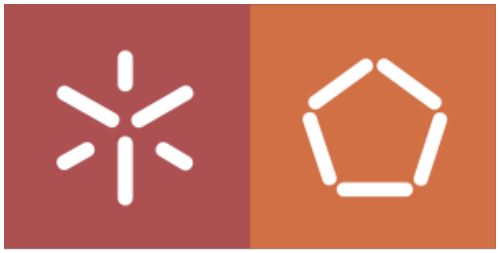

Universidade do Minho
Escola de Engenharia

# Computação Gráfica

## Fase 3 - Curves, Cubic Surfaces and VBOs

**Daniel José Silva Furtado A97327**
**Ricardo Lopes Lucena A97746**
**Ricardo Silva Machado Araújo A96394**
**Nuno Miguel Leite da Costa A96897**

A97327 A97746 A96394 A96897

# Conteúdo

# 1 Introdução

Na fase atual do trabalho prático, demos continuidade às duas fases anteriores, com o objetivo de realizar alterações no Generator e no Engine. Relativamente ao Generator, foi efetuada uma alteração, de modo, a ser possível criar um modelo com Bezier patches, permitindo assim gerarmos superfícies complexas e suaves usando as curvas de Bezier. Por outro lado, o Engine foi modificado para suportar translações e rotações dinâmicas, utilizando curvas de Catmull-Rom para o primeiro caso. Além disso, implementamos a funcionalidade de desenhar modelos com VBOs aumentando drasticamente os *FPS(frames per second)* do projeto.

No fim, foi-nos solicitado apresentar um cenário final que representasse um modelo dinâmico do Sistema Solar, que inclui um cometa que tivesse a trajetória definida por uma curva de Catmull-Rom.

# 2 Arquitetura do Projeto

Na terceira fase deste projeto, a arquitetura permaneceu a mesma, consistindo em duas aplicações principais: o Generator e o Engine. No entanto, foram necessárias alterações significativas em ambas as aplicações para atender às funcionalidades exigidas nesta fase, que serão descritas a seguir.

## 2.1 Generator

Durante esta etapa, realizamos uma atualização no gerador para permitir a conversão de um arquivo no formato de patch de bezier para um arquivo contendo uma lista de pontos necessários para criar triângulos e desenhar a figura 3D. Além disso, agora é possível especificar um nível de tesselação para a conversão, o que permite ajustar a qualidade e a resolução do modelo final.

## 2.2 Engine (*Dynamic Translate* e *Dynamic Rotate)*

Com a adição das translações dinâmicas e as rotações dinâmicas tivemos que alterar o projeto. De forma a conseguir suportar as translações dinâmicas através das curvas cúbicas de Catmull-Rom decidimos alterar a nossa classe *transformations*, adicionando uma class nova *TranslacaoG* que armazena as informações essenciais para criar a curva e calcular o ponto atual da translação. Além disso para suportar as rotações dinâmicas, foi necessário criar outra class *RotacaoG* que realiza o cálculo do tempo decorrido, de forma a obter o ângulo de rotação no momento atual.

### 2.2.1 engine.cpp

Neste ficheiro, tivemos de alterar a função drawGroup(), tornando-a capaz de indentificar estas duas novas transformações. Se uma delas for encontrada, as respetivas funções são invocadas, aplicando a transformação aos modelos correspondentes. Consequentemente, tivemos de alterar a função responsável

pelo parse. Agora nós verificamos se o elemento do XML "Translate"/ "Rotate"contém um atributo time e se conterem sabemos que é uma transformação dinámica.

### 2.3 Engine *(VBOs)*

Nesta fase do trabalho, foi-nos pedido que usassemos VBOs para desenhar os modelos geométricos, ao invés do modo imediato, aumentando assim o desempenho da nossa engine.

#### 2.3.1 engine.cpp

No ficheiro *engine.cpp* houve uma alteração na maneira como fazemos parsing. Na fase anterior, a nossa estratégia era adicionar todos os ficheiros .3d num vector e desenha-los no renderScene com a ajuda da função drawObjects.

Atualmente, sempre que encontramos um modelo nós executamos a função drawModels, tendo esta uma funcionalidade diferente à da fase anterior,

- *cria um vector com as coordenadas dos pontos*
- *cria o VBO*
- *copia o vector para a memória gráfica*

Por fim, retorna um objeto Model. A função parsing adiciona este Model a um vetor que depois é executado na função drawGroups da renderScene, desenhando assim os modelos.

#### 2.3.2 model.h

Este novo ficheiro foi adicionado ao projeto, em que consiste numa nova class chamada *Model*, que armazena as informações necessárias para desenhar um modelo guardado num VBO, nomeadamente o índice desta e o número de vértices.

#### 2.3.3 group.h

Neste ficheiro, a estrutura da classe Group foi alterada. O vector que guarda os modelos do cenário, passa a ter elementos da classe Model em vez de strings.

## 3 Modificação na Estrutura do ficheiro XML

Com a adição de novas funcionalidades foi preciso fazer algumas alterações no arquivo XML para conseguir suportar estas novas adições.

Uma das principais mudanças, acontece no elemento **"translate"** que pode incluir a adição de um novo atributo, o **"time"**. Neste caso, o **"time"** representa o tempo necessário para percorrer toda a curva de Catmull-Rom. Adicionalmente, os atributos (X, Y e Z) foram substituídos por uma sequência

de pontos que compõem a curva. É importante destacar que este elemento agora também possui o atributo **"closed"** que indica se os pontos formam uma curva fechada.

A outra mudança ocorreu no elemento **"rotate"**, em que agora o atributo **"angle"** pode ser substituído pelo atributo **"time"**. Este novo atributo representa o tempo necessário para concluir uma rotação completa, em vez de um ângulo específico.

# 4 Alterações na Câmera

Nesta fase, resolvemos implementar um sistema de câmera fps, através da classe *fpsCamera*, que permite ao utilizador navegar livremente pelo Sistema Solar no plano *xOz* com o teclado e no eixo *Oy* através do rato. Para armazenar o estado atual é necessário ter variáveis tais como: *alpha*, *beta*, *eyeX*, *eyeY*, *eyeZ*, *sensitivity*, *speed*, *startx*, *starty*, *deltax* ,*deltay* e *tracking*; Para que servem estas variáveis:

- alpha : representa a rotação horizontal da câmera em torno de um eixo vertical. Ele determina para qual direção a câmera está apontando no plano horizontal. Um aumento no valor de alpha faz com que a câmera gire para a direita, enquanto uma diminuição no valor de alpha faz com que a câmera gire para a esquerda.
- beta : representa a rotação vertical da câmera em torno de um eixo horizontal. Ele determina o ângulo de inclinação da câmera para cima ou para baixo. Um aumento no valor de beta faz com que a câmera aponte para cima, enquanto uma diminuição no valor de beta faz com que a câmera aponte para baixo.
- eyeX : representa a posição da câmera em relação ao eixoOx;
- eyeY : representa a posição da câmera em relação ao eixoOy;
- eyeZ : representa a posição da câmera em relação ao eixoOz;
- sensitivity : responśaveis por alterar a sensibilidade da câmera ao mover o rato;
- speed : responsáveis por alterar a velocidade da translação da câmera;
- startx e starty : reponsáveis por guardar a posição anterior do rato;
- deltax e deltay : representam as diferenças entre as coordenadas do rato no eixo X e no eixo Y, respectivamente, em relação à posição inicial;
- tracking : é um array de tamanho 2 utilizado para controlar o estado do rastreamento do mouse;

# 5 Cenário Final

Após as alterações realizadas nesta etapa, é evidente a presença de diferenças significativas em relação ao cenário final da etapa anterior. Agora, nesta fase, temos o movimento dos modelos, permitindo visualizar os planetas orbitando o sol e sobre si mesmos, assim como as luas orbitando seus respectivos planetas. Por último, conseguimos verificar um aumento dos fps graças a implementação dos VBOs

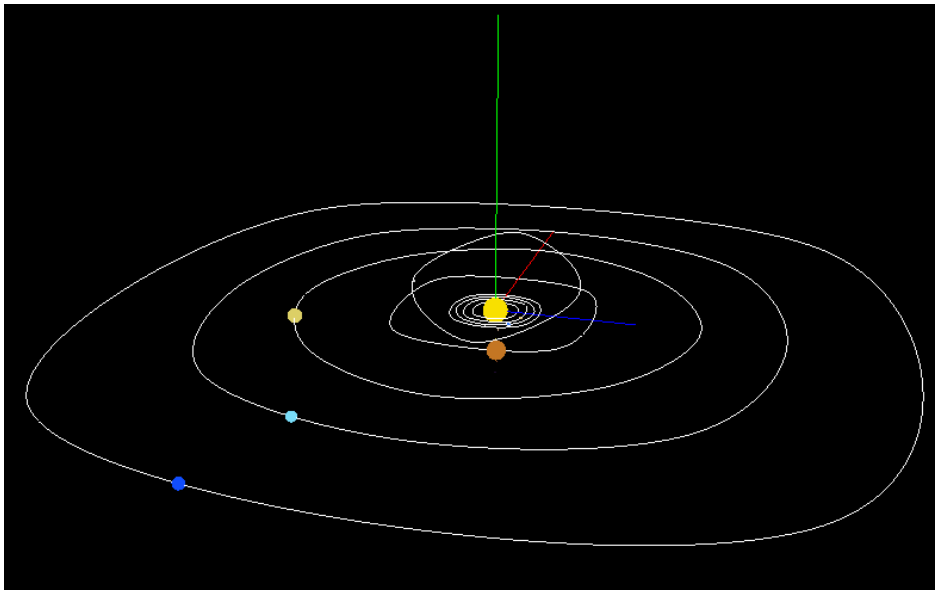

Figura 1: Cenário Final Geral

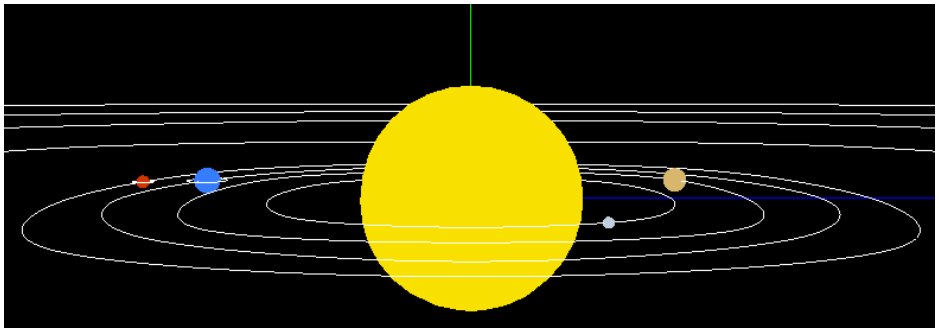

Figura 2: Ãngulo Horizontal da parte inicial do Sistema Solar

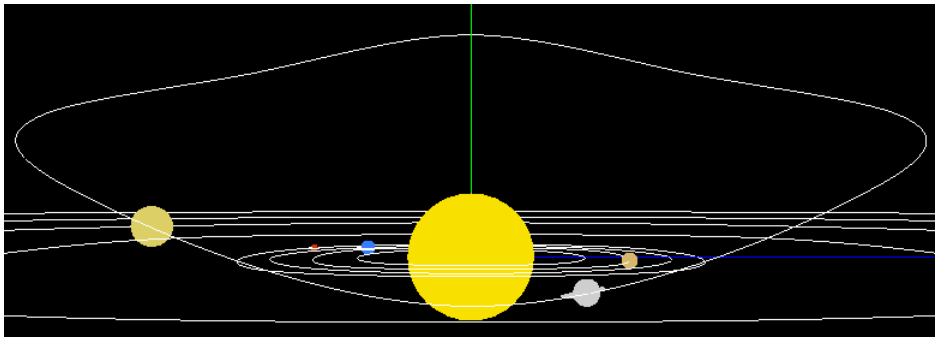

Figura 3: Percurso do Cometa

## 6 Aspetos a melhorar na próxima fase

Na próxima fase, planejamos adicionar um novo modelo para criar os anéis dos planetas, como os de Saturno, a fim de tornar nosso sistema solar o mais realista possível. Além disso, temos o objetivo de tornar o projeto e o código mais estruturado e organizado.

## 7 Conclusão

Após concluirmos a terceira fase do projeto, estamos satisfeitos com a aplicação bem-sucedida dos conceitos teóricos de curvas e superfícies cúbicas. Além disso, tivemos a oportunidade de utilizar VBOs, que já haviam sido introduzidos em uma aula prática anterior. Agora, na próxima etapa do projeto, estamos entusiasmados em expandir ainda mais o nosso trabalho, adicionando recursos de iluminação e texturas ao Sistema Solar.