Escola de Engenharia
Universidade do Minho

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA
Mestrado Integrado em Engenharia Informática
Computação Gráfica

Fase 2
Geometric Transforms

Grupo 18

Célia Figueiredo a67637

Luís Pedro Fonseca a60993

Braga, 26 de Março de 2017

Conteúdo

Resumo

O presente relatório descreve o trabalho realizado no âmbito da Unidade Curricular de Computação Gráfica, onde nos foi pedido a implementação de um projeto em OpenGL com a representação de um Sistema Solar. Este projeto será feito em quatro fases, sendo este relatório correspondente à segunda fase do respetivo projeto: *Geometric Transforms*.

1. Introdução

No âmbito da Unidade Curricular de Computação Gráfica pertencente ao plano de estudos do 3º ano do Mestrado Integrado em Engenharia Informática e no seguimento das fases propostas para a realização do projeto que visa constriur um sistema solar.

Concluída a primeira fase tem-se agora de introduzir transformações geométricas, *translate*, *rotate* e *scale*, ao projeto. Assim, nesta fase, o *engine* deverá ser capaz de realizar translações, rotações e escalas das várias figuras a desenhar. As translações, rotações e escalas a realizar deverão ser especificadas no ficheiro XML, seguindo um formato definido, assim como as figuras a desenhar.

A *demo* para a segunda fase é um modelo estático do sistema solar com o Sol, Planetas e Luas definidos em hierarquia.

2. Descrição do processo de Leitura

O ficheiro de configuração define toda a informação sobre os ficheiros a carregar e as suas transformações, como tal é composto por 7 elementos: *scene*, *group*, *translate*, *text*, *scale*, *models*, *model*. Destes as transformações são opcionais e o ficheiro, definido em *XML*, assenta sob a forma de uma árvore de *groups*.

Ou seja, cada nodo *group* é composto por um conjunto de ficheiros *models*, até 3 transformações diferentes, e vários nodos filho *group*. A vantagem desta alternativa é realizar transformações relativas a outro objeto ao invés de absolutas à origem. Por exemplo, para desenhar as luas de Júpiter, basta deslocar com base na distância a Júpiter, ao invés de ser com base na distância ao sol, o que facilitará depois nas translações.

Seguindo o exemplo anterior, como Júpiter realiza uma translação à volta do sol, no ficheiro de configuração Júpiter fará parte de um elemento que é filho do Sol, e o mesmo para a lua em relação a Júpiter.

As informações necessárias relativamente aos elementos são em todos os casos atributos como *file* para representar o nome de um ficheiro num elemento *model* ou como os eixos e ângulo de uma rotação, segue de seguida um exemplo.

3. Descrição das estruturas de dados para armazenar os Grupos

De acordo com a estrutura definida pelo ficheiro de configuração, foi necessário também criar uma estrutura para carregar toda essa informação. Assim utiliza-se a classe *Group* que é composta pela informação das transformações, um conjunto de ficheiros e grupos filho. Segue a definição das suas variáveis:

4. Descrição do ciclo de rendering

Rendering é converter uma série de símbolos gráficos num arquivo visual. Para renderizar uma cena é necessário, entre outras coisas, definir um tipo de textura para os objetos existentes, sua cor, transparência e reflexão, localizar um ou mais pontos de iluminação e um ponto de vista sob o qual os objetos serão visualizados. Ao renderizar, o programa calcula a perspectiva do plano, as sombras e a luz dos objetos.

5. Conclusão

Nesta fase procedeu-se a vários ajustes a nível da estrutura das classes, mas também otimizaçõess de código. Assumir isto como uma prioridade significou aliviar a implementaçãoo de novas funcionalidades futuras que podem ser mais exigentes a nível de processamento. Culmatou-se desta forma alguns dos problemas de desorganizaçãoo do código apresentados anteriormente.

Assim sendo introduziu-se VBOs no modo que os pontos são guardados, e separou-se este processo para uma classe distinta do engine. Resolveu-se também o problema de recarregamento dos ficheiros sempre que se fazia render. Foram realizadas algumas alterações a nivel da câmara, dando um maior controle ao utilizador, permitindo, para além do que já podia realizar na primeira fase, fazer zoom in e zoom out e melhorou-se a rotação da câmara.

Continua a não ser possivel diferenciar por cores as diferentes figuras, sendo estes um dos aspectos que queremos melhorar nas seguintes fases. Existem, também, ainda alguns problemas de otimização, tal como a leitura repetida de ficheiros iguais. Por exemplo, para dois planetas definidos pelo mesmo ficheiro, a estrutura carregada terá essa informaçã repetida, e o ficheiro será lido duas vezes. Em suma, achamos que, com esta fase 2, definiu-se as estruturas base a utilizar e simplificou-se a introdução de novas funcionalidades tendo-se melhorado as funcionalidades já existentes.