Computação Gráfica

Segunda Fase: Transformações Geométricas

Ciências da Computação

Bruna Alvarães	Igor Braga	Guilherme Santos	Ricardo Teixeira
(A84729)	(A85965)	(A85115)	(A85688)

4 de abril de 2021

Resumo

No seguimento das aulas deste semestre, o seguinte trabalho continua a elaboração de um motor 3D e fornece um modelo estático do Sistema Solar para mostrar as suas potencialidades.

Capítulo 1

Introdução

Nesta segunda fase do trabalho, alteramos o nosso ficheiro XML de maneira a acomodar Scenes. Estas são estruturadas numa árvore que contem as variadas transformações e os modelos aos quais aplicamos essas transformações.

1.1 Estrutura do Relatório

Neste relatório começamos por analisar o enunciado da segunda fase, e identificamos os requisitos necessários nesta fase. Depois falamos dos elementos dos XML das decisões de implementação e das alterações realizadas. Por fim mostramos os testes realizados e concluímos o relatório

Capítulo 2

Análise e Especificação

2.1 Descrição e Enunciado

2.1.1 Gerador

Como não estamos a gerar figuras novas nem a modificar a maneira como as geramos, nesta fase apenas o motor precisa de ser modificado. O gerador apresentado nesta fase é igual ao gerador da primeira entrega.

2.1.2 Motor

O motor precisa de ser bastante alterado nesta fase. A sua função continua a ser ler o ficheiro XML que serve de configuração, identificar as transformações, perceber quais as figuras a desenhar, e desenhar essas figuras no ecrã. Porém, agora a estruturação do ficheiro XML é bastante diferente e implica bastantes mudanças no seu funcionamento

2.2 Especificação do Requisitos

2.2.1 Motor

O motor tem de ser capaz de identificar no ficheiro XML os diferentes grupos dentro da Scene e aplicar as transformações corretas a cada grupo. Tem de ter em conta também a hierarquia de maneira a que as transformações também se apliquem aos grupos-filhos que esse grupo possa ter eventualmente.

Capítulo 3

Concepção/desenho da Resolução

3.1 Abordagem ao Problema

3.1.1 Construção da árvore

<scene> Reconhece a criação da cena
<\scene> Reconhece o termino da cena
<group> Reconhece a criação de um grupo
<\group> Reconhece o termino de um grupo
<models> Secção onde estão as figuras a desenhar
<model file="something.3d"\> Identifica o ficheiro com a figura
<\models> Reconhece o fim dessa secção

3.1.2 Transformações

<translate X=x Y=y Z=z \> Reconhece uma translação
<scale X=x Y=y Z=z \> Reconhece uma escala
<rotate angle = a axisX=x axisY=y axisZ=z \> Reconhece uma rotação

Capítulo 4

Codificação e Testes

4.1 Decisões de Implementação

Foi decidido que a melhor maneira de guardar a informação do XML para que esta possa ser processada é numa estrutura de árvore, e que os nodos iriam conter uma classe Grupo e as suas sub-classes, cada uma correspondente a uma informação do XML : Translação, Rotação, Scale e Modelo. Assim, com esta árvore, garantimos que a ordem das informações guardadas é mantida, visto que a ordem das transformações é essencial.

4.2 Código e Ferramentas Utilizadas

Recorrendo à linguagem C++, desenvolvemos um motor que usa *tinyXML2* e reconhece os ficheiros com a extensão .3D . Estes foram criados pelo gerador que também foi desenvolvido usando C++.

4.3 Alterações ao Motor

No nosso motor, tivemos que alterar as funções de leitura do XML para acomodar as novas informações, criamos as estruturas de árvores e as novas classes e por fim alteramos o modo de rendering do OpenGL para utilizar VBOs.

4.3.1 Leitura do Ficheiro XML para uma Árvore

A leitura é feita pela ordem do XML, sendo criado um nodo raíz para a criação da árvore em si, e por cada grupo lido é criado um nodo de grupo e é feita a leitura das instruções e possíveis sub-grupos. É guardado o tipo de instrução utilizando a sua sub-classe correspondente, uma string correspondente ao seu nome e um array com os seus descendentes.

4.3.2 Aplicação das transformações

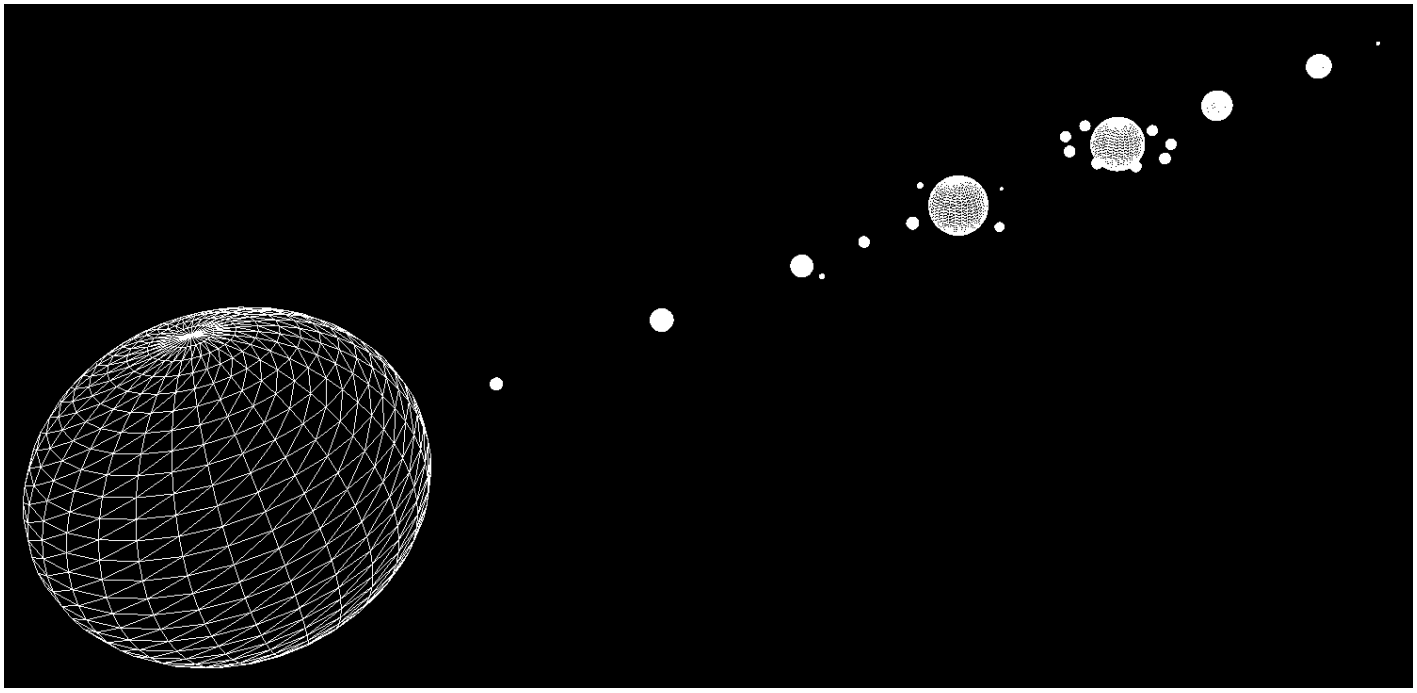
Aquando da leitura das instruções da árvore, é feita uma travessia depth-first, na qual é chamada a instrução correspondente de OpenGL : glPushMatrix no caso de se encontrar um grupo/subgrupo, glTranslatef no caso de uma translação, glScalef no caso de ser uma escala e glRotatef no caso de uma rotação. Caso a instrução seja a de leitura de um modelo, é lido o ficheiro correspondente e os seus vértices são guardados

num VBO. Sempre que a leitura de um grupo acaba, é executado um glPopMatrix para restaurar a matriz e poderem ser dadas novas transformações.

4.4 Testes realizados e Resultados

Estes foram os resultados obtidos:

4.4.1 Sistema Solar



Capítulo 5

Conclusão

A realização desta primeira fase ajudou-nos a por em prática os conhecimentos que aprendemos nas aulas, e também ajudou-nos a aprofundar o nosso conhecimento na linguagem C++.