# Computação Gráfica e
# Modelação Tri-Dimensional

# Implementação de Algoritmos para a Construção e Exibição de *Octrees*

*Documentação*

**Version 1.0**

**Autor:**Ricardo Filipe Alves Martins
(rmartins@isr.uc.pt)

**Departamento de Engenharia Electrotécnica e de Computadores**
**Faculdade de Ciências e Tecnologia da Universidade de Coimbra**

# Versões

| Versão | Data | Estado | Autor | Conteúdo |
|---|---|---|---|---|
| V1.0 | 06.03.2010 | | Ricardo Martins | Criacção do documento |

# Acrónimos

# Table of Contents

# 1 Motivação

Pretende-se através desta documentação descrever de forma sumária as principais funcionalidades do código implementado referente aos algoritmos para a construção de uma representação através de *Octrees* de conjunto de dados fornecido e exibição dessa representação.

# 2 *OctreeTree.cpp*

## *2.1 OctreeTree::OctreeTree(FILE * dataFileXYZ, int minnumPoints, int maxlevel)*

Recebe um ponteiro para um ficheiro de dados e o valor dos parâmetros correspondentes ao número mínimo de pontos de dados que um *octree* deverá ter para ser dividido em 8 novos *octrees* e o número máximo de gerações que se pretende que a árvore de *octrees* possua. Nesta implementação pressupõe-se que o ficheiro de dados se trate de um ficheiro **xyz* em que em cada linha se encontram registados os valores das coordenadas x,y,z de um ponto de dados.

O centróide dos pontos de dados fornecidos é estimado e as coordenadas dos pontos de dados são re-calculadas de forma a ter-se o centróide do conjunto de dados localizado na origem do referencial. O nodo inicial (parâmetro *rootNode*) da árvore é preenchido com todos os pontos do *dataset* e as dimensões do octree associado é estimada a partir da máxima amplitude dos pontos de dados em cada uma das dimensões.

## *2.2 void OctreeTree::ProcessDataset(void)*

Efectua o tratamento dos dados fornecidos no ficheiro **.xyz* de forma a terem-se os dados centrados na origem do referencial.

## *2.3 void OctreeTree::BuildOctree(Node *n)*

Recebe uma referência para um nodo da árvore de Octrees. Se o nodo tiver associado um número de pontos superior ao número mínimo de pontos definidos e a geração a que pertence é inferior ao igual ao limite máximo de gerações definido, procede-se à divisão dos pontos do nodo por 8 novos nodos (nodos flho). Este procedimento é efectuado recursivamente até um dos critérios definidos anteriormente não se verificar.

### *2.4 Node *OctreeTree::BuildNode(Node *nParent, int index)*

Recebe uma referência de um nodo Pai e efectua a distribuição dos pontos do nodo Pai por um dos nodo Filho (tipo/localização de nodo Filho indicado pelo parâmetro *índex*).

## 3 *Node*.cpp

Classe utilizada para representar os nodos da árvore de Octrees. Cada objecto desta classe tem como parâmetros a posição no espaço do Octree correspondente a este nodo (posX, posY, posZ), assim como as suas dimensões (dX,dY,dZ), a geração a que pertence o nodo, a lista de 8 referências para cada um dos nodo Filho e a lista de pontos de dados associados a este nodo.

## 4 *Display.cpp*

Descrevem-se de seguida algumas das principais funções implementadas neste ficheiro.

### *4.1 void displayOctreeTree(Node * n,int grid)*

Função que recebe uma referência para um nodo de geração 0 e desencadeia os processos envolvidos na visualização de toda a representação da árvore de Octrees.

### *4.2 void displayOctree(Node * n,int grid)*

Função que explora recursivamente o nodo da geração 0 fornecido. Caso o nodo tenha nodos Filho, explora esses nodosFilho. Caso se chegue a um nodo terminal da árvore de Octrees, se este tiver um número de pontos associados superior a 0, trata-se de uma célula do espaço que deve ser representada como célula preenchida, caso não tenha qualquer ponto associado, trata-se de uma célula que deverá ser representada como uma

célula vazia. É dada a possibilidade, através do parâmetro *grid*, escolher se na representação inicial, as células vazias não são representadas ou se são representadas pelas arestas dessas células. Esta opção pode ser alterada dinamicamente ao longo da visualização, pressionando as teclas do teclado apropriadas.

## *4.3 void drawSkelethonCube(float posX, float posY,float posZ,float dX, float dY,float dZ)*

Representação de uma célula do espaço através das arestas (de cor vermelha) que delimitam essa região.

## *4.4 void drawFilledCube(float posX, float posY, float posZ, float dX, float dY, float dZ)*

Representação de uma célula do espaço através de um sólido com faces (de cor preta) preenchidas.

## *4.5 void eventKeyboard(unsigned char key, int x, int y)*

Usando o teclado é possível explorar a representação através de Octrees dos dados fornecidos.

| **Tecla** | **Função** |
|---|---|
| 8 | Aumentar o ângulo de rotação em torno do eixo X |
| 2 | Diminuir o ângulo de rotação em torno do eixo X |
| 6 | Aumentar o ângulo de rotação em torno do eixo Y |
| 4 | Diminuir o ângulo de rotação em torno do eixo Y |
| 7 | Aumentar ângulo de rotação em torno do eixo Z |
| 9 | Diminuir ângulo de rotação em torno do eixo X |
| + | Aproximar a representação |
| - | Afastar a representação |
| espaço | Voltar às condições de visualização iniciais |

| | |
|---|---|
| . | Não representar células vazias |
| , | Representar células vazias. |