Compte rendu du TP n° 2 d'imagerie 3D

Imagerie médicale 3D

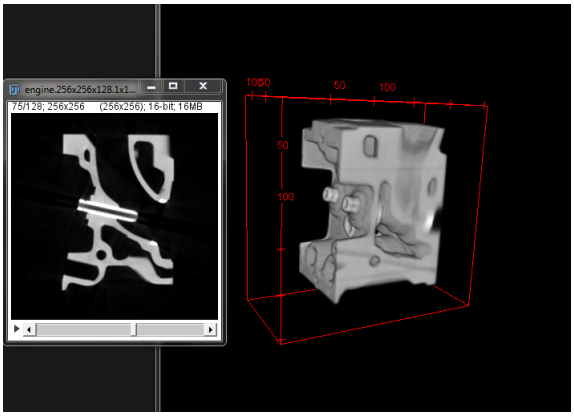
Thibaut Castanié
Master IMAGINA

26 mars 2015

1 Explications

Le principe du TP est de réutiliser les fonctions réalisées précédemment afin de transformer les images binaires en un objet 3D composé d'un maillage. Pour cela, on utilisera une valeur de seuil qui permettra de ne conserver que les voxels dont la valeur est supérieure au seuil. Ensuite, si la condition précédente est vérifiée, on vérifie si les voxels voisins ont aussi une valeur supérieure au seuil. Quand c'est le cas, on dessine deux triangles surfaciques entre ces deux voxels (pour former un quadrangle avec la combinaison des deux triangles).

Afin de coder le programme permettant de réaliser cette conversion, j'ai utilisé l'objet ENGINE, avec une valeur de seuil de 200.



L'objet ENGINE sous fiji

Cependant, je n'arrive pas à obtenir un fichier stl fonctionnel. En effet, même si celui ci pèse 226 Mo après exécution de mon algorithme, rien ne s'affiche lorsque je l'ouvre dans un logiciel de création 3D ou dans MeshLab. Il doit y avoir une erreur dans mon code que je n'arrive pas à déceler.

Annexe - Code

```cpp
#include <iostream>
#include <stdio.h>
#include <cstdlib>
#include <math.h>
#include <string>
using namespace std;

int tailleX = 256;
int tailleY = 256;
int tailleZ = 128;
float sizeX=1;
float sizeY=1;
float sizeZ=1;
unsigned short * buffer;

int getValue(unsigned short * buffer,int x, int y, int z){
    int num = z*(tailleX*tailleY)+(y*tailleX)+x;
    int oct1 = buffer[num]%256;
    int oct2 = buffer[num]/256;
    int valPixel = oct1*256+oct2;
    return valPixel;
}

int main(int argc, char const *argv[]){
    long taille;
    size_t result;
    FILE * imgFile;
    FILE * stockFile;
    short valeurPixel = 0;
    short octet1;
    short octet2;
    int seuil = 200;
    imgFile = fopen ("./engine/engine.256x256x128.1x1x1.img","rb");
    if (imgFile!=NULL){
        fseek (imgFile , 0 , SEEK_END);
        taille = ftell (imgFile);
        rewind (imgFile);
        unsigned short *buffer = new unsigned short[taille];
        result = fread (buffer,2,taille,imgFile);

        stockFile = fopen ("engine.stl","wb");

        int (*render)[8][3] = new int[(tailleX-2)*(tailleY-2)*(tailleZ-2)][8][3];
        for (int k = 1; k < tailleZ-1; k++){
            for (int j = 1; j < tailleY-1; j++){
                for (int i = 1; i < tailleX-1; i++){
                    render[i*j*k][0][0]=(i-0,5)*sizeX;
                    render[i*j*k][0][1]=(j-0,5)*sizeY;
                    render[i*j*k][0][2]=(k-0,5)*sizeZ;

                    render[i*j*k][1][0]=(i+0,5)*sizeX;
                    render[i*j*k][1][1]=(j-0,5)*sizeY;
                    render[i*j*k][1][2]=(k-0,5)*sizeZ;

                    render[i*j*k][2][0]=(i+0,5)*sizeX;
                    render[i*j*k][2][1]=(j-0,5)*sizeY;
                    render[i*j*k][2][2]=(k+0,5)*sizeZ;

                    render[i*j*k][3][0]=(i-0,5)*sizeX;
                    render[i*j*k][3][1]=(j-0,5)*sizeY;
                    render[i*j*k][3][2]=(k+0,5)*sizeZ;

                    render[i*j*k][4][0]=(i-0,5)*sizeX;
                    render[i*j*k][4][1]=(j+0,5)*sizeY;
                    render[i*j*k][4][2]=(k-0,5)*sizeZ;

                    render[i*j*k][5][0]=(i+0,5)*sizeX;
```

```
                render[i*j*k][5][1]=(j+0,5)*sizeY;
                render[i*j*k][5][2]=(k-0,5)*sizeZ;

                render[i*j*k][6][0]=(i+0,5)*sizeX;
                render[i*j*k][6][1]=(j+0,5)*sizeY;
                render[i*j*k][6][2]=(k+0,5)*sizeZ;

                render[i*j*k][7][0]=(i-0,5)*sizeX;
                render[i*j*k][7][1]=(j+0,5)*sizeY;
                render[i*j*k][7][2]=(k+0,5)*sizeZ;
            }
        }
    }

    //debut ecriture stl
    char ligne[250];
    sprintf(ligne, "%s \n", "solid engine");
    cout<<ligne;
    fwrite(ligne, sizeof(ligne),1,stockFile);

    for (int k = 1; k < tailleZ-1; k++){
        for (int j = 1; j < tailleY-1; j++){
            for (int i = 1; i < tailleX-1; i++){
                int val = getValue(buffer,i,j,k);
                if(val > seuil){
                    if(getValue(buffer,i+1,j,k) > seuil){
                        sprintf(ligne, "%s \n", "facet normal");
                        fwrite(ligne, sizeof(ligne),1,stockFile);
                        sprintf(ligne, "%s \n", "outer loop");
                        fwrite(ligne, sizeof(ligne),1,stockFile);
                        sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][1][0],
                        render[i*j*k][1][1],render[i*j*k][1][2]);
                        fwrite(ligne, sizeof(ligne),1,stockFile);
                        sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][5][0],
                        render[i*j*k][5][1],render[i*j*k][5][2]);
                        fwrite(ligne, sizeof(ligne),1,stockFile);
                        sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][2][0],
                        render[i*j*k][2][1],render[i*j*k][2][2]);
                        fwrite(ligne, sizeof(ligne),1,stockFile);
                        sprintf(ligne, "%s \n", "endloop");
                        fwrite(ligne, sizeof(ligne),1,stockFile);
                        sprintf(ligne, "%s \n", "endfacet");
                        fwrite(ligne, sizeof(ligne),1,stockFile);

                        sprintf(ligne, "%s \n", "facet normal");
                        fwrite(ligne, sizeof(ligne),1,stockFile);
                        sprintf(ligne, "%s \n", "outer loop");
                        fwrite(ligne, sizeof(ligne),1,stockFile);
                        sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][5][0],
                        render[i*j*k][5][1],render[i*j*k][5][2]);
                        fwrite(ligne, sizeof(ligne),1,stockFile);
                        sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][6][0],
                        render[i*j*k][6][1],render[i*j*k][6][2]);
                        fwrite(ligne, sizeof(ligne),1,stockFile);
                        sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][2][0],
                        render[i*j*k][2][1],render[i*j*k][2][2]);
                        fwrite(ligne, sizeof(ligne),1,stockFile);
                        sprintf(ligne, "%s \n", "endloop");
                        fwrite(ligne, sizeof(ligne),1,stockFile);
                        sprintf(ligne, "%s \n", "endfacet");
                        fwrite(ligne, sizeof(ligne),1,stockFile);
                    }
                    else if(getValue(buffer,i-1,j,k) > seuil){
                        sprintf(ligne, "%s \n", "facet normal");
                        fwrite(ligne, sizeof(ligne),1,stockFile);
                        sprintf(ligne, "%s \n", "outer loop");
                        fwrite(ligne, sizeof(ligne),1,stockFile);
```

```
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][0][0],
                render[i*j*k][0][1],render[i*j*k][0][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][4][0],
                render[i*j*k][4][1],render[i*j*k][4][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][3][0],
                render[i*j*k][3][1],render[i*j*k][3][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "endloop");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "endfacet");
                fwrite(ligne, sizeof(ligne),1,stockFile);

                sprintf(ligne, "%s \n", "facet normal");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "outer loop");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][4][0],
                render[i*j*k][4][1],render[i*j*k][4][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][7][0],
                render[i*j*k][7][1],render[i*j*k][7][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][3][0],
                render[i*j*k][3][1],render[i*j*k][3][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "endloop");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "endfacet");
                fwrite(ligne, sizeof(ligne),1,stockFile);
            }
            else if(getValue(buffer,i,j-1,k) > seuil){
                sprintf(ligne, "%s \n", "facet normal");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "outer loop");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][4][0],
                render[i*j*k][4][1],render[i*j*k][4][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][5][0],
                render[i*j*k][5][1],render[i*j*k][5][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][7][0],
                render[i*j*k][7][1],render[i*j*k][7][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "endloop");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "endfacet");
                fwrite(ligne, sizeof(ligne),1,stockFile);

                sprintf(ligne, "%s \n", "facet normal");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "outer loop");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][6][0],
                render[i*j*k][6][1],render[i*j*k][6][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][5][0],
                render[i*j*k][5][1],render[i*j*k][5][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][7][0],
                render[i*j*k][7][1],render[i*j*k][7][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "endloop");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "endfacet");
```

```
                fwrite(ligne, sizeof(ligne),1,stockFile);
            }
            else if(getValue(buffer,i,j+1,k) > seuil){
                sprintf(ligne, "%s \n", "facet normal");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "outer loop");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][0][0],
                render[i*j*k][0][1],render[i*j*k][0][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][1][0],
                render[i*j*k][1][1],render[i*j*k][1][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][3][0],
                render[i*j*k][3][1],render[i*j*k][3][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "endloop");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "endfacet");
                fwrite(ligne, sizeof(ligne),1,stockFile);

                sprintf(ligne, "%s \n", "facet normal");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "outer loop");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][2][0],
                render[i*j*k][2][1],render[i*j*k][2][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][3][0],
                render[i*j*k][3][1],render[i*j*k][3][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][1][0],
                render[i*j*k][1][1],render[i*j*k][1][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "endloop");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "endfacet");
                fwrite(ligne, sizeof(ligne),1,stockFile);
            }
            else if(getValue(buffer,i,j,k-1) > seuil){
                sprintf(ligne, "%s \n", "facet normal");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "outer loop");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][3][0],
                render[i*j*k][3][1],render[i*j*k][3][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][2][0],
                render[i*j*k][2][1],render[i*j*k][2][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][7][0],
                render[i*j*k][7][1],render[i*j*k][7][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "endloop");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "endfacet");
                fwrite(ligne, sizeof(ligne),1,stockFile);

                sprintf(ligne, "%s \n", "facet normal");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s \n", "outer loop");
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][6][0],
                render[i*j*k][6][1],render[i*j*k][6][2]);
                fwrite(ligne, sizeof(ligne),1,stockFile);
                sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][7][0],
                render[i*j*k][7][1],render[i*j*k][7][2]);
```

```
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                            sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][2][0],
                            render[i*j*k][2][1],render[i*j*k][2][2]);
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                            sprintf(ligne, "%s \n", "endloop");
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                            sprintf(ligne, "%s \n", "endfacet");
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                        }
                        else if(getValue(buffer,i,j,k+1) > seuil){
                            sprintf(ligne, "%s \n", "facet normal");
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                            sprintf(ligne, "%s \n", "outer loop");
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                            sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][0][0],
                            render[i*j*k][0][1],render[i*j*k][0][2]);
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                            sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][1][0],
                            render[i*j*k][1][1],render[i*j*k][1][2]);
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                            sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][4][0],
                            render[i*j*k][4][1],render[i*j*k][4][2]);
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                            sprintf(ligne, "%s \n", "endloop");
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                            sprintf(ligne, "%s \n", "endfacet");
                            fwrite(ligne, sizeof(ligne),1,stockFile);

                            sprintf(ligne, "%s \n", "facet normal");
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                            sprintf(ligne, "%s \n", "outer loop");
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                            sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][5][0],
                            render[i*j*k][5][1],render[i*j*k][5][2]);
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                            sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][4][0],
                            render[i*j*k][4][1],render[i*j*k][4][2]);
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                            sprintf(ligne, "%s %d %d %d \n", "vertex ",render[i*j*k][1][0],
                            render[i*j*k][1][1],render[i*j*k][1][2]);
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                            sprintf(ligne, "%s \n", "endloop");
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                            sprintf(ligne, "%s \n", "endfacet");
                            fwrite(ligne, sizeof(ligne),1,stockFile);
                        }
                    }
                }
            }
        }

        sprintf(ligne, "%s \n", "endsolid engine");
        cout<<ligne;
        fwrite(ligne, sizeof(ligne),1,stockFile);

        fclose(stockFile);
        fclose(imgFile);
        free (buffer);
    }
    else{
        cout << "erreur ouverture" << endl;
    }
    system("pause");
    return 0;
}
```