TP Imagerie 3D n° 2 (3 heures)

Gérard Subsol, 13 mars 2014

- Les programmes doivent être écrits en **C/C++** en n'utilisant que des **fonctions classiques** (stdio, stdlib, math).
- **Le TP est noté**. Il faut rendre un **compte-rendu avant** le prochain cours prévu **le vendredi 27 mars 2015 sous forme électronique à : gerard.subsol@lirmm.fr**
- Le compte-rendu doit inclure quelques lignes d'explication sur l'algorithme et le code, des captures d'écran (ici, le maillage stl surperposé sur une vue MPR) et le code source intégral. Le tout doit être sous la forme d'un **unique fichier pdf**.
- Le TP est individuel.
- La participation active pendant le TP pourra aussi être prise en compte.
- **Tout plagiat sera lourdement sanctionné**.

1. **Ecrire en C/C++ un programme :**

   a. Qui lit les images img qui sont au format brut (sans en-tête) et codées en unsigned short (2 octets).
   b. Qui les stocke en mémoire
   c. Qui intègre une fonction getValue(i,j,k) qui renvoie la valeur du voxel (i,j,k)

   Pour les 3 points ci-dessus, on reprendra les fonctions du TP 1

   d. Qui seuille l'image
   e. Qui transforme l'image binaire en un maillage surfacique 3D codé au format STL (cf. ci-après).

      Pour cela, on programmera une version simplifiée de l'algorithme du Marching Cubes vu en cours.

      - Parcours de tous les voxels de l'image (en ne prenant pas en compte les voxels du bord).
      - Pour chaque voxel, calcul des coordonnées réelles (c'est-à-dire en intégrant la taille du voxel) des 8 sommets de la façon suivante :

```
/* (i,j,k) is the center of the voxel */
/* Real coordinates of the center: (i*sizeX, j*sizeY, k*sizeZ)
/* The X vertices abscissae of the voxel corners are then
        (i-0.5)*sizeX and (i+0.5)*sizeX */
/*      z^        7 --- 6                */
/*       I       /I    /I                */
/*       I    4 --- 5  I                 */
/*        I y I  3 -I- 2                 */
/*       I /  I /   I /                  */
/*       I/   0 --- 1                    */
/*       +-----------------> x           */
/* 8 vertices */
```

- Si la valeur du voxel (i,j,k) est supérieure au seuil :
  - Alors si la valeur d'un des 6 voxels adjacents est inférieure au seuil, rajouter au fichier résultat les 2 triangles qui forment la face entre les deux. Cette face sera une approximation grossière de l'isosurface. Les triangles seront créés au format STL.
  - Les points de la face doivent toujours être dans un ordre tel que quand on calcule la normale au triangle par l'orientation canonique (on visse), celle-ci doit être « sortante » (c'est-à-dire aller du centre du voxel vers l'extérieur). Par exemple, la face éventuellement créée entre le voxel et le voxel du dessus devra être formée des triangles [456] et [674]

On pourra utiliser les paramètres suivants :

<imageFile> <dimX (int)> <dimY (int)> <dimZ (int)> <sizeXVoxel (float)> <sizeYVoxel (float)> <sizeZVoxel (float)> <resultFile (string)> <threshold (unsigned short)>

2. Tester sur les images 3D, en particulier :

engine : seuil=200 (petit fichier)
seuil=100 (gros fichier)

whatisit : seuil=50 (moyen fichier)

MANIX : seuil=1250 (gros fichier)

BEAUFIX : seuil=120 (très gros fichier)

On pourra visualiser l'ensemble image 3D + surface avec Fiji ou MicroView et la surface avec MeshLab.

A. MicroView *http://microview.sourceforge.net/*

- Pour lire une image 3D : File/Open/Interfile format (*.hdr) lit directement les fichiers hdr (en-tête) + img (données brutes)
- Pour lire un maillage surfacique au format stl : Visualize/Overlay Geometry/Load (stl). On peut décocher « Display Wireframe » pour ne pas avoir la visualisation fil de fer des arêtes du maillage.

B. Fiji *http://fiji.sc/*

- Plugins/3D Viewer permet de visualiser l'image en Volume (rendering) ou Orthoslice (mettre 1 à Resampling factor pour ne pas sous-échantillonner l'image). Attention, il faut d'abord convertir l'image en 8 bits par Image/Type/8-bit.
- On peut alors lire un maillage surfacique au foramt stl par : File/Import Surfaces/STL

C. MeshLab *http://meshlab.sourceforge.net/* est un programme très puissant de traitement et de modélisation de maillages 3D.

D. Stockage de l'image 3D

On supposera que l'image est lue coupe par coupe et qu'elle est balayée horizontalement.

E. Fonction C de lecture de données binaires

FILE *fopen(const char *_filename_, const char *_mode_);
Opens the filename pointed to by filename. The mode argument may be one of the following constant strings:

rb read binary mode

wb write binary mode (truncates file to zero length or creates new file)

size_t fread(void *_ptr_, size_t _size_, size_t _nmemb_, FILE *_stream_);
Reads data from the given stream into the array pointed to by *ptr*. It reads *nmemb* number of elements of size *size*. The total number of bytes read is (**size*nmemb**).
On success the number of elements read is returned. On error or end-of-file the total number of elements successfully read (which may be zero) is returned.

F. Format STL

STL (Standard Tessellation Language) is a file format (binary or ASCII) native to the stereolithography CAD software created by 3D Systems. This file format is supported by many other software packages; it is widely used for rapid prototyping and computer-aided manufacturing.

An ASCII STL file begins with the line:

solid name

where name is an optional string.

The file continues with any number of triangles, each represented as follows:

facet normal 0 0 0 (the normal coordinates are optional)
outer loop
vertex v1x v1y v1z
vertex v2x v2y v2z
vertex v3x v3y v3z
endloop
endfacet

where each n or v is a floating point number.
The file concludes with:

endsolid name

(Source : *http://en.wikipedia.org/wiki/STL_%28file_format%29*)

G. Données disponibles à : *http://www.lirmm.fr/~subsol/GMIN215/*