Imagerie 3D (2)

Gérard Subsol

gerard.subsol@lirmm.fr

19 avril 2019

1. Quelques rappels

2. Quelles sont les différences entre le traitement 2D et 3D ?

3. Visualisation surfacique

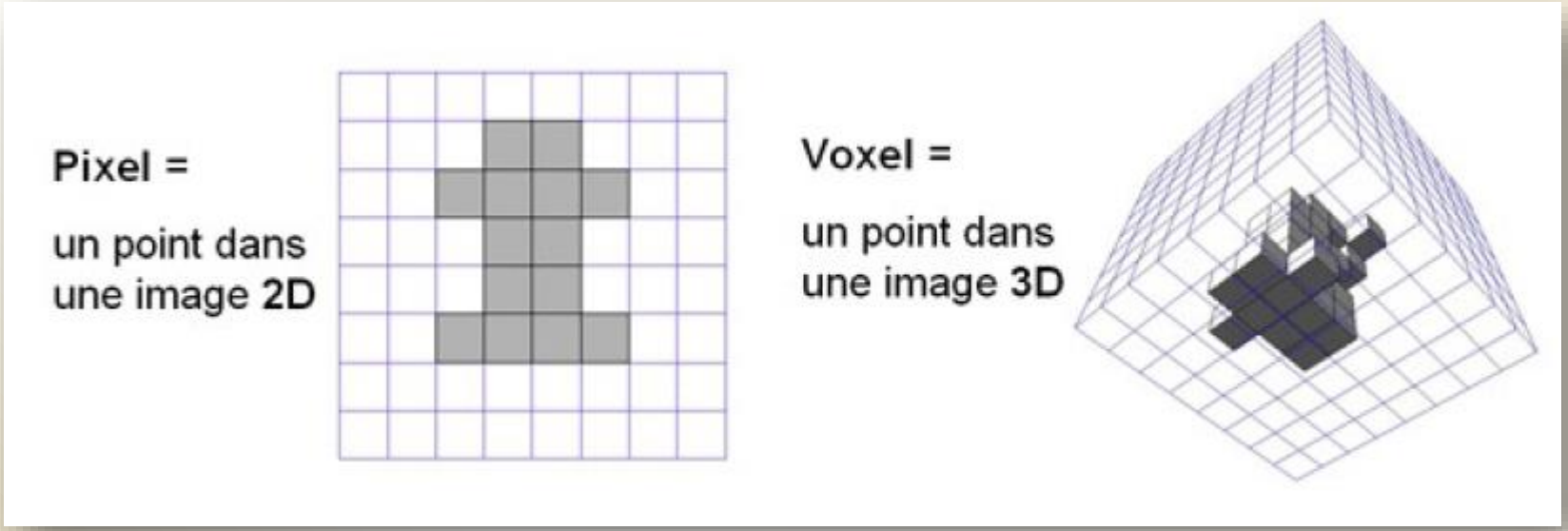
- a. Segmentation d'une Région d'Intérêt
- b. Quelques rappels sur les maillages
- c. Extraction d'isosurface par l'algorithme du « Marching Cubes »

4. Travaux pratiques

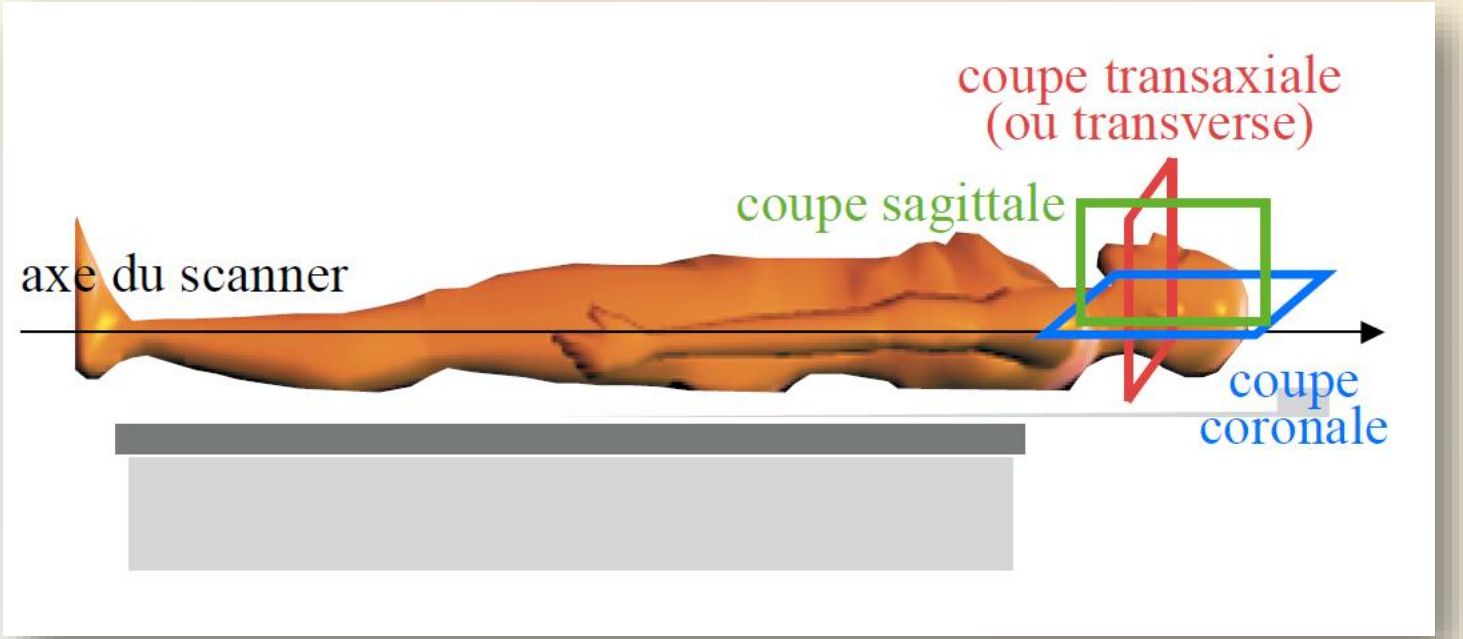
Extraction d'isosurface

Rappel 1 : qu'est-ce qu'une image 3D ?

- Matrice en **3** dimensions et non plus en 2 dimensions.
- Représentées par un empilement d'images 2D (**coupes**)
- Notion d'**épaisseur** en plus de la largeur et de la longueur
- Pixel (*Picture Element*) → **Voxel** (*Volume Element*)
- I=f(x,y,z) où I=intensité (ou une couleur)



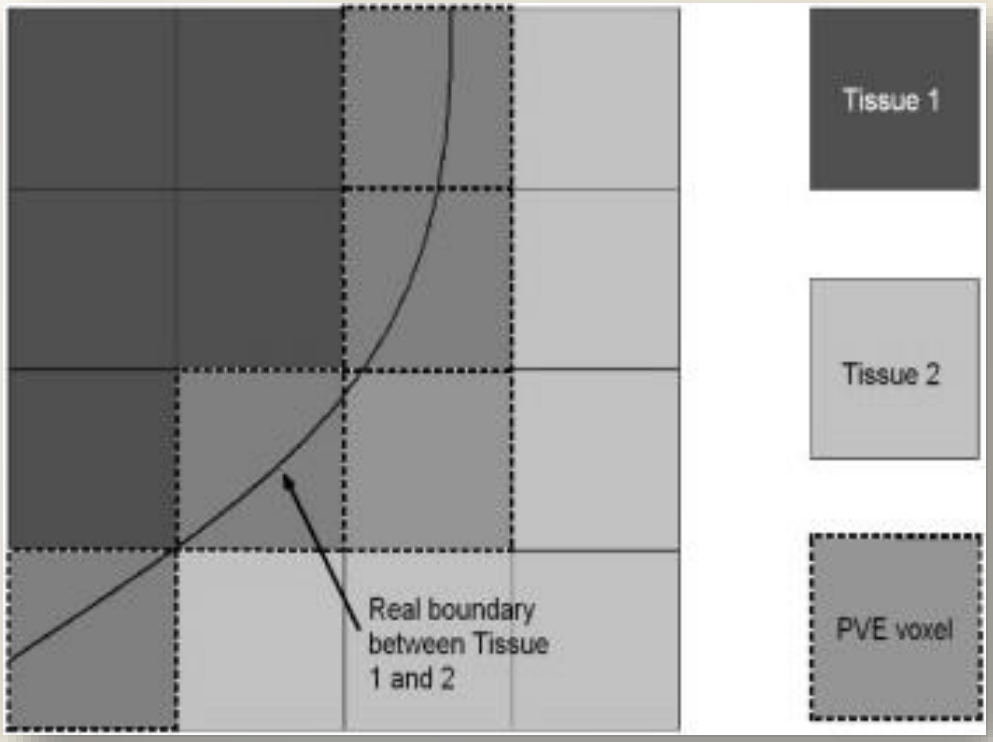
Rappel 2 : Comment sont représentées et stockées ces images 3D ?

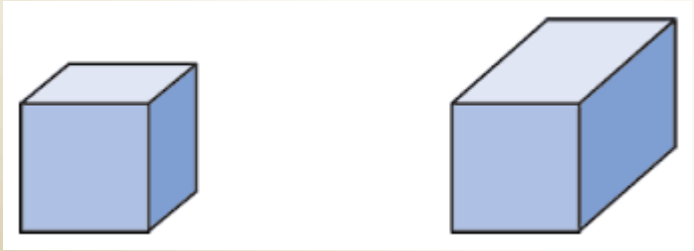


- Tout simplement comme un empilement de coupes suivant la direction d'acquisition ou de reconstruction…
- Intensité codée sur 12 bits ou 8 bits, souvent sur 2 octets : attention au codage big endian/little endian : 1000 = 256 x **3** + **232**
  → faut-il stocker : 03 E8 ou E8 03 ?
- Ne pas oublier de conserver la taille du pixel et son épaisseur (SliceThickness) ≠ écart entre coupes (SpacingBetweenSlices)
- Standard DICOM

Rappel 3 : Le « volume partiel »

• Un grand problème : le « volume partiel » (partial volume)



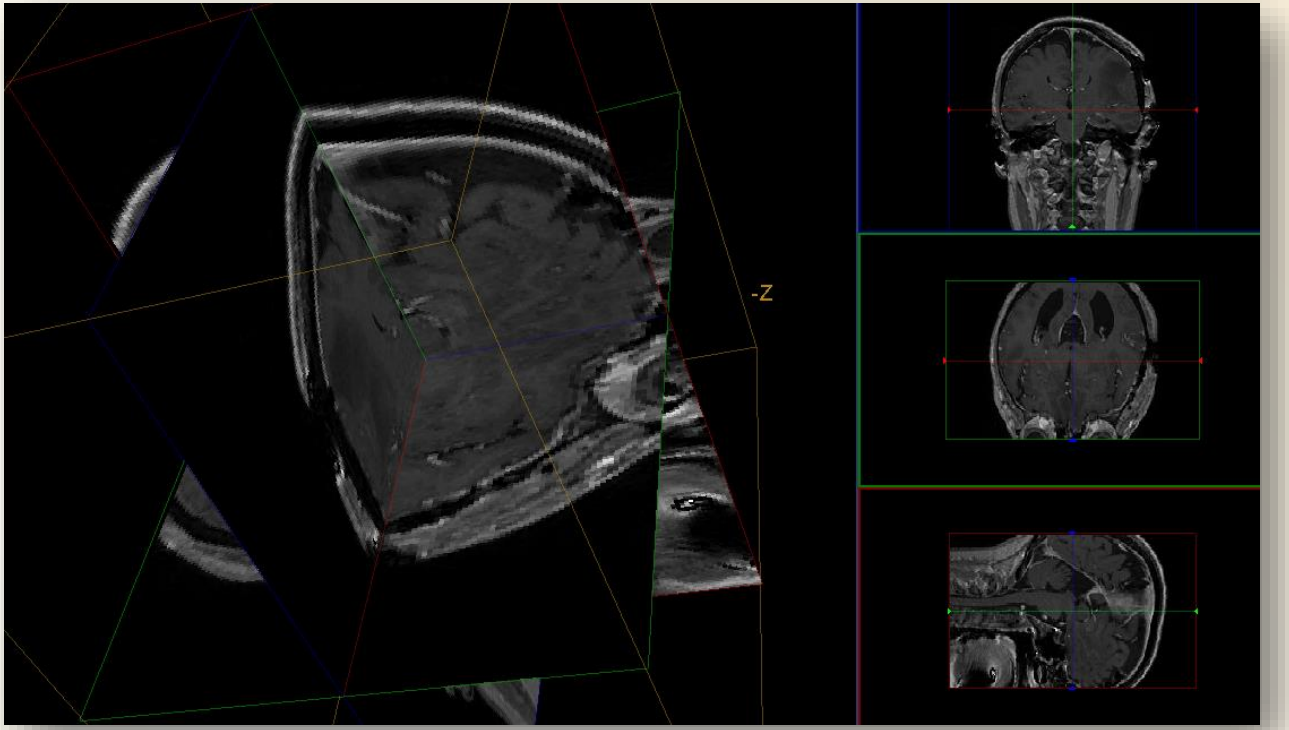


Renforcé par la résolution souvent plus limitée en z

Rappel 4 : Comment visualiser une image 3D ?

Problème : comment voir à l'intérieur de l'image ?

En visualisant suivant 3 plans orthogonaux (*Multi-Planar Reconstruction*) avec éventuellement une vision « 3D » de ces plans



→ Ne permet pas de visualiser des structures obliques…

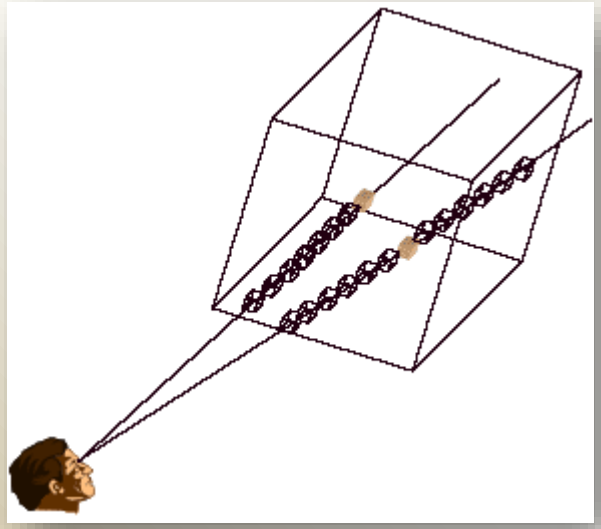
Rappel 4 : Comment visualiser une image 3D ?

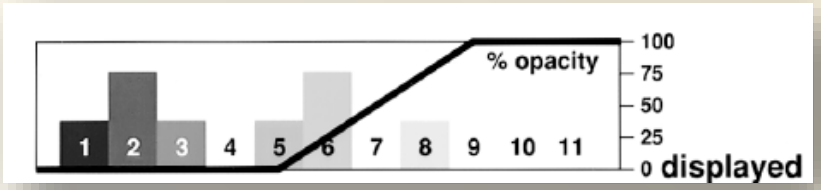
Problème : comment voir à l'intérieur de l'image ?

4. Visualisation volumique (*Volume Rendering*)

Principe :

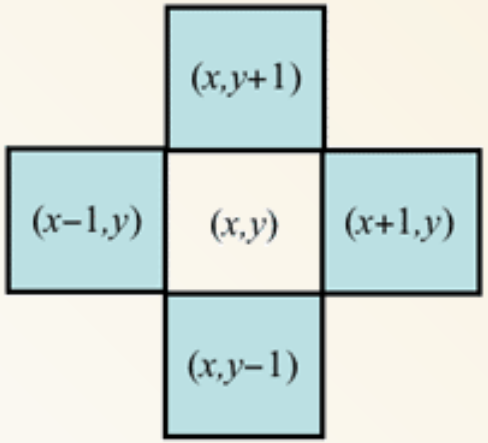
- définir une couleur et une opacité pour chaque voxel en fonction de son intensité ;
- sélectionner un point de vue d'observation de l'image 3D
- « intégrer » les informations de couleur et d'opacité en fonction des voxels traversés par les rayons issus de l'œil (*ray casting*).



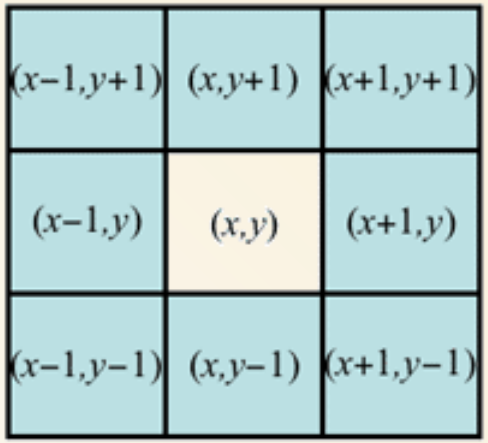


2D/3D : Un voxel… parmi beaucoup plus de voisins

- Voisinage (**neighborhood**) défini par la connexité (**connectivity**).
- Notion de taille (**size**) de voisinage : voisin du voisin du voisin….

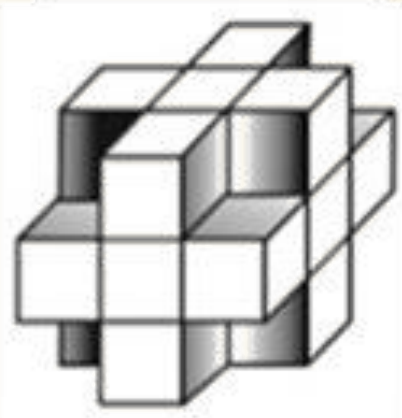


4-neighbourhood



8-neighbourhood

Face (6)



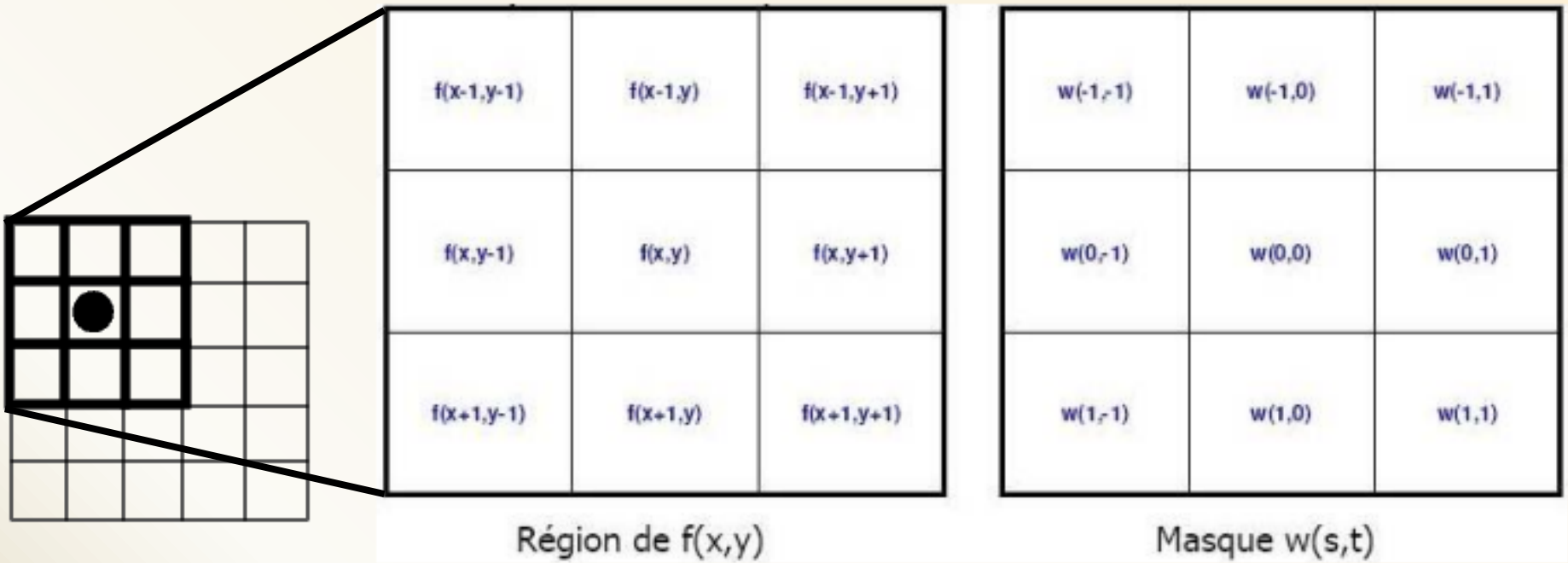
Edge (18)

Corner (26)

2D/3D : des méthodes généralisables (filtrage linéaire)

Beaucoup de filtres de traitement d'images (2D ou) 3D sont linéaires.

→ définis par une convolution (**convolution**) avec une fonction de pondération (masque / **mask ou noyau / kernel**) définie sur un voisinage 3D.



$$g(x,y) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} w(i,j) f(x+i, y+j)$$

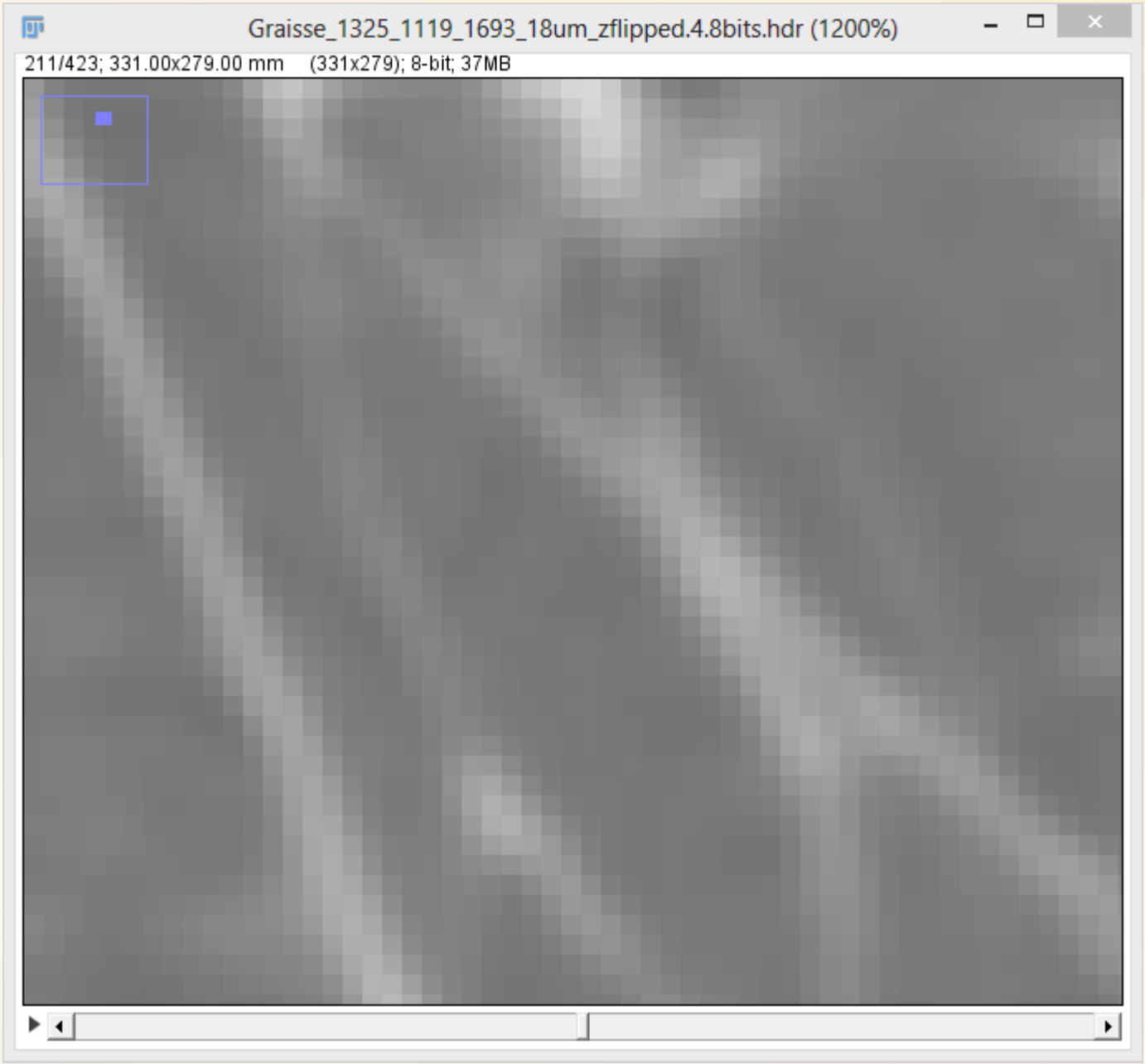
Σ w(i,j) = 1;

Exemples de filtre linéaire

Moyenne locale par exemple pour lisser (**smoothing**) ou flouter (**blurring**)

M(x,y,z)=1/27

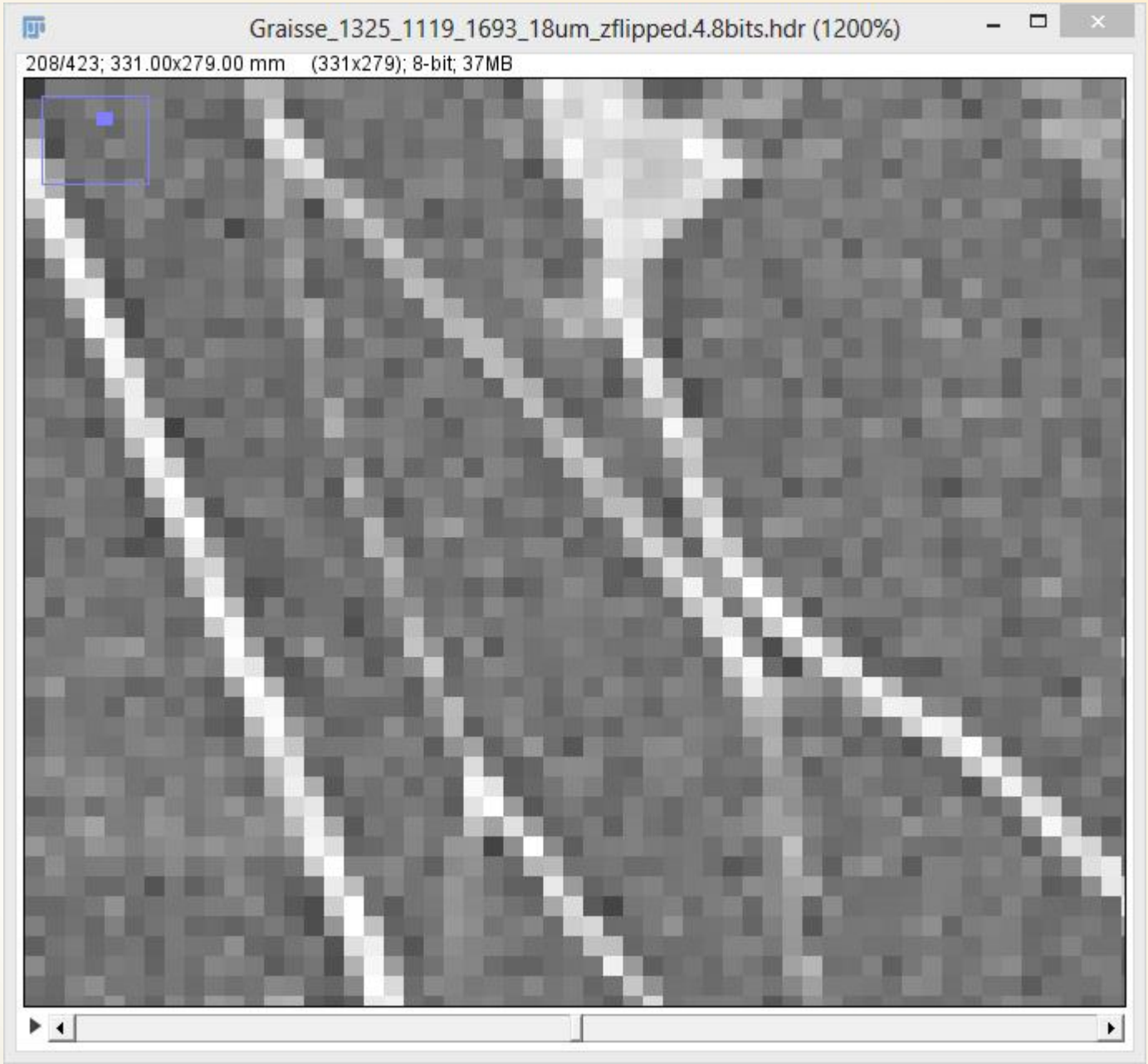
1	1	1
1	1	1
1	1	1



Différences pour rehausser le contraste (**contrast enhancement**)

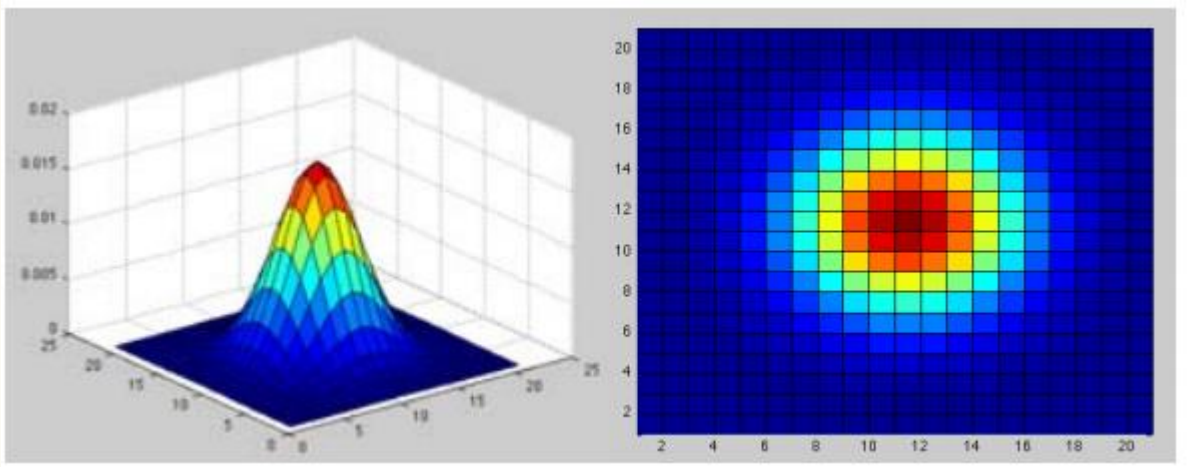
M(x,y,z)=

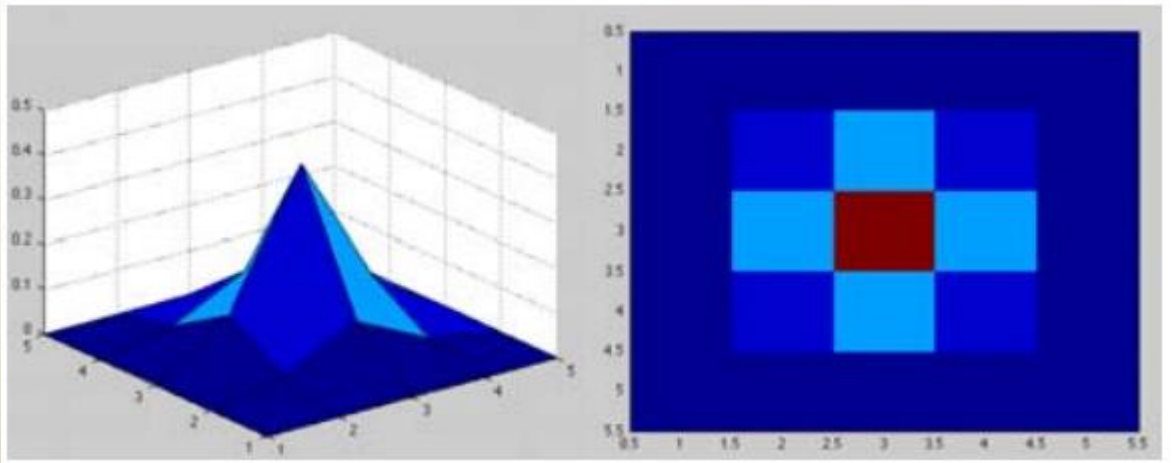
0	-1	0
-1	7	-1
0	-1	0



Exemples de filtre linéaire

Lissage Gaussien





Exemple : σ = 1,4

$$h = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

Autres filtres

Il existe des filtres non linéaires comme le filtre médian (**median filter**), particulièrement efficace contre le bruit “poivre et sel” ou bruit impulsionnel.

Par exemple, si on considère ces neufs pixels, dont une valeur aberrante (ici 111) :

5	6	7
6	111	8
7	8	9

le filtre médian va considérer les valeurs du voisinage par valeurs croissantes :

5	6	6	7	**7**	8	8	9	111

et prendre la valeur médiane, ici la valeur 7. La sortie du filtre donnera donc :

5	6	7
6	**7**	8
7	8	9

https://fr.wikipedia.org/wiki/Filtre_m%C3%A9dian

La notion de gradient 3D

Gradient 3D = vecteur des dérivées du signal de l'image (= intensité) selon les 3 directions x y z.

$$\nabla f = \text{grad } f = < \frac{\partial f}{\partial x}(x,y,z), \frac{\partial f}{\partial y}(x,y,z), \frac{\partial f}{\partial z}(x,y,z) >$$

- Le vecteur gradient est dirigé dans la direction de la plus forte variation locale de l'intensité.
- Pointe vers les zones de plus forte intensité.
- Sa norme donne une indication sur l'importance de la variation.
  → Les contours sont caractérisés par des valeurs élevées de la norme du gradient.

Détection de contours dans une image 3D

Les coordonnées du gradient 3D peuvent être calculées à l'aide de filtres linéaires par différences finies. Exemple : filtres de **Sobel** en 3D

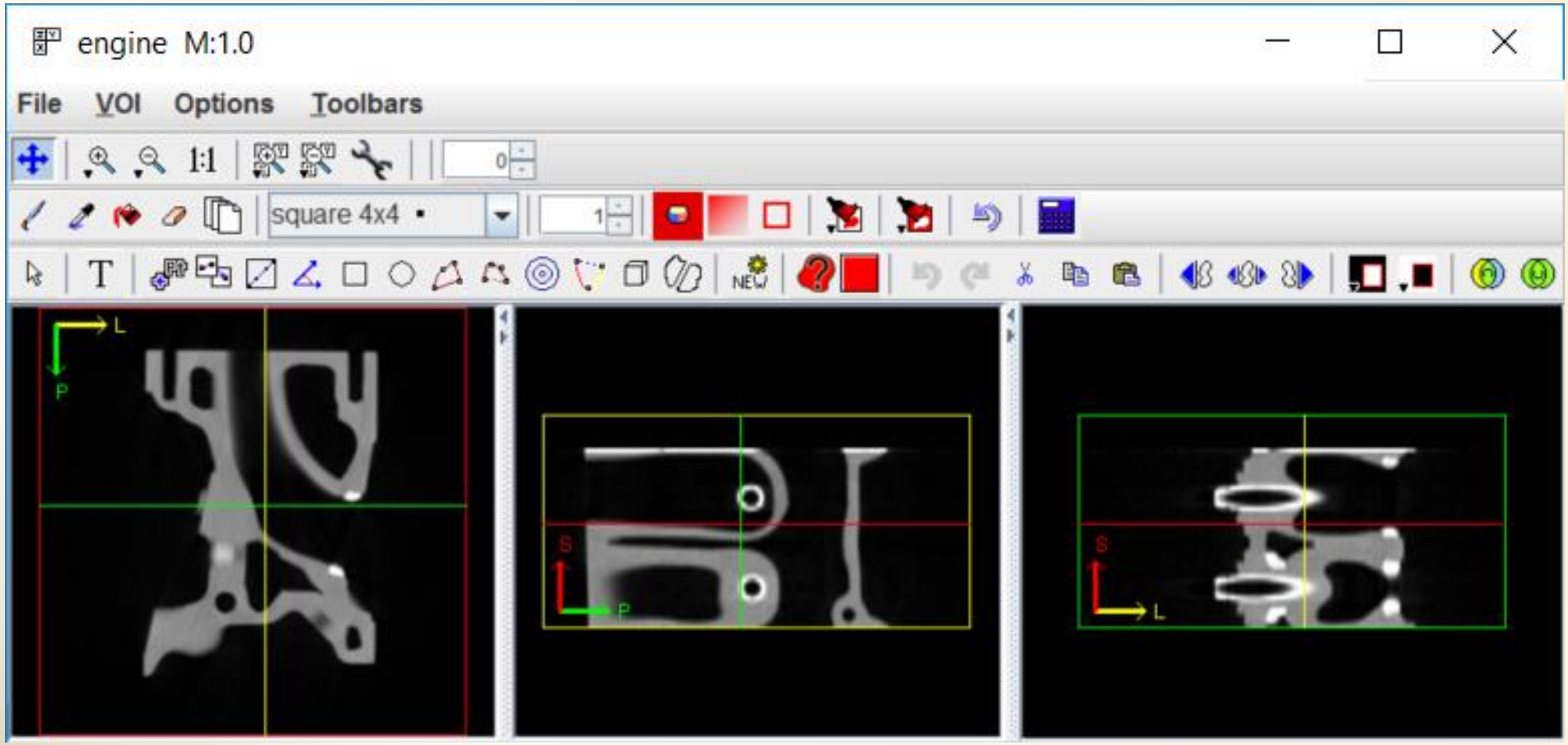
-1	-2	-1
-2	-4	-2
-1	-2	-1

0	0	0
0	**0**	0
0	0	0

+1	+2	+1
+2	+4	+2
+1	+2	+1

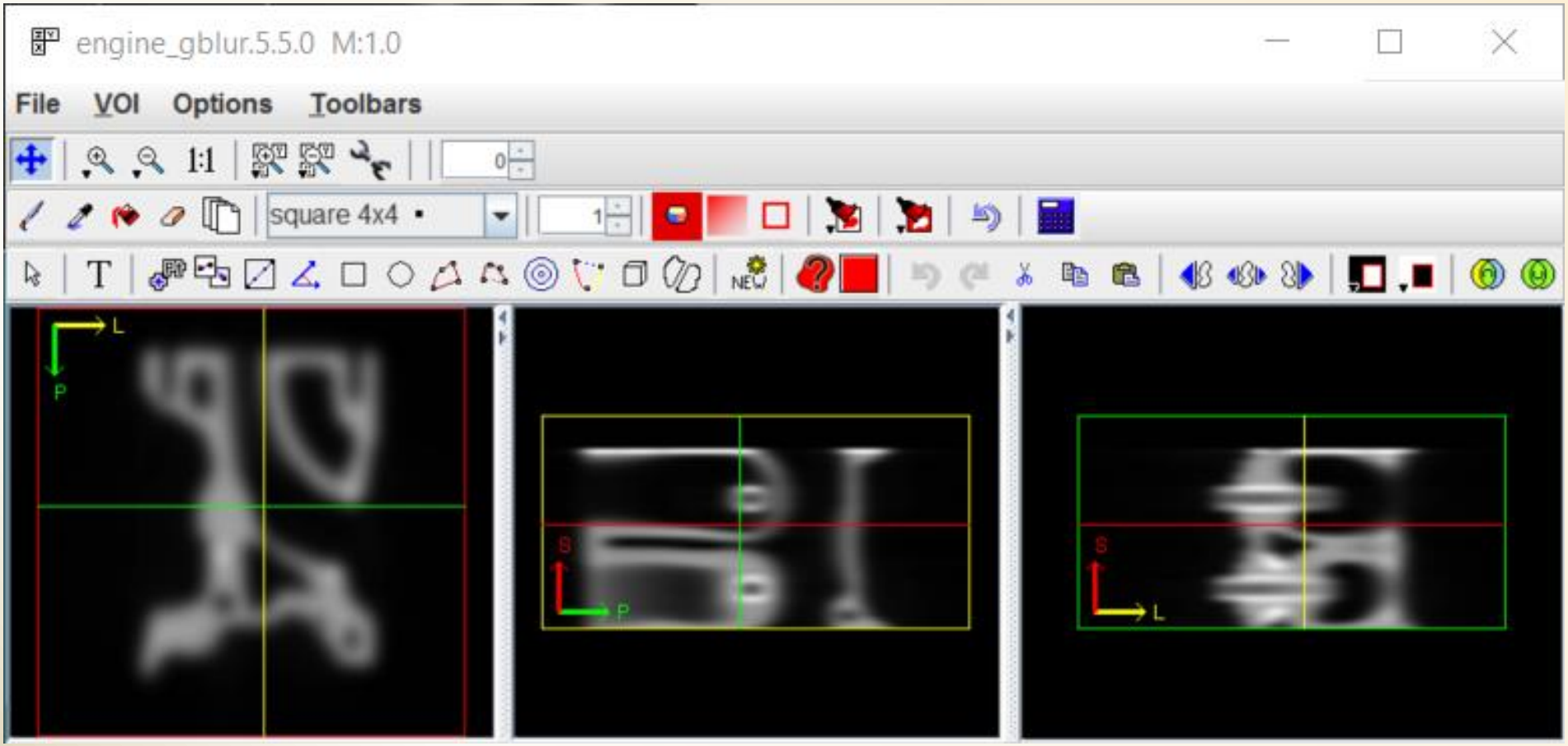
Pour détecter les contours, on peut prendre les "grandes" valeurs du gradient mais il faut trouver un seuil qui ne doit pas être trop sensible au bruit...

2D/3D : résultats différents !



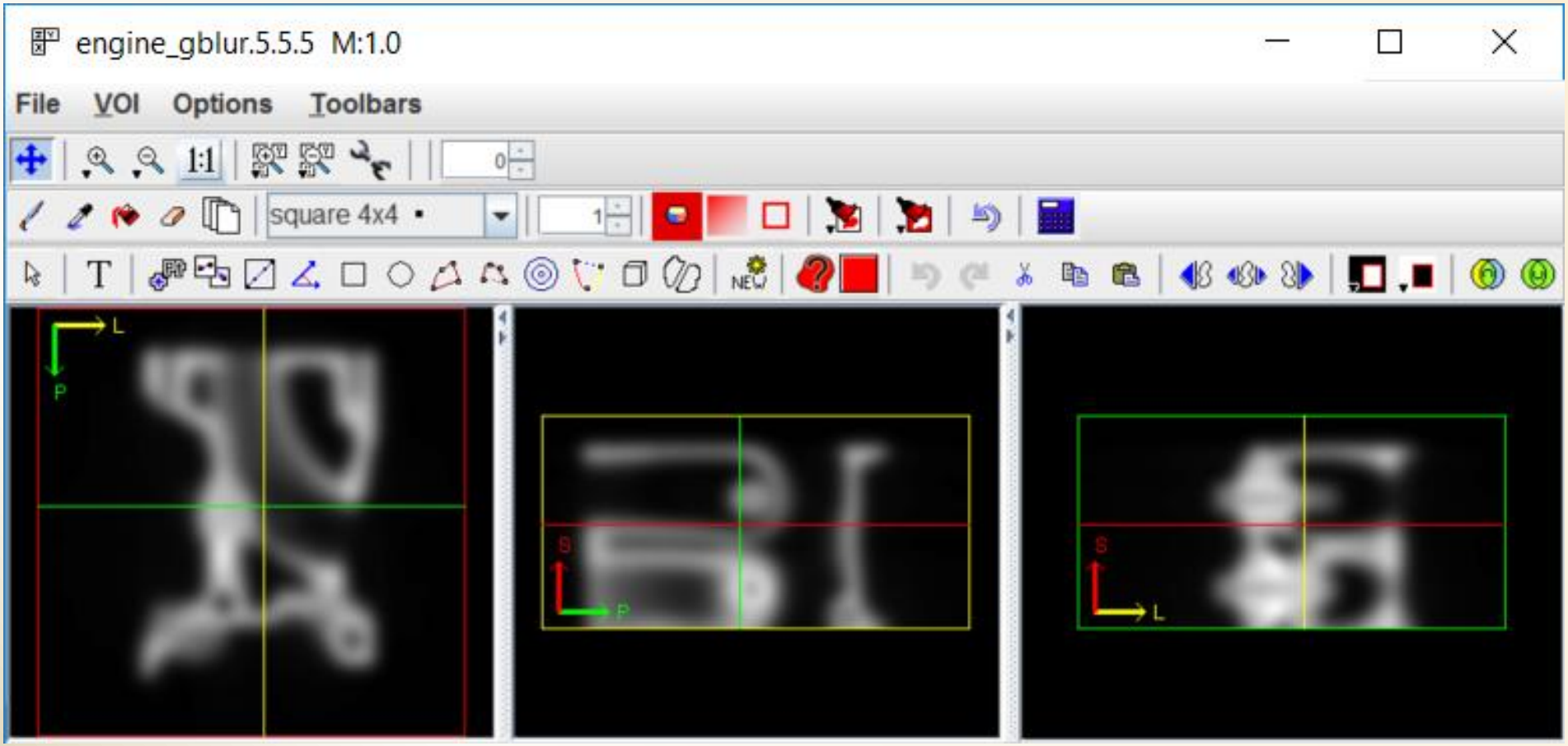
2D/3D : résultats différents !

Lissage 2D coupe à coupe (Gaussien, sigma =5)



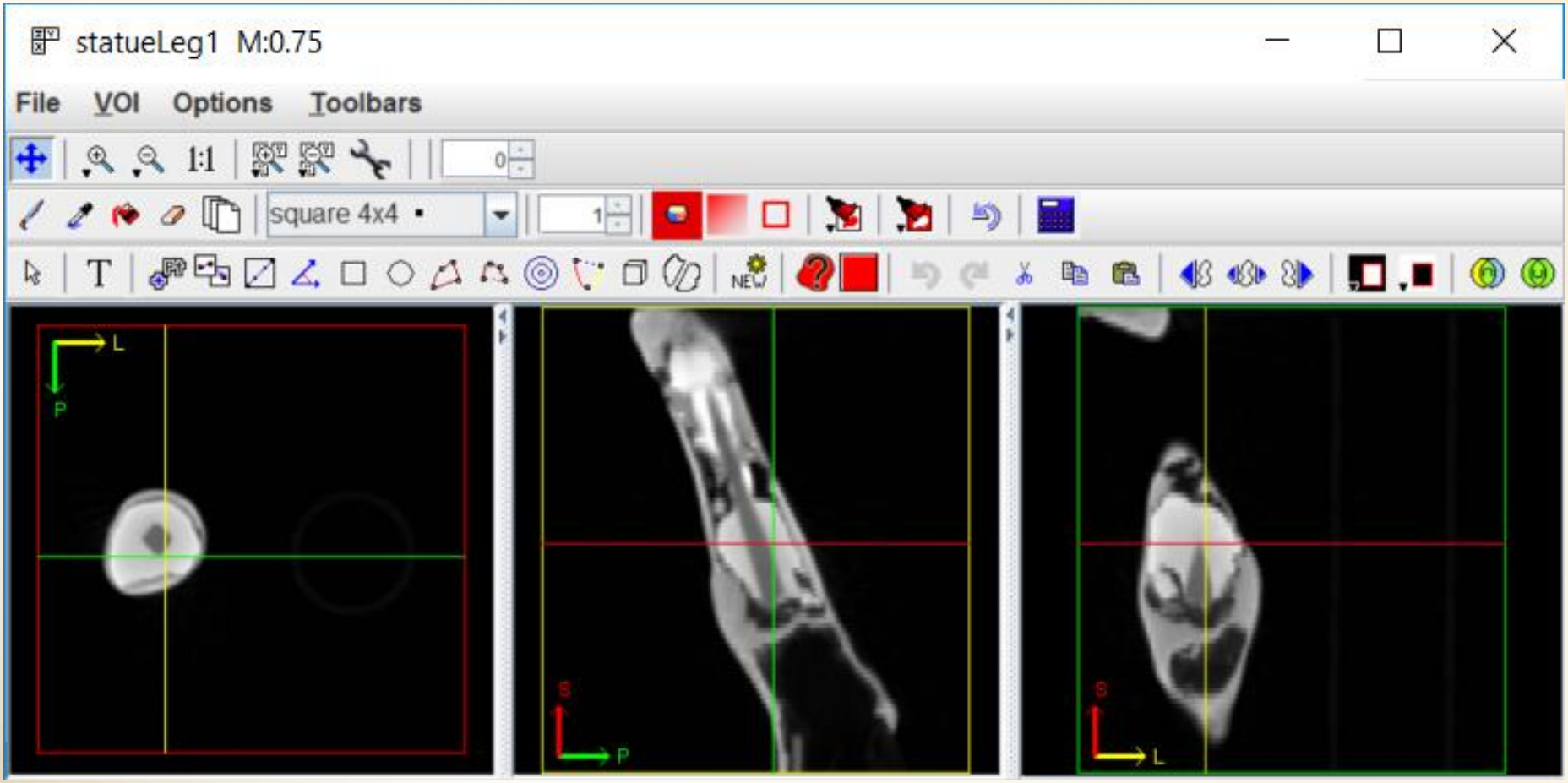
2D/3D : résultats différents !

Lissage 3D isotrope (Gaussien, sigma =5)



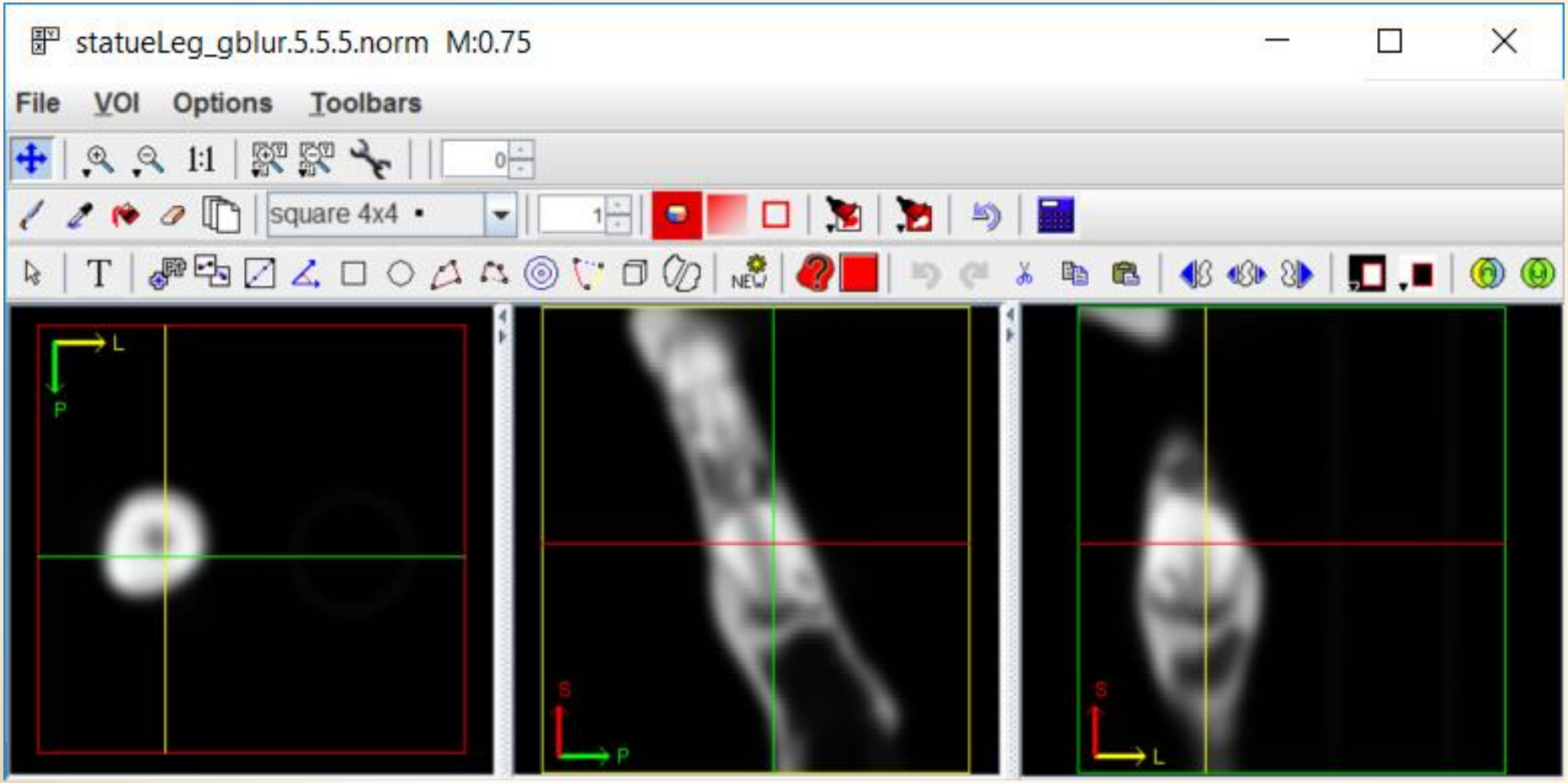
2D/3D : résultats différents !

Image anisotrope 1:1:4



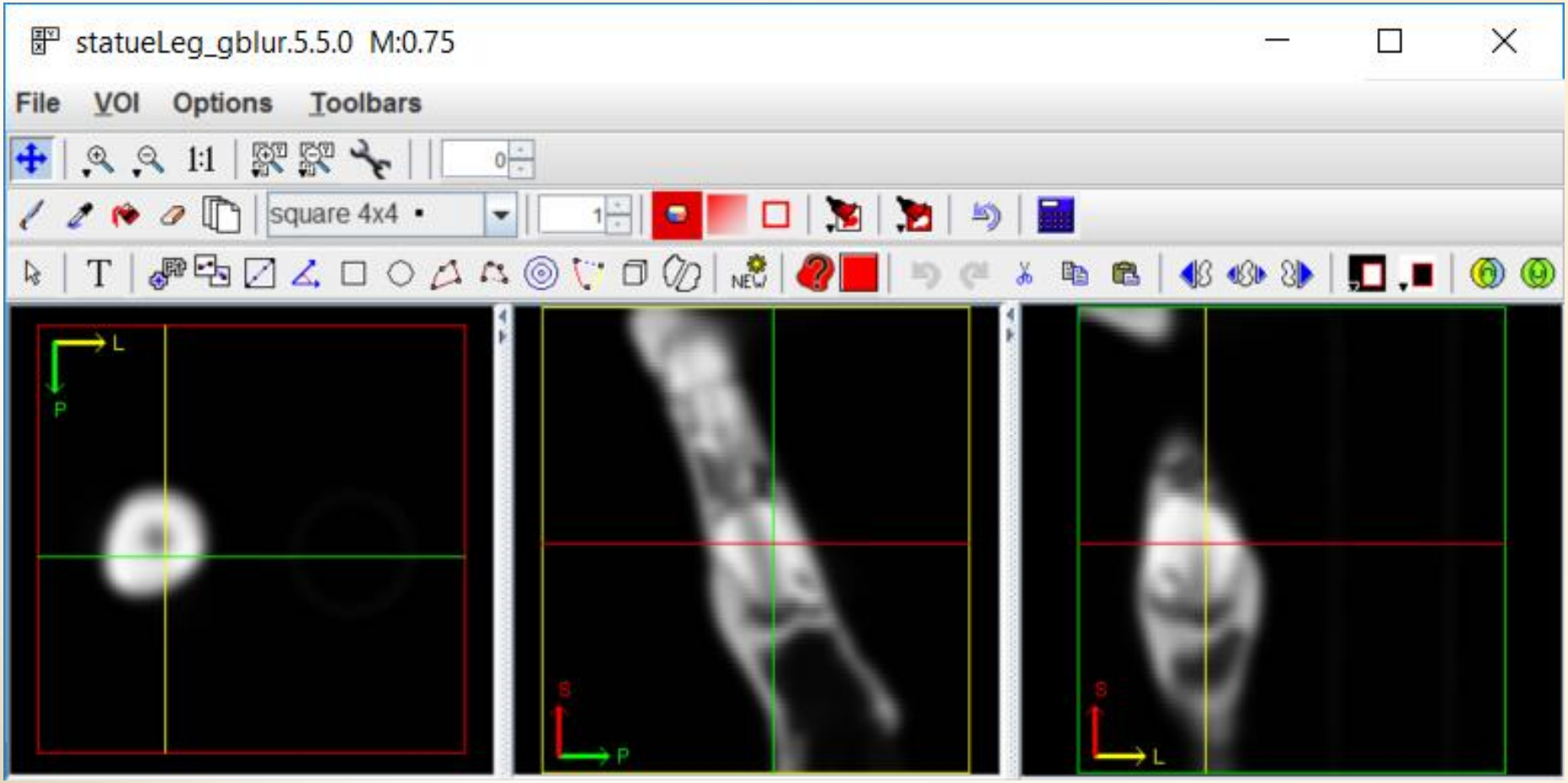
2D/3D : résultats différents !

Image anisotrope 1:1:4 – lissage Gaussien sigma=5 normé



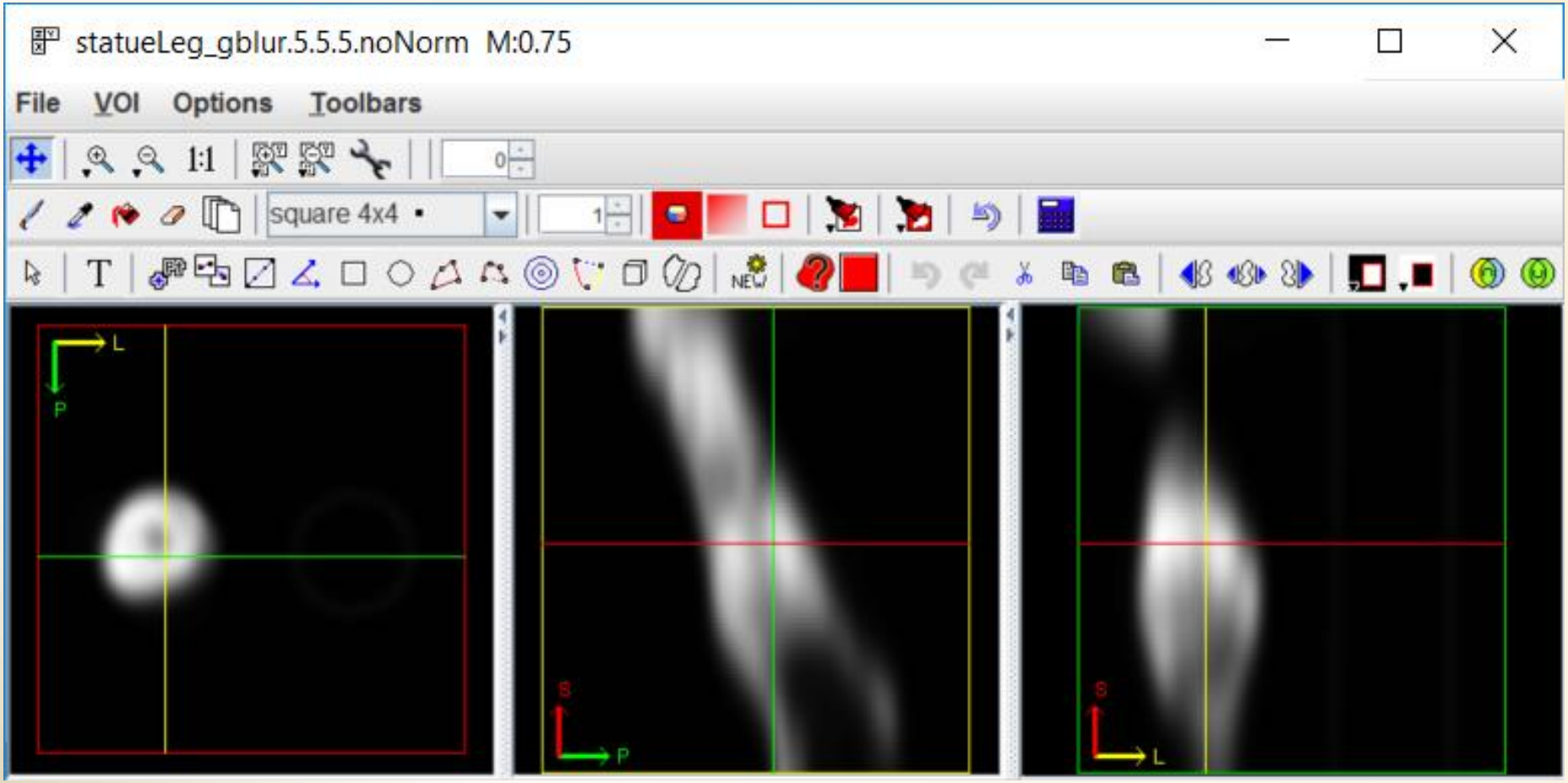
2D/3D : résultats différents !

Image anisotrope 1:1:4 ≈ lissage coupe à coupe



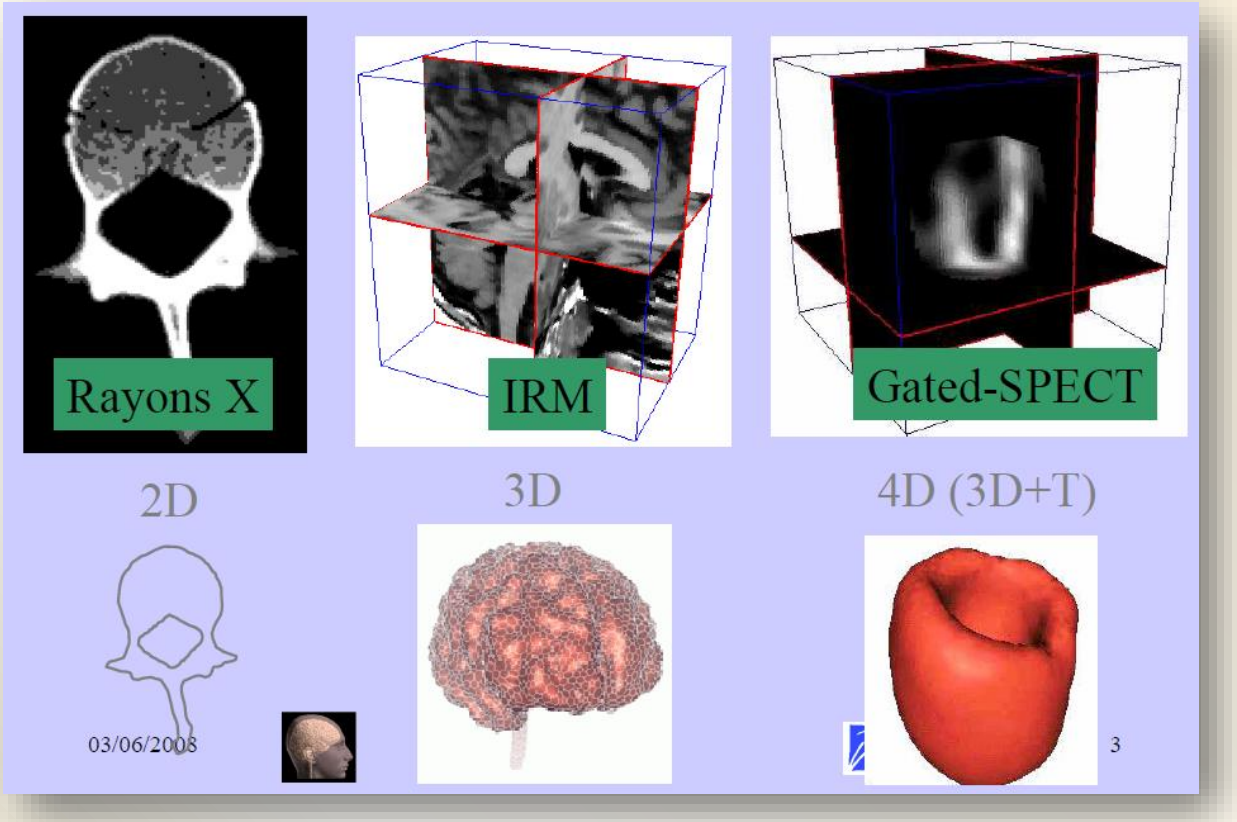
2D/3D : résultats différents !

Image anisotrope 1:1:4 – <> lissage Gaussien sigma=5 non normé



Le problème de la segmentation (1)

- Pour les applications, on a besoin de :
  - Définir une RDI ou ROI (**Region Of Interest**)
  - Définir la surface de la structure étudiée
  - Obtenir des paramètres quantitatifs (ex. volume)
  - …



Une méthode « région » simple : le seuillage

La structure que l'on cherche à segmenter est caractérisée par ses niveaux de gris dans l'image 3D → seuillage de l'intensité de l'image

MANIX

Tête : ~500-700
Os : ~1200

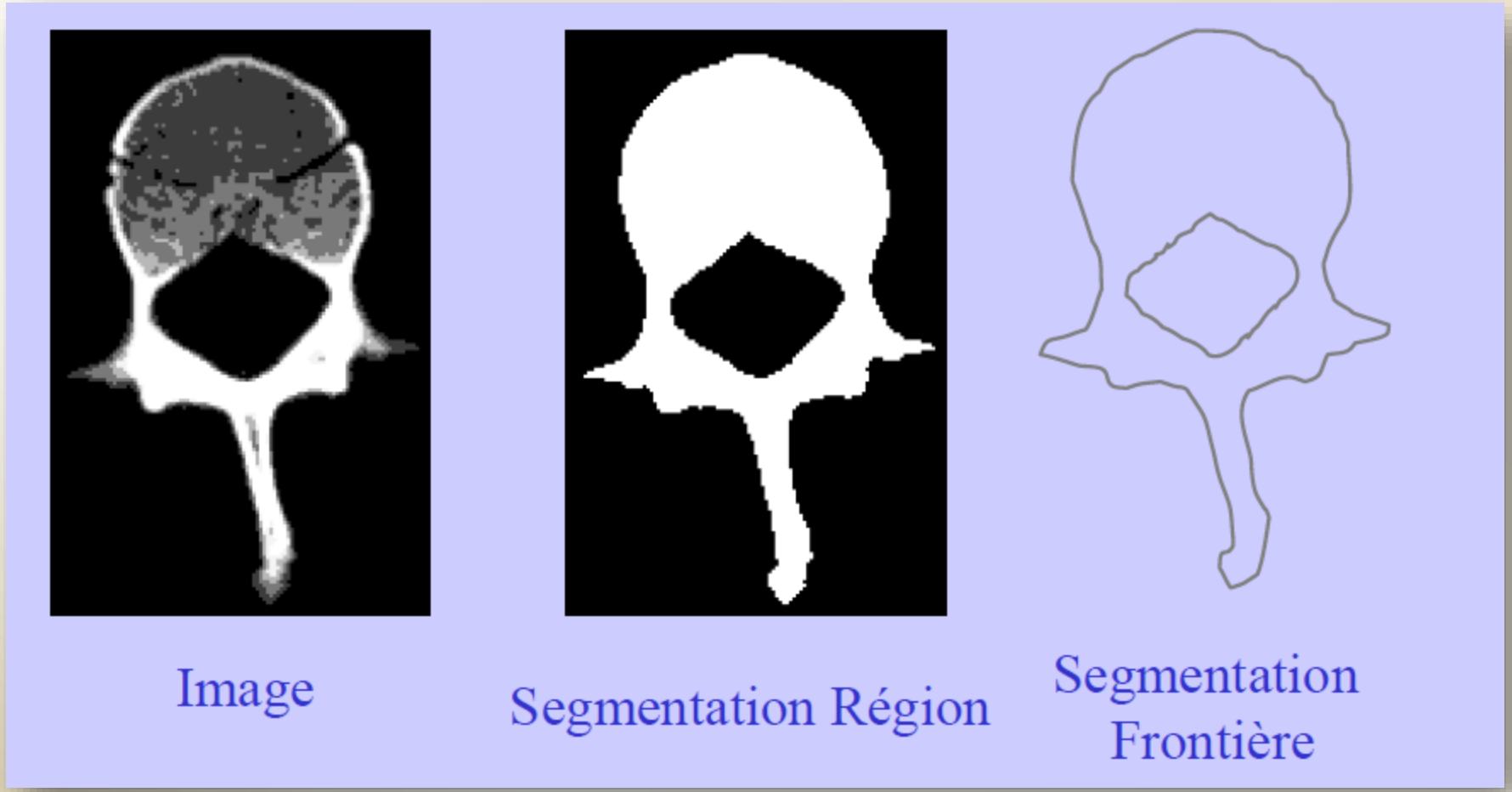
Engine

Structure 1 : pic autour de 140
Structure 2 : > 170

Mais…. Comment trouver un seuil significatif et reproductible à partir de l'histogramme ?

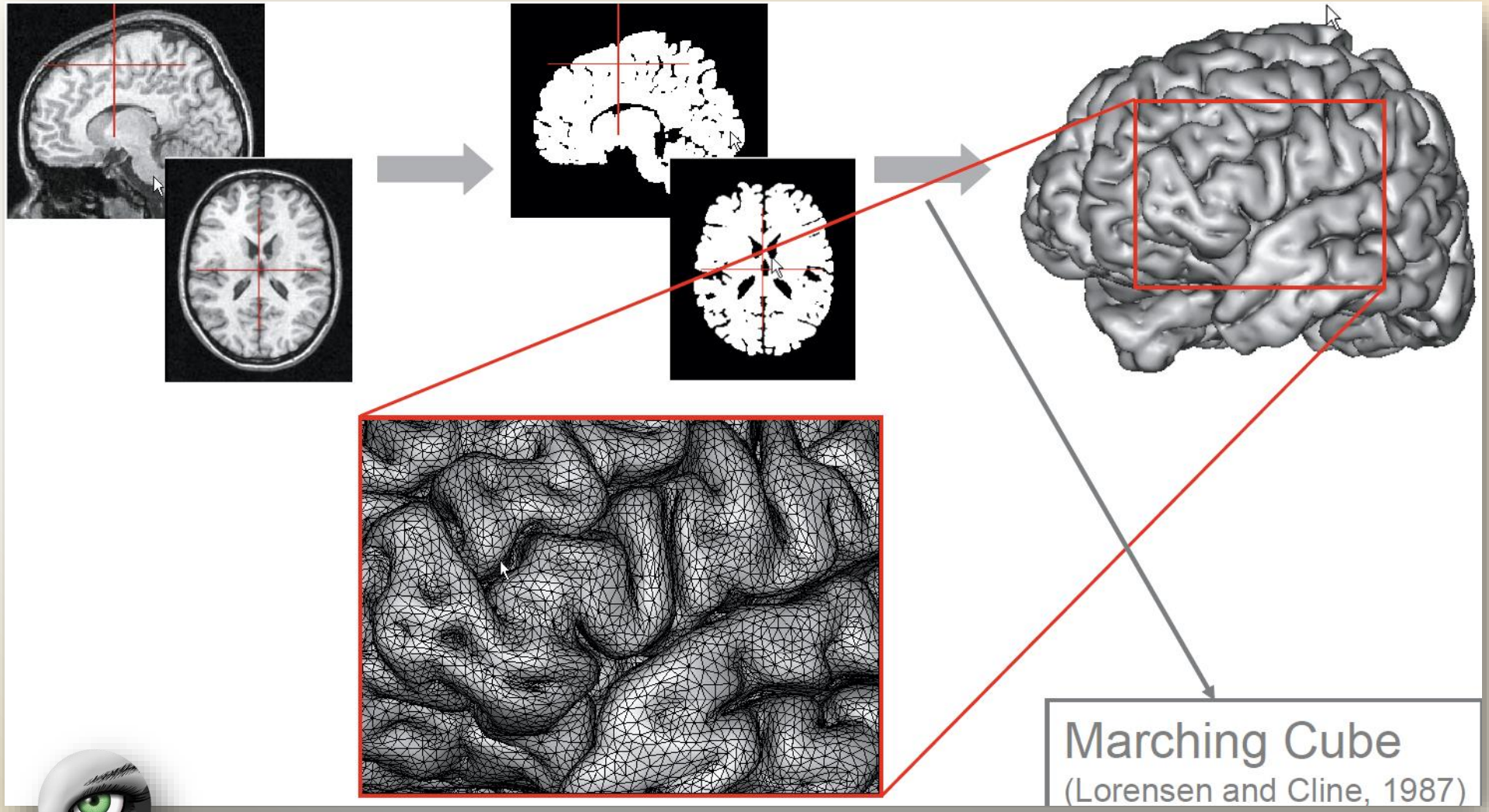
Mais comment visualiser la RDI ?

→ On a une RDI (ou ROI) définie dans l'image par une région ou une frontière
→ Forme une surface



Mais comment visualiser la RDI ?

→ Visualisation surfacique



→ Obtenir un maillage **3D** représentant la surface-frontière de la ROI dans l'image segmentée

Extraction d'une iso-surface

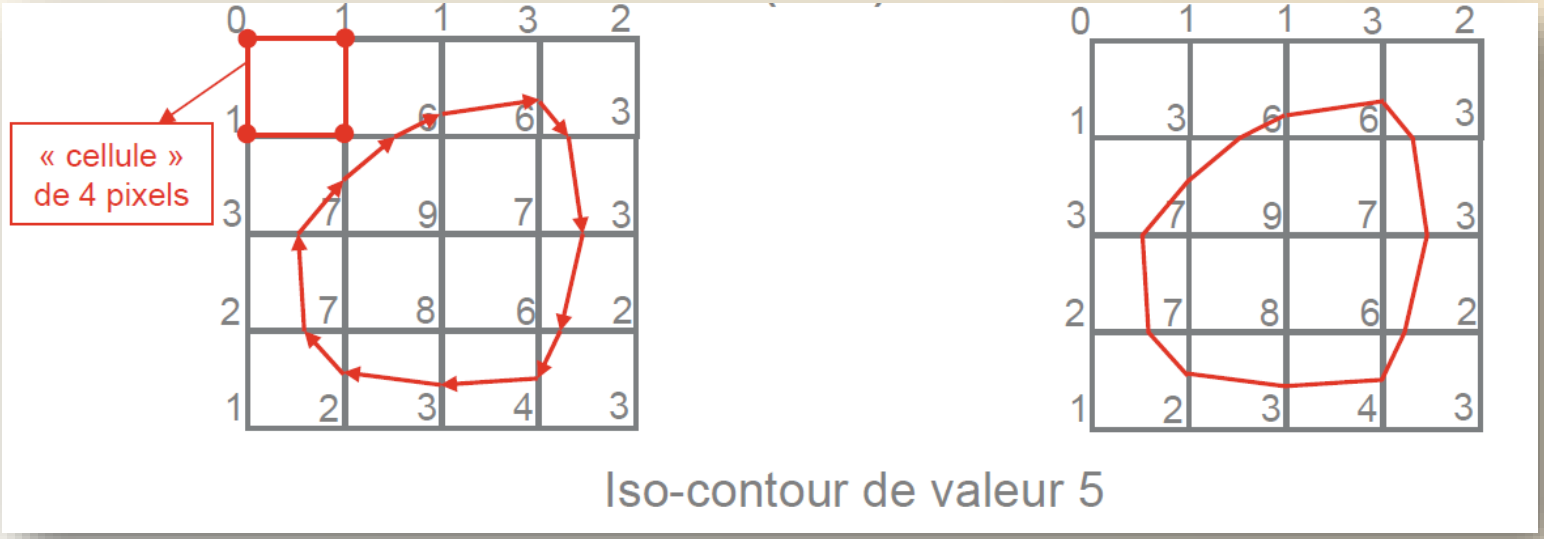
•L'**iso-surface** (resp. iso-contour en 2D) est la surface (resp. le contour) qui « passe » par les voxels (resp. pixels) d'une intensité donnée.

•Représentation implicite : $I(x,y,z)=I_0$

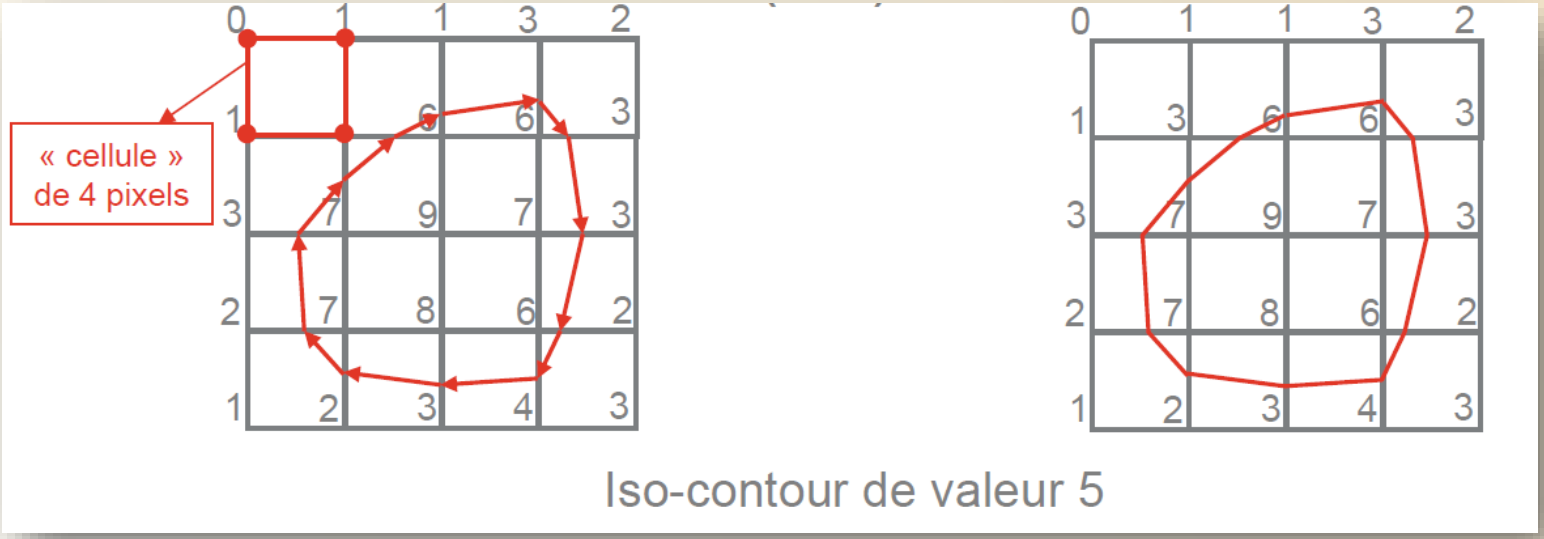
•Surface (resp. contour) fermée

→ maillage surfacique de la structure seuillée à I_0
→ si l'image a été préalablement segmentée : binaire $I_0 = 255$ ou 0.5

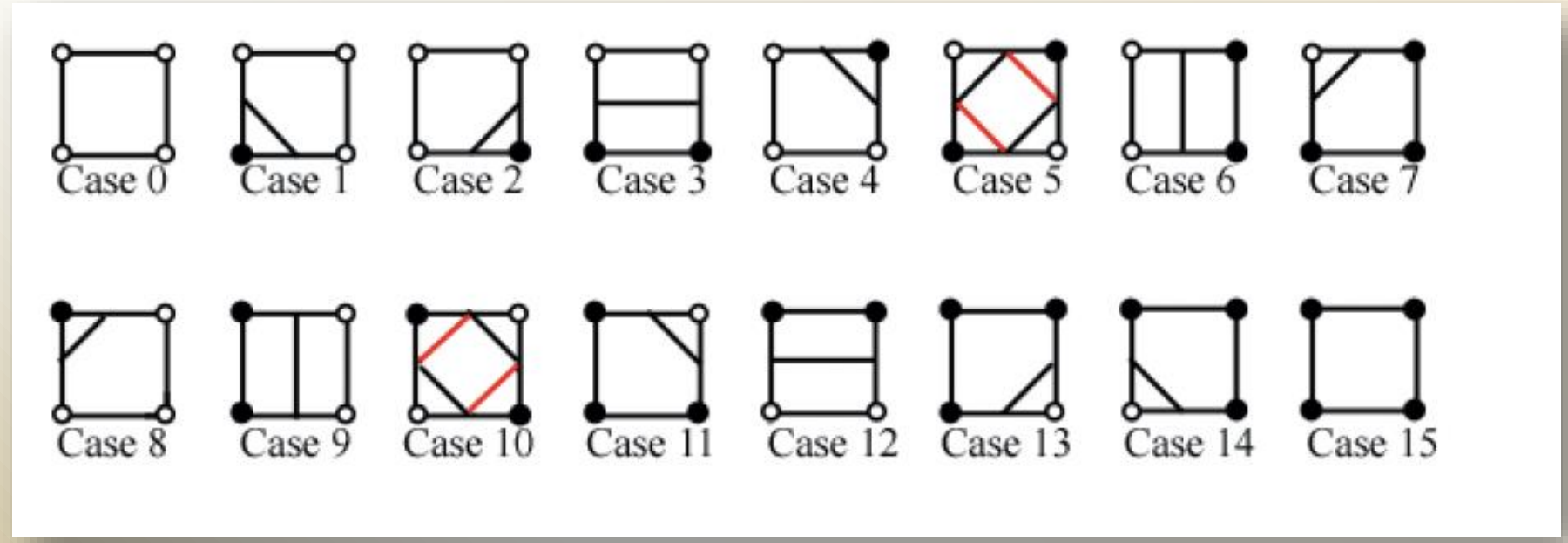
Etudions d'abord le problème en 2D !

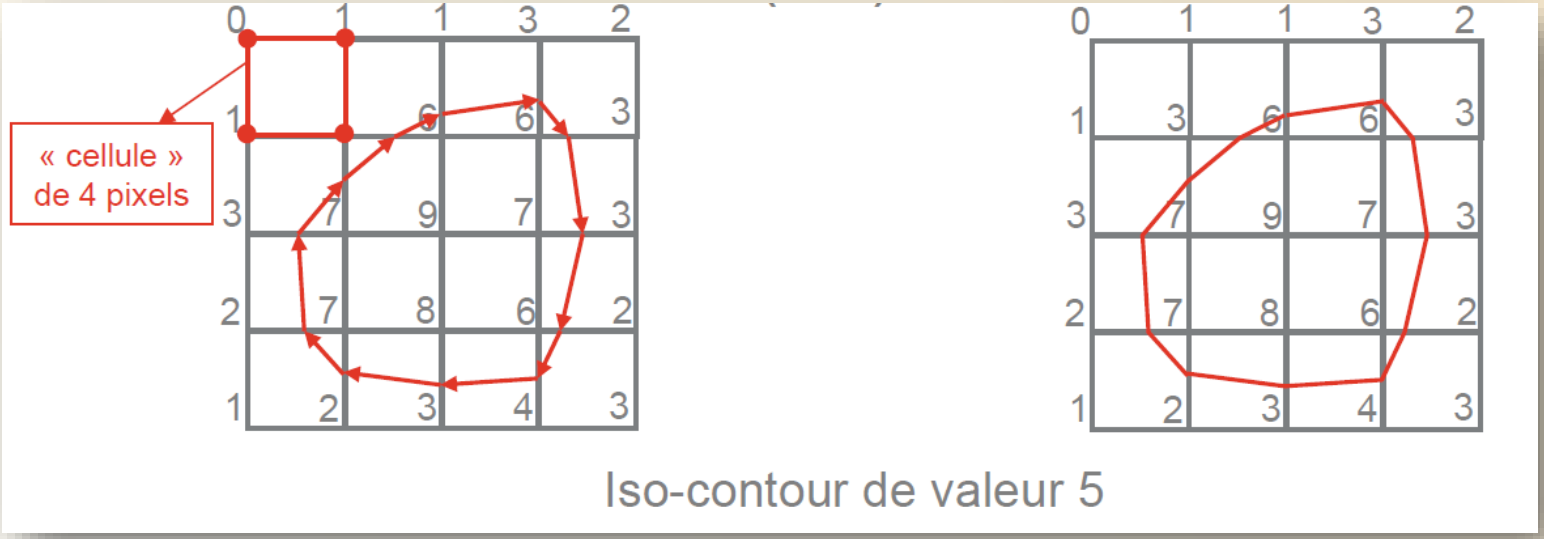


- Création de la grille 2D/3D reliant les centres des pixels/voxels
- Examen de toutes les cellules (= 4 pixels ou 8 voxels).
- Un sommet de la cellule est en dehors [$I(x,y) < I_0$] ou à l'intérieur [$I(x,y) > I_0$] du contour.

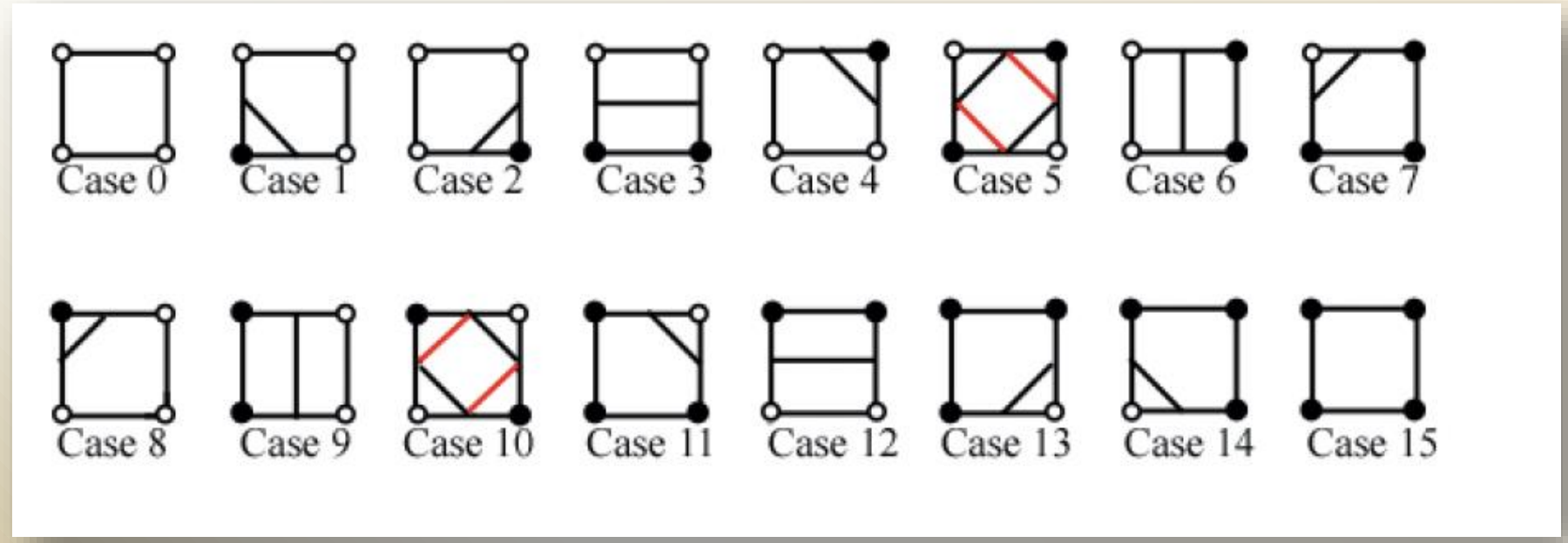


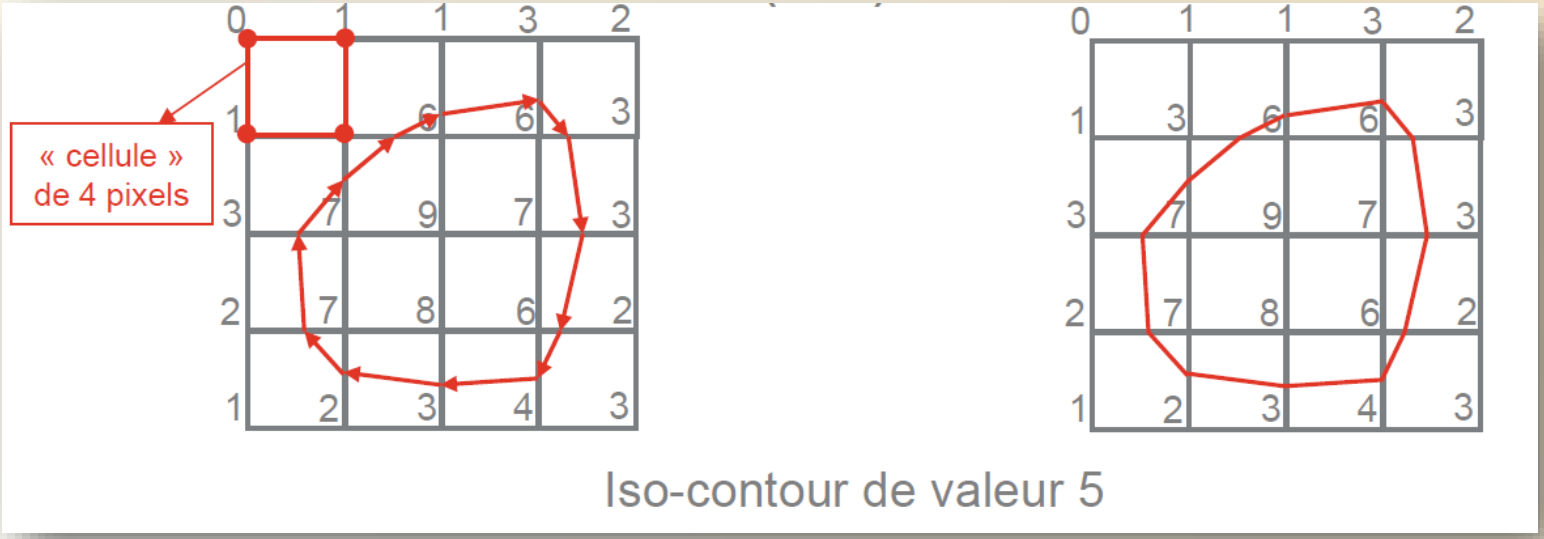
- Création de la grille 2D/3D reliant les centres des pixels/voxels
- Examen de toutes les cellules (= 4 pixels ou 8 voxels).
- Un sommet de la cellule est en dehors [$I(x,y) < I_0$] ou à l'intérieur [$I(x,y) > I_0$] du contour.



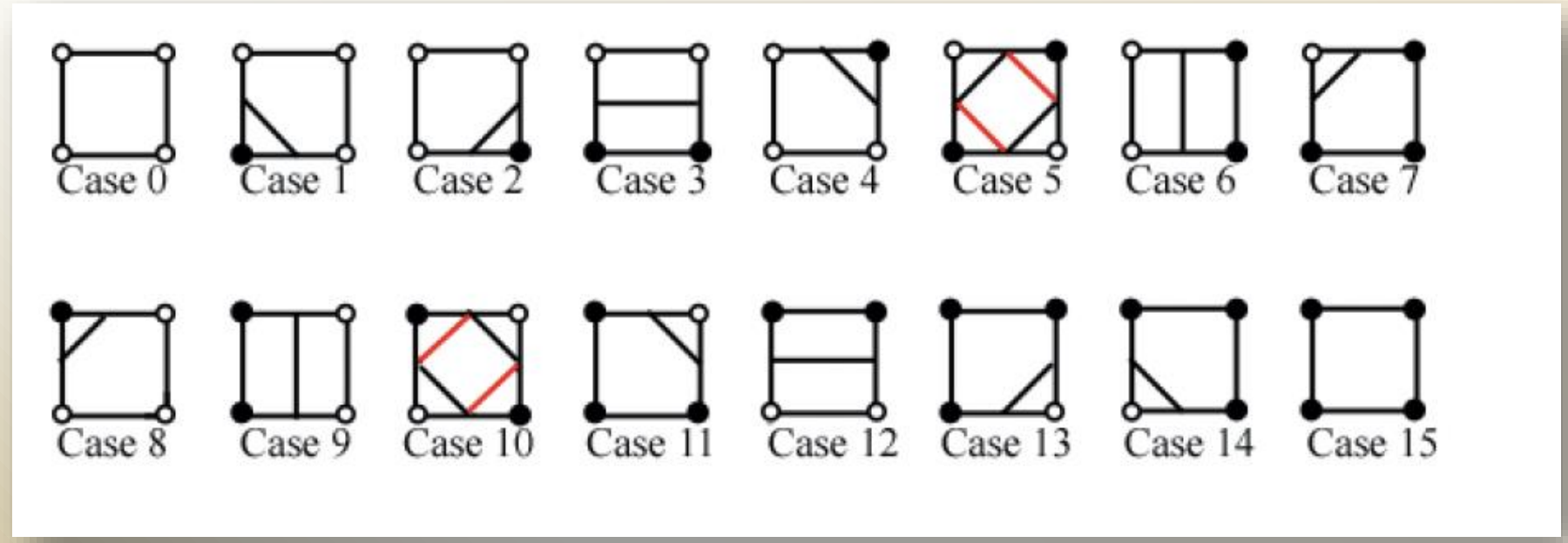


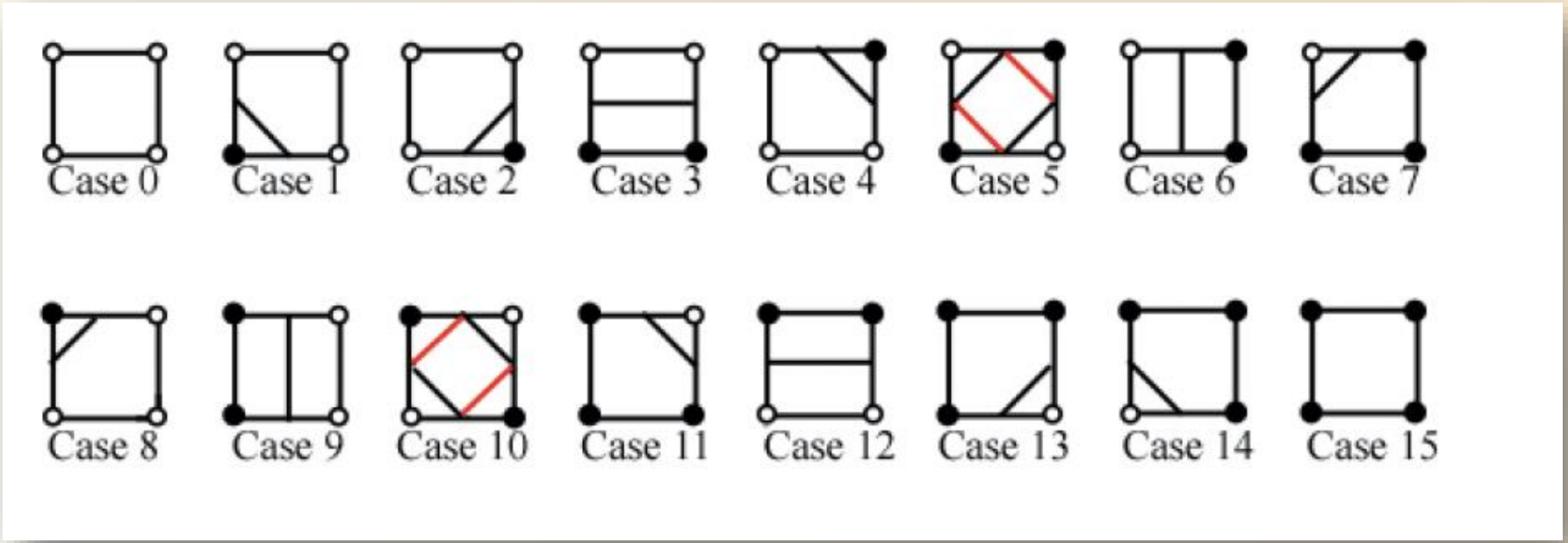
- Examen de toutes les cellules (= 4 pixels ou 8 voxels).
- Un sommet de la cellule est en dehors [$I(x,y) < I_0$] ou à l'intérieur [$I(x,y) > I_0$] du contour.
- Pour une cellule, un nombre fini ($2^4=16$) de configurations est possible.



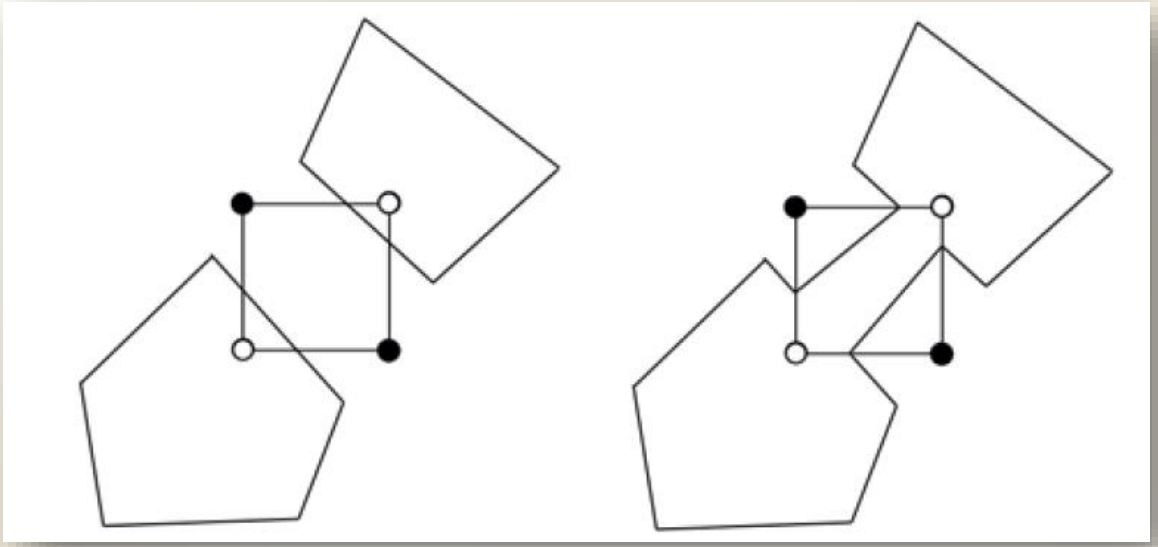


- On peut alors définir un ou plusieurs segments correspondant au contour
- Qui vont se rabouter de cellule en cellule….
- Pour former une représentation discrète du contour

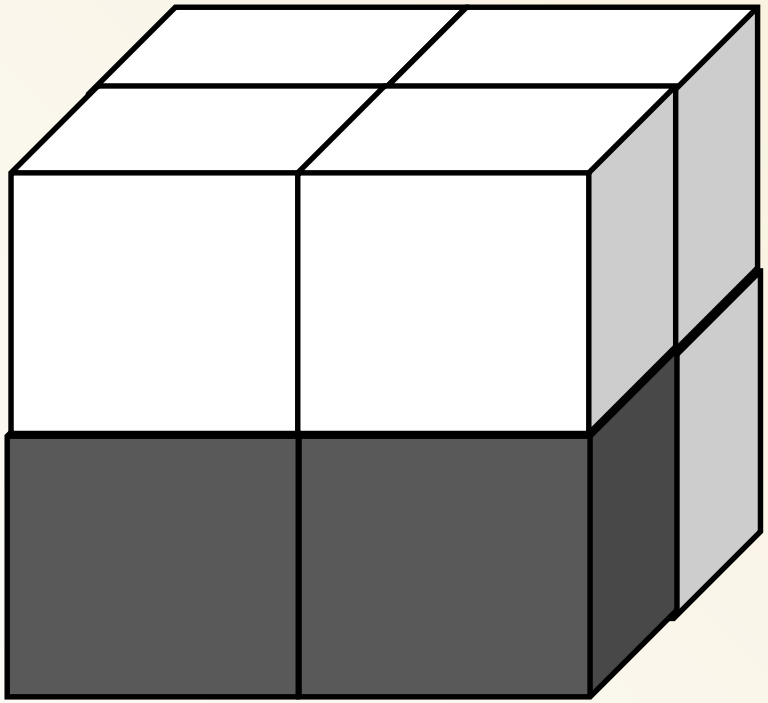




- D’un point de vue informatique, algorithme très facile à programmer.
- Il peut y avoir des ambiguïtés (case 5 / 10) mais le contour reste bien fermé.

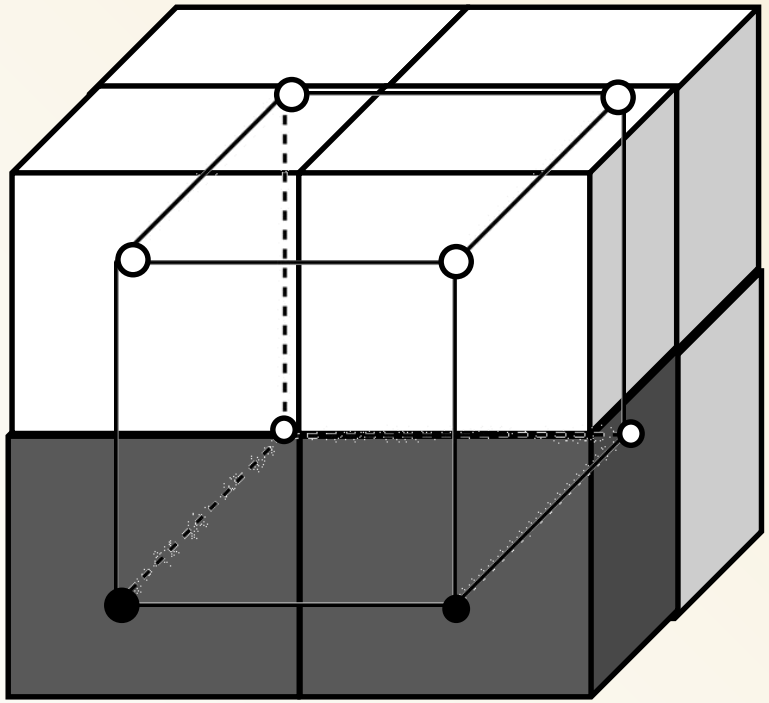


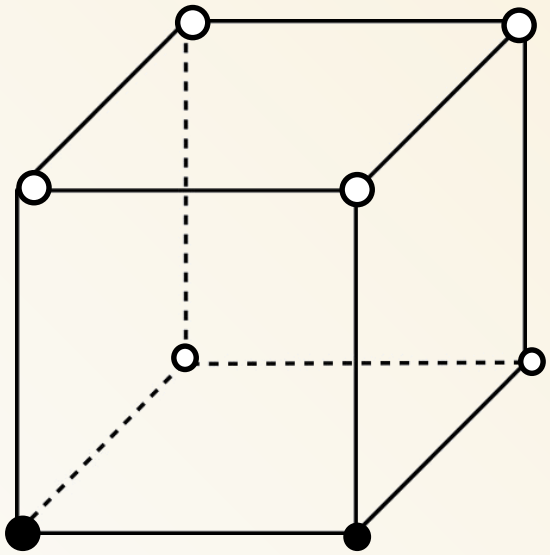
En 3 dimensions…

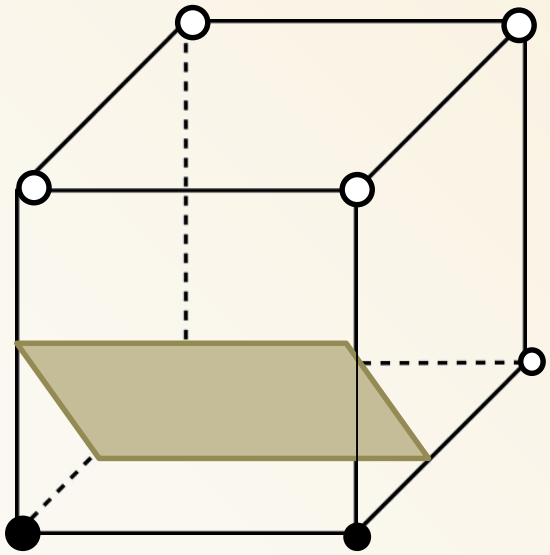


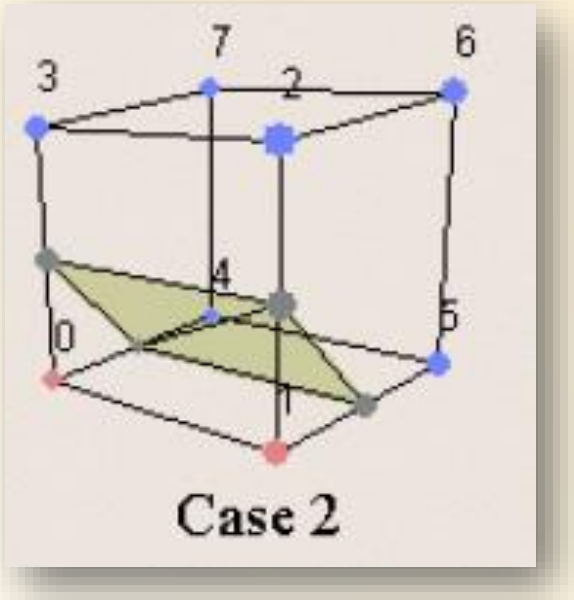
→ Algorithme du **Marching Cubes** qui va construire la surface par groupes de 2x2x2 voxels.

Lorensen, W. E.; Cline, Harvey E. (1987). "Marching cubes: A high resolution 3d surface construction algorithm". ACM Computer Graphics 21 (4): 163–169.







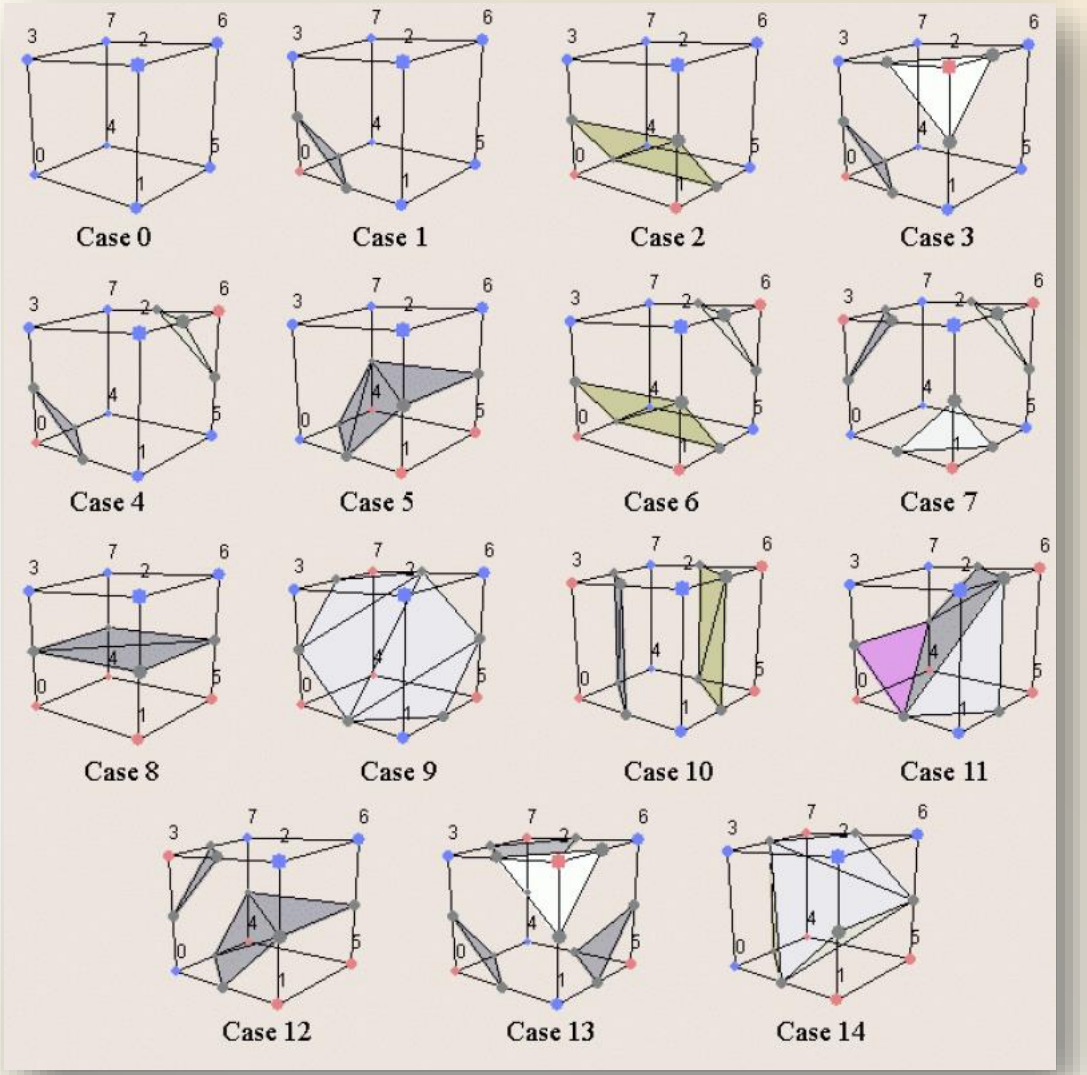


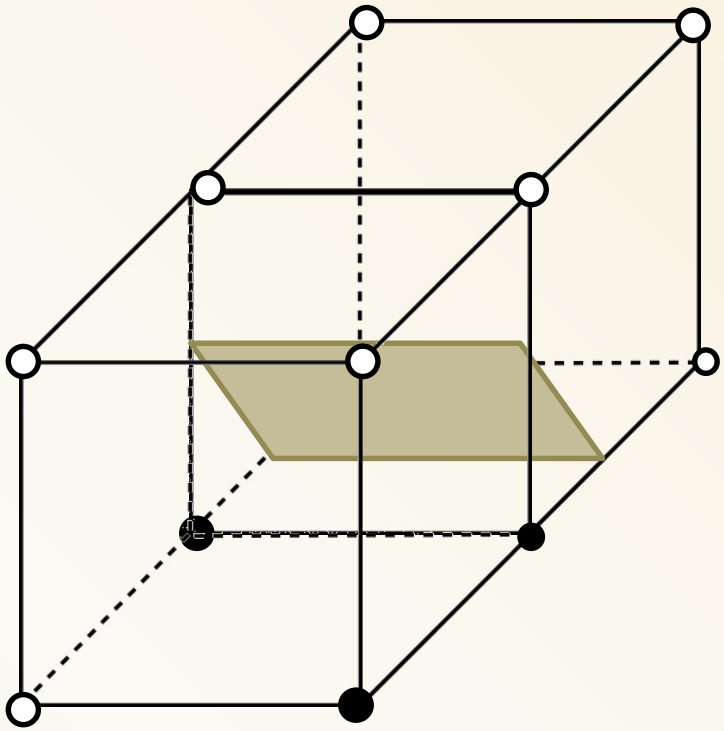
•Toutes les configurations (33) ont été listées.

Le problème se généralise en 3D avec 2^8= 256 configurations possibles.

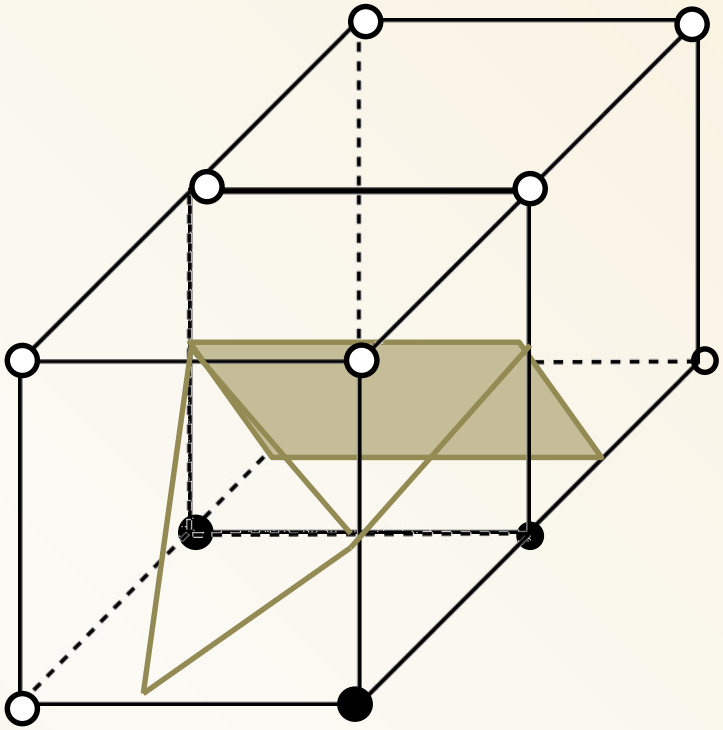
→ 15 configurations de base, les autres par symétrie

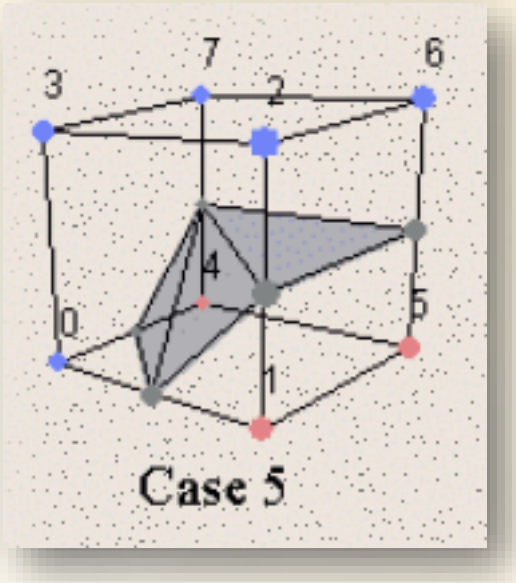
Et on peut construire un ou plusieurs polygones qui représentent l'isosurface à l'intérieur de la cellule..



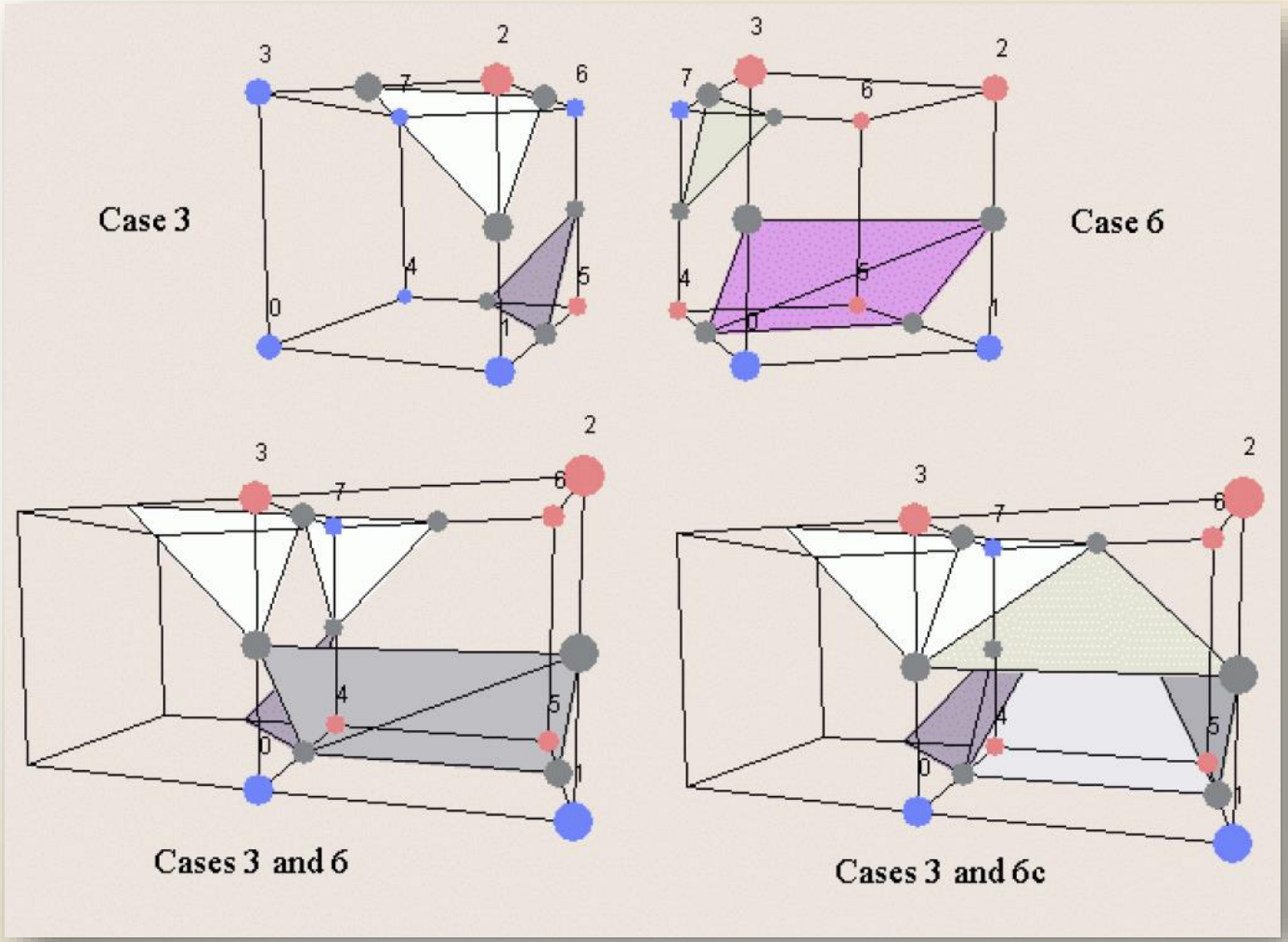


•Nouveau bloc de 2x2x2 voxels

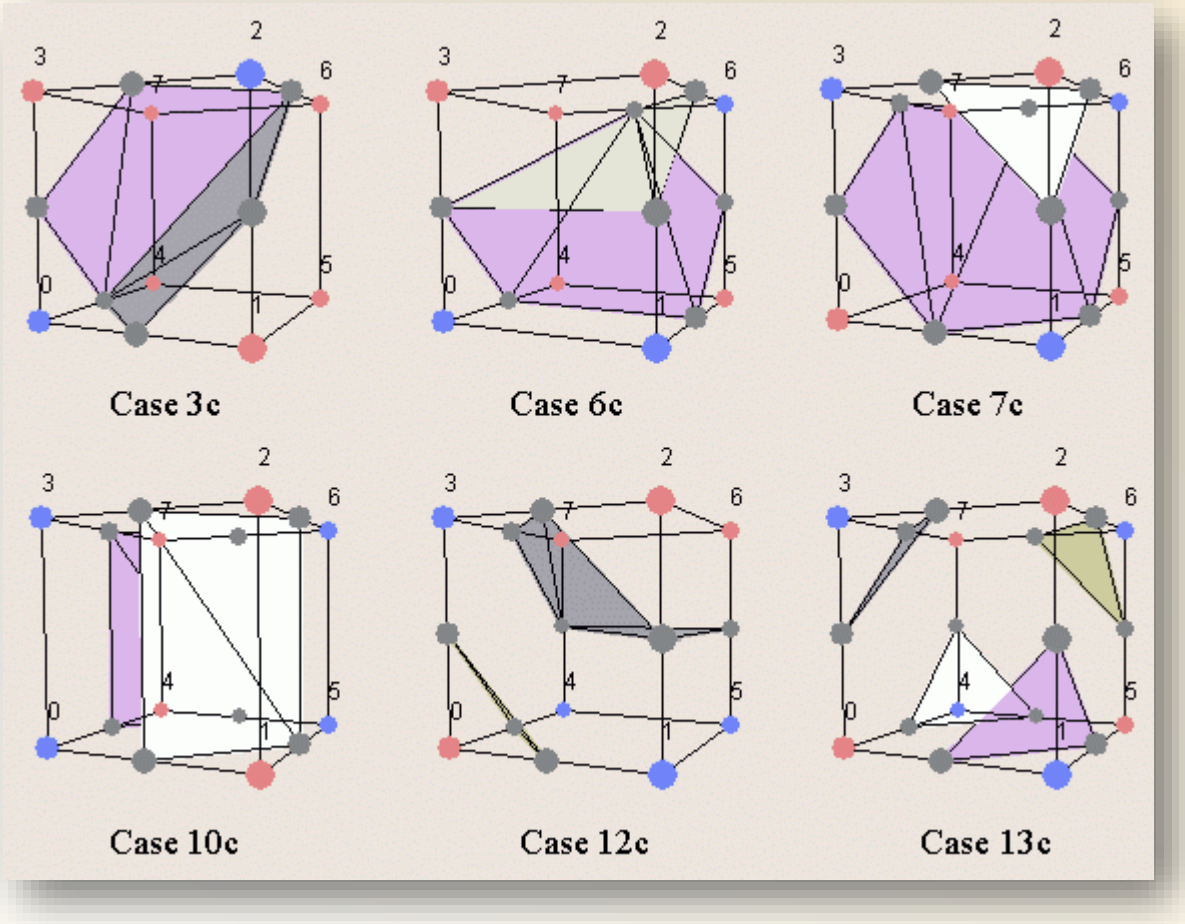



3
7
2
6
4
0
5
1
Case 5

Le problème est plus complexe qu'en 2D : le choix d'une configuration dans une cellule n'est pas indépendant des autres cellules → un mauvais choix peut générer des trous dans la surface.



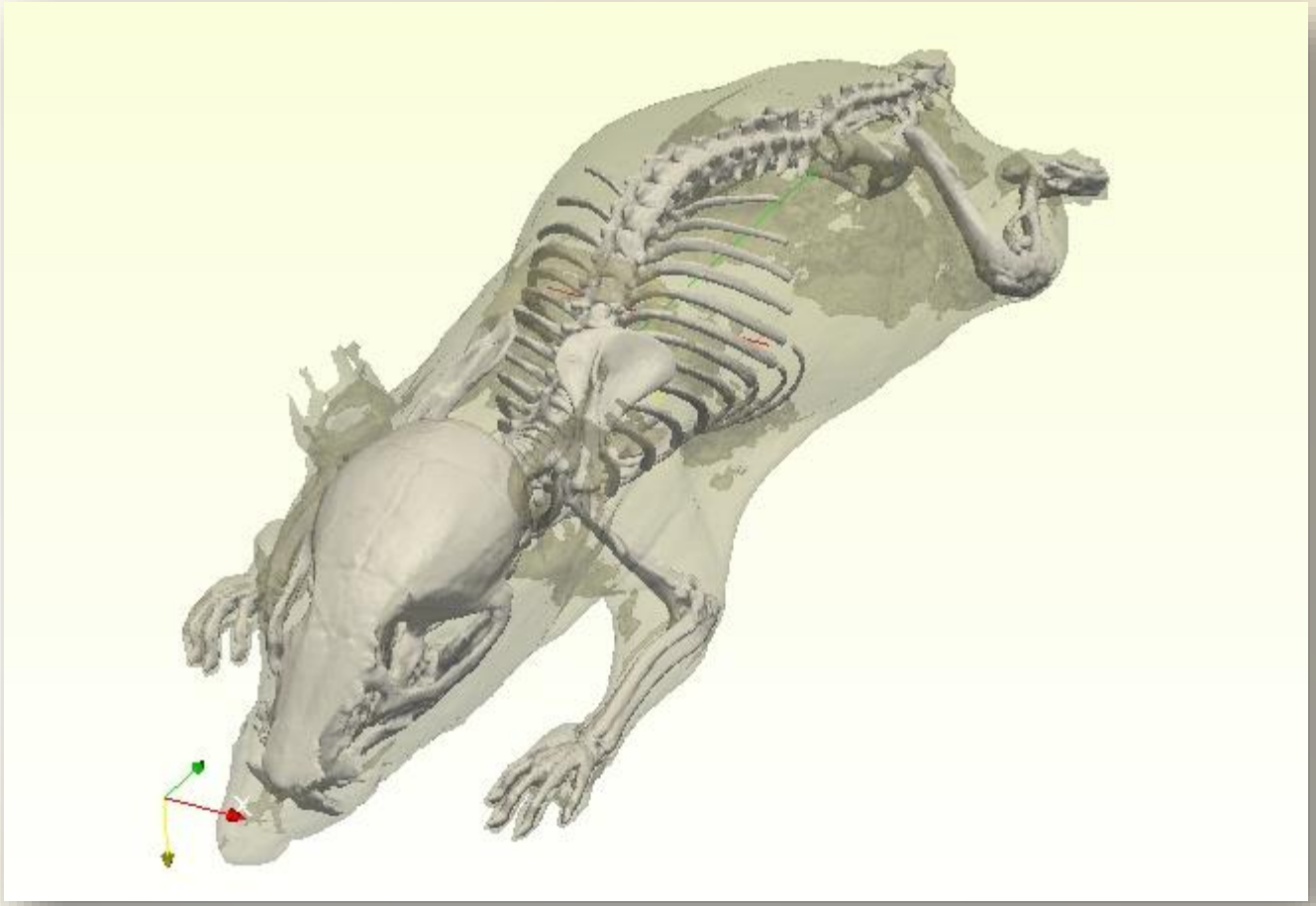
6 configurations complémentaires sont ajoutées pour résoudre les cas problématiques



Brain / engine (Otsu)

Visualisation surfacique

→ Une ou plusieurs surfaces **fermées** (sauf sur les bords de l'image 3D)
→ Visualisation avec des logiciels d'infographie ou de CAO
→ Effets graphiques (ombrage, transparence…)
→ Simulation possible (découpe par exemple)



Pour en savoir plus…

HMIN318M « Imagerie 3D »

2018-2019

Mis à jour le 4 février 2019

http://www.lirmm.fr/~subsol/HMIN318/

Responsable : Gérard Subsol, Chargé de Recherche CNRS, Equipe-projet ICAR, LIRMM http://www.lirmm.fr/~subsol/
Contact : gerard.subsol@lirmm.fr

Description

Ce module a pour but de :

- Découvrir et approfondir l'ensemble des techniques de traitement des images numériques 3D.
- Appliquer concrètement ces techniques dans des cas pratiques, dans le domaine médical, industriel, de la science des matériaux, du patrimoine culturel…
- Découvrir le rôle de l'ingénieur et de l'informaticien en imagerie 3D.

Le module abordera les points suivants :

- Systèmes d'acquisition 3D et modalités d'imagerie 3D
- Visualisation 3D
- Segmentation
- Recalage
- Applications dans différents domaines : médical mais aussi biologique, industriel, patrimoine

---

Organisation

Volume horaire des enseignements : 45 heures (cours : 18 h / TP + projet sur ordinateur : 27 h)

Les séances se dérouleront **sauf exception** dans la salle de TP informatisée du Département informatique (bâtiment 16 du campus Triolet de l'Université Montpellier (http://www.umontpellier.fr/wp-content/uploads/2014/06/Campus-Triolet-UM.pdf).

Modalités de contrôle de connaissances (voir http://sciences.edu.umontpellier.fr/files/2016/12/règlement-des-examens-master-2017-2018.pdf)

- 60 % : contrôle écrit avec documents de 2 h ;

Sources

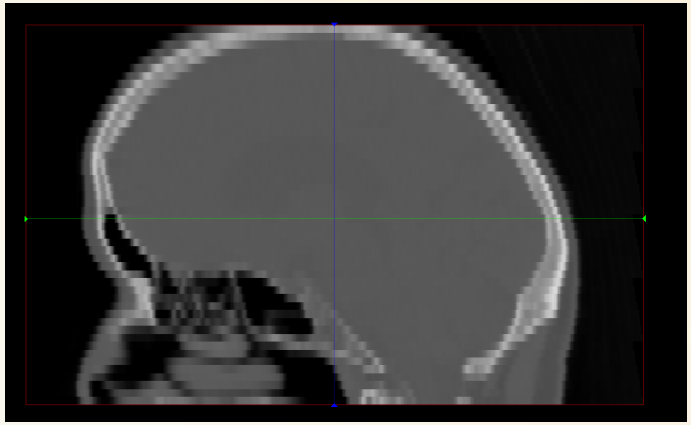
• Visualisation scientifique - Algorithmes : Marching cube/Diagrammes de Voronoï et triangulation de Delaunay, O. Coulon et S. Mavromatis
http://www.esil.univ-mrs.fr/~ocoulon

• Wikipedia

Images 3D

Raw data : 2 octets / unsigned short

engine : http://www.volvis.org
256 x 256 x 128
1:1:1
Two cylinders of an engine block
GE Industrial CT-Scanner

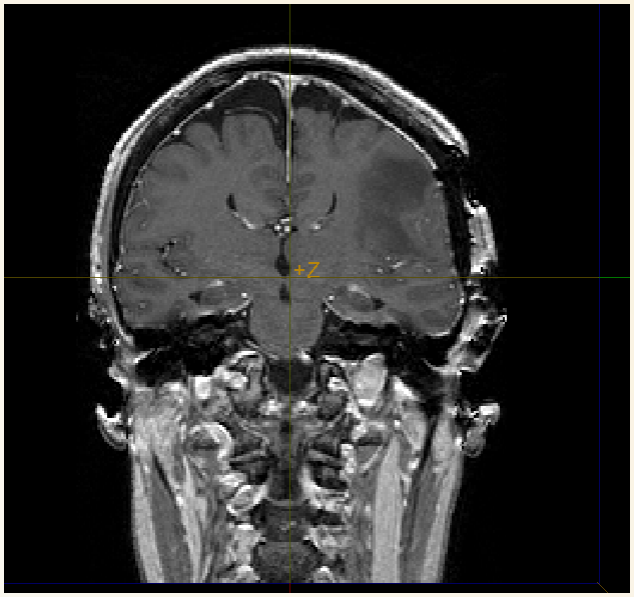


MANIX : http://pubimage.hcuge.ch:8080/
512 x 512 x 48
0.457 x 0.457 x 3.0 mm
Head CT

Images 3D

Raw data : 2 octets / unsigned short

http://pubimage.hcuge.ch:8080/
BEAUFIX
448 x 576 x 72
0.625 x 0.625 x 1.4 mm
Contrast-enhanced renal MRA acquired on a 3T scanner. Normal study



http://pubimage.hcuge.ch:8080/
BRAINIX
256 x 256 x 100
0.9375 x 0.9375 x 1.5 mm
MR Brain tumor.

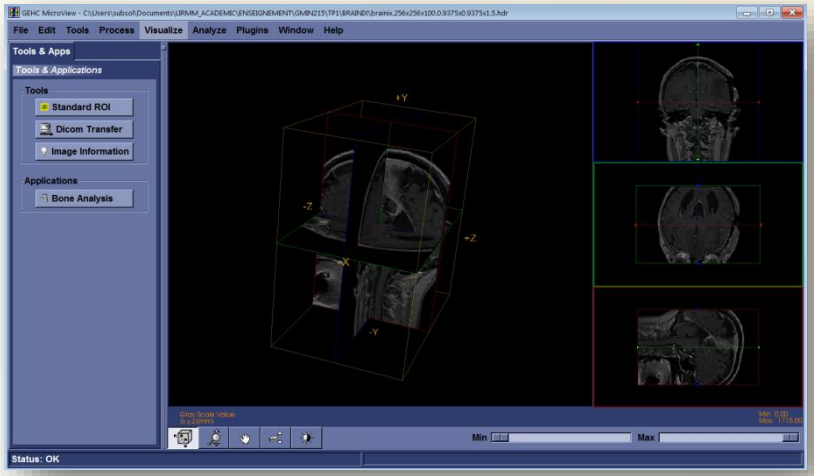
Images 3D

Raw data : 2 octets / unsigned short

engine : http://www.volvis.org
256 x 256 x 256
1:1:1
Rotational C-arm x-ray scan of a human foot.
Tissue and bone are present in the dataset.

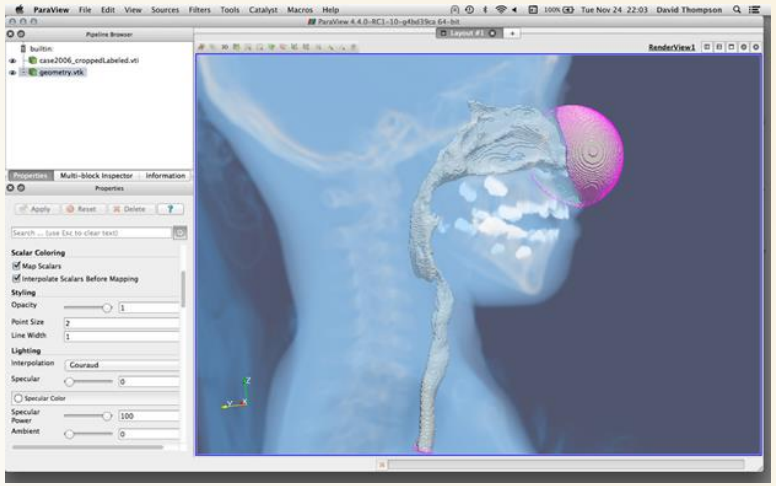
Logiciels

MicroView (Windows, Linux, MaxOs) : http://microview.sourceforge.net/webindex.html



Paraview (Windows, Linux, MacOs) : http://www.paraview.org/

ParaView



Logiciels

MeshLab (Windows, Linux, MaxOs) : http://meshlab.sourceforge.net/l