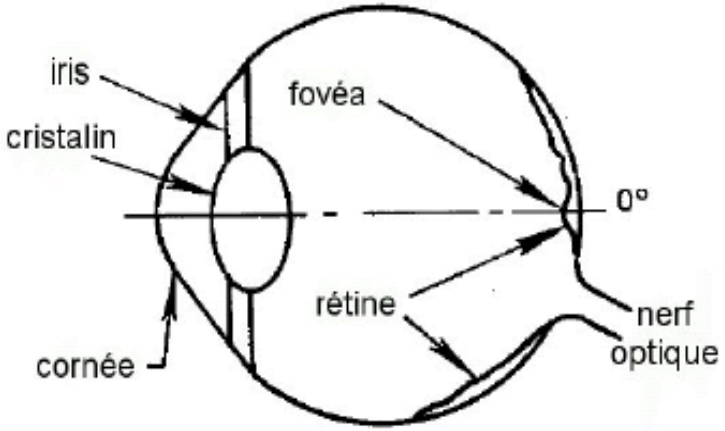
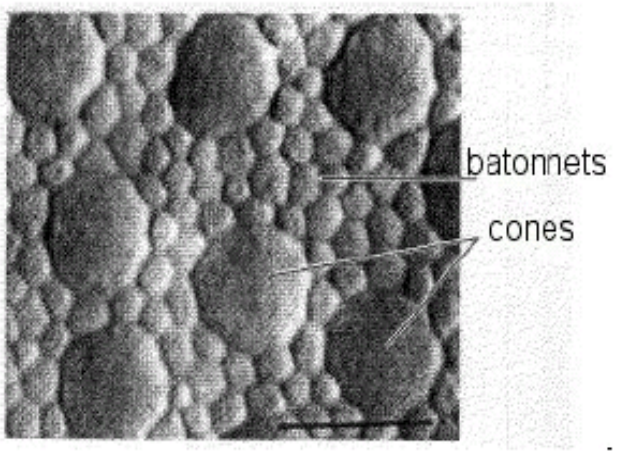
Image Processing and Analysis
Introduction for Computational Biology

http://www.imageprocessingbasics.com/
http://www.cosc.canterbury.ac.nz/mukundan/covn/applcovn.html

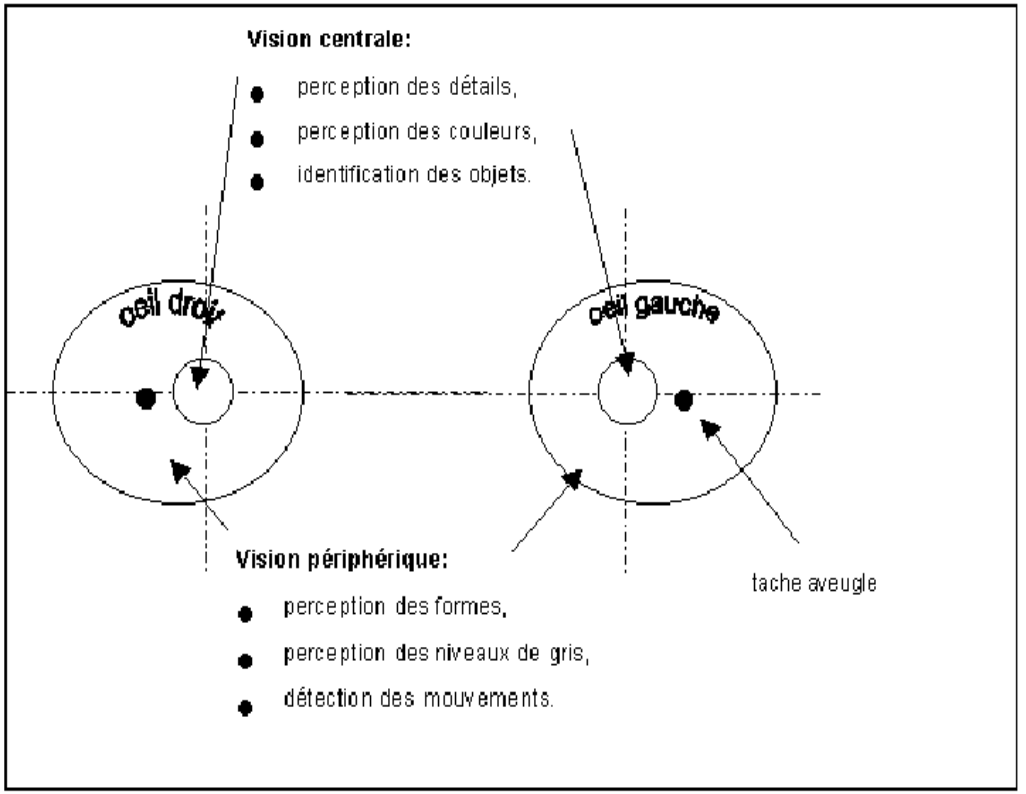
https://transparent-human-embryo.com/ Iconic most recent development in imaging

Vision des Primates





- Cristallin : lentille de focale et d'ouverture variable
- Rétine : couche photosensible, transducteur optique -> électrique
- Rétine : hétérogène en nature et densité des photorécepteurs
- Dans la fovéa, zone d'hyperacuité visuelle, 6 à 7 Millions de cônes exclusivement, à branchement synaptique simple : caméra CCD
- Champ visuel asymétrique + point aveugle (http://www.vonrechenberg.ch/blindspots.html)



Robotic vision

POINT GREY RESEARCH

Home | Site Map | Contact

Products | Sales | Support | News | Corporate

Quick Access

Find a Product

Knowledge Base Search

Go

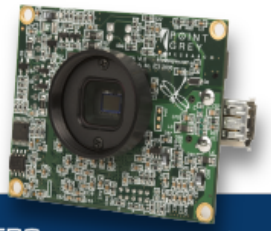
Downloads Login

Email Address

Password

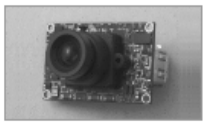
Login Forgot password?

NEW 1.3MP DRAGONFLY2



Sony 1/3" CCD · 1296x964 at 20 FPS view all

Point Grey Research® Inc. is a worldwide leader in the development of advanced digital camera technology products. With a number of local distributors throughout the world, Point Grey designs, manufactures and distributes IEEE-1394 (FireWire) cameras, stereo vision cameras and spherical digital video cameras to a broad spectrum of industries.



IMAGING
Single-lens IEEE-1394 (FireWire) and 1394b CCD and CMOS cameras.

STEREO VISION
Two- and three-camera IEEE-1394 (FireWire) stereo systems for 3D imaging.

SPHERICAL VISION
IEEE-1394 spherical imaging hardware and software.

Products • Sales • Support • News • Corporate • Contact

Recent News

8/15/2007
Point Grey Research Announces Remote Head Addition to the Dragonfly2 Family [more]

6/12/2007
Point Grey Research Launches the Grasshopper™ Series of IEEE-1394b Cameras [more]

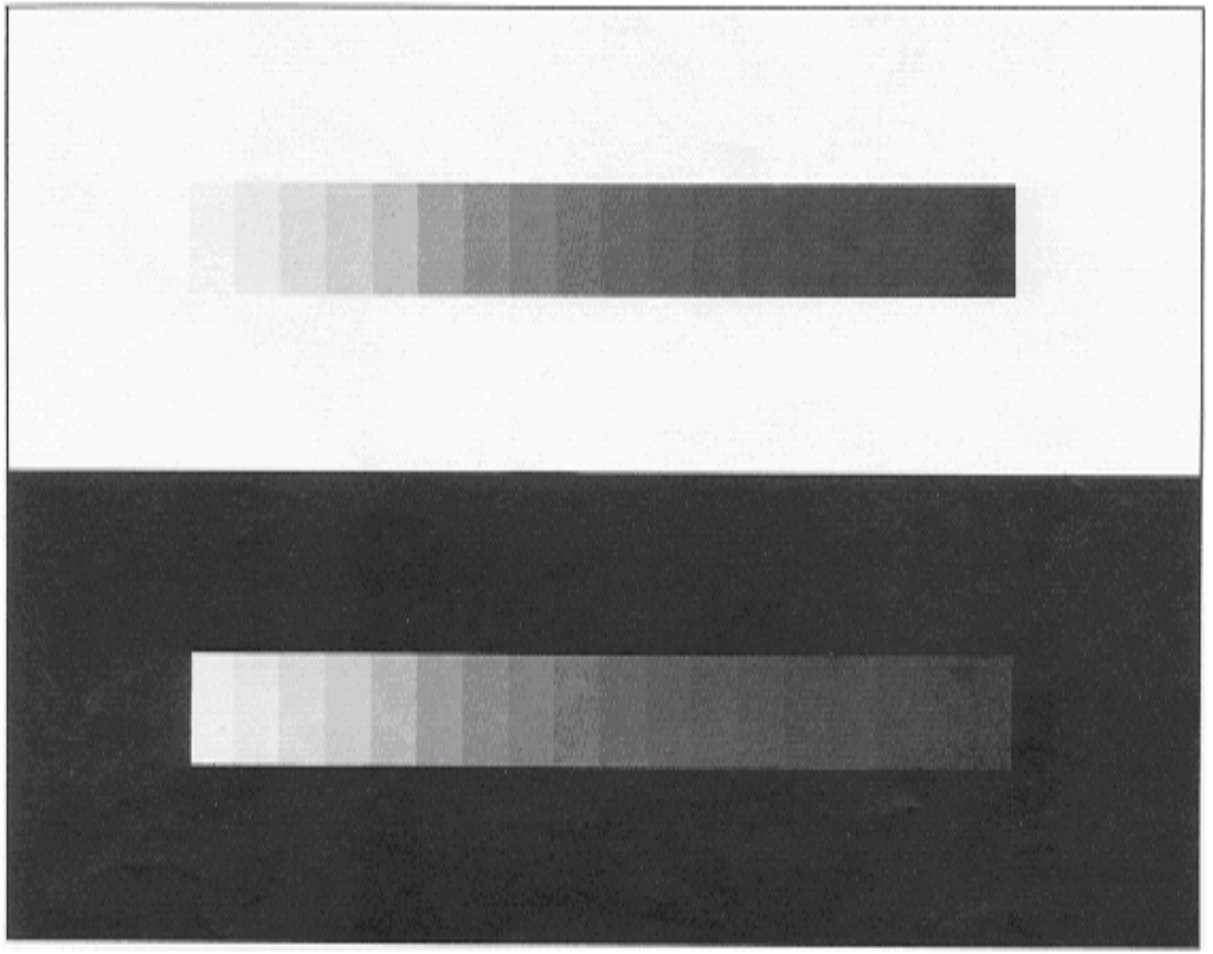
Insights Newsletter

Stay updated with the latest Point Grey news by subscribing to our Insights mailing list.

www.ptgrey.com

10 Years of Innovation in Imaging

Universal Perception ?

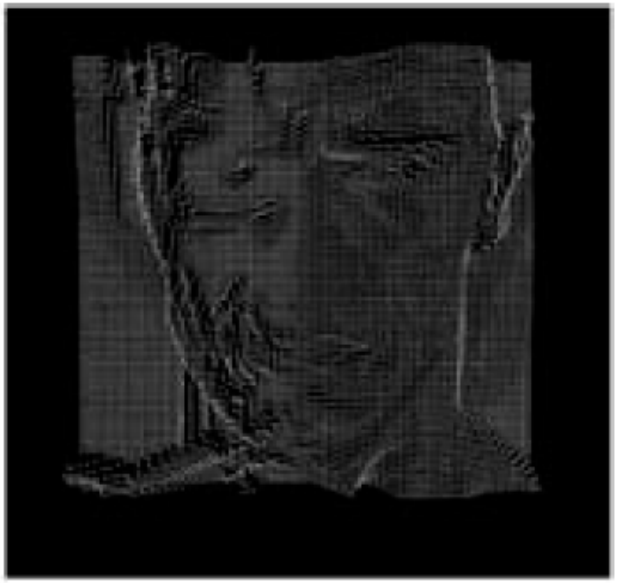


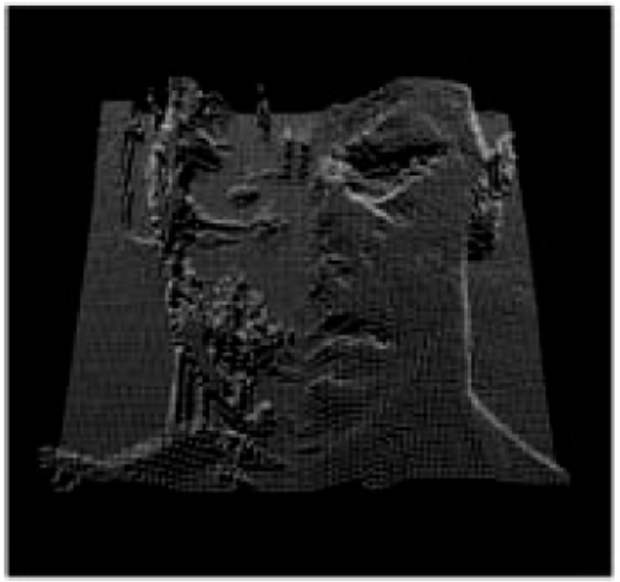
La perception des différentes gradations d'une échelle de gris dépend du niveau lumineux du fond.

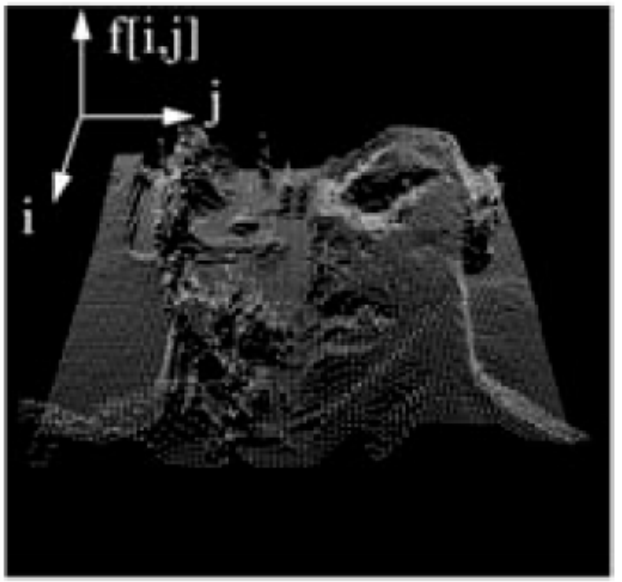

http://assimilateur_chromatique/poney.com

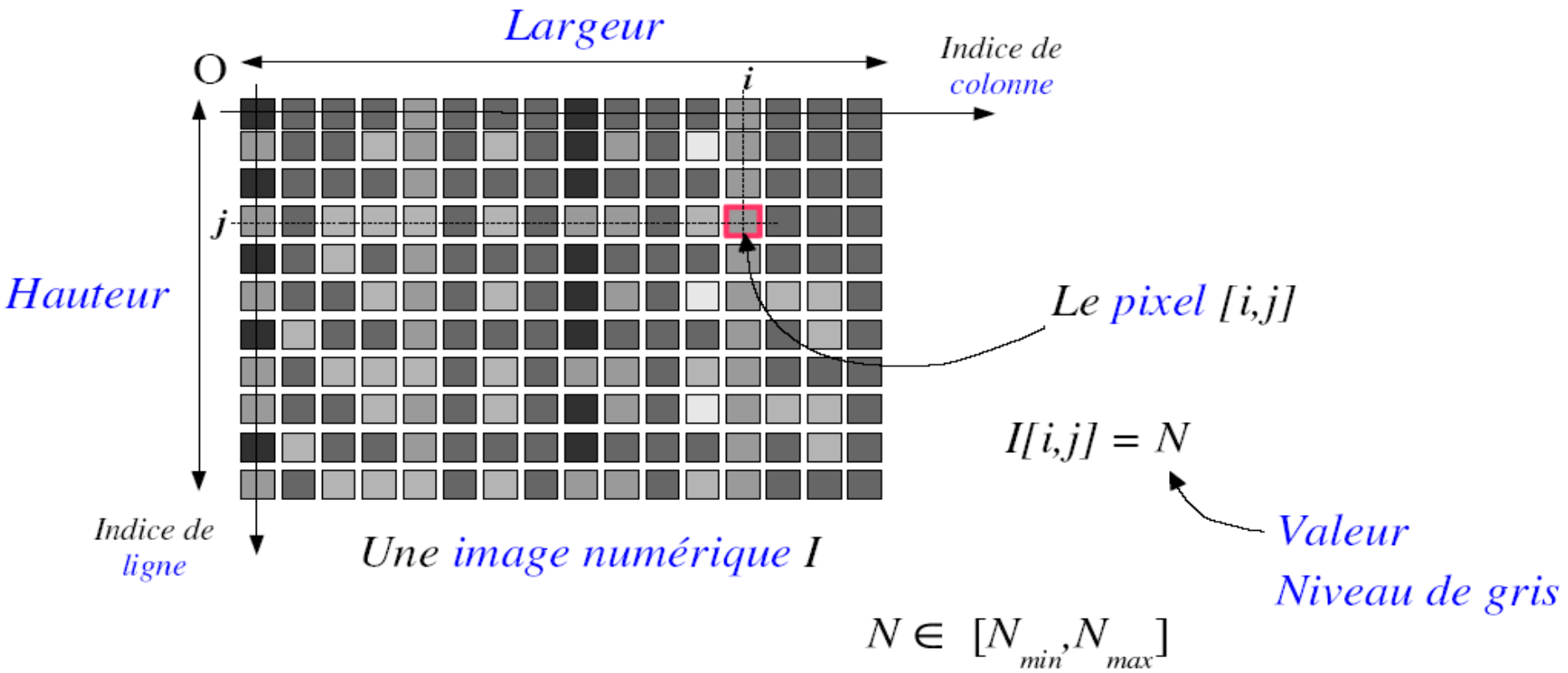
Handling a digital image ? As a discrete matrix

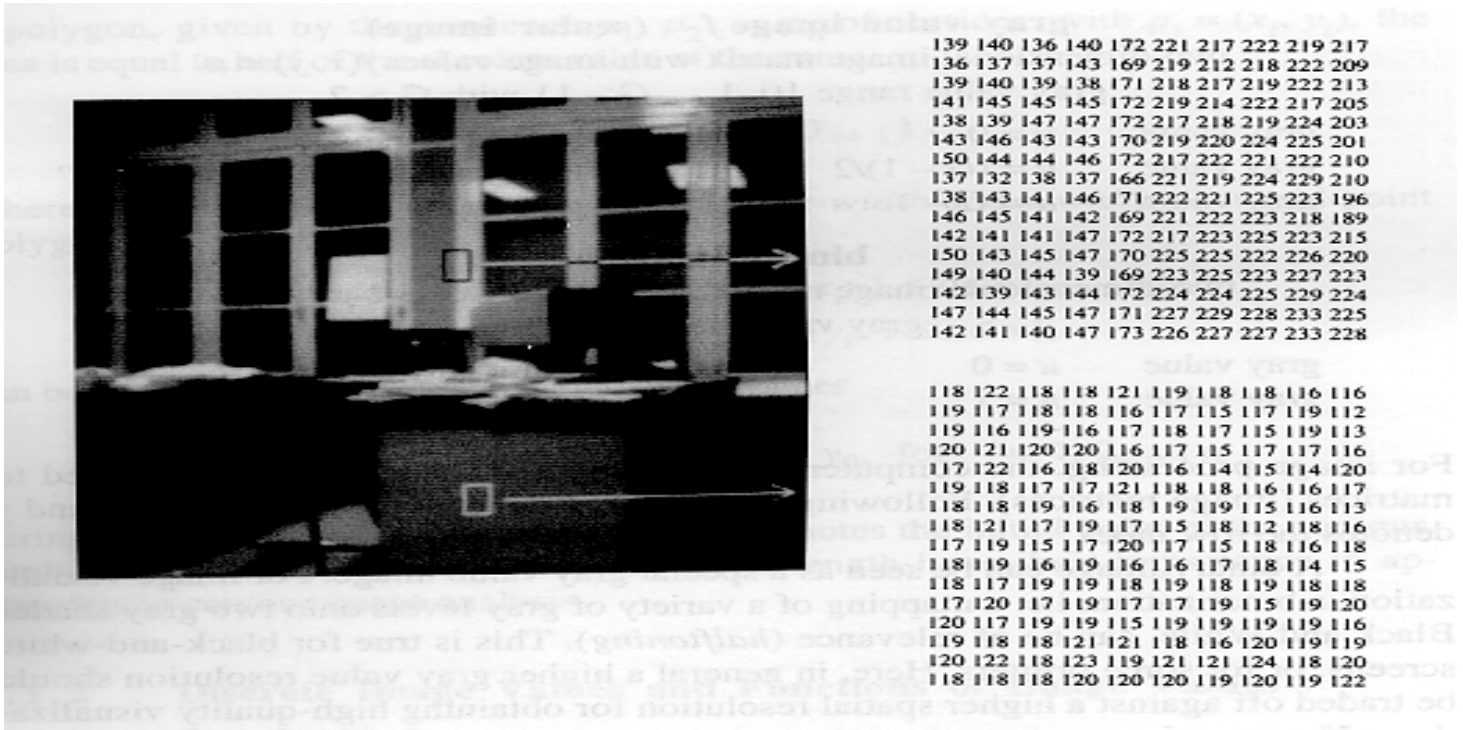
$f : [a,b]x[c,d]->[0,1]$








Largeur
O
Indice de
colonne
i
j
Hauteur
Le pixel [i,j]
I[i,j] = N
Valeur
Niveau de gris
Indice de
ligne
Une image numérique I
$N \in [N_{min}, N_{max}]$


139 140 136 140 172 221 217 222 219 217
136 137 137 143 169 219 212 218 222 209
139 140 139 138 171 218 217 219 222 213
141 145 145 145 172 219 214 222 217 205
138 139 147 147 172 217 218 219 224 203
143 146 143 143 170 219 220 224 225 201
150 144 144 146 172 217 222 221 222 210
137 132 138 137 166 221 219 224 229 210
138 143 141 146 171 222 221 225 223 196
146 145 141 142 169 221 222 223 218 189
142 141 141 147 172 217 223 225 223 215
150 143 145 147 170 225 225 222 226 220
149 140 144 139 169 223 225 223 227 223
142 139 143 144 172 224 224 225 229 224
147 144 145 147 171 227 229 228 233 225
142 141 140 147 173 226 227 227 233 228
118 122 118 118 121 119 118 118 116 116
119 117 118 118 116 117 115 117 119 112
119 116 119 116 117 118 117 115 119 113
120 121 120 120 116 117 115 117 117 116
117 122 116 118 120 116 114 115 114 120
119 118 117 117 121 118 118 116 116 117
118 118 119 116 118 119 119 115 116 113
118 121 117 119 117 115 118 112 118 116
117 119 115 117 120 117 115 118 116 118
118 119 116 119 116 116 117 118 114 115
118 117 119 119 118 119 118 119 118 118
117 120 117 119 117 117 119 115 119 120
120 117 119 119 115 119 119 119 119 116
119 118 118 121 121 118 116 120 119 119
120 122 118 123 119 121 121 124 118 120
118 118 118 120 120 120 119 120 119 122

Format : very portable : *pgm* or *ppm* or *pbm*

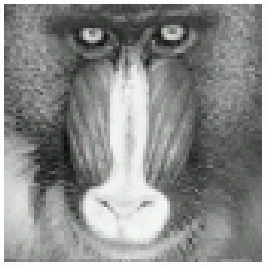
```
P2
#feep.pgm
10  5
4
0 0 0 1 1 4 4 3 0 0
0 1 4 1 1 4 4 3 0 0
0 0 3 1 1 4 4 3 0 0
0 0 0 1 1 3 3 3 0 0
0 0 0 1 1 2 2 2 2 2
```

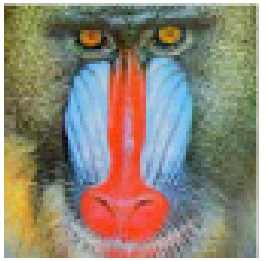
```
P3
# example.ppm
4  4
15
0 0 0   0 0 0   0 0 0   15 0
15
0 0 0   0 15 7   0 0 0   0 0 0
0 0 0   0 0 0   0 15 7   0 0 0
15 0 15   0 0 0   0 0 0   0 0
0
```

Free software for manipulations : *gimp and imageJ*

La jungle des formats d'image

- bitmap ⇒ **image de niveaux de gris ou couleurs de type photo (Exemple: GIF, PCX, BMP, JPEG, PGM…)**





- Binaire: $I(x,y) \in \{0,1\}$
- Niveau de gris: $I(x,y) \in [0,255]$
- Couleur: $I_R(x,y)\ I_V(x,y)\ I_B(x,y)$

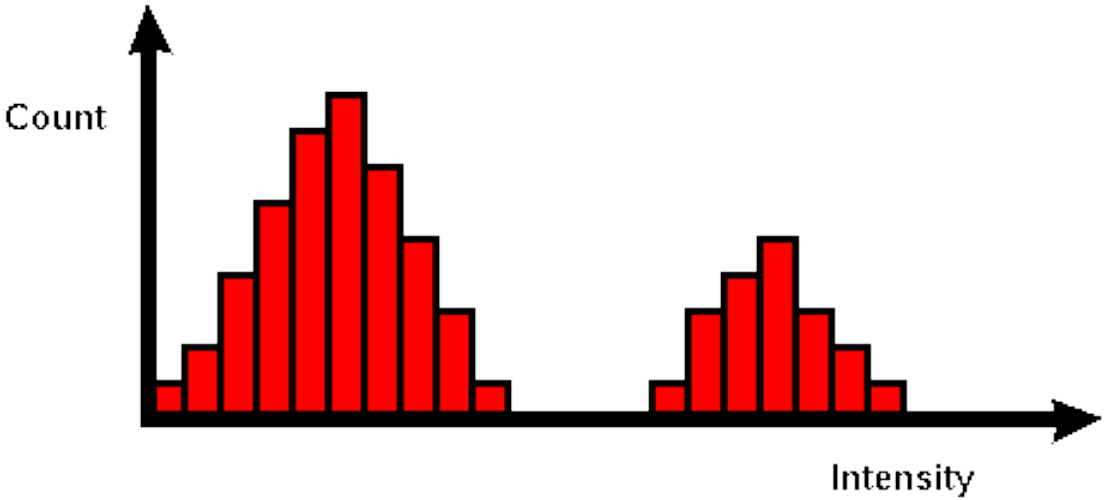
#ligne:256,512,480,600,768,1024
#colonne:256,512,640,800,1024,1280
#NdeG:2,64,256,1024,4096,16384

2^k (architecture, FFT…)

- Vectoriel : SVG etc.

Global representation of an image : histogram

Statistical distribution of grey levels : histogram spaces

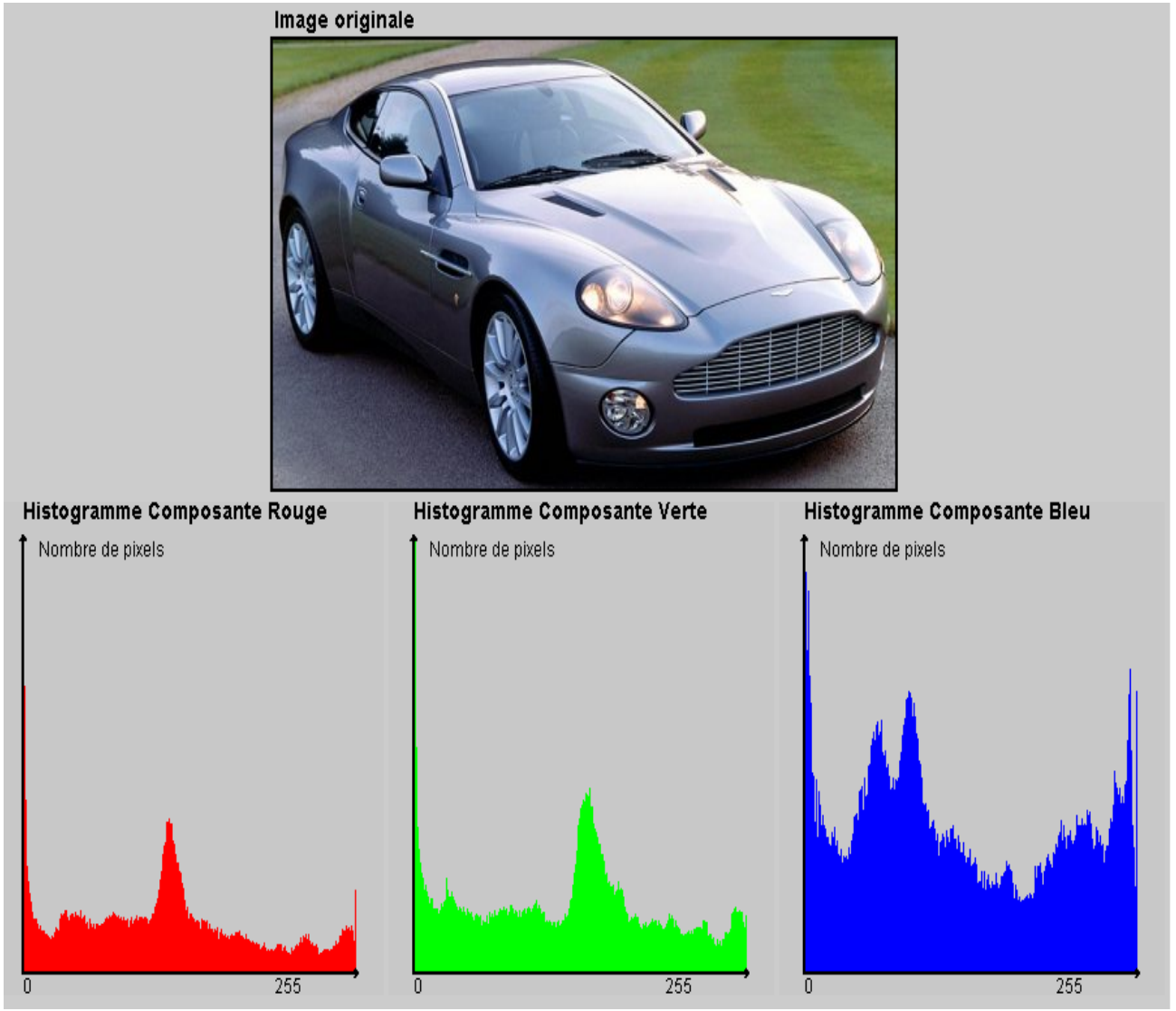


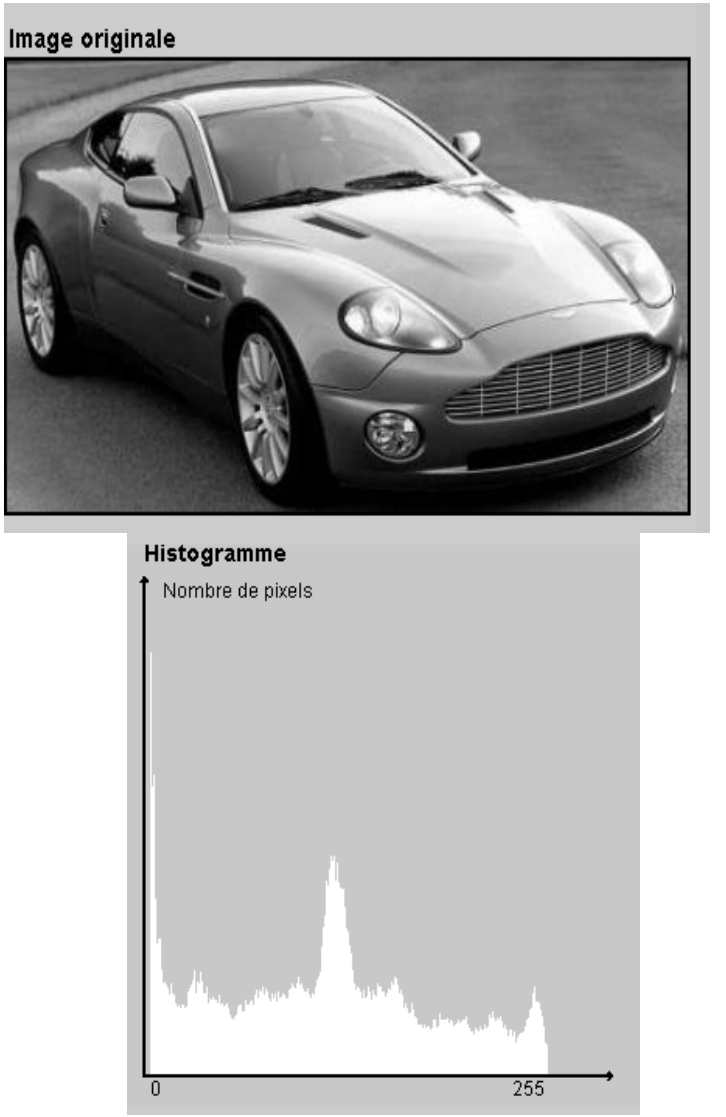

```
def contraste(A,f):
    n,m=np.shape(A)
    B=[[0 for j in range(m)] for i in
range(n)]
    for in in range(n):
        for j in range(m):
            B[i][j]=f(A[i][j])
    return B
```

```
nom='mandrill.jpg'
im = Image.open(nom).convert("L")
A=np.array(im)
plt.figure("Original")
plt.imshow(A,cmap=cm.gray)
plt.axis('off')
plt.savefig("Original",dpi=200)
plt.show()
plt.figure(u"Contraste1")
f=lambda x:np.sqrt(x)
B=contraste(A,f)
plt.imshow(B,cmap=cm.gray)
plt.axis('off')
plt.savefig('Constraste sqrt',dpi=200)
```

And a color image : f(X,Y)-> (R,G,B) : Vectorial functions analysis

Color Histogram

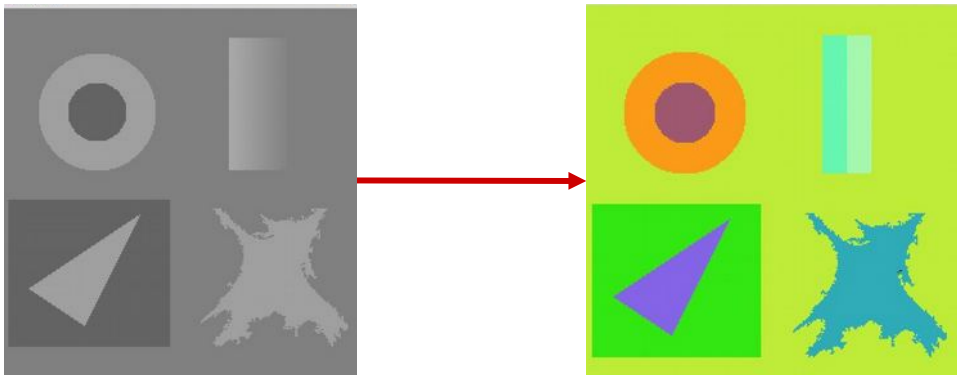




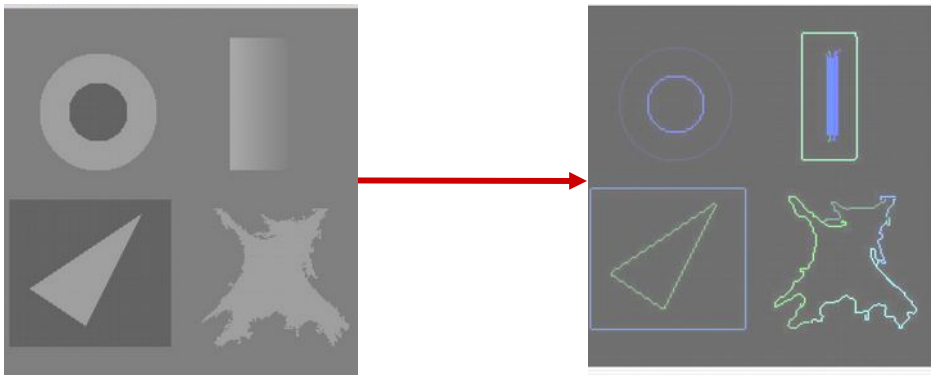
From Image processing to image analysis

We can SEGMENT (spatial clustering) an image with two approaches :

REGIONS or TEXTURES



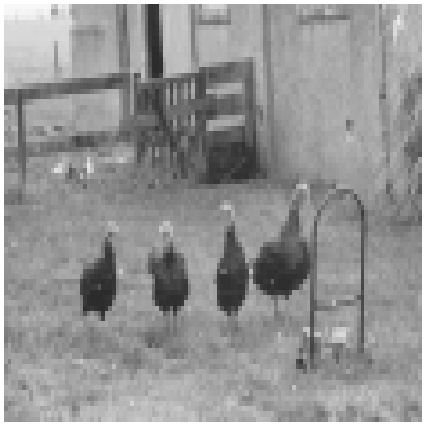
CONTOURS

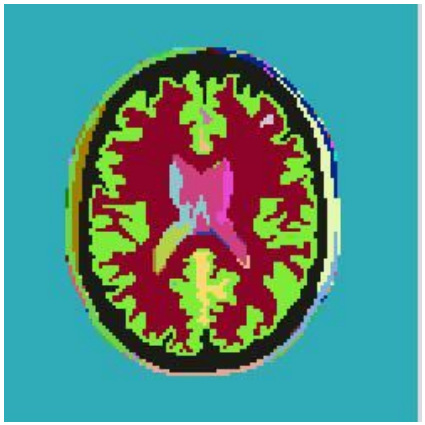


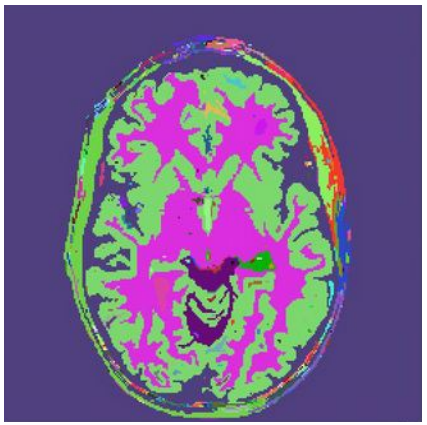
•An image should be pre-processed

```
def filtre_median(A,taille):
    n,m=np.shape(A)
    ic = int(taille/2.)
    B=[[0 for j in range(m)] for i in range(n)]
    for i in range(ic,n-ic):
        for j in range(ic,m-ic):
            liste = []
            for ik in range(-ic,ic+1):
                for jk in range(-ic,ic+1):
                    liste.append(A[i+ik][j+jk])
            B[i][j]=np.median(liste)
    return B
```

Regions







Segmentation en régions globale

Le Seuillage (*threshold*)

- Le seuillage est une une technique de segmentation simple, non contextuelle et efficace :
  - Seuillage d'intensité
  - Classification des pixels (voxels) en deux catégories
  - Création d'une image binaire (binarisation)

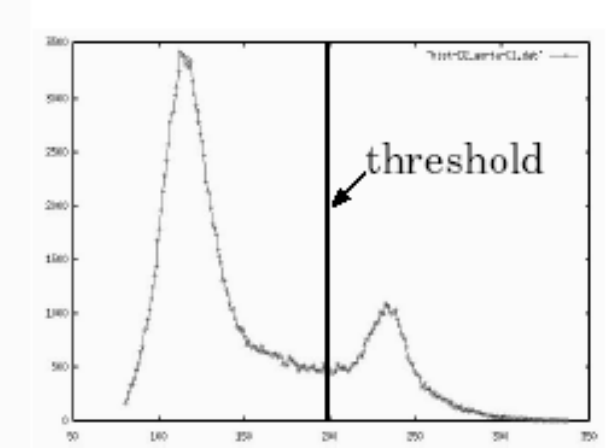
- Le seuillage peut utiliser un seuil soit ***fixé*** soit ***adaptatif***

- Diverses techniques ont été imaginées pour définir automatiquement ce seuil mais aucune n'est complètement robuste →

Seuil et Histogramme

Seuiller implique généralement l'analyse de l'histogramme :

- Différentes caractéristiques image donne naissance à des modes dans un histogramme (bimodale)
- En général les pics (modes) de l'histogramme correspondant à des primitives image différentes vont se chevaucher



Un exemple de valeur de seuil est la moyenne des niveaux de gris

Seuillage fixe

- Seuil *T fixé* ou *global*: constant sur toute l'image

$$g(x,y) = \begin{cases} 0 & f(x,y) < T \\ 1 & f(x,y) \geq T \end{cases}$$

Normal Threshold

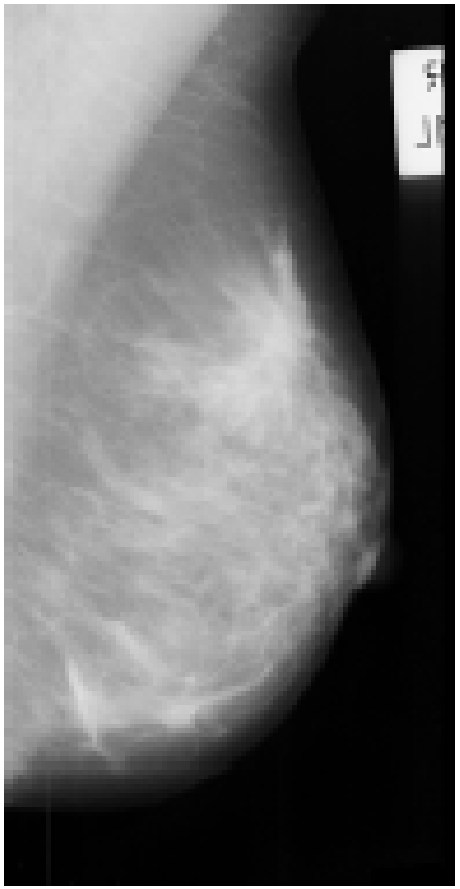
$$g(x,y) = \begin{cases} 1 & f(x,y) \leq T \\ 0 & f(x,y) > T \end{cases}$$

Reverse Threshold

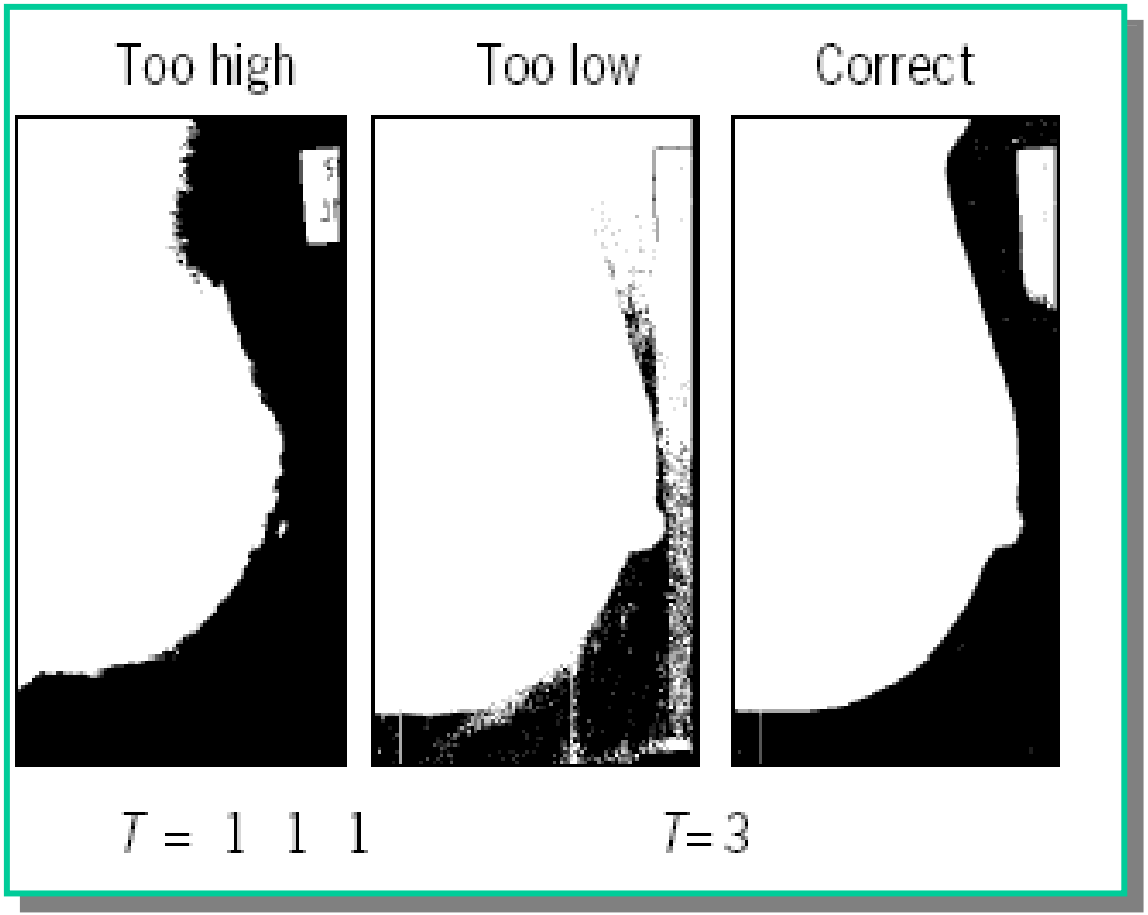
$$g(x,y) = \begin{cases} 0 & f(x,y) < T_1 \\ 1 & T_1 \leq f(x,y) \leq T_2 \\ 0 & f(x,y) > T_2 \end{cases}$$

Deux seuils : intervalle d'intensités

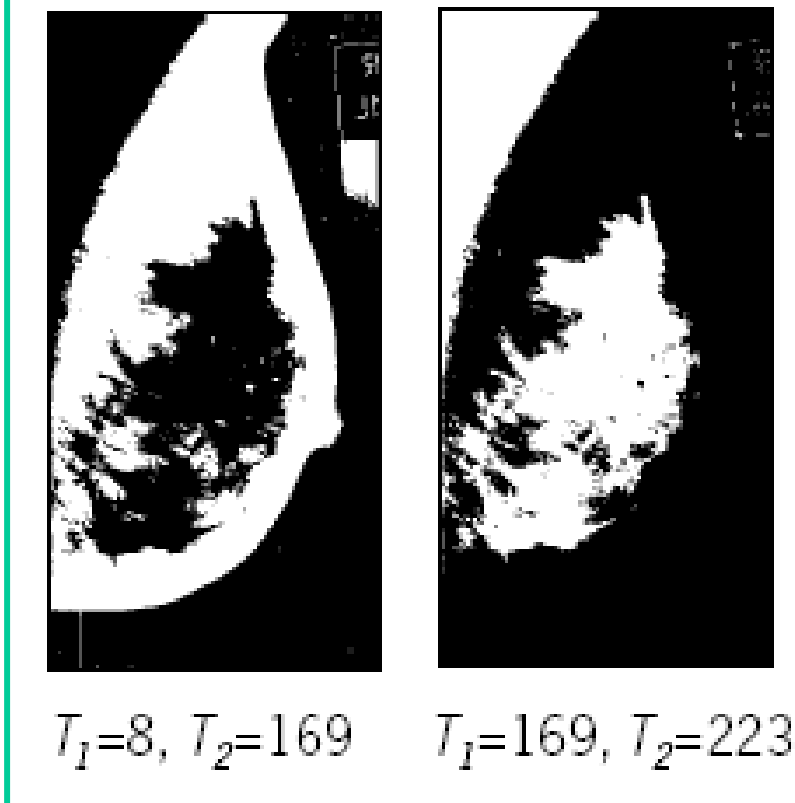
Seuillage fixe



Régions

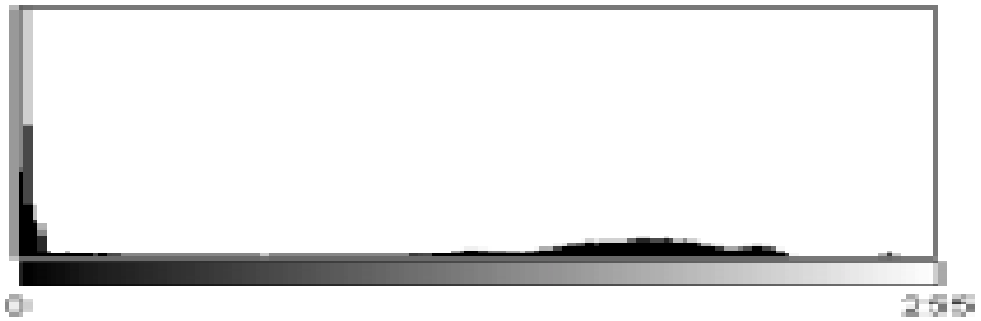


comment choisir ?



Sélection d'un Seuil Optimal ?

- Utilité de la forme de l'histogramme quand les pics ne sont pas bien définis ?



- Seuillage optimal : on mesure la séparation (statistique) entre deux régions à partir d'une fonction critère, et le seuil qui optimise ce critère est conservé : ISODATA

ISODATA Algorithm :
Algo générique : Iterative Self Organizing DATA
A rapprocher des algorithmes de classification des C-Moyennes ou d'estimation statistiques type EM (Expectation - Maximization)

http://io9.com/the-10-algorithms-that-dominate-our-world-1580110464
http://www.fukakai.fr/top-10-des-algorithmes-du-20eme-siecle/

Iterative threshold selection technique (pseudo-code)

- Select an initial threshold T_0 (*e.g.* the mean intensity)
- Partition the image into two groups (R_1 and R_2) using T_0
- Calculate the mean intensity values μ_1 and μ_2 of the partitions R_1 and R_2.
- Select a new threshold: $T_i = (\mu_1 + \mu_2)/2$
- Repeat steps 2-4 until: $T_i = T_{i-1}$

Exercice : faire tourner l'algo sur un exemple 1D avec les valeurs : 23, 34, 42, 57, 123, 123, 252

La méthode d'Otsu

- La méthode de seuillage d'Otsu repose sur la sélection d'un minimum *entre* deux *modes* (pics).
- Fréquence et Valeur Moyenne :
  - Fréquence: $\omega = \sum_{i=0}^{T} P(i) \qquad P(i) = n_i / N$
  - Moyenne : $\mu = \sum_{i=0}^{T} iP(i) / \omega$

N: Nombre total de pixels

n_i: nombre de pixels dans le niveau *i*

- Analyse de la variance (variance= déviation standard²)
  - Variance totale :

$$\partial_t^2 = \sum_{i=0}^{T} (i-\mu)^2 P(i)$$

La méthode d'Otsu

- variance inter-classes (δ_b^2):

$$\delta_b^2 = \omega_0 (\mu_0 - \mu_t)^2 + \omega_1 (\mu_1 - \mu_t)^2,$$

en utilisant $\mu_t = \omega_0 \mu_0 + \omega_1 \mu_1$, on obtient:

$$\delta_b^2 = \omega_0 \omega_1 (\mu_1 - \mu_0)^2$$

Avec $\omega_0, \omega_1, \mu_0, \mu_1$ respectivement fréquence et moyenne pour chaque classe

La méthode d'Otsu

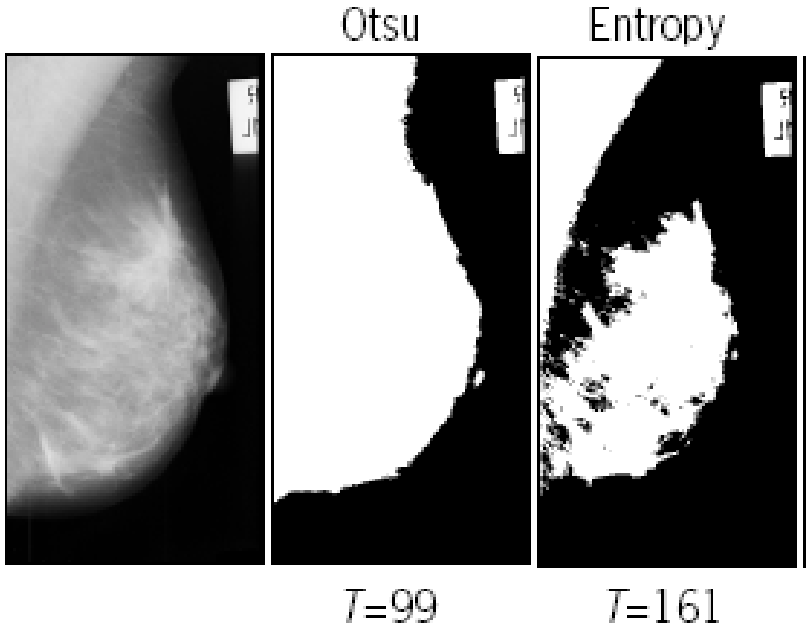
- La fonction critère utilise le rapport de la variance inter-classes rapportée à la variance totale:

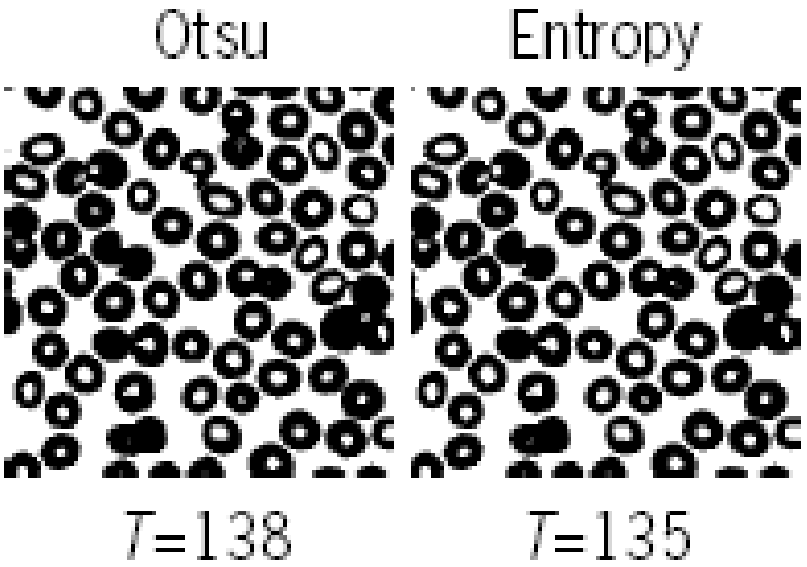
$$\eta = \delta_b^2 / \delta_t^2$$

- Tous les seuils possibles sont évalués et celui qui maximise η est choisi comme le seuil optimal

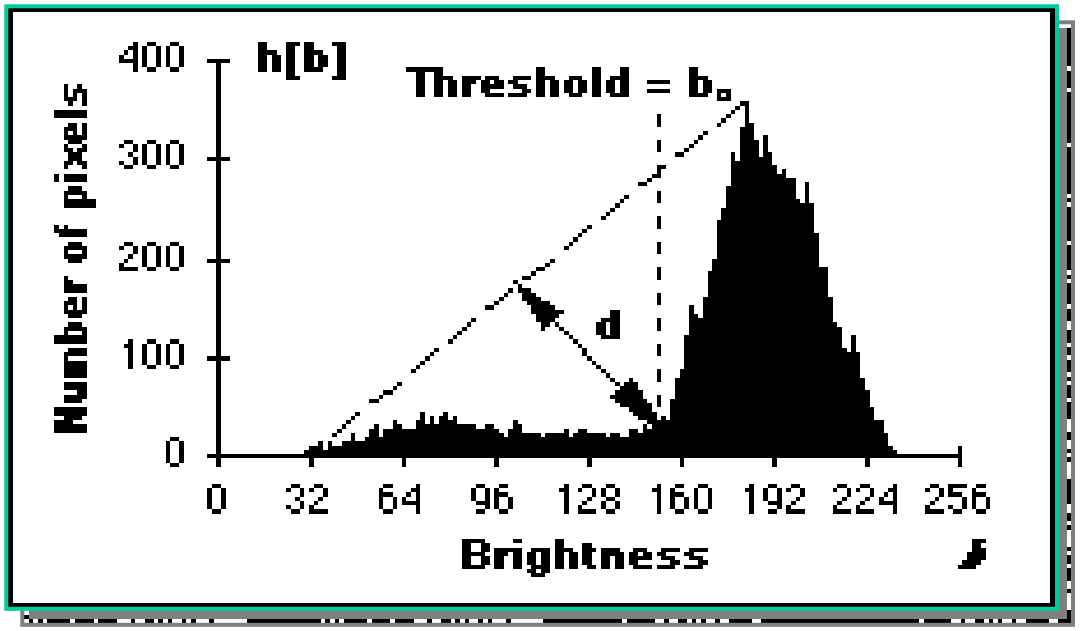
- Un pseudo-code ? C'est à vous...

Comparons les valeurs de seuil :



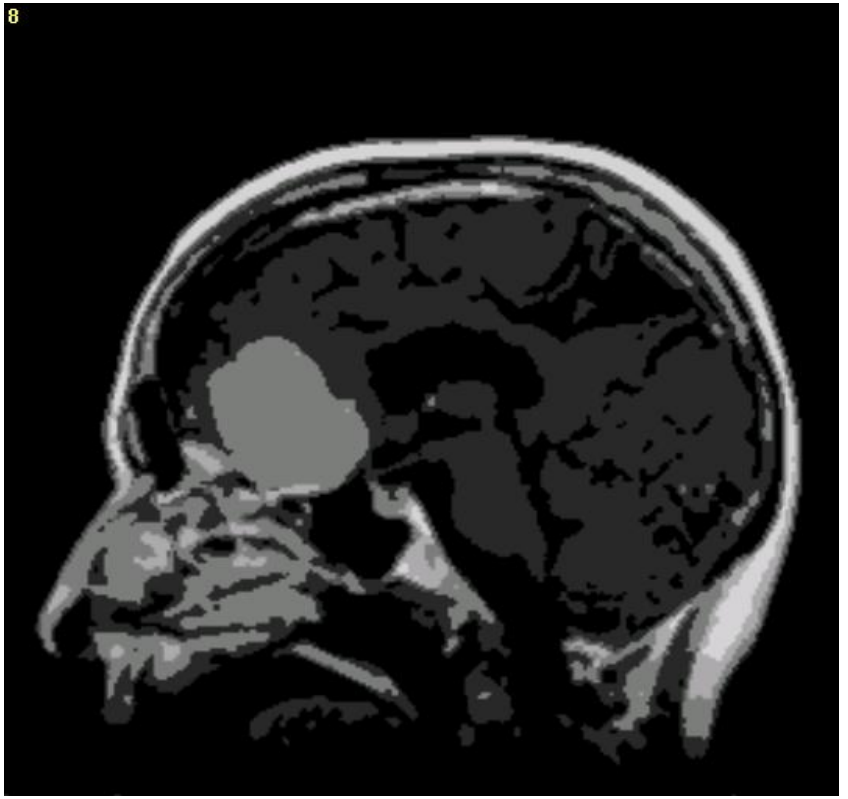


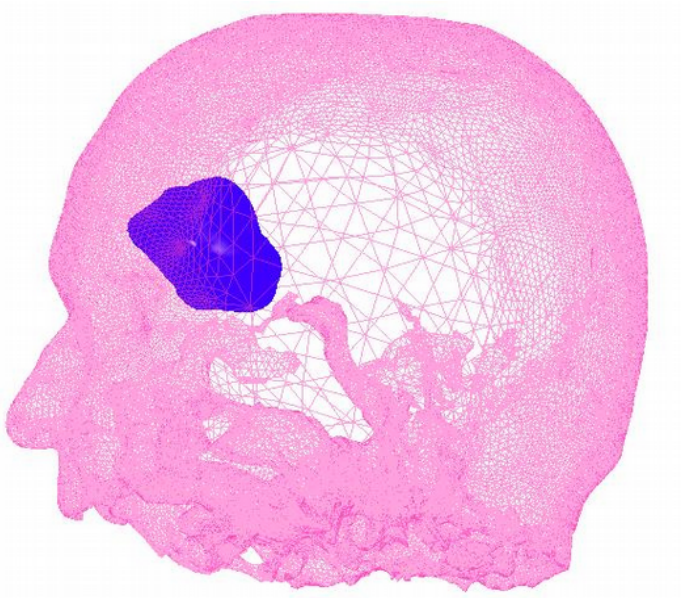
Plein d'autres algorithmes :

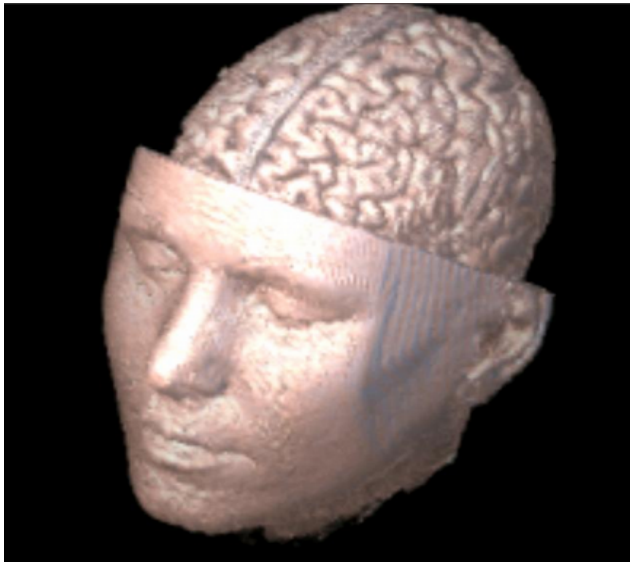


- Algorithme du Triangle :
  - La distance maximum ***d*** retourne “le” seuil optimal

http://www.ablesw.com/3d-doctor/3ddoctor.html







Propriétés des régions

- Périmètre et Surface :
  - Périmètre: La longueur du **contour** d'une **composante connexe** (région).
    - Calculé à partir du codage de chaine de la région;
    - Ou Estimé par le nombre de pixels du contour.
  - Surface : Le nombre de pixels (carré unitaires) contenus dans la région/forme P
    - Formule de Pick : $S(P) = n_I + n_B/2-1$

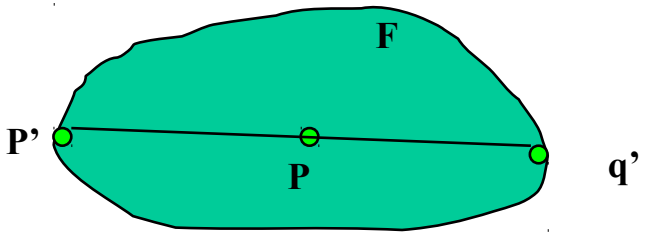
      n_I , n_B :nombre de points intérieurs ou de points reposant sur la frontière, respectivement

Propriétés des régions

- Centre, Rayon et Diamètre :
  - Excentricité d'un point *P* dans *F* est le maximum des distances *d(p,q)* pour tous les points *q∈F:*

$$exc(p) = max\ d(p,q) \mid q \in F^*$$

  - **Centre** : L'ensemble des points *P* de moindre *excentricité*
  - **Rayon** : La valeur de la moindre *excentricité d(p,p')*
  - **Diamètre** : La valeur de la plus grande *excentricité d(p',q')*



Propriétés des régions

- Centroïdes, Moments et Orientation:
  - Centroïde: Soit *F*, un ensemble de *n* pixels connectés (x_i,y_i),on peut définir un centroïde *c* comme :

$$x_c = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad y_c = \frac{1}{n}\sum_{i=1}^{n} y_i$$

  - Moments: le moment discret central d'ordre *(k,l)est défini par* :

$$\mu_{k,l} = \sum_{i=1}^{n} (x_i - x_c)^k (y_i - y_c)^l$$

  - Orientation: l'orientation est définie ici par un angle θ :

$$\theta = \frac{1}{2}\arctan\left(\frac{2\mu_{1,1}}{\mu_{2,0} - \mu_{0,2}}\right)$$

Threshold.java

Below is the syntax highlighted version of Threshold.java from §3.1 Using Data Types.

```
/*************************************************************************
 *  Compilation:  javac Threshold.java
 *  Execution:    java Threshold frame00001.jpg
 *  Dependencies: Picture.java Luminance.java
 *
 *  Reads in a JPEG/GIF/PNG file, displays it on the screen,
 *  converts all pixels to grayscale, and displays those
 *  pixels with a grayscale value >= 180.
 *
 *************************************************************************/

import java.awt.Color;

public class Threshold {

    public static void main(String[] args) {
        int THRESHOLD = 180;
        String filename = args[0];
        Picture pic = new Picture(filename);
        pic.show();
        for (int i = 0; i < pic.width(); i++) {
            for (int j = 0; j < pic.height(); j++) {
                Color color = pic.get(i, j);
                double lum = Luminance.lum(color);
                if (lum >= THRESHOLD) pic.set(i, j, Color.WHITE);
                else                  pic.set(i, j, Color.BLACK);
            }
        }
        pic.show();
    }
```

http://introcs.cs.princeton.edu/java/31datatype/Threshold.java.html
http://introcs.cs.princeton.edu/java/31datatype/Threshold.java
http://rsbweb.nih.gov/ij/plugins/download/Entropy_Threshold.java

Contours

Convolution : Etant donnés une image *I[i,j]* et un noyau *h[m,n]*

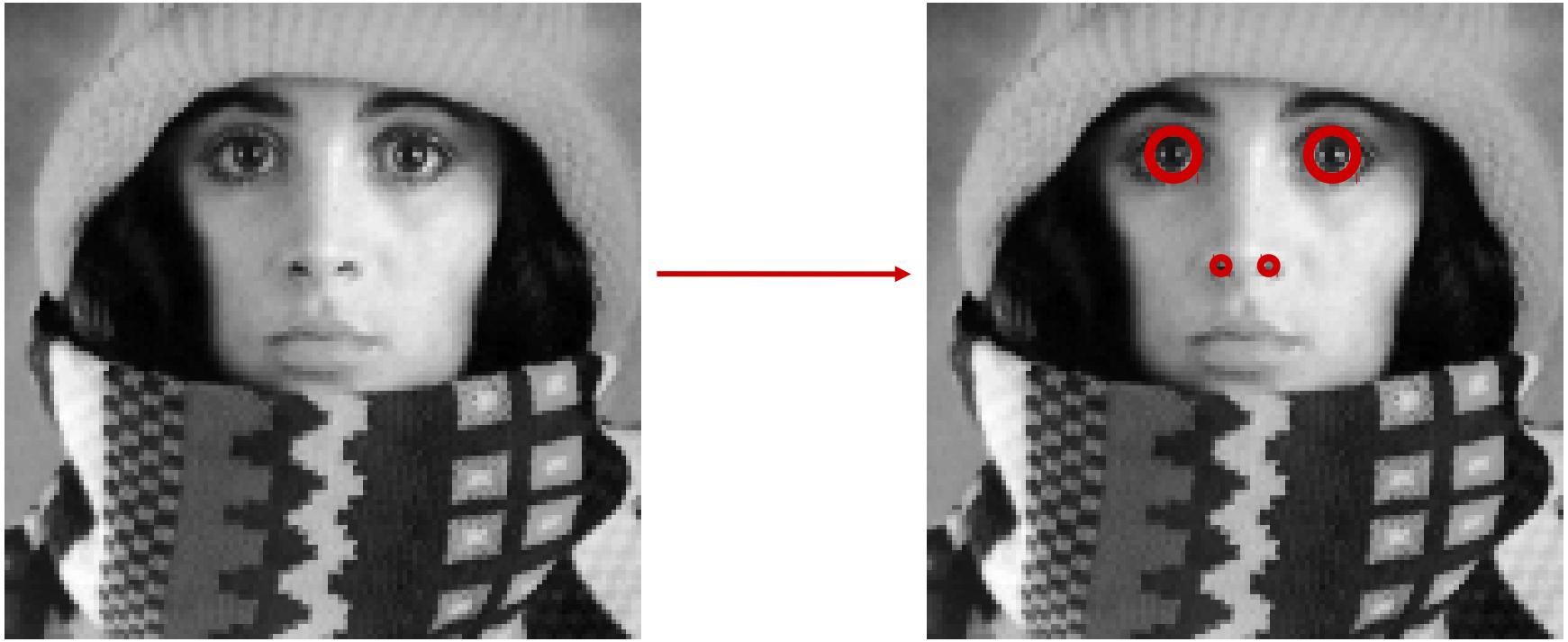
$$h * I[i,j] = \sum_{m=-M}^{M} \sum_{n=-N}^{N} I[i-m, j-n] h[m,n]$$

*

-1	0	1
-2	0	2
-1	0	1

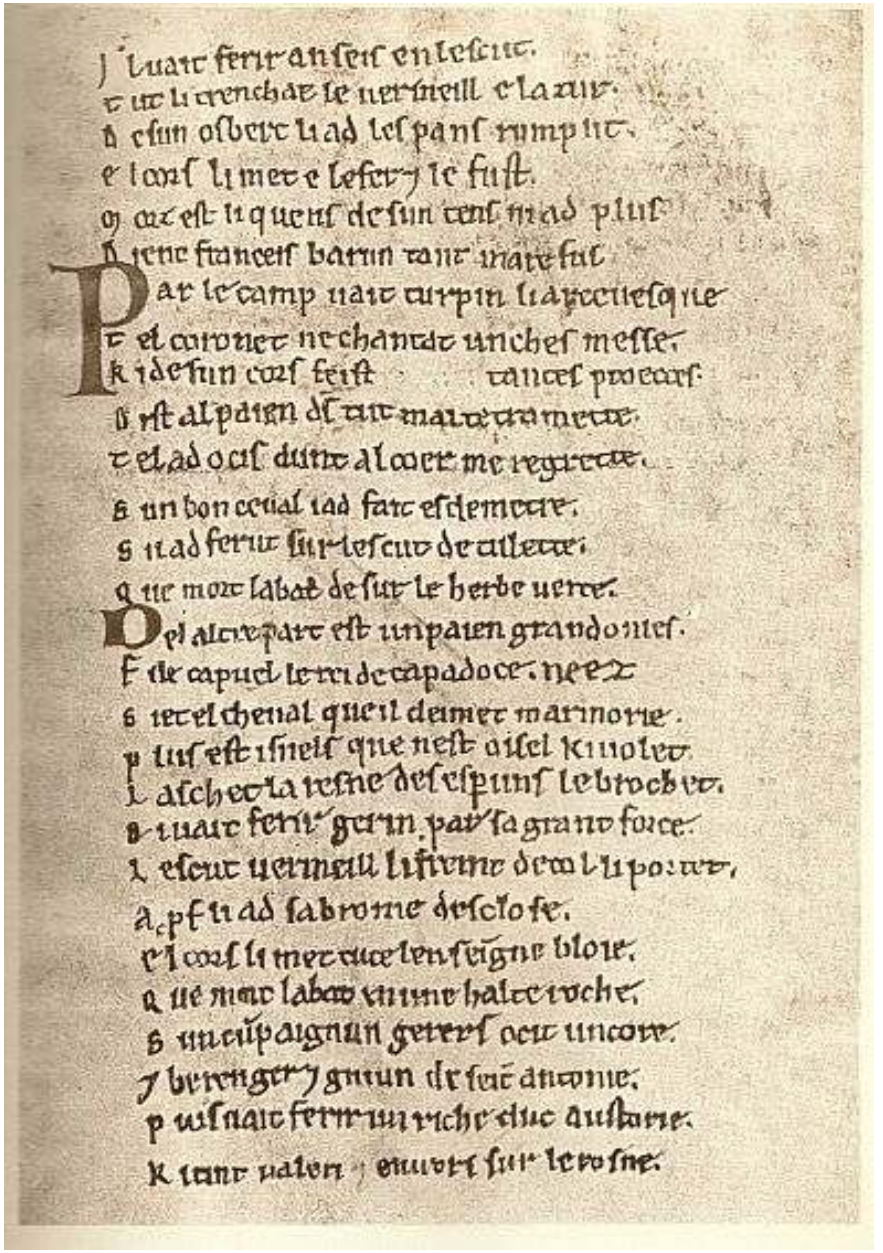
Sobel (1980)

Primitives évoluées

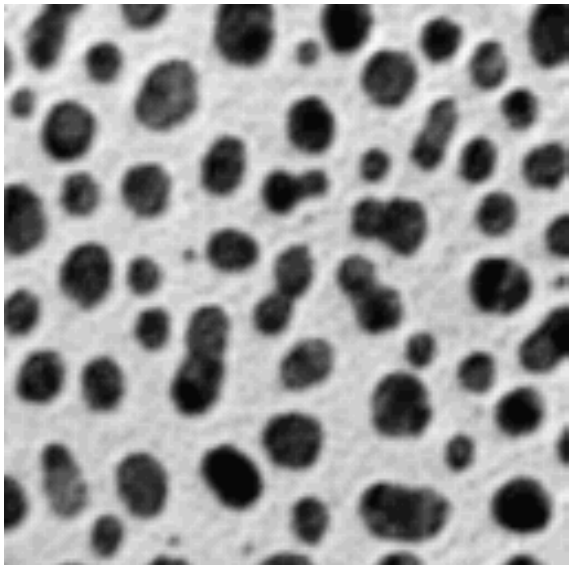


Une fois les traitements de bas-niveau effectués, on peut envisager des traitements de plus haut-niveau.
Nous allons essayer d'expliquer ce passage progressif du **traitement syntaxique** du signal vers un **traitement sémantique.**

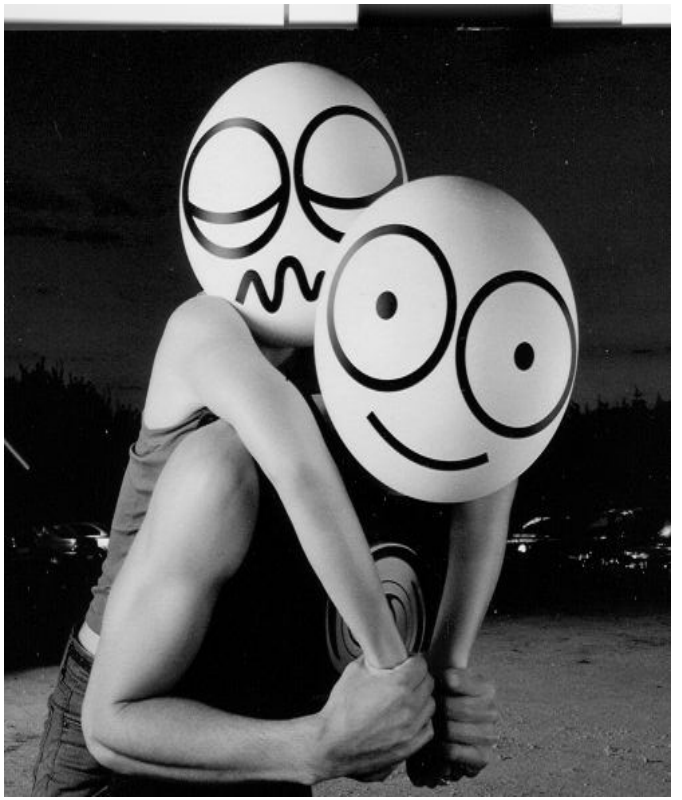
Prenons un manuscrit textuel à analyser. On sort momentanément de l'image à proprement parler. Pour en extraire le sens, on commence généralement par corriger les artéfacts du signal : corriger les bavures de l'encre (bruit d'acquisition), les fautes d'orthographes et de grammaires (bruit de transmission), traduire dans une langue normalisée (transformation de la distribution des lettres pour en améliorer la lisibilité par l'opérateur), puis progressivement, on va structurer le document en phrases, paragraphes, chapitres (segmentation en région et contour). Toutes ces opérations ont été plus ou moins décrites pour le cas spécifique d'une image et correspondent aux prétraitement syntaxique et structurel du signal pour le mettre en forme avant d'en anlyser le fond.



Une fois ce filtrage syntaxico-structurel effectué, on peut commencer à analyser le sens du document. C'est ce qu'on appelera , filtrer sémantiquement le signal. Par exemple, on voudra chercher tous les paragraphes qui parlent de « chômage ». Dans une image, c'est la même idée. On cherchera toutes les zones de l'image qui contiennent des formes circulaires ou plus sémantique encore, toutes les images satellites qui contiennent des cheminées d'usine (dont l'apparence est circulaire en général).



Un opérateur algorithmique qui réalise ce filtrage quasi- sémantique est l'opérateur de Hough par exemple (voir plugin imageJ).



La suite logique de ces opérateurs converge une autre discipline qu'on appelle la Reconnaisse de Formes.

Par exemple, comment reconnaître des visages dans une image, puis une classe de personnes, puis identifier une personne en particulier. Ces techniques sont des procédés souvent statistiques s'appuyant sur des mesures effectuées dans l'image (cf. biométrie, empreintes digitales).

Dans le cas de l'imagerie médicale par exemple, il s'agira de reconnaître automatiquement une mammographie présentant une tumeur d'une autre.

Une technique qui est à la limite du filtrage syntaxique (niveau pixel seul) et du filtrage sémantique (niveau forme ou ensemble de pixels) s'appelle la Morphologie Mathématique : dans ce cas **il ne s'agit plus de filtrage linéaire mais de filtrage d'ordre et cette distinction est conceptuellement fondamentale.**