MAICHANIA

université angers

HOB5: Phenomics

Methods of phenotyping based on imaging
with a focus on dedicated image processing technics with an open source software (Fiji).

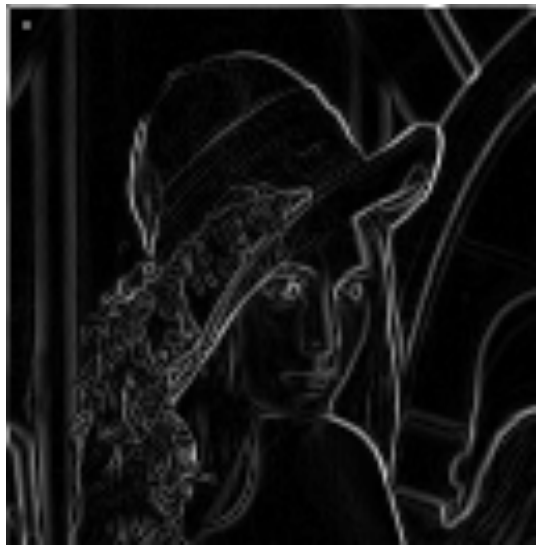
December 2018

Proposed by
Etienne BELIN (Ass. Prof @ university of Angers, France)
David ROUSSEAU (Prof @ university of Angers, France)

LARIS **Teams** IMHORPHEN **Platform** PHENOTIC SEMENCES & PLANTES IRHS Institut de Recherche en Horticulture et Semences INRA SCIENCE & IMPACT AGRO CAMPUS OUEST GEVES université angers

Part 3: image processing


→

→

→ *0/ 1*

3 steps in image processing to extract information

Sensor
Physical values

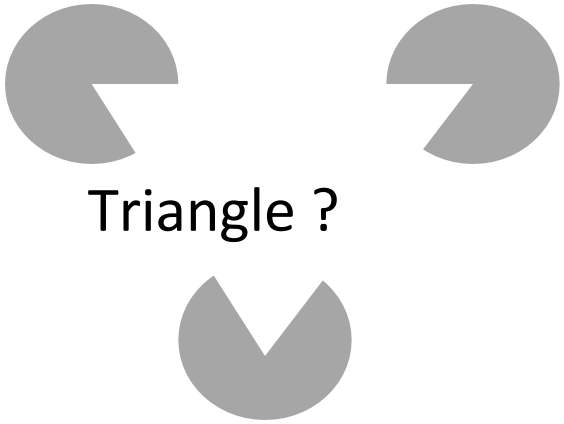
Interpretation
Measurements

Low level → Middle level → High level

- LL: enhancement, correction of the sensor.
- ML: morphometric features extraction
- HL: measurement on these features as estimation, detection, ….

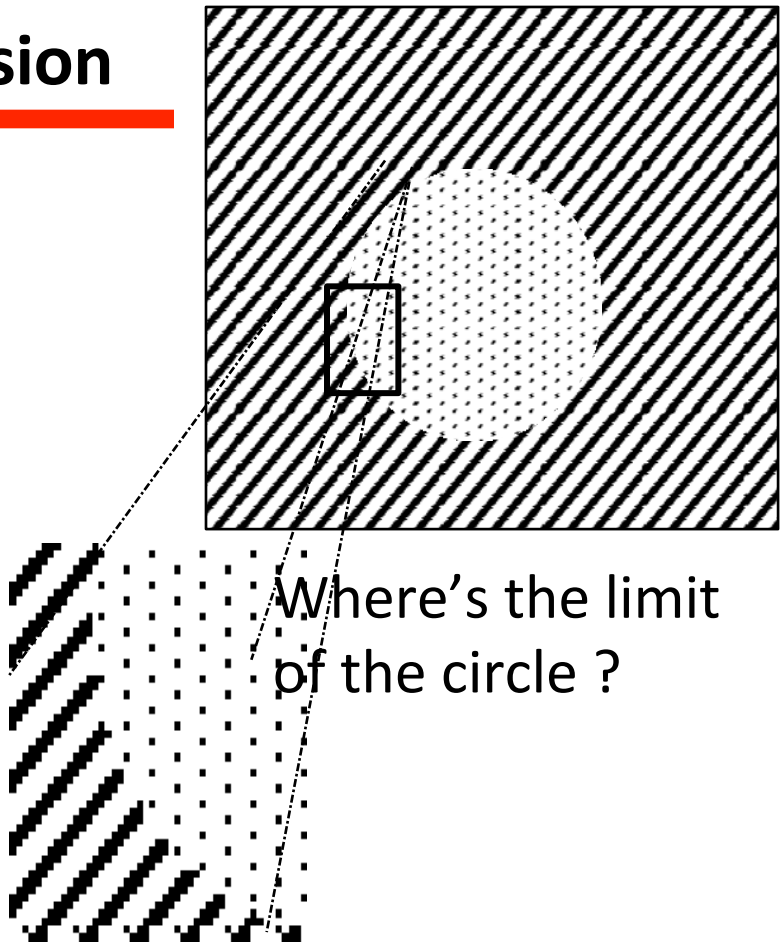
Data size decreases as semantic value increases

Computer vision VS human vision

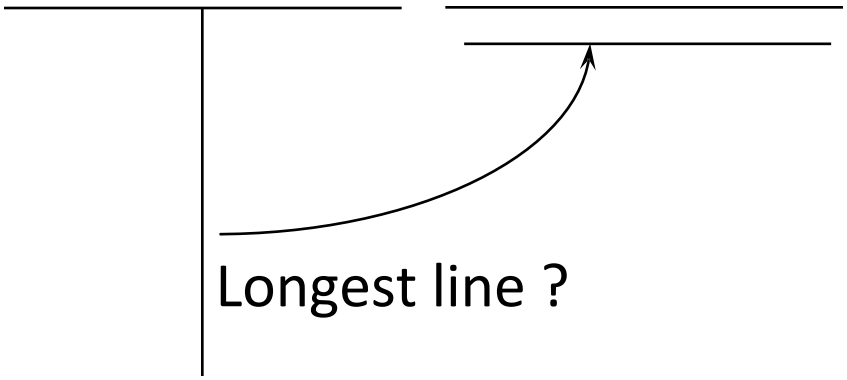


Triangle ?

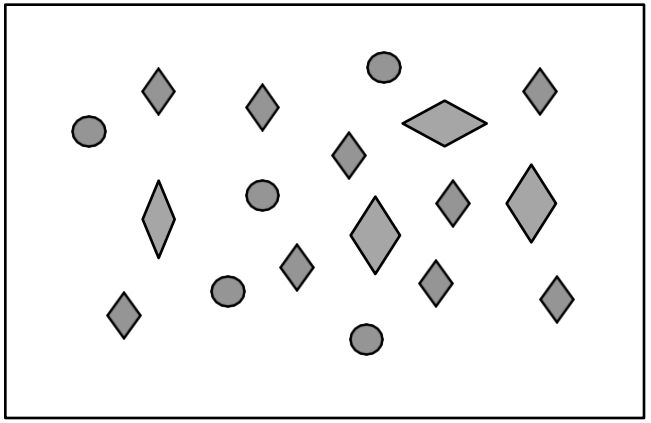
Brightest rectangle ?



Where's the limit of the circle ?

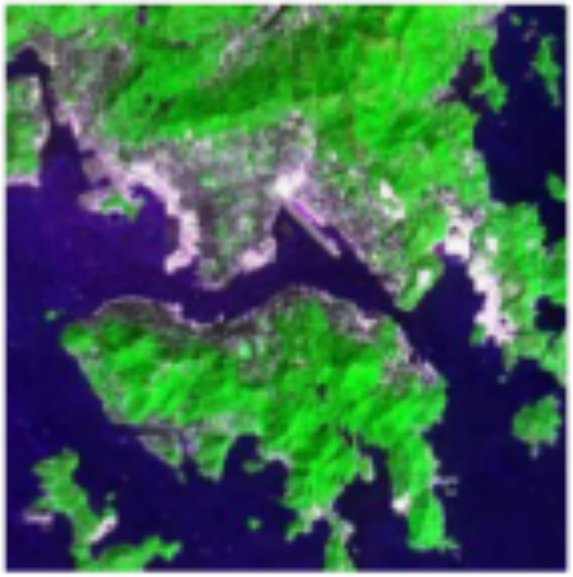


Longest line ?



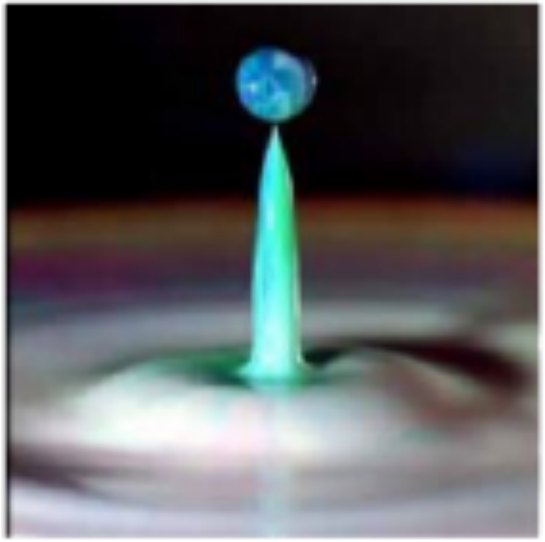
How many diamonds ?

Computer vision VS human vision



Scale & resolution

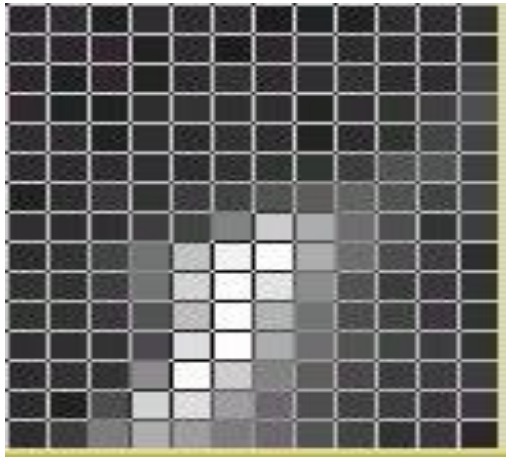
Wavelength domain



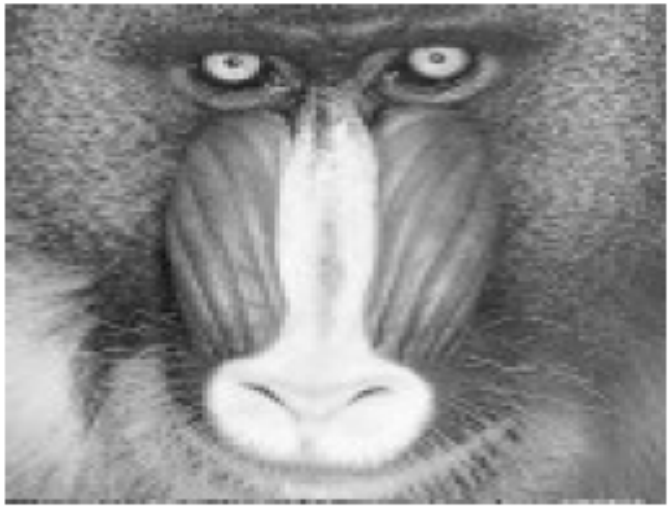
High-speed

Types of digital images

Digital image = Matrix = Tab



4	2	1	0
2	4	0	0
1	0	6	1
0	0	1	2

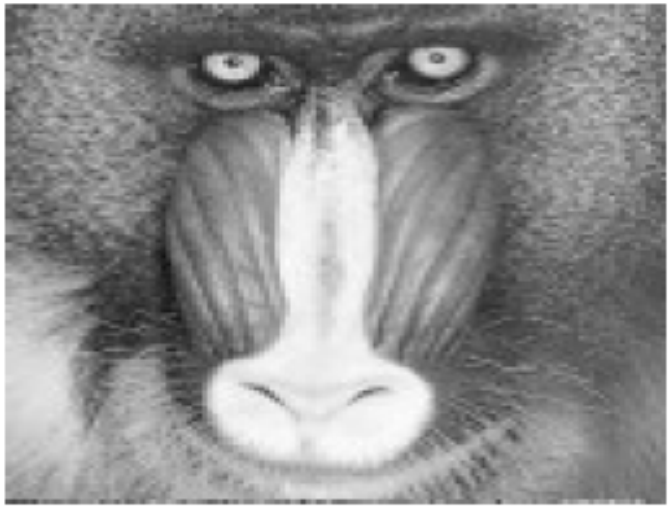


Intensity (grey-level)

Binary

Color (RGB)

Types of digital images



Intensity (grey-level)

$I(x,y) \in [0..255]$

Mainly coded on 2^8 levels

Binary

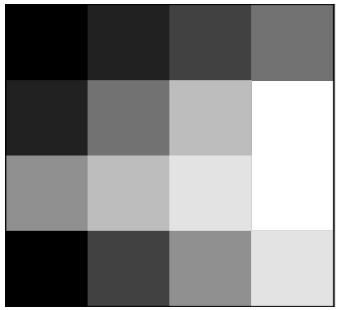
$I(x,y) \in \{0, 1\}$

Coded on 2^1 levels

Color (RGB)

$I_R(x,y)$ $I_G(x,y)$ $I_B(x,y)$

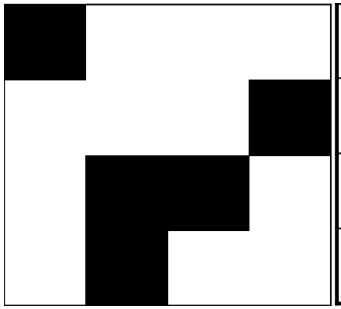
Coded on 3 x 2^8 levels



0	1	2	3
1	3	5	7
4	5	6	7
0	2	4	6

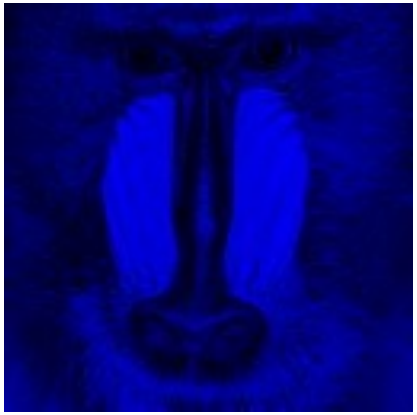
000	001	010	011
001	011	101	111
100	101	110	111
000	010	100	110

Example: grey-level coded on $2^3 = 8$ levels

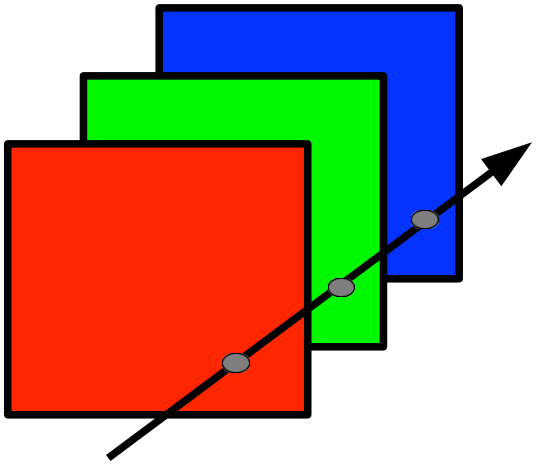


0	1	1	1
1	1	1	0
1	0	0	1
1	0	1	1

Talking about RGB image…

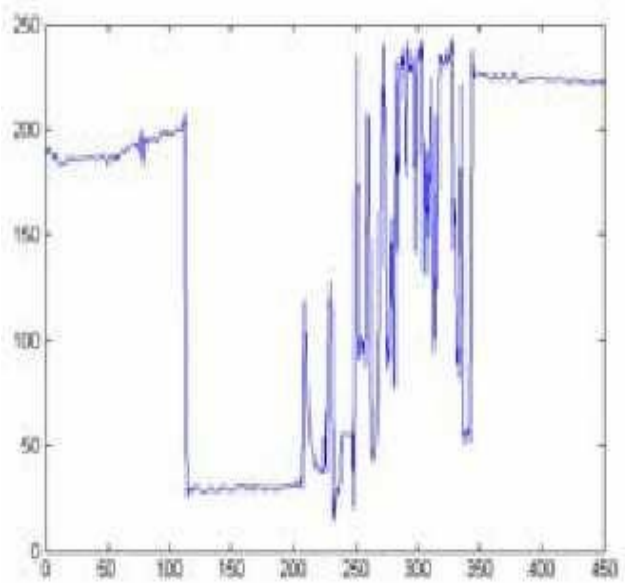


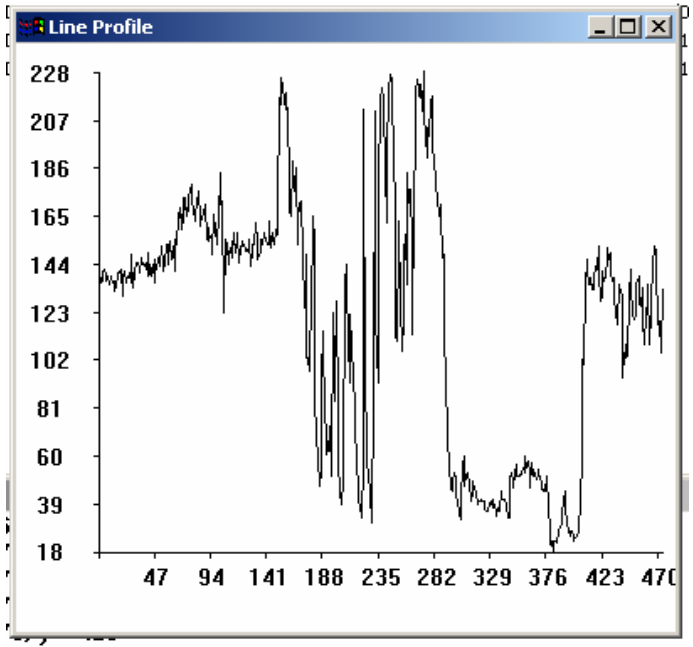
Intensity (grey-level) Intensity (grey-level) Intensity (grey-level)



Also existing other color spaces than RGB …

Intensity profile

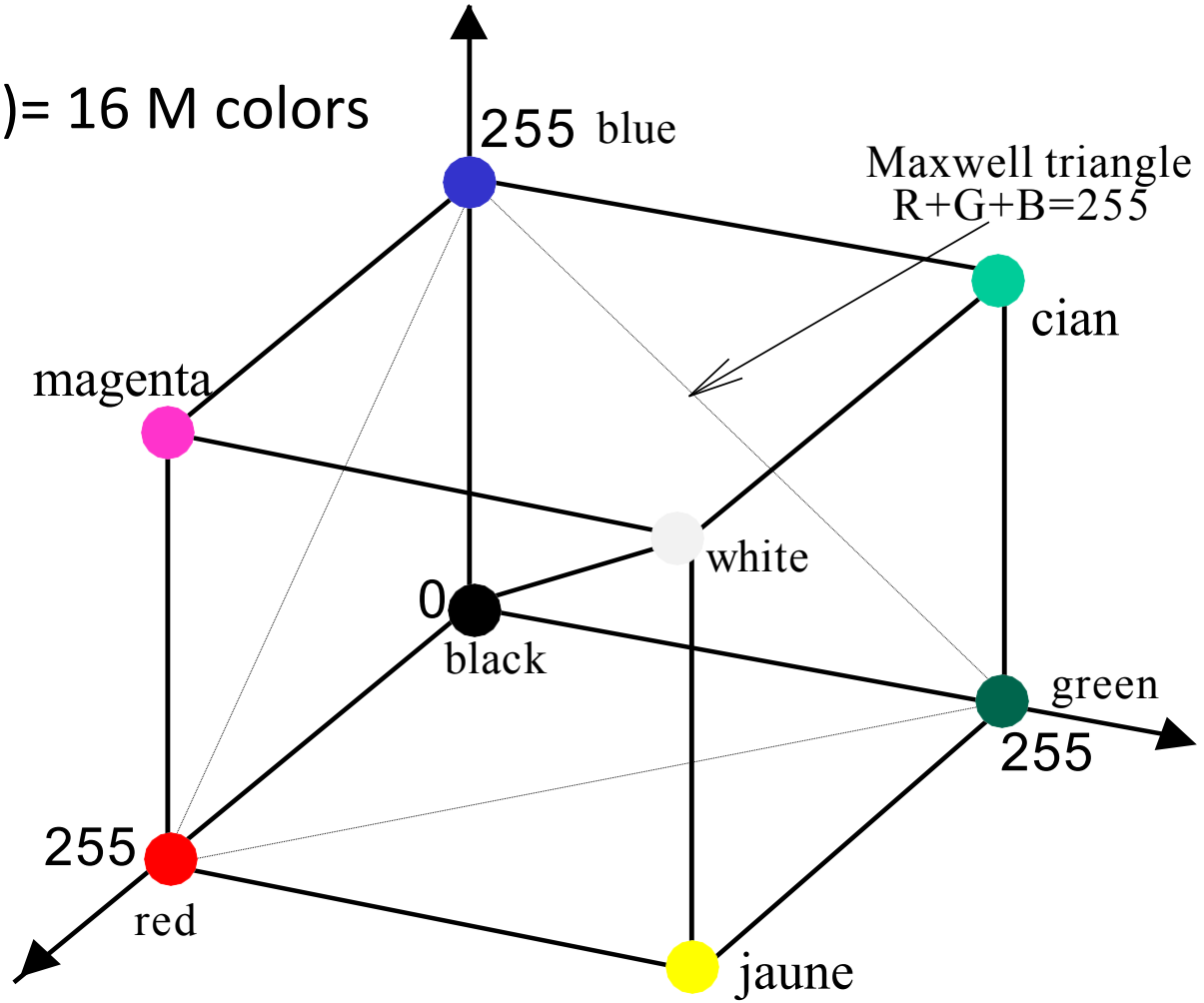




Color spaces: RGB & CMY

RGB: Red, Green, Blue & CMY: Cyan, Magenta, Yellow

- Additive synthesis of the color
- Screen, graphical card
- Images 24 bits (3*8 bits)= 16 M colors

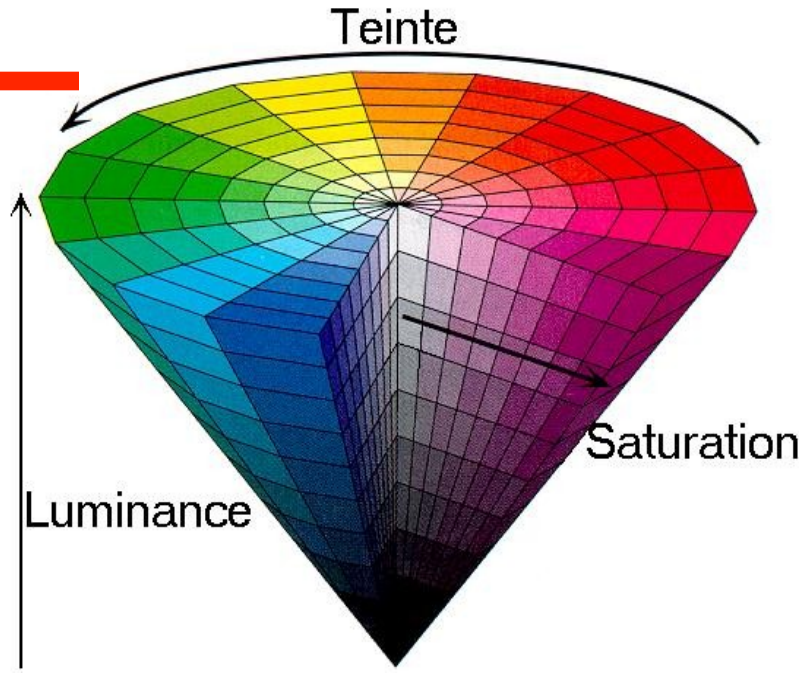


Color spaces: HSL

HSV: Hue, Saturation, Luminance

- Dominant wavelength (hue)
- Contribution (saturation)
- Intensity by surface unit (luminance)

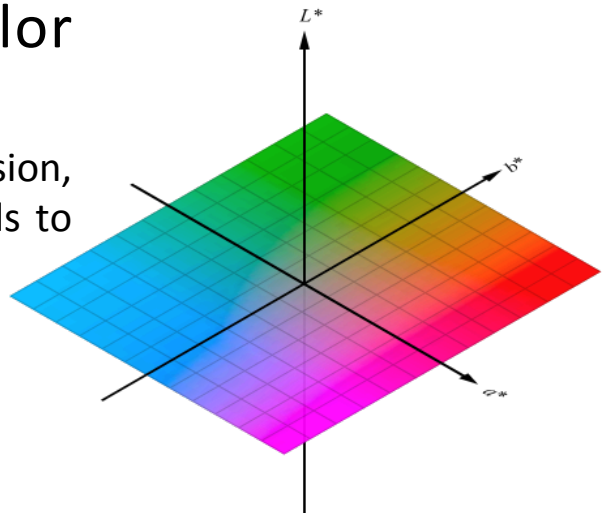
More closely align with the way human vision perceives color-making attributes



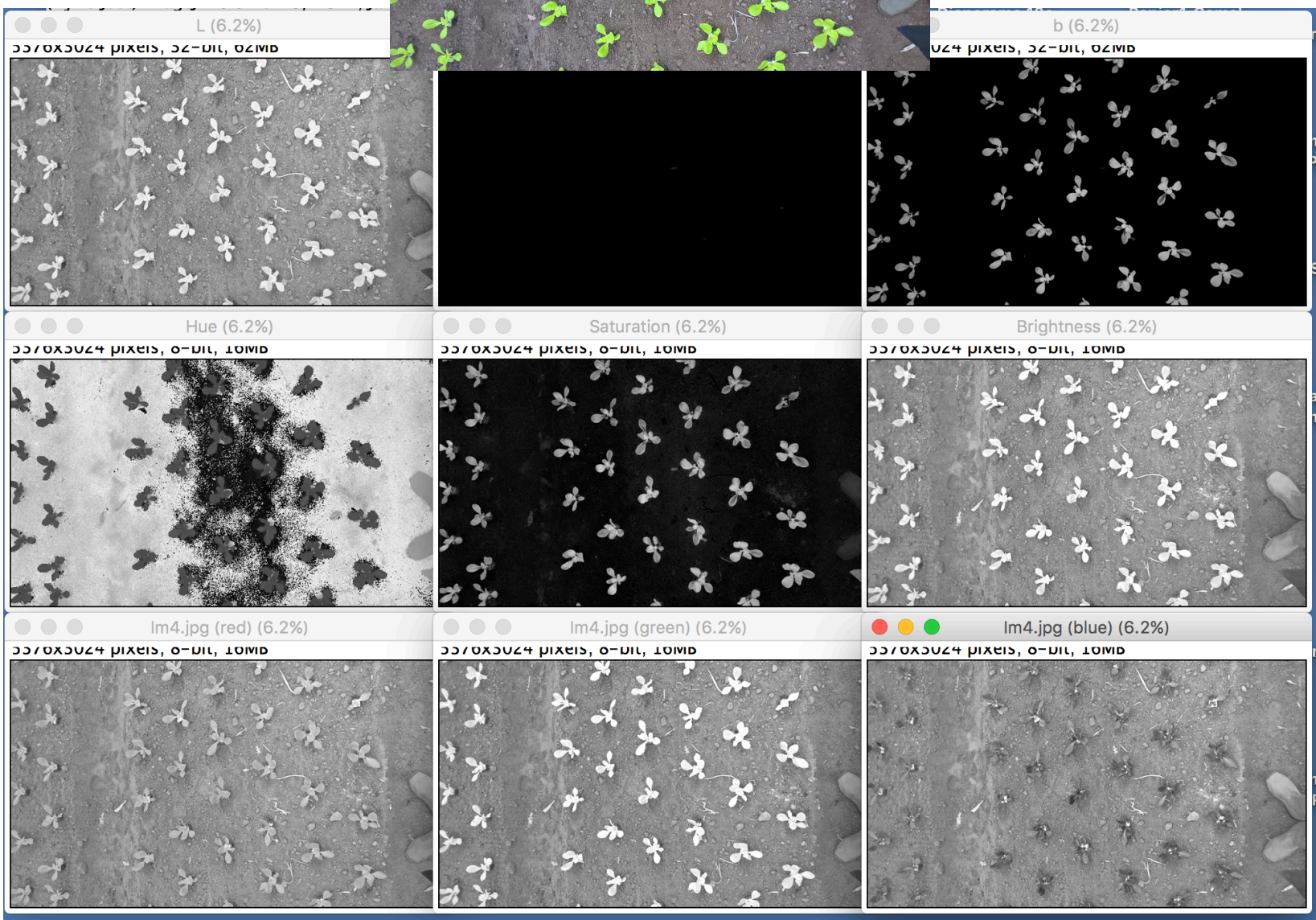
Color spaces: Lab

- L: lightness
- a and b: the green–red and blue–yellow color components.

CIELAB was designed to be perceptually uniform with respect to human color vision, meaning that the same amount of numerical change in these values corresponds to about the same amount of visually perceived change.



Example: color spaces



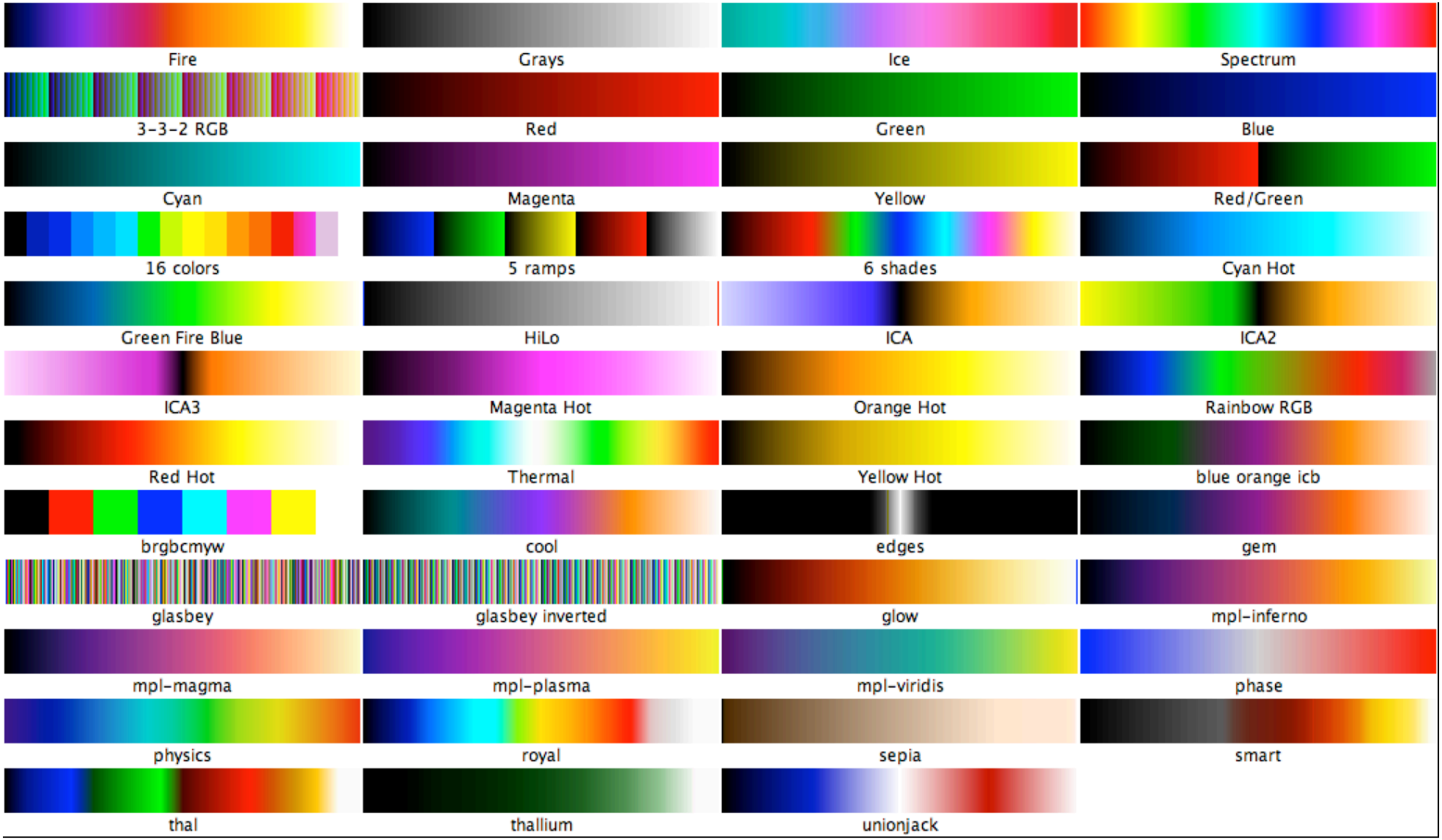
Lab

HSV

RGB

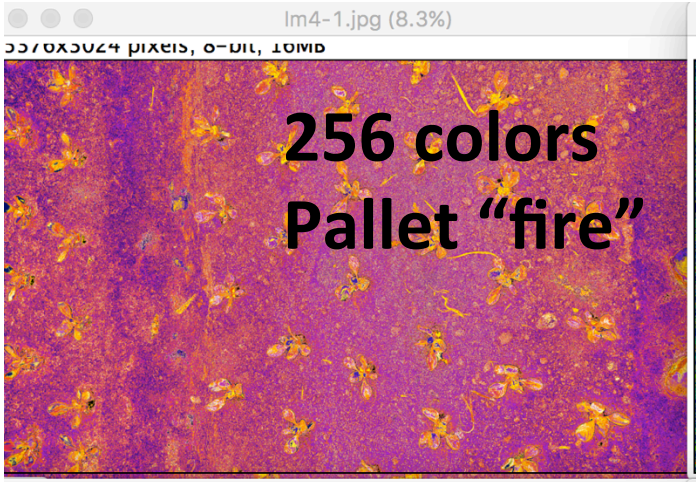
Color pallet / LUT (Look Up Tables)

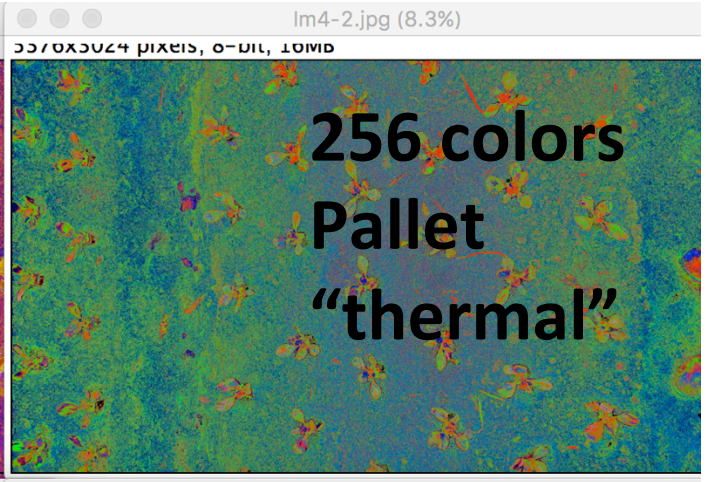
- 16 M of colors → 256 colors using color pallet/tables
- Indexed image = color pallet + index matrix



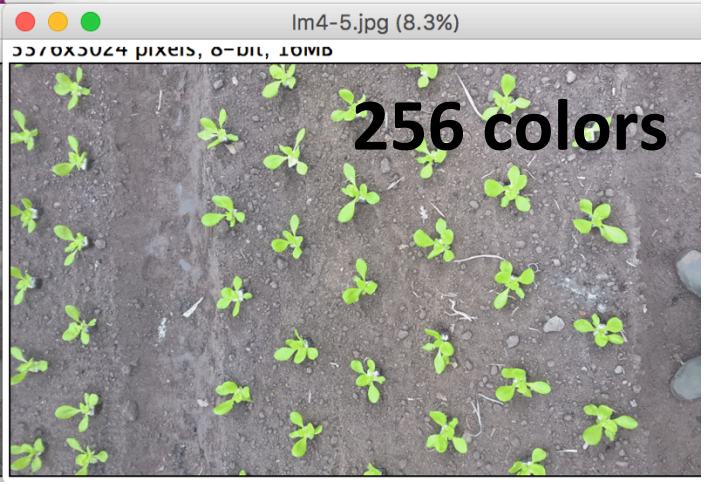
Examples


RGB


Im4-1.jpg (8.3%)
256 colors
Pallet “fire”


Im4-2.jpg (8.3%)
256 colors
Pallet
“thermal”

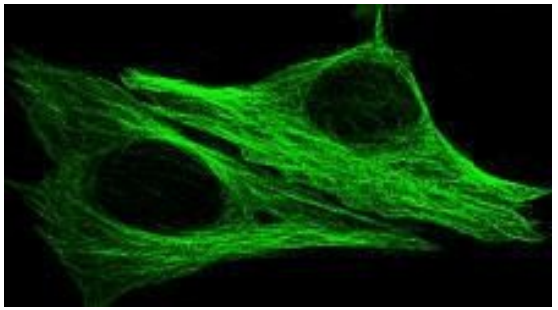

Im4-4.jpg (8.3%)
16 colors

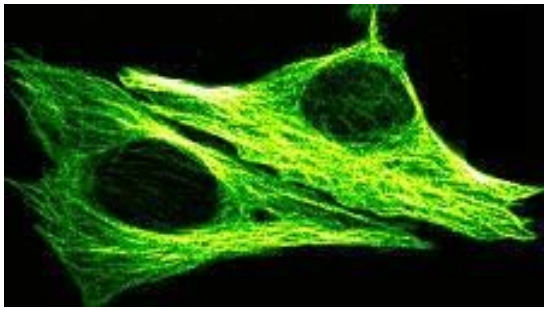

Im4-5.jpg (8.3%)
256 colors

Using LUT

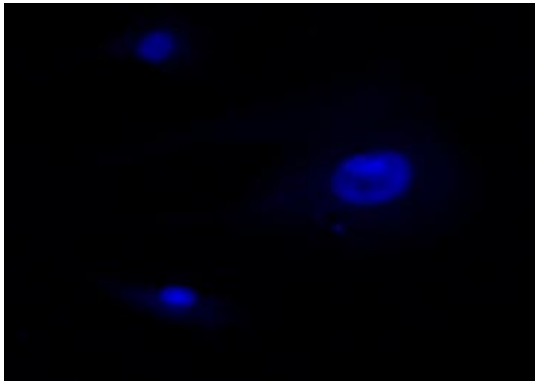
The judicious use of LUT can be very useful to highlight the characteristics of an image. Indeed, the human eye can perceive relatively few shades of colors in the same scene. In addition, it should be noted that the sensitivity of the human eye to light intensities related to wavelengths is uneven. It is easier to perceive shades of yellow, orange or turquoise than shades of dark blues or reds. The pseudo-coloring of the images can therefore make the data more visible without altering the data included in the image.

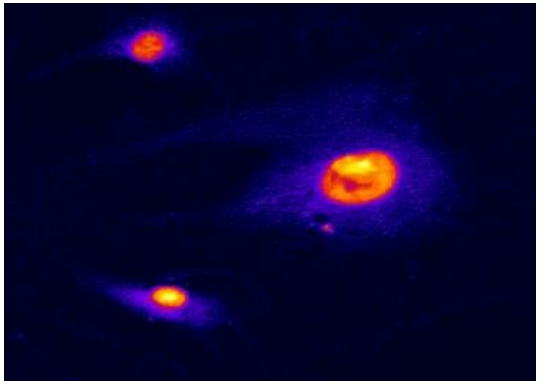
- *LUT classic « Green »*
- *LUT « Hot Green »*





- *LUT classique « Blue»*
- *LUT « Fire »*





Format of digital images

JPEG, JPEG2000, PNG, GIF, TIFF, BMP, XMP, Targa , FITS, PM, ...

What characterizes them?

Main characteristics intrinsic to a format:

- status in relation to patents: free formats and some " Owners".
- number of colors supported
- data compression: lossless or lossy (JPEG)

...

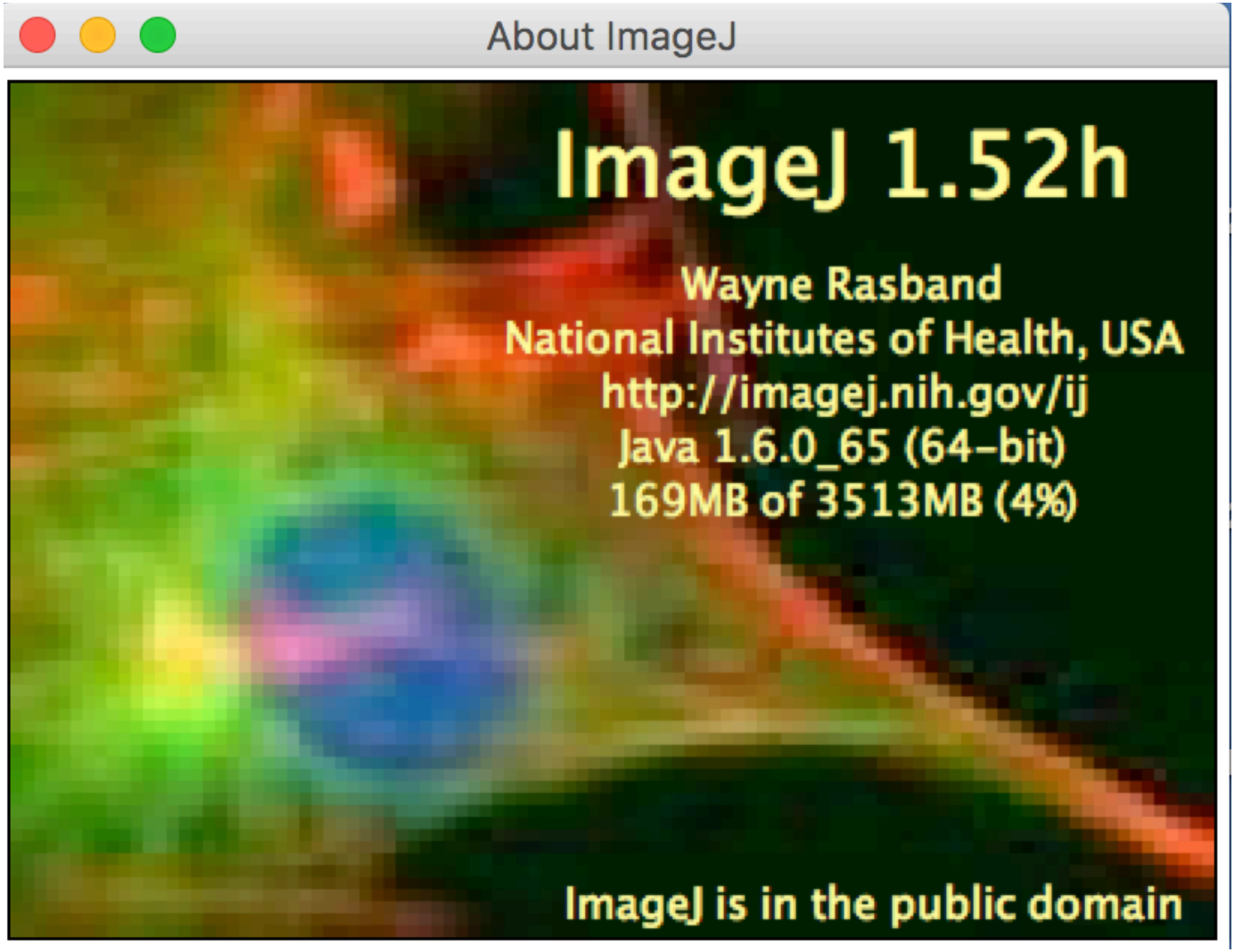
- transparency: One of the colors of the palette can be ignored when display
- interlacing: displaying a low resolution version refined when loading
- animation movies, 3D visualization
- uses: scientific computing , archiving, Internet

Format of digital images: purpose

	No compression →analysis	For webpage	Stacks of image	Video
GIF	NON	OUI	OUI	OUI
PNG	NON	OUI	NON	Variante MNG
JPG	NON	OUI	NON	MPEG
JPEG2000	NON	OUI	NON	NON
TIFF	OUI	NON	OUI	NON
RAW	OUI	NON	OUI	NON

Converting images

to / from	8-Bits	8-Bits Color	16-Bits	32-Bits	RGB Color
8-Bits		LUT removing	Nothing in appearance. Loss of values and loss of precision.	Nothing in appearance. Loss of values and loss of precision.	Grey-level = average of RGB channels.
8-Bits Color	Nothing		Nothing in appearance. Loss of values and loss of precision.	Nothing in appearance. Loss of values and loss of precision.	Nothing in appearance. LUT creating.
16-Bits	Nothing	LUT preserving		Nothing in appearance. Loss of values and loss of precision.	Loss of colors
32-Bits	Nothing	LUT preserving	Nothing in appearance.		Loss of colors
RGB Color	RGB channels = grey-level	Nothing in appearance. Creating RGB channels	RGB channels = grey-level	RGB channels = grey-level	



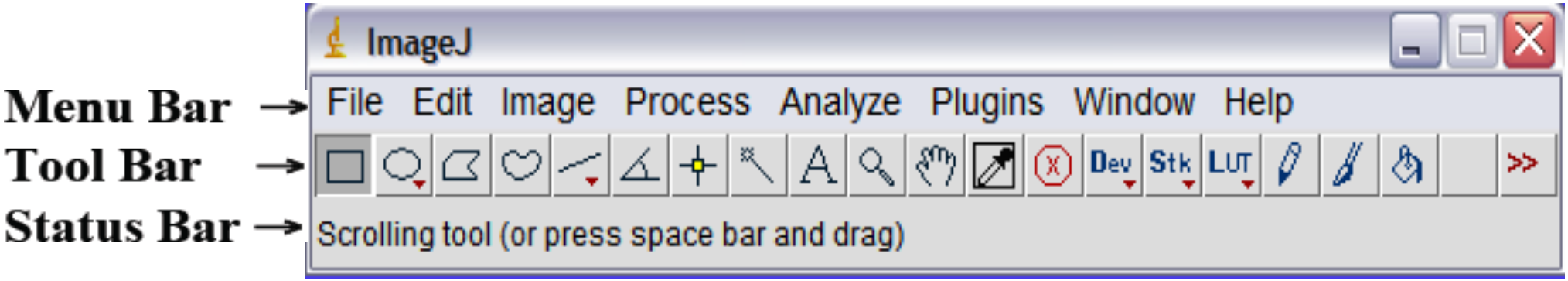
http://rsb.info.nih.gov/ij

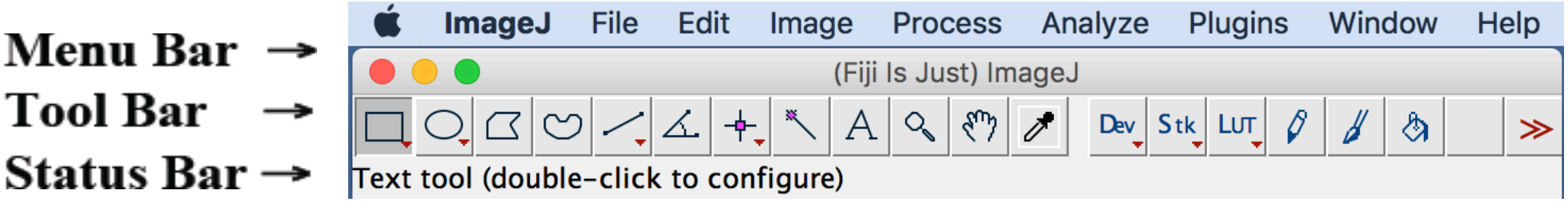
Resources

ImageJ Web Site http://rsb.info.nih.gov/ij

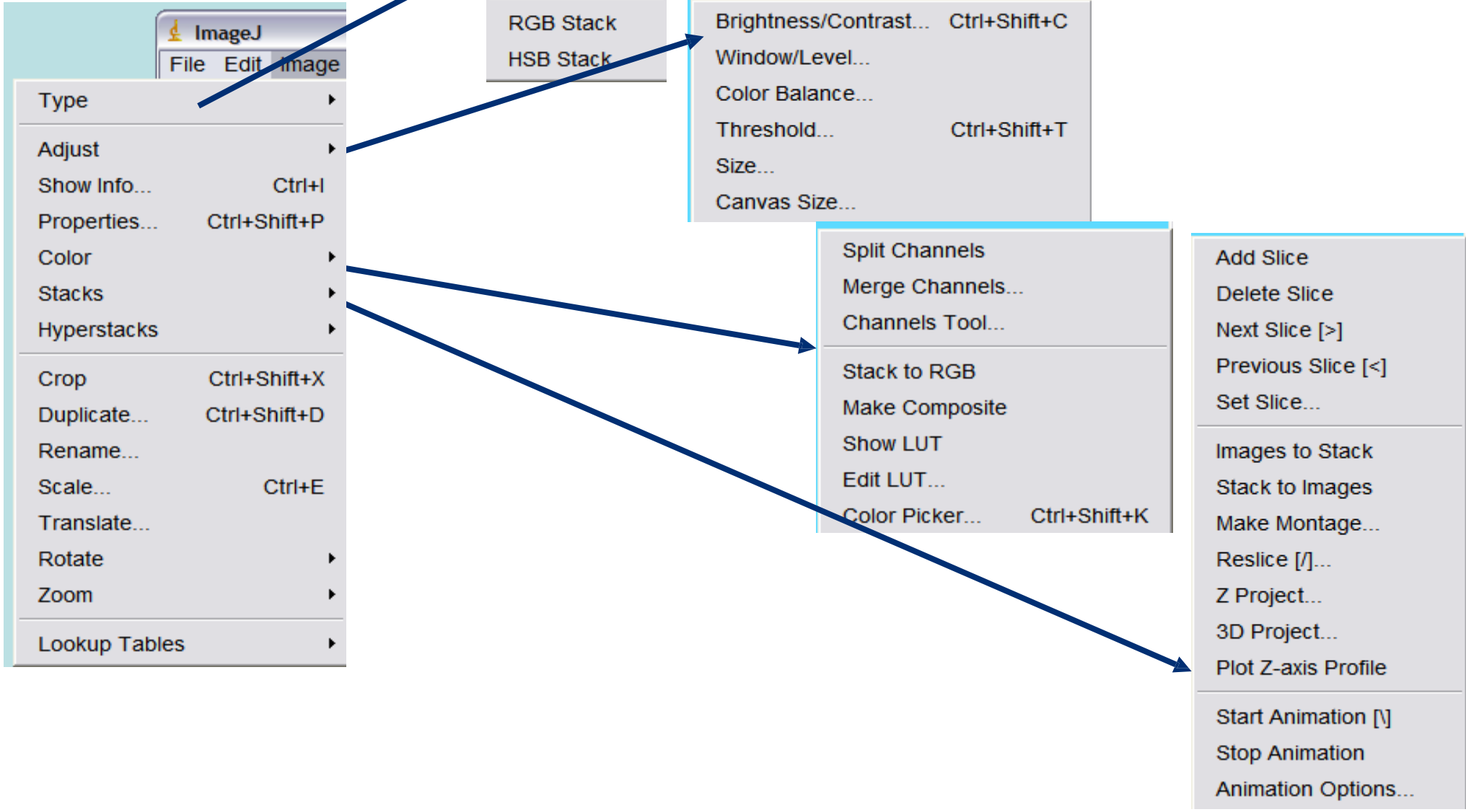
Wiki: http://imagejdocu.tudor.lu/

Digital Image Processing, An Algorithmic Introduction using Java; Springer Verlag, 2008

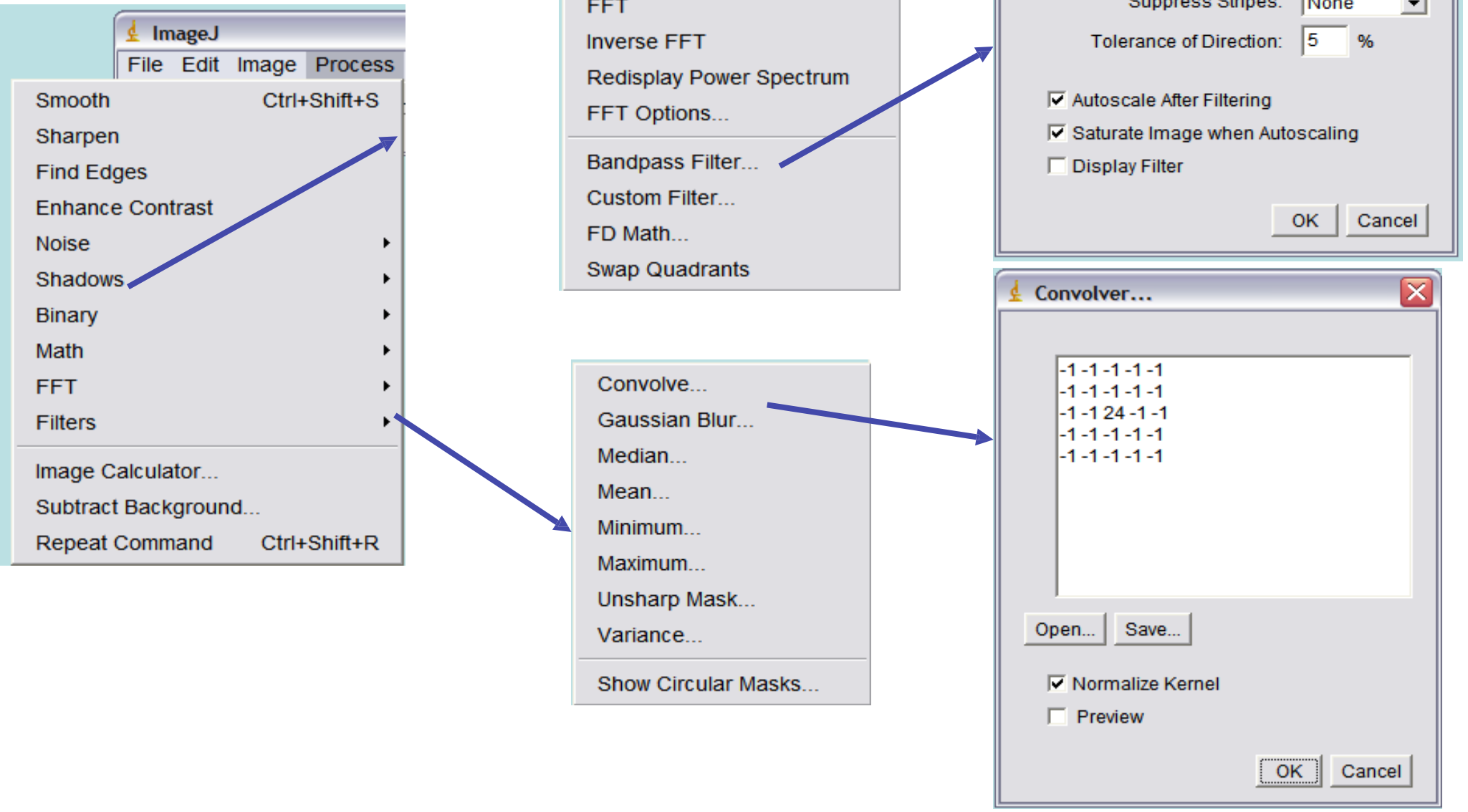

Menu Bar
Tool Bar
Status Bar
ImageJ
File Edit Image Process Analyze Plugins Window Help
Dev Stk LUT
Scrolling tool (or press space bar and drag)

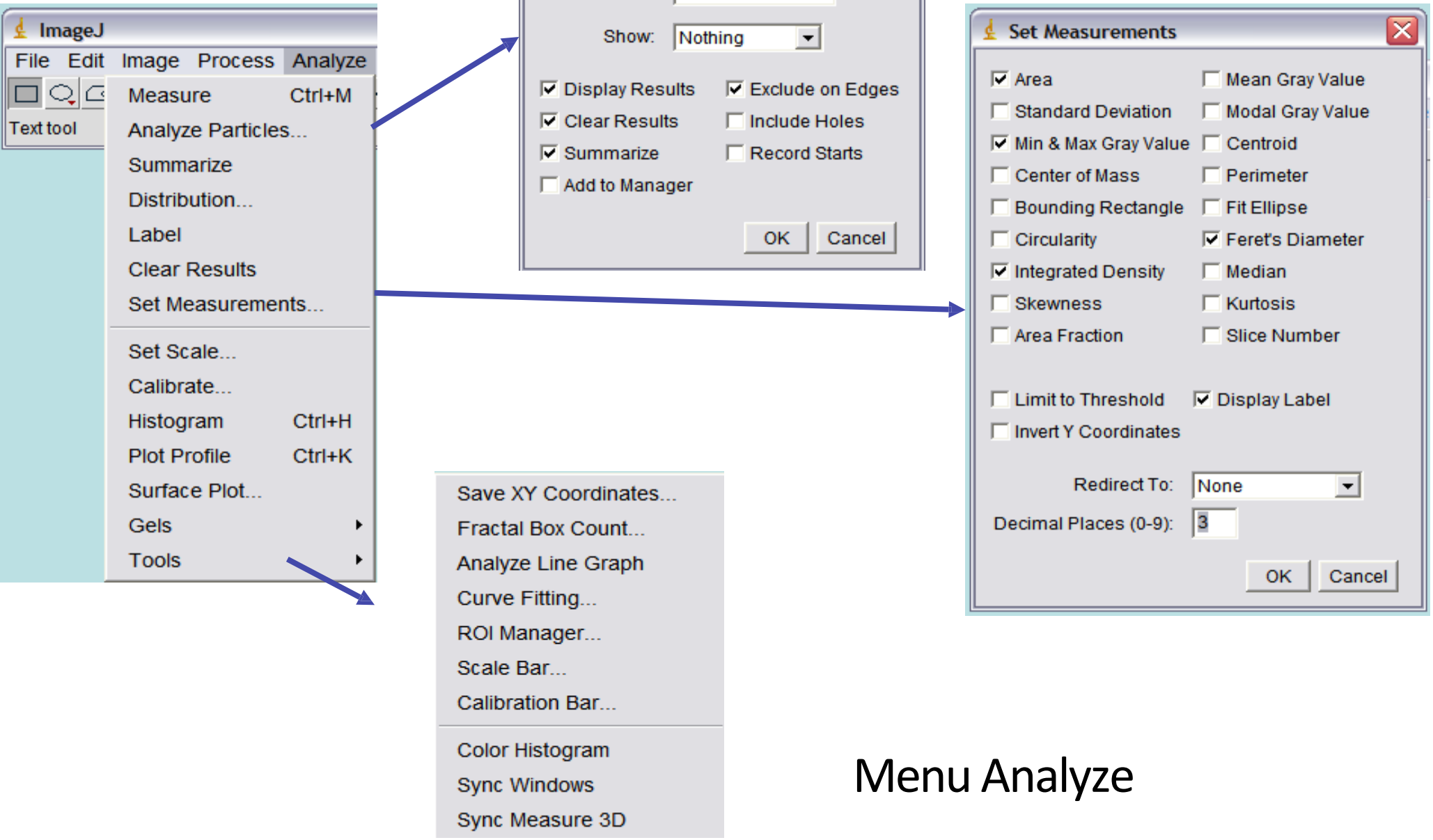

Menu Bar
Tool Bar
Status Bar
ImageJ File Edit Image Process Analyze Plugins Window Help
(Fiji Is Just) ImageJ
Dev Stk LUT
Text tool (double-click to configure)

Menu Image


ImageJ
File Edit Image
Type
Adjust
Show Info... Ctrl+I
Properties... Ctrl+Shift+P
Color
Stacks
Hyperstacks
Crop Ctrl+Shift+X
Duplicate... Ctrl+Shift+D
Rename...
Scale... Ctrl+E
Translate...
Rotate
Zoom
Lookup Tables
8-bit
16-bit
32-bit
8-bit Color
RGB Color
RGB Stack
HSB Stack
Brightness/Contrast... Ctrl+Shift+C
Window/Level...
Color Balance...
Threshold... Ctrl+Shift+T
Size...
Canvas Size...
Split Channels
Merge Channels...
Channels Tool...
Stack to RGB
Make Composite
Show LUT
Edit LUT...
Color Picker... Ctrl+Shift+K
Add Slice
Delete Slice
Next Slice [>]
Previous Slice [<]
Set Slice...
Images to Stack
Stack to Images
Make Montage...
Reslice [/]...
Z Project...
3D Project...
Plot Z-axis Profile
Start Animation [\]
Stop Animation
Animation Options...

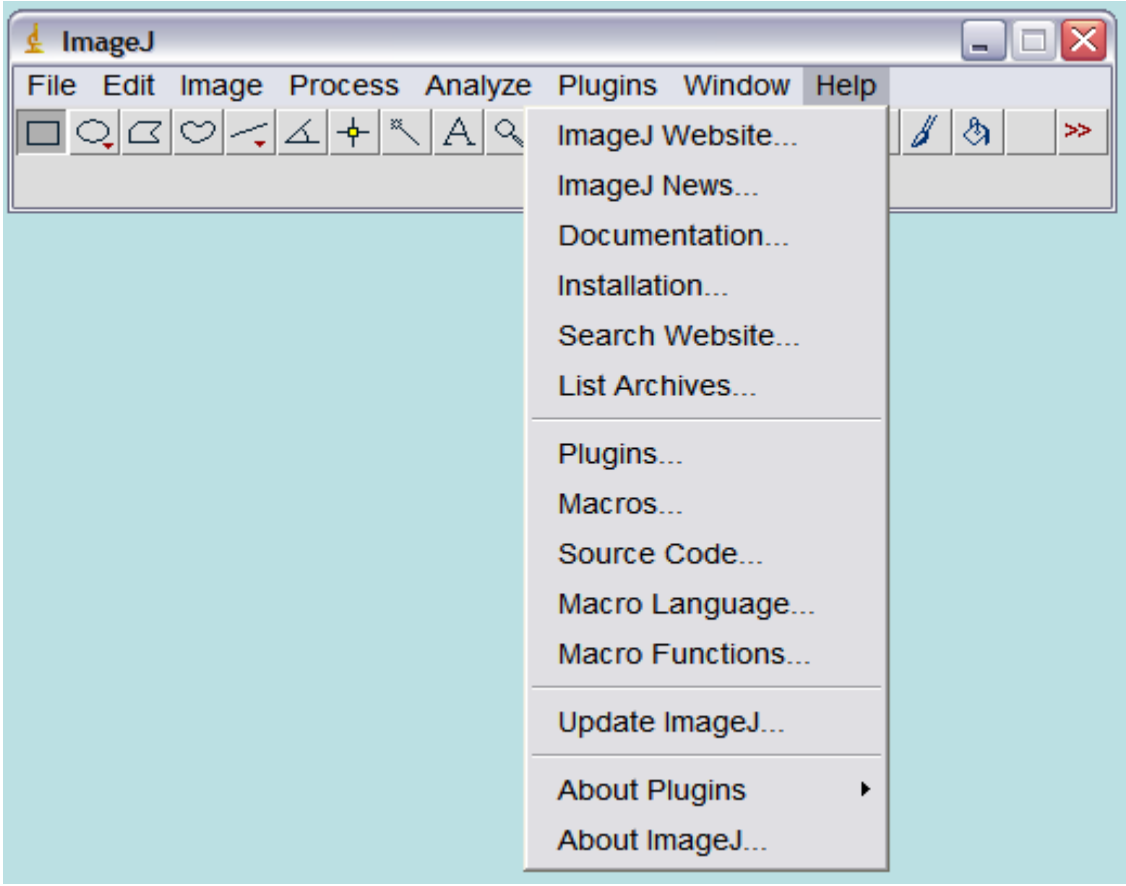
Menu Process





Menu Analyze

Menu help


ImageJ
File Edit Image Process Analyze Plugins Window Help
ImageJ Website...
ImageJ News...
Documentation...
Installation...
Search Website...
List Archives...
Plugins...
Macros...
Source Code...
Macro Language...
Macro Functions...
Update ImageJ...
About Plugins
About ImageJ...

Hands on n°1

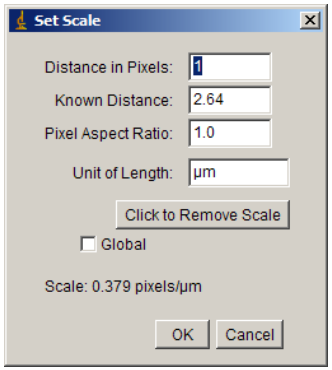
1) Load image meb.bmp

2) Calibrate image with 10 pixels=1 micron

3) Display calibration and measure diameter of biggest objects

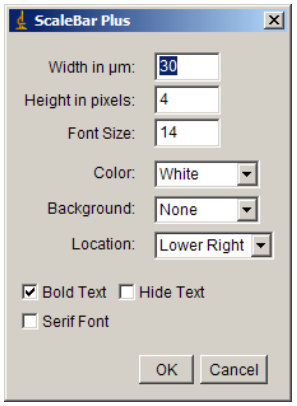
Calibration:

Analyze > Set Scale…

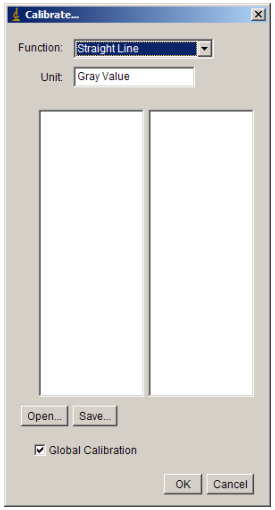


Display calibration:

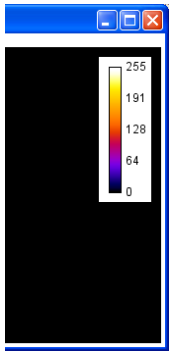
Analyze > Tools > Scale Bar…

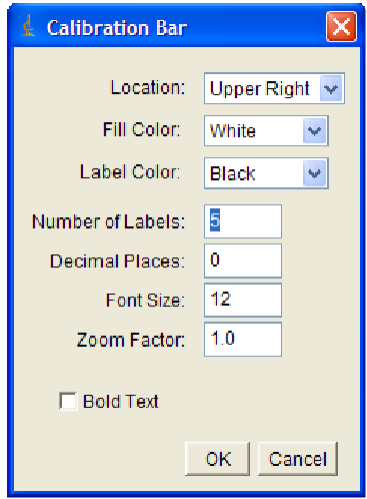


Menu Analyze > Calibrate…

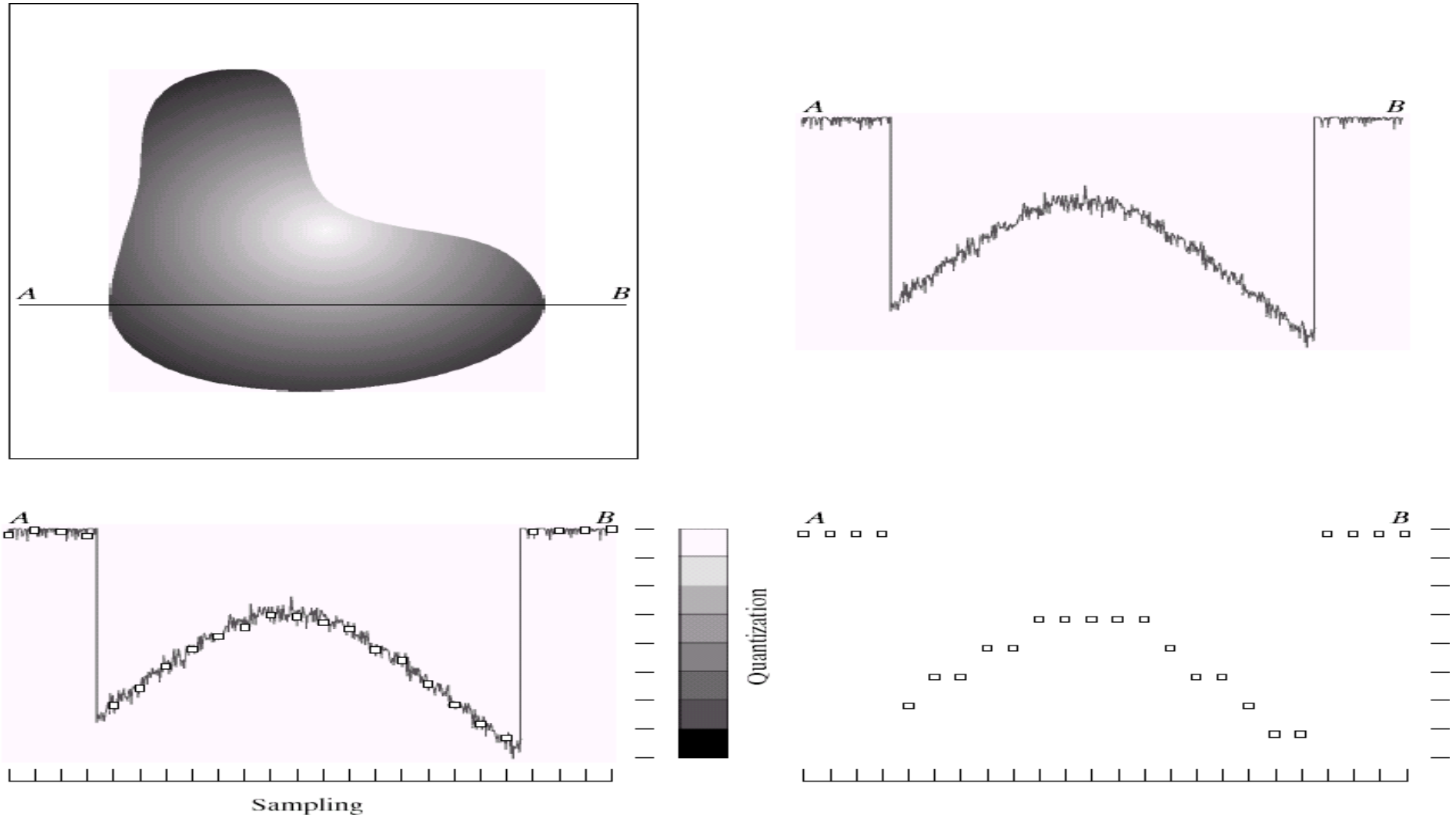


Analyse > Tools > Calibration Bar…





Sampling and quantification

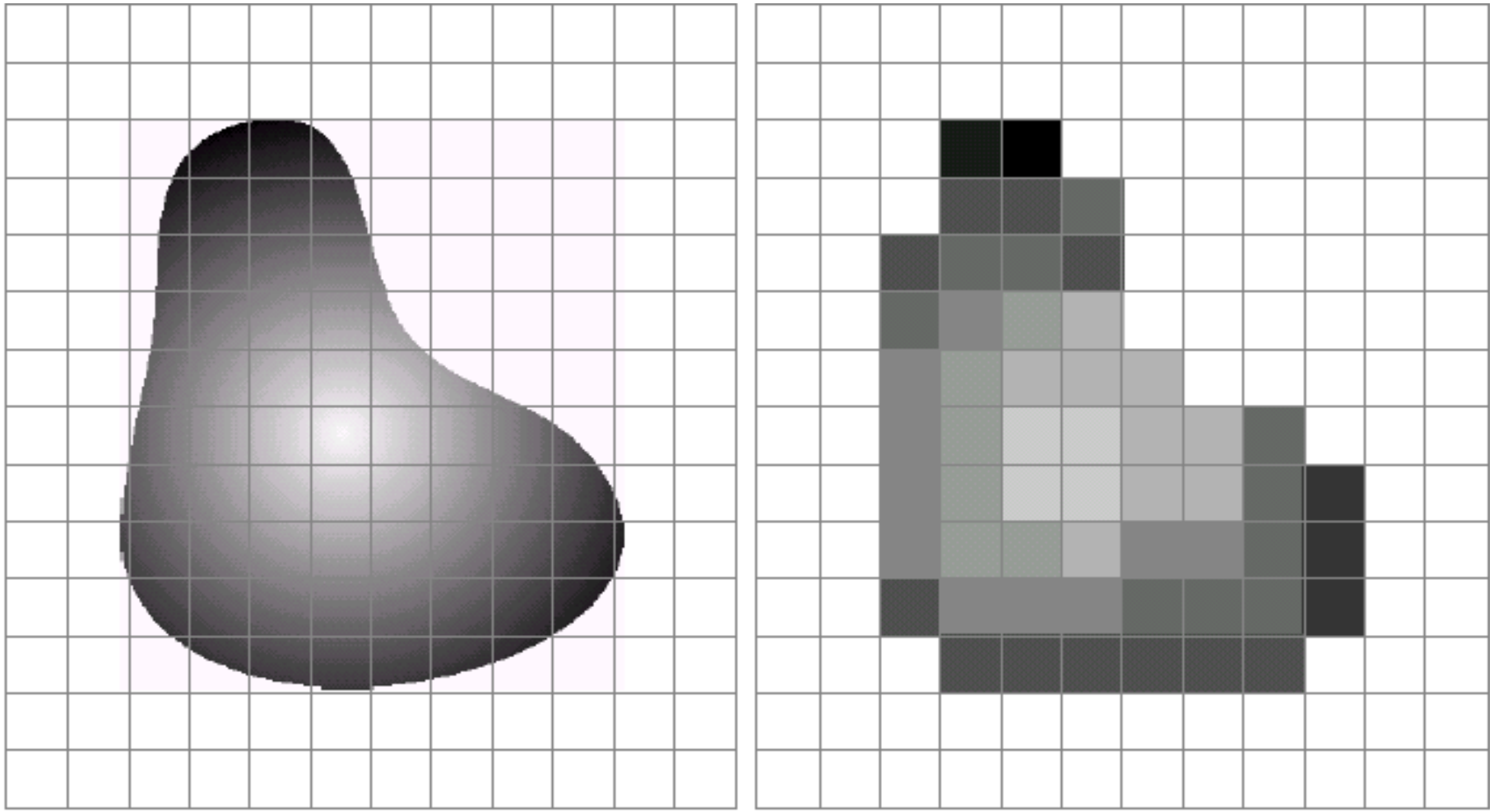


a b
c d

FIGURE 2.16 Generating a digital image. (a) Continuous image. (b) A scan line from *A* to *B* in the continuous image, used to illustrate the concepts of sampling and quantization. (c) Sampling and quantization. (d) Digital scan line.

Source : Gonzalez and Woods. Digital Image Processing. Prentice-Hall, 2002.

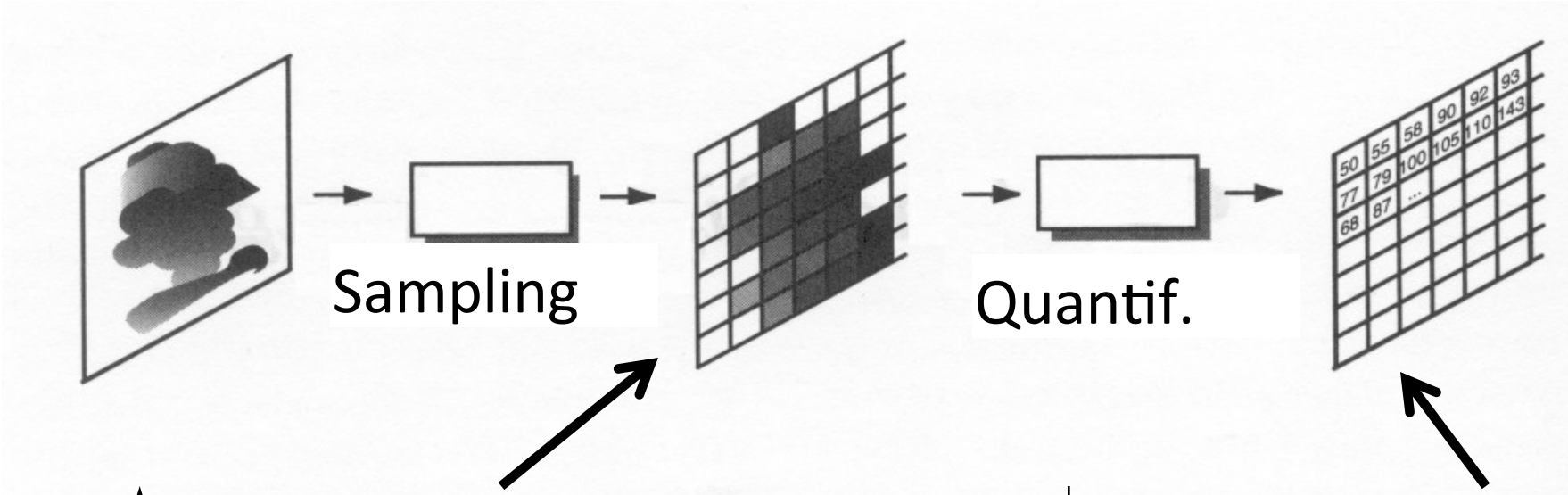
Sampling and quantification



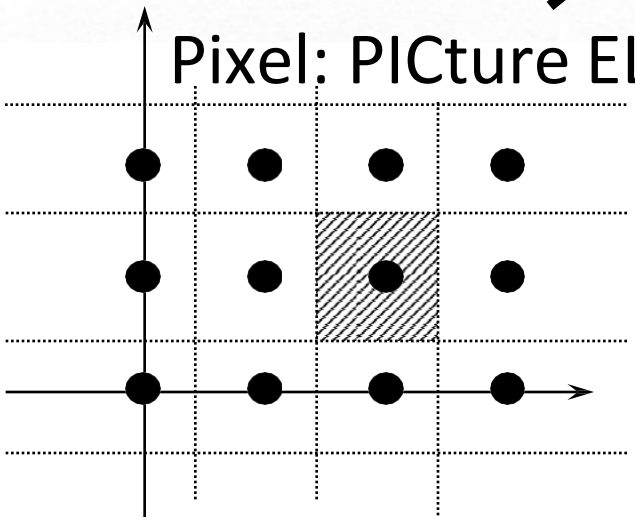
a b

FIGURE 2.17 (a) Continuos image projected onto a sensor array. (b) Result of image sampling and quantization.

Sampling and quantification



Pixel: PICture ELement



0.1	0.23
0.15	0.50

51	117
76	255

→ spatial and tonal resolution

Spatial and tonal resolution

256x256

128x128

64x64

32x32

- Pixellisation
- Loss of clearness
- Less visible details

6 bits

4 bits

3 bits

2 bits

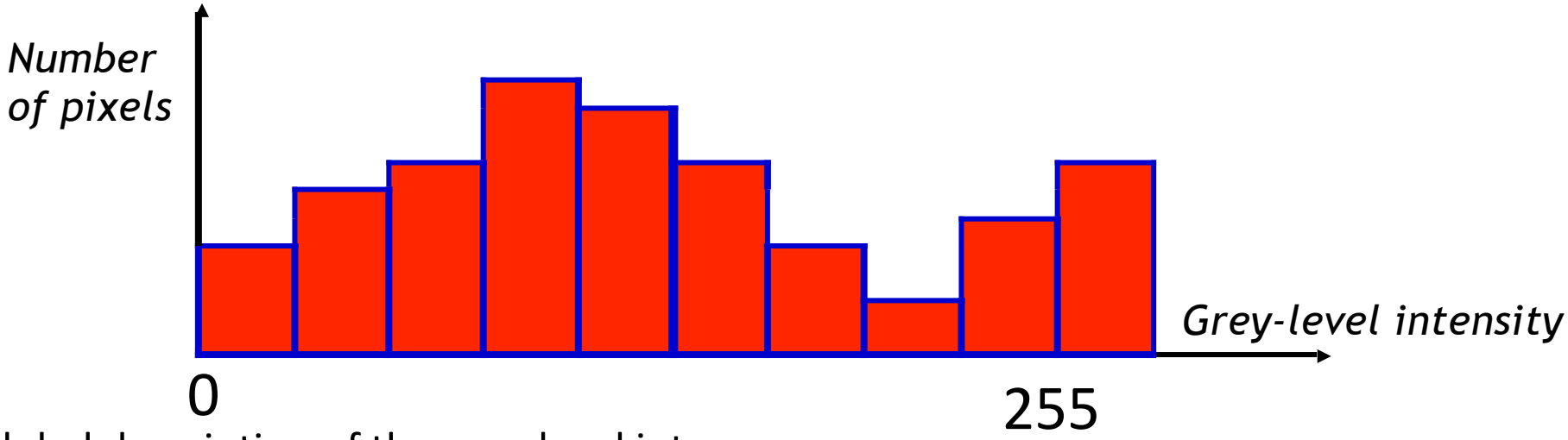
1 bit

- Appearence of fake edges
- Quantification noise

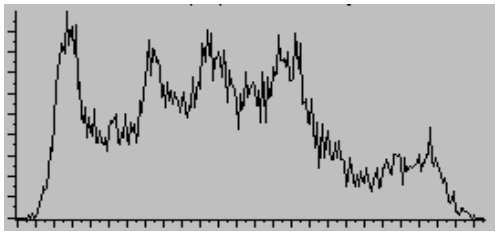
Source : Antoine Mazanera. Introduction et filtrage. ENSTA (France).

Histogram

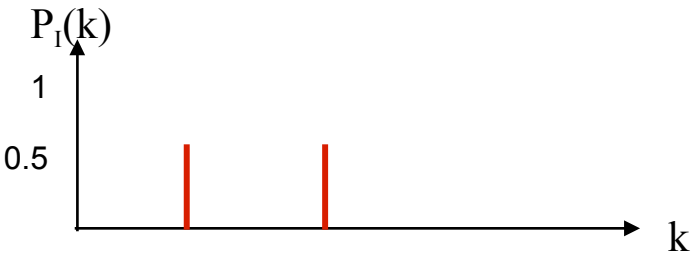
Histogram = displaying number of pixels according to grey-level intensities (0-255)

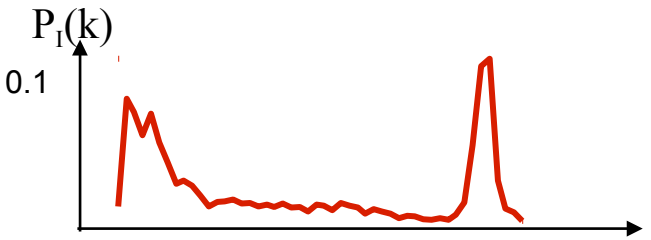


- Global description of the grey-level int.,
- Can be seen as probability distribution of grey-level int.,
- Useful tool to analyze, to improve or transform image,
- The simplest segmentation techniques come from the analysis of the histogram.



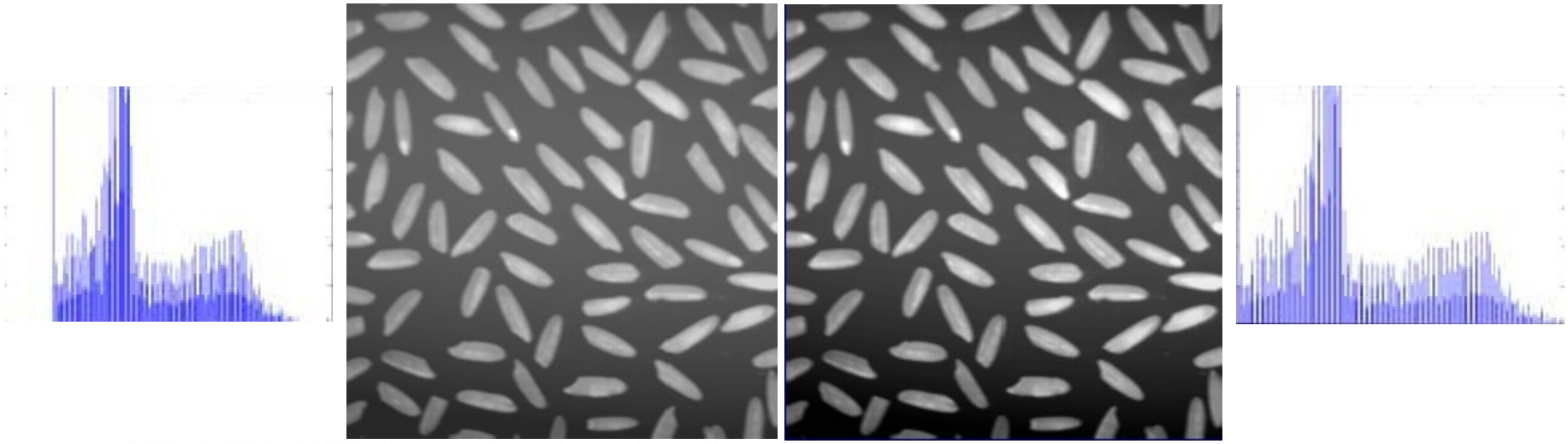
Histogram

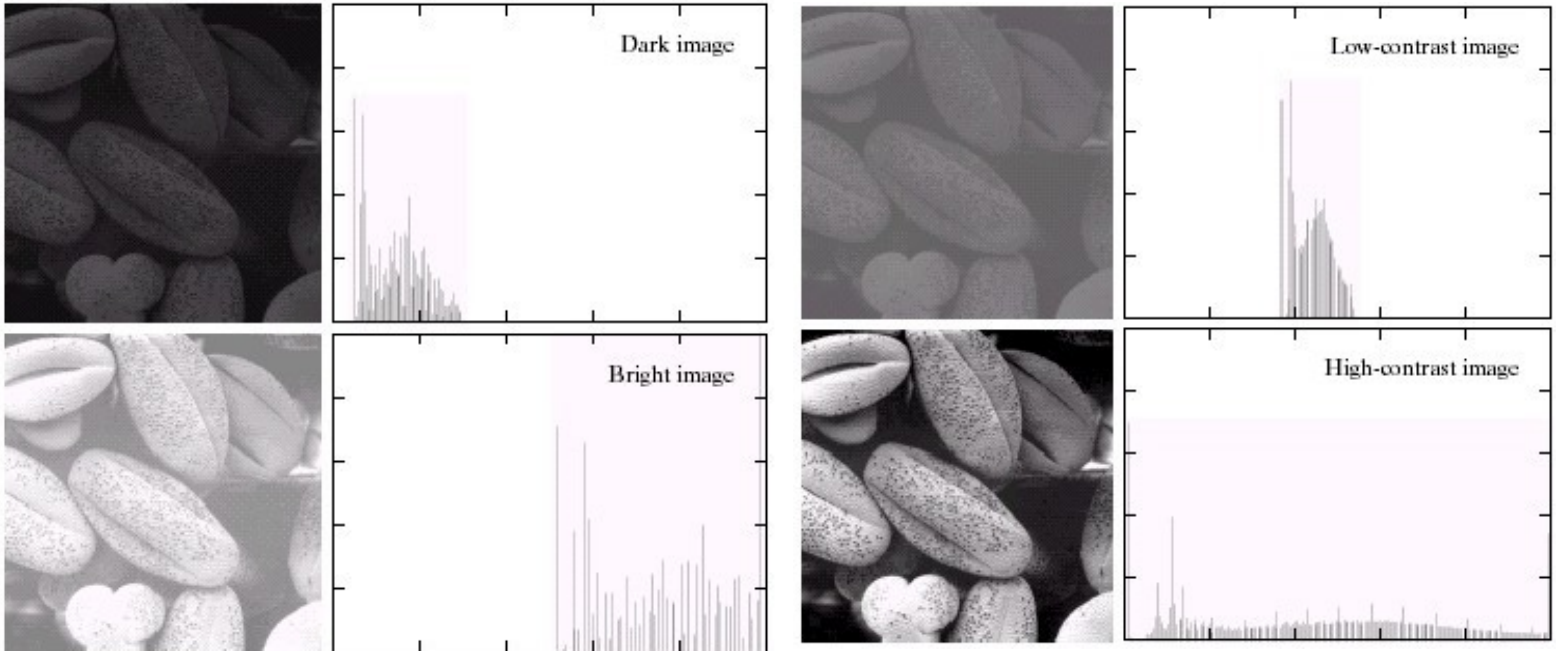




Dynamic of an image = [min value ,max value]

Histogram





a b

FIGURE 3.15 Four basic image types: dark, light, low contrast, high contrast, and their corresponding histograms. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)

Histogram

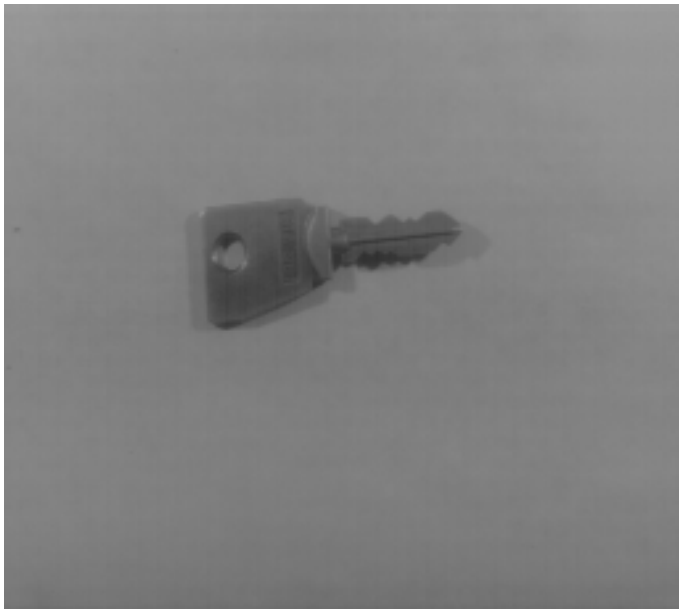
Let’s build a histogram

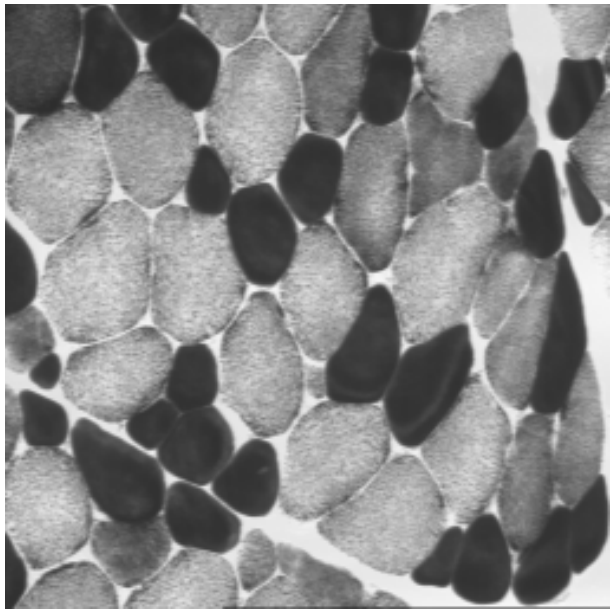
1	2	1	3	1
2	1	1	3	1
1	2	3	1	1

Image 3x5

Hands on n°2

1) load images clef.bmp & muscle.bmp
2) display their respective histograms Process/Enhance contrast/Normalize (set *saturated pixel* to 0).
3) Can you explain the shape of histograms ?
4) Focussing on histogram of muscle.bmp, how can we decide which muscular fibers are denser (red (dark) and white (gray)) ?





Operations on images by transforming histogram

Transforming histogram → changing grey-level values of pixels

For each value of grey-levl in initial image → new value in transformed operated image

Main transformations are:

- Resizing dynamic
- Negative
- Equalization
- Exponential, logarithme, ….
- Thresholding

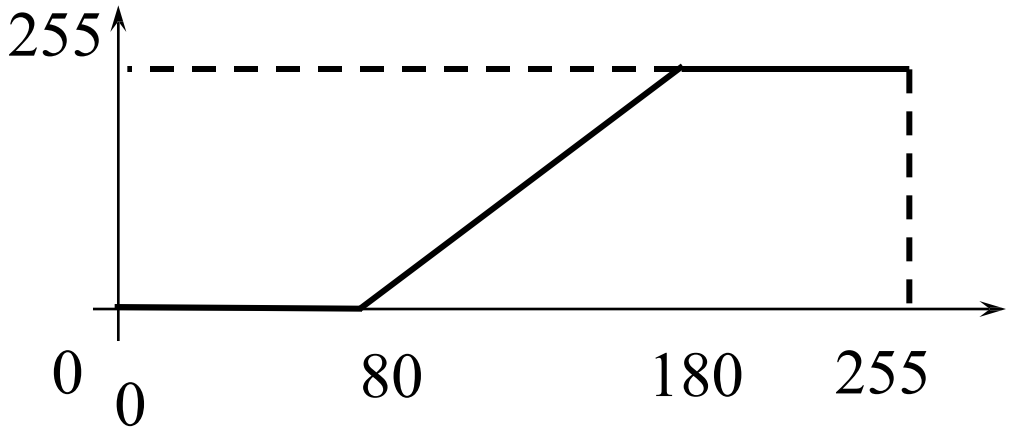
Transformation of grey-level : *v=f(u)*

u grey-level in initial image

v grey-level in operated image

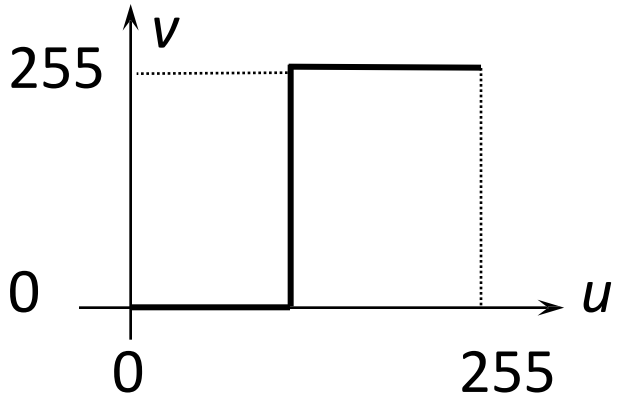
Example:

$$\begin{cases} 0 < u < 80\,, & v = 0 \\ 80 < u < 180\,, & \textit{relation linear} \\ 180 < u < 255\,, & v = 255 \end{cases}$$



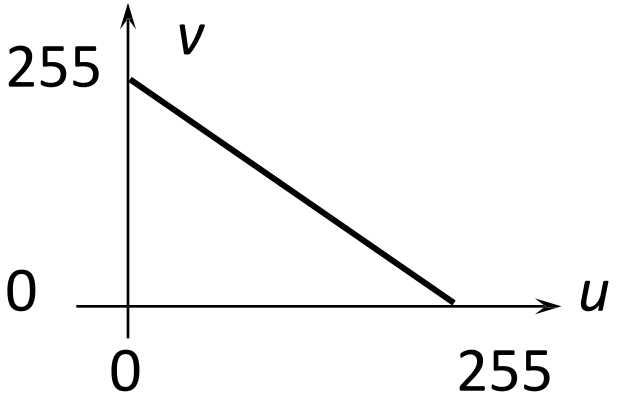
Operations on images by transforming histogram

Thresholding (segmentation)



Negative

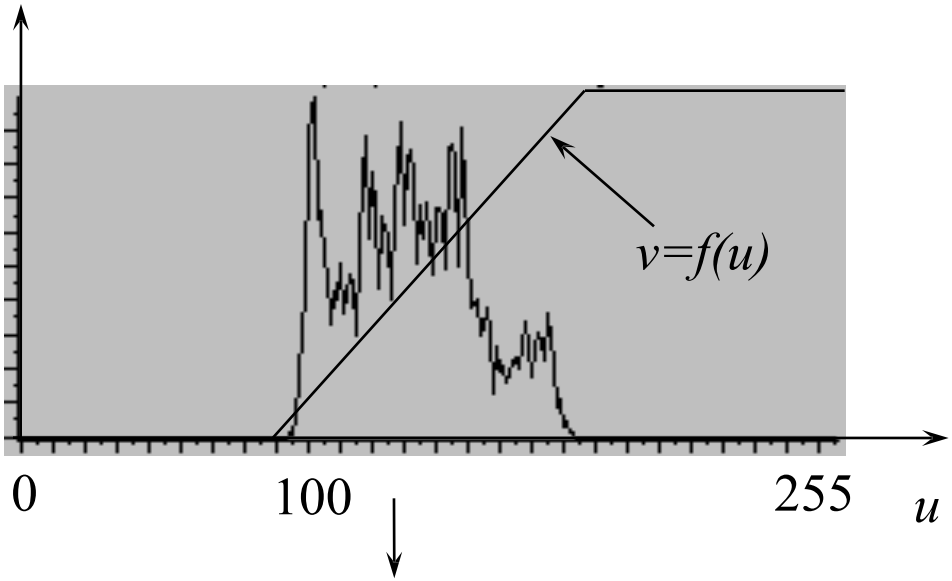
A

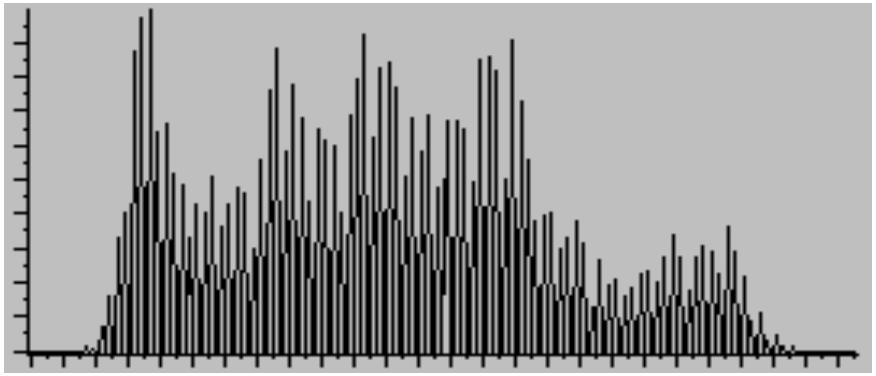


$\overline{\mathbf{A}}$

Operations on images by transforming histogram

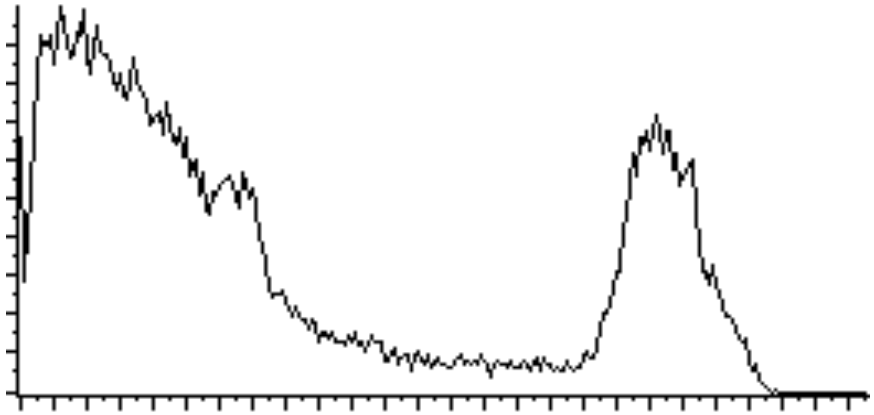
Resizing dynamic

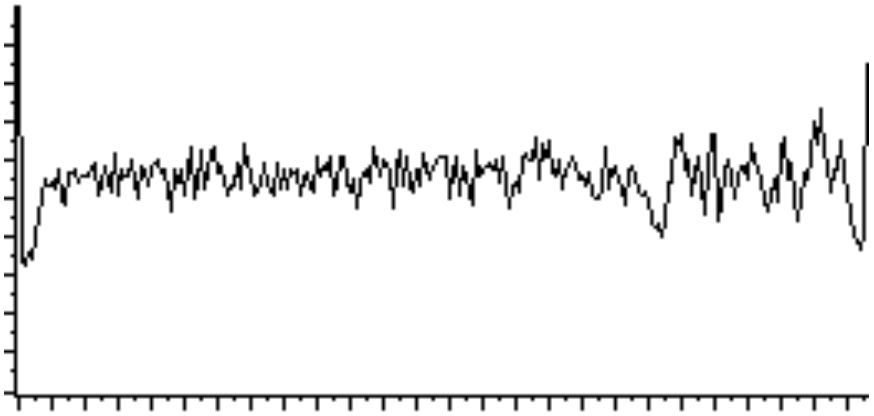




Operations on images by transforming histogram

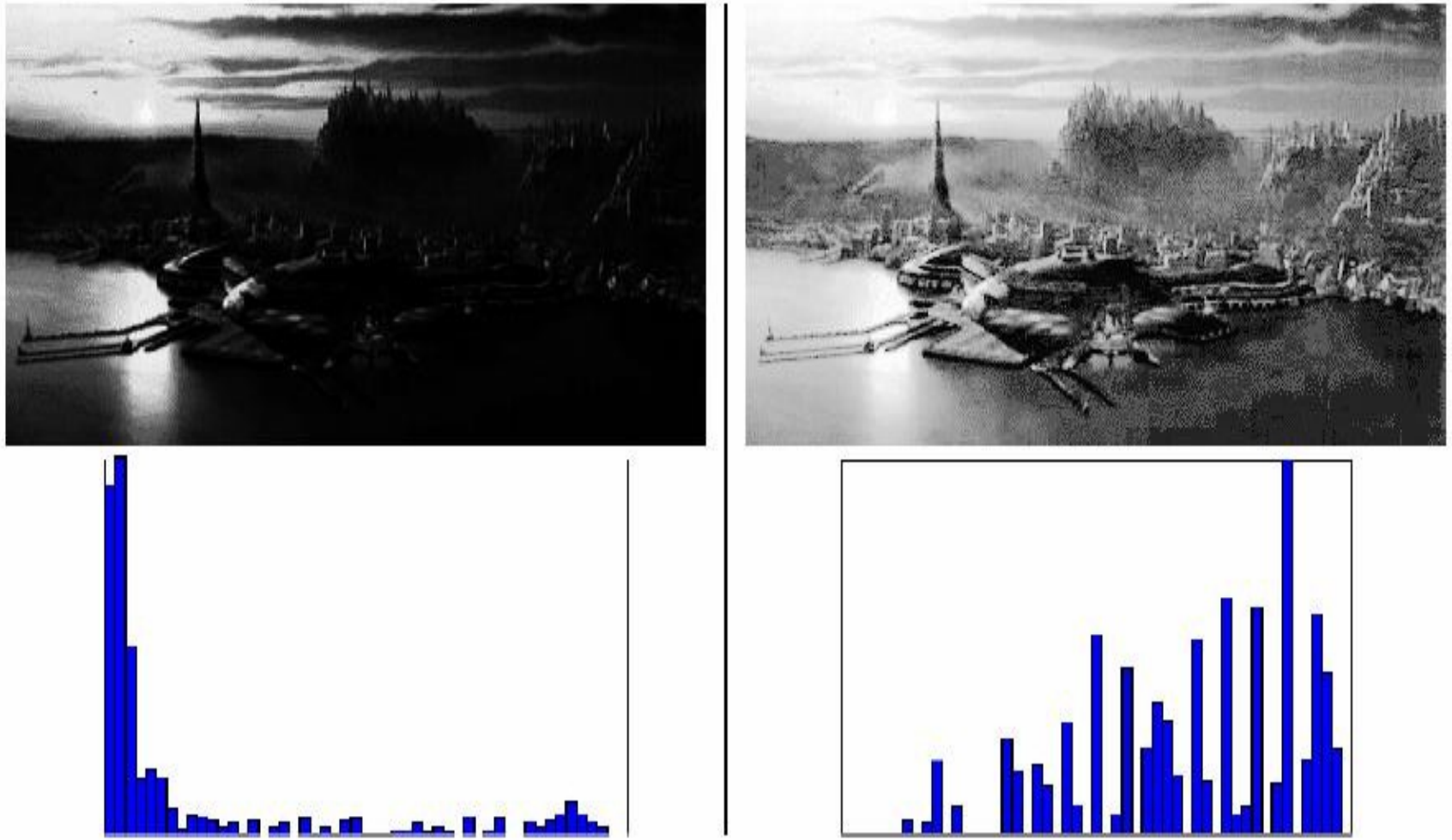
Equalization (→ flattenig the histogram)





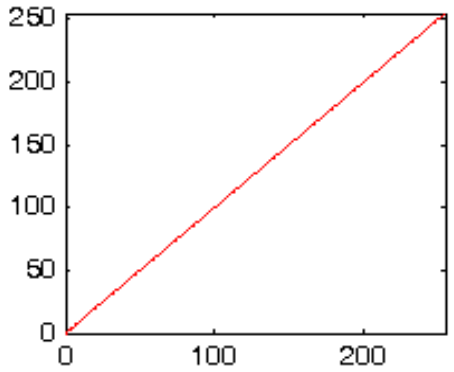
Operations on images by transforming histogram

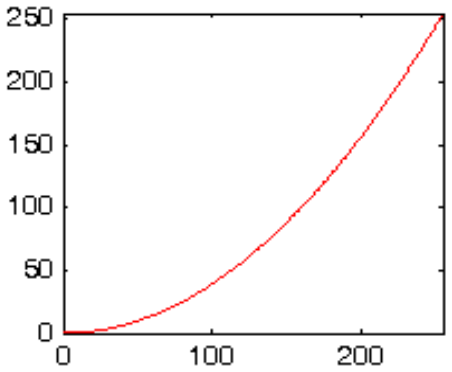
Equalization (→ flattenig the histogram)

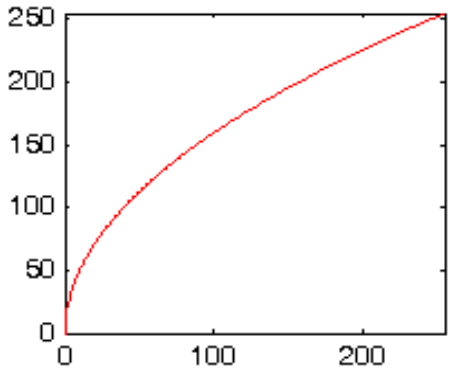


Operations on images by transforming histogram

Exp. Log.







Hands on n°3

Resizing dynamic

1) Load image quito.bmp.

2) Display histogram.

3) Resize the dynamic and display operated image

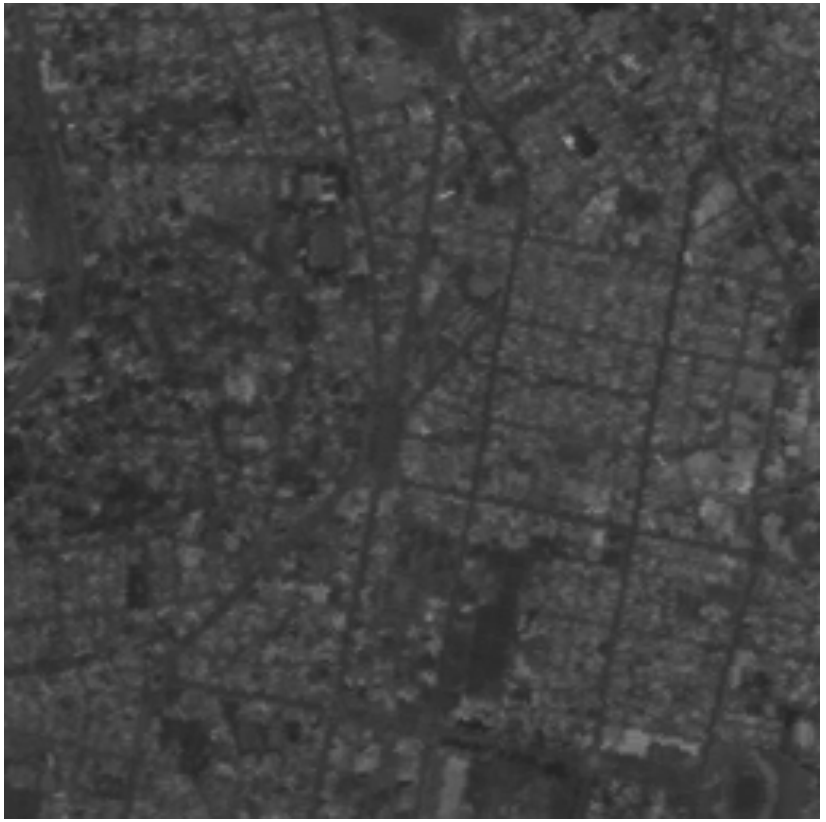
Equalization

1) Load image quito.bmp.

2) Display histogram.

3) Egalize histogram and display operated image

4) Repeat this operation. What happens ? Why ?

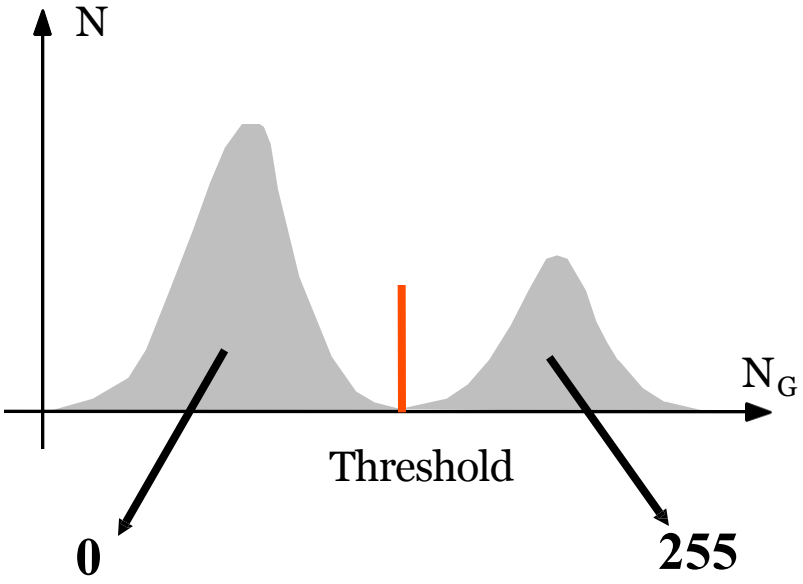


Thresholding

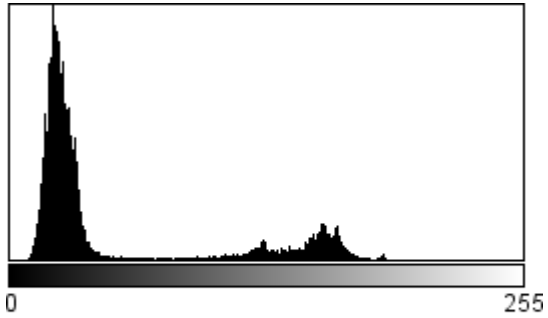
Single thresholding:

- Pixels with grey-level value < threshold → classe 0
- Pixels with grey-level value > threshold → classe 1 (1 or 255)

Simple technique/approach for region segmentation → various improvements

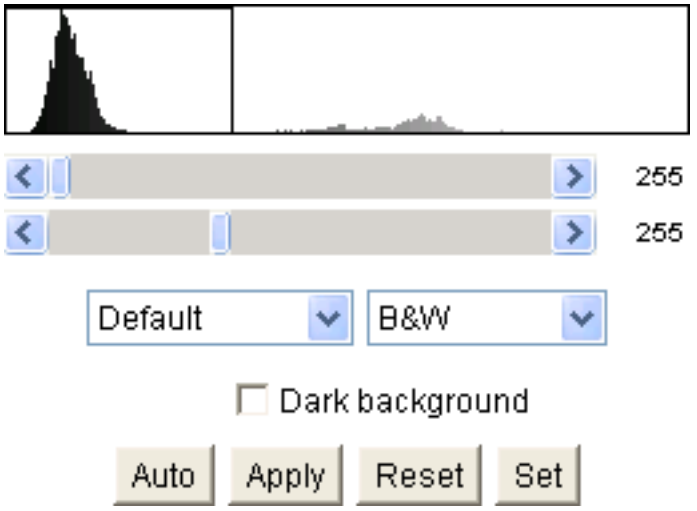


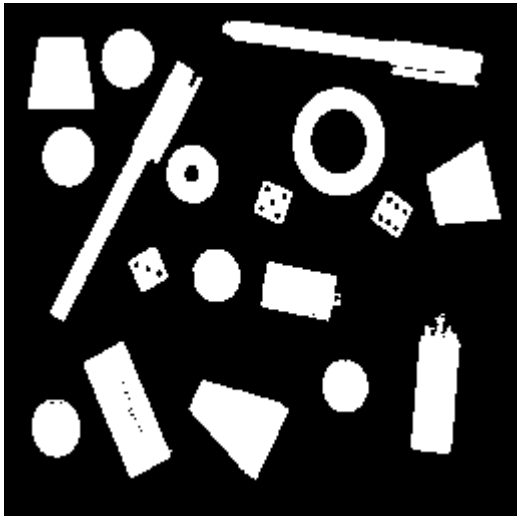
Thresholding



Count: 65536
Mean: 53.293
StdDev: 51.807

Min: 7
Max: 188
Mode: 21 (3542)

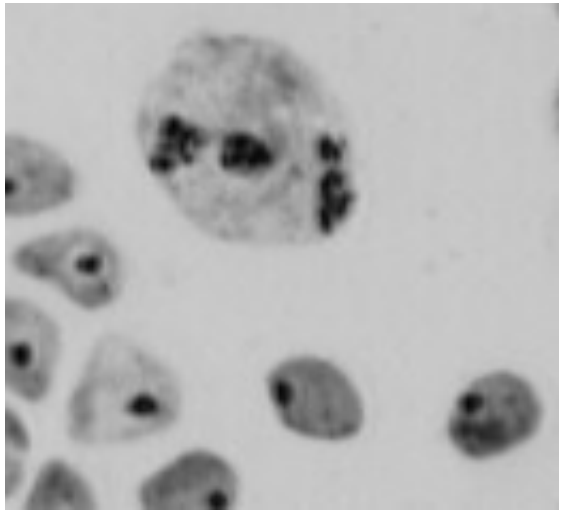


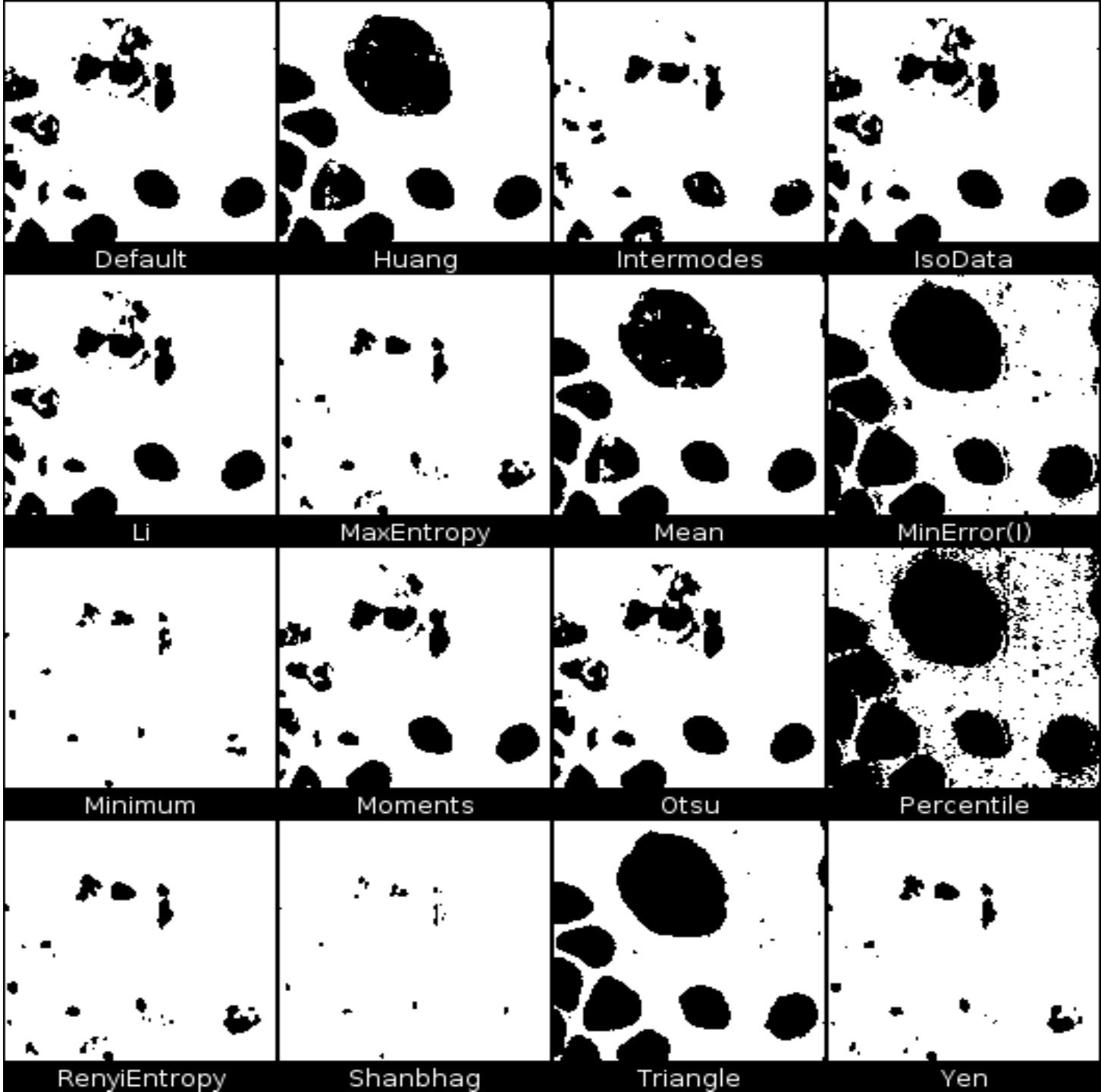


Count: 50190
Mean: 255
StdDev: 0

Min: 255
Max: 255
Mode: 255 (50190)

Choice of threshold ?





Operations on images with arithmetic calculus

Adding 2 images: C= A + B

- $f_C(i,j) = \{f_A(i,j) + f_B(i,j)\}/2$ Divided by 2 → intensity between 0-255
- Or $f_C(i,j) = \text{Min}\{f_A(i,j) + f_B(i,j) ; 255\}$

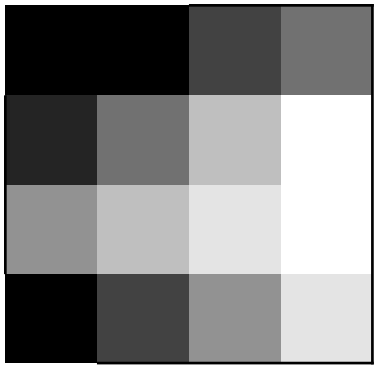
In order to

- Reduce noise in a stack of images
- Increase luminance

Operations on images with arithmetic calculus

Negative

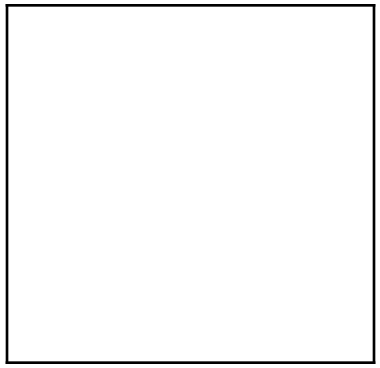
A



0	1	2	3
1	3	5	7
4	5	6	7
0	2	4	6

000	001	010	011
001	011	101	111
100	101	110	111
000	010	100	110

U_M

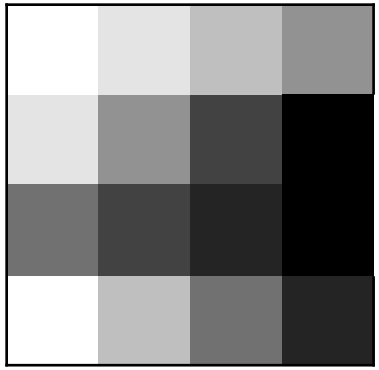


7	7	7	7
7	7	7	7
7	7	7	7
7	7	7	7

111	111	111	111
111	111	111	111
111	111	111	111
111	111	111	111

Calculate: U_M-A

U_M-A= $\overline{A}$



7	6	5	4
6	4	2	0
3	2	1	0
7	5	3	1

111	110	101	100
110	100	010	000
011	010	001	000
111	101	011	001

Operations on images with arithmetic calculus

Subtracking 2 images: C = A – B

- C=A-B = {A+(I-B)}/2 = (A-B)/2 + I/2
- $f_C(i,j) = \{f_A(i,j) - f_B(i,j)\}/2 + 128$
- $f_C(i,j) = Max(f_A(i,j)-f_B(i,j) ; 0)$

In order to
- Detect defaults
- Detect movements

f(i,j)

g(i,j)

0,5{f(i,j)+g(i,j)}

g(i,j)-f(i,j)

f(i,j)-g(i,j)

Hands on n°4

1) Load images clef.bmp & regle.bmp

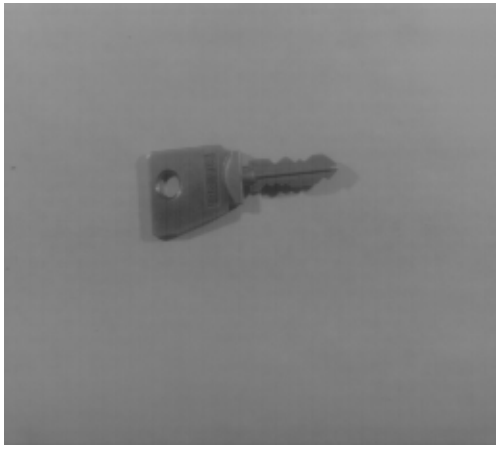
2) Add clef to regle (Process/Image calculator)

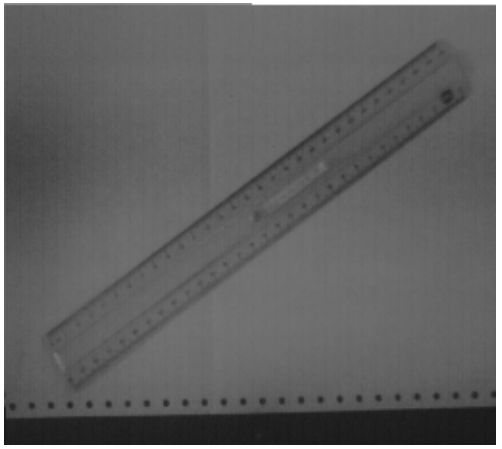
3) Display histogram of the resulting image. Comment grey-level values. Give the equation processed by ImageJ for the sum of the 2 images

4) Resize the dynamic to obtain an image between 0 and 255 (set saturated pixel to 0) (Process/Enhance contrast/Normalize ou Equalize Histogram)

5) Subtrack clef to regle

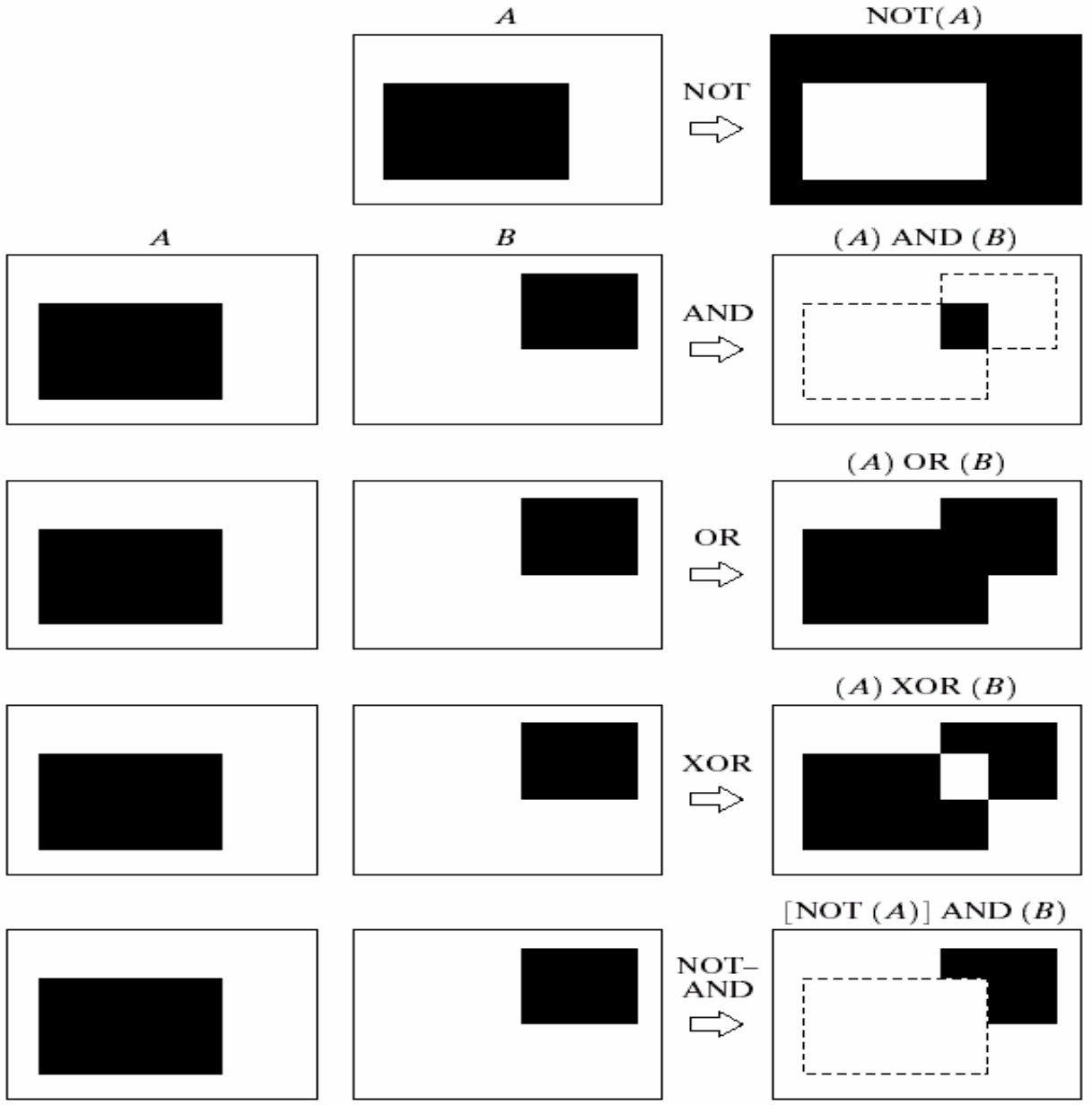
6) Display histogram of the resulting image. Comment grey-level values.



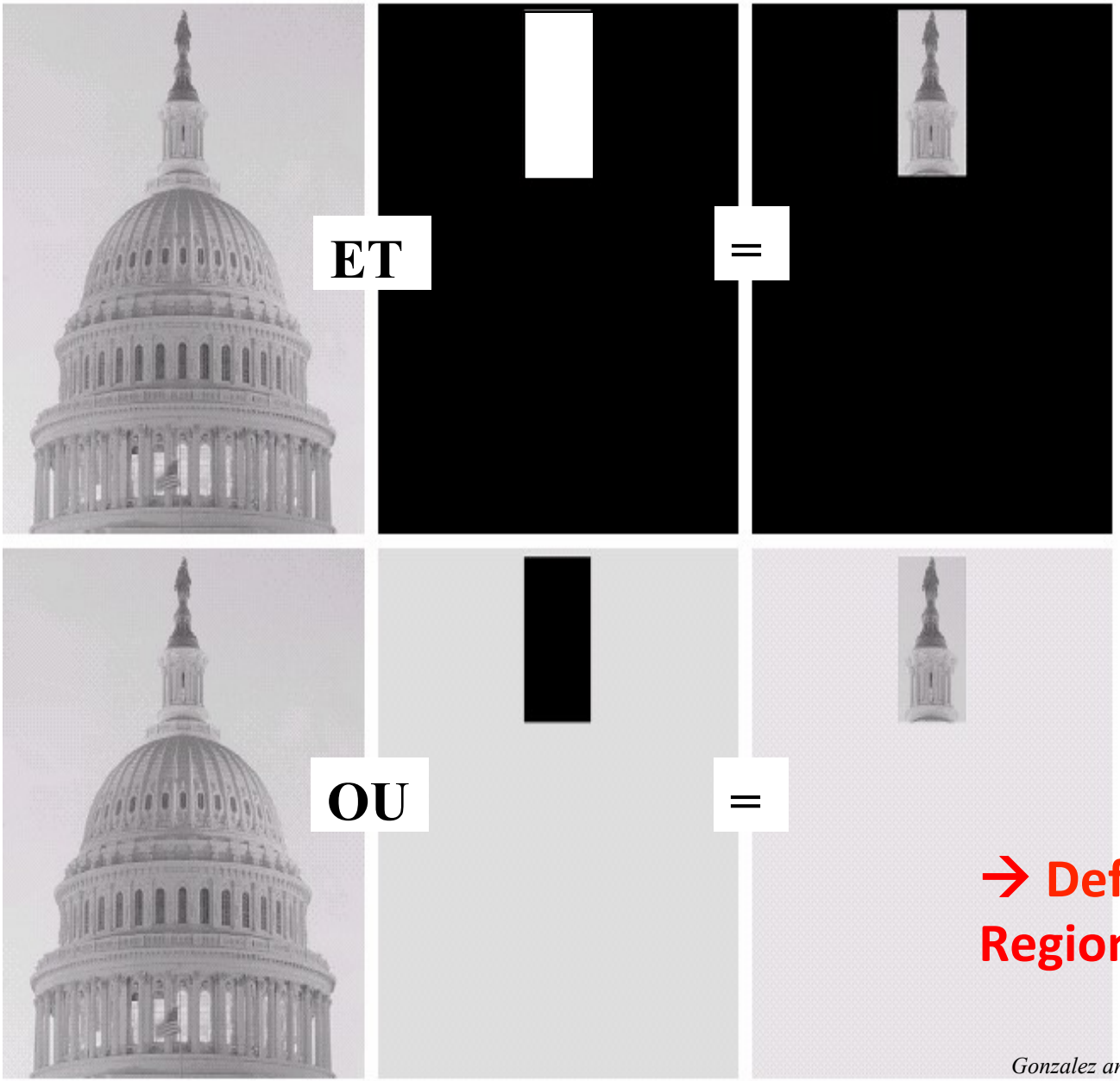


Operations on images with logic operations

Binary images



Operations on images with logic operations



a b c
d e f

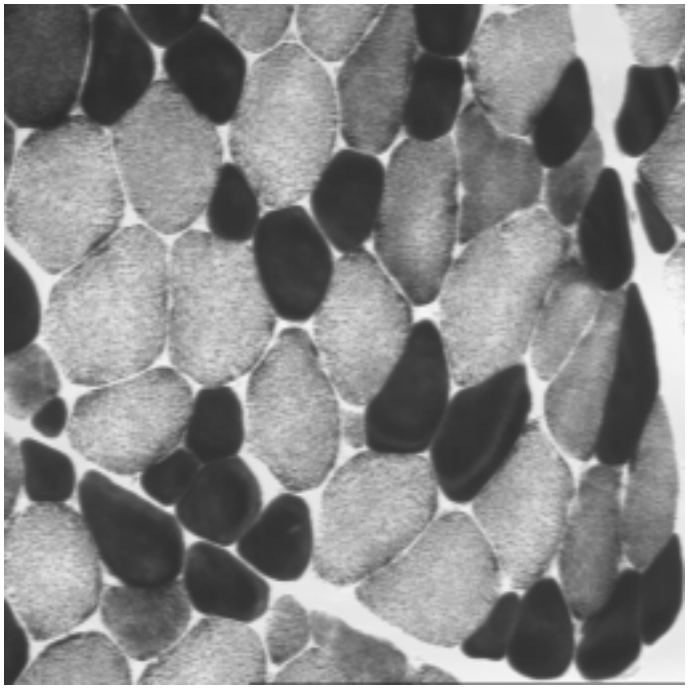
FIGURE 3.27
(a) Original image. (b) AND image mask. (c) Result of the AND operation on images (a) and (b). (d) Original image. (e) OR image mask. (f) Result of operation OR on images (d) and (e).

→ Define/highlight R.O.I
Region Of Interest

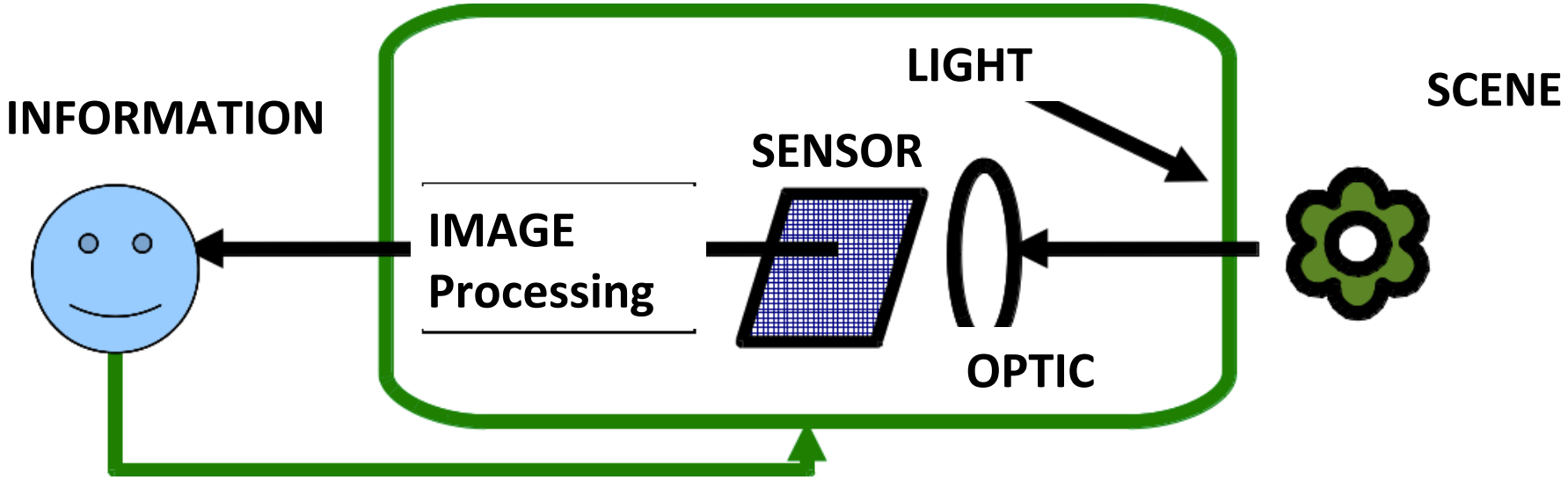
Gonzalez and Woods. Digital Image Processing. Prentice-Hall, 2002.

Hands on n°5

1) Load images muscle.bmp & spot.bmp
2) Observe gray-level on spot.bmp
3) Do operations OR, AND and XOR between the 2 images



What is noise in images ?



- We acquire images to extract the information they carry
- We call **noise** whatever is degrading the performances of the information task that motivated the image acquisition.
- An informational definition distinct from physics
- The level of noise depends on the informational tasks

Due to…

Light:

- Non uniformity → **slow variations of intensity**
- Shadows on objects sur l'objet → **fast variations of intensit**

Light properties:

- Light flow is not continuous (particles – photons)
- Poisson noise when weak flow of light

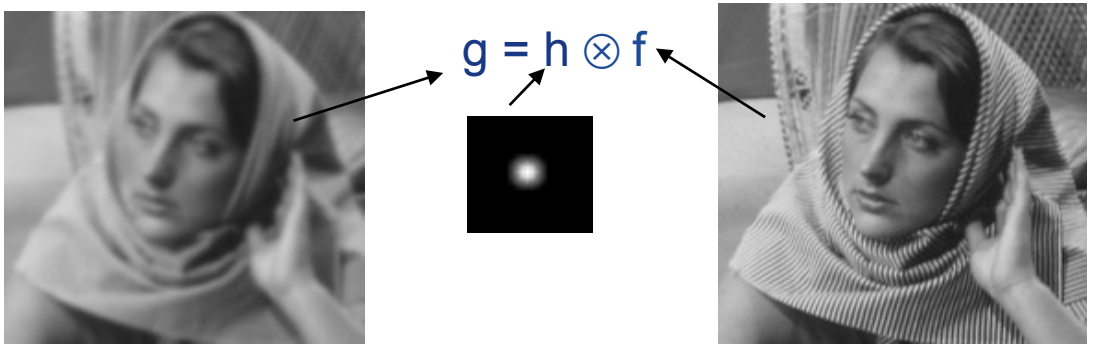
Objects can move. If motion time > integrated time of the camera → blurred image

Optic:

Lens have aberrations

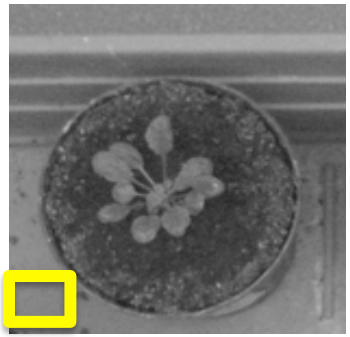
Optical filtering of fine details

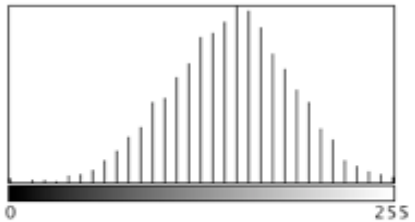
→ blurred image



Due to…

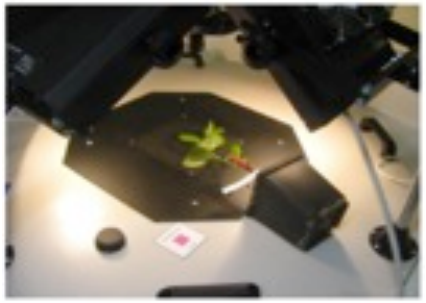
Sensor:

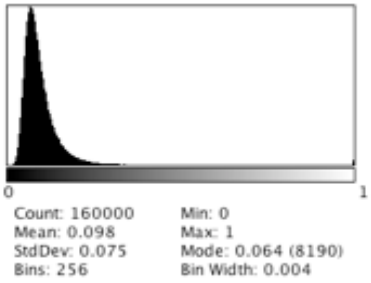




Thermal noise: independant and equally distibuted according to Gaussian law

Impulse noise eq. to salt & pepper
Some pixels are not defined





Combination of image with thermal noise (Ex.: Qymax=(Fo-Fm)/Fm
Cauchy noise

Quantification:

Noise with nniform distribution

8 bits (256 levels)

4 bits (16 levels)

2 bits (4 levels)

Now let's try to denoise !

Let's assume we have a thermal noise, i.e. Gaussian noise.
Optimal statistic (in the sense of the of the mean squares) is averaging.

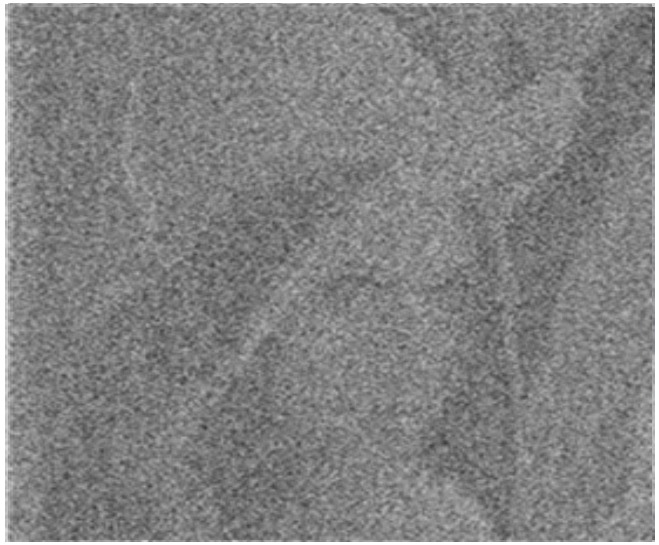
A panel of techniques:

- Filtering by temporal averaging
- Filtering by spatial linear averaging (convolution, Fourier)
- Filtering by spatial nonlinear processing

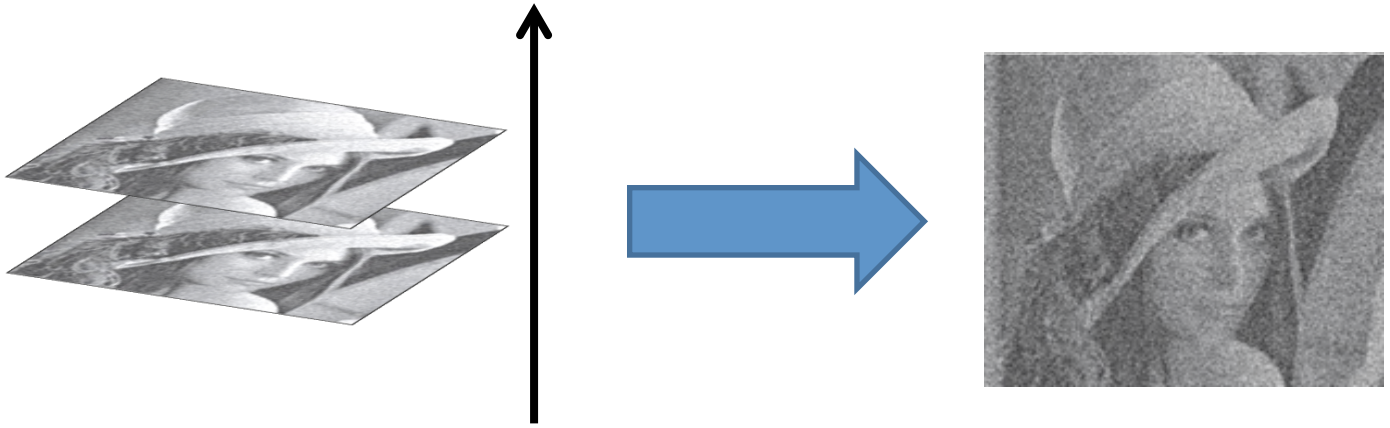
- And so on: anisotropic filtering, non local mean, mean Shift, rolling ball, top hat, bilateral filtering,...

Temporal denoising

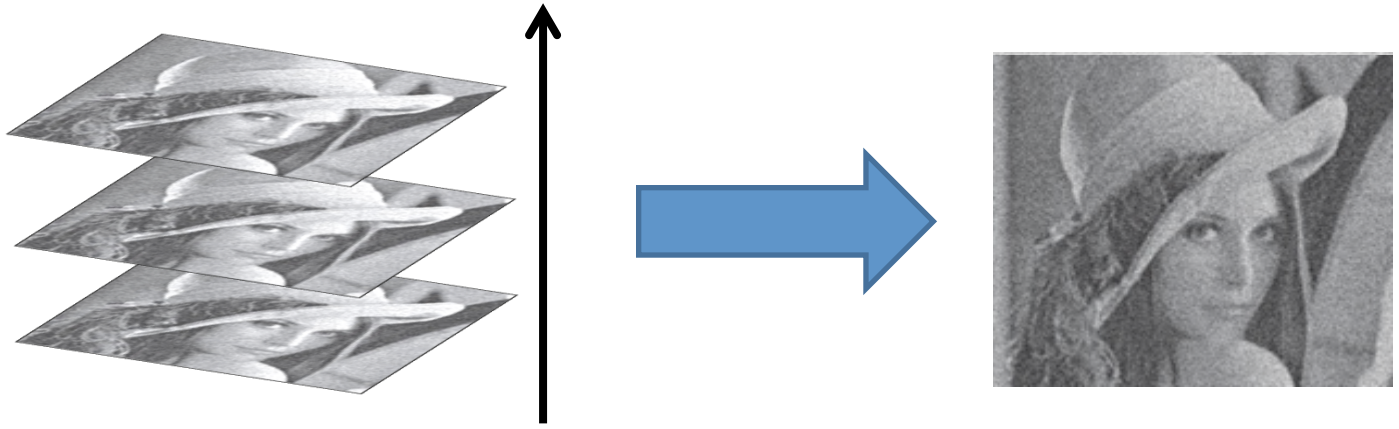
- We assume:
  - A static scene
  - Possibility to acquire multiple images at different times



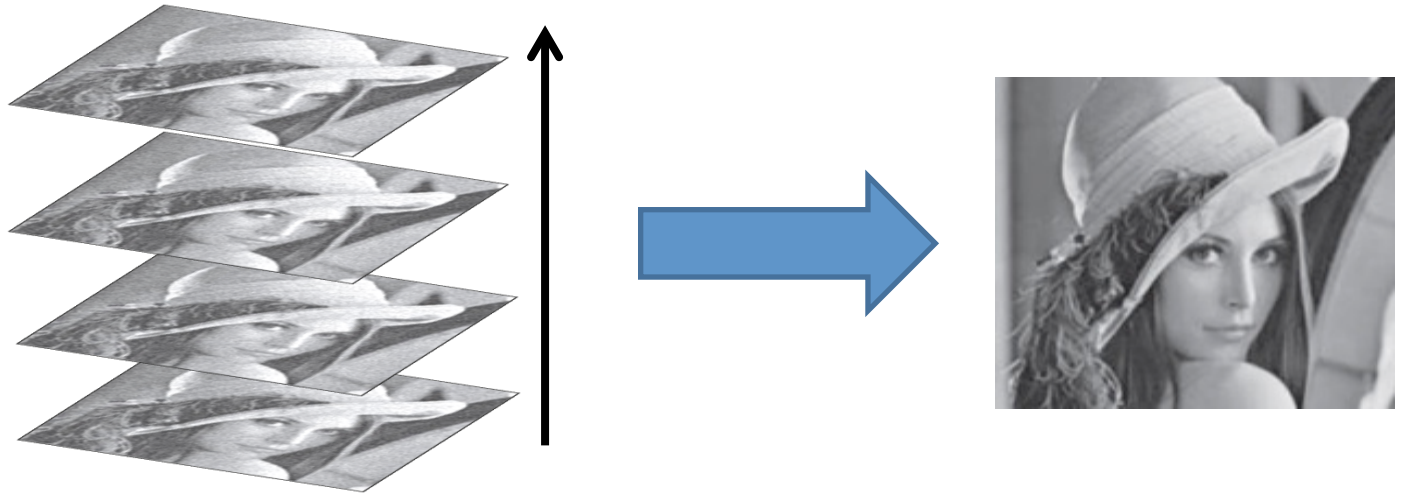
N=2 images



N=3 images



N=4 images



Spatial denoising

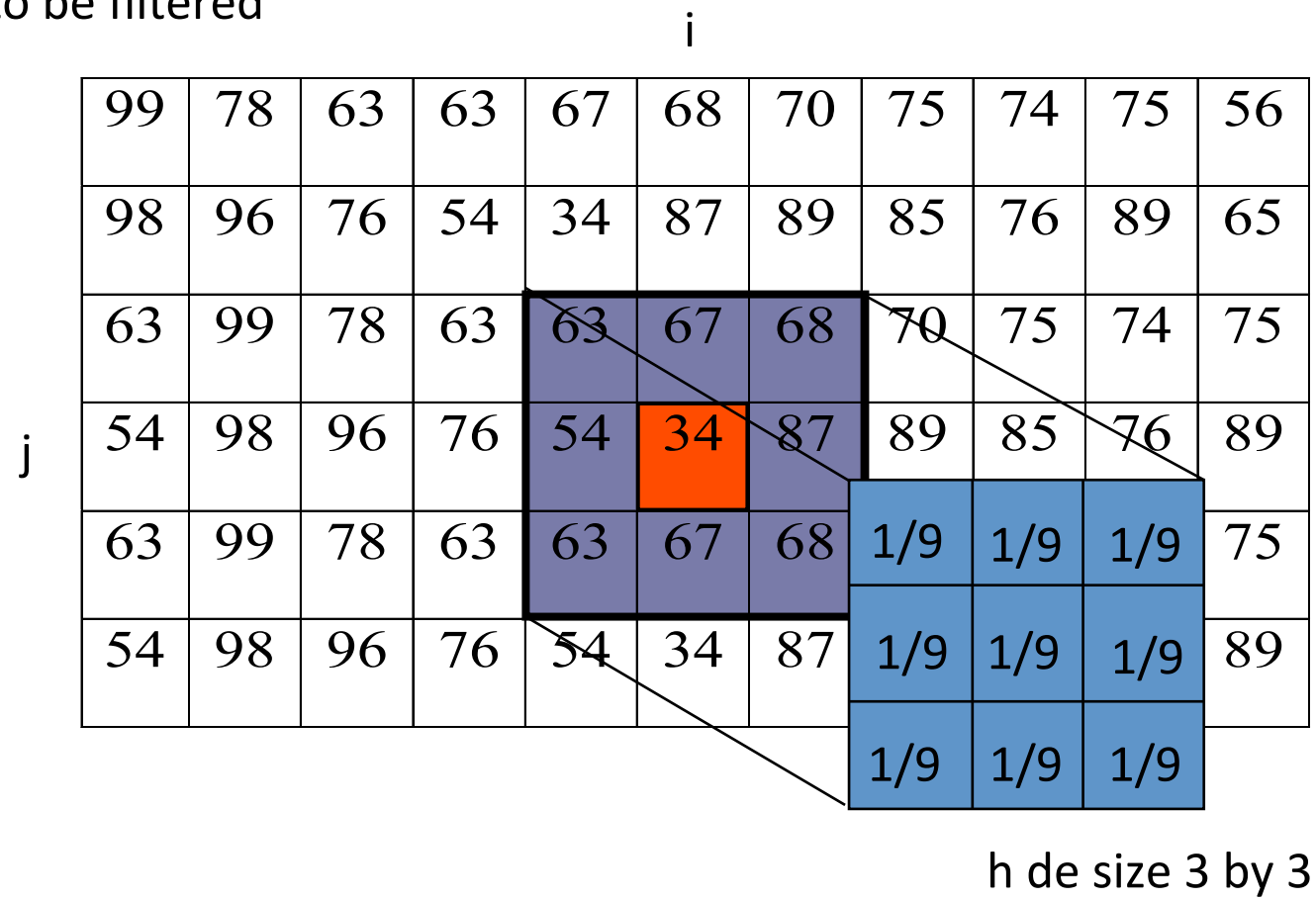
- If temporal denoising non accessible, we assume objects in images are piecewise constant
- Neighbors pixels are highly correlated
- Possibility to denoise locally

Convolution

$$g(i,j) = \sum_{(k,l)\in W} \sum h(k,l)\, f(i-k, j-l)$$

- f image to denoise, g denoised image
- h coefficients of the filter, non zero only on the neighborhood W
- A pixel f(i,j) is replaced by its weighted sum on the neighborhood
- This is realized on the entire image

f image to be filtered



Compute g(i,j) the denoised version of pixel f(i,j)

g(i,j) = (63x1 + 67x1 + 68x1 + 54x1 + 34x1 + 87x1 + 63x1 + 67x1 + 67x1) / 9

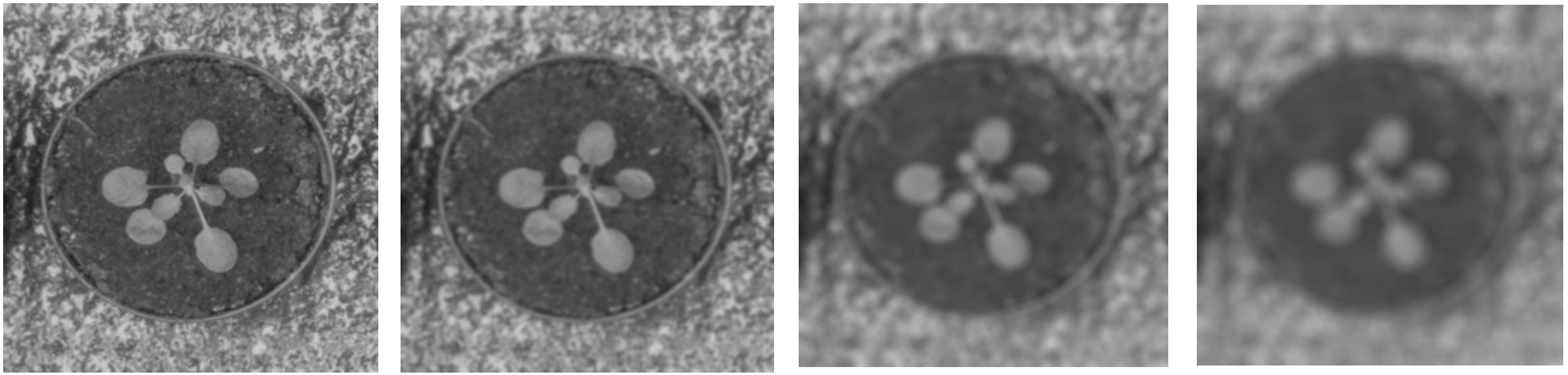
In ImageJ : Process-> Convolve

NB : for pixels edging images that can be calculated :

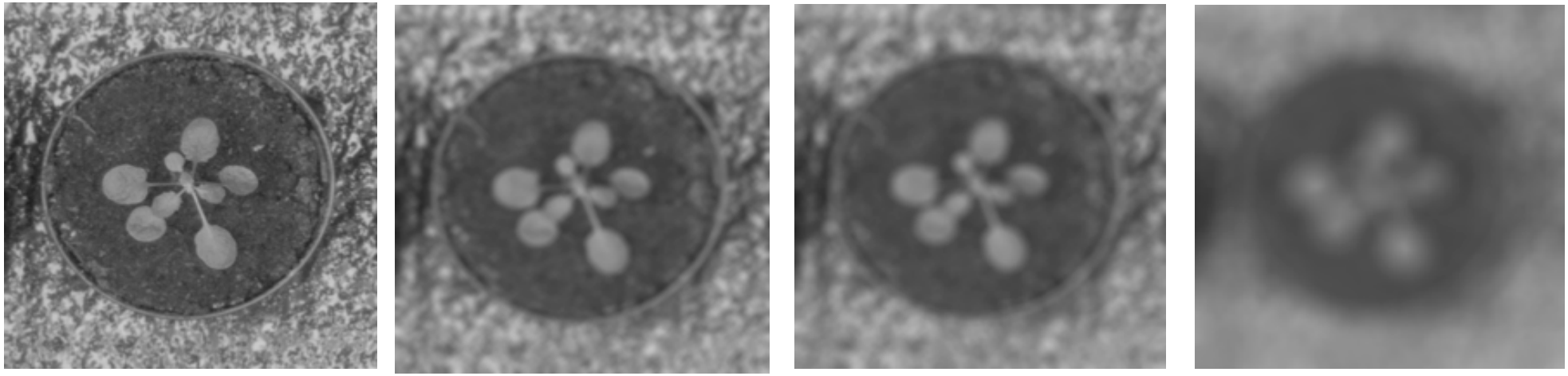
- Keep initial value
- Give value of the closest neighboored pixel calculated

Influence of the kernel size W of filter h

Original 530*565 W=3*3 W=5*5 W=9*9



Original 265*282

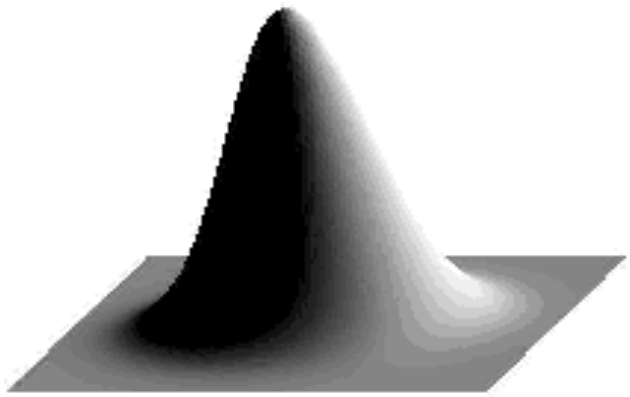


Choose the filter size smaller than the size of the structure to be preserved.

Beyond averaging

- Instead of giving the same weight to each pixel, it is possible to do a weighted averaging with the weight decreasing when we depart from the center of the window (for instance a Gaussian mask).

$$o(x,y) = I(x,y) * \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{\left(x^2+y^2\right)}{2\sigma^2}}$$



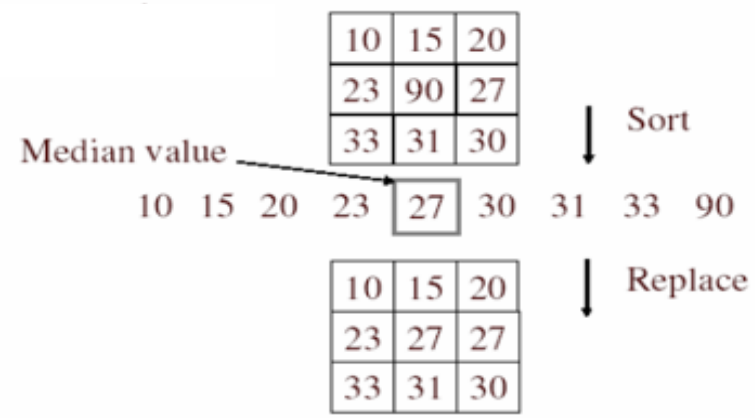
Parameter sigma

$x^2 + y^2 = r^2$ Filter with circular symetry

In ImageJ : Process-> Filter-> Gaussian blur

Non linear filtering

- If noise is non Gaussian, the optimal statistics is not the mean.
- For instance with impulsional noise, the median is more efficient than the mean.
- The central pixel is replaced by the median of its neighbourhood



In ImageJ : Process-> Filter-> Median

Hands on n°6

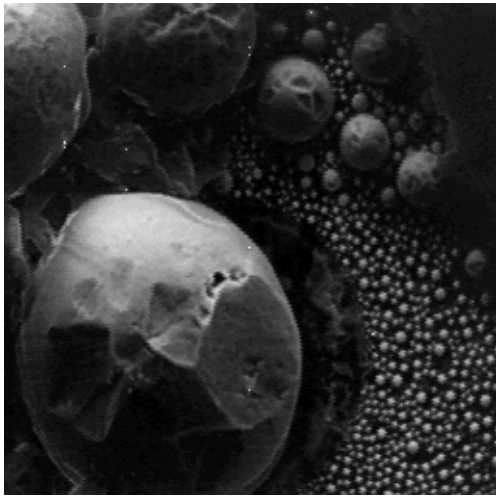
1) Load images meb.bmp and display its histogram

2) From original image (use Image/duplicate), apply different filters (Process/smooth, Sharpen, Process/filters/mean and gaussian) using a radius of 1.5.

3) Display all histograms.

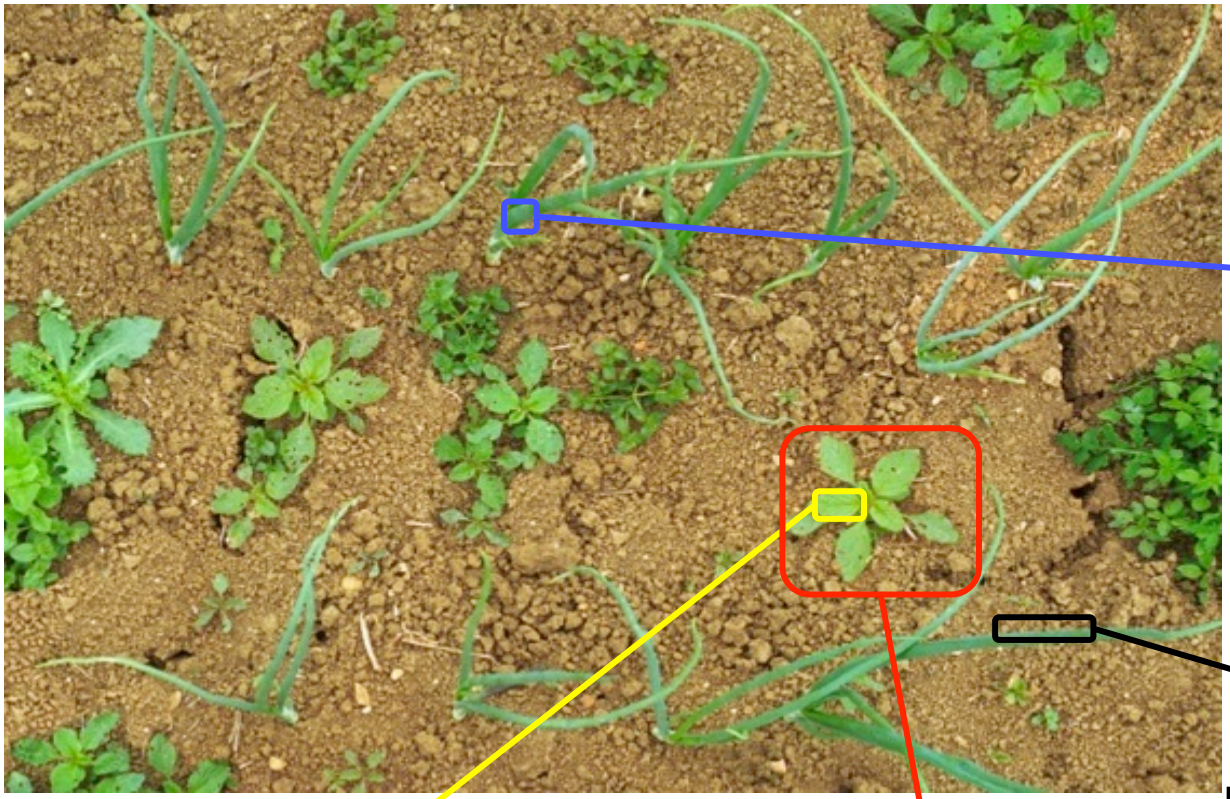
4) To compare results, for each filtered image, calculate the difference between original and the filtered images. Explain what you obtain.

5) Evaluate the influence of the size of the radius.



6) On the same image, add some impulse noise as Salt&Pepper (Process/noise)
7) Duplicate this image
8) Compare mean and median (Process/filter/median) filters with radius = 1.5

Content in an image



Homogeneous region

Edge

Texture

Object

Tribute to F. COINTAULT

Other types of filter

Structures in images defined in terms of region and edges

A region is an area of the images homogenous from a certain point of view.

An edge is the limit between two adjacent regions

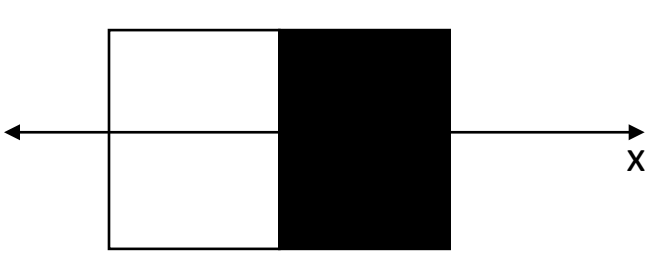
Regions

Edges

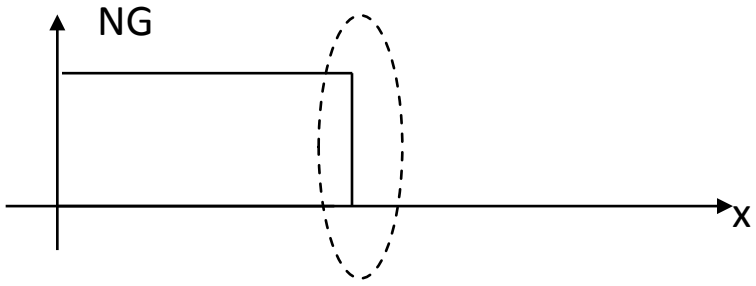
In black: region with pixels with a Gray level <128
In white : region with pixels with a Gray level >128

Detecting edges

An edge is a rapid variation from a certain point of view

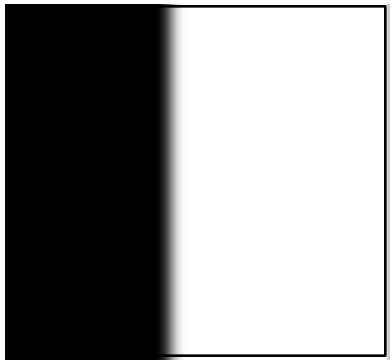


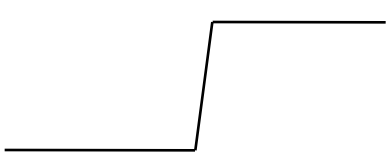
Image



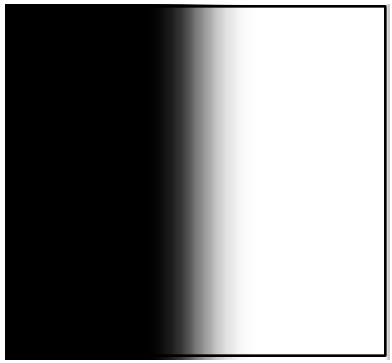
Profil axial

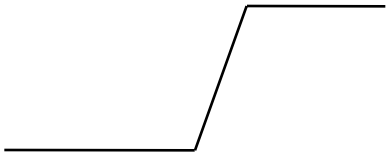
What is a rapid variation ?



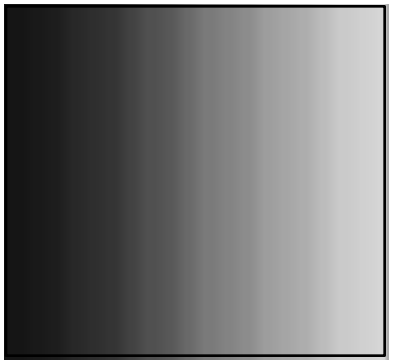


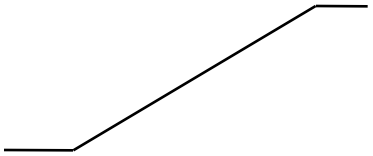
Edge

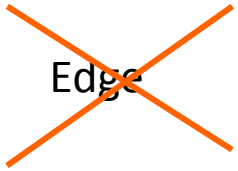




Edge?

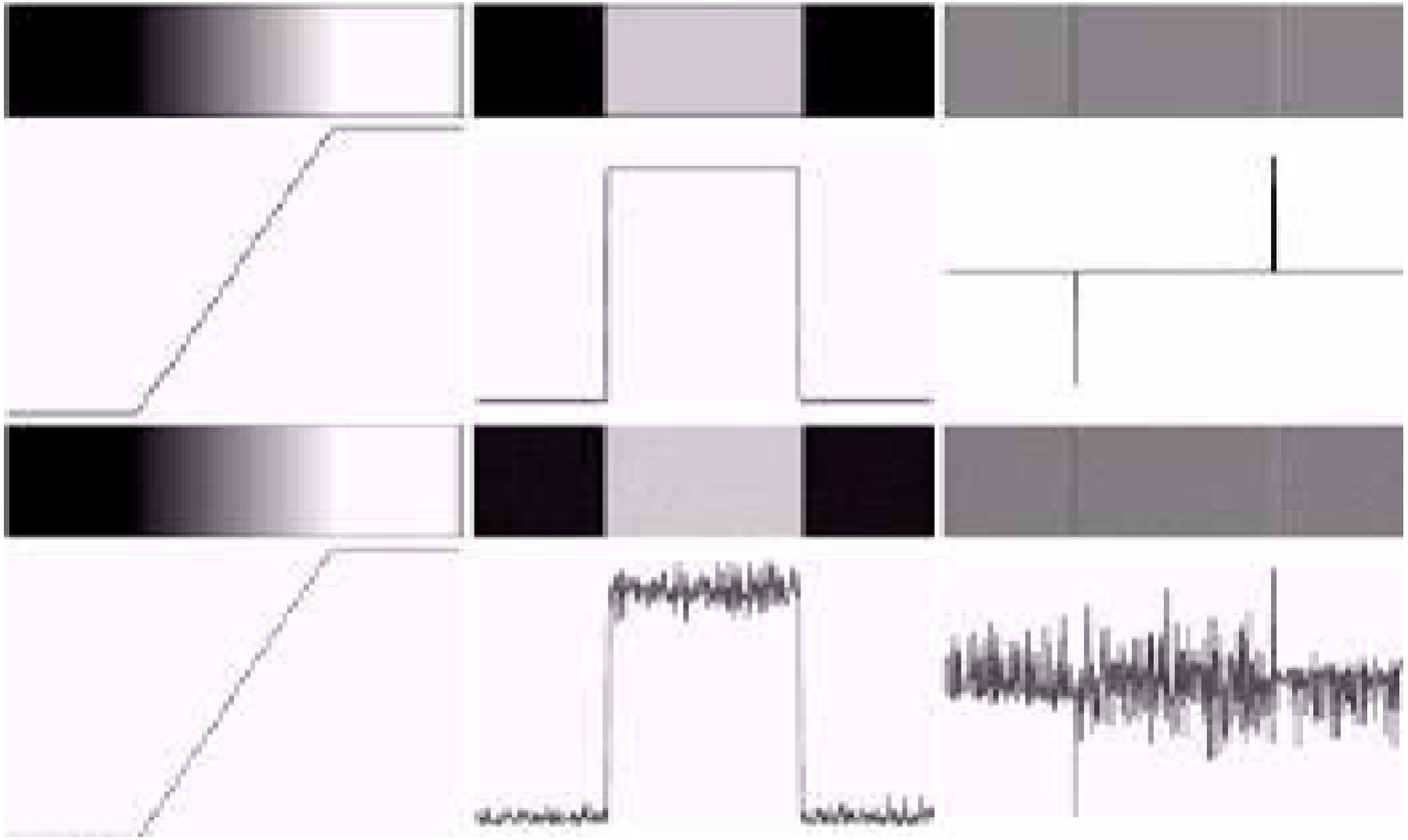




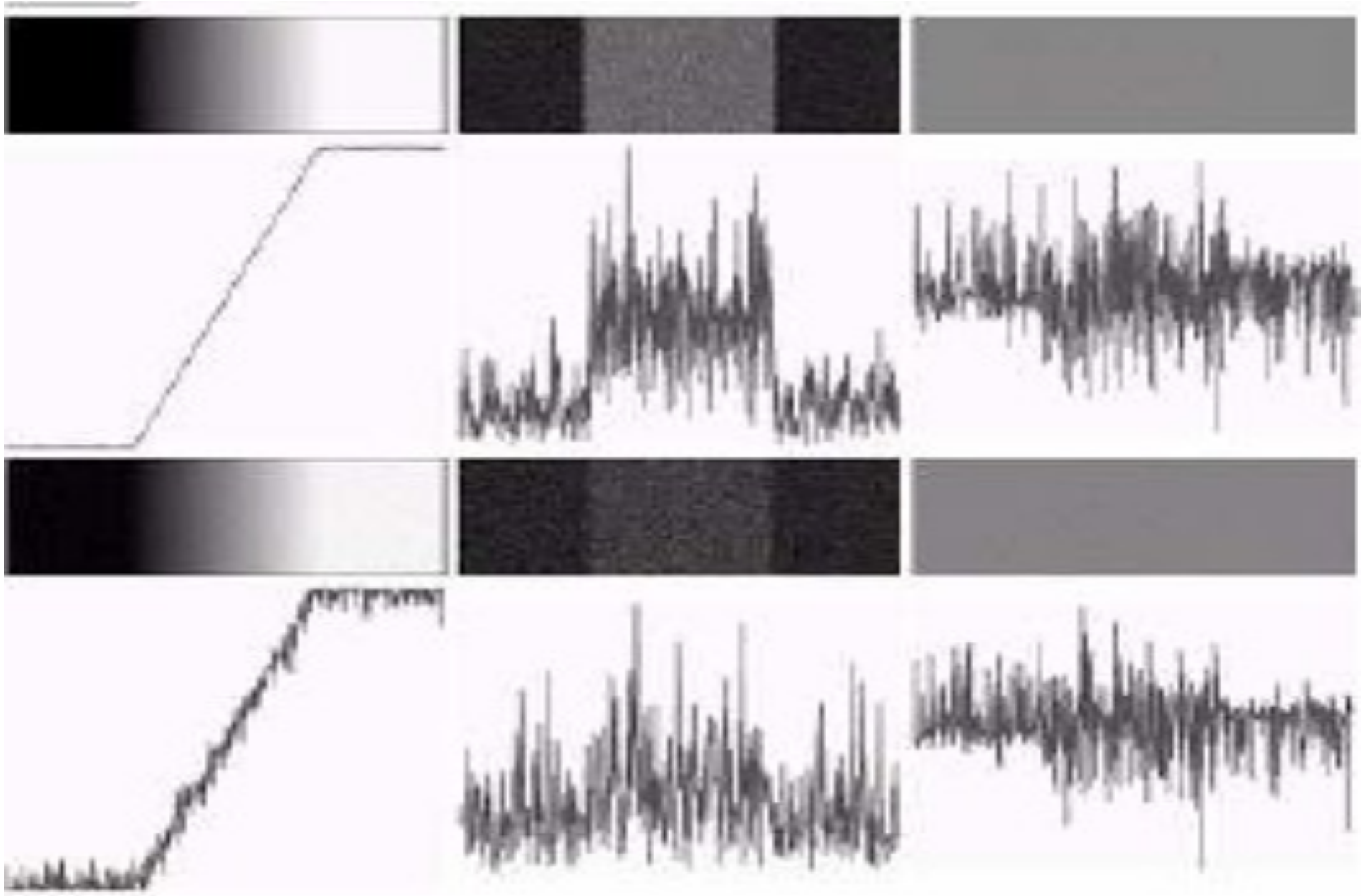


Edge

Edges with little noise



Edges with lots of noise



Gradient

Computation *Profil*

Image 1D f(x)

1ère dérivée f'(x)

|f'(x)|

Pixels contours:
|f'(x)| > Seuil

Gradient

- First derivative along x axis

$$G_x = \frac{\Delta I}{\Delta x} = \frac{I(x+\Delta x) - I(x)}{\Delta x}$$

- Approximation of discrete derivative
- Same operation along y

-1
1

-1	1

- Norm of the gradient

$$|G| = \sqrt{G_x^2 + G_y^2} \approx |G_x| + |G_y|$$

1	
-1	

gradient

1	
	-1

Roberts

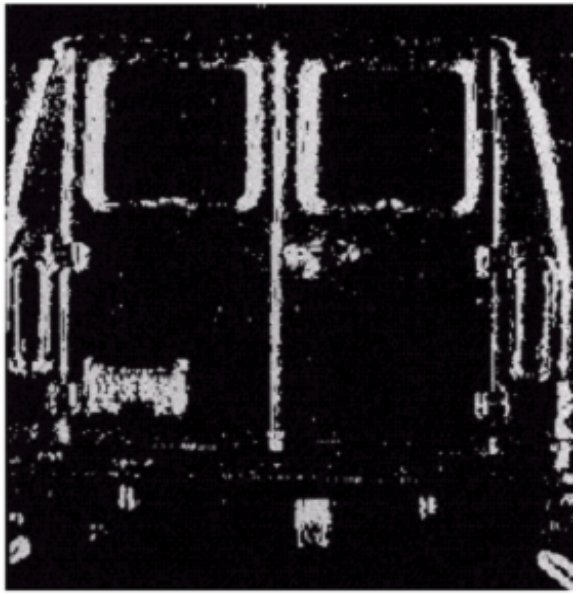
1		-1
1		-1
1		-1

Prewitt

1		-1
2		-2
1		-1

Sobel

Gradient

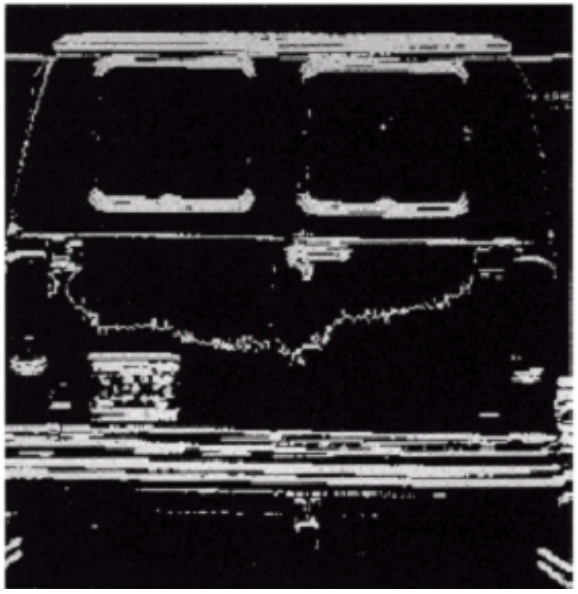


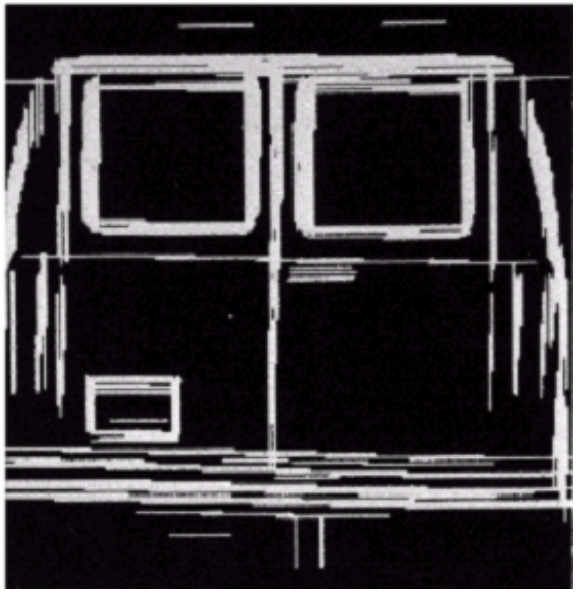
Vertical edges

$$\frac{\Delta I}{\Delta x}$$

Horizontal edges

$$\frac{\Delta I}{\Delta y}$$





Norm of gradient

$$\frac{\Delta I}{\Delta x} + \frac{\Delta I}{\Delta y}$$

Second derivative: Laplacian

- Another approach to find edges in images is to use the second derivative
- Detect zero crossings
- Laplacian

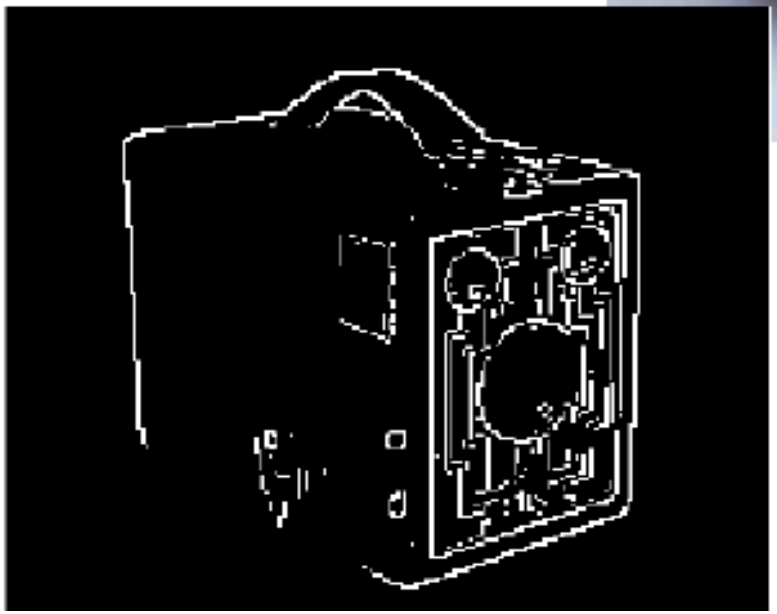
$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

- Discrete approximation

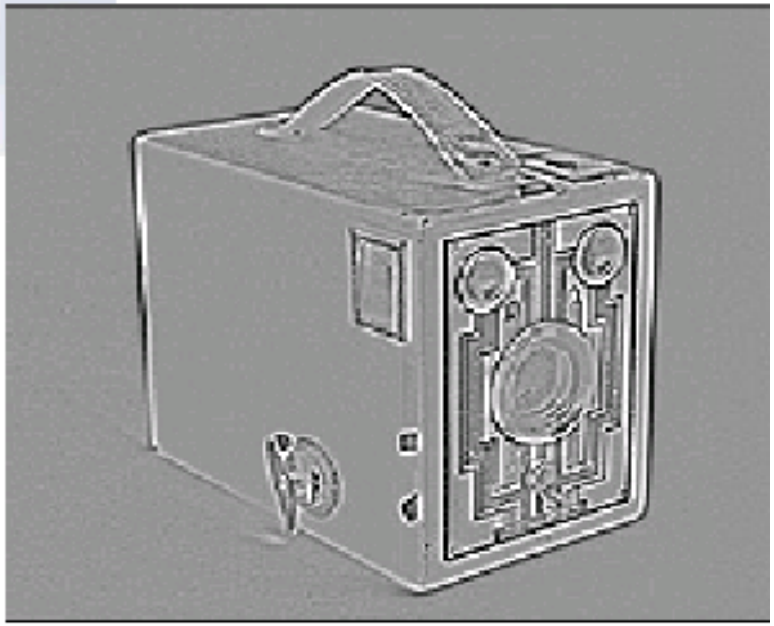
$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Comparison: gradient VS laplacian

Gradient



Laplacien



Hands on n°7

1) Load morpho.bmp

2) Detect edges with Process/Find edges

3) Duplicate image morpho.bmp

4) Add noise in morpho.bmp

5) Detect edges in the noisy image with find edges.

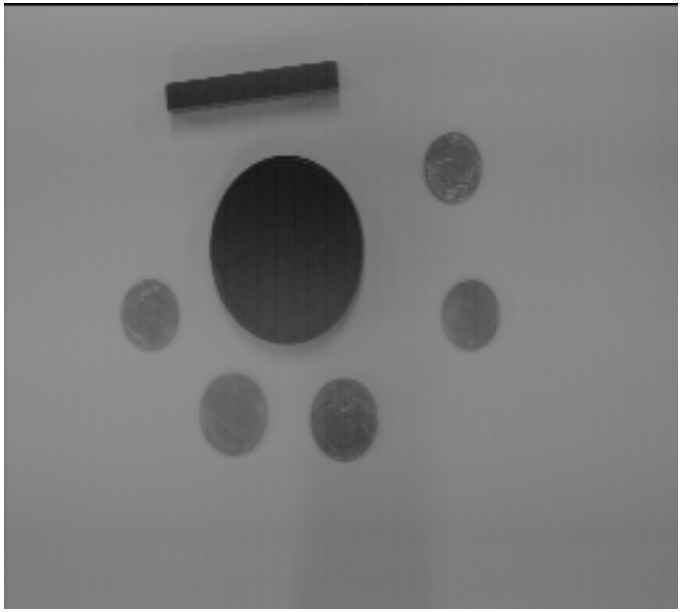
Conclusion.

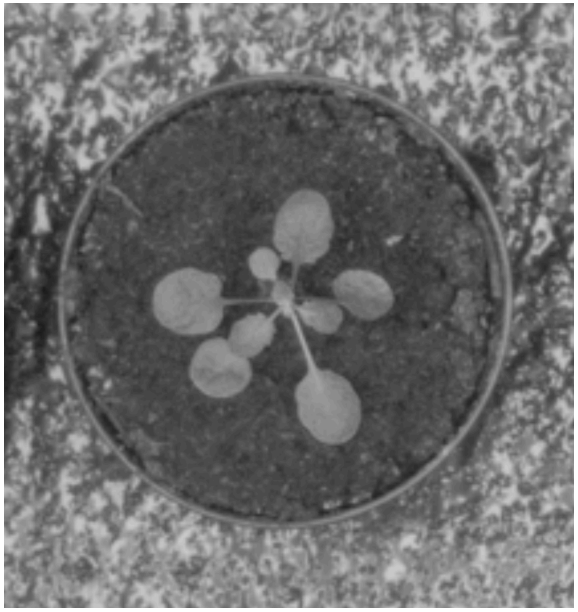
6) Apply a filter on the noisy morpho.bmp

7) Detect edges in the denoised image.

Conclusion.

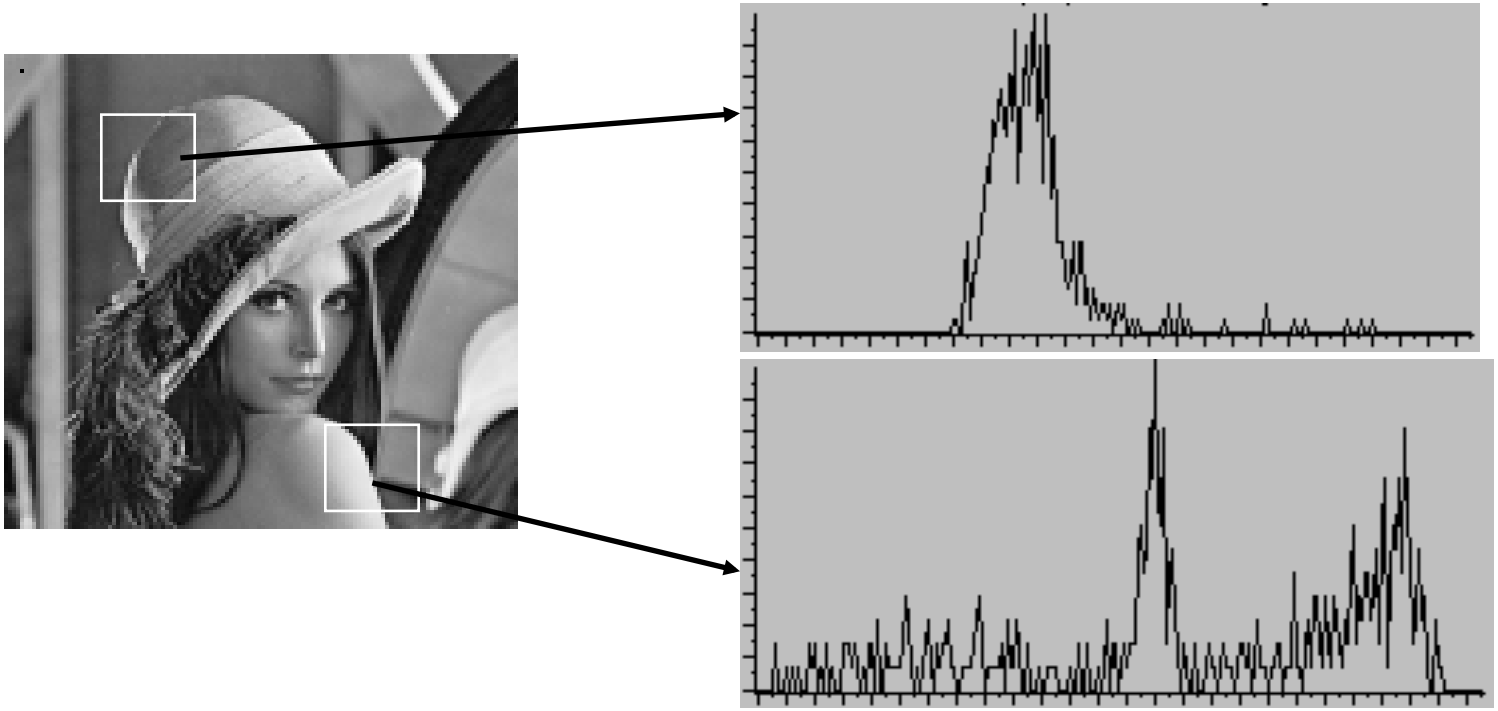
8) Do the same with Arabette2





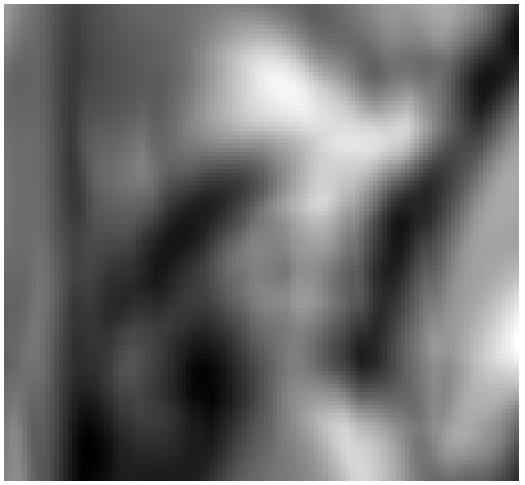
Examples of spatial features

Measures on a local scale :



Computation on a 16x16 scale

128x128 pixels



Mean

Variance

First order statistics

Satistiques du premier ordre

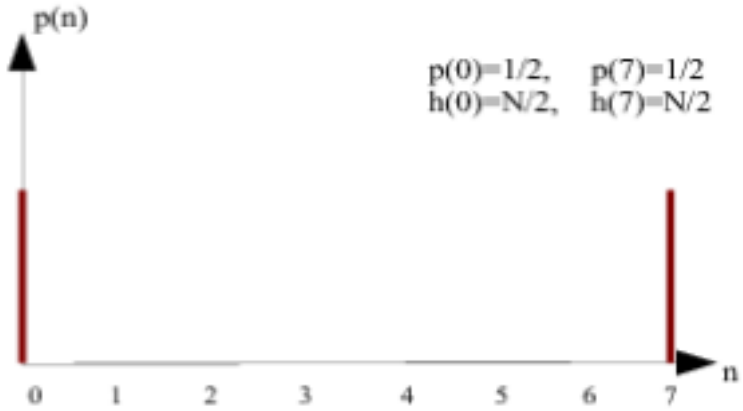
Se déduisent de la probabilité $p(n)$ du niveau de gris n ou de l'histogramme $h(n) \simeq Np(n)$, avec N le nombre de pixels de l'image.

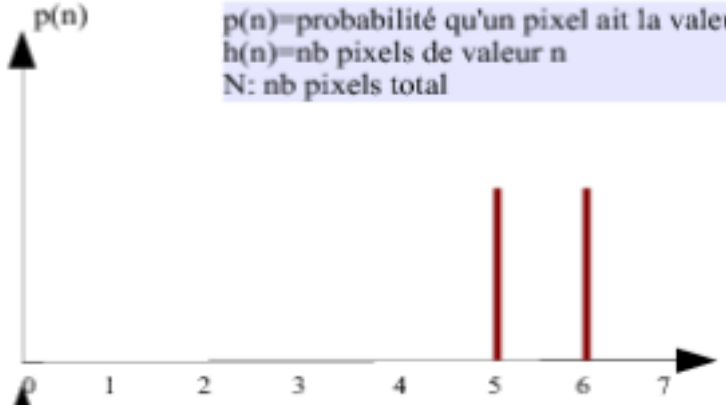
- **Moments d'ordre k** $\mu_k = \sum_n n^k p(n)$ (μ_1 : moyenne)
- **Moments centrés d'ordre k** $\eta_k = \sum_n (n - \mu_1)^k p(n)$

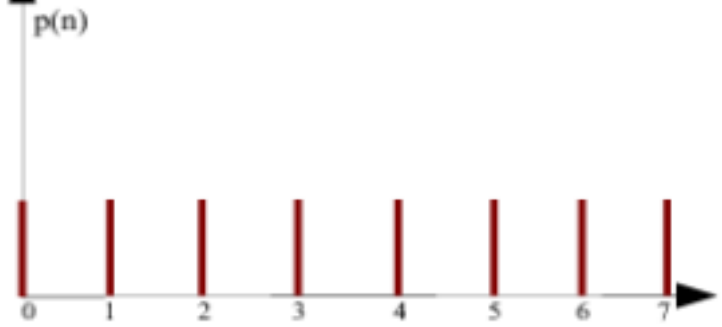
En particulier : la variance $\sigma^2 = \eta_2$, le biais $\gamma_1 = \eta_3/\sigma^3$, le kurtosis (aplatissement) $\gamma_2 = \eta_4/\sigma^4 - 3$

- **Énergie** $W = \sum_n |p(n)|^2$
- **Entropie** $E = -\sum_n p(n) \log p(n)$
- **Contraste** $C = (\max(n) - \min(n))/(\max(n) + \min(n))$
- **Dynamique** $D = \max(n) - \min(n)$
- **Coefficient de variation** $\varsigma = \mu/\sigma$
- **Dimension fractale**

Example of first order statistics

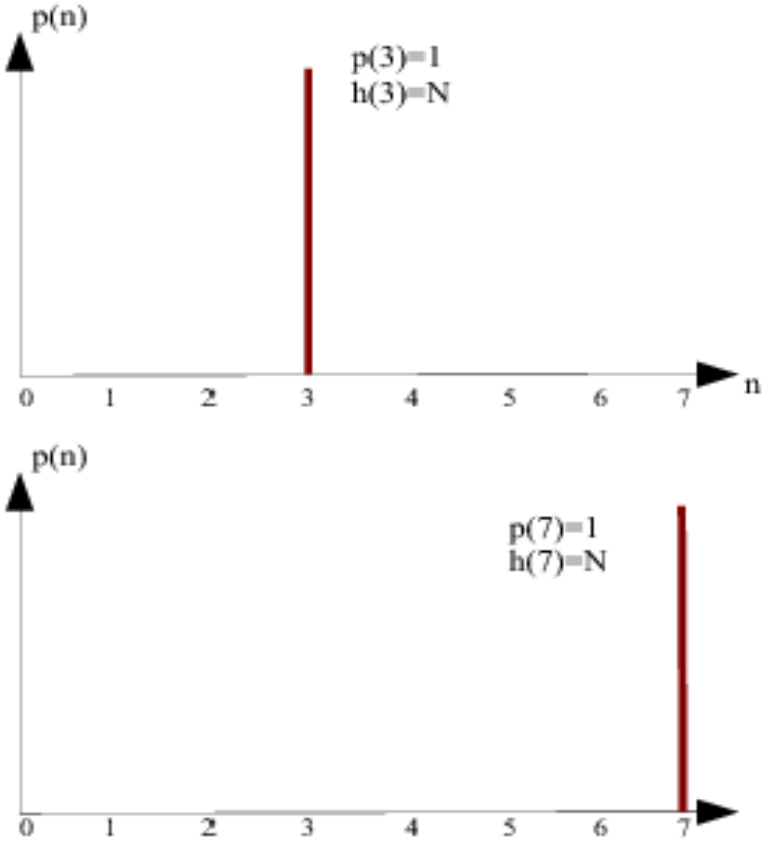






μ1	W	C	D	E	η1	η2	η3
7/2	1/ 2	1	7	-log(1/2) 0,30	0	12,25	0
11/2	1/ 2	1/11	1	-log(1/2) 0,30	0	0,25	0
+ les niveaux de gris sont similaires, plus η2 est faible, plus la dynamique et le contraste sont faibles							
7/ 2	1/ 8	1	7	-log(1/8) 0,90	0	5,25	0
+ il y a de niveaux de gris différents dans la texture + l'entropie est élevée							

Example of first order statistics



μ1	W	C	D	E	η1	η2	η3
3	1	0	0	0	0	0	0
7	1	0	0	0	0	0	0

2 textures homogènes ne se caractérisent que par leur moyenne

Illustration

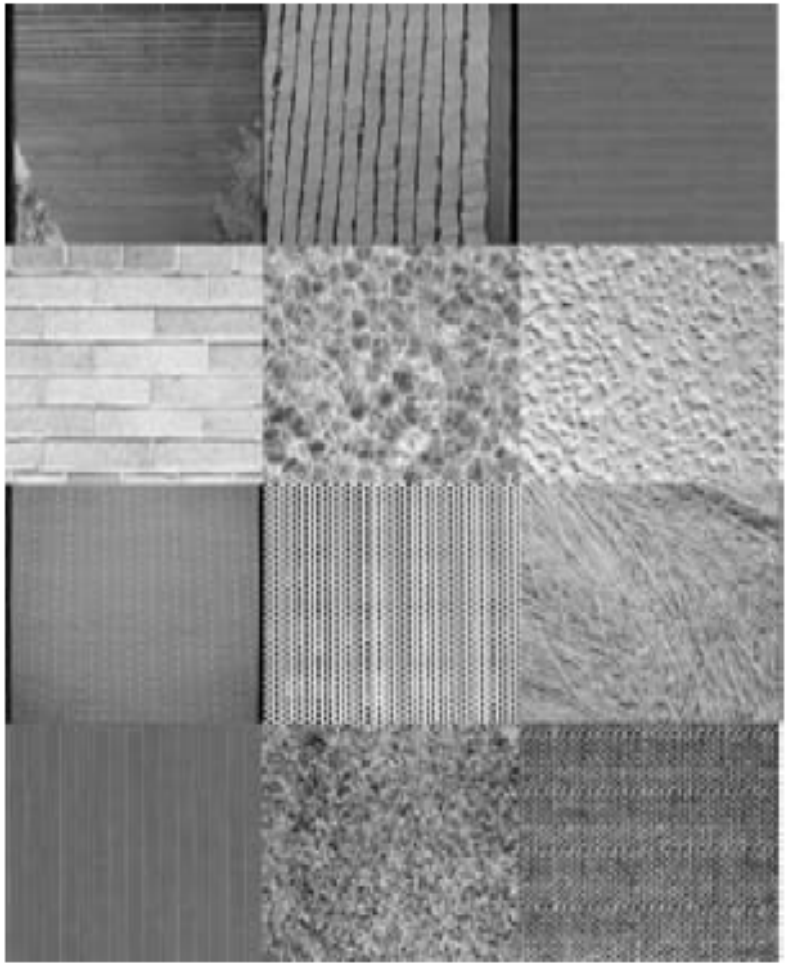
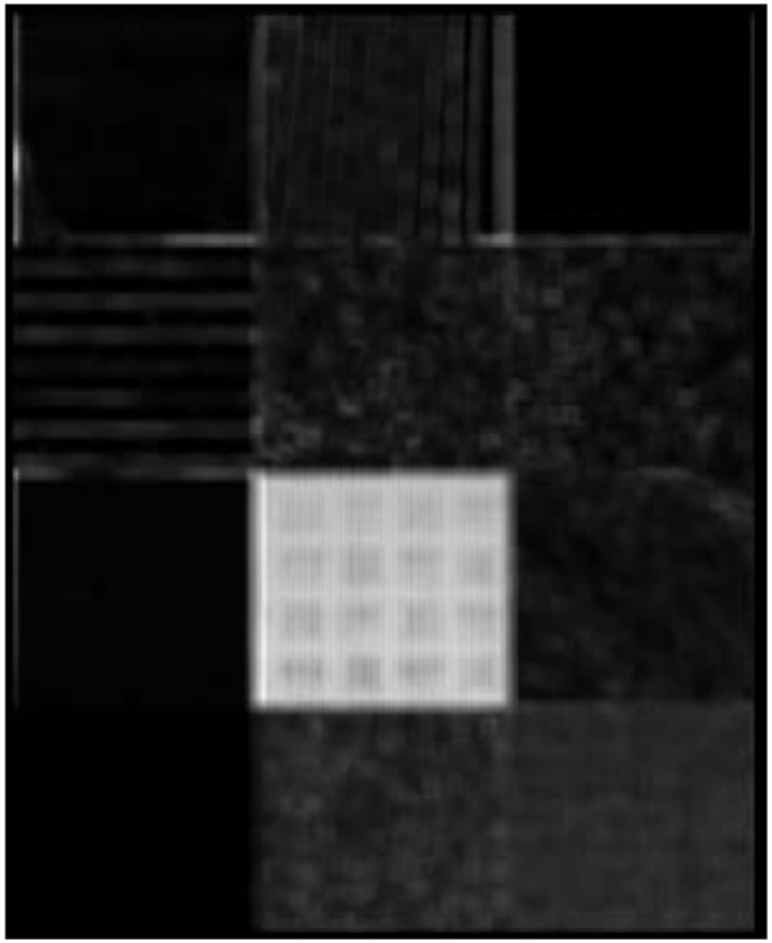


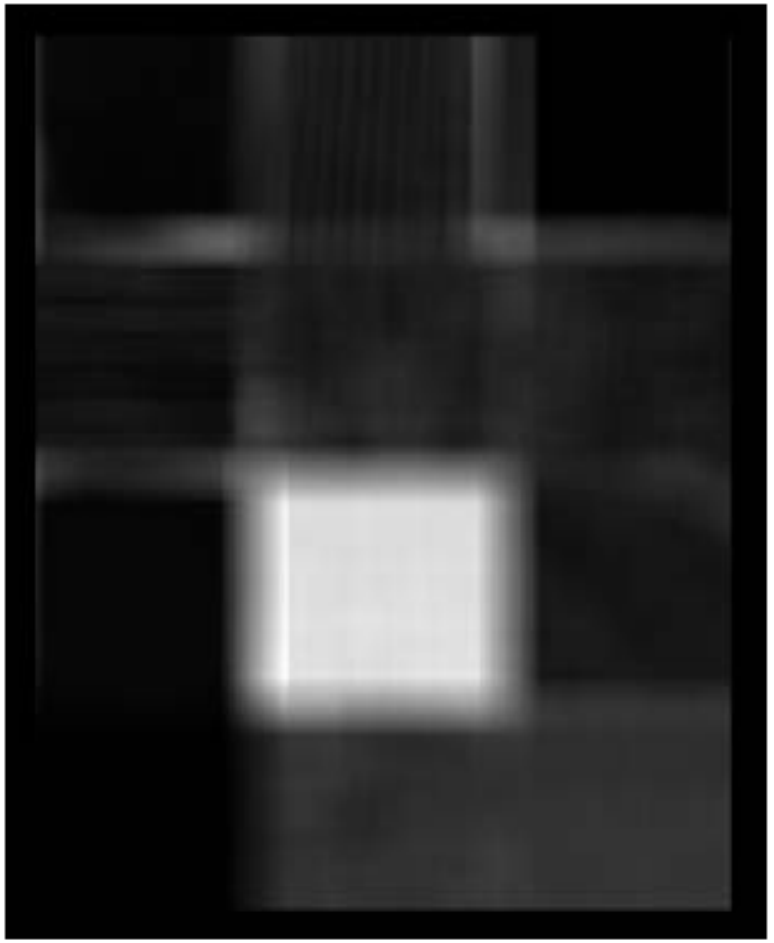
Image de résolution 1600×800

Soit W la taille de fenêtre : $W = 15$ et $W = 50$ pour les exemples.

Variance



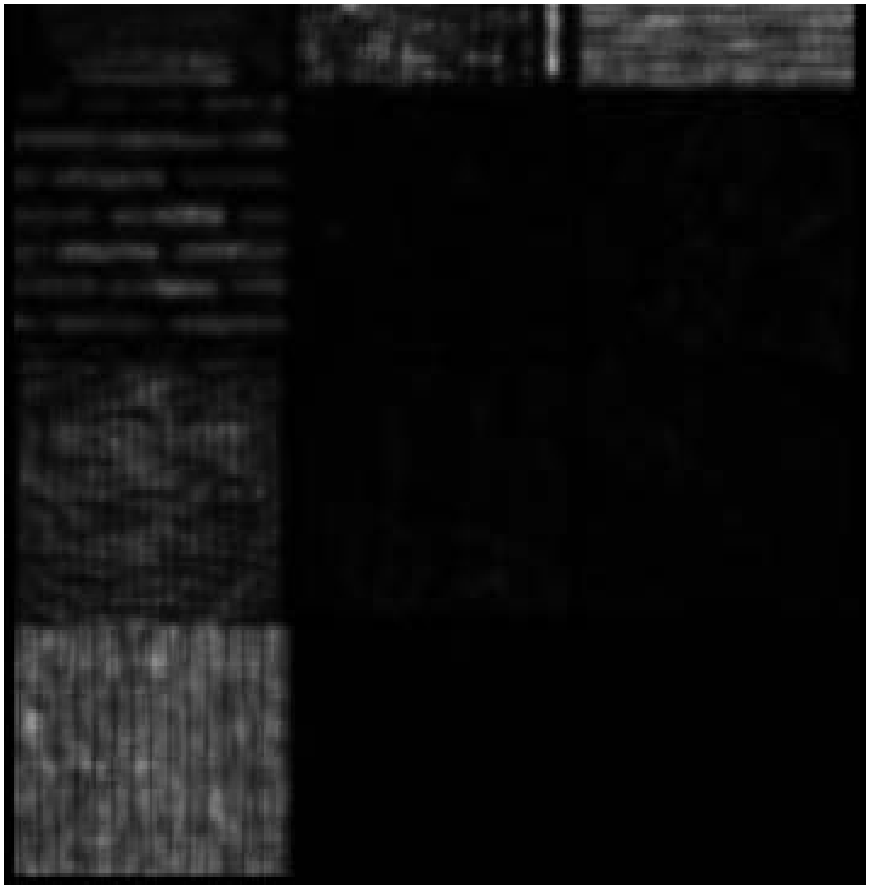
W=15



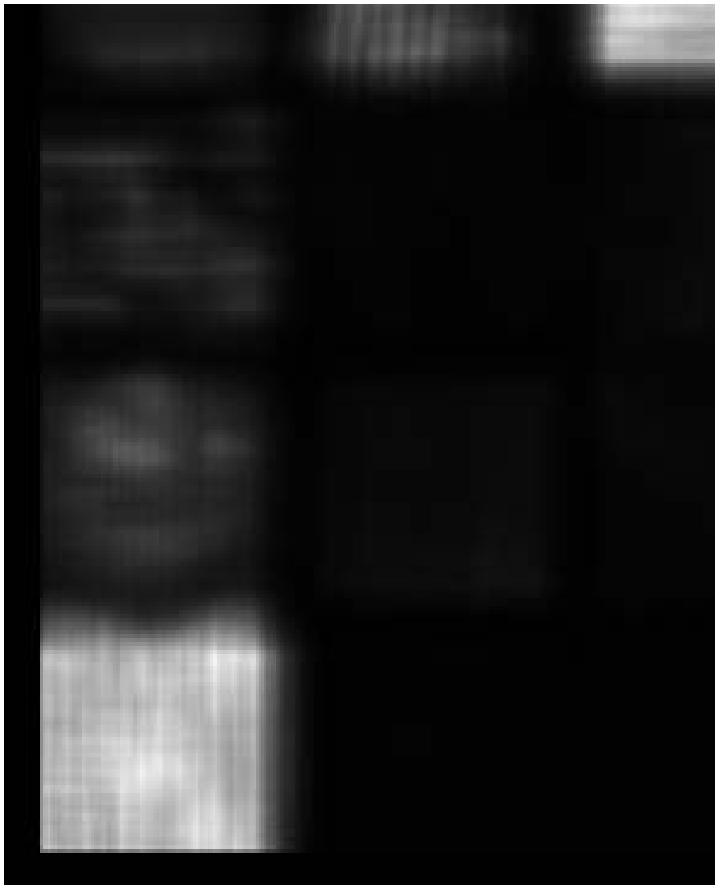
W=50

Variance : mesure l'ampleur des variations d'intensité par rapport à la moyenne.
Moment centré d'ordre 3 = 0 si les intensités sont équitablement réparties autour de la moyenne (histogramme symétrique autour de la moyenne)

Energy



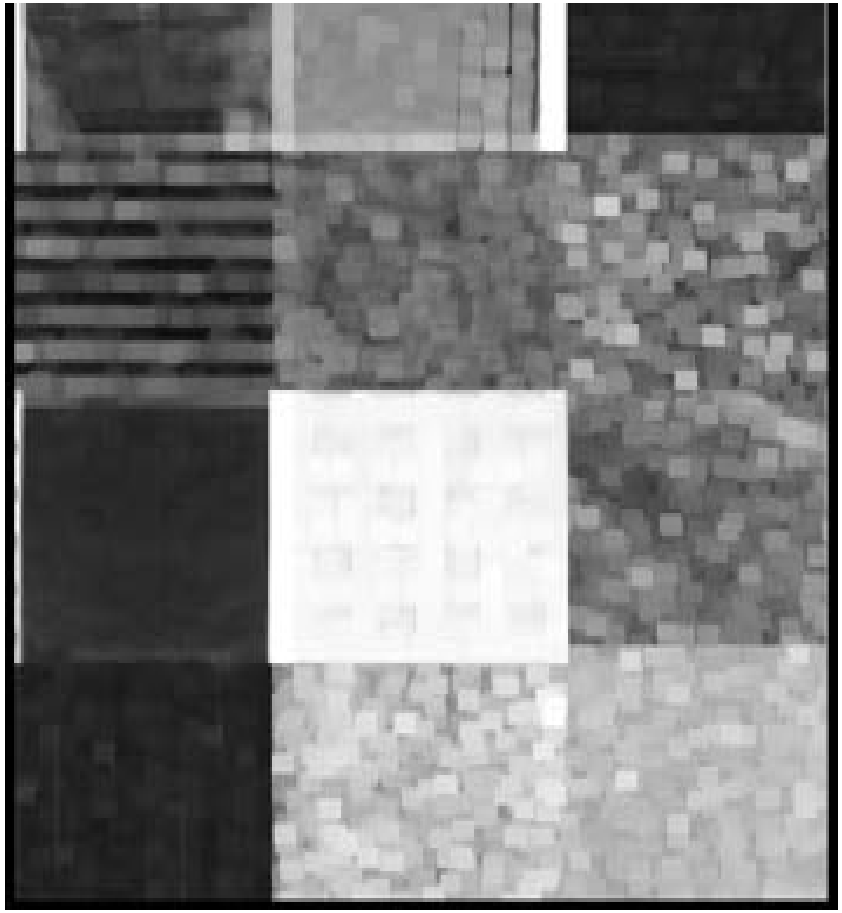
W=15



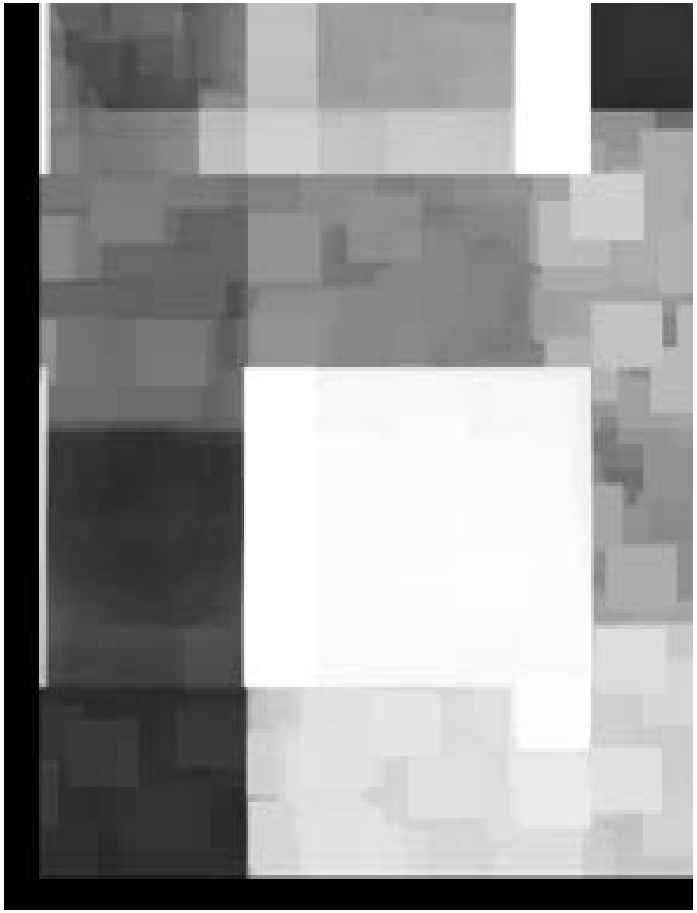
W=50

Énergie : faible pour des probabilités équiréparties.

Contrast

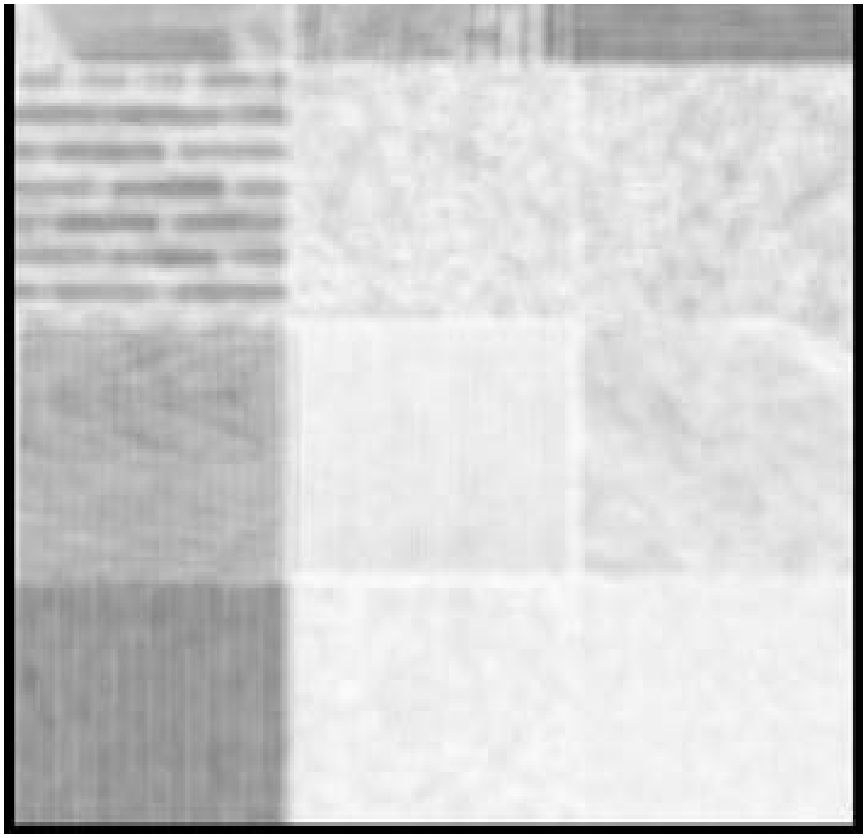


W=15



W=50

Entropy



W=15

W=50

Faible entropie pour une image d'intensité homogène.

Forte entropie lorsque l'histogramme est étiré.

Mathematical morphology

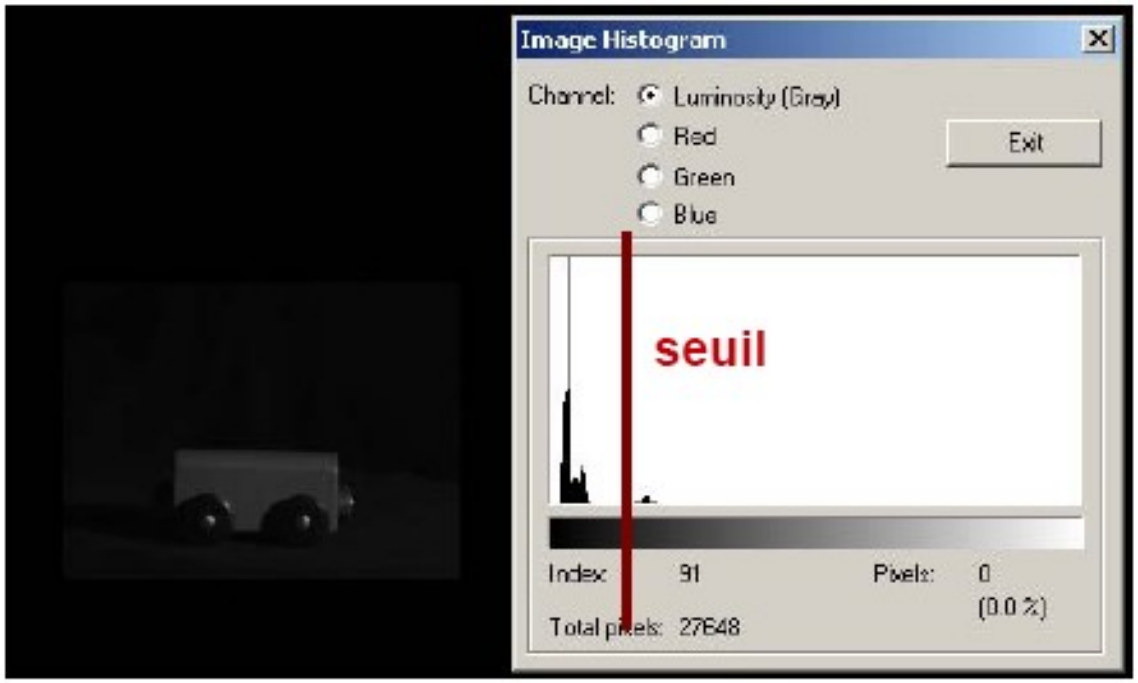
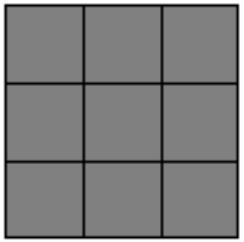


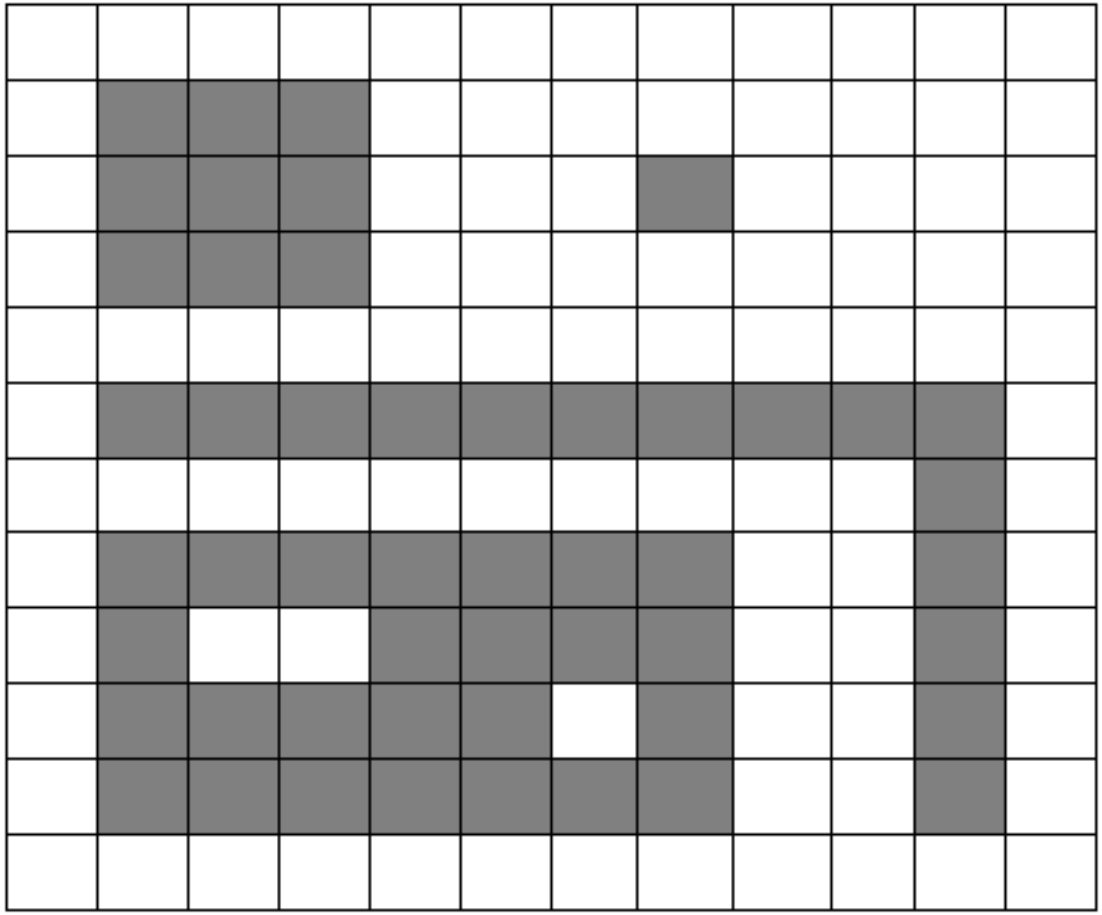
Image binaire

Dilatation

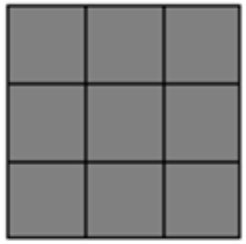
Erosion

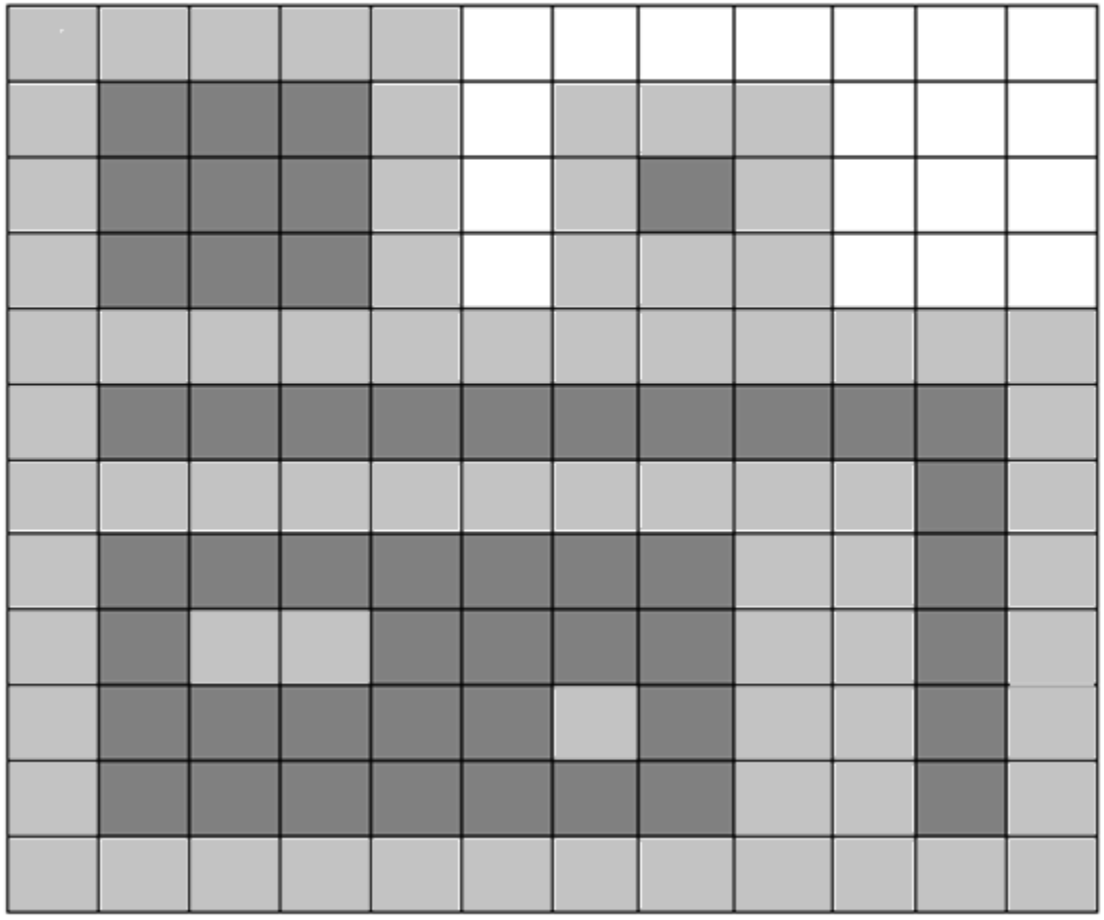
Dilatation: dilate image with the 3 by 3 structuring element



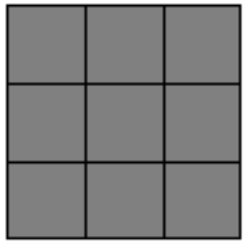


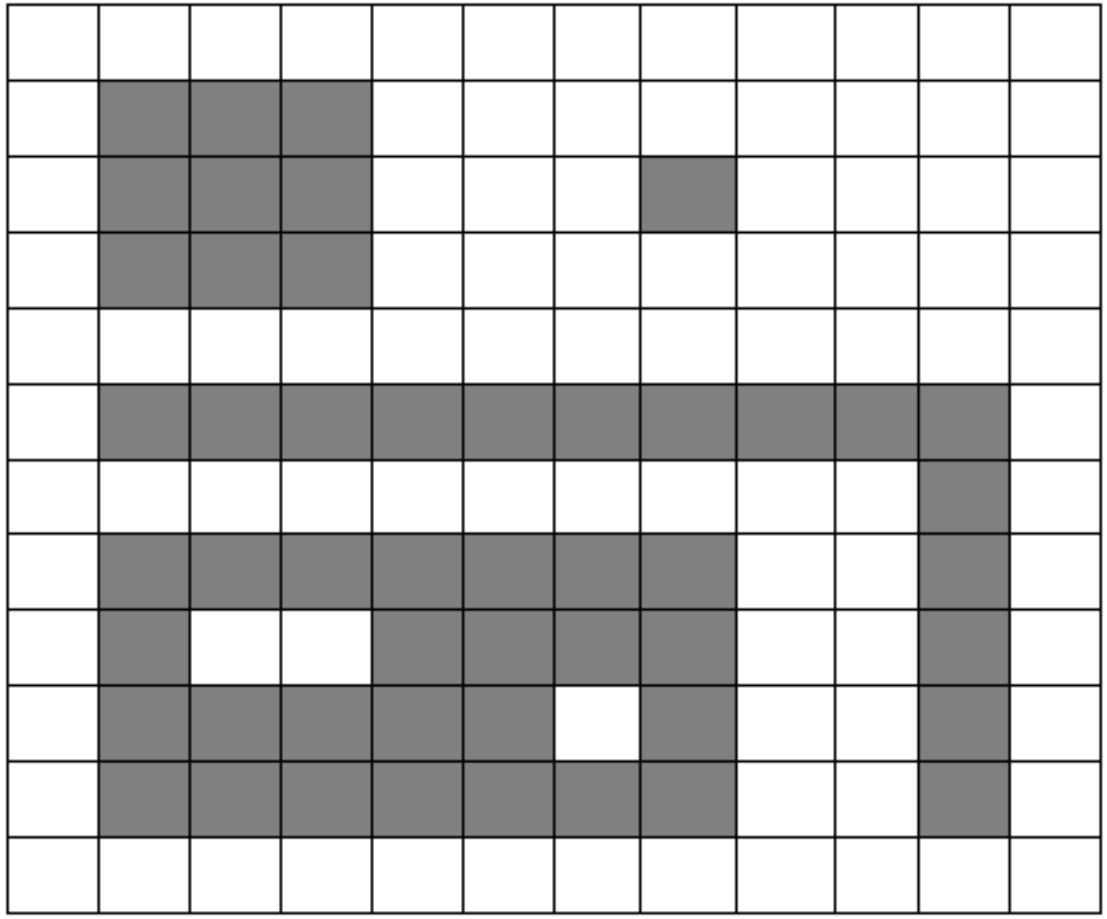
Solution



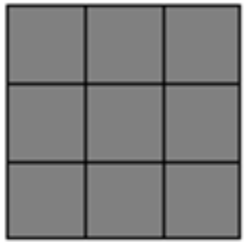


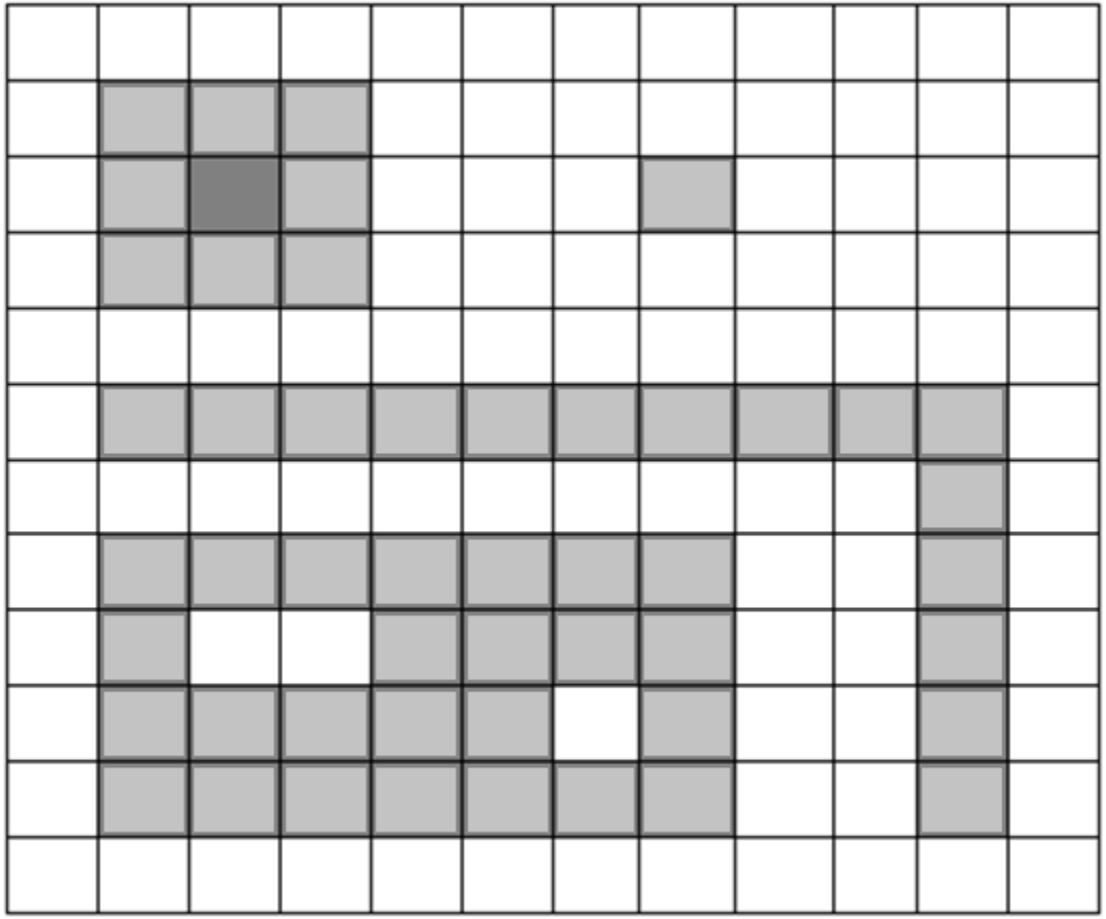
Erosion: erode image with the 3 by 3 structuring element



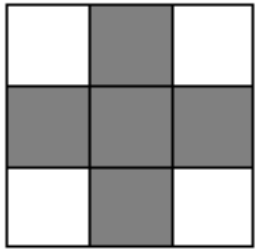


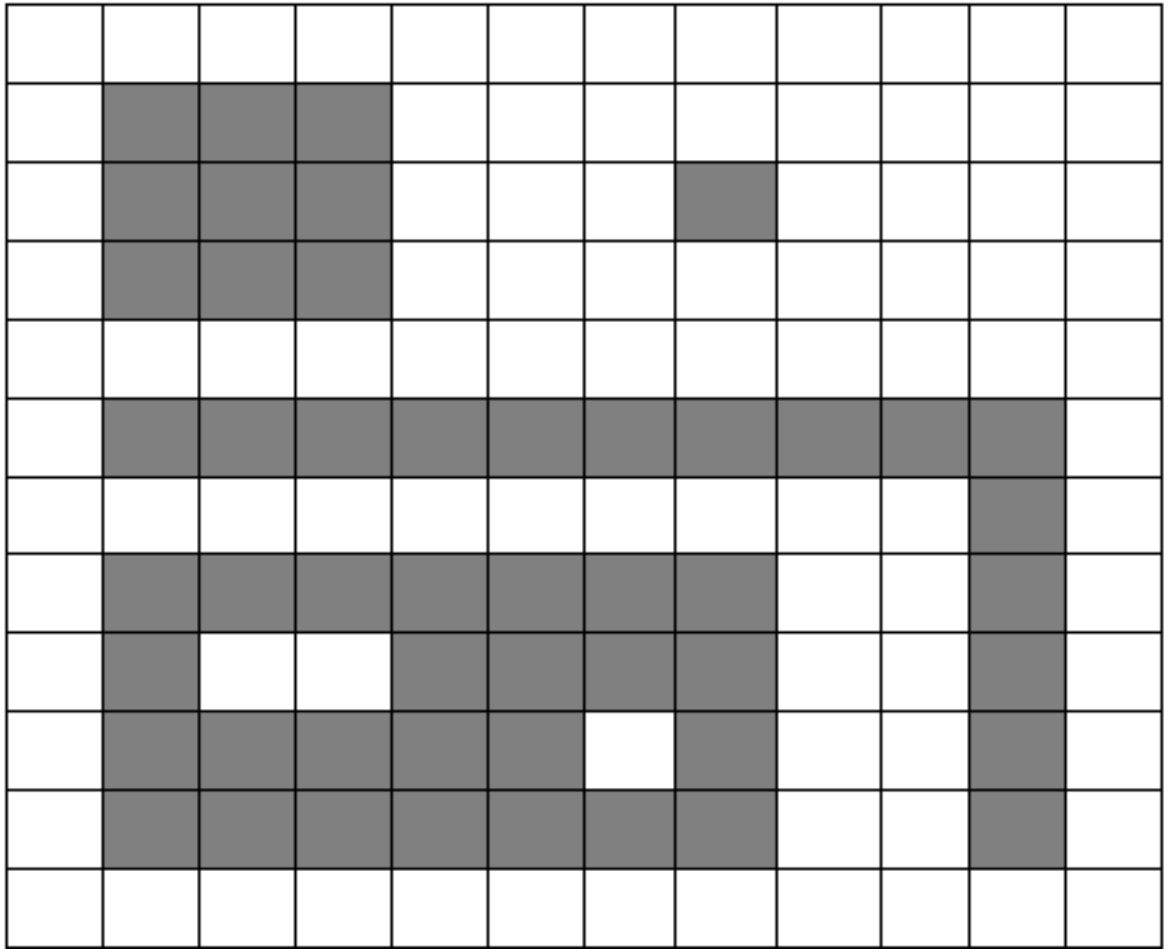
Solution



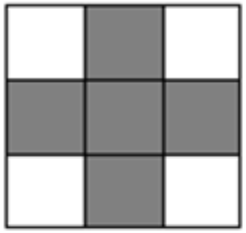


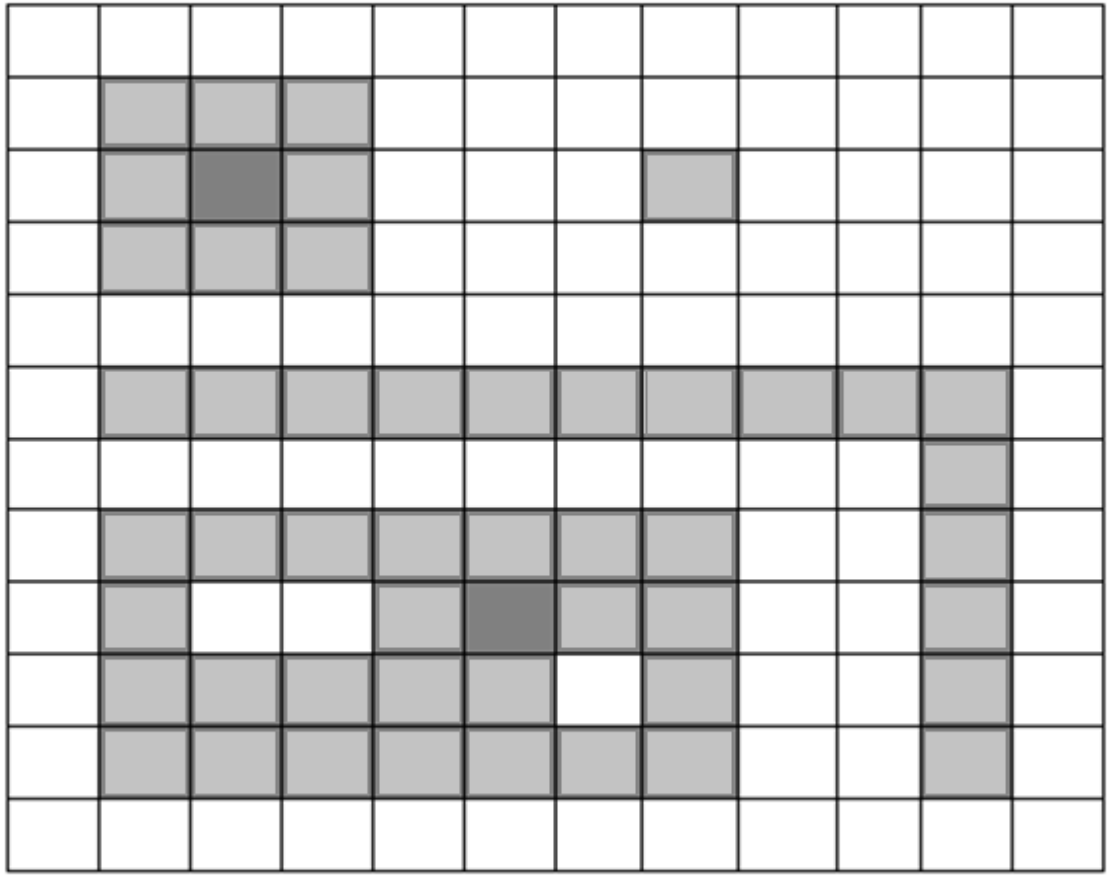
Eroding image with the 3 by 3 structuring element



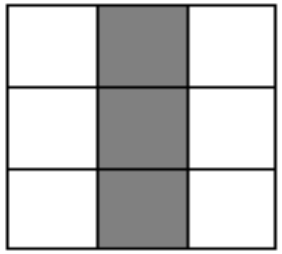


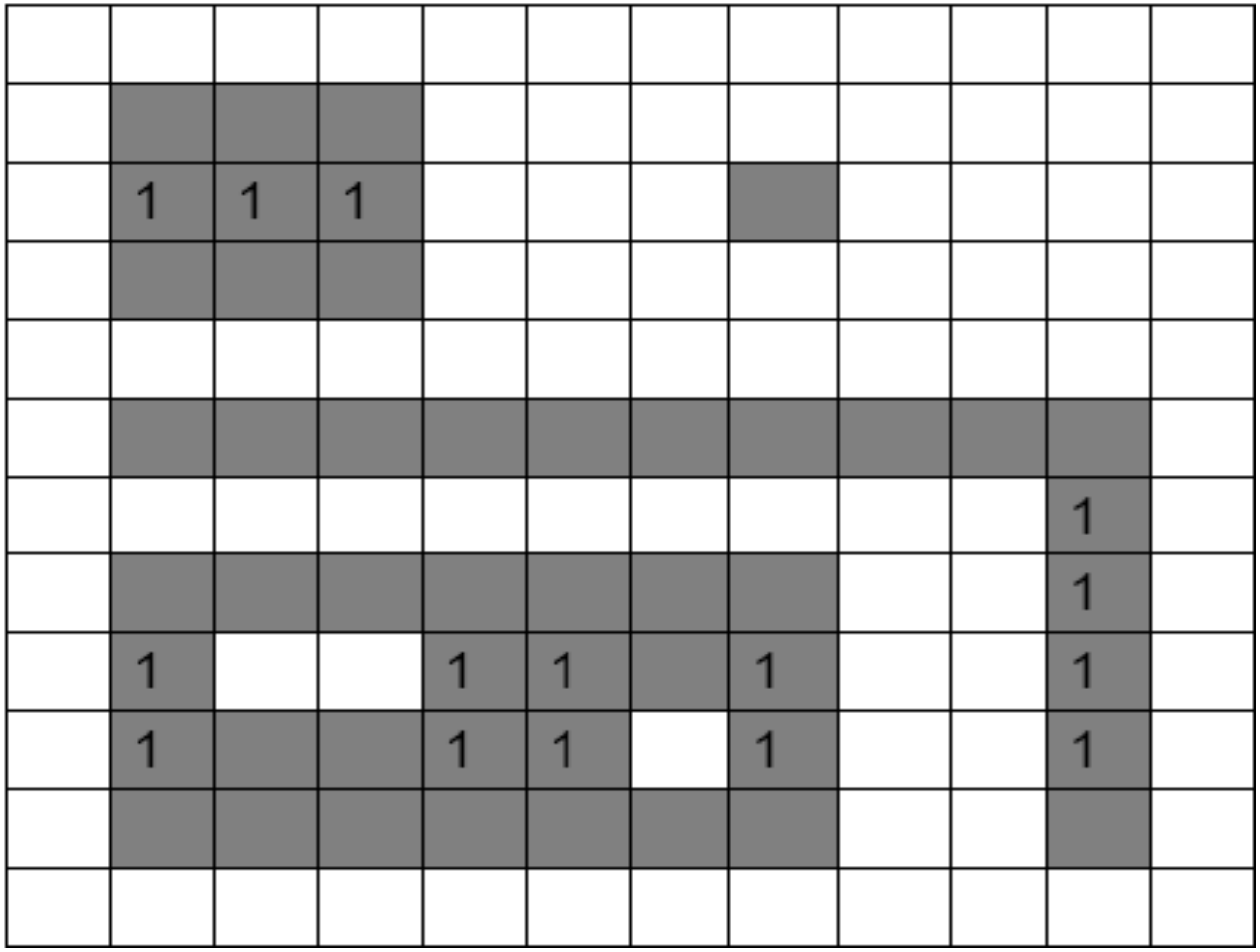
Solution



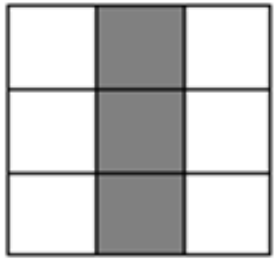


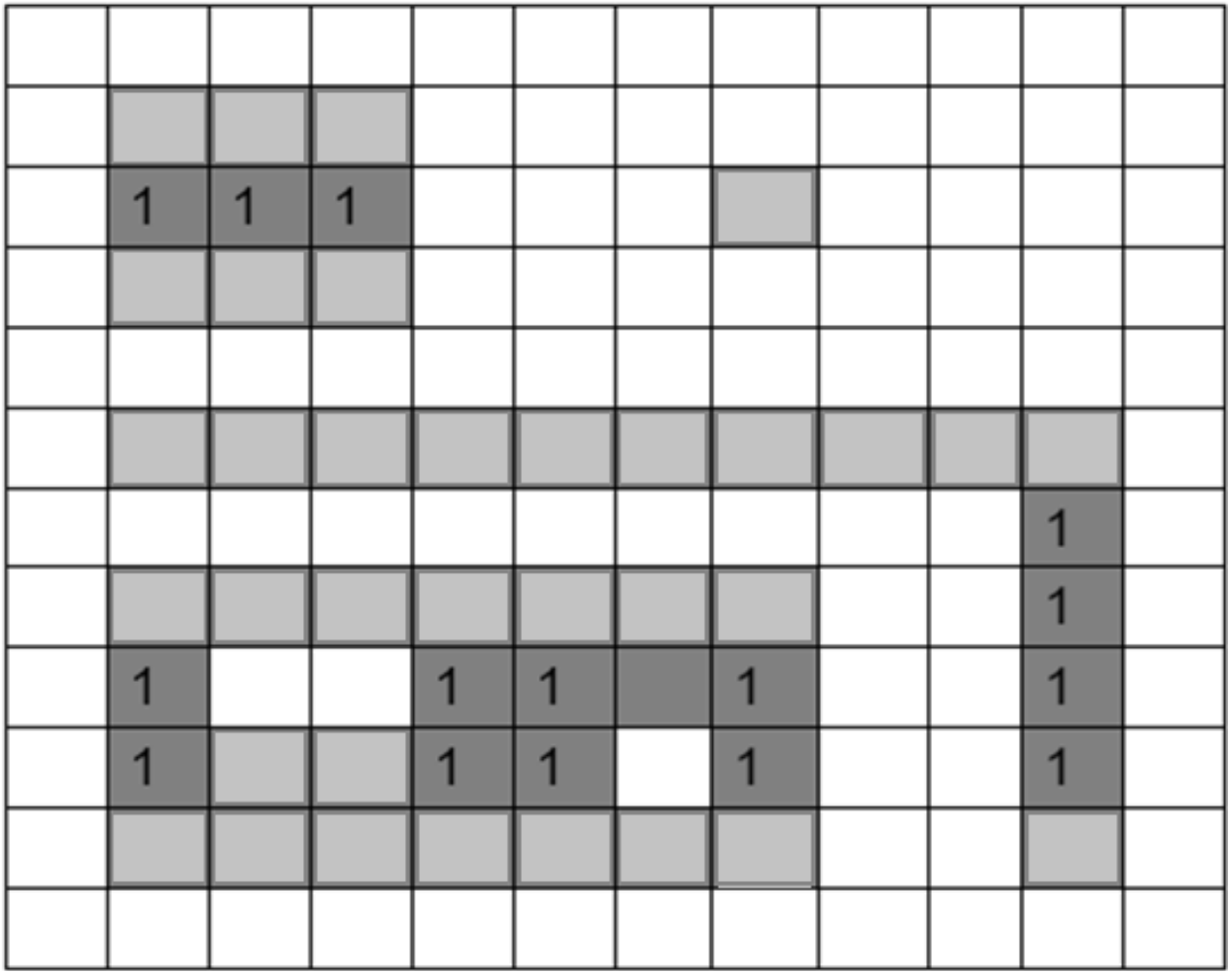
Eroding image with the 3 by 3 structuring element





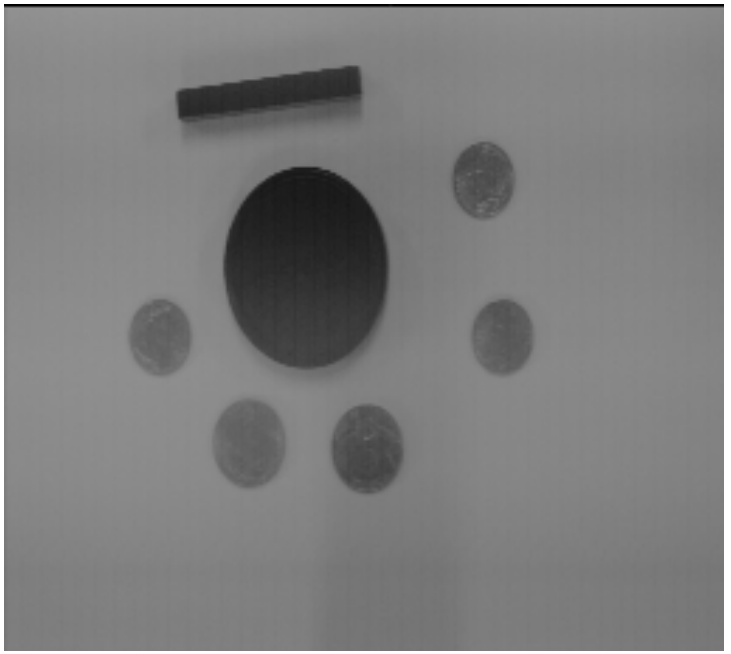
Solution



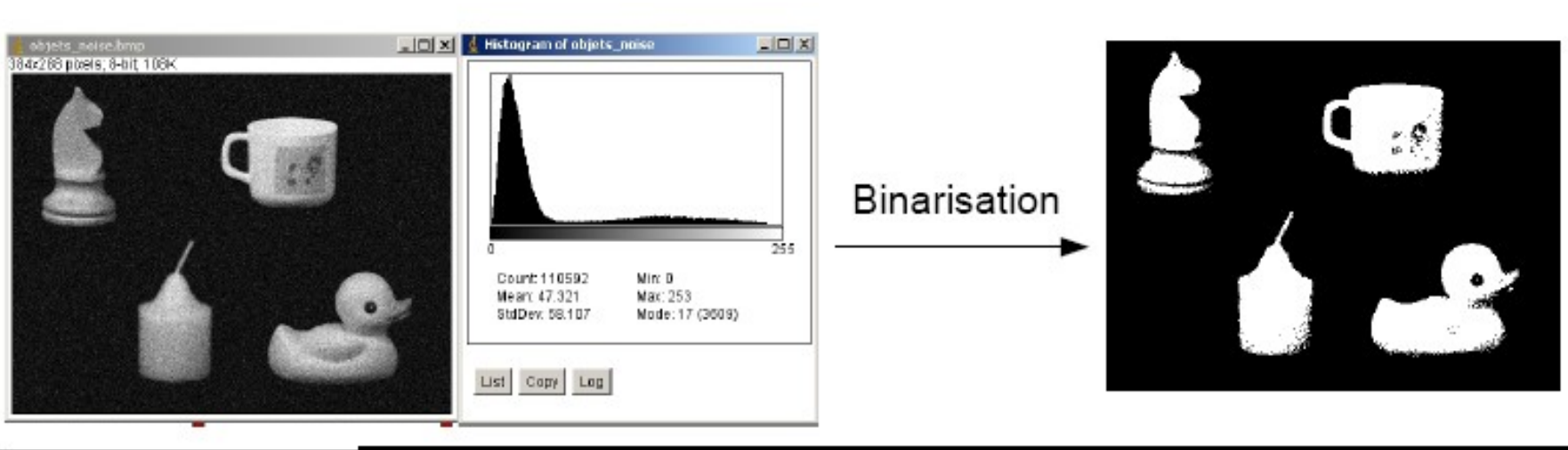


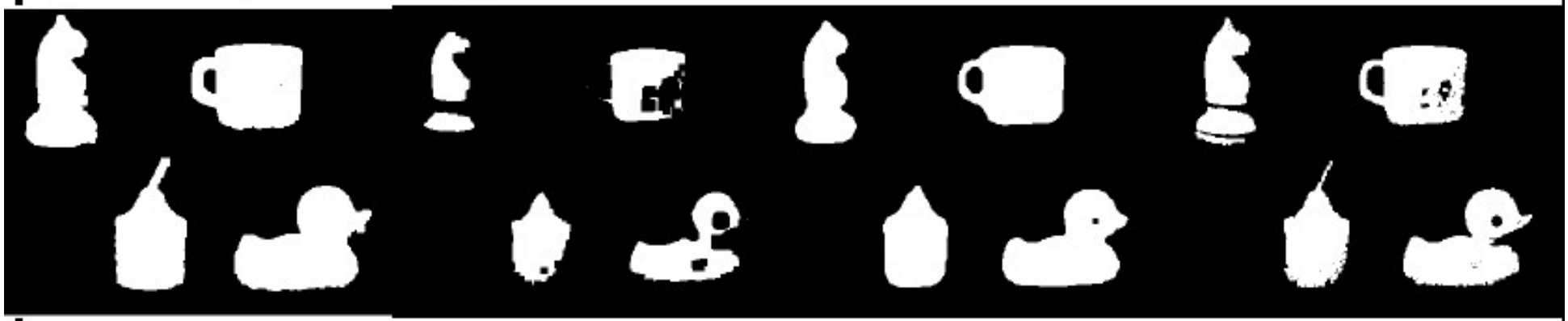
Hands on n°8

1) Load morpho.bmp.
2) Binarize image to obtain black objects on white background
3) Duplicate this image
4) We want to erase small objects.
5) In menu Process/Binary/options set parameters
6) We have to proceed as :
   - Dilatation
   - Erosion with the same mask
   - 5 erosion with a mask at 4 neighbours



Mathematical morphology





Dilatations Erosions Fermeture Ouverture

Hands on n°9

1) Load the image amour.bmp
2) Binarize it.
3) Do opening operation , with a square mask of size 5X5. What are the parts that have been eliminated ?
4) Take back the original image of origin, and made a threshold to binarize to compare.
4- Realize successively 10 closures with a 5X5 mask . What is the effect obtained ?
5. Use the selection function, to select than the biggest object (click on the form desired , then press on the button right of the mouse).

Principle of FFT

Spatial frequency in images correspond to periodic patterns

Possible to visualize them with the spatial frequency spectrum of the image

Passing from the image to the spatial frequency spectrum with **Fourier transform**

Implemented in all image processing softwares under the name FFT for Fast Fourier Transform.

$$F(f_x, f_y) = \iint f(x,y) \exp(-2j\pi(f_x x + f_y y))dxdy$$

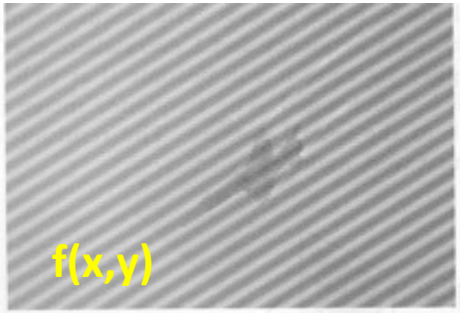


Figure 4.24a Original jet fighter image corrupted with a diagonal periodic noise pattern.

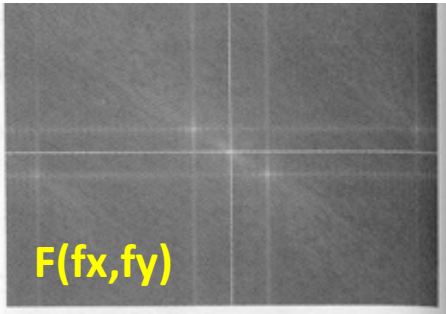


***Figure 4.24b** Fourier-transform image.*

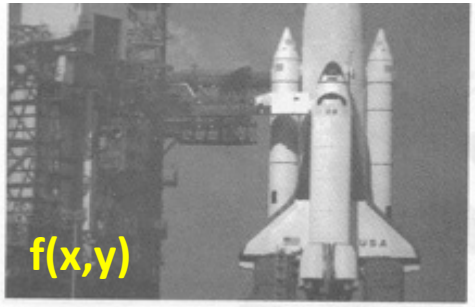


***Figure 4.25a** Original Space Shuttle image.*

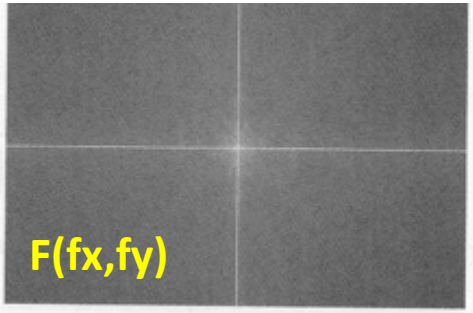
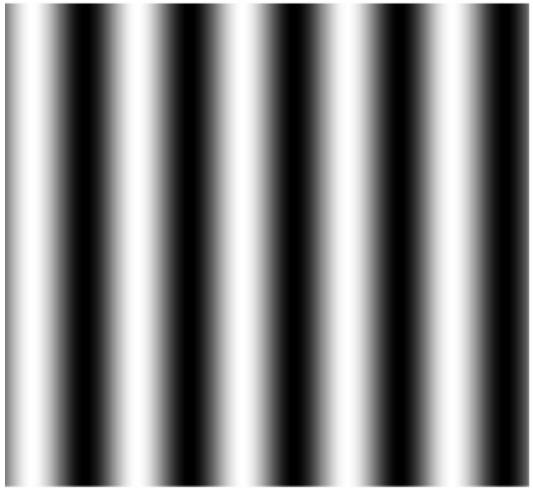
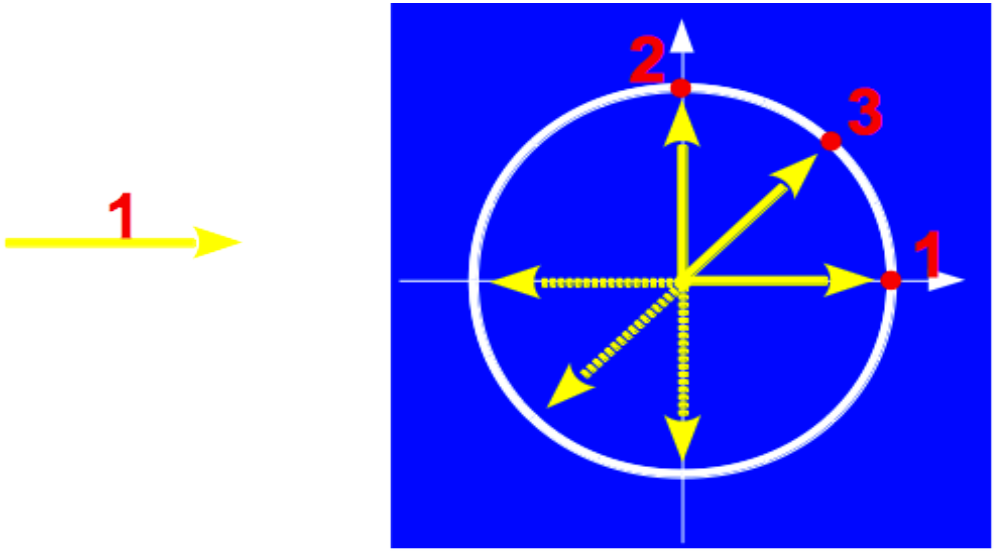


***Figure 4.25b** Fourier-transform image.*

Examples of simple FFT





Removing structures with FFT

Withdraw part of the spectrum corresponding to unwanted frequencies

Easy when they are localised in the spectrum like with the periodic noise brought by curtaining

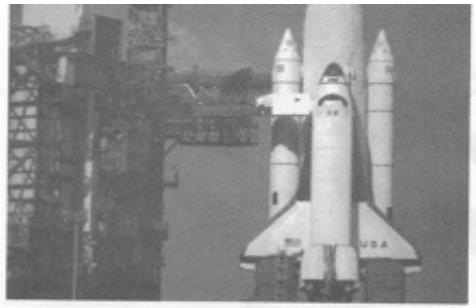


Figure 4.25a Original Space Shuttle image.

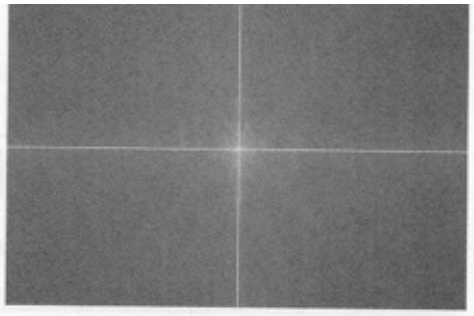


Figure 4.25b Fourier-transform image.

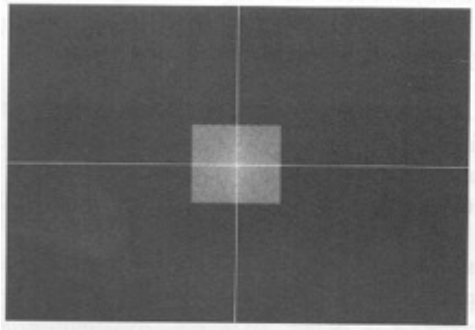


Figure 4.25c Fourier-transform image with the high frequencies zeroed.

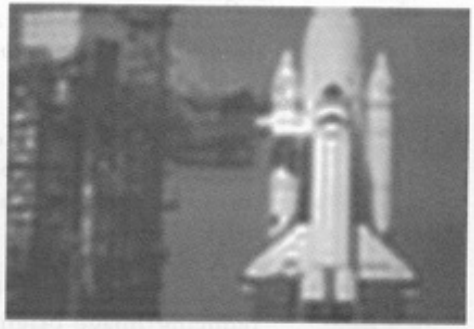


Figure 4.25d Inverse Fourier-transform image showing a low-pass filtered version of the original image.

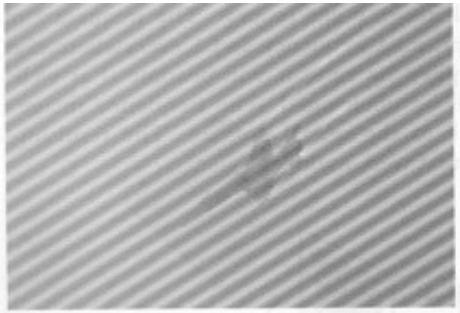


Figure 4.24a Original jet fighter image corrupted with a diagonal periodic noise pattern.

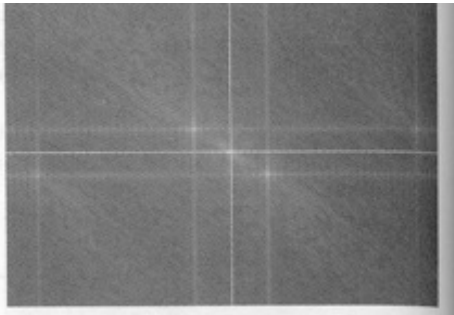


Figure 4.24b Fourier-transform image.

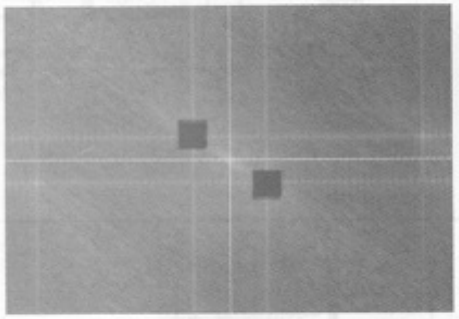
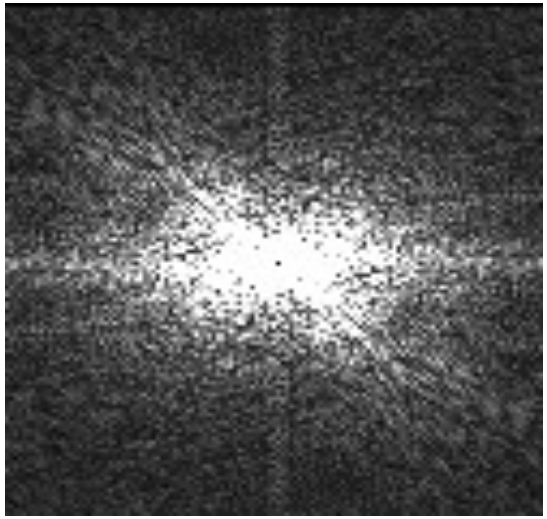


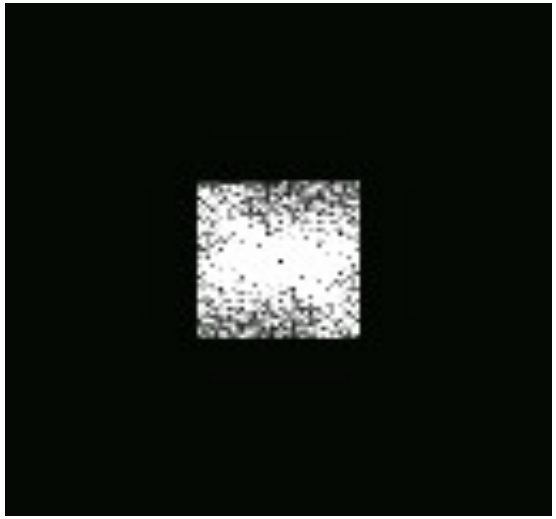
Figure 4.24c Fourier-transform image with the noise frequencies zeroed.

Figure 4.24d Inverse Fourier-transform image with most of the periodic noise removed—visibility of the aircraft is greatly improved.

TF

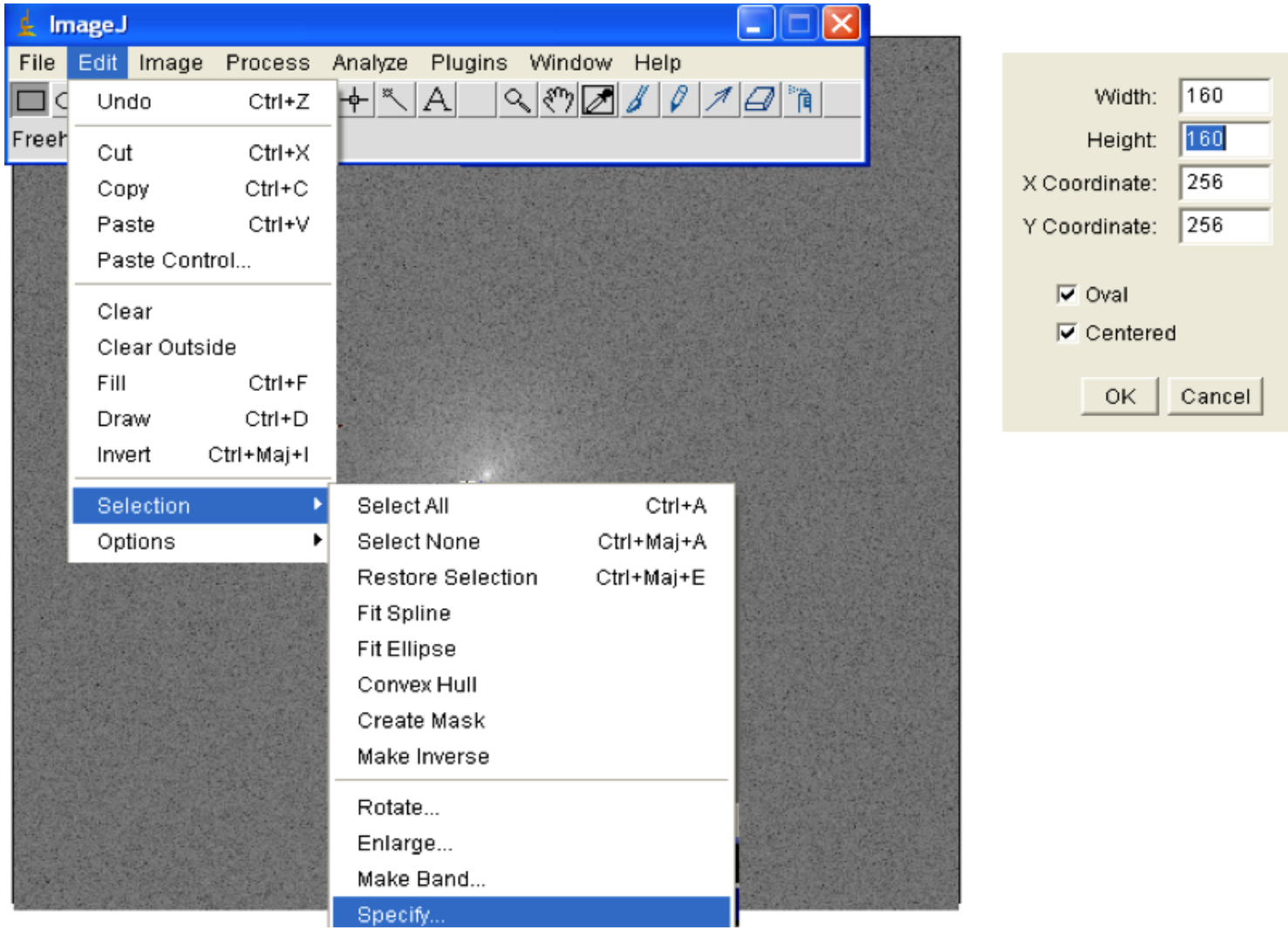


Filtrage

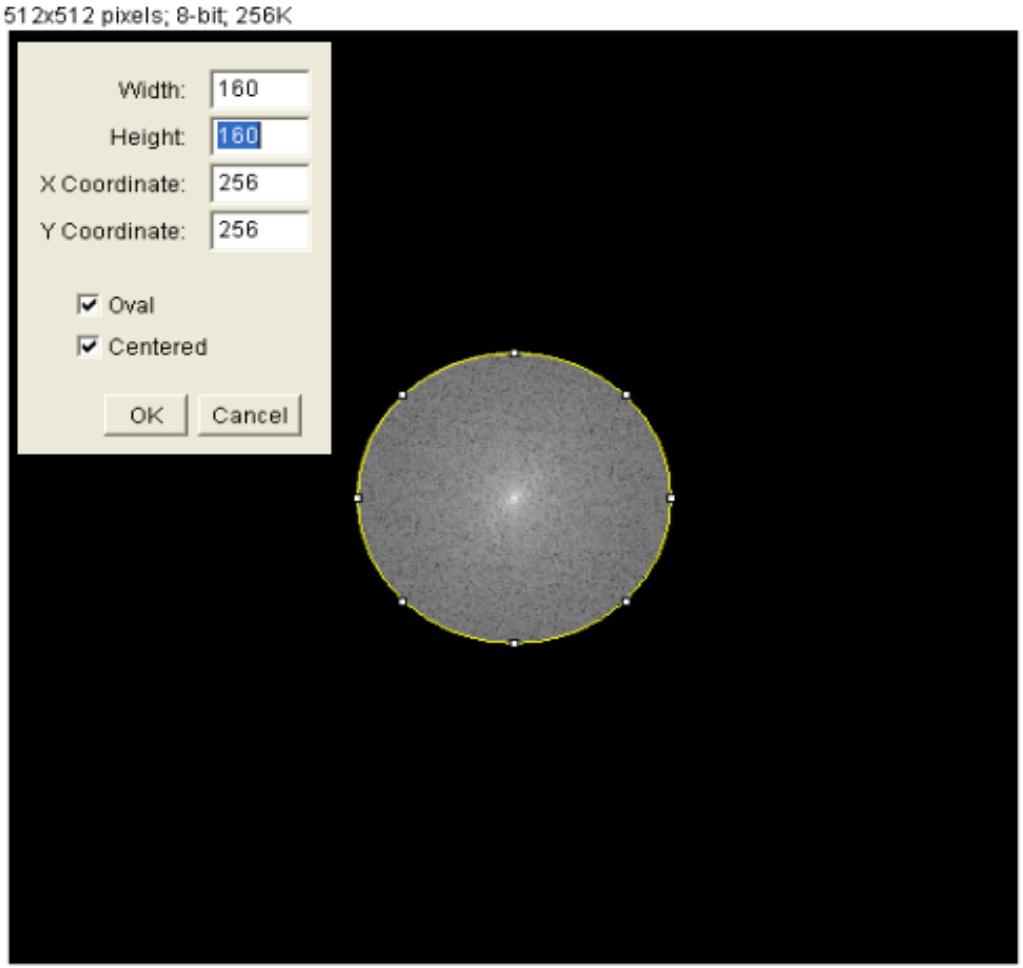


TF^{-1}

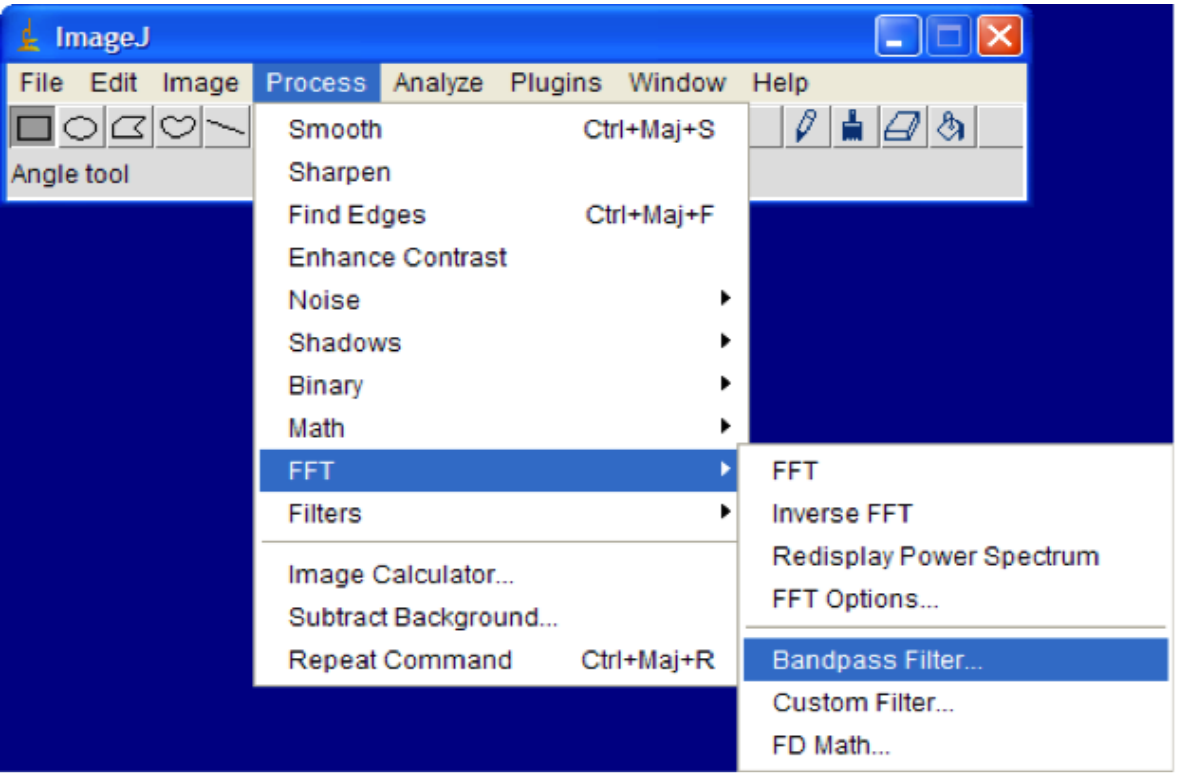
Practical realization

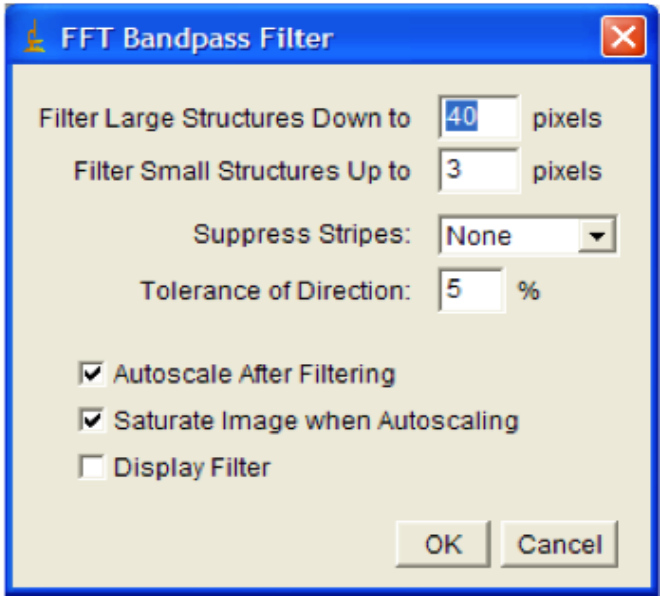


Practical realization

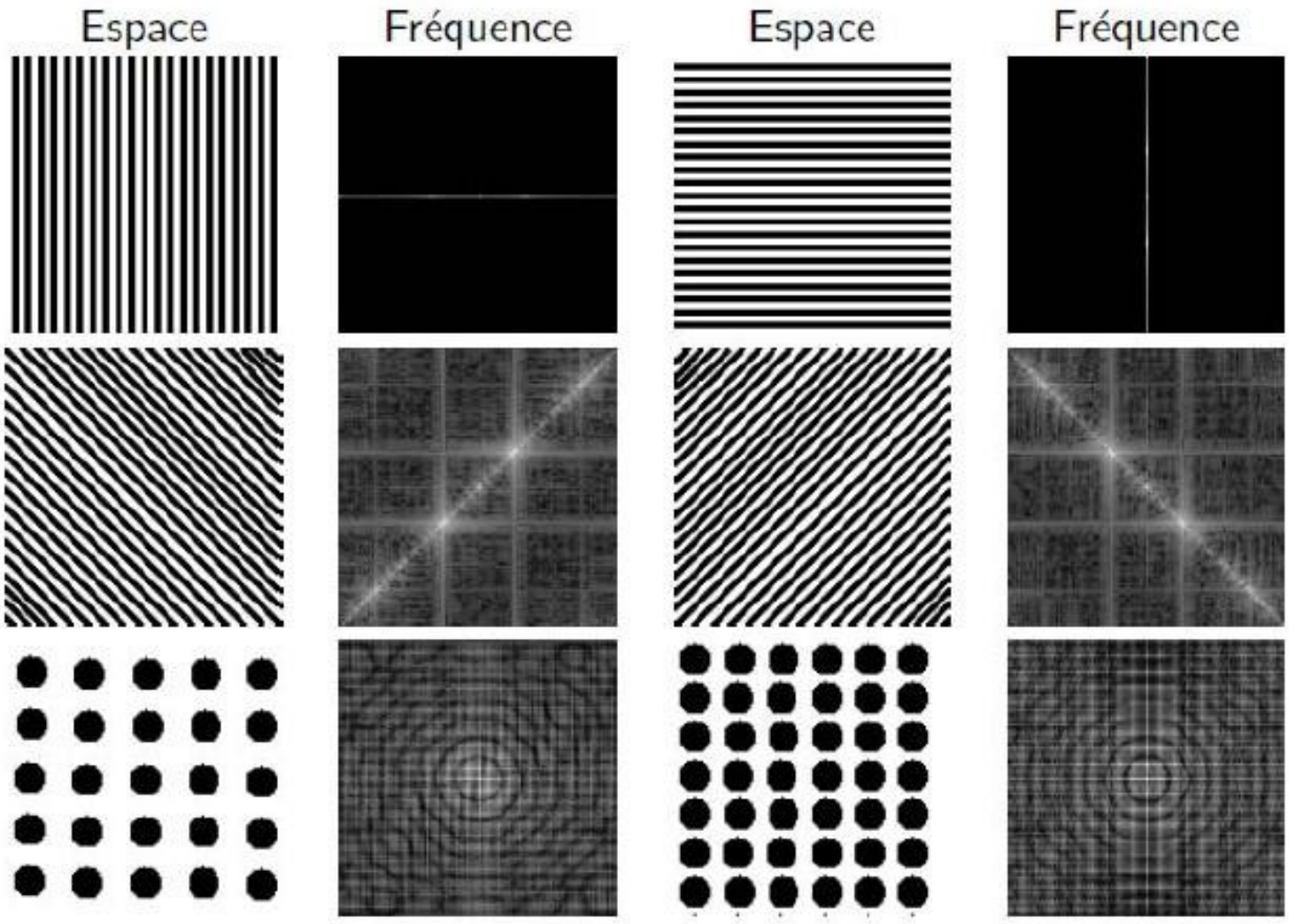


Practical realization

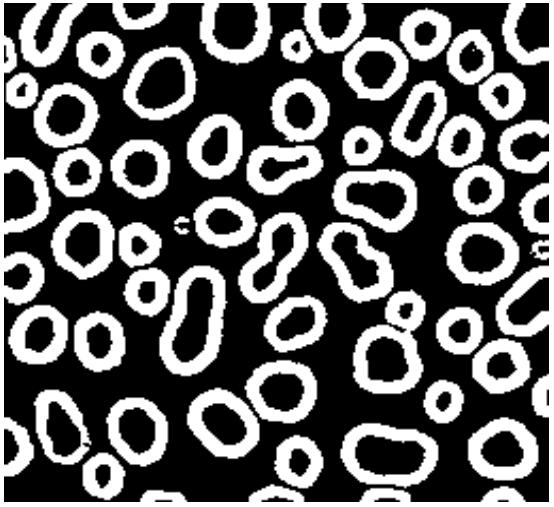


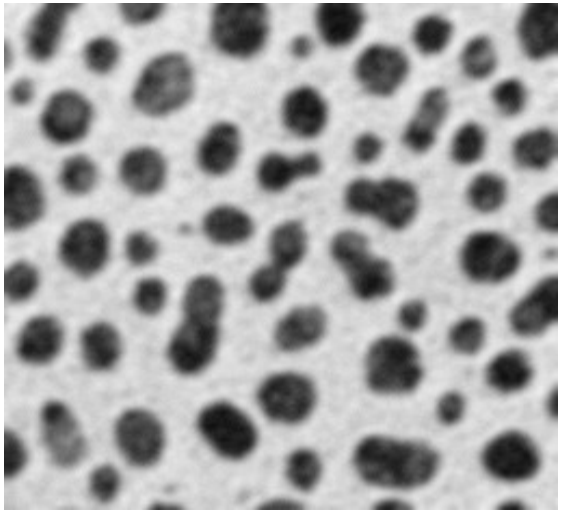


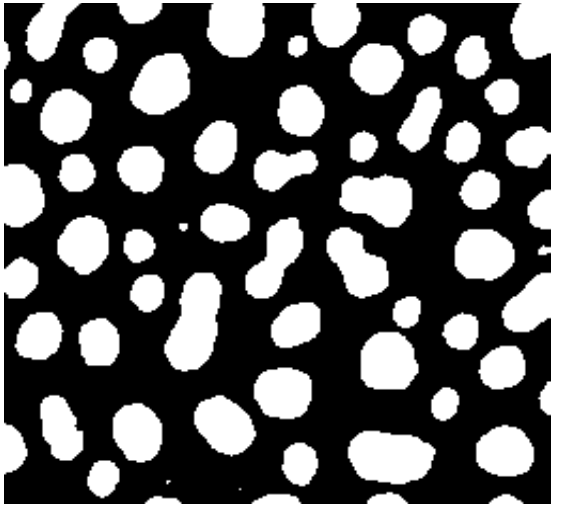
Textures characterization



Representation and description for shapes recognition

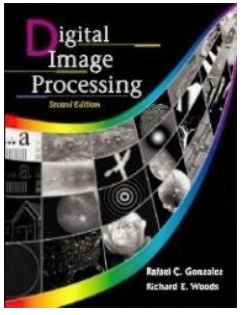




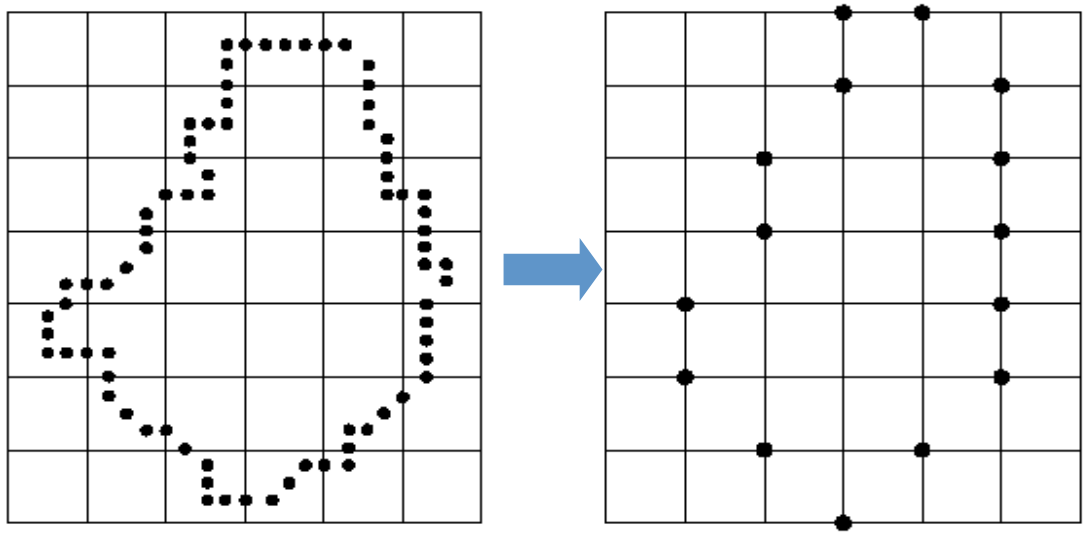


Methods to:

- **describe shapes (boundary or region)**
- with features/attributes.
- These can be viewed as external (boundary) or internal (region) characteristics, respectively.

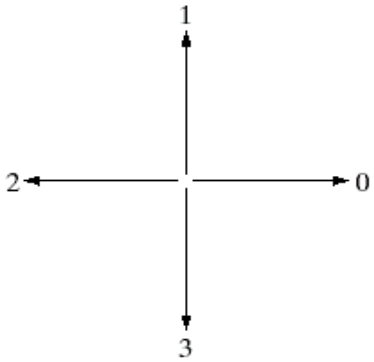


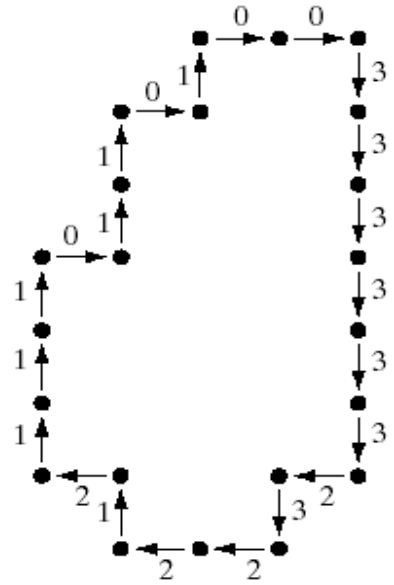
Chain codes: represent a boundary of a connected region.

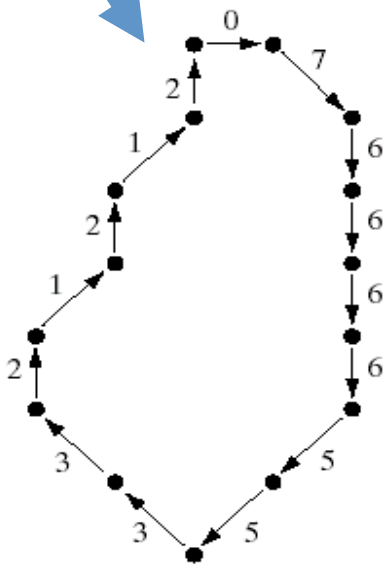


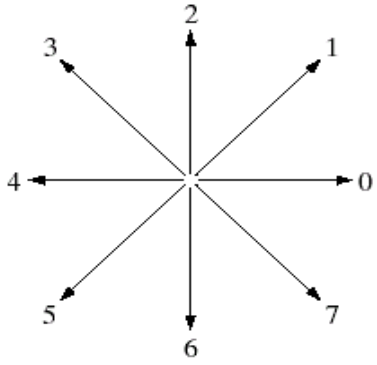
a b
c d

FIGURE 11.2
(a) Digital boundary with resampling grid superimposed. (b) Result of resampling. (c) 4-directional chain code. (d) 8-directional chain code.









Chain codes: represent a boundary of a connected region.

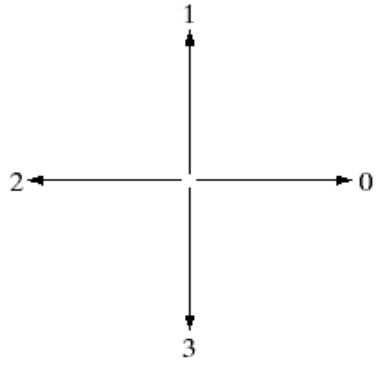
Chain codes can be based on either 4-connectedness or 8-connectedness.

The first difference of the chain code:

- This difference is obtained by counting the number of direction changes (in a counterclockwise direction)
- For example, the first difference of the 4-direction chain code 10103322 is 3133030.

Assuming the first difference code represent a closed path, rotation normalization can be achieved by circularly shifting the number of the code so that the list of numbers forms the smallest possible integer.

Size normalization can be achieved by adjusting the size of the resampling grid.

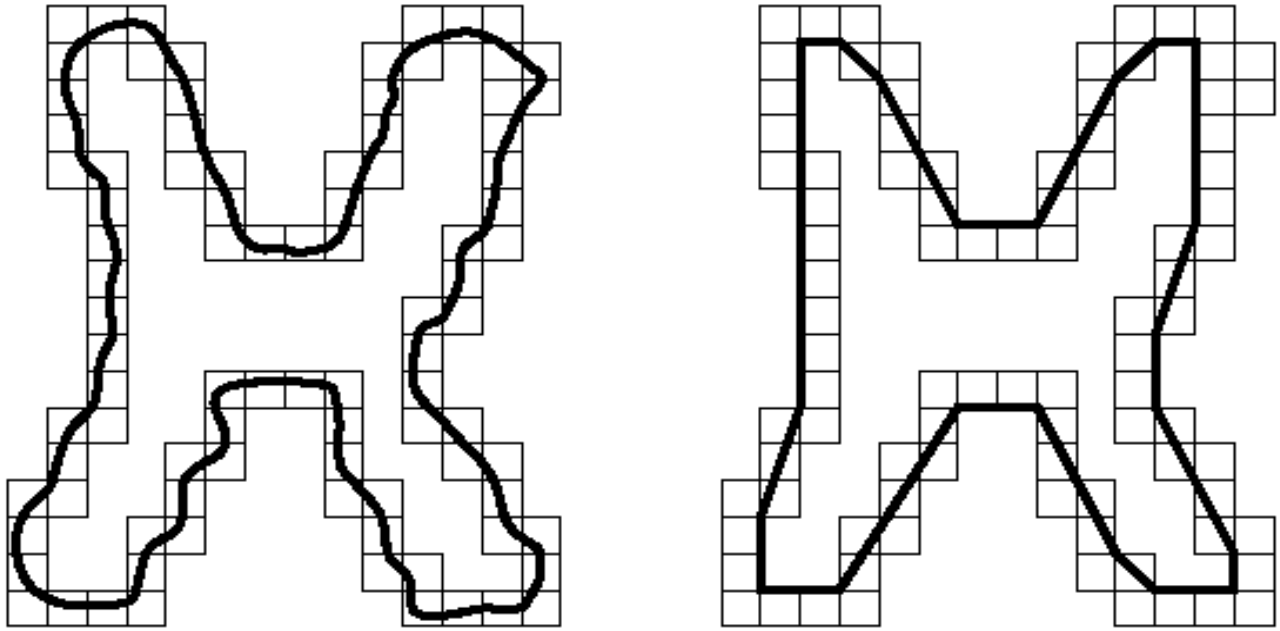


Polygonal approximation

Polygonal approximations: to represent a boundary by straight line segments, and a closed path becomes a polygon.
The number of straight line segments used determines the accuracy of the approximation.
Only the minimum required number of sides necessary to preserve the needed shape information should be used (Minimum perimeter polygons).
A larger number of sides will only add noise to the model.

a b

FIGURE 11.3
(a) Object boundary enclosed by cells. (b) Minimum perimeter polygon.

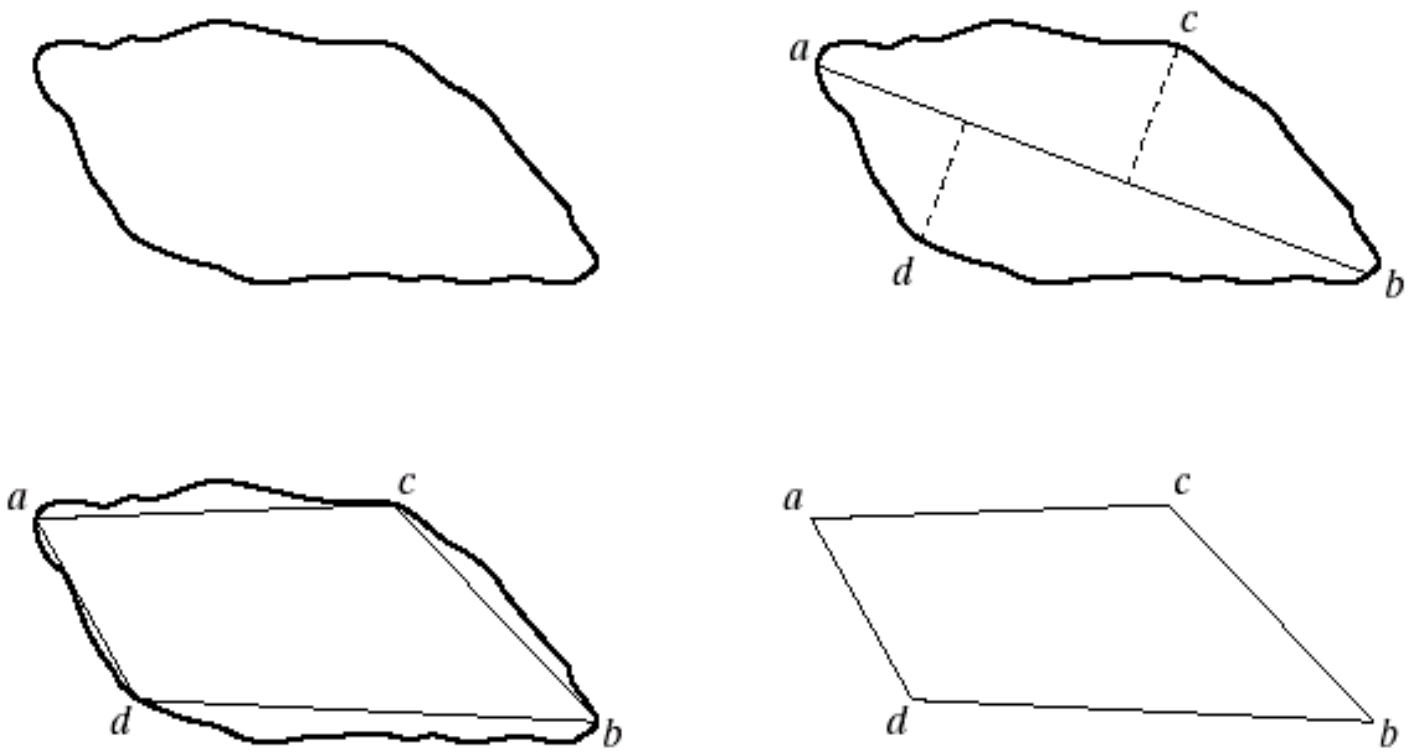


Polygonal approximation

Minimum perimeter polygons: (Merging and splitting)

Merging and splitting are often used together to ensure that vertices appear where they would naturally in the boundary.

A least squares criterion to a straight line is used to stop the processing.



a b
c d

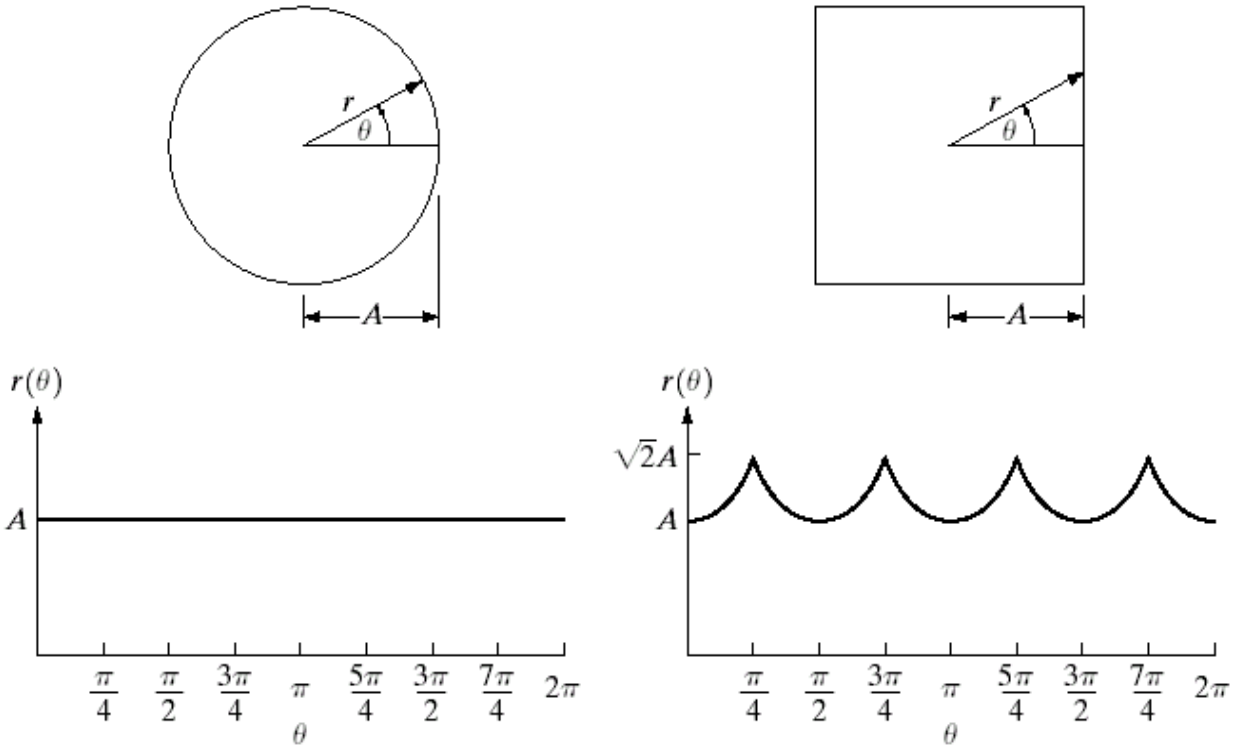
FIGURE 11.4
(a) Original boundary. (b) Boundary divided into segments based on extreme points. (c) Joining of vertices. (d) Resulting polygon.

Signature

The idea behind a signature is to convert a two dimensional boundary into a representative one dimensional function. Signatures are invariant to location, but will depend on rotation and scaling.

a b

FIGURE 11.5 Distance-versus-angle signatures. In (a) $r(\theta)$ is constant. In (b), the signature consists of repetitions of the pattern $r(\theta) = A\sec\theta$ for $0 \le \theta \le \pi/4$ and $r(\theta) = A\csc\theta$ for $\pi/4 < \theta \le \pi/2$.

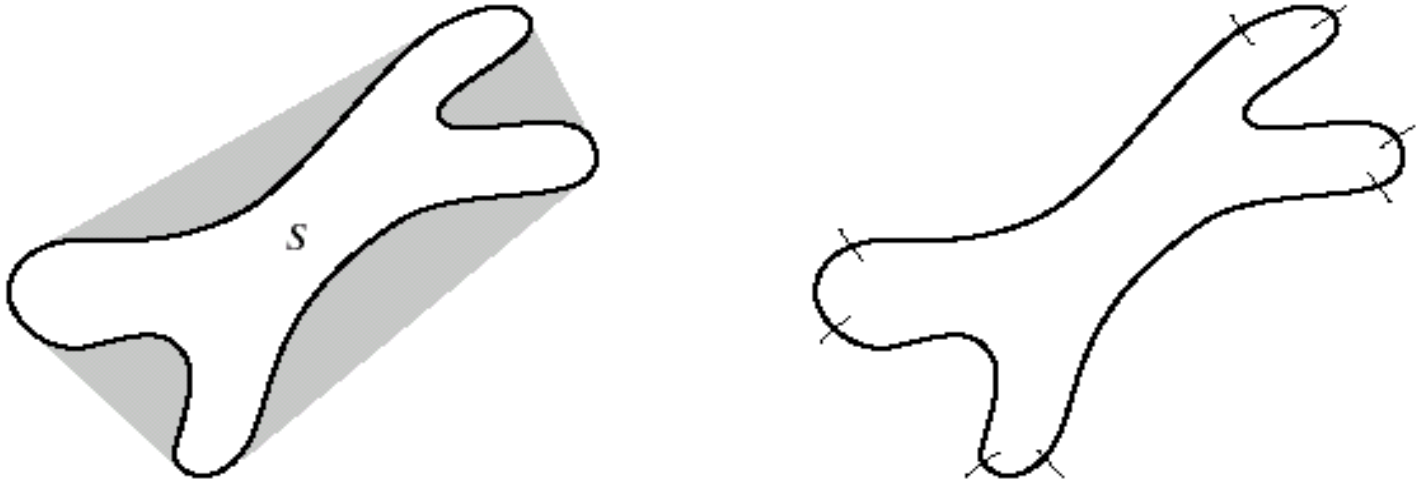


Starting at the point farthest from the reference point or using the major axis of the region can be used to decrease dependence on rotation.

Scale invariance can be achieved by either scaling the signature function to fixed amplitude or by dividing the function values by the std of the function.

Boundary segments

Boundary segments: decompose a boundary into segments. Use of the convex hull of the region enclosed by the boundary is a powerful tool for robust decomposition of the boundary.

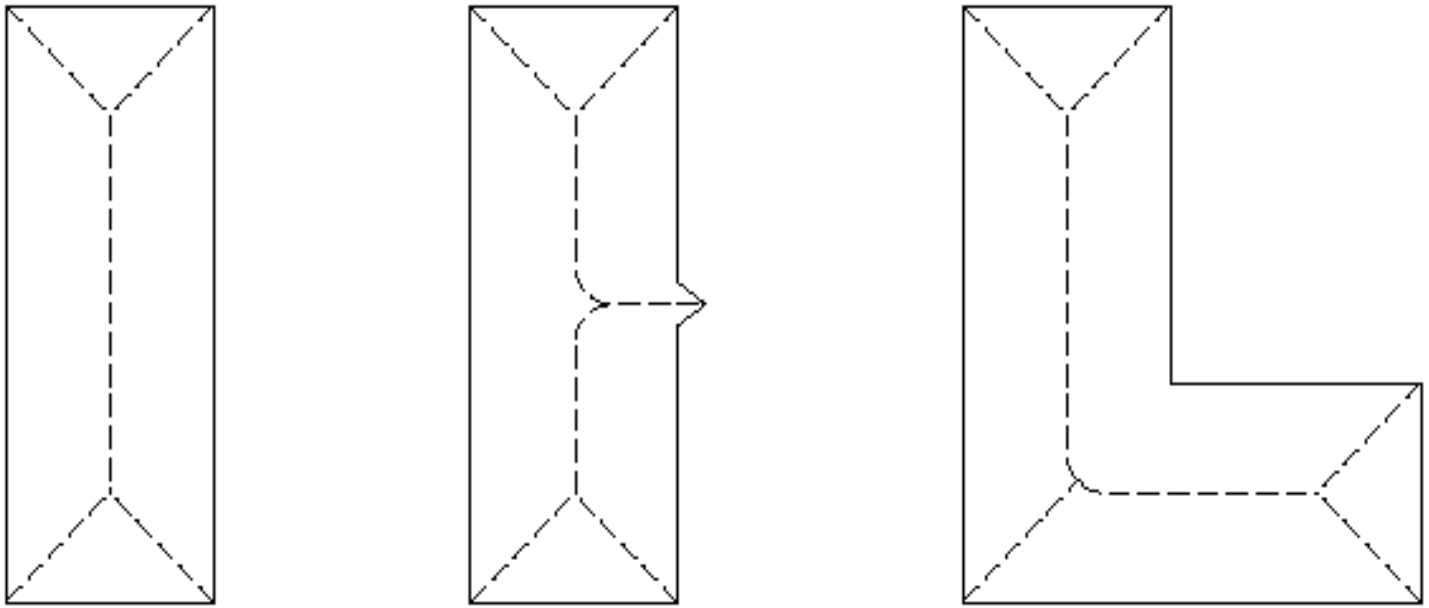


a b

FIGURE 11.6
(a) A region, S, and its convex deficiency (shaded). (b) Partitioned boundary.

Skeleton

Skeletons: produce a one pixel wide graph that has the same basic shape of the region, like a stick figure of a human. It can be used to analyze the geometric structure of a region which has bumps and “arms”.



a b c

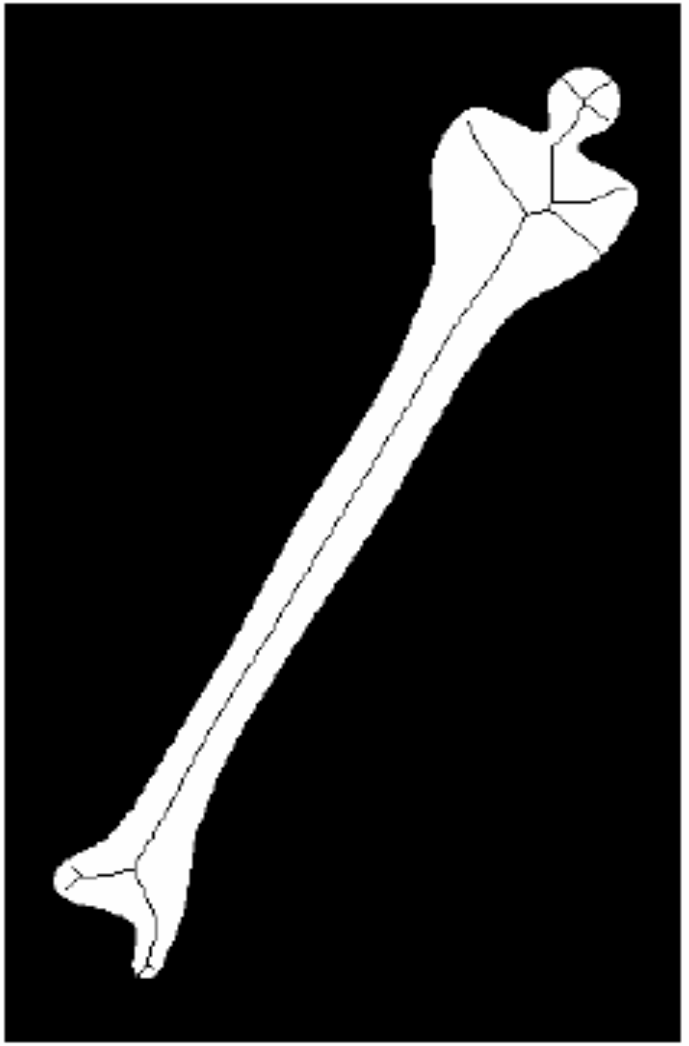
FIGURE 11.7
Medial axes (dashed) of three simple regions.

Skeleton

One application of skeletonization is for character recognition.
A letter or character is determined by the center-line of its strokes, and is unrelated to the width of the stroke lines.

Image J → Process/Binary/Skeletonize

FIGURE 11.10
Human leg bone and skeleton of the region shown superimposed.



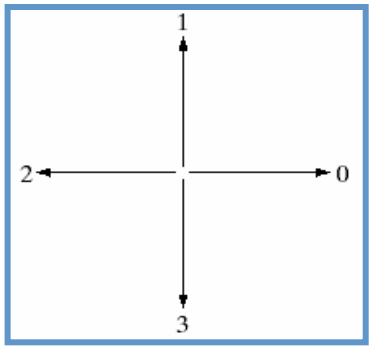
Boundary descriptors

There are several simple geometric measures that can be useful for describing a boundary.

The length of a boundary: the number of pixels along a boundary gives a rough approximation of its length → Perimeter

Curvature: the rate of change of slope. To measure a curvature accurately at a point in a digital boundary is difficult. The difference between the slops of adjacent boundary segments is used as a descriptor of curvature at the point of intersection of segments

Boundary descriptors: shape numbers



First difference

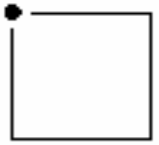
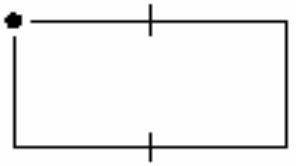
	Order 4	Order 6
		
Chain code:	0 3 2 1	0 0 3 2 2 1
Difference:	3 3 3 3	3 0 3 3 0 3
Shape no.:	3 3 3 3	0 3 3 0 3 3

FIGURE 11.11 All shapes of order 4, 6, and 8. The directions are from Fig. 11.1(a), and the dot indicates the starting point.

The shape number of a boundary is defined as the first difference of smallest magnitude.
The order *n* of a shape number is defined as the number of digits in its representation.

Boundary descriptors: shape numbers

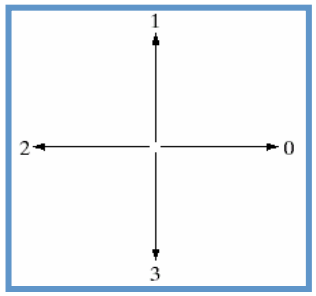
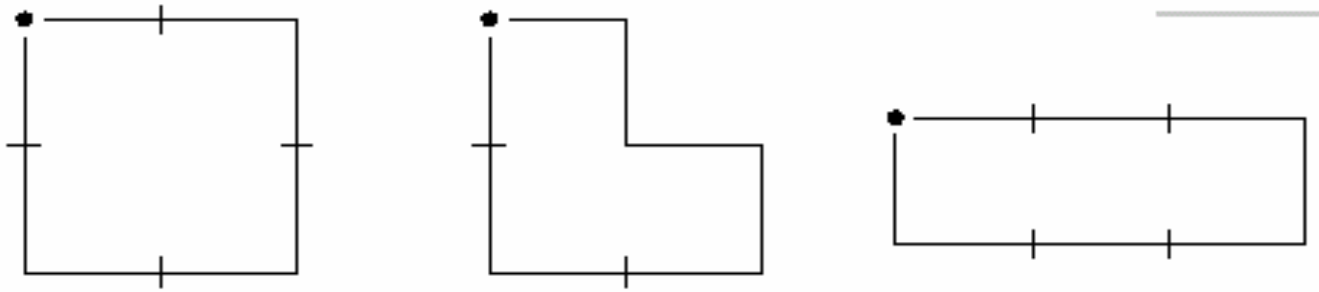


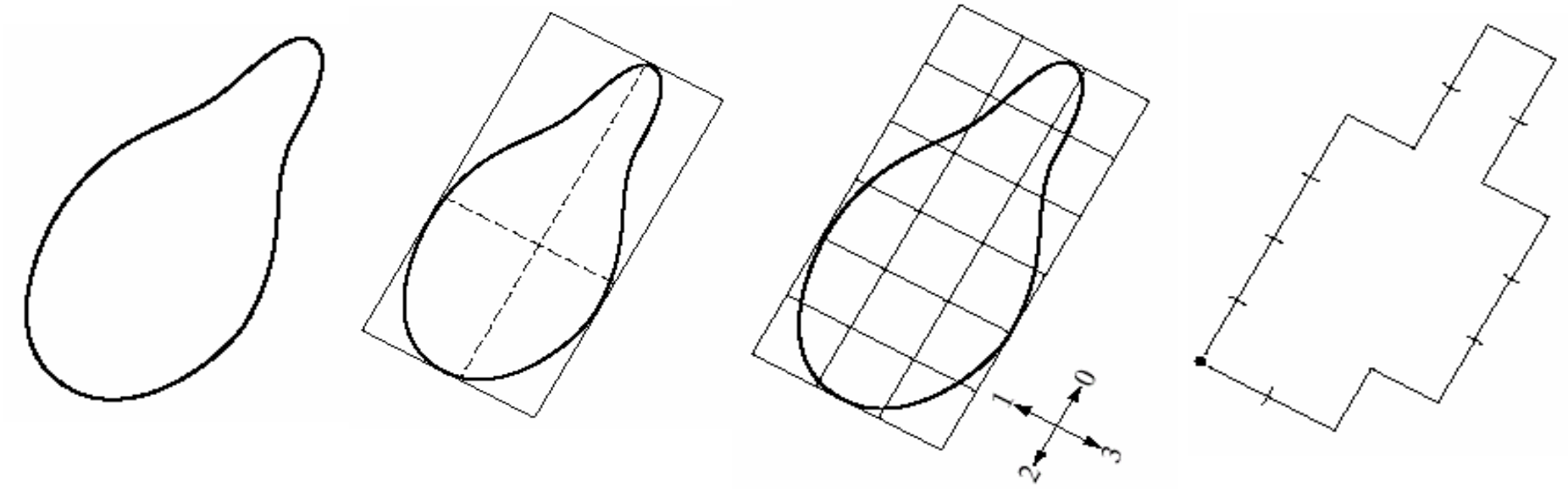
FIGURE 11.11 All shapes of order 4, 6, and 8. The directions are from Fig. 11.1(a), and the dot indicates the starting point.

Order 8



Chain code:	0 0 3 3 2 2 1 1	0 3 0 3 2 2 1 1	0 0 0 3 2 2 2 1
Difference:	3 0 3 0 3 0 3 0	3 3 1 3 3 0 3 0	3 0 0 3 3 0 0 3
Shape no.:	0 3 0 3 0 3 0 3	0 3 0 3 3 1 3 3	0 0 3 3 0 0 3 3

Boundary descriptors: shape numbers



Chain code:	0	0	0	0	3	0	0	3	2	2	3	2	2	2	1	2	1	1
Difference:	3	0	0	0	3	1	0	3	3	0	1	3	0	0	3	1	3	0
Shape no.:	0	0	0	3	1	0	3	3	0	1	3	0	0	3	1	3	0	3

Boundary descriptors: Fourier descriptors

This is a way of using the Fourier transform to analyze the shape of a boundary.

The *x*-*y* coordinates of the boundary are treated as the real and imaginary parts of a complex number → $u_k = x_k + iy_k$

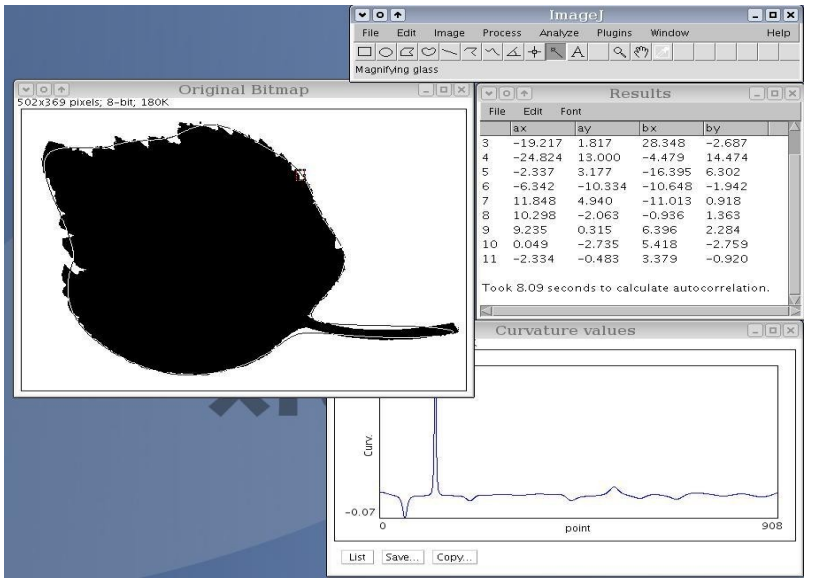
Then the list of coordinates is Fourier transformed using the FFT.

The Fourier coefficients are called the Fourier descriptors.

$$f_l = \sum_{k=0}^{N-1} u_k e^{-\frac{j2\pi kl}{N}}$$

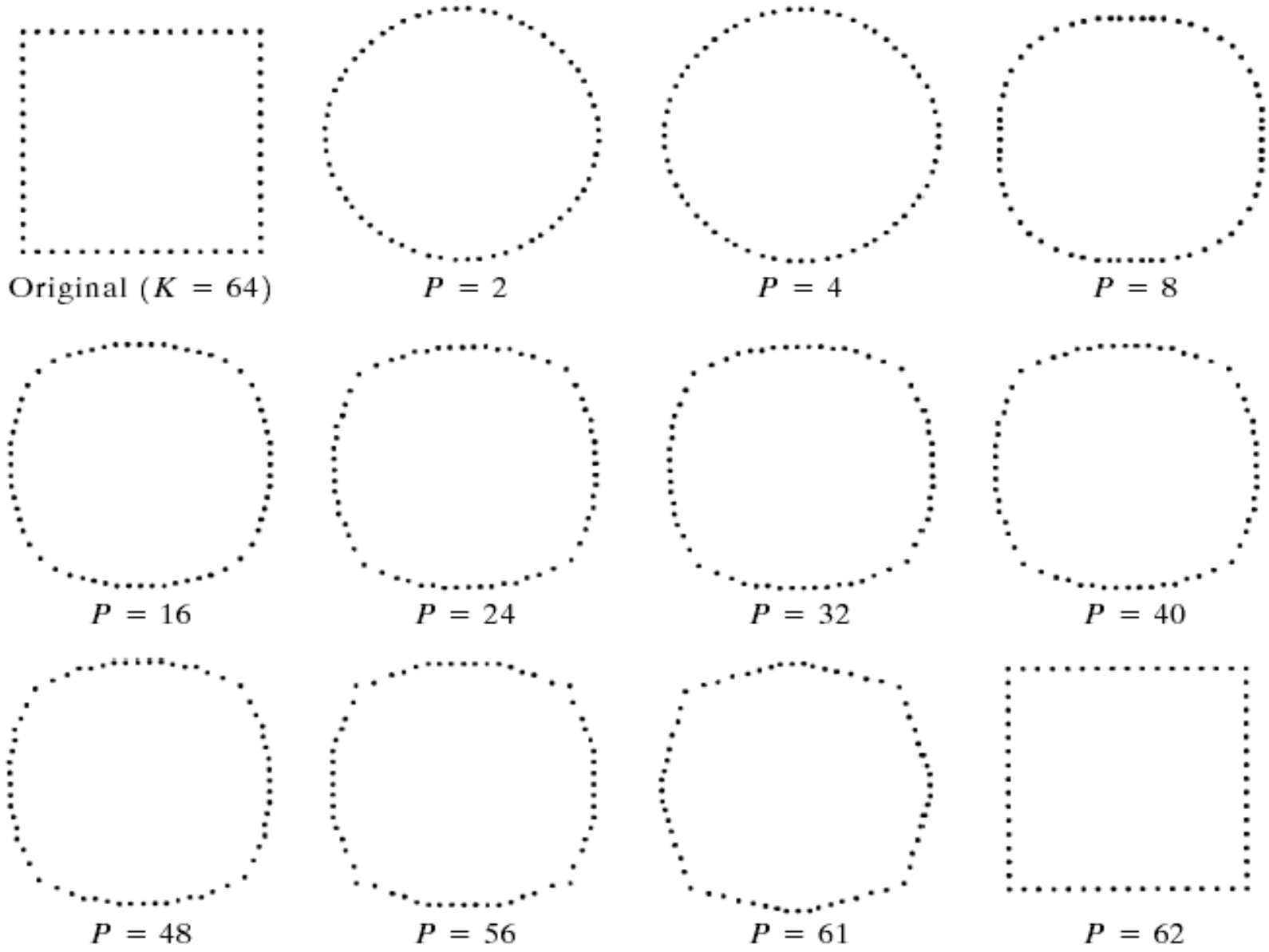
The basic shape of the region is determined by the first several coefficients, which represent lower frequencies. Higher frequency terms provide information on the fine detail of the boundary.

Image J → Plugin Fourier_.jar



Boundary descriptors: Fourier descriptors

FIGURE 11.14 Examples of reconstruction from Fourier descriptors. P is the number of Fourier coefficients used in the reconstruction of the boundary.



Region descriptors

Some simple descriptors

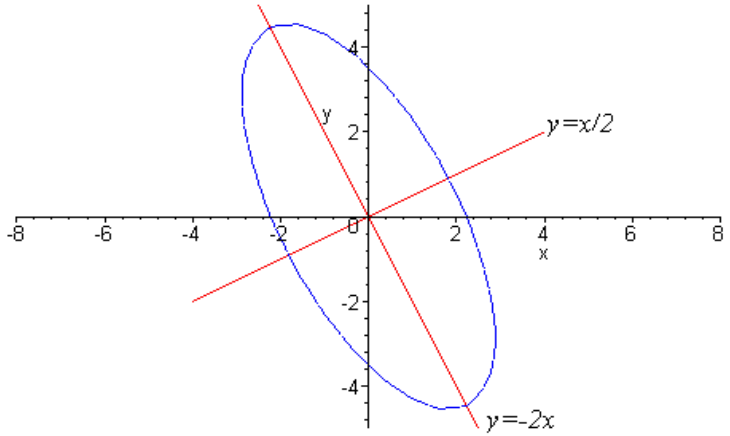
- The area of a region: the number of pixels in the region
- The perimeter of a region: the length of its boundary
- The compactness of a region: $(\text{perimeter})^2/\text{area}$
- The mean and median of the gray levels
- The minimum and maximum gray-level values
- The number of pixels with values above and below the mean

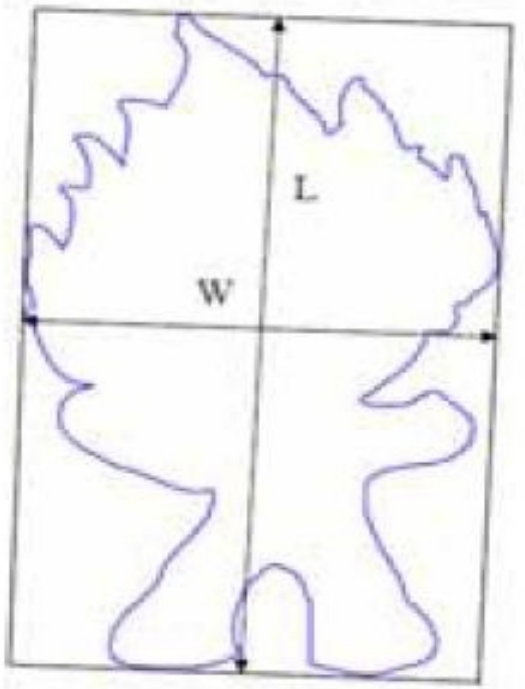
Region descriptors: eccentricity

Eccentricity is the measure of aspectratio

It's ratio of length of major axis to minor axis (think ellipse for example)

Calculated by principal axes method (PCA) or minimum bounding rectangularbox





Minimum bounding rectangle (minimum bounding box):

Smallest rectangle containg every pt. in the shape

- Eccentricity: E=L/W

L: length of bounding box W: width of boundingbox

- Elongation: Elo = 1 - W/L Elo 2 [0,1]

Circle of square (symmetric): Elo = 0 Shape w/ large aspect ratio: Elo close to 1

Region descriptors: circularity ratio

Circularity ratio: How similar to a circle is the shape 3 definitions:

Circularity ratio 1: $C_1 = A_s/A_c$ = (Area of a shape)/(Area of circle) where circle has the same perimeter

Circularity ratio 2: $C_2 = A_s/p^2$ (p = perim of shape) Area to squared perimeter ratio.

Region descriptors: rectangularity

Rectangularity represents how rectangular a shape is, i.e. how much it fills its minimum bounding rectangle:

$$Rectangularity = A_S / A_R$$

where A_S is the area of a shape; A_R is the area of the minimum bounding rectangle.

What is rectangularity for a square? Circle? Ellipse?

Region descriptors: convexity

2.8 Convexity

Convexity is defined as the ratio of perimeters of the convex hull $\mathcal{O}_{Convexhull}$ over that of the original contour $\mathcal{O}$ [7]:

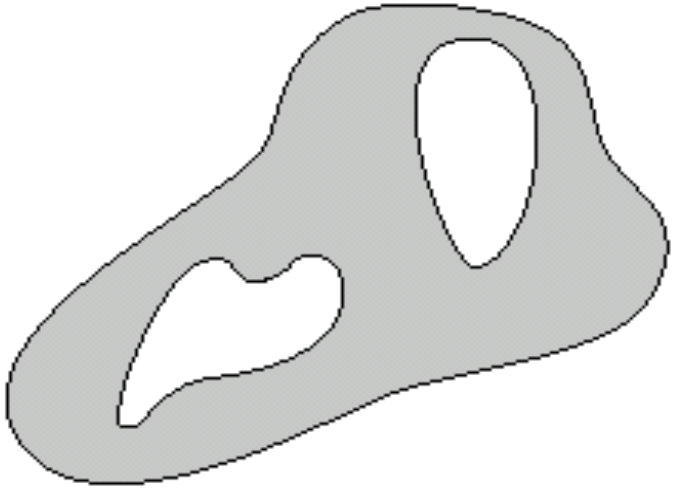
$$Convexity = \frac{\mathcal{O}_{Convexhull}}{\mathcal{O}} \tag{13}$$

The region R^2 is a convex if and only if for any two points P_1, $P_2 \in R^2$, the entire line segment P_1P_2 is inside the region. The convex hull of a region is the smallest convex region including it. In Figure 6, the outline is the convex hull of the region.

Illustration of convex hull

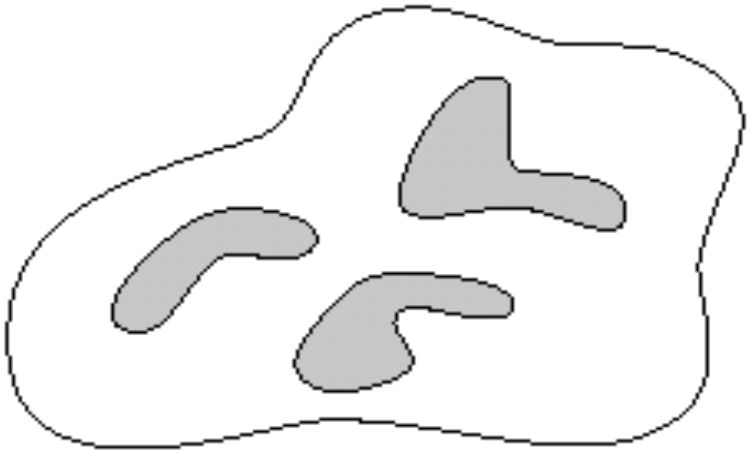
Region descriptors: topological descriptors

Topological property 1:
the number of holes (*H*)



A region with two holes.

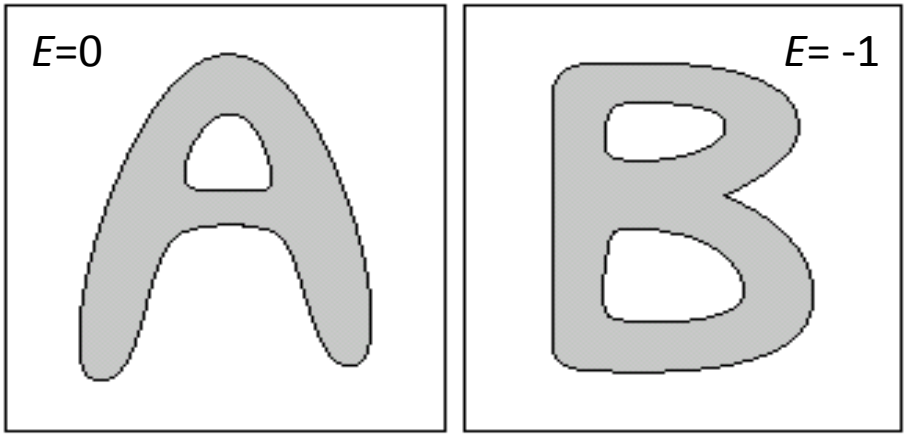
Topological property 2: the number of connected components (*C*)

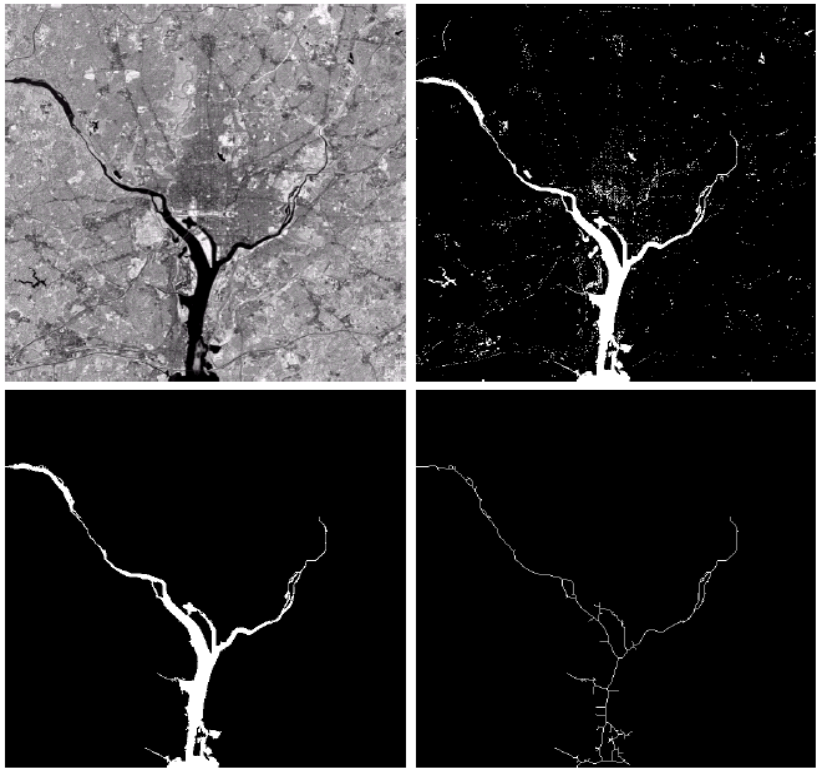


A region with three connected components.

Region descriptors: topological descriptors

Topological property 3:
Euler number: the number of connected components subtract the number of holes $E = C - H$





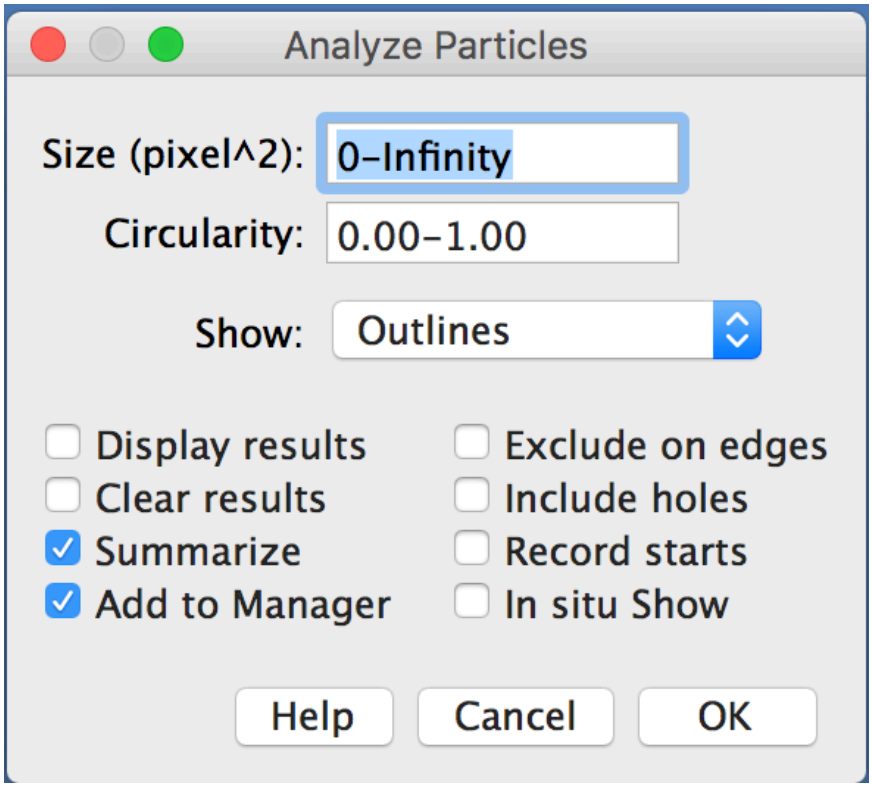
a b
c d

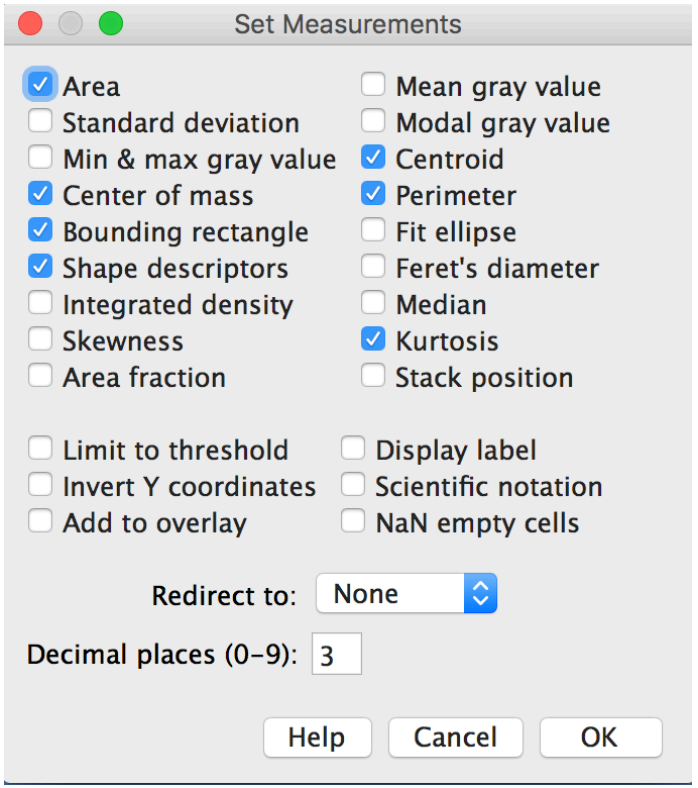
FIGURE 11.21
(a) Infrared image of the Washington, D.C. area. (b) Thresholded image. (c) The largest connected component of (b). Skeleton of (c).

Topological property 4:
the largest connected component.

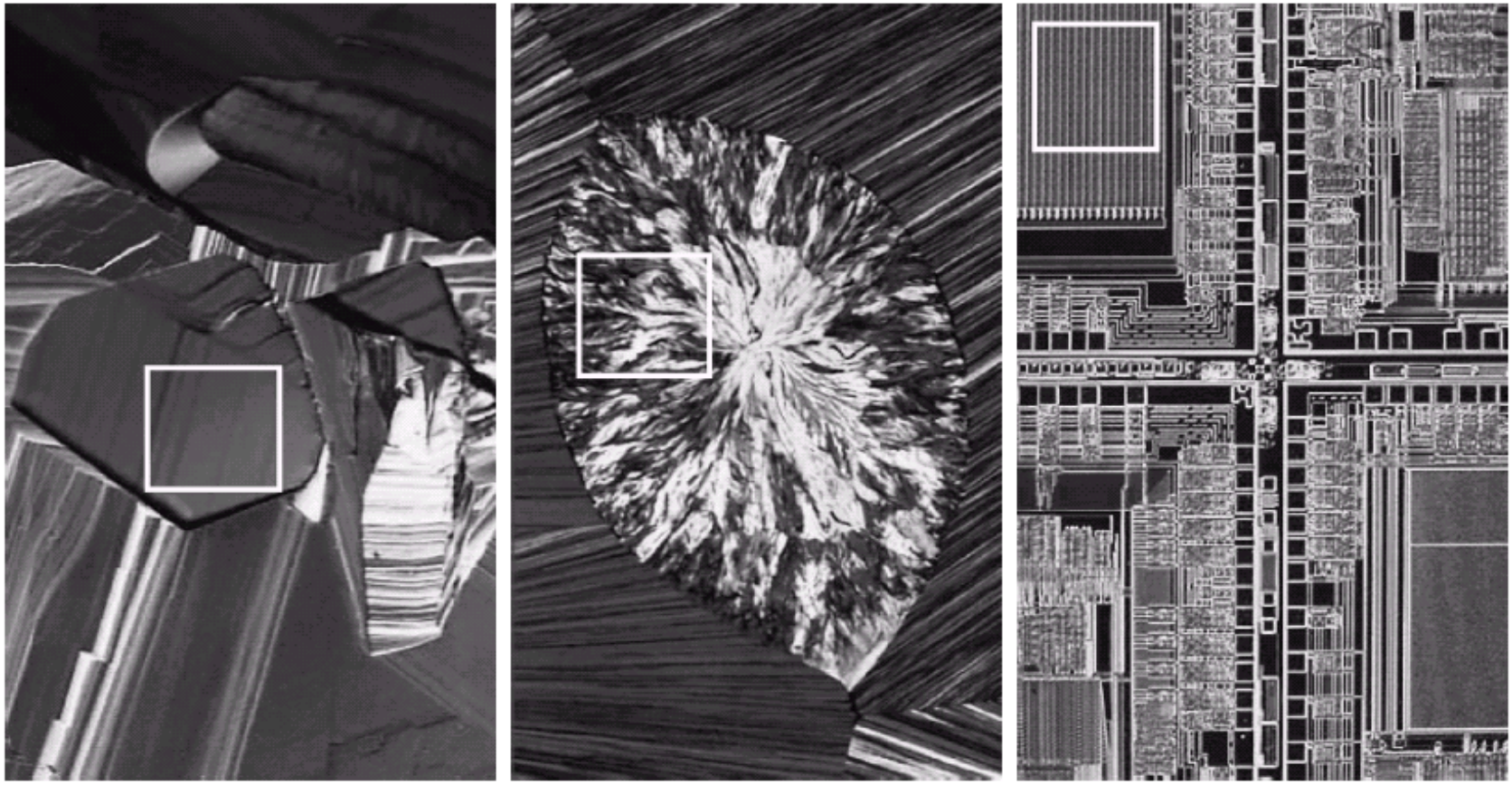
Simple descriptors with Image J

- Before any analysis he is necessary to set measures to perform. ImageJ has a wide range of settings in the menu Analysis> Set Measurements
- The function Analyze >Measure performs the measurements and displays the result in a table ("Results" window) that is possible to export by File>Save as or simply copy and paste.
- The "Particles Analysis" Analyze>Analyze Particles must be used sure a binary image or an image 8-bit thresholded by Image Feature> Adjust> Threshold.





Texture



a b c

FIGURE 11.22 The white squares mark, from left to right, smooth, coarse, and regular textures. These are optical microscope images of a superconductor, human cholesterol, and a microprocessor. (Courtesy of Dr. Michael W. Davidson, Florida State University.)

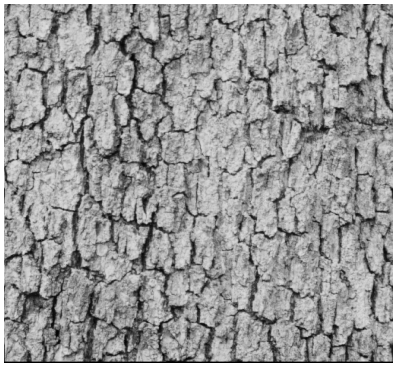
Texture

Texture is usually defined as the smoothness or roughness of a surface.
In computer vision, it is the visual appearance of the uniformity or lack of uniformity of brightness and color.

There are two types of texture: random and regular.

- Random texture cannot be exactly described by words or equations; it must be described statistically. The surface of a pile of dirt or rocks of many sizes would be random. Random texture is analyzed by statistical methods.

- Regular texture can be described by words or equations or repeating pattern primitives. Clothes are frequently made with regularly repeating patterns. Regular texture can be analyzed by spectral (Fourier) methods.



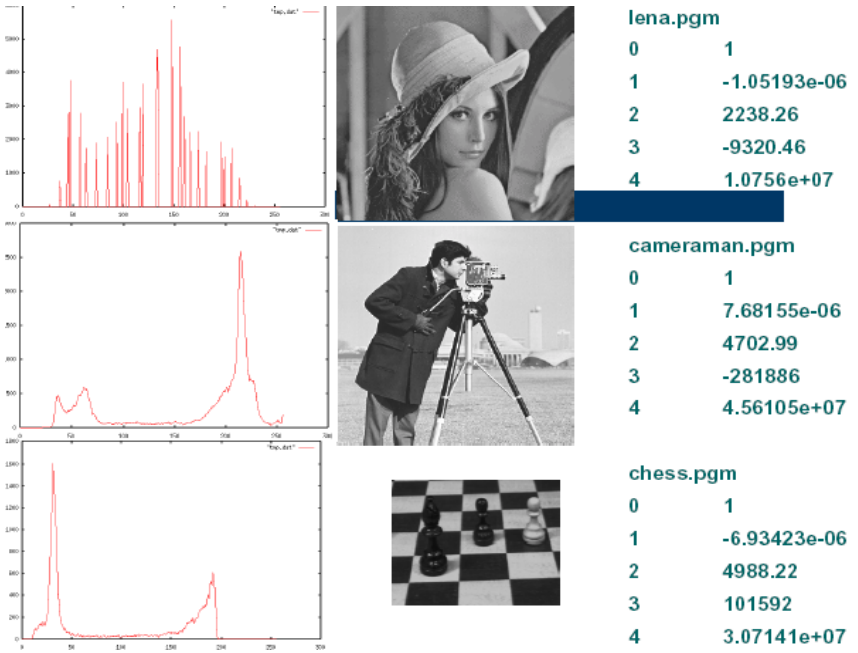
Statistical methods

Use the moment of gray-level histogram to describe the texture

$$\mu_n(z) = \sum_{i=1}^{l} (z_i - m)^n p(z_i)$$

$$p(z_i) : probability$$

$$m = \sum_{i=1}^{l} z_i p(z_i) : mean$$



lena.pgm

0	1
1	-1.05193e-06
2	2238.26
3	-9320.46
4	1.0756e+07

cameraman.pgm

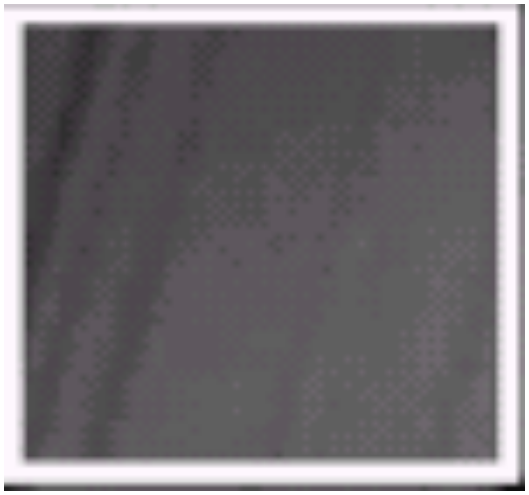
0	1
1	7.68155e-06
2	4702.99
3	-281886
4	4.56105e+07

chess.pgm

0	1
1	-6.93423e-06
2	4988.22
3	101592
4	3.07141e+07

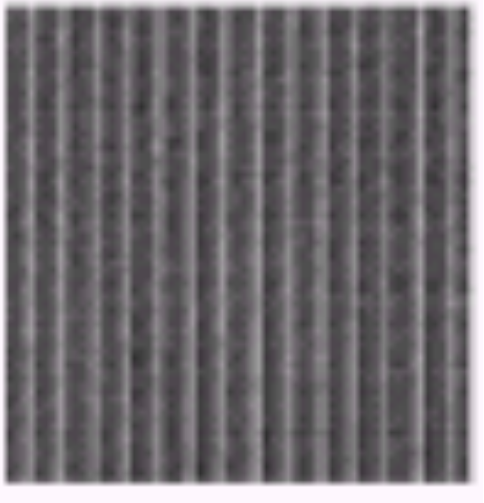
- Order 0 is just the number of points in the data.
- Order 1 is the sum and is used to find the average.
- Order 2 is related to the variance, and can be used to describe the smoothness
- Order 3 is the skew of the data, and measures skewness of the histogram
- Higher orders can also be used, but don't have simple meanings (as order (kurtosis) that measures the flatness of the histogram

Statistical methods



Smooth

Coarse

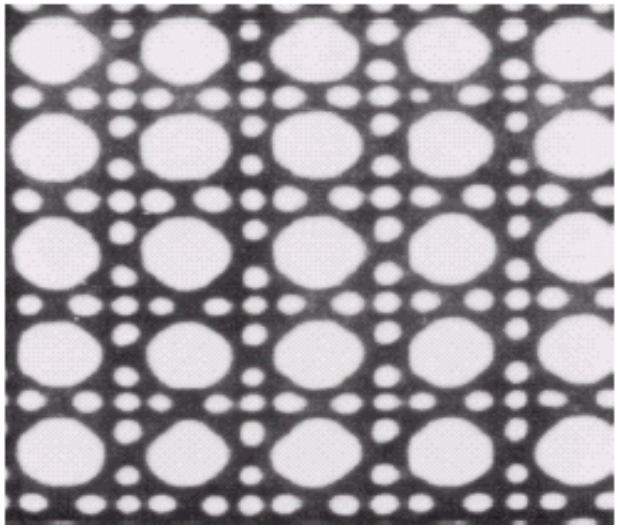


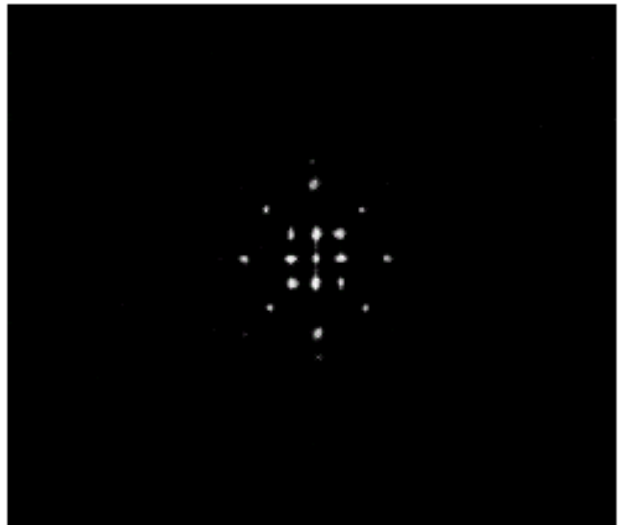
Regular

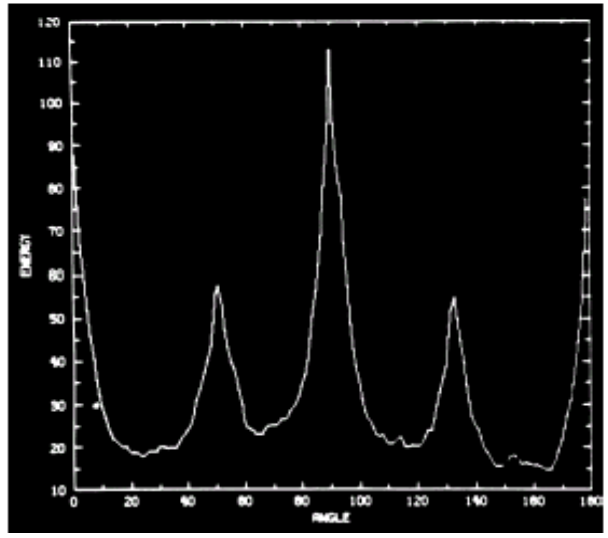
TABLE 11.2
Texture measures for the subimages shown in Fig. 11.22.

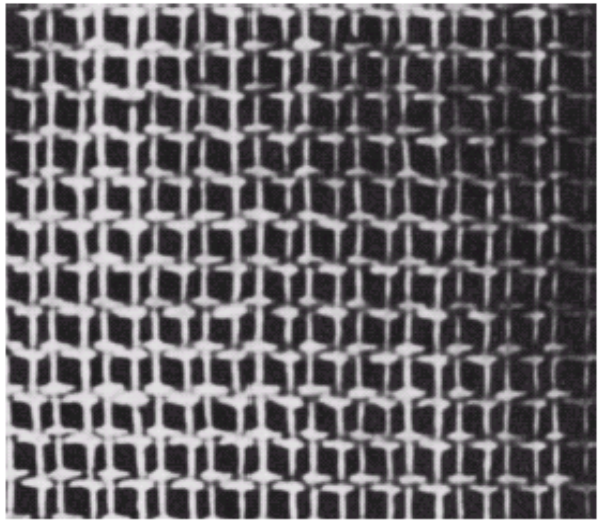
Texture	Mean	Standard deviation	R (normalized)	Third moment	Uniformity	Entropy
Smooth	82.64	11.79	0.002	−0.105	0.026	5.434
Coarse	143.56	74.63	0.079	−0.151	0.005	7.783
Regular	99.72	33.73	0.017	0.750	0.013	6.674

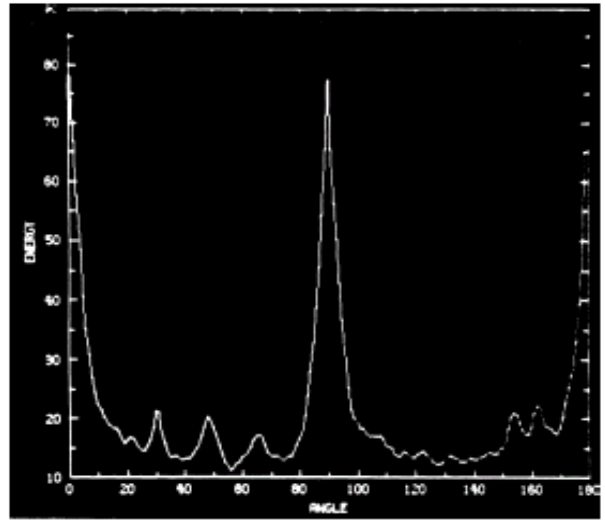
Spectral methods











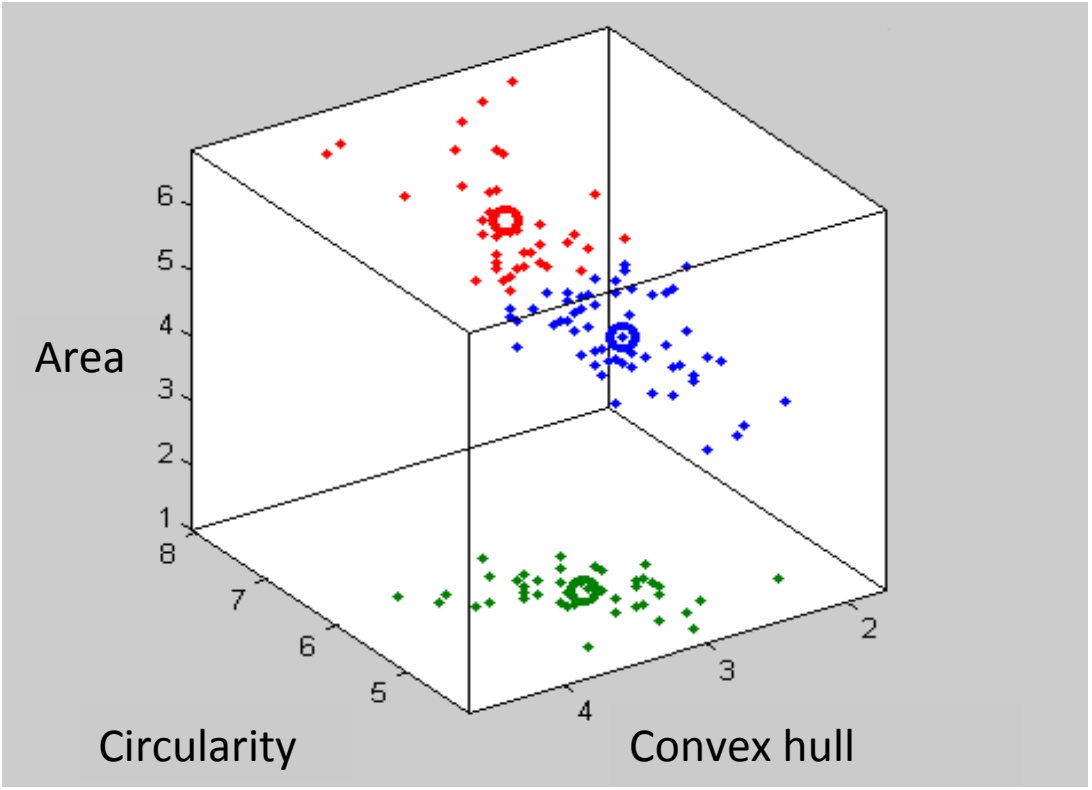
a b
c d
e f

FIGURE 11.24 (a) Image showing periodic texture. (b) Spectrum. (c) Plot of $S(r)$. (d) Plot of $S(\theta)$. (e) Another image with a different type of periodic texture. (f) Plot of $S(\theta)$. (Courtesy of Dr. Dragana Brzakovic, University of Tennessee.)

Conclusion

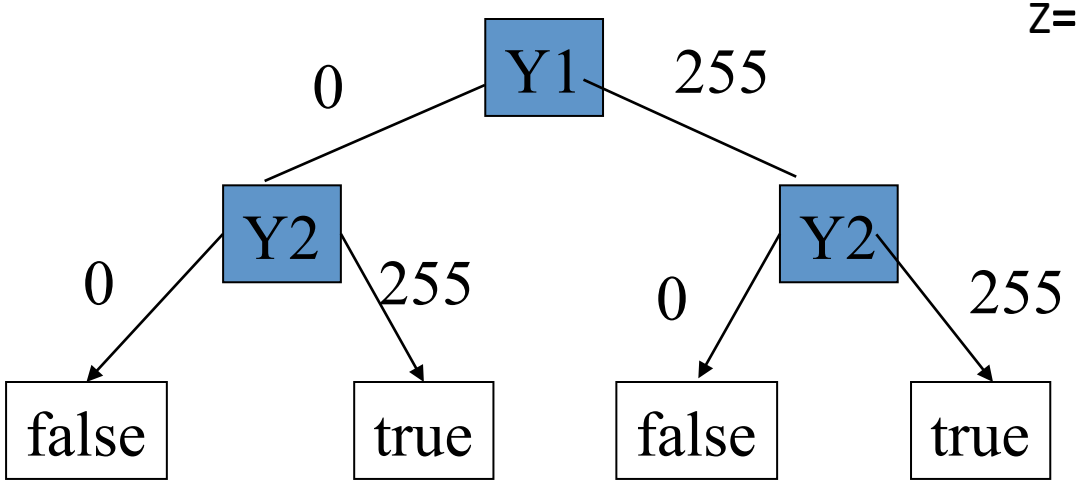
The shapes recognition will consist in:

- choosing the best descriptors/features for a given application,
- and to use classifier on these descriptors/features to recognize objects in the descriptors/features space



Features space

More advanced classifier: Random forest

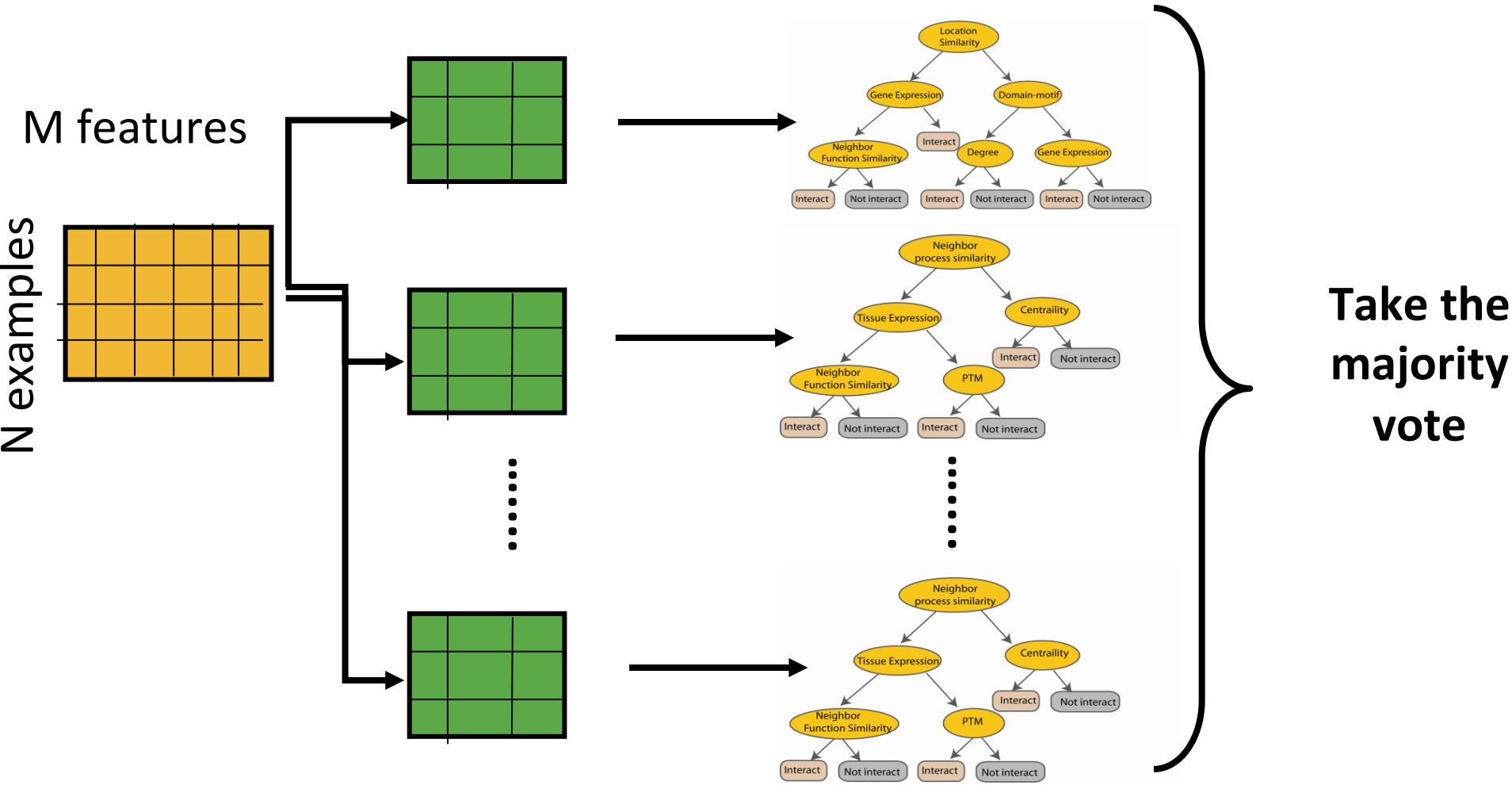


Z=((Y1 and Y2) or ((not Y1) and Y2))

Select best combination (information gain in the Shannon way)

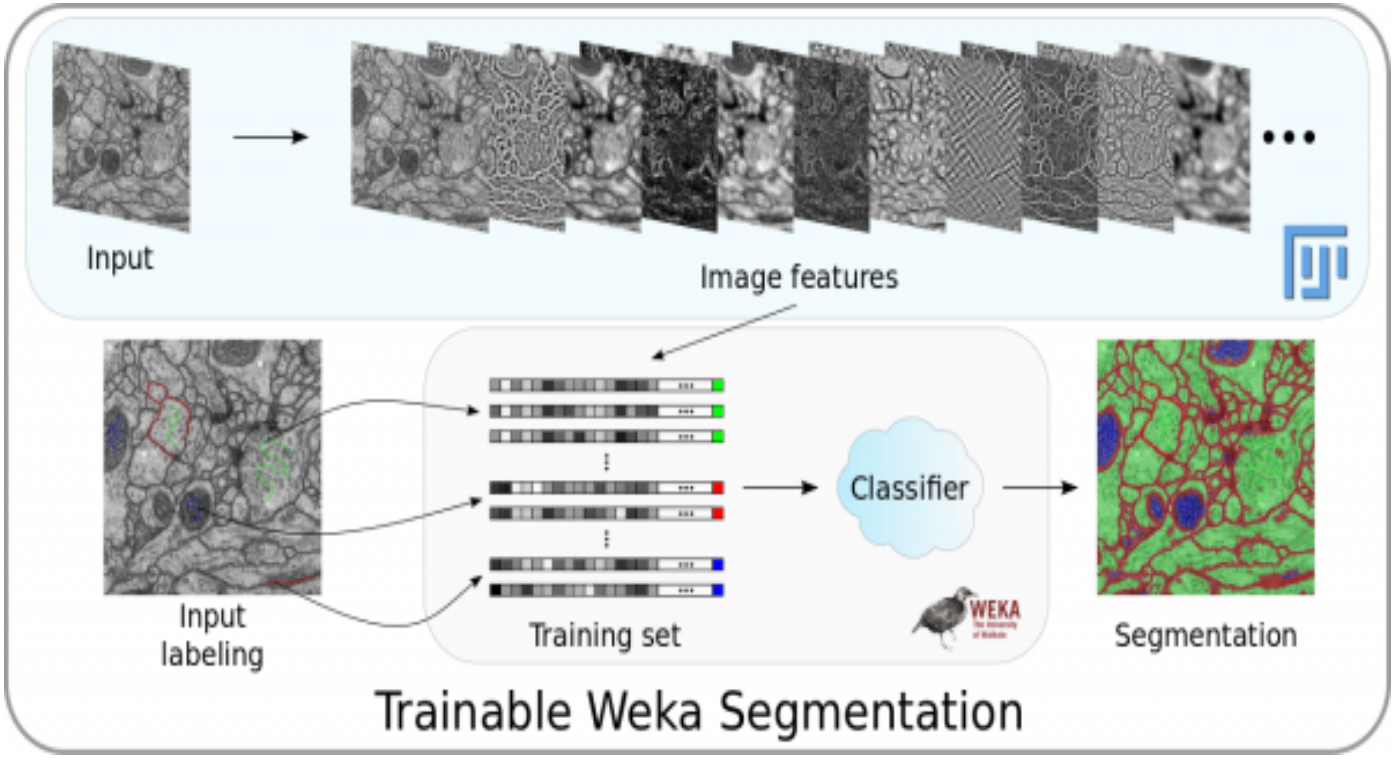
Tree bagging +feature sampling : generate multiple trees with randomly picked features using randomly picked part of the annotated data.

More advanced classifier: Random forest



Hands on with WEKA ImageJ plugin

Please annotate some of your data



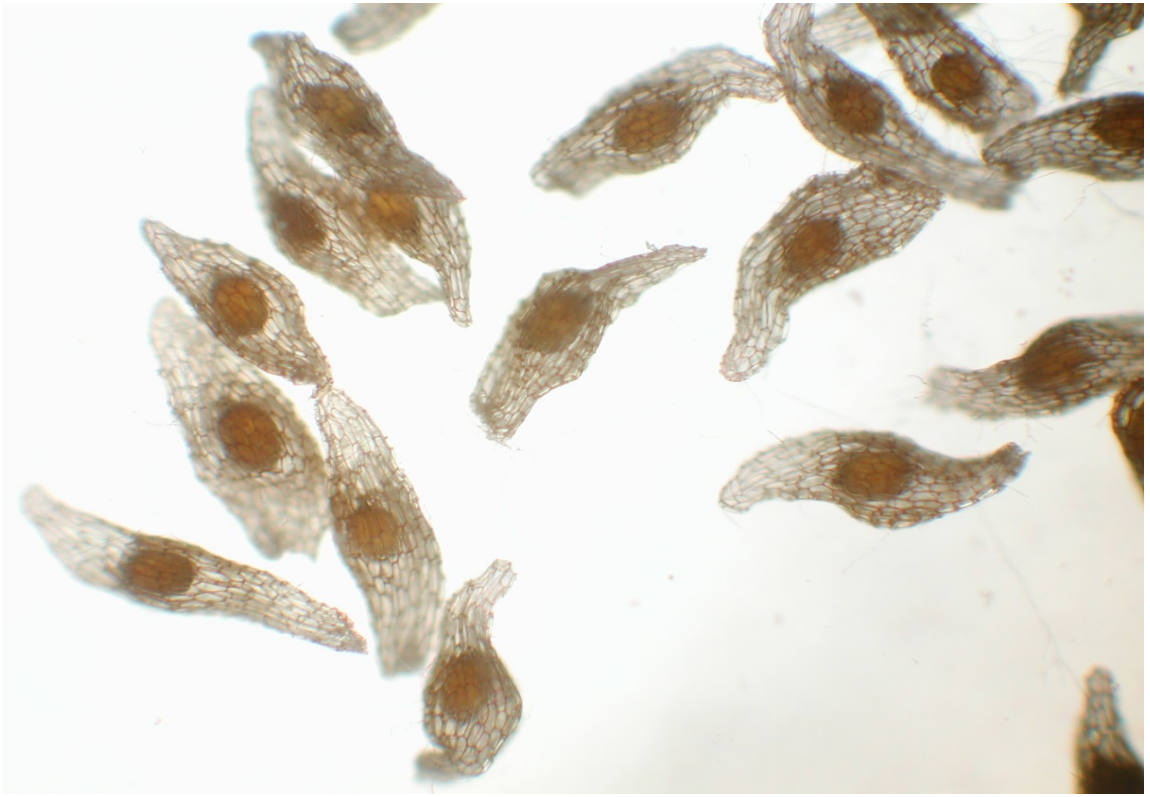
Prerequested knowledges

- in bio science: yes
- In data science: none

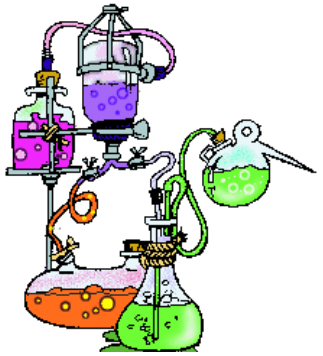
Hands on n°10

1) Load seedsincoat.bmp.

2) After weka classification, count the number of seeds



TAKE HOME MESSAGE n°1
ABOUT IMAGE PROCESSING



Analogy / similarity with analytical chemistry

Take some input compound, filter it in many ways

- Through away what you don’t want
- Extract what you want
- Quantify and characterize the amount of what you extracted

In automated industrial mode, it takes the form of pipeline

Low level: mainly denoising if necessary

- Process/Filters/ Mean ou Process/FFT/ Bandpass Filter
- Set the size of the filter according to strcutures of interest to preserve

Middle level: enhancement of structure of interest, morphomathematic filtering

- Edges: Process/ Find Edges (high sensitivity to noise)
- Region: Process/Filters (Mean, Variance, Min, Max, …) according to what is invariant in the region
- Binarization: image/Adjust/Threshold → connexe components (particles)
- If noise: erode/dilate/open/close via Process/Binary

High level: counting, measurement, classif.

- Measurement: Analyse/Set measurements then Analyse/Analyse particle
- Use of results in statistical analysis software

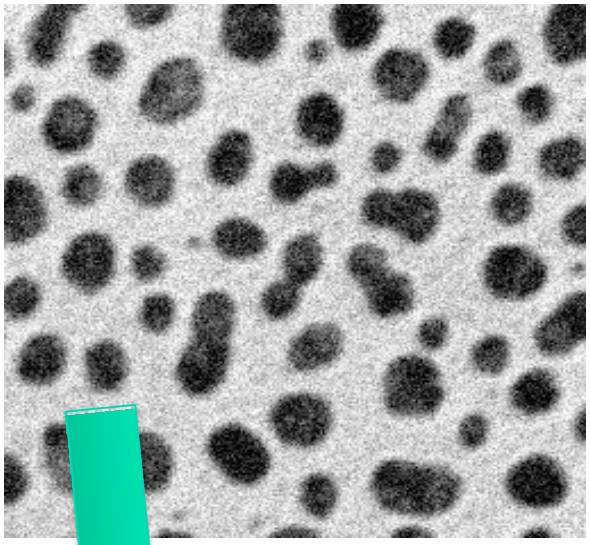
Do not forget to firstly observe the image

To set up parameters used to extract information (filter size, choice of enhancement, edge or region approach, etc.), it requires to have a quantitative idea of the content of the image.

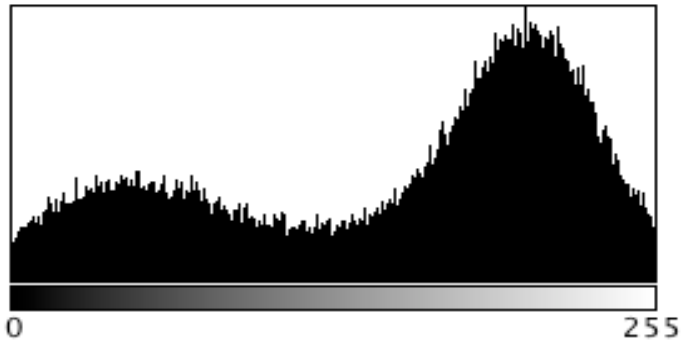
So that, some tools are useful ot oberve quantitatively in ImageJ :

- Analyse/Set scale: to convert pixel in metric unit métrique
- Analyse/Calibrate: to convert grey-level intensity in physical unit (if possible)
- Analyse/Histogram: to « see » regions
- Analyse/Plot profile: to « see » edges

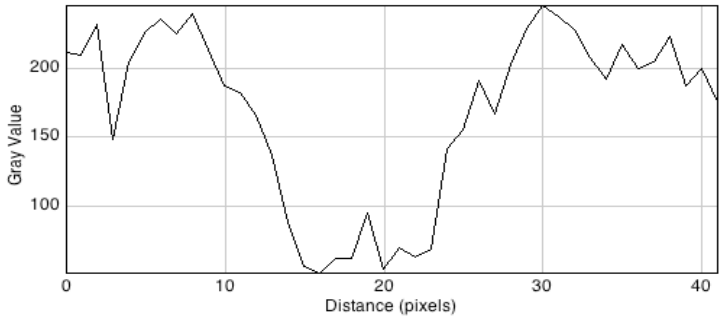
Illustration



2 structures → 2 modes in the histogram
Well-contrasted image bien contrastée



Plot profil throug a cell



Typical size of a cell: 10 pixels diamater
Some noise (maybe gaussian)

Histogram in the background area

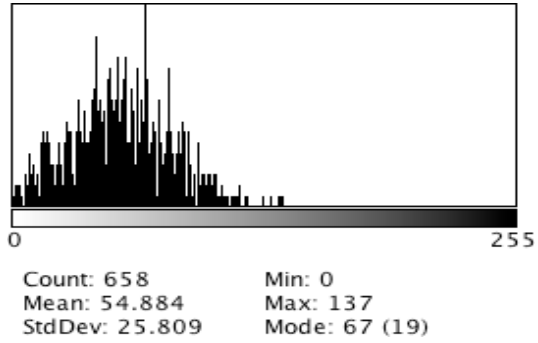
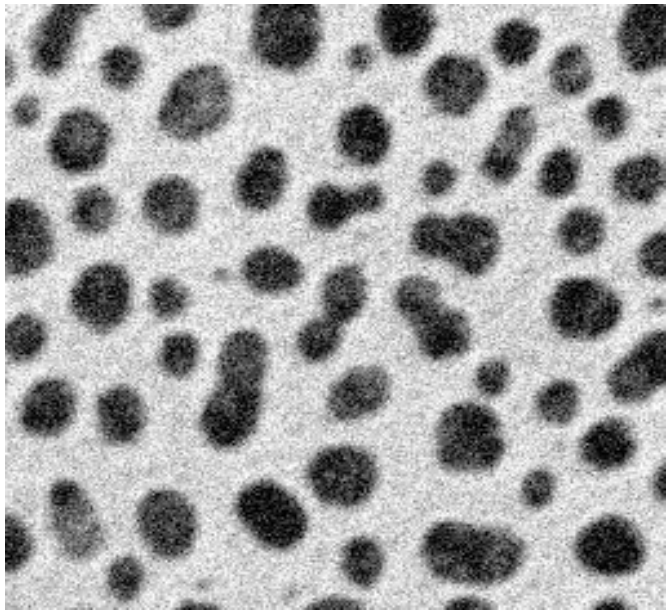
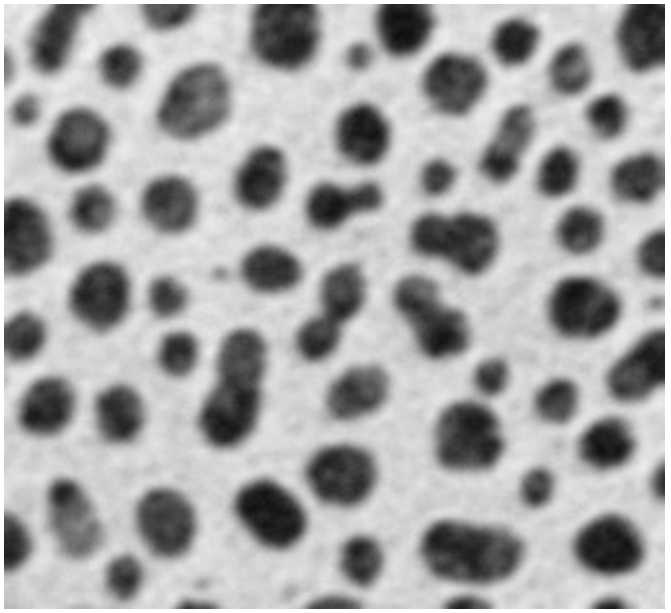


Illustration – Low level processing

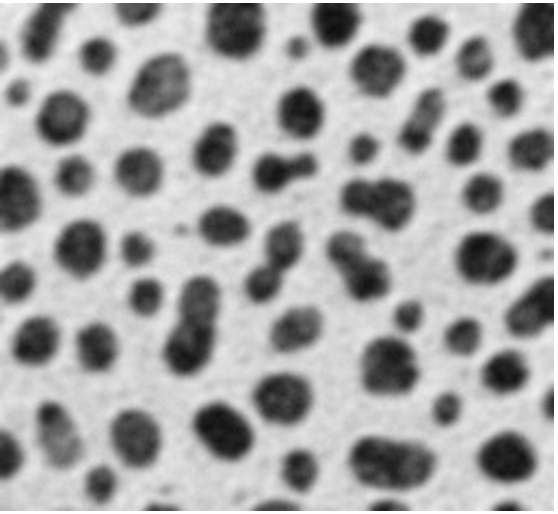


Average filter (mean)
Size 3x3

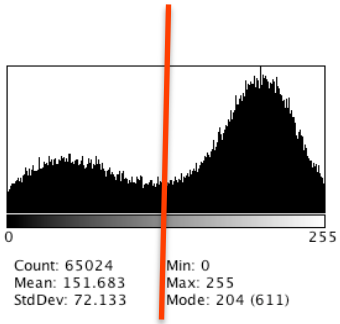


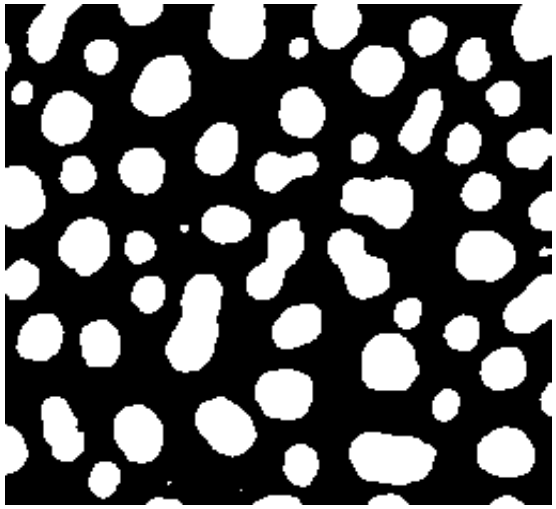
Average filter
Size 20x20

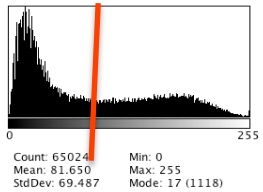
Illustration – Middle level processing



Region approach
(thresholding)

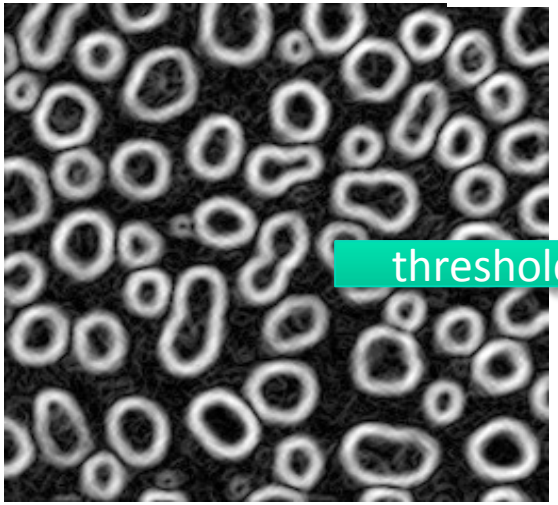




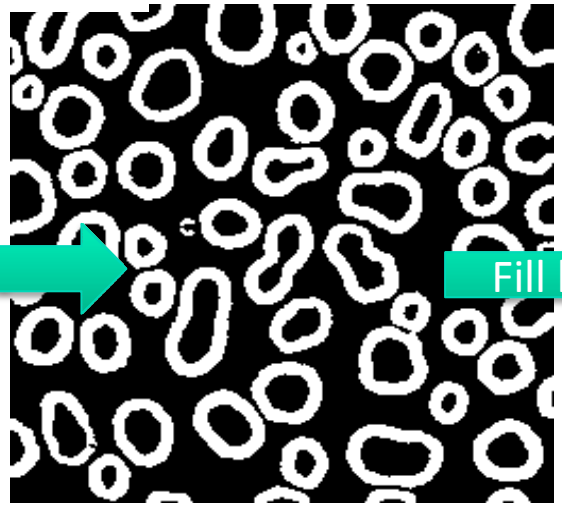


Edge image
histogram

Edge approach
Find edge



threshold



Fill holes

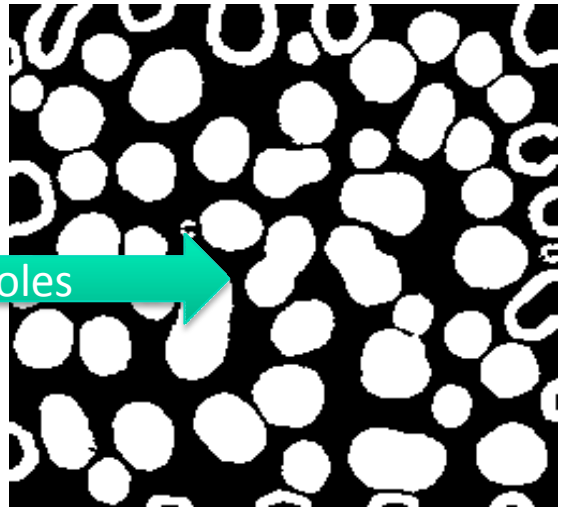
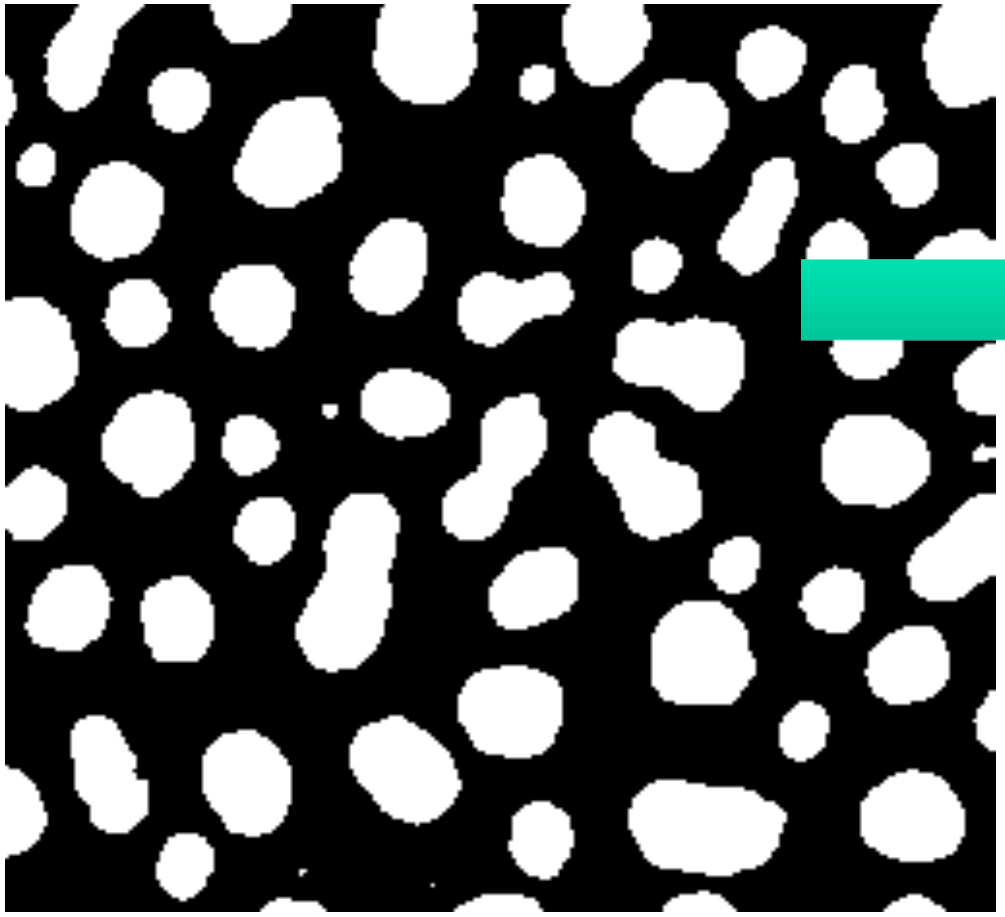
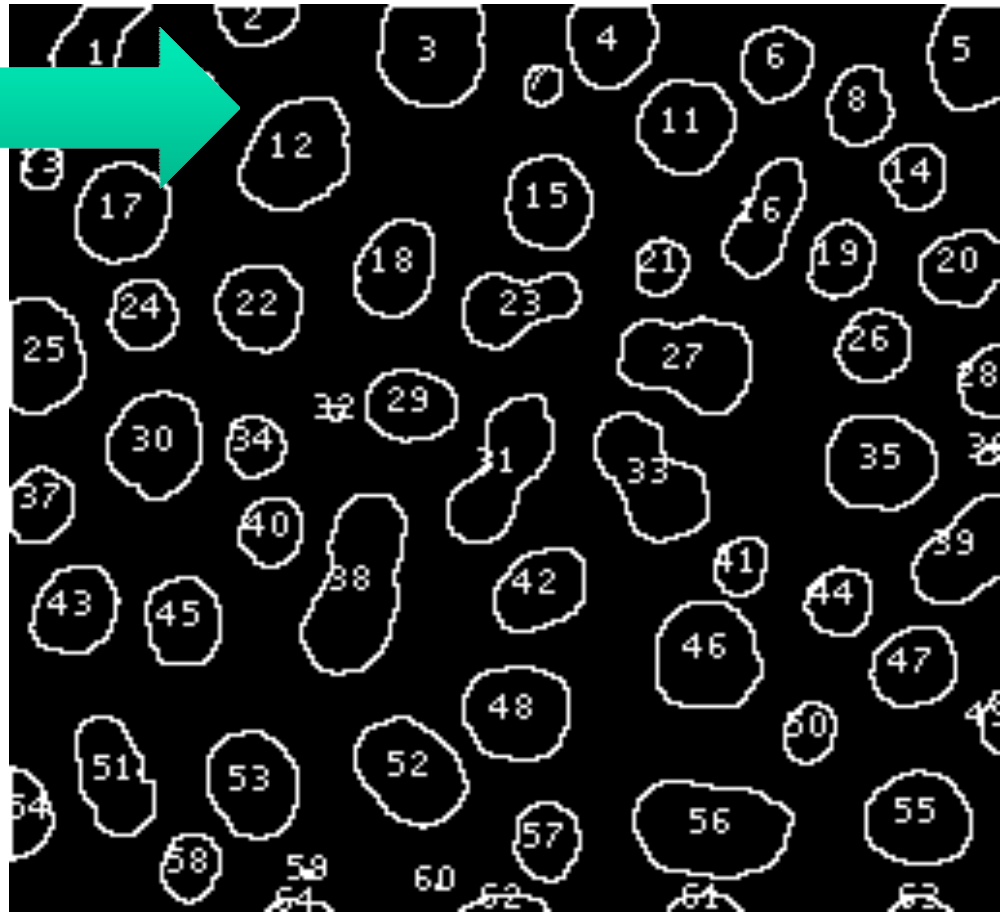


Illustration – High level processing





Counting and morphometric characterization

TAKE HOME MESSAGE n°2
ABOUT IMAGE PROCESSING

You are facing a new image processing problem and you are not an expert in image processing

Test simple ideas presented here :

- **Try take home message n°1 ☺**
- **Try supervised ranfom forest classification: Weka**
- **If it fails… call an expert:**

Etienne.belin@univ-angers.fr
David.rousseau@univ-angers.fr