Digital image processing

Aline Roumy
Inria, Rennes, team Sirocco

The course: you and me!

- Who am I? Researcher at Inria, Rennes.

  Research topic:

  communication of visual data (image and video)

  Research areas:

  image processing and signal processing,

  information theory.

- Who are you?

Course organization

- introduction to image processing: 3 lectures of 1:10 each
- When $\boxed{\textbf{Quiz}}$ in the slides, answer on Socrative
  https://b.socrative.com/login/student/
  Room Name: ALINER

  No need to give your true name (no marks for the answer)
- 2 computer lab sessions: 3:30 each in matlab
  - general exercises on image processing (seen in the course)
  - one project
- **evaluation**: during the computer lab session(s). You will get marks for the two sessions + multiple choice questions test.

Part 1 - General introduction

What is image processing?

...a part of Computer Vision (*vision par ordinateur*)

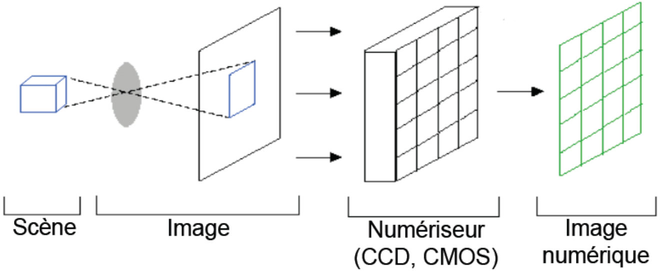
- **What it is?**
  Technology to design machines that can see.

- **First level:** vision
  acquisition of an image thanks to a chain that contains optical devices and sensors

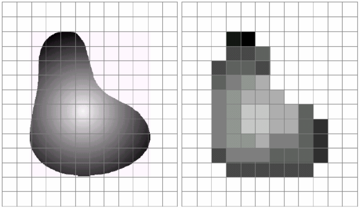
- **Second level: image processing**
  modify the image in order to highlight some elements of interest (objects, edges)

- **Third level:** interpretation, decision making
  use artificial intelligence (AI) to do
  - pattern recognition (*reconnaissance*): identify some shapes in the image (face, coin, ...)
  - image restoration (noise removal, object removal...)
  - 3D scene reconstruction

First level: acquisition



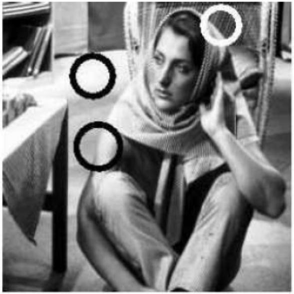
digitalize = sample (finite number of points)
+ quantize (finite number of possible gray levels)

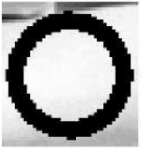


Source: T. Hassner. Computer Vision

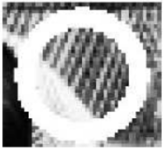
Second level: image processing

Detect “simple” elements of an image

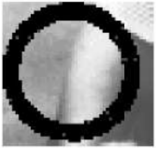




Une zone homogène



Une texture



Un contour

Computer vision: a way to extend human perception?

- **Yes!**
  - because the human visual system is limited
    - in frequencies (wavelength),
    - in angle,
    - in acquisition frequencies (number of images per second): high speed camera (camera haute cadence) with thousands images per second...
- **and No!** Computer vision systems are still limited...
  - a machine is less intelligent than our brain... but the battle keeps going: (Google's AI Wins Against Go Genius, March 2016)
  - limited computational capacity of a machine vs our brain
  - we use much more information (not only vision) to make decision

Application of Computer Vision (vision par ordinateur): Machine Vision (vision industrielle)

- **Use**:
  automatic inspection, (quality inspection)
  process control,
  and robot guidance

- **Works well because**...
  camera designed for a specific task
  controlled environment

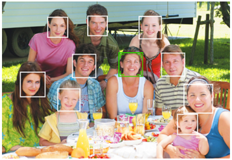
Application of Computer Vision (vision par ordinateur): natural images

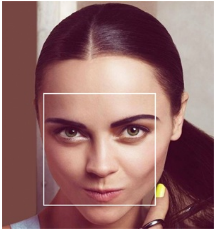
Very challenging (still under research) because **Un**controlled environment:

- Acquire an image
  Correct color balance,
  Reconstruct image from projections
- Prepare for display or printing
  Adjust image size, Color mapping
- Facilitate picture storage and transmission
  Efficiently store an image in a digital camera,
  Send an image from mobile
- Enhance and restore images
  Touch up personal photos
- Extract information from images
  Read 2-d bar codes, Character recognition
- Many more ... image processing is ubiquitous

Application

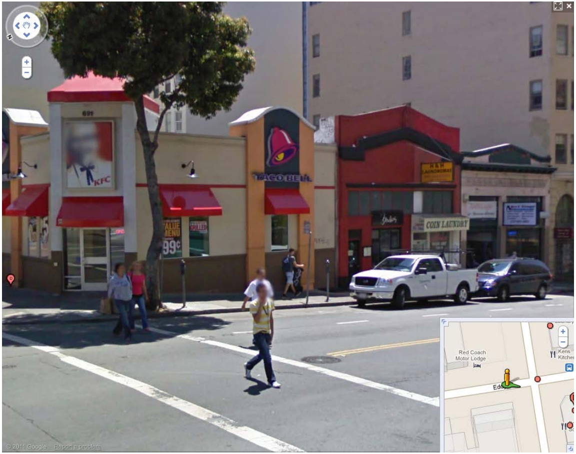
Face detection

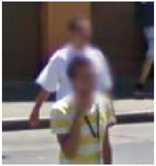




Application

Face blurring for privacy protection

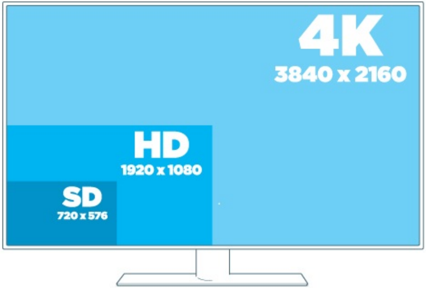




Application

Object removal (Movie production)

Application

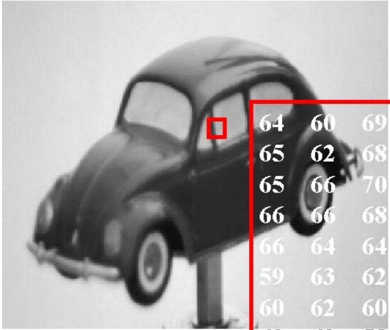


Super Resolution
a.k.a. Upscaling

Seiki's U-Vision Upscaling HDMI Cable (CES 14)

input upsizing super-resolution

What is an image?

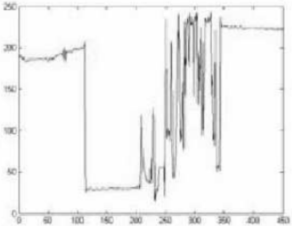


64	60	69	100	149	151	176	182	179
65	62	68	97	145	148	175	183	181
65	66	70	95	142	146	176	185	184
66	66	68	90	135	140	172	184	184
66	64	64	84	129	134	168	181	182
59	63	62	88	130	128	166	185	180
60	62	60	85	127	125	163	183	178
62	62	58	81	122	120	160	181	176
63	64	58	78	118	117	159	180	176

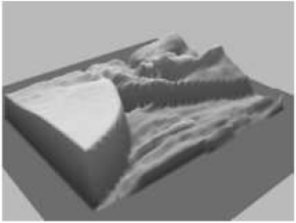
- **Digital image**: 2D array $I[x, y]$
- Each element is called a pixel or pel (from "picture element")
- Gray levels - 1 byte = 8 bits $\in [0, 255]$,
  - 0 - black
  - 255 - white

What is an image?

a row of an image: is a curve



the whole image: a surface



Images as matrices

- **Digital image**: is a 2D array (or matrix) with *Nrows* rows and *Ncolumns* columns with elements $I[i,j] \in \{0,\dots,255\}$

TP In MATLAB, an image is a matrix "**I**" with elements $I(i,j) \in \{0,\dots,255\}$: () NOT [ ]
for $i = 1,\dots,Nrows$, $j = 1,\dots,Ncolumns$.

TP In Matlab, $I(:,i)$ denotes the i-th column of I.

TP Warning: type of the 2D array (uint8 or double)
To display an image stored as uint8 imshow(I)
To display an image stored as double imshow(I/255)
Look at the type of your matrix in the Matlab's workspace window → *see Matlab*

What is an image?

Reading and displaying order of $I[x, y]$ is:

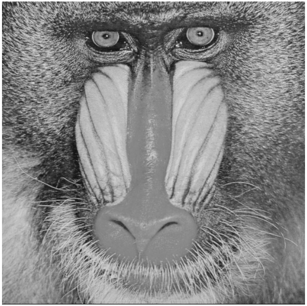
as a matrix, not as the cartesian coordinate plane

x = row index / y = column index

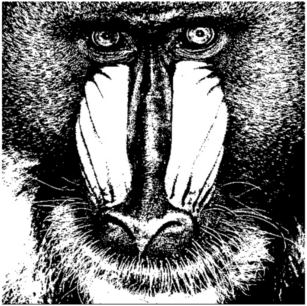
BUT the definition is Nb_columns × Nb_rows (convention)

y =	58	59	60	61	62	63	64	65	66	67					
x = 41	210	209	204	202	197	247	143	71	64	80					
42	206	196	203	197	195	210	207	56	63	58					
43	201	207	192	201	198	213	156	69	65	57					
44	216	206	211	193	202	207	208	57	69	60					
45	221	206	211	194	196	197	220	56	63	60					
46	209	214	224	199	194	193	204	173	64	60					
47	204	212	213	208	191	190	191	214	60	62					
48	214	215	215	207	208	180	172	188	69	72					
49	209	205	214	205	204	196	187	196	86	62					
50	208	209	205	203	202	186	174	185	149	71	63	55	55	45	56
51	207	210	211	199	217	194	183	177	209	90	62	64	52	93	52
52	208	205	209	209	197	194	183	187	187	239	58	68	61	51	56
53	204	206	203	209	195	203	188	185	183	221	75	61	58	60	60
54	200	203	199	236	188	197	183	190	183	196	122	63	58	64	66
55	205	210	202	203	199	197	196	181	173	186	105	62	57	64	63

Some types of images



Gray level image
$I[x,y] \in [0,255]$



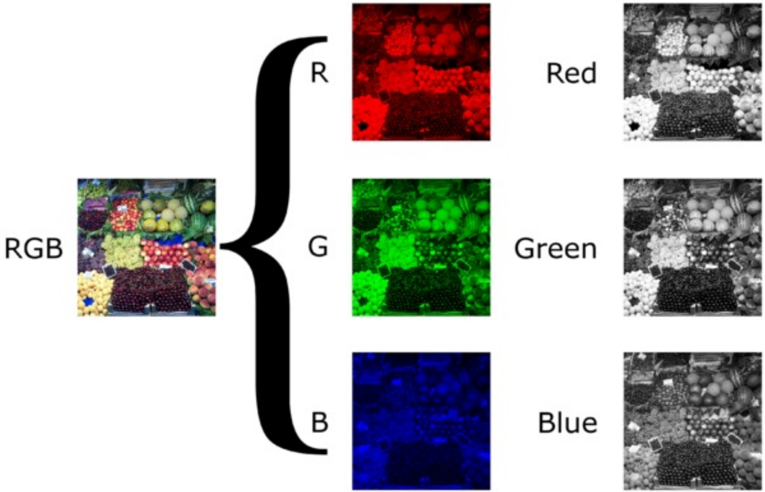
Gray level image
$I[x,y] \in \{0,1\}$

Color image
$I_R[x,y] I_G[x,y] I_B[x,y]$

and many more types (3D, High Dynamic Range (HDR)...)

$\boxed{\textbf{Quiz 1,2,3}}$

Color Components



but there exists other representations for color images (Luminance Chrominance)

Outline of the course

- Part 1 - General introduction
- Part 2 - Image transformation
  - 3 kinds of transformation
- Part 3 - Point-to-point transformation
  - Geometric transformation
  - Contrast enhancement/reduction
  - Gray level histograms
  - Quantization, binarization
- Part 4 - Local to point transformation
  - Morphological image processing
  - Image filtering, convolution
    - LOW pass filtering
    - HIGH pass filtering
    - Downsampling and Interpolation/Upsampling
  - Segmentation
- Part 5 - Global to point transformation
- Part 6 - Quality measure
- Part 7 - Machine learning

Part 2 - Image transformation

How to formalize an image transformation
such that it can be automatically processed by a computer?

Image transformation

What is an image?

- from the mathematical perspective:
  - an image is a matrix
  - many tools exist to manipulate matrices
- from the human perspective:
  - an image contains semantic information
  - need to interpret the image beyond the luminance values

Transformation: T

$$\begin{array}{rcl} T : [0,255]^{Nrows \times Ncolumns} & \rightarrow & [0,255]^{Nrows' \times Ncolumns'} \\ im[x,y] & \mapsto & IM[u,v] \end{array}$$

- $im=$ input or original image
- $IM=$ transformed image
- x, y and u, v are the spatial coordinates of a pixel

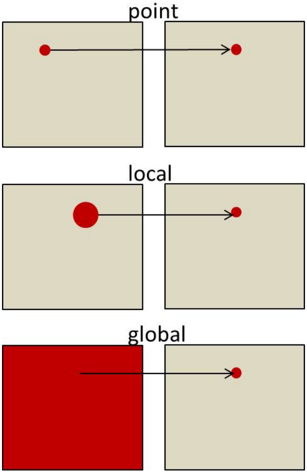
Goal of a transformation:

get a a new representation of the input picture
to ease the extraction of particular properties of the picture.

3 kinds of transformation

There exist 3 types of transformation:

- Point to point transformation: The output value at a specific coordinate depends **only** on one input value but **not necessarily at the same coordinate**;
- Local to point transformation: The output value at a specific coordinate depends on the input values in the **neighborhood** of that same coordinate;
- Global to point transformation: The output value at a specific coordinate depends on **all** the values in the input image.



Note that the complexity increases with the size of the considered neighborhood...

Part 3 - Point-to-point transformation

When a pixel of the new (transformed) image only depends on a single pixel of the input image

Geometric transformation

- Geometric transformation such as rotation, scaling...
- Examples:

original

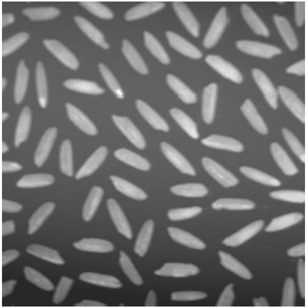
a.

b.

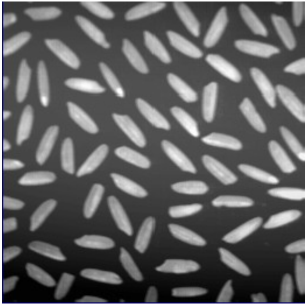
$\boxed{\textbf{Quiz 4,5,6}}$

Contrast enhancement/reduction

Images with different contrast



low contrast



high contrast

Contrast definition

Many definitions for the contrast for a NxM image $f[x, y]$:

- standard deviation (average variation) of the Gray levels

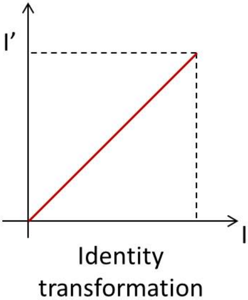
$$C = \sqrt{\frac{1}{N\ M} \sum_{x=1}^{N} \sum_{y=1}^{M} (f[x, y] - mean)^2}$$

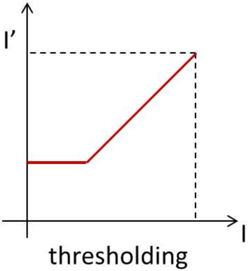
- variation between the minimum and the maximum of the Gray levels

$$C = \frac{\max_{x,y}\left(f[x, y]\right) - \min_{x,y}\left(f[x, y]\right)}{\max_{x,y}\left(f[x, y]\right) + \min_{x,y}\left(f[x, y]\right)}$$

Functions applied to the luminance

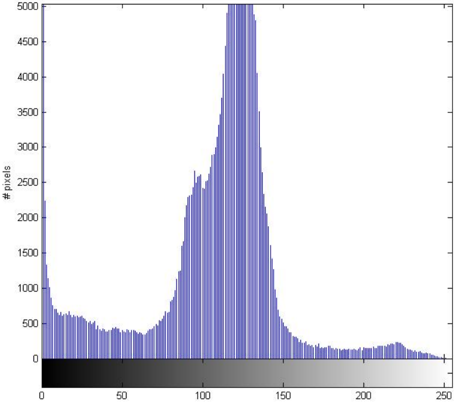
- I: input luminance value I': output luminance value





Quiz 7

Gray level histograms



zebra image

- the histogram represents the "distribution" of the gray level in an image
- $H(k) = \#$pixels having gray level k.

Gray level histograms

- To measure a histogram:
  - For B-bit image, initialize 2^B counters with 0
  - Loop over all pixels x, y
  - When encountering gray level $f[x, y] = i$, increment counter $\#i$
- Normalized histogram can be thought of as an estimate of the probability distribution (pdf) of the continuous signal amplitude
- Use fewer, larger bins to trade off amplitude resolution against sample size.

Quiz 8, 9, 10

Histogram adjustment... with linear transform

Goal: Augment the contrast (i.e. augment the difference between two displayed gray levels).

Idea: Find an affine transformation

$$g = F(f)$$

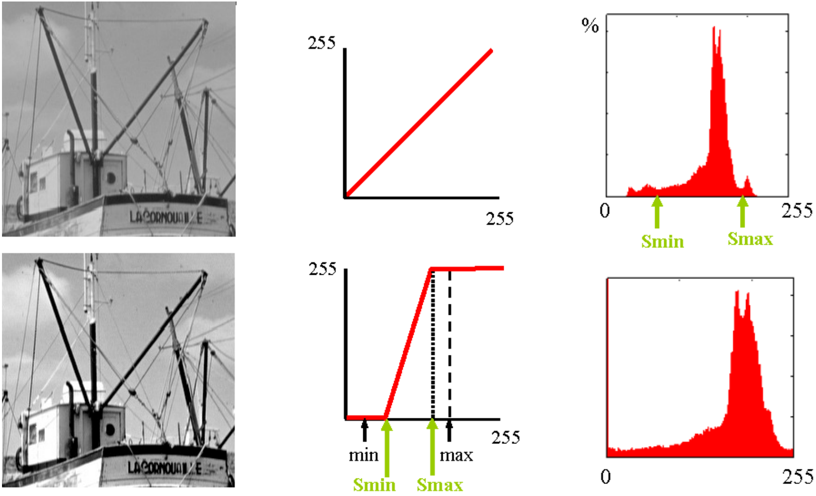
that is applied to each pixel of the input image $f[x, y]$, such that the distribution of gray levels for the output image $g[x, y]$ spans the whole range $[0, 255]$.

$$\begin{array}{rcl} F : [min, max] & \rightarrow & [0, 255] \\ x & \mapsto & F(x) = ax + b \end{array}$$

where x stands for the gray level.
i.e. find a and b.

$\boxed{\textbf{Quiz 11}}$

Histogram adjustement... with linear transform and saturation



Histogram equalization

Idea: Find a non-linear transformation

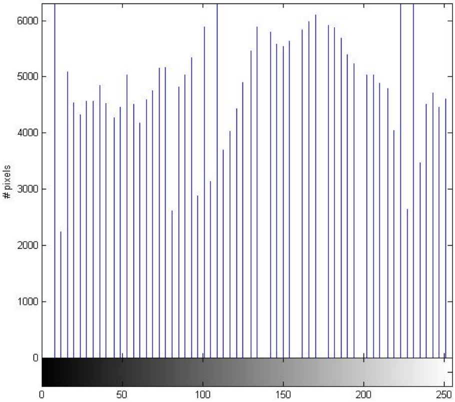
$$g = T(f)$$

that is applied to each pixel of the input image $f[x, y]$, such that a uniform distribution of gray levels results for the output image $g[x, y]$.

Use the cumulative distribution function (cdf) (*Fonction de répartition*).

$\boxed{\textbf{Quiz 12}}$

Histogram equalization example

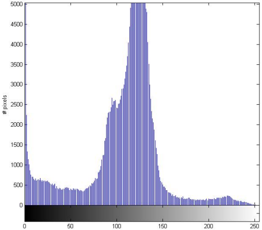


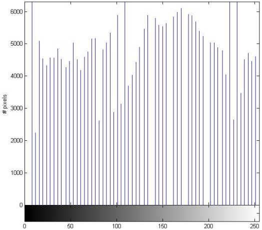
#pixels vs gray level

zebra image

Histogram equalization example

Original image zebra ... after histogram equalization





Quantization... if time permits

The quantization is a process to represent a large set of values with a smaller set.
Scalar quantization:

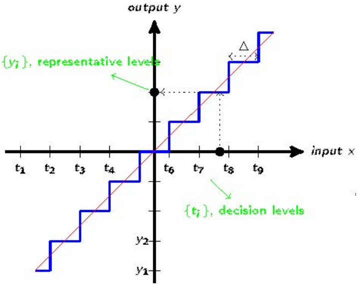
$$\begin{aligned} Q : \mathcal{X} &\rightarrow \mathcal{C} = \{y_i\}_{i=1,2,\ldots,N} \\ x &\mapsto Q(x) \end{aligned}$$

- N is the number of quantization level;
- $\mathcal{X}$ can be continuous (ex: R) or discrete;
- $\mathcal{C}$ is always discrete (codebook,dictionary);
- if $\mathcal{X}$ discrete, $card(\mathcal{X}) > card(\mathcal{C})$;
- Since $card(\mathcal{X}) > card(\mathcal{C})$, Q is non injective, we loose some information (**lossy compression**).

$\boxed{\textbf{Quiz 13}}$

Uniform scalar Quantization (if time permits)

In uniform scalar quantization, the quantization step size is fixed, no matter the signal amplitude is.



Definition:

- The quantization thresholds are uniformly spread: $\forall i \in \{1, ...N\}, t_{i+1} - t_i = \Delta$
- The output values are the centers of the quantization interval: $\forall i \in \{1, ...N\}, y_i = \frac{t_{i+1}+t_i}{2}$

Implementation:

$$Q(x) = \Delta \times \lfloor \frac{x}{\Delta} + 0.5 \rfloor$$

Uniform scalar Quantization

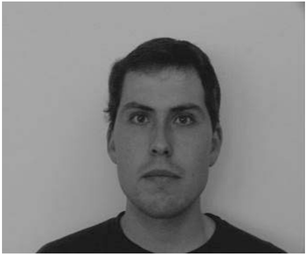
Application to images:

6 bits *4 bits* *3 bits* *2 bits* *1 bit*

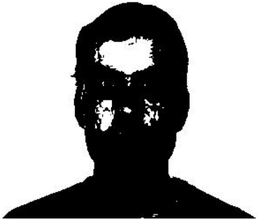
quality resolution (résolution tonale)

$\boxed{\textbf{Quiz 14 15}}$

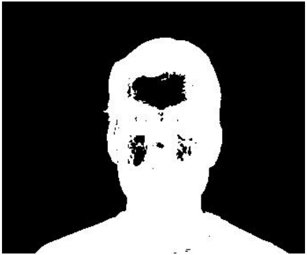
Binarization: by Gray-level thresholding



Original image
Peter $f[x,y]$



Thresholded
Peter $t[x,y]$



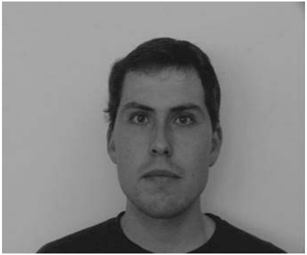
Mask
$1 - t[x,y]$

Thresholding: Loop over all pixels x, y

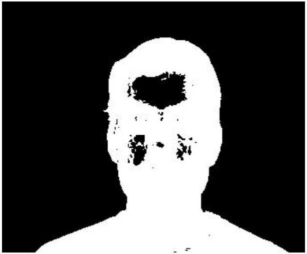
- if $f[x,y] >$ threshold, $t[x,y] = 1$
- else $t[x,y] = 0$.

Mask: logical image
region of interest$=1$
rest$=0$

Binarization: to extract foreground



Original image
Peter $f[x, y]$



Mask
$1 - t[x, y]$

Peter **foreground**
$f[x, y].(1 - t[x, y])$

Thresholding: Loop over all pixels x, y

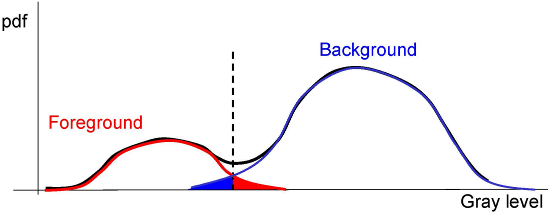
- if $f[x, y] >$ threshold, $t[x, y] = 1$
- else $t[x, y] = 0$.

Mask: logical image
region of interest=1
rest=0
f.*mask=**foreground**

Question: How can holes be filled?

How can holes be filled?

1. By **adjusting the threshold**



   There exists automatic threshold determination.
   In TP, we will do it manually.

2. Decision based on a single pixel is **not sufficient**
   need decision based on **neighborhood**:
   local to point transformation

Part 4 - Local to point transformation

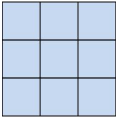
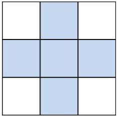
When a pixel of the new (transformed) image depends on a limited set of pixels of the input image

Morphological image processing for **binary** images

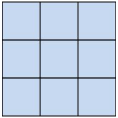
- Binary images are common
  - Source image is binary:
    Text, line graphics
  - Features extracted from a gray-scale/color image
    Edges;
    Presence/absence of some object in the image
- Representation of individual pixels as 0 or 1, convention:
  object (foreground) = 1 (white)
  background = 0 (black)
- Morphological image processing has been generalized to gray-level images via level sets

Structuring element

- **Structuring element = a mask**

3 × 3 square   cross

- **For each pixel** (i, j), apply the mask/structuring element centered in (i, j)



mask

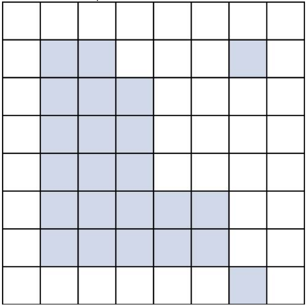


image 8x8

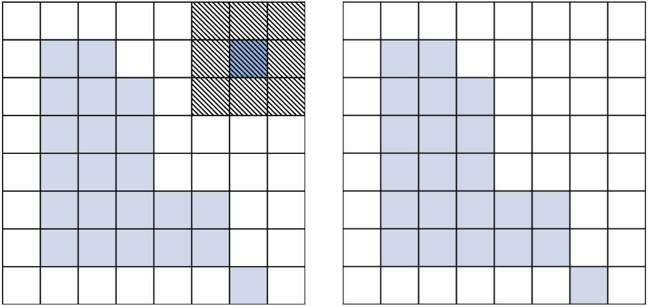
For ease of visualization (unlike convention):

background 0 black → white/ object 1 white → blue

Erosion

Erosion if one pixel under the mask belongs to the background (white), then the centered pixel becomes background (white)

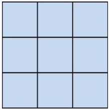
i.e. *erosion of the object*



- Erosion of the object!

Erosion

Quiz 16 Which a or b is the eroded image? The mask is



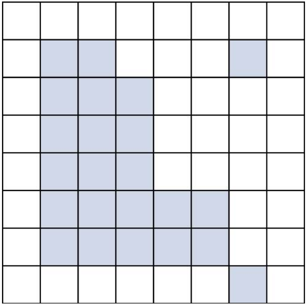
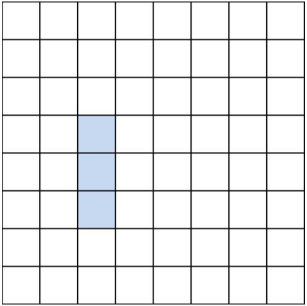
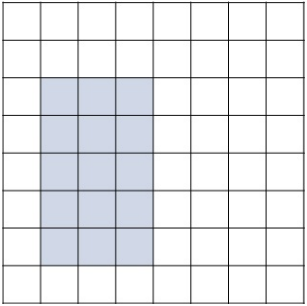


image 8x8



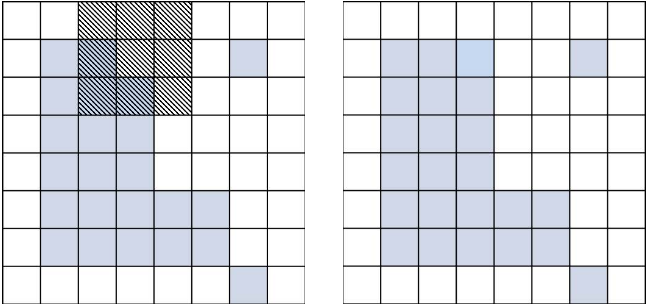
a. eroded image?



b. eroded image?

Dilation

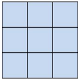
Dilation if one pixel under the mask belongs to the object (blue), then the centered pixel becomes object



- Dilation of the object!
- Duality: erosion is dilation of the background

Dilation

Quiz 17 Which (a or b) is the dilated image? The mask is



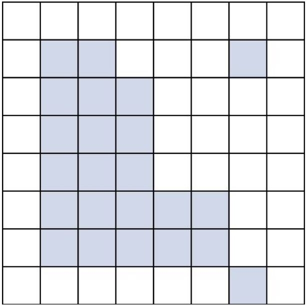
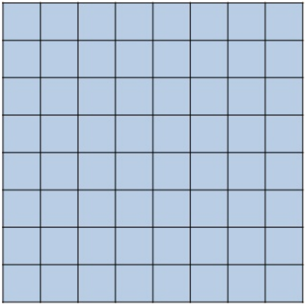
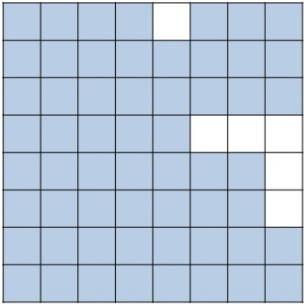


image 8x8

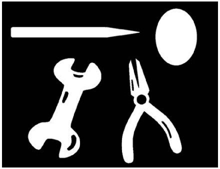


a. dilated image?

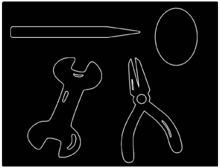


b. dilated image?

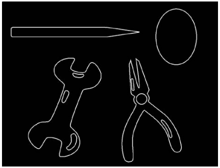
Application: Morphological edge detectors



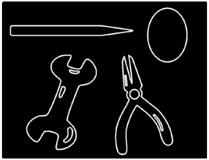
f[x,y]
object



dilate(f)-f
a.



f-erode(f)
b.



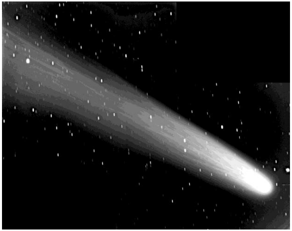
dilate(f)-erode(f)
c.

$\boxed{\textbf{Quiz 18,19, 20}}$

Morphological operators

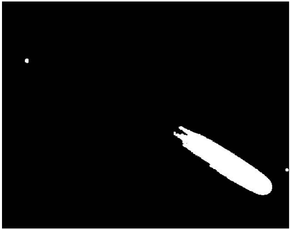
- **Erosion** if one pixel under the mask belongs to the background, then the centered pixel becomes background
  i.e. *erosion of the object*
- **Dilation** if one pixel under the mask belongs to the object, then the centered pixel becomes object
  i.e. *dilation of the object*
- **Opening** erosion and then dilation
  GOAL: remove noise (small elements)
- **Closing** dilation and then erosion
  GOAL: fill in holes in the object or make connex 2 objects that were separated due to noise

Application: Denoising



f[x,y]

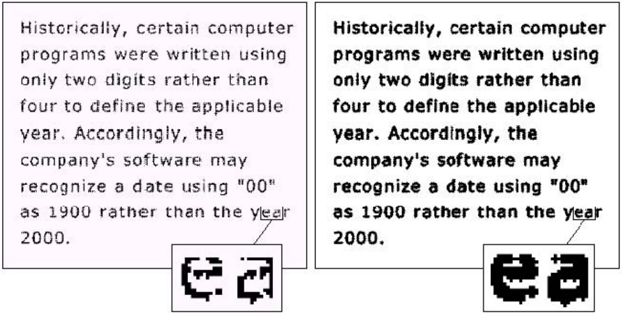
binarized



opening

Source: Gonzalez and Woods. Digital Image Processing.

Application: Recognition with closing



0	1	0
1	1	1
0	1	0

a c
b

FIGURE 9.5
(a) Sample text of poor resolution with broken characters (magnified view).
(b) Structuring element.
(c) Dilation of (a) by (b). Broken segments were joined.

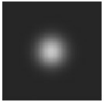
Source: Gonzalez and Woods. Digital Image Processing.

Filtering

- Filtering is a local to point transformation: the new pixel value depends on the current but also neighboring pixels
- For each pixel, we apply a mask.
- If the filter is linear, then, for each applied mask, the centered pixel becomes a linear (more precisely convex) combination of the pixel values in the mask.

input image

$*$

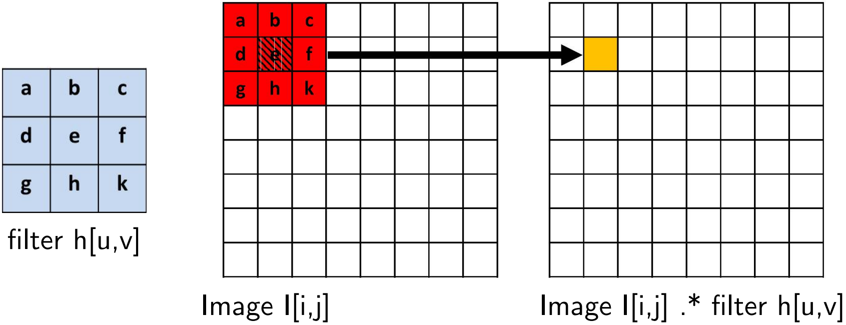


filter

$=$

output image

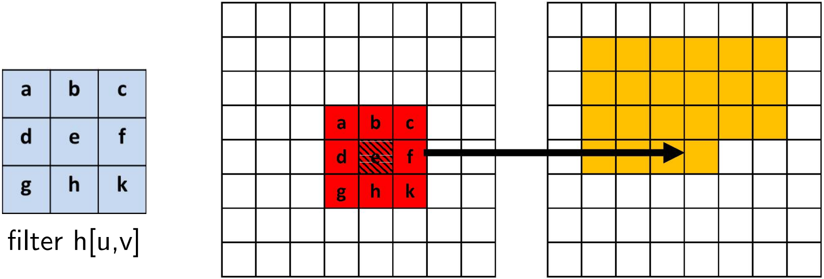
Image filtering: the steps



J is the output of filtering of I by h:

$$
\begin{aligned}
J[2,2] = \quad & a \times I[1,1] + b \times I[1,2] + c \times I[1,3] \\
+ \quad & d \times I[2,1] + e \times I[2,2] + f \times I[2,3] \\
+ \quad & g \times I[3,1] + h \times I[3,2] + k \times I[3,3]
\end{aligned}
$$

Image filtering: the steps



filter h[u,v]

Image I[i,j]

Image I[i,j] .* filter h[u,v]

$$
\begin{aligned}
J[i,j] = \quad & a \times I[i-1,j-1] && + b \times I[i-1,j] && + c \times I[i-1,j+1] \\
+ \quad & d \times I[i,j-1] && + e \times I[i,j] && + f \times I[i,j+1] \\
+ \quad & g \times I[i+1,j-1] && + h \times I[i+1,j] && + k \times I[i+1,j+1]
\end{aligned}
$$

Filtering: the equations

- A compact way to write the equations of 2D linear filtering is: 2D discrete convolution.
- Reminder: 1D discrete convolution

$$g[x] = f[x] * h[x] = \sum_k h[x-k]\ f[k] = \sum_k h[k]\ f[x-k]$$

  - f: input
  - g: output
  - h: filter, convolution kernel
- a linear filter: is the convolution of f by the filter h i.e. it is the linear combination of the neighboring pixel values

Image filtering: the equations

$$J[x,y] = I[x,y] * h[x,y] = \sum_{u,v} h[x-u, y-v]\ I[u,v]$$

$$= \sum_{u,v \in \{-1,0,1\}^2} h[u,v]\ I[x-u, y-v]$$

Example: Gaussian filter

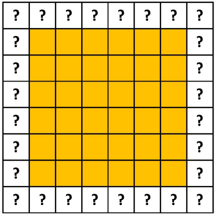
I[x,y]

J[x,y]

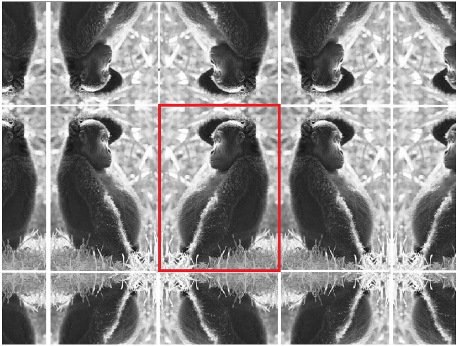
Image filtering: border effect and normalization

Border effect: solutions:

- Set the borders of J to 0
- Convolution of part of the mask, where I exists



- mirror of the image $I[i, j]$



Normalization:
to visualize the filtered image, one needs to **normalize** the output by dividing by the sum of the filter coefficients.

Filtering what? some frequency components in the image

- What is a frequency in an image?

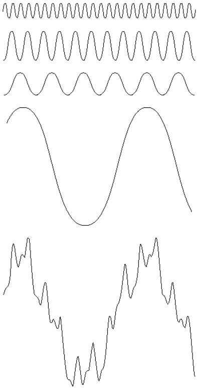
To answer the question:

- What is a frequency in a signal?

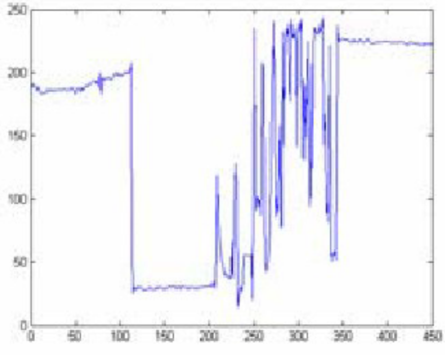
high frequencies

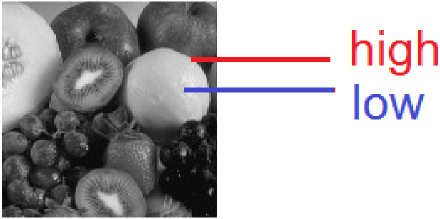
low frequency

sum of all 4 frequencies



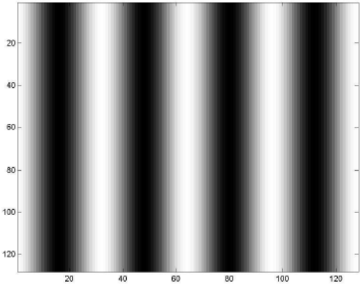
What is a frequency... in the image

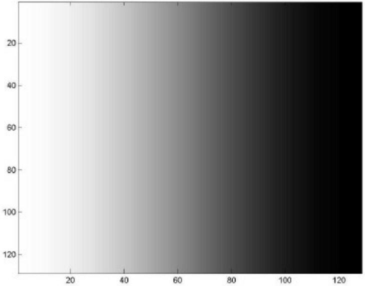




- low frequency: smooth areas
- high frequency: abrupt changes in the luminance

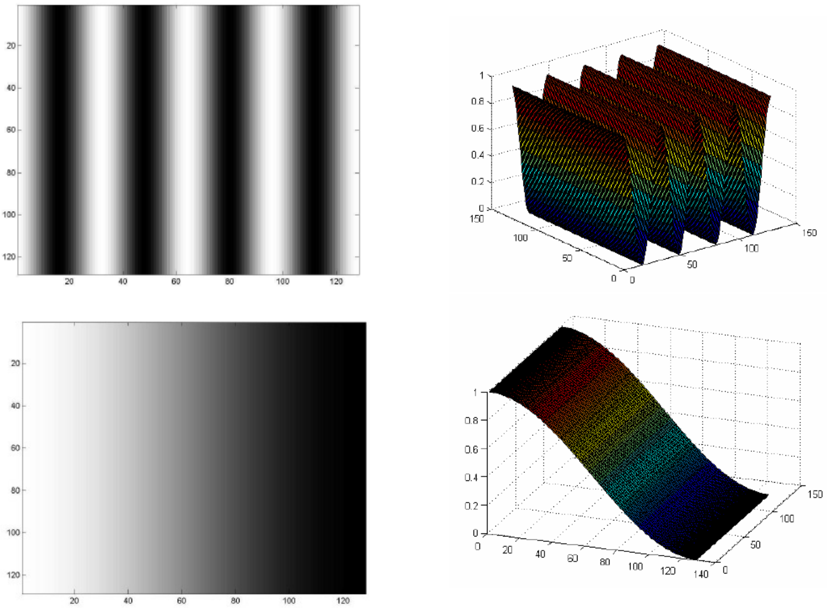
What is a frequency... in the image





low/ high frequency?

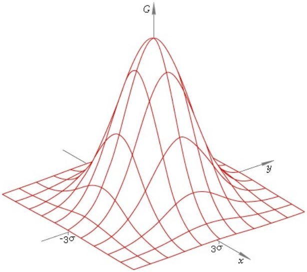
What is a frequency... in the image



LOW pass filtering

- Mean filter
- Gaussian

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9



$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$\frac{1}{273}$

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

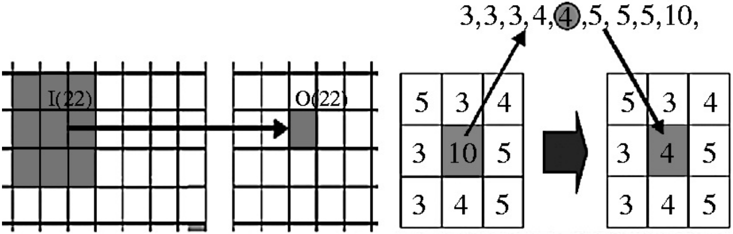
discrete $G(x,y) \quad \sigma = 1$

Discretized approximation to $G(x,y)$:

- by averaging over a pixel area (0.001 precision)
- fspecial in Matlab uses the value of the Gaussian at the centre of a pixel in the mask.

LOW pass filtering

- Median



Quiz 21 to 24

LOW pass filtering: results

original image

5 x 5 mean filter

5 x 5 Gaussian filter

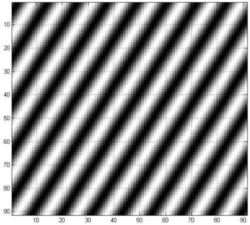
5 x 5 median filter

LOW pass filtering: results

Quiz 25, 26

25 Is it true?
LP filtering keeps low frequencies of an image.

26 Is it true?
image a (below) contains higher frequencies than image b.



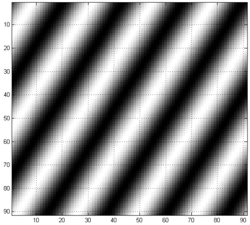
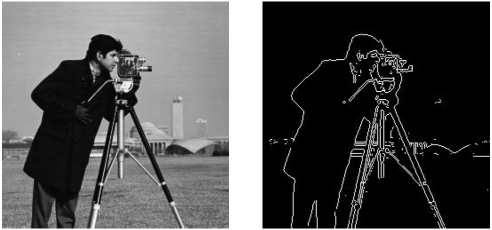


Image filtering: type of filters

- LOW pass filter:
  eliminate high frequencies.
  Application: smoothing, blurring, erasing noise...



- HIGH pass filter:
  eliminate low frequencies.
  Application:
  edge detection, enhance details...

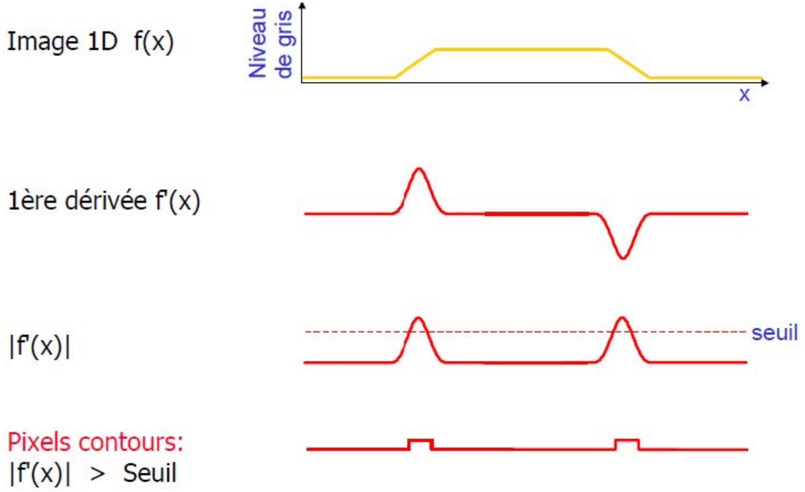
HIGH pass filtering

Properties:

- Edge (contour): corresponds to abrupt changes in the lumimance i.e. high frequencies
- i.e. HP filters exhibit regions with high gradient of the gray level values

$\boxed{\textbf{Quiz 27}}$

Edge detection in 1D



- Edges corresponds to the extrema of the norm of the 1st derivative or equivalently to the zeros of the 2nd derivative.

HIGH pass filtering

2D derivation operators: gradient, laplacian...

- 1st derivative: Gradient

$$\begin{aligned} \nabla f|_{x_0,y_0} &= \frac{\partial f}{\partial x}(x_0, y_0)\mathbf{e}_x + \frac{\partial f}{\partial y}(x_0, y_0)\mathbf{e}_y, \\ \text{norm } |\nabla f|^2 &= \left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2 \end{aligned}$$

- 2nd derivative: Laplacian

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

- edges corresponds to the extrema of the norm of the gradient or equivalently to the zeros of the Laplacian.

Gradient computation

Discretization of the Gradient **Quiz 28,29**

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

Let us assume that $\Delta x = 1$

- vertical axis direction,(detection of horizontal edges) , then $\Delta I = I[1+i,j] - I[i,j]$

$$\text{filter: } \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \text{ or } \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

- horizontal axis direction,(detection of vertical edges) , then $\Delta I = I[i,j+1] - I[i,j]$

$$\text{filter: } (-1,0,1) \text{ or } (-1,1)$$

HIGH pass filtering (gradient): results

original image

Gradient vertical

Gradient horizontal

Gradient norm

Quiz 30

Other HP filters

Filtre de Prewitt : Moyenneur + Dérivée

$$\begin{pmatrix}-1 & 0 & 1\\-1 & 0 & 1\\-1 & 0 & 1\end{pmatrix} = \begin{pmatrix}1\\1\\1\end{pmatrix} * \begin{pmatrix}-1 & 0 & 1\end{pmatrix} \qquad \begin{pmatrix}-1 & -1 & -1\\0 & 0 & 0\\1 & 1 & 1\end{pmatrix} = \begin{pmatrix}-1\\0\\1\end{pmatrix} * \begin{pmatrix}1 & 1 & 1\end{pmatrix}$$

Filtre de Sobel : Gaussienne + Dérivée

$$\begin{pmatrix}-1 & 0 & 1\\-2 & 0 & 2\\-1 & 0 & 1\end{pmatrix} = \begin{pmatrix}1\\2\\1\end{pmatrix} * \begin{pmatrix}-1 & 0 & 1\end{pmatrix} \qquad \begin{pmatrix}-1 & -2 & -1\\0 & 0 & 0\\1 & 2 & 1\end{pmatrix} = \begin{pmatrix}-1\\0\\1\end{pmatrix} * \begin{pmatrix}1 & 2 & 1\end{pmatrix}$$

⇒ Détection des contours moins sensible au bruit

Quiz 31, 32

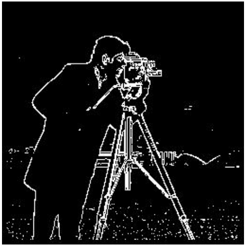
original

Prewitt

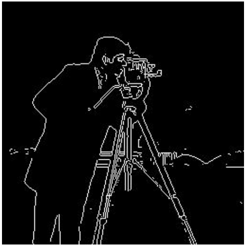
Sobel

HIGH pass filtering: results after thresholding

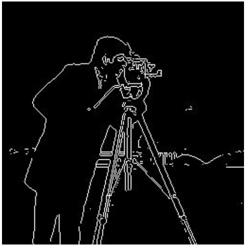
original image



Gradient norm



Prewitt



Sobel

Edge detection based on Laplacian

- Second derivative of a contniuous 2D function f = Laplacian

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

- for a discrete image I, discretized by

$$\nabla^2 I = I[i+1,j] + I[i-1,j] + I[i,j+1] + I[i,j-1] - 4\ I[i,j]$$

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

- the edges correspond to zero crossing of the Laplacian

Result of edge detection

original

Sobel

Laplacian

Downsampling (if time permits)

Aliasing: occurs when downsampling without low pass filtering

Properly sampled image of brick wall.

Spatial aliasing

Downsampling with proper low pass filtering:

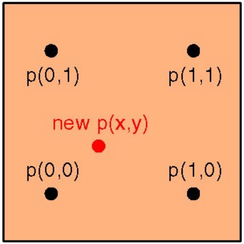
256x256

128x128

64x64

32x32

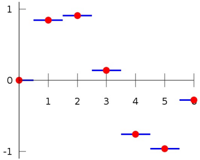
Interpolation/Upsampling



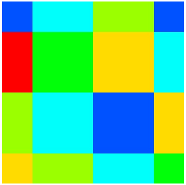
Interpolation/Upsampling

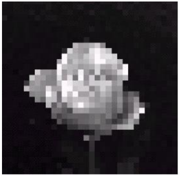
- Nearest Neighbor (NN) method

1D



2D

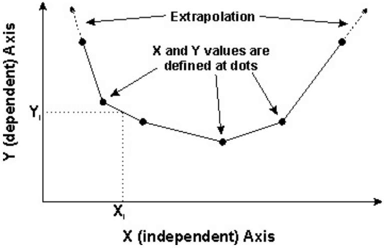




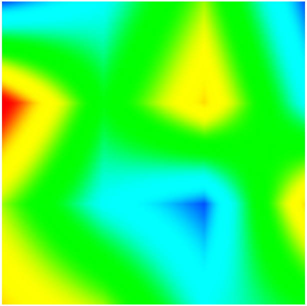
Drawback: blocking effect

Bilinear Interpolation

1D



2D

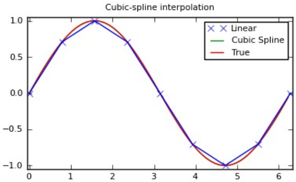


Avoid blocking effect BUT non smooth i.e. non natural

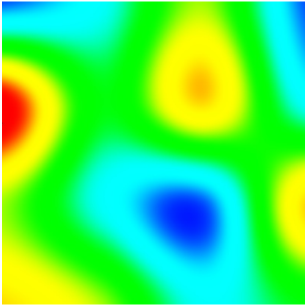
Bicubic Interpolation

- linear combination of the 16 neighboring pixels. The weights depend on the distance between the neighboring pixel and the pixel to be interpolated
- **bicubic interpolation** achieves a good complexity/rendering tradeoff

1D



2D



Interpolation: the results

Plus proche voisin

Bilinéaire

a b c
d e f

FIGURE 2.25 Top row: images zoomed from 128 × 128, 64 × 64, and 32 × 32 pixels to 1024 × 1024 pixels, using nearest neighbor gray-level interpolation. Bottom row: same sequence, but using bilinear interpolation.

Interpolation: the results

Plus proche voisin

Bilinéaire (4 voisins)

Interpolation: the results

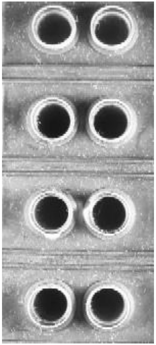
Bilinéaire (4 voisins)

Bicubique (16 voisins)

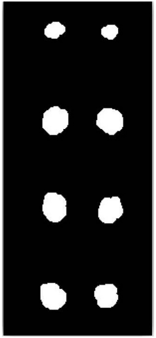
Image segmentation is...

- the process of partitioning a digital image into multiple segments,
- the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics.

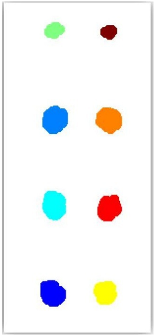
Example: counting holes.



original



detect holes

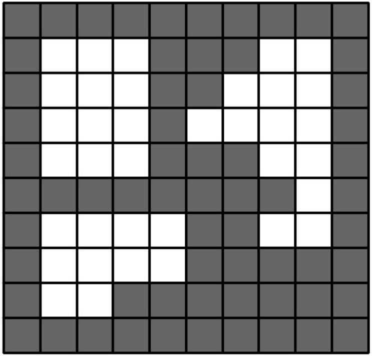


label

Label connected components in 2-D binary image

← *Fond*
← *Objets segmentés*

- Nous allons effectuer un parcours de l'image pour affecter un numéro unique (étiquette) pour chaque région
- Tous les pixels d'une même région doivent avoir le même numéro (étiquette)



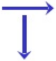
Source: A. Boucher IFI

first iteration

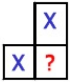
Premier parcours de l'image

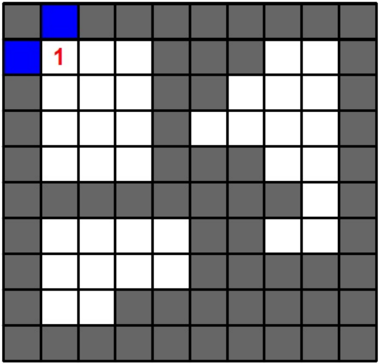
- Pour chaque pixel d'une région, on lui affecte
  - soit la plus petite étiquette parmi ses voisins haut et gauche
  - soit une nouvelle étiquette.

Parcours



Voisinage



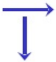


first iteration

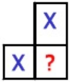
Premier parcours de l'image

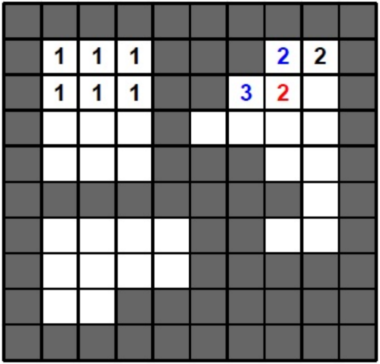
- Pour chaque pixel d'une région, on lui affecte
  - soit la plus petite étiquette parmi ses voisins haut et gauche
  - soit une nouvelle étiquette.

Parcours



Voisinage



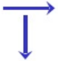


first iteration

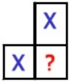
Premier parcours de l'image

- Pour chaque pixel d'une région, on lui affecte
  - soit la plus petite étiquette parmi ses voisins haut et gauche
  - soit une nouvelle étiquette.

Parcours



Voisinage



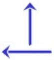
	1	1	1				2	2	
	1	1	1			3	2	2	
	1	1	1		4	3	2	2	
	1	1	1				2	2	
								2	
	5	5	5	5			6	2	
	5	5	5	5					
	5	5							

SECOND iteration

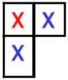
Deuxième parcours de l'image

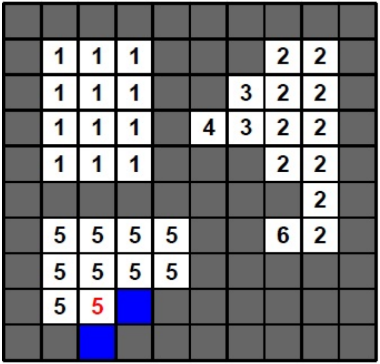
- Pour chaque pixel d'une région, on lui affecte
  - la plus petite étiquette parmi la sienne et celles ses voisins bas et droite

Parcours



Voisinage



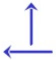


SECOND iteration

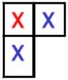
Deuxième parcours de l'image

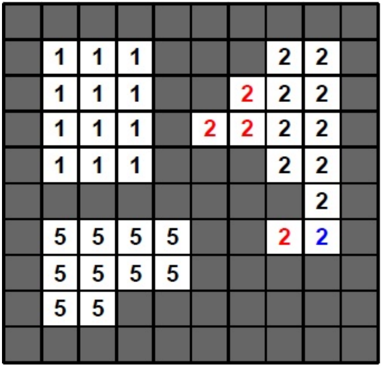
- Pour chaque pixel d'une région, on lui affecte
  - la plus petite étiquette parmi la sienne et celles ses voisins bas et droite

Parcours



Voisinage





Label connected components in 2-D binary image

- in this example, 2 iterations were sufficient to perform segmentation,
- in general, more than 2 iterations are needed,
- the iterations are performed until there is no more change in the labels: so indeed, we should have perform a last iteration to be sure that the labels would no change.

Quiz 33 to 35

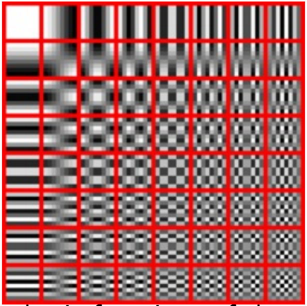
Part 5 - Global to point transformation

When a pixel of the new (transformed) image depends on a the whole input image

DCT:Discrete cosine transform

Change of basis:
Decomposition of the input patch x_{n_1,n_2}
into the coefficients X_{k_1,k_2} in the new basis

$$X_{k_1,k_2} = \sum_{n_1=0}^{N_1-1} \left(\sum_{n_2=0}^{N_2-1} x_{n_1,n_2} \cos\left[\frac{\pi}{N_2}\left(n_2+\frac{1}{2}\right)k_2\right]\right) \cos\left[\frac{\pi}{N_1}\left(n_1+\frac{1}{2}\right)k_1\right]$$



basis functions of the 2D-DCT for 8×8 patches

Used in jpeg image compression, and mpeg-2, mpeg-4, hevc video compression

Quiz 36 to 39

Part 6 - Quality measure

How to assess that an image transformation works correctly?

An **objective** quality measure

Peak signal-to-noise ratio (PSNR):

- measures the distortion between two images I and K
- defined via the mean squared error (MSE):

$$MSE = \frac{1}{m\,n} \sum_{i=1}^{m} \sum_{j=1}^{n} (I[i,j] - K[i,j])^2$$

and

$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX_I^2}{MSE} \right)$$

For instance, $MAX = 255$.

- if I = true image and K the processed image, PSNR is a quantitative measure of the quality of the image
- The higher the better...

Part 7 - Machine learning

How to learn automatically how humans understand/do things?

Artificial intelligence (AI)

AI applied to images: computer vision

- **System of experts**: emulates the decision-making ability of a human expert. (Wikipedia)

  Expert: specifies all steps he took to make the decision
  Machine: is programmed such as to reproduce the steps.

- **Machine learning** gives "computers the ability to learn without being explicitly programmed." [A. Samuel 59] (Wikipedia)

  Expert: gives the decision for a set of training examples
  Machine: is programmed such as to mimic the decision.

Big interest into machine learning due to deep learning!!

An example: digit recognition solved with machine learning

Algorithm: nearest neighbor search.

Source: V. Athitsos. Nearest Neighbor Retrieval and Classification.

Pro's and con's of machine learning

- + efficient... – if many many (training) samples
- – expert needs to give decisions for each training data **(supervised)**
- – there exists **non supervised** (k-means clustering) but less efficient

Today's most efficient machine learning algorithms: deep learning. Classification with Deep Convolutional Neural Networks by A Krizhevsky won ImageNet 2012 challenge with major decrease in classification error.

Since then, tremendous work in the community on deep learning.

First layers in deep learning... look like classical image processing!!!