VIC-Introduction to Visual Computing Course 2 : Low Level Processing - Filtering

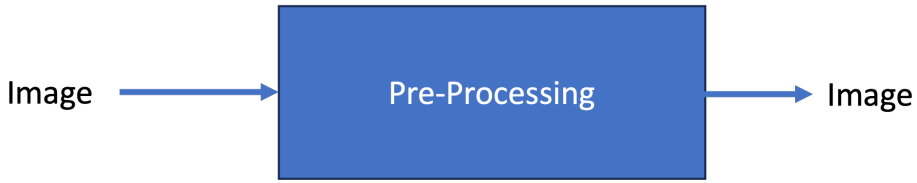
Maria Vakalopoulou and Céline Hudelot

13 janvier 2025

Outline

Image Preprocessing

Preprocessing : encompasses all processes aimed at improving the characteristics of an image.



The goal of image enhancement is to make images more suitable for human or machine interpretation.

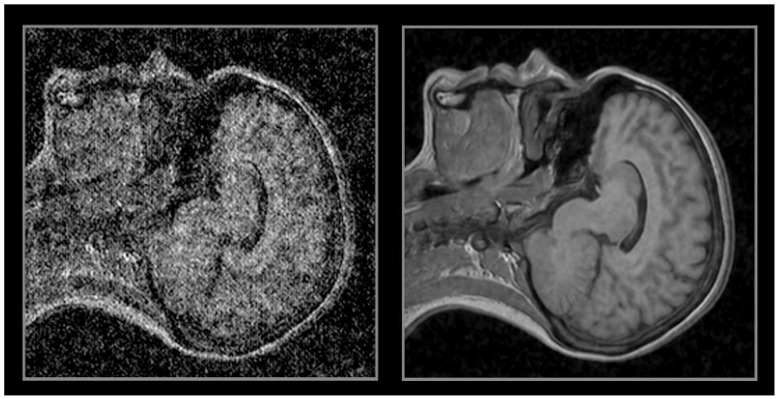
Image Preprocessing

- Local Smoothing : removes noise or small variations present in an image. The intensity of a pixel is transformed based on the intensities in a **small neighborhood of the pixel** (also referred to as local enhancement).
- Image Enhancement : modifies the visual characteristics of the image (contrast, etc.) to facilitate its interpretation by the human eye (often pointwise enhancement).
- Image Restoration : removes degradations suffered by an image using prior knowledge about these degradations.

We can work in the spatial domain or in the frequency domain (Fourier transform).

Image Preprocessing

Examples of preprocessing



débruitage

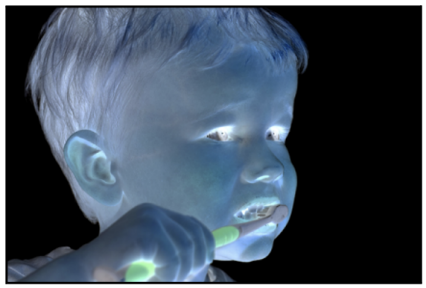
inpainting

Image super resolution

Image pre-processing

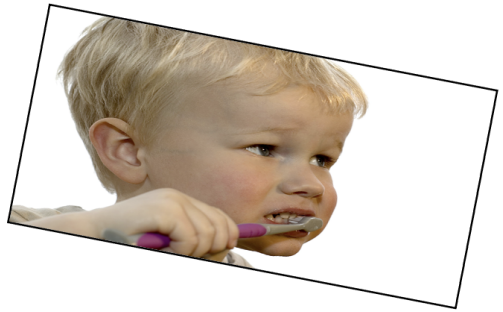
What types of image transformations can we do?

Filtering



changes pixel *values*

Warping



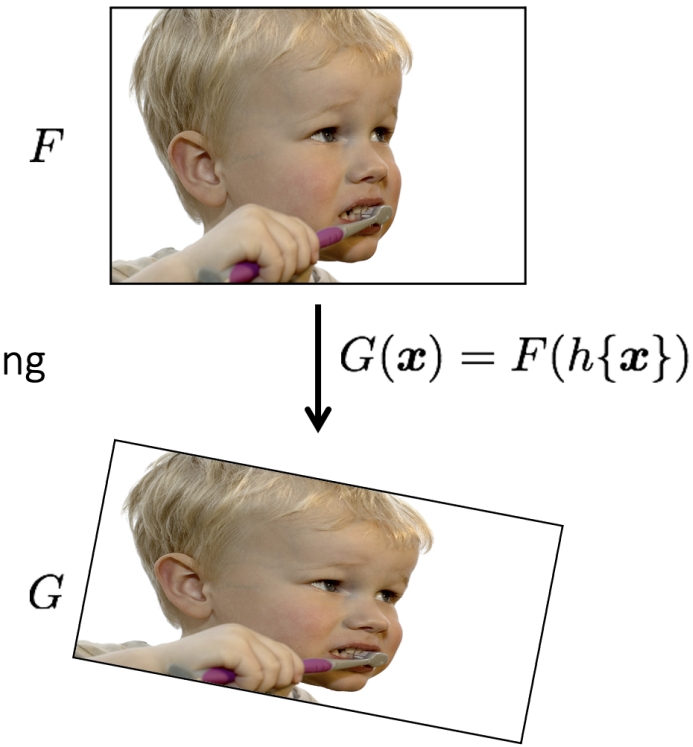
changes pixel *locations*

slide credit : `http://www.cs.cmu.edu/~16385/`

Image pre-processing

What types of image transformations can we do?

changes *range* of image function



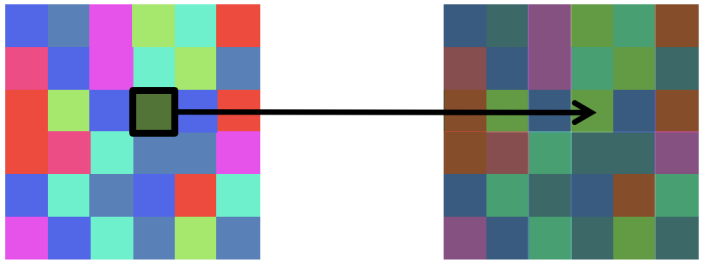
changes *domain* of image function

slide credit : `http://www.cs.cmu.edu/~16385/`

Image pre-processing

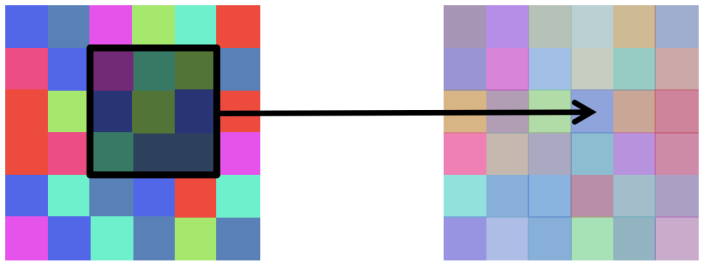
What types of image filtering can we do?

Point Operation



point processing

Neighborhood Operation

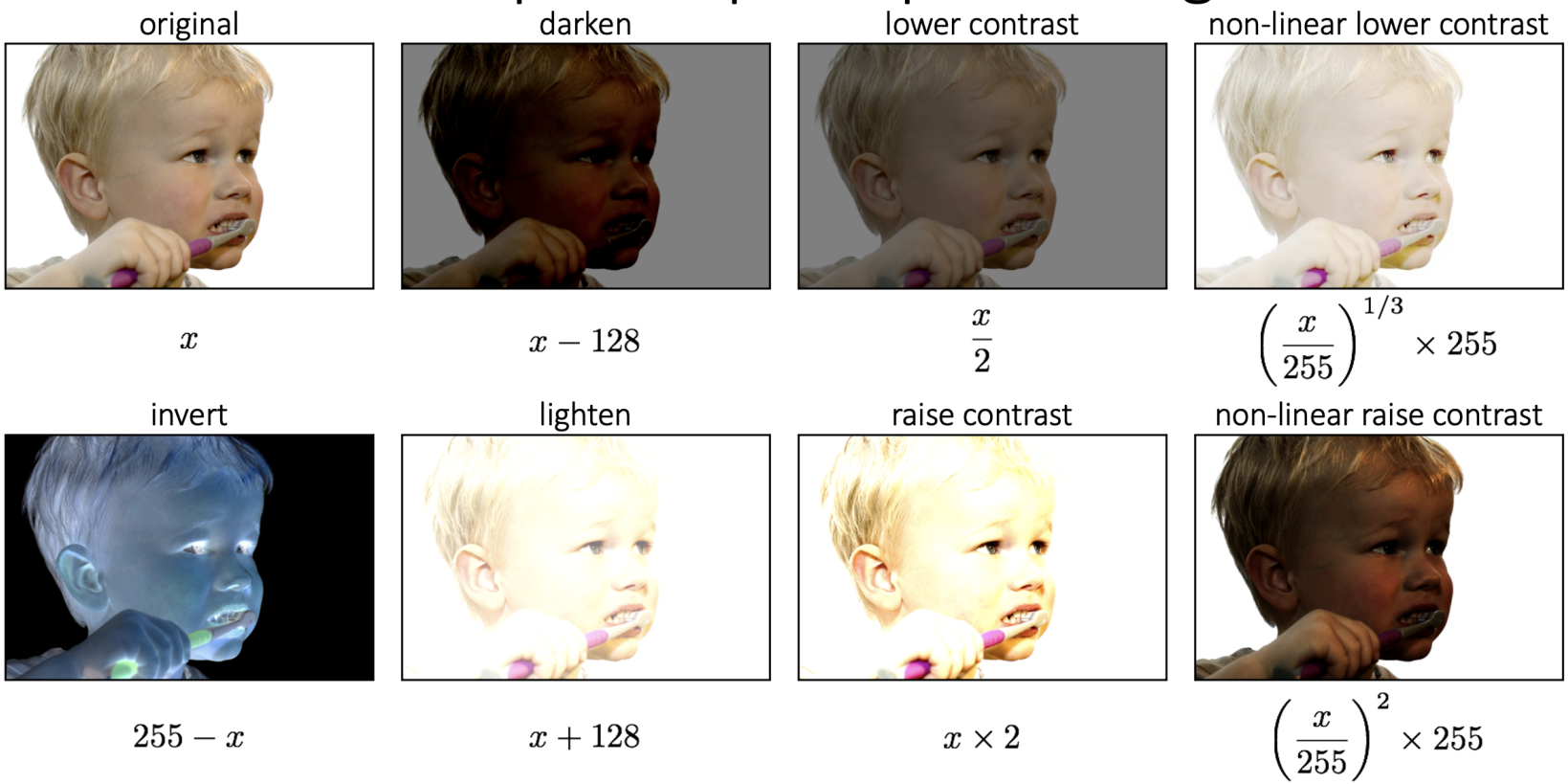


"filtering"

slide credit : `http://www.cs.cmu.edu/~16385/`

Image pre-processing

Examples of point processing



slide credit : `http://www.cs.cmu.edu/~16385/`

Outline

Image Histogram

- Distribution of grayscale levels (or colors) in an image.
- $H(k)$ = number of pixels with the value k in the image.

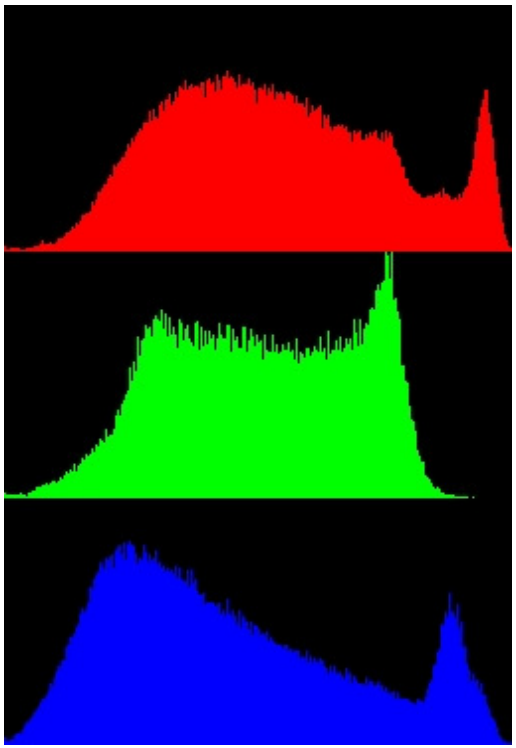


Image Histogram

Normalized Histogram

- Proportion of pixels as a function of grayscale level (probability density).
- H_n gives the probability (in terms of occurrence frequency) that a pixel has the grayscale level k.
- $x \rightarrow H_n(x) = \frac{H(x)}{\text{number of pixels}}$

Image Histogram

Cumulative Histogram

- Cumulative histogram :

$$H_c(k) = \sum_{i \leq k} H(i)$$

- Normalized cumulative histogram :

$$H_c(k) = \sum_{i \leq k} H_n(i)$$

- $H_c(k)$ represents the probability of having a grayscale level less than or equal to k.

Image Histogram

Dynamic range of an image $[val_{min}, val_{max}]$:

- val_{min} : minimum grayscale level in the image.
- val_{max} : maximum grayscale level in the image.

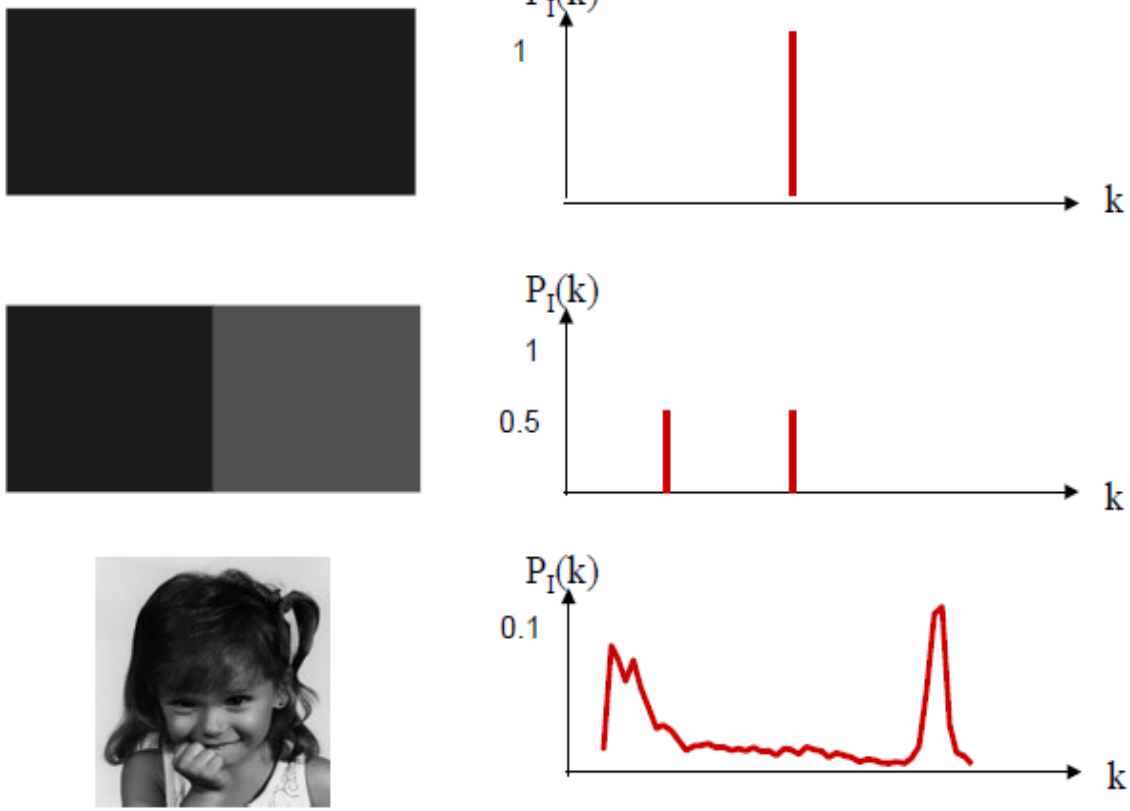


Image Luminance

Definition

- Average intensity of all pixels in the image.
- To increase luminance, simply shift the histogram.

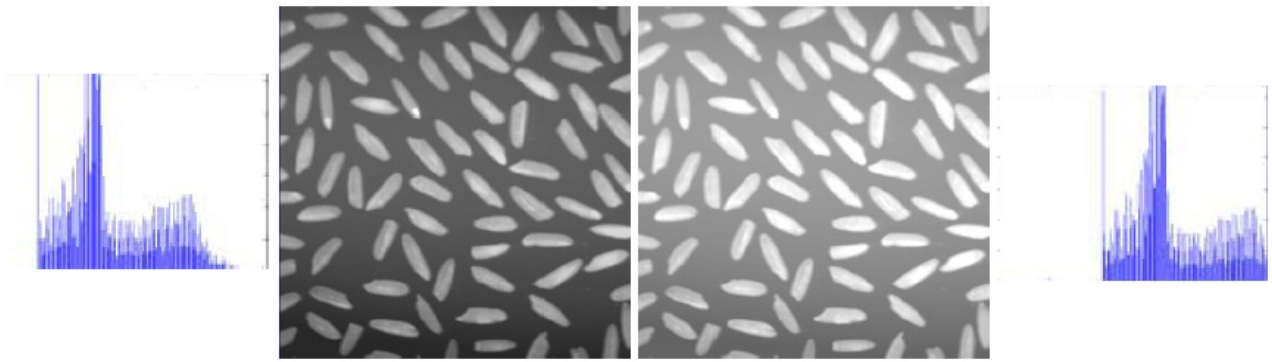


Image Contrast

Two Definitions

- Standard deviation of grayscale level variations.

$$C = \sqrt{\frac{1}{M \times N} \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} (f(x,y) - \text{Mean})^2}$$

- Variation between minimum and maximum grayscale levels.

$$C = \frac{\max[f(x,y)] - \min[f(x,y)]}{\max[f(x,y)] + \min[f(x,y)]}$$

Image Contrast

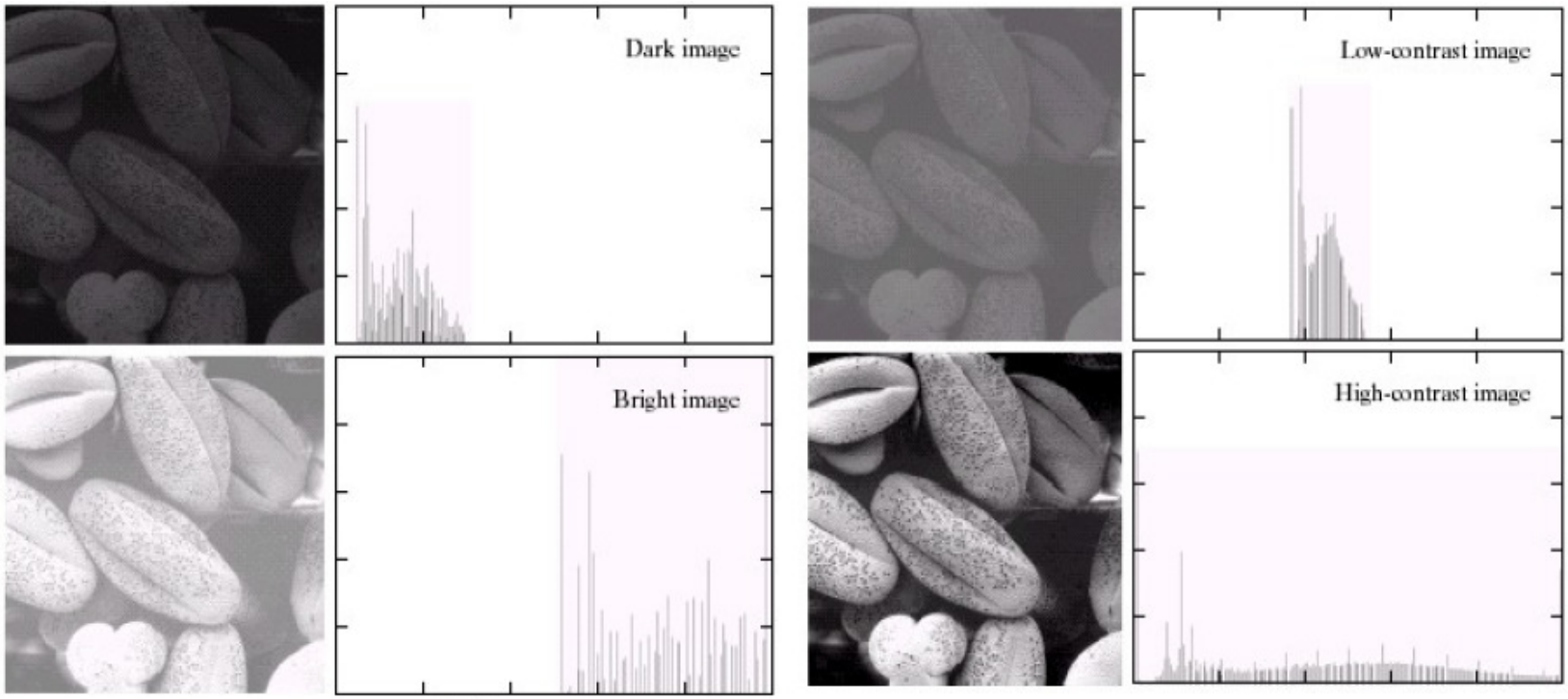


FIGURE 3.15 Four basic image types: dark, light, low contrast, high contrast, and their corresponding histograms. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)

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

Contrast Enhancement

- Linear transformation
- Linear transformation with saturation
- Piecewise linear transformation
- Nonlinear transformation
- Histogram equalization

Dynamic Range Expansion

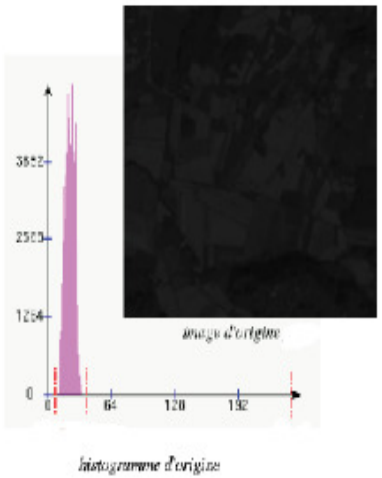
Principle

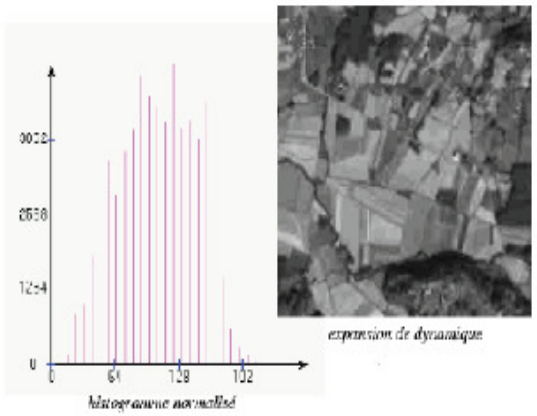
Transformation of grayscale levels so that the image uses the full dynamic range (between 0 and 255).

- $I_{Exp}(i,j) = (I(i,j) - \min)(\frac{255}{\max - \min})$

Or more generally, expansion to the interval $[0, L-1]$:

$$I_{Exp}(i,j) = (I(i,j) - \min)(\frac{L-1}{\max - \min})$$

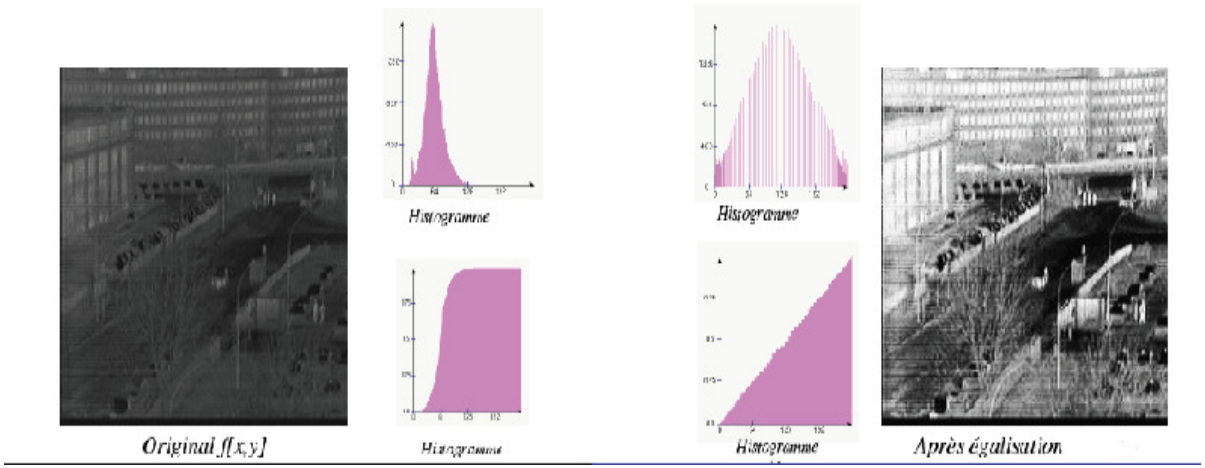




Equalization

Principle

- Grayscale transformation aimed at balancing the pixel distribution within the dynamic range as evenly as possible.
- The cumulative histogram $h_c(x) = \sum_{z \leq x} h_n(z)$ should be as linear as possible.
- $I_{Ega}(i,j) = \text{Int}(255 \times h_c(I(i,j)))$, where Int is the function that rounds to the nearest integer.

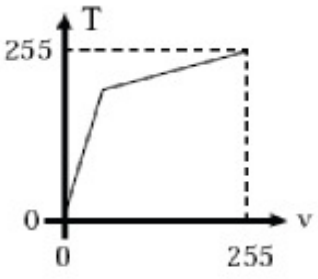


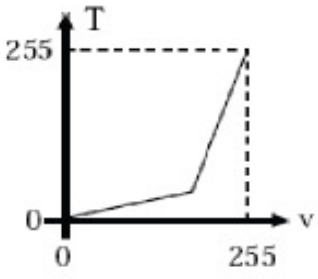
Histogram Transformations

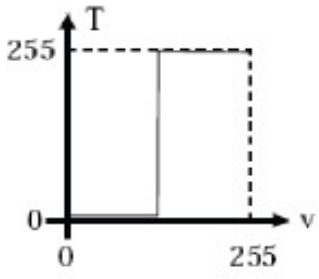
Algorithmic Principle

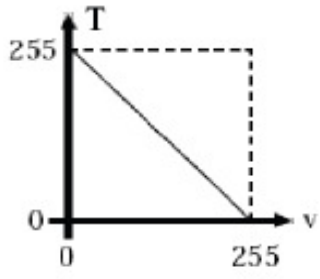
- Traverse the pixels of the image.
- For each pixel (i, j) :
  - Read the value x.
  - Replace x with $T(x)$.
- The function $T(.)$ is represented by its plot. The choice of $T(.)$ allows modifying the characteristics of an image.

Histogram Transformations



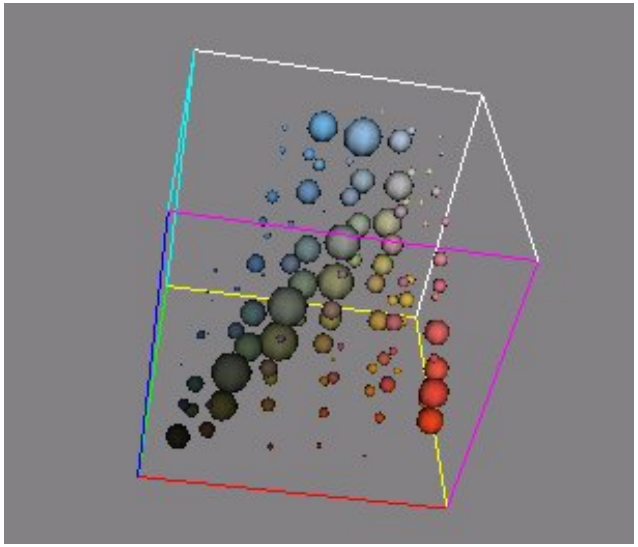


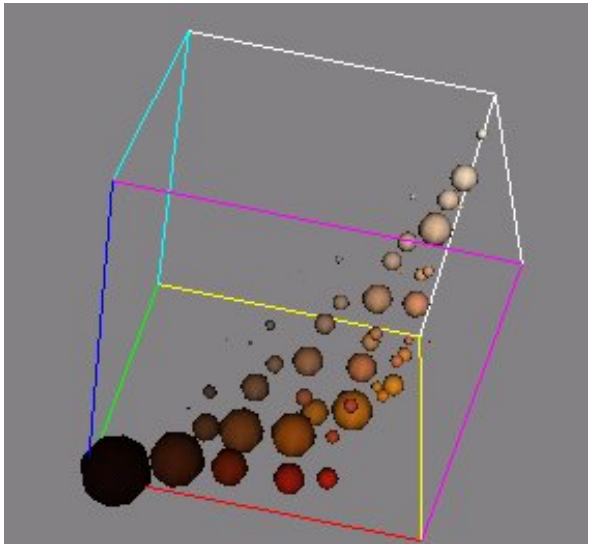




Color Histogram

Complete 3D color histogram





Credit : http ://ij-plugins.sourceforge.net/ij-vtk/color-space/

Point-Based Approach

- These approaches are generally **point-based**, i.e., the value of each output pixel depends only on the value of the corresponding input pixel (and potentially, some additional global information).
- Approaches aimed primarily at improving the visual quality of images.
- Important applications in enhancing modern visual recognition techniques (data augmentation, self-supervised learning, etc.).
- Refer to Section 3.1 of Richard Szeliski's book.

Recent Applications

Image augmentation
source : https://github.com/aleju/imgaug

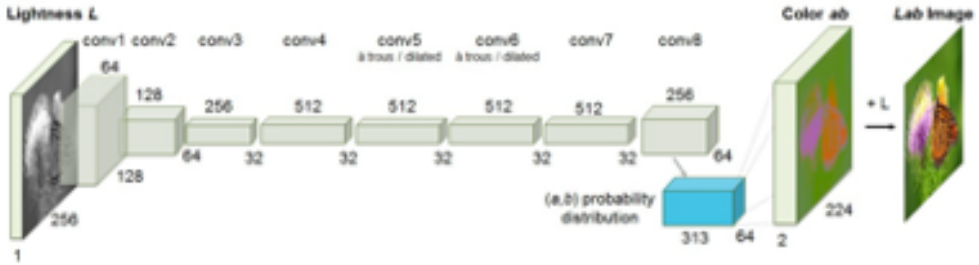


Fig. 2. Our network architecture. Each `conv` layer refers to a block of 2 or 3 repeated `conv` and `ReLU` layers, followed by a `BatchNorm` [30] layer. The net has no `pool` layers. All changes in resolution are achieved through spatial downsampling or upsampling between `conv` blocks.

Colorful image colorization
Self-supervised learning with colorization
source : https://richzhang.github.io/colorization/

Outline

Filtering

Filtering

An operation that generates a new image whose pixels are a combination of the values of the pixels from an original image.

Objectives

- Extract useful information from the image :
  - Features : edges, corners, etc.
- Modify or enhance specific properties of the image :
  - Denoising, super-resolution, inpainting, etc.

Filtering

- Similar to filtering in 1D signals : remove noise or select specific frequencies.
- In images, we often refer to spatial frequency :
  - High spatial frequency : details that repeat frequently over a small number of pixels.
  - Low spatial frequency : variations that repeat less frequently.

Filtering

Filter Categories

- High-pass filters : eliminate low frequencies.
- Low-pass filters : eliminate high frequencies.
- Band-pass filters : allow only a specific range of frequencies.

Image Smoothing

- Low-pass filters.
- Objective : Noise reduction and detail attenuation (low frequencies).

Image Derivation

- High-pass filters.
- Objective : Extract edges and interest points by emphasizing details and edges (high frequencies).

Filtering

General Properties

- Signal representation domain :
  - Spatial
  - Frequency
- Linear vs. Non-linear filters.
- Isotropic vs. Anisotropic filters.

Systems and Filters

Image as a Function

Some definitions

- **Signals** : measurement of signal physical quantity (light, sound, ...) a a fonction of another independent quantity (time, space, wavelength...).
  - Mainly **continous** in nature.
  - Transformed into a finite sequence of numbers, called a **discrete signal** through a process of **sampling**.

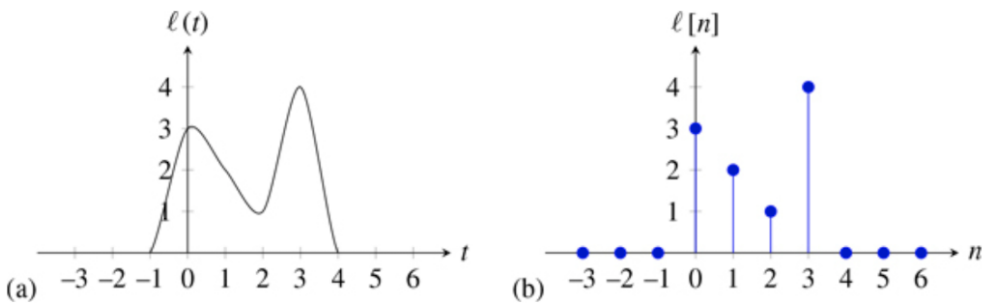


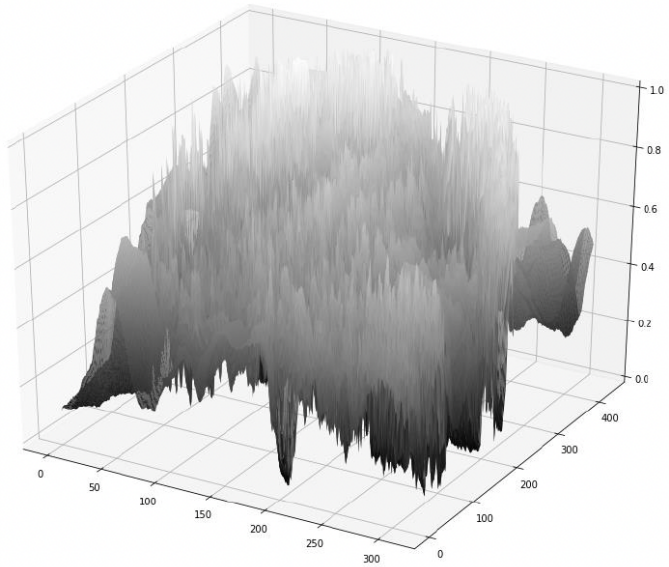
Figure 15.1: (a) A continuous signal, and (b) a discrete signal obtained by sampling the continuous signal at the times $t = n$.

- **Systems** : process/function that transforms a signal into another.

Image as a Function

- An image is a function f from $\mathbb{R}^2$ to $\mathbb{R}^M$.
- $f[x, y]$ gives the **intensity** at position $[x, y]$.
- Defined on a rectangular domain, with finite bounds.

$$f: \underbrace{[a, b] \times [c, d]}_{\text{Domain support}} \rightarrow \underbrace{[0,255]}_{\text{range}}$$



Systems and Filters

$$f[n,m] \longrightarrow \boxed{\text{System } \mathcal{S}} \longrightarrow g[n,m]$$

- An image is considered a function $f : \mathbb{R}^2 \rightarrow \mathbb{R}^M$.
- A system S converts an input function $f[n,m]$ into an output function $g[n,m]$.
- For images, $[n,m]$ represents the **spatial position within the image**.

Systems and Filters

$$f[n,m] \rightarrow \boxed{\text{System } \mathcal{S}} \rightarrow g[n,m]$$

- S is the system (operator), defined as a mapping that transforms the input f into the output g.

$$g = \mathcal{S}[f] \quad ; g[n,m] = \mathcal{S}\{f[n,m]\}$$

$$f[n,m] \overset{\mathcal{S}}{\rightarrow} g[n,m]$$

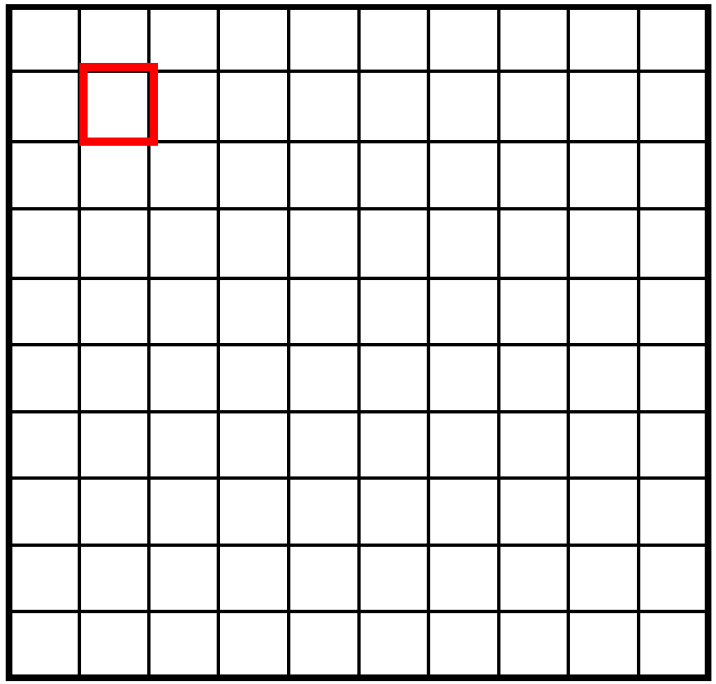
System - Example 1 : Averaging Filter

2D moving average over a 3×3 neighborhood.

$f[n, m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$g[n, m]$



Courtesy of S. Seitz

System - Example 1 : Averaging Filter

$f[n,m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$g[n,m]$

	0	10							

System - Example 1 : Averaging Filter

$f[n,m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$g[n,m]$

	0	10	20						

System - Example 1 : Averaging Filter

$f[n,m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$g[n,m]$

	0	10	20	30					

System - Example 1 : Averaging Filter

$f[n,m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$g[n,m]$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

System - Example 1 : Averaging Filter

Sum over rows

Sum over columns

$$g[n,m] = \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{l=m-1}^{m+1} f[k,l]$$

$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} f[n-k, m-l]$$

Filter "kernel", "mask"

$$\frac{1}{9} \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

h

The applied function appears to be a linear combination (weighted sum) of the pixel and its neighbors :

$$g[i,j] = \sum_{k}^{l} f(i+k, j+l) h(k,l)$$

Here we see the concept of a filter.

System - Example 1 : Averaging Filter

Summary

- This filter transforms each pixel value into the average of its neighbors.
- Produces a smoothing effect.

System - Example 2 : Image Segmentation

- Image segmentation based on a simple threshold:

$$g[n,m] = \begin{cases} 255, & f[n,m] > 100 \\ 0, & \text{otherwise.} \end{cases}$$

Source : CS 131 Stanford

Systems and Filters

Systems can be classified based on their properties :

- **Amplitude Properties**
  - **Additivity** : $\mathcal{S}[f_i[m,n] + f_j[m,n]] = \mathcal{S}[f_i[m,n]] + \mathcal{S}[f_j[m,n]]$
  - **Homogeneity** : $\mathcal{S}[\alpha f_i[m,n]] = \alpha \mathcal{S}[f_i[m,n]]$
  - **Superposition** : $\mathcal{S}[\alpha f_i[m,n] + \beta f_j[m,n]] = \alpha \mathcal{S}[f_i[m,n]] + \beta \mathcal{S}[f_j[m,n]]$
  - **Invertibility** : $\mathcal{S}^{-1}[\mathcal{S}[f[m,n]]] = f[m,n]$
- **Spatial Properties**
  - **Causality** (effects cannot precede the cause) : for $m < m_0$ and $n < n_0$, $f[m,n] = 0 \implies g[m,n] = 0$
  - **Shift Invariance** :

$$f[n - n_0, m - m_0] \xrightarrow{\mathcal{S}} g[n - n_0, m - m_0]$$

Systems and Filters

Shift-Invariant Systems (SI)

If $f[n,m] \xrightarrow{\mathcal{S}} g[n,m]$
then, $\mathcal{S}$ is shift-invariant (SI) if

$$f[n-n_0, m-m_0] \xrightarrow{\mathcal{S}} g[n-n_0, m-m_0]$$

for every input $f[n,m]$ and shifts n_0 and m_0.

To do at home !

Is the averaging filter shift-invariant ?

Linear Systems and Filters

$$f[n,m] \rightarrow \boxed{\text{System } \mathcal{S}} \rightarrow g[n,m]$$

- Linear filtering :
  - Creates a new image where each pixel is a weighted sum of the values of the original pixels.
  - The same set of weights is used at each point.
- Reminder : linear function
  - $f(\alpha x) = \alpha f(x)$
  - $f(x+y) = f(x) + f(y)$

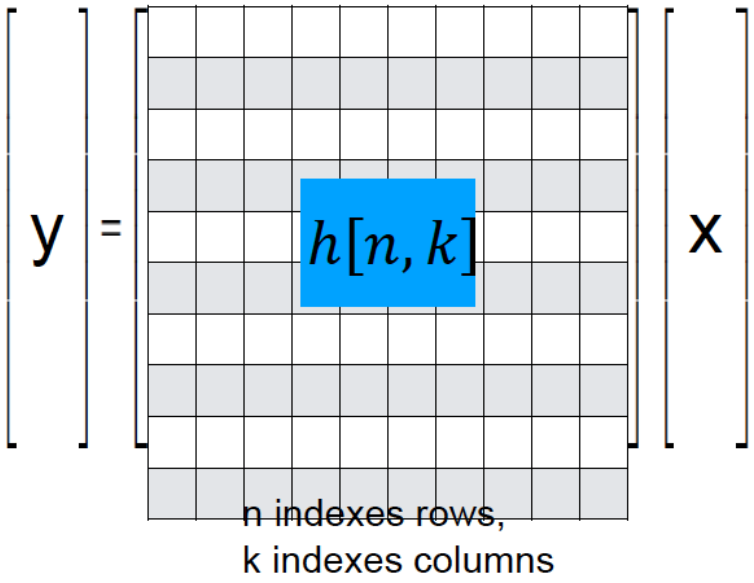
$\mathcal{S}$ is a linear system if and only if

$$\mathcal{S}\{\alpha f_1[n,m] + \beta f_2[n,m]\} = \alpha \mathcal{S}\{f_1[n,m]\} + \beta \mathcal{S}\{f_2[n,m]\}$$

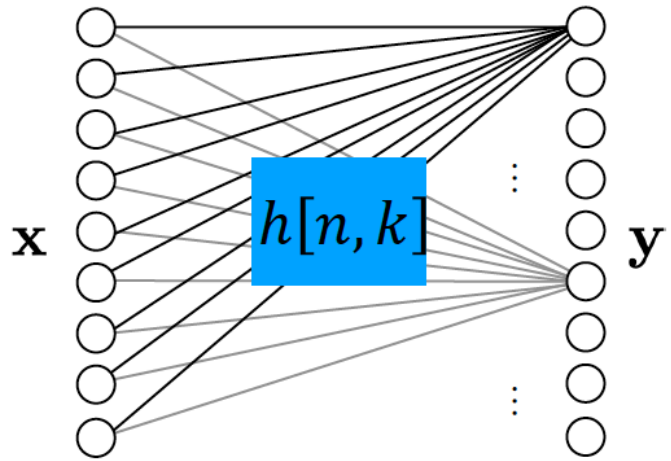
(Superposition property)

Linear Systems and Filters

A linear function f can be written as a matrix multiplication:



It can also be represented as a fully connected linear neural network



$h[n,k]$ Is the strength of the connection between x[k] and y[n]

Shift-Invariant Linear Systems and Filters

LSI Systems (Linear Shift-Invariant)

- Superposition property

$$\mathcal{S}\{\alpha f_1[n,m] + \beta f_2[n,m]\} = \alpha\mathcal{S}\{f_1[n,m]\} + \beta\mathcal{S}\{f_2[n,m]\}$$

- Shift-invariance

$$f[n-n_0, m-m_0] \xrightarrow{\mathcal{S}} g[n-n_0, m-m_0]$$

Shift-Invariant Linear Systems and Filters (LSI)

- How to compute an LSI system ?
- An LSI system is fully specified by its impulse response.

$$\delta_2[n,m] \xrightarrow{S} h[n,m]$$

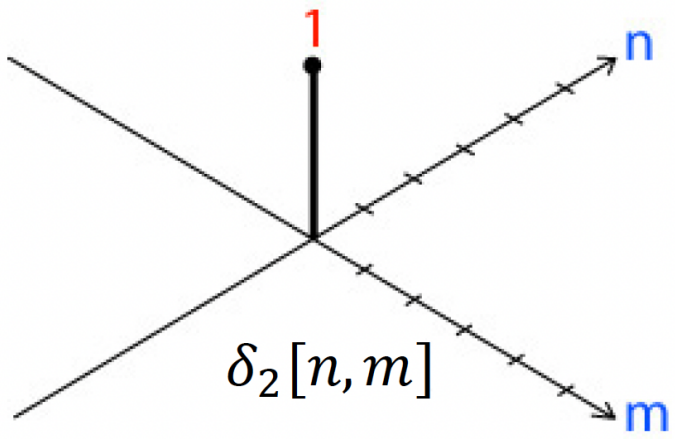
Pass in an impulse function — Record its response

Shift-Invariant Linear Systems and Filters (LSI)

2D impulse function $\delta_2[n,m]$

- 1 at [0,0].
- 0 everywhere else

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



Averaging Filter and Impulse Response

- Averaging filter

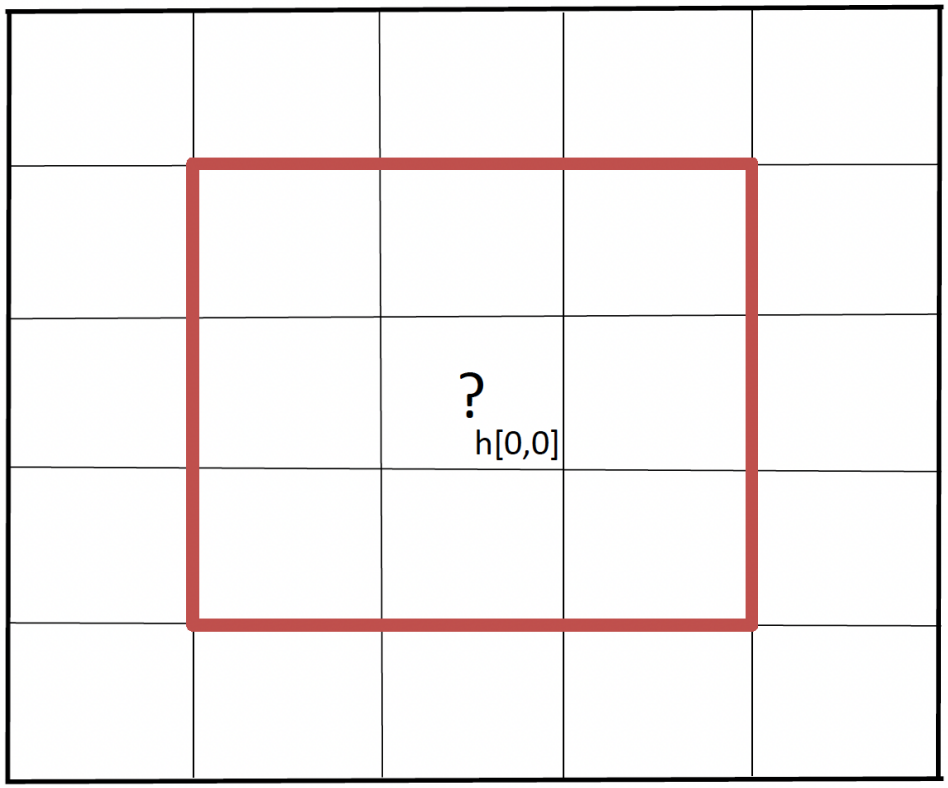
$$f[n,m] \xrightarrow{\mathcal{S}} \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} f[n-k, m-l]$$

- We use it to derive the expression of the impulse response

$$\delta_2[n,m] \xrightarrow{\mathcal{S}} h[n,m]$$

$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \delta_2[n-k, m-l]$$

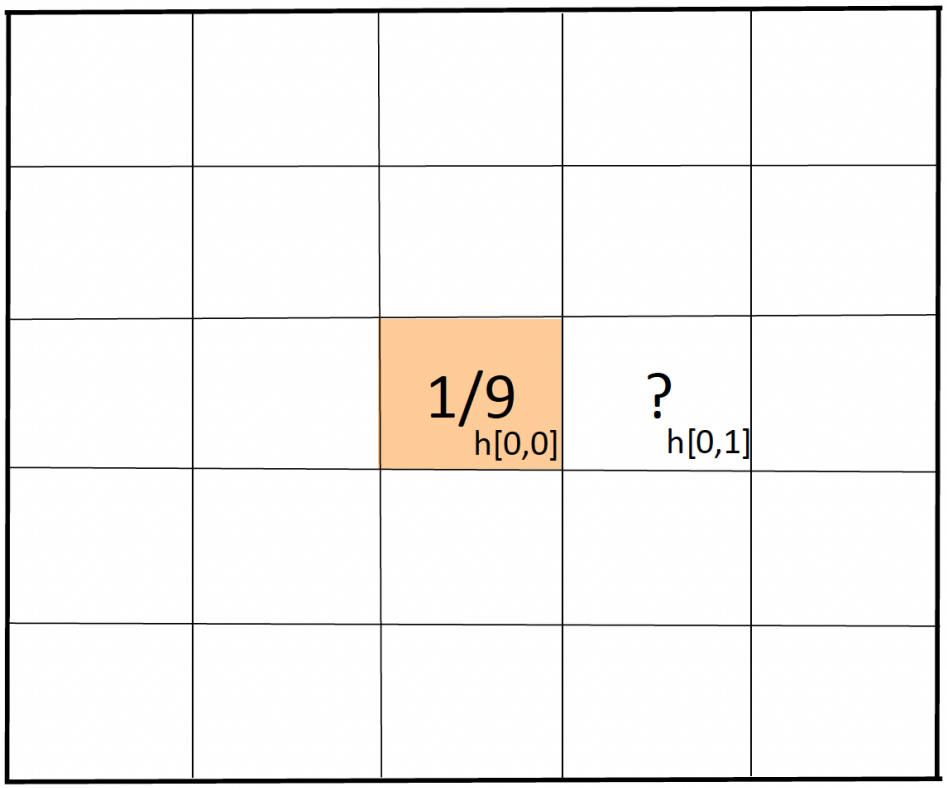
Averaging Filter and Impulse Response



$$\delta_2[n,m] \xrightarrow{S} h[n,m]$$

$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \delta_2[n-k, m-l]$$

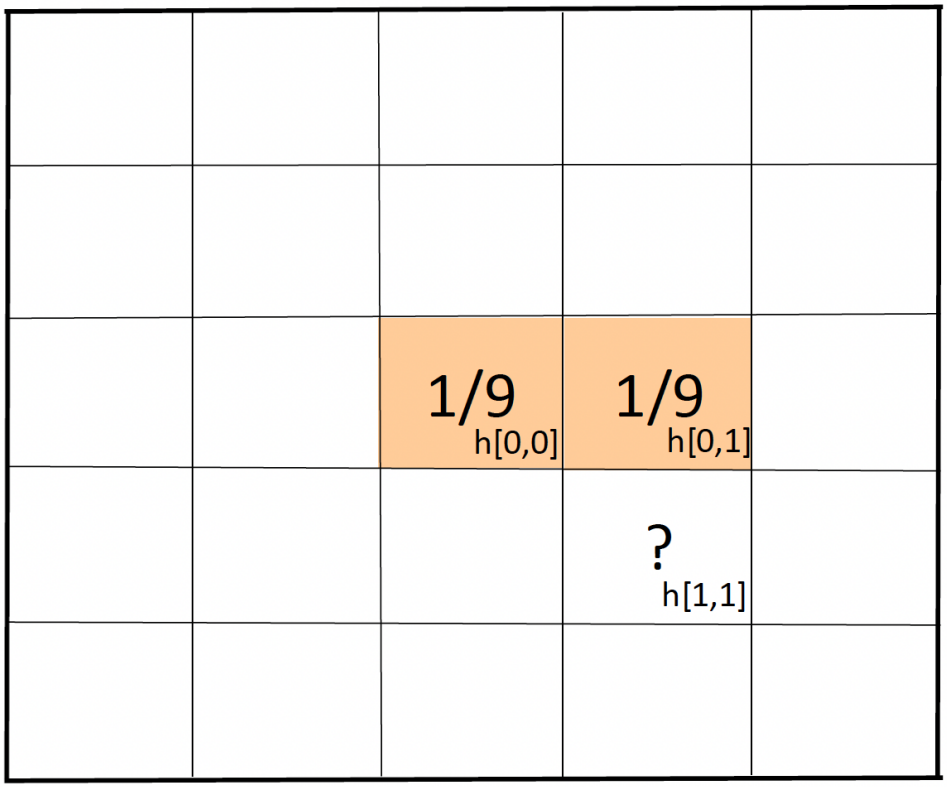
Averaging Filter and Impulse Response



$$\delta_2[n,m] \xrightarrow{S} h[n,m]$$

$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \delta_2[n-k, m-l]$$

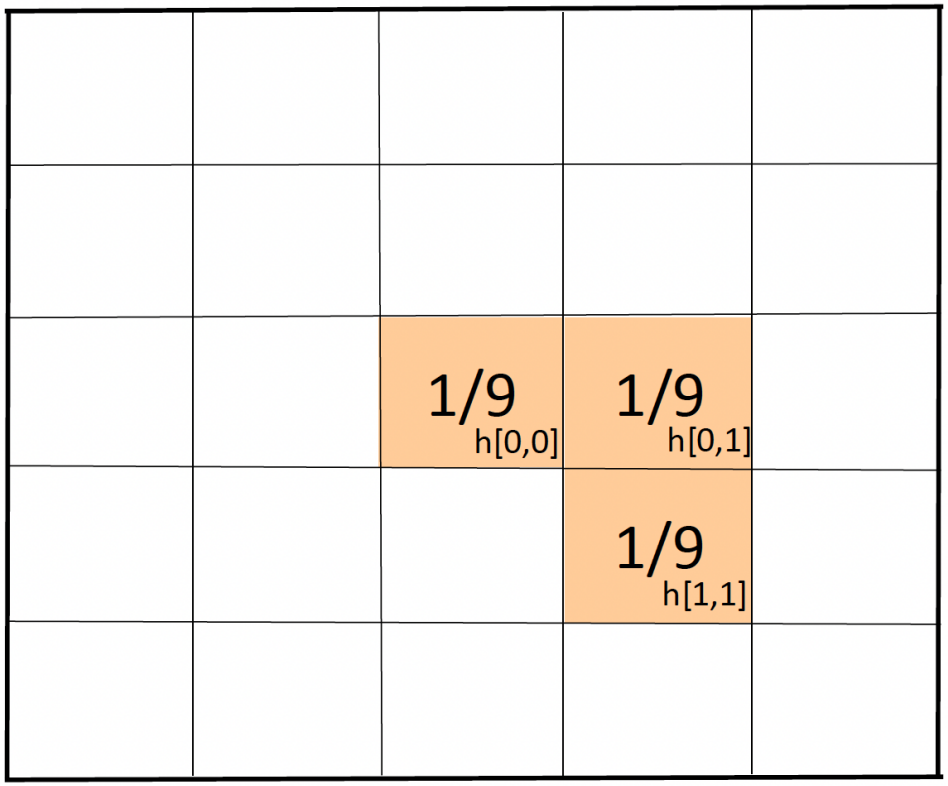
Averaging Filter and Impulse Response



$$\delta_2[n,m] \xrightarrow{S} h[n,m]$$

$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \delta_2[n-k, m-l]$$

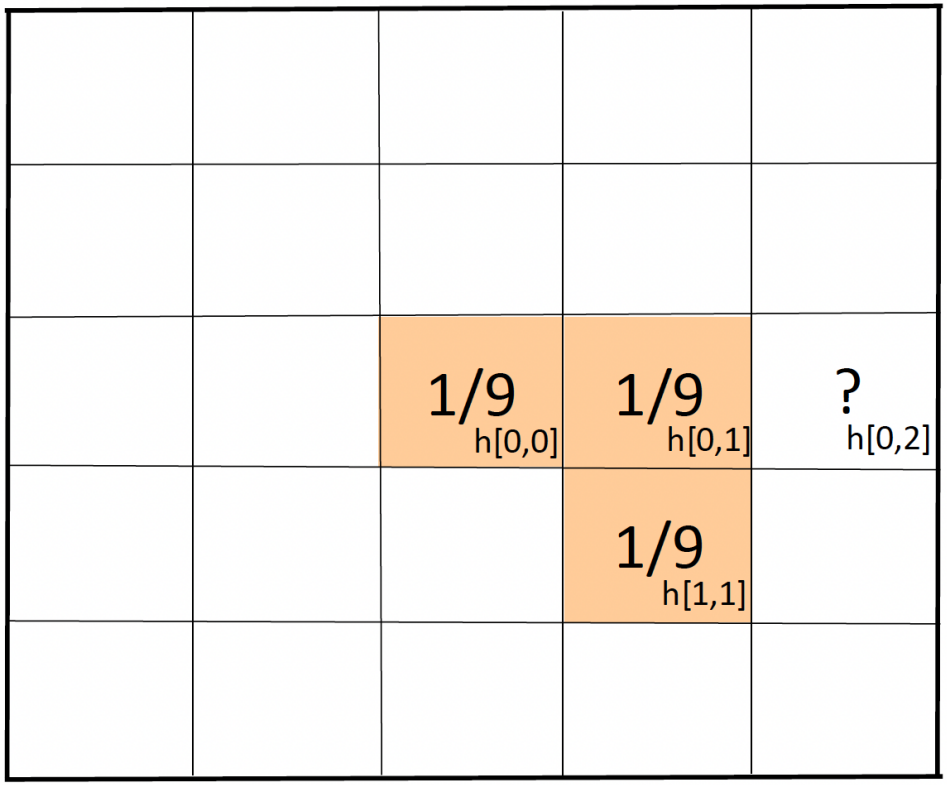
Averaging Filter and Impulse Response



$$\delta_2[n,m] \xrightarrow{S} h[n,m]$$

$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \delta_2[n-k, m-l]$$

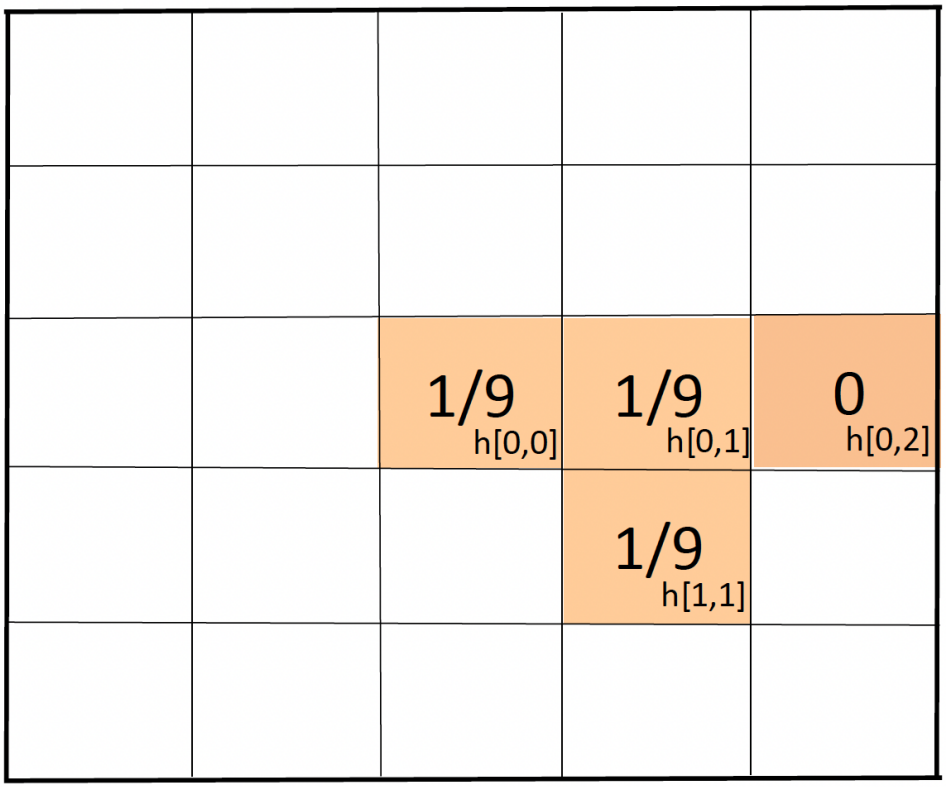
Averaging Filter and Impulse Response



$$\delta_2[n,m] \xrightarrow{S} h[n,m]$$

$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \delta_2[n-k, m-l]$$

Averaging Filter and Impulse Response



$$\delta_2[n,m] \xrightarrow{S} h[n,m]$$

$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \delta_2[n-k, m-l]$$

Averaging Filter and Impulse Response

0	0	0	0	0
0	1/9 h[-1,-1]	1/9	1/9	0
0	1/9	1/9 h[0,0]	1/9 h[0,1]	0 h[0,2]
0	1/9	1/9	1/9 h[1,1]	0
0	0	0	0	0

$$\delta_2[n,m] \xrightarrow{S} h[n,m]$$

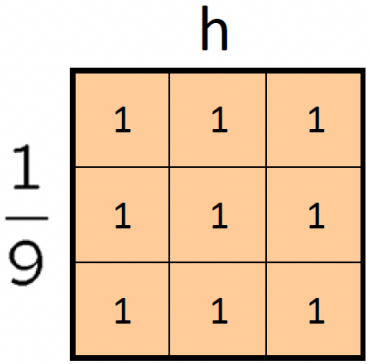
$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \delta_2[n-k, m-l]$$

Averaging Filter and Impulse Response

Impulse response of a 3×3 averaging filter

$$h[n,m] = \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \delta_2[n-k, m-l]$$

$$= \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$$



Shift-Invariant Linear Systems and Filters (LSI)

- An LSI system is fully specified by its impulse response.
  - For every input f, we can compute the output g in terms of the impulse response.
  - Next, **we will express g as a function of h.**
  - We know that an LSI system respects superposition and shift-invariance properties.
  - We know h :

$$\delta_2[n, m] \xrightarrow{\mathcal{S}} h[n, m]$$

Linear Shift-Invariant Systems and Filters (LSI)

Main Idea

Express f as a sum of impulse responses.

Consider a 3×3 image f.

f[0,0]	f[0,1]	f[1,1]
f[1,0]	f[1,1]	f[1,2]
f[2,0]	f[2,1]	f[2,2]

$=$

f[0,0]	0	0
0	0	0
0	0	0

$+$

0	f[0,1]	0
0	0	0
0	0	0

$+ \ldots +$

0	0	0
0	0	0
0	0	f[2,2]

$=$ f[0,0]*

1	0	0
0	0	0
0	0	0

$+$ f[0,1]*

0	1	0
0	0	0
0	0	0

$+ \ldots +$ f[2,2]*

0	0	0
0	0	0
0	0	1

$$= f[0,0] \cdot \delta_2[n,m] + f[0,1] \cdot \delta_2[n,m-1] + \ldots + f[2,2] \cdot \delta[n-2,n-2]$$

Linear Shift-Invariant Systems and Filters (LSI)

- More generally :

$$f[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l].\delta_2[n-k,m-l]$$

- Superposition property :

$$f[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l].\delta_2[n-k,m-l]$$

$$\xrightarrow{\mathcal{S}} \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l].\mathcal{S}\{\delta_2[n-k,m-l]\}$$

- Shift invariance :

$$\mathcal{S}\{\delta_2[n-k,m-l]\} = h[n-k,m-l]$$

Linear Shift-Invariant Systems and Filters (LSI)

- An LSI system is fully specified by its impulse response.
- For each input f, g can be computed in terms of the impulse response :

$$f[n,m] \xrightarrow{\mathcal{S}} g[n,m]$$

$$f[n,m] \xrightarrow{\mathcal{S}} \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l].h[n-k,m-l]$$

- Since a LSI system satisfies the properties of superposition and shift invariance, we also have h.
- Discrete Convolution :

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l].h[n-k,m-l]$$

Filtering

Spatial Filtering of an Image : Noise Reduction

Definition of Noise

- Random parasitic phenomenon caused by various sources (sensors, acquisition, context...).
- For linear filtering, noise is considered additive :

$$I_b(i,j) = I(i,j) + b(i,j)$$

- Examples :
  - Gaussian noise : $I_b(x,y) = I(x,y) + \mathcal{N}(0,\sigma)$
  - Salt-and-pepper impulse noise of order n : adding n random white and n random black pixels to the image.
  - ...

Fundamental hypothesis for noise reduction : the useful signal and noise have different frequency components.
Useful signal = low frequencies ; noise = high frequencies.

Linear Filter

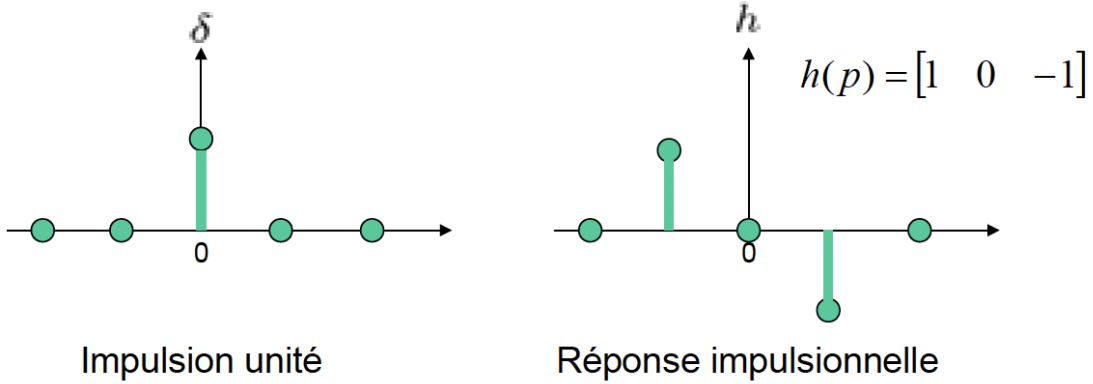
Spatial Linear Filtering : Convolution Operation

- **Spatial linear filtering** : convolving the image with the impulse response of the filter.

$$g[i,j] = (f * h)[i,j] = \sum_{m=-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} f[i-m, j-n]h[m,n]$$

En 1D

$$g(p) = \sum_{i=-\infty}^{i=+\infty} f(p-i)h(i)$$



Impulsion unité | Réponse impulsionnelle

Spatial Linear Filter

Discrete 2D Spatial Linear Filtering

- The convolution of a 2D signal $f(i,j)$ (an image) with a filter h is given by :

$$g[i,j] = (f * h)[i,j] = \sum_{m=-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} f[i-m, j-n]h[m,n]$$

- Two situations :
  - Filters with Finite Impulse Response $h(m,n)$ (FIR)
  - Filters with Infinite Impulse Response $h(m,n)$ (IIR)

Spatial Linear Filter

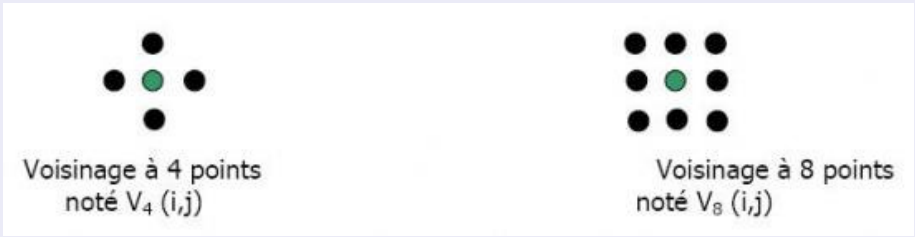
Discrete 2D Spatial Linear Filtering

1. Filters with Infinite Impulse Response $h(m, n)$ (IIR) : $h(m, n)$ is non-zero.
2. Filters with Finite Impulse Response $h(m, n)$ (FIR) : $h(m, n)$ is zero outside an interval between m and n.
   - These are called convolution masks.

Spatial Filtering of an Image : Implementation

Discrete Convolution

Filtering is a neighborhood operation : the value of a pixel is replaced by the value of a function applied to this pixel and its neighbors.



- Linear filters : the applied function is a linear combination, using convolution masks (convolution of the image with the filter's impulse response).

Linear Filter

Implementation of Linear Filters

Let I be a digital image. Let h be a function on $[x_1, x_2] \times [y_1, y_2]$ with real values (convolution kernel). The convolution of I by h is defined as :

$$(I * h)(x, y) = \sum_{i=x_1}^{x_2} \sum_{j=y_1}^{y_2} I(x - i, y - j)h(i, j) = \sum_{i=x_1}^{x_2} \sum_{j=y_1}^{y_2} I(i, j)h(x - i, y - j)$$

The new values are computed by the dot product between the convolution kernel (mask) and the pixel neighborhood.

Linear Filter : CalculationPrinciple

Example

- Let $h(n,m)$ be a square mask of size $d = 3$ (odd). Then,

$$f'(i,j) = (f * h)(i,j) = \sum_{n=-1}^{1} \sum_{m=-1}^{1} f(i-n, j-m)h(n,m)$$

- If we set $g(n,m) = h(-n,-m)$, we then obtain the weighted sum of the image with g

Linear Filter : Calculation Principle

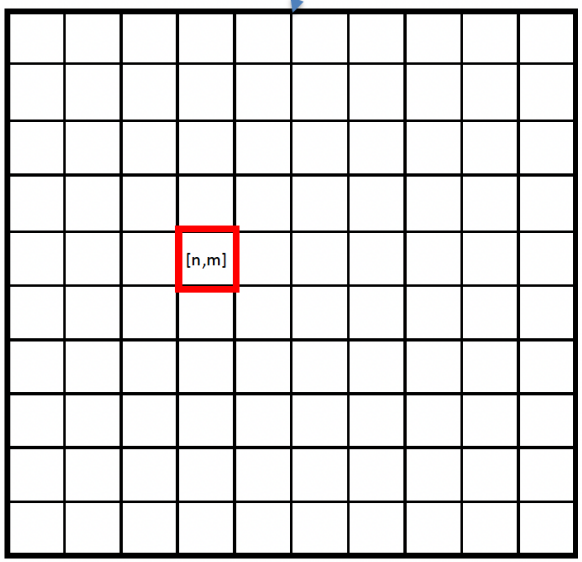
$$h = \begin{pmatrix} w_9 & w_8 & w_7 \\ w_6 & w_5 & w_4 \\ w_3 & w_2 & w_1 \end{pmatrix} \quad g = \begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{bmatrix}$$

Convolution at (i,j) by h

$$\begin{aligned} f'(i,j) &= w_1 f(i-1,j-1) + w_2 f(i-1,j) + w_3 f(i-1,j+1) \\ &+ w_4 f(i,j-1) + w_5 f(i,j) + w_6 f(i,j+1) \\ &+ w_7 f(i+1,j-1) + w_8 f(i+1,j) + w_9 f(i+1,j+1) \end{aligned}$$

Linear Filter : Calculation Principle

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l] \cdot h[n-k, m-l]$$



Output $f*h$

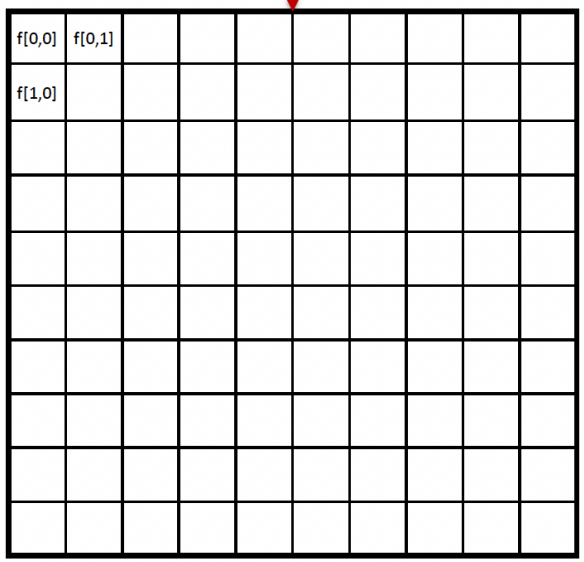


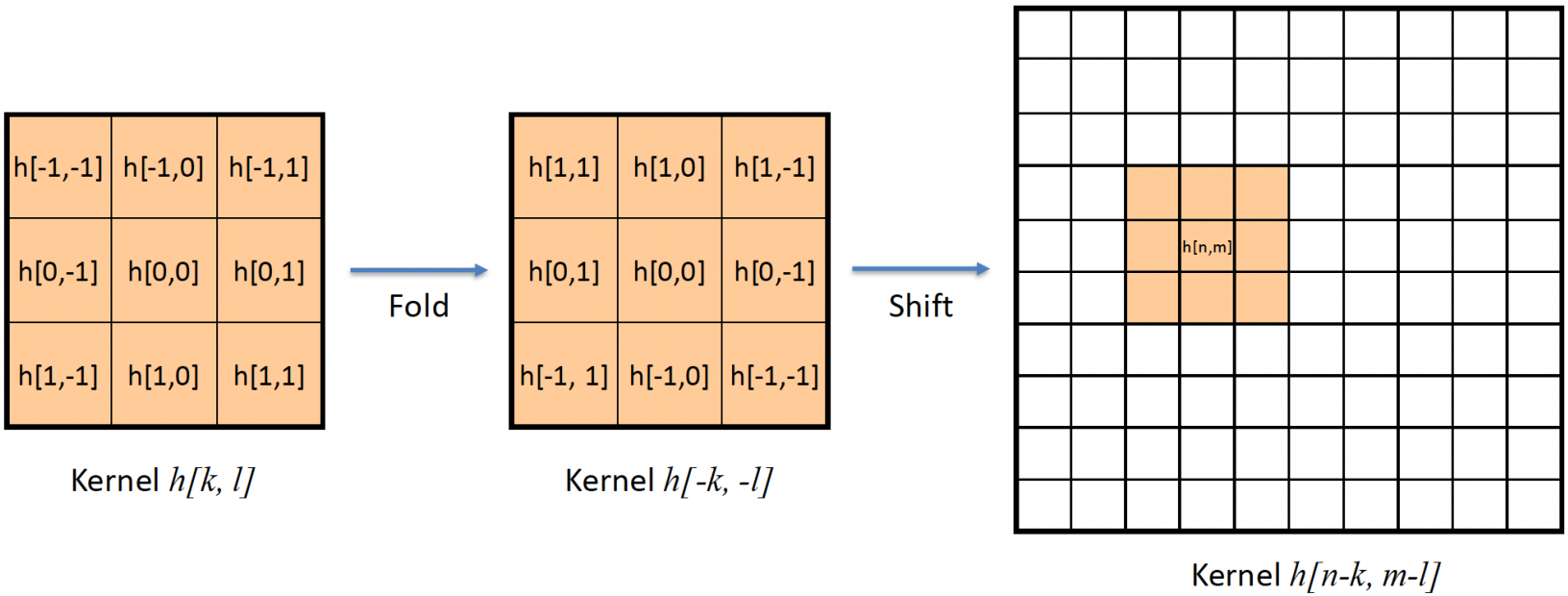
Image $f[k, l]$

h[-1,-1]	h[-1,0]	h[-1,1]
h[0,-1]	h[0,0]	h[0,1]
h[1,-1]	h[1,0]	h[1,1]

Kernel $h[k, l]$

Linear Filter : Calculation Principle

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l] \cdot h[n-k, m-l]$$



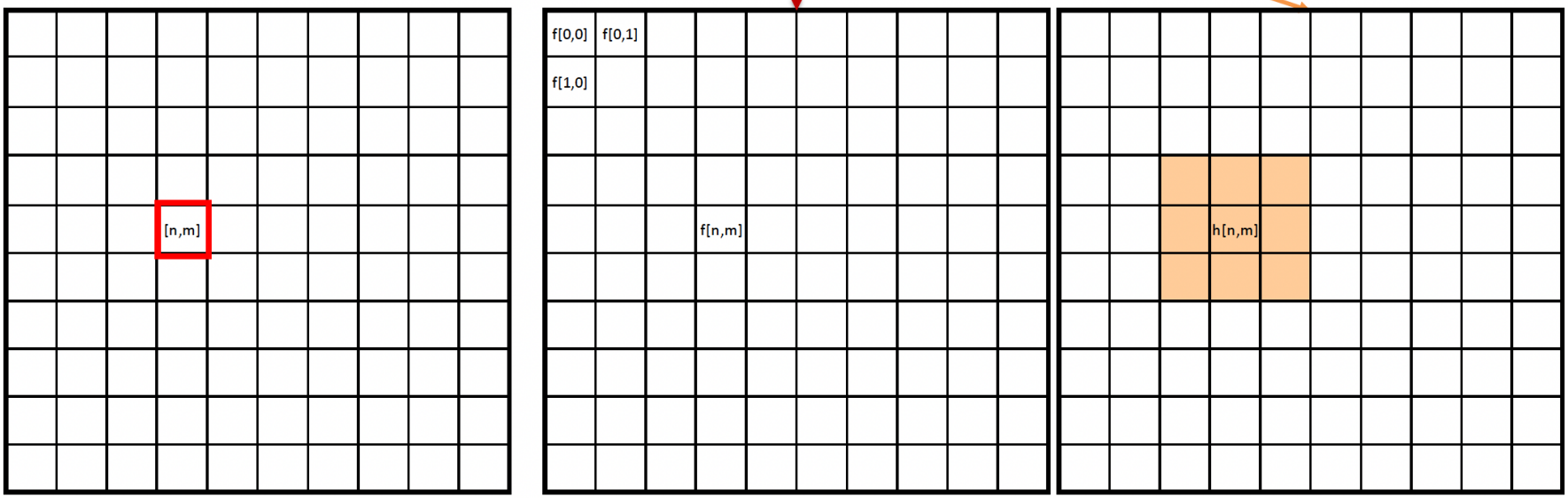
Kernel *h[k, l]*

Kernel *h[-k, -l]*

Kernel *h[n-k, m-l]*

Linear Filter : Calculation Principle

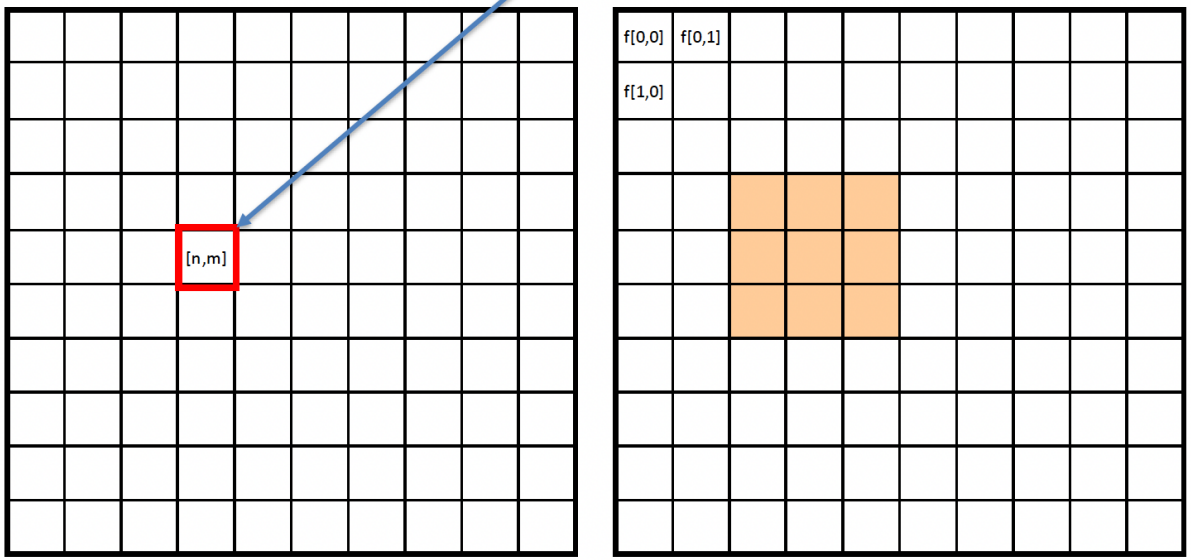
$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l] \cdot h[n-k, m-l]$$



Output $f*h$ | Image $f[k, l]$ | Kernel $h[n-k, m-l]$

Linear Filter : Calculation Principle

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l] \cdot h[n-k, m-l]$$

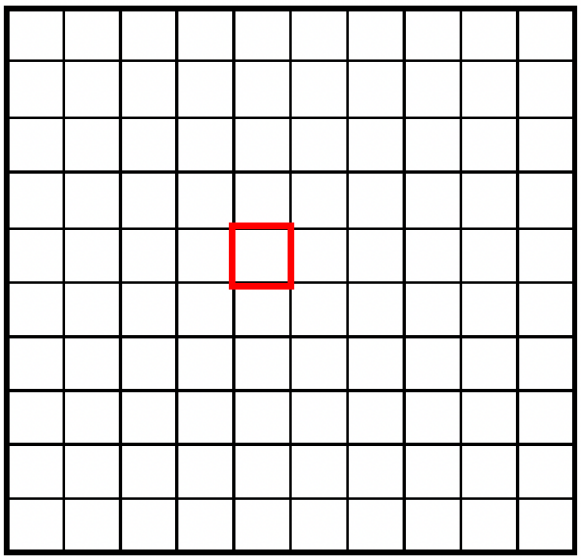


Output $f*h$

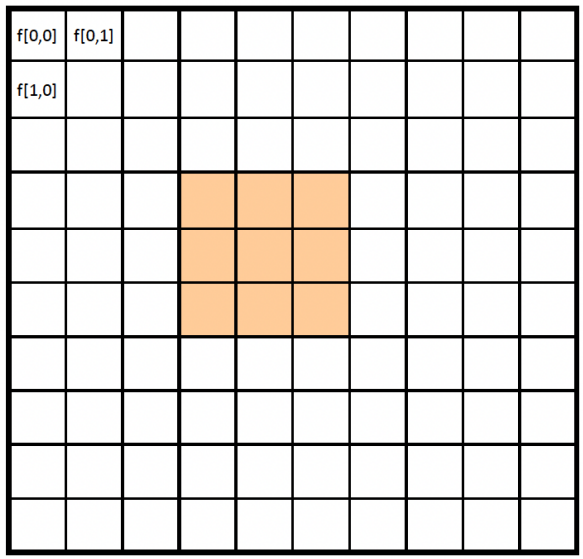
Element-wise multiplication
Image $f[k, l]$ • Kernel $h[n-k, m-l]$

Linear Filter : Calculation Principle

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l] \cdot h[n-k, m-l]$$



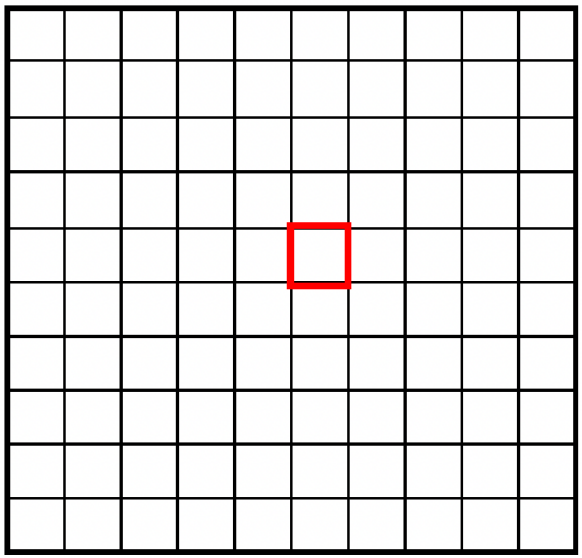
Output $f*h$



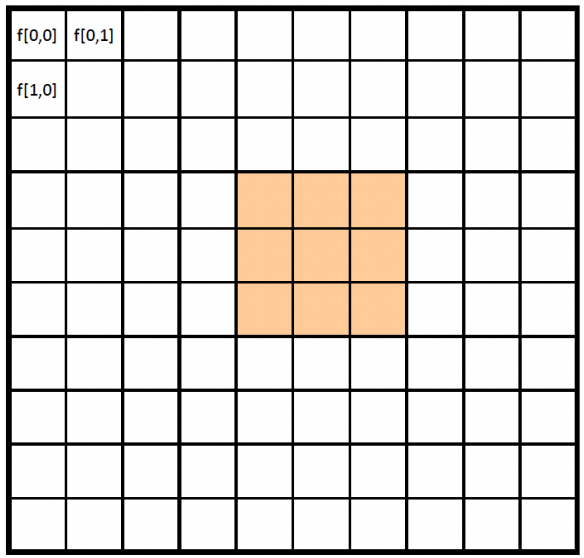
Element-wise multiplication
Image $f[k, l]$ • Kernel $h[n-k, m-l]$

Linear Filter : Calculation Principle

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l] \cdot h[n-k, m-l]$$



Output $f*h$



Element-wise multiplication
Image $f[k, l]$ • Kernel $h[n-k, m-l]$

Linear Filter : Calculation Principle

Let $h(m, n)$ be a FIR filter. At pixel $p = f(i, j)$:

- Perform a central symmetry of the convolution kernel with respect to its center :

$$h(m, n) \Rightarrow h(-m, -n) = g(m, n)$$

- Center the filter on p by superposing it onto the image.
- Perform the weighted sum between the image pixels and the filter coefficients $g(m, n)$.
- The pixel p in the output (filtered) image will have the weighted sum value.

Linear Filter : Properties

One can always represent a spatially invariant linear filter by convolution. We write :

$$h = f * g$$

- Linearity (distributive convolution) :

$$f * (g + h) = (f * g) + (f * h)$$

- Spatial invariance = independent of the pixel position :

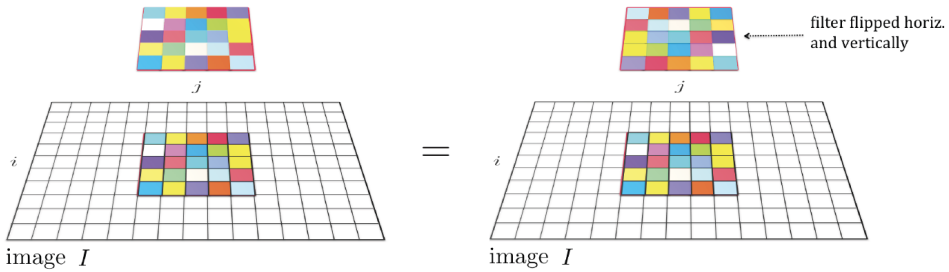
$$\tau_x(f * g) = (\tau_x f) * g = f * (\tau_x g)$$

with :

$$(\tau_x f)(y) = f(y - x)$$

LSI (Linear Shift-Invariant)

Correlation vs Convolution



Correlation | Convolution

Convolution

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{-\infty} \sum_{l=-\infty}^{-\infty} f[k,l].h[n-k,m-l]$$

Correlation or Cross-correlation

$$f[n,m] * *h[n,m] = f[n,m] \otimes h[n,m] = \sum_{k=-\infty}^{-\infty} \sum_{l=-\infty}^{-\infty} f[k,l].h[n+k,m+l]$$

Correlation is identical to convolution, except the filter kernel is not inverted. It is primarily used as a measure of similarity between h and f.

Correlation vs Convolution

Both correlation and convolution can be written as a matrix-vector multiplication if we first convert the 2D images $f(i,j)$ and $g(i,j)$ into ordered vectors f and g according to a grid.

$$\boxed{72}\boxed{88}\boxed{62}\boxed{52}\boxed{37} * \boxed{1/4}\boxed{1/2}\boxed{1/4} \Leftrightarrow \frac{1}{4}\begin{bmatrix} 2 & 1 & . & . & . \\ 1 & 2 & 1 & . & . \\ . & 1 & 2 & 1 & . \\ . & . & 1 & 2 & 1 \\ . & . & . & 1 & 2 \end{bmatrix}\begin{bmatrix} 72 \\ 88 \\ 62 \\ 52 \\ 37 \end{bmatrix}$$

Correlation vs Convolution

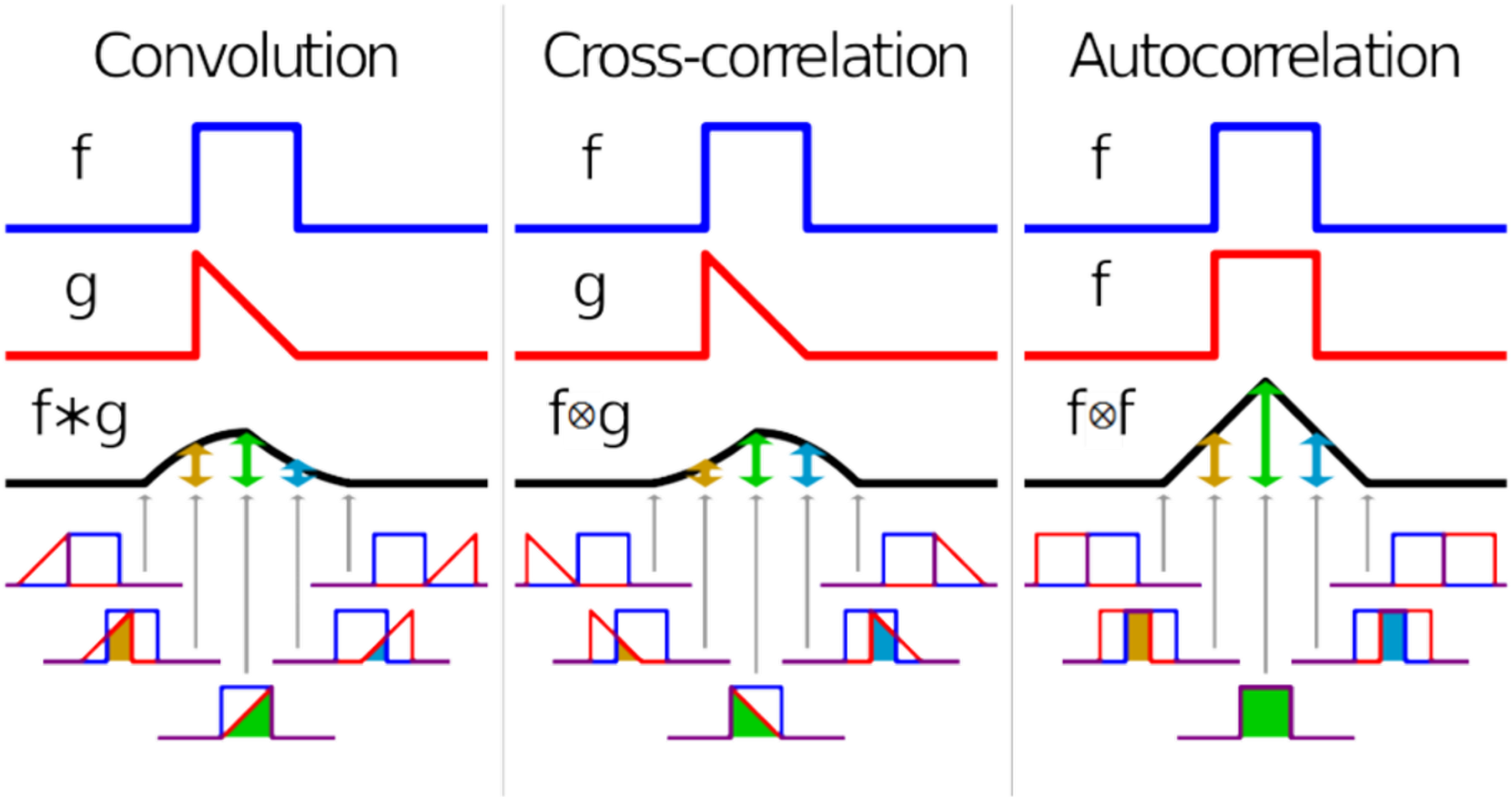


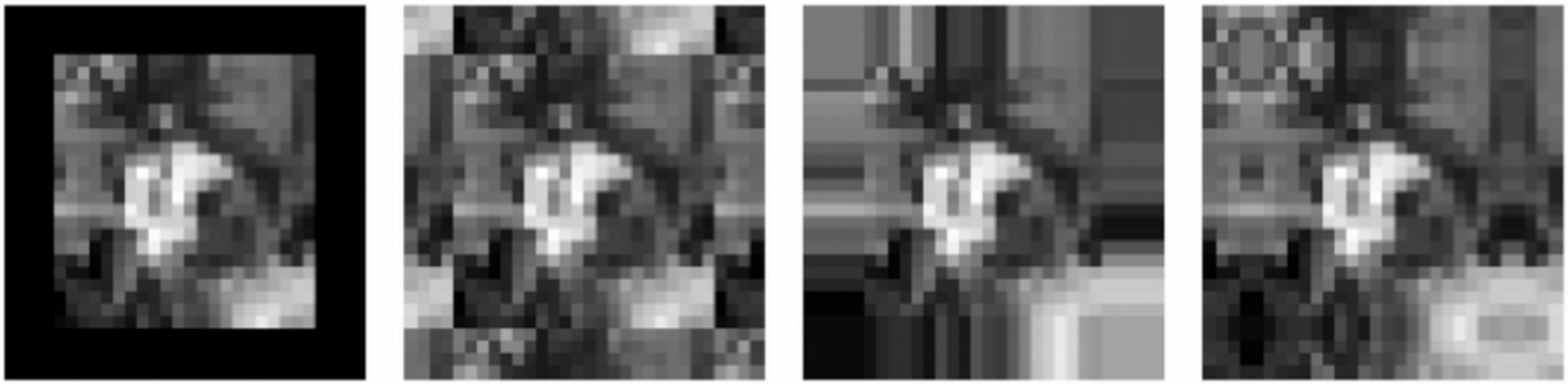
Image Filtering : Implementation

Border Issues

- Zero padding or using a constant value.
- Partial convolution on a portion of the mask.
- Mirror the image : $f(-x, y) = f(x, y)$.
- Repeat the image borders : the image is surrounded by the same values as its borders.
- No ideal solution.

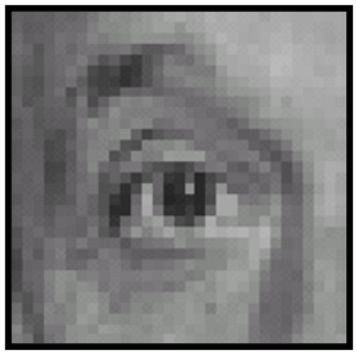
Image Filtering : Implementation

Border Issues



Source : Szeliski

Linear Filter : Example

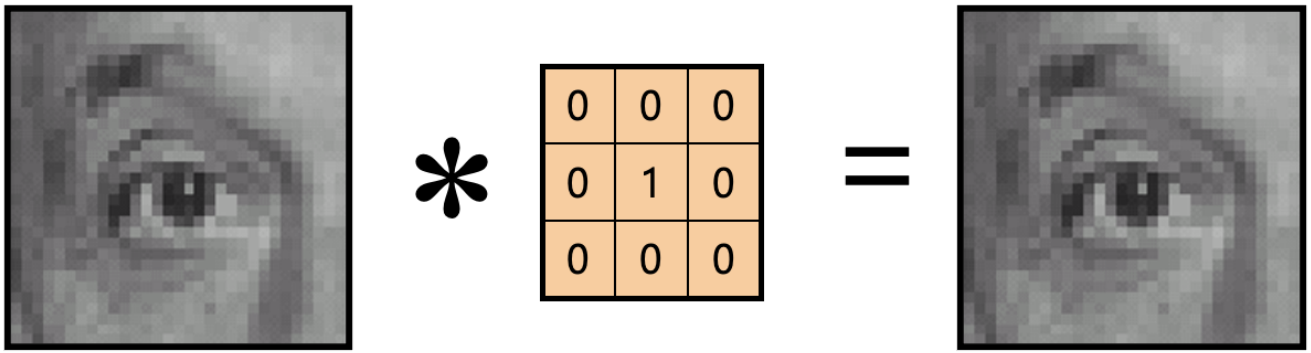


Original

$$*\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

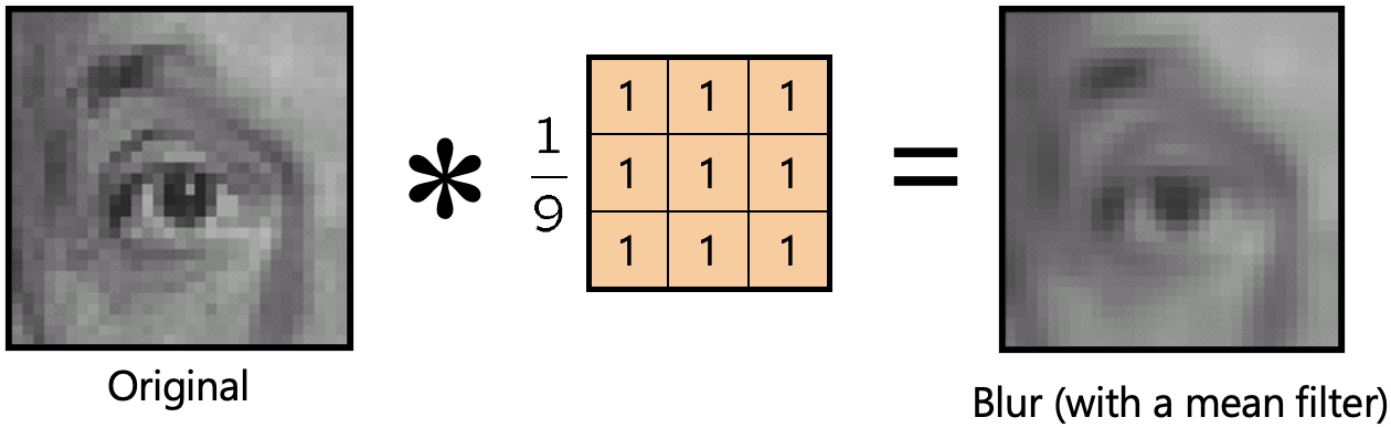
Source : Szeliski

Linear Filter : Example



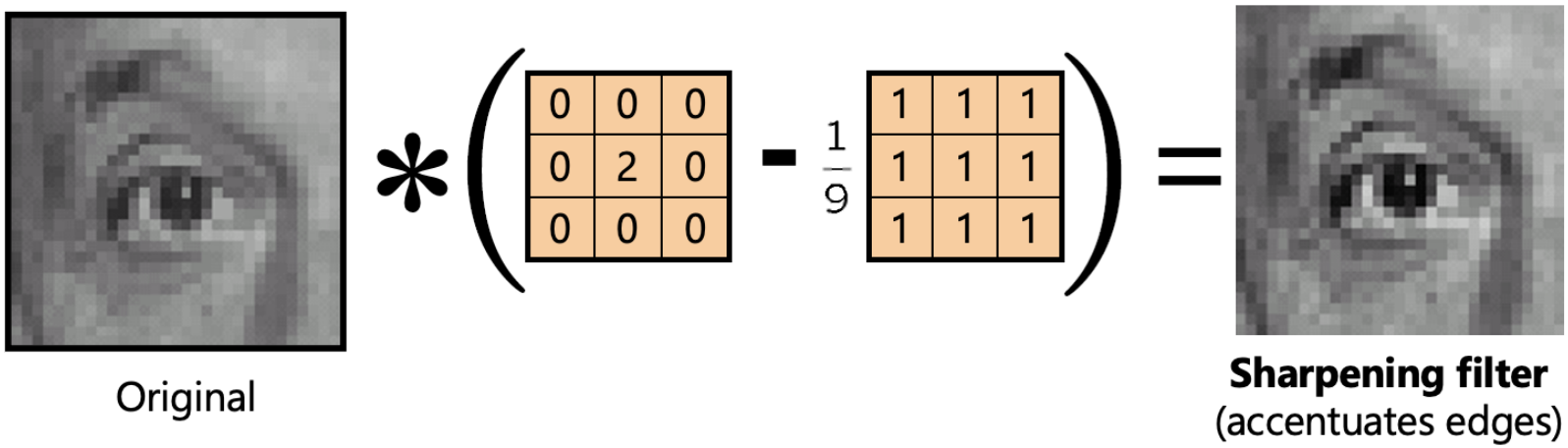
Source : Szeliski

Linear Filter : Example



Source : Szeliski

Linear Filter : Example



Source : Szeliski

Nice illustrations

- https://setosa.io/ev/image-kernels/
- https://poloclub.github.io/cnn-explainer/

Image Filtering

Isotropic vs Anisotropic Filters

- In 2D, a filter is isotropic if the filtering performed is independent of the orientation of the image structures.
- For an image, a symmetric filter is isotropic.
- An anisotropic filter reacts depending on the luminance directions in the image, useful for edge detection.

Image Filtering

Separable Linear Filters

- A response impulse g is separable along x and y if and only if :

$$g(x, y) = g_x(x) \cdot g_y(y)$$

- In terms of filtering an image by convolution :

$$h(x, y) = g(x, y) * I(x, y) = g_y(y) * (g_x(x) * I(x, y))$$

- Advantages of a separable filter :
  - We reduce a 2D signal filter to a 1D filter.
  - **Reduction of computation time**.
  - Recursive implementation of the filtering is often possible.

Image Filtering

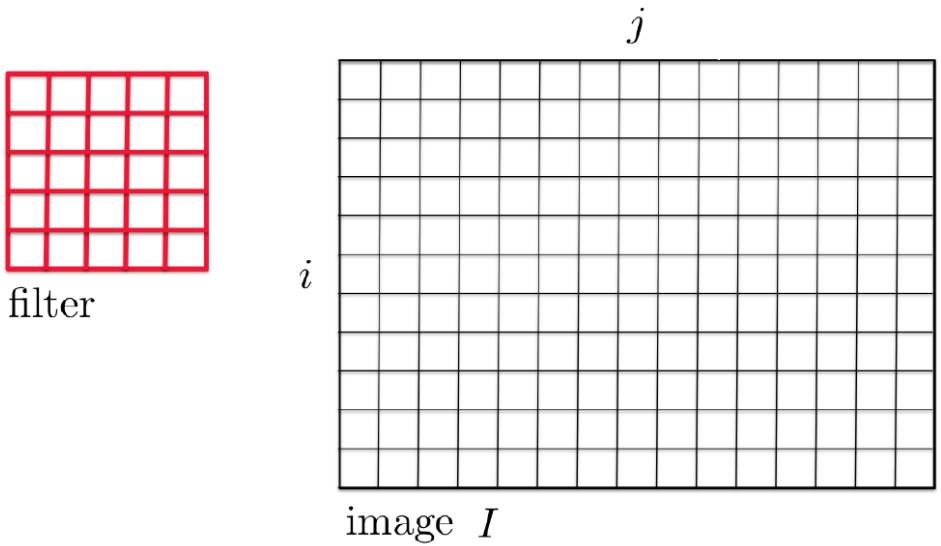


Image Filtering

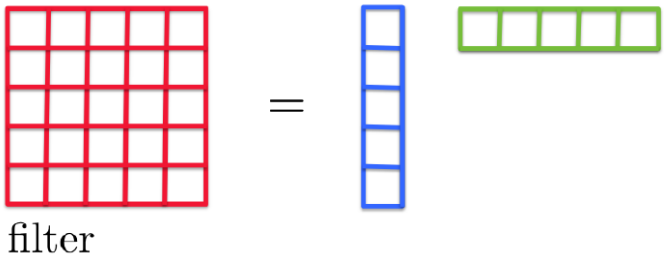


Image Filtering

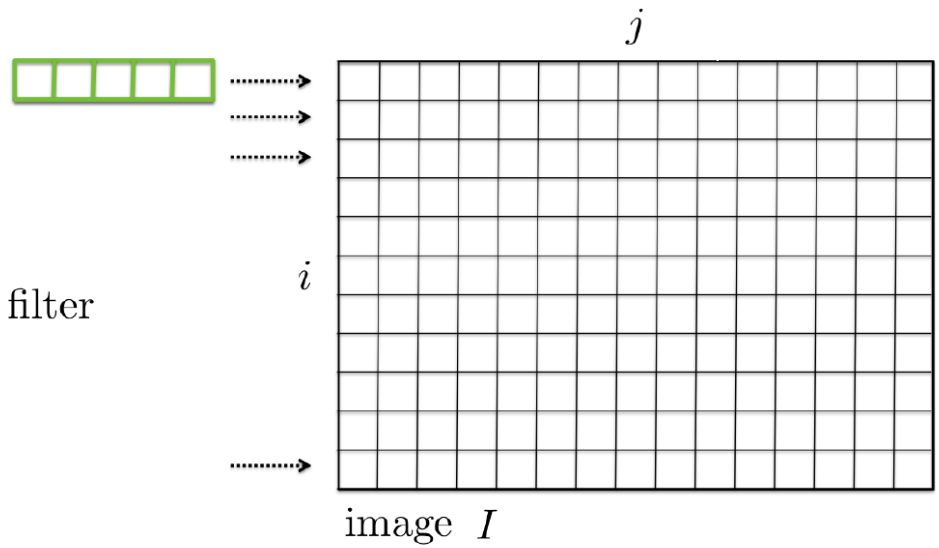


Image Filtering

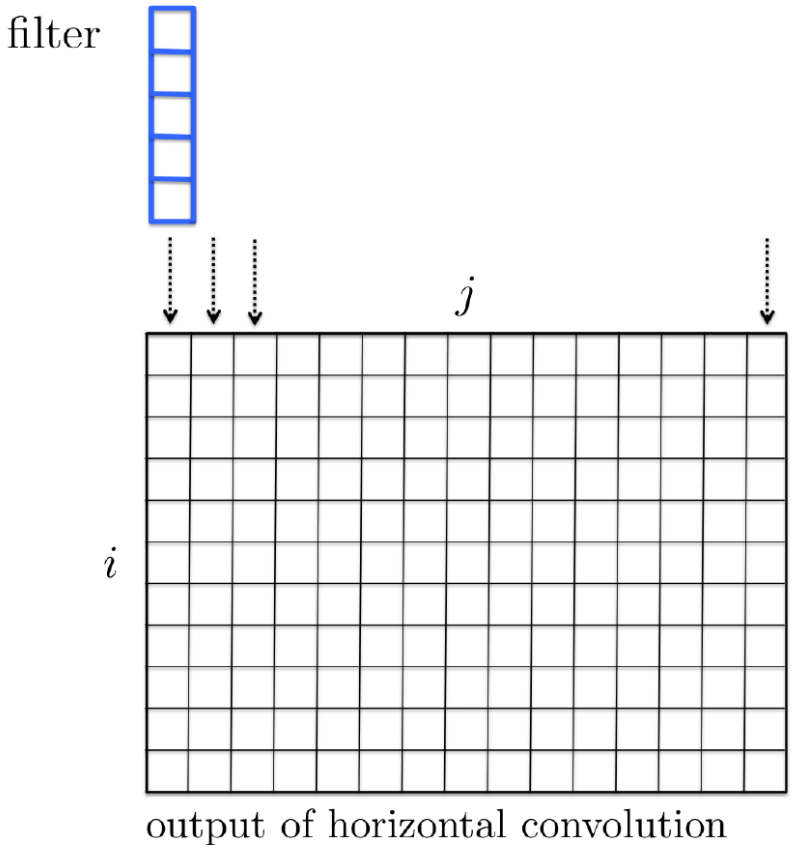


Image Filtering

Example of the Gaussian Filter

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

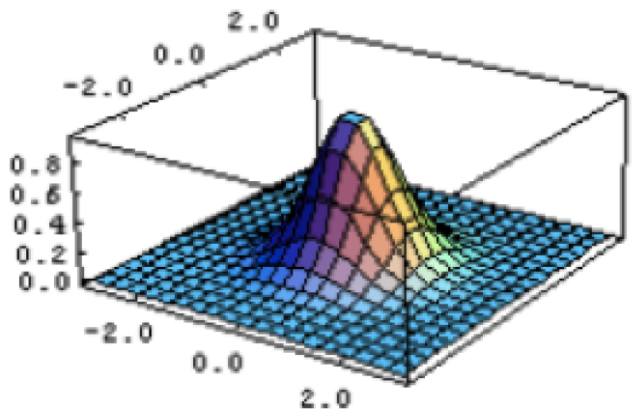
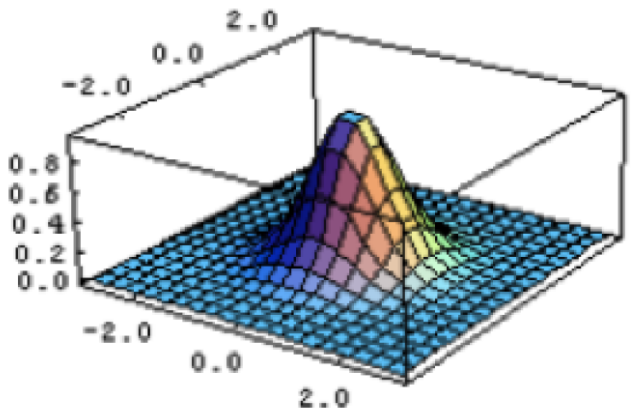


Image Filtering

Example of the Gaussian Filter

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

$= \left(\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{\sigma^2}}\right) \cdot \left(\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{y^2}{\sigma^2}}\right)$



Outline

Mean filter : linear filter

Principle

- Allows smoothing the image.
- Replaces each pixel by the average value of its neighbors.
- Reduces noise and *unimportant* details.
- Makes the image blurry (*blur edges*).
- Can be applied iteratively.
- Low-pass filter.

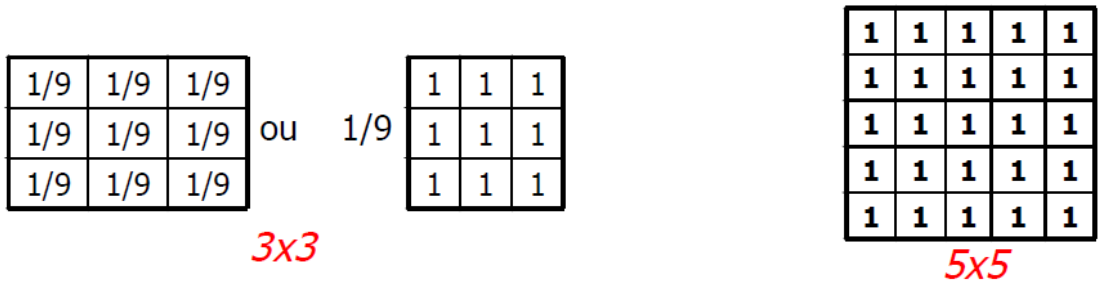


FIGURE – Source : Alain Boucher

Mean filter : example

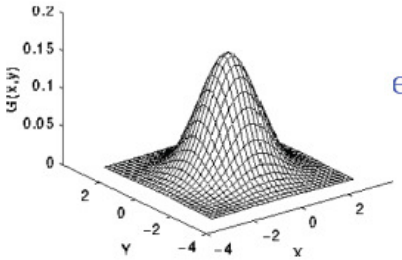
Moyenne 3x3

Moyenne 5x5

Gaussian filter

Principle

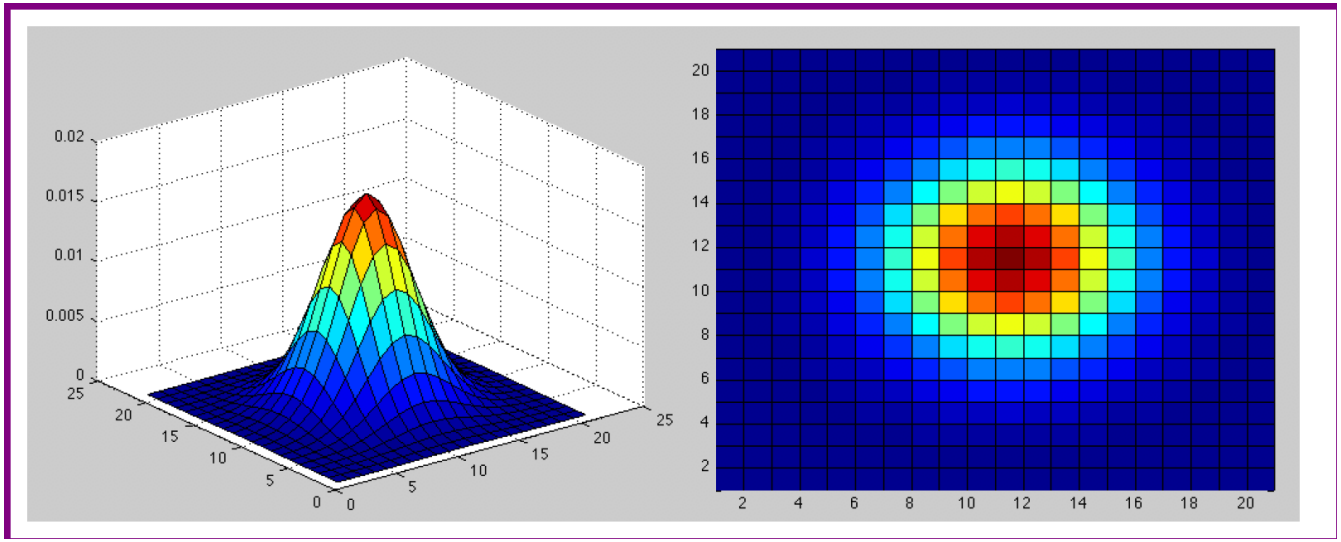
- Improvement of the mean filter : weighted averaging of the image based on the distance of the neighboring pixel.
- Gives more weight to central pixels.
- Discrete approximation of a 2D Gaussian kernel.
- The width of the filter is given by its standard deviation σ.



$$\exp - \left(\frac{x^2+y^2}{2\sigma^2} \right)$$

$$h_{3\times3} = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad h_{5\times5} = \frac{1}{246} \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Gaussian filter

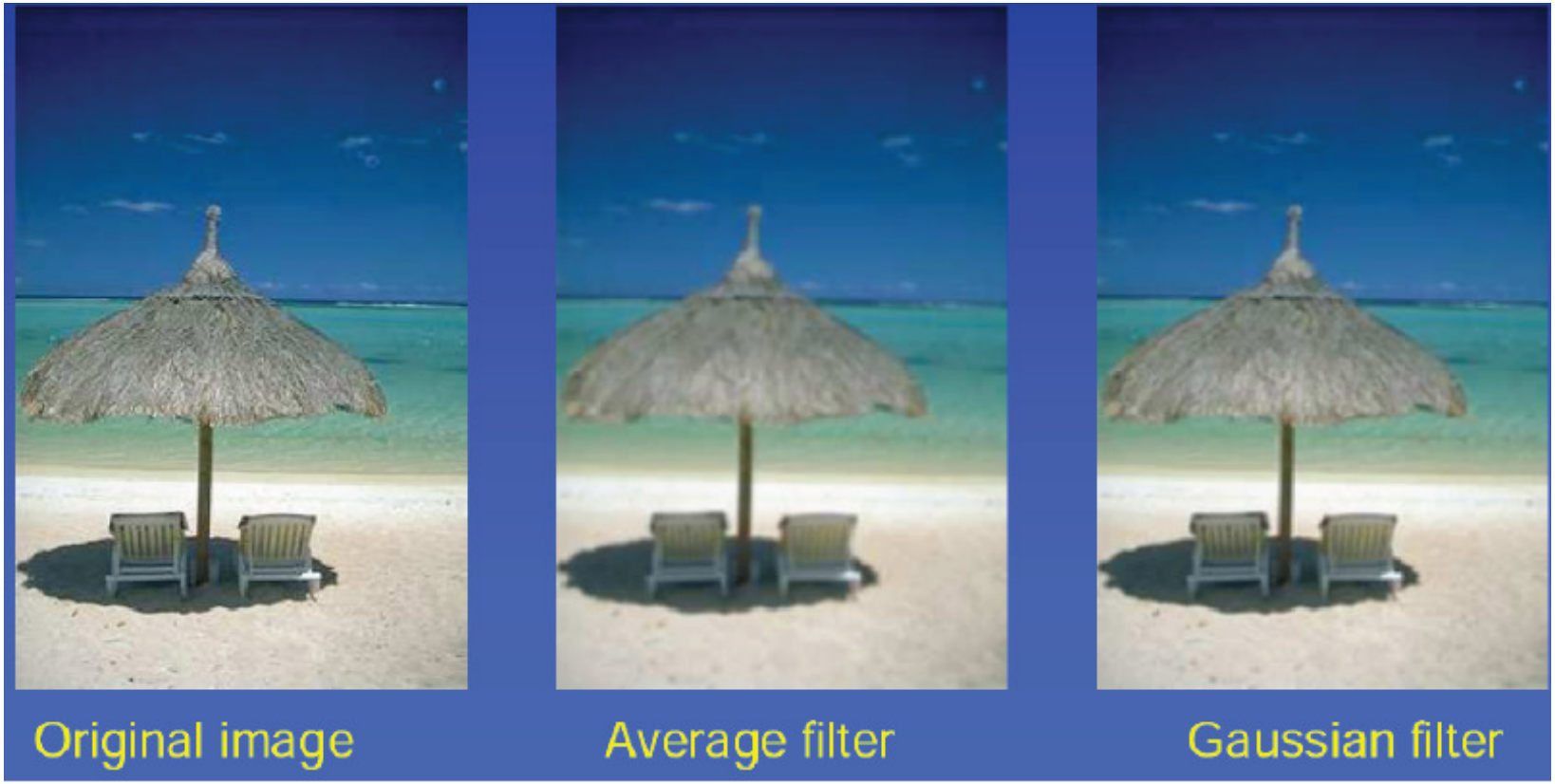


- The Gaussian kernel is defined by a set of coefficients that are samples of the 2D Gaussian.
- The width of the filter is given by its standard deviation σ, i.e., the width on either side of the central point :

$$2\mathrm{Ent}^{+}(3\sigma) + 1$$

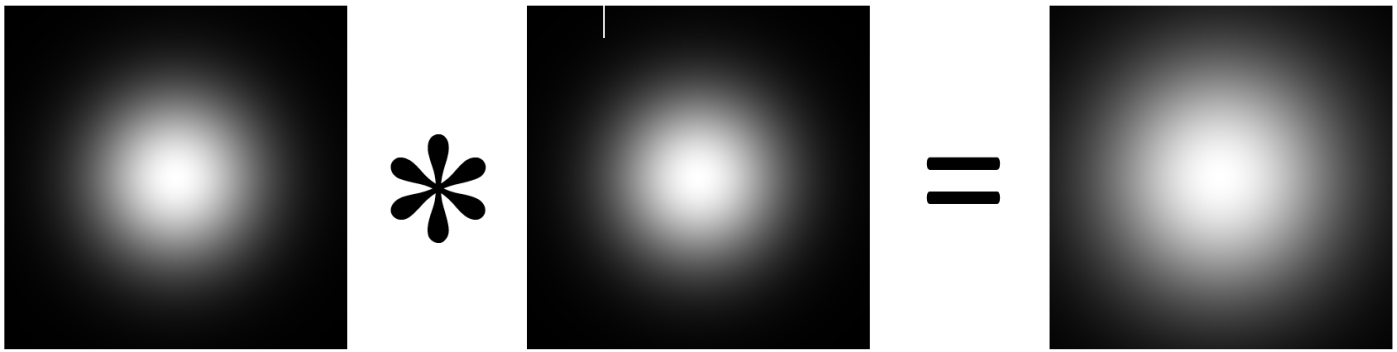
- If σ is small, smoothing has almost no effect.
- The larger σ is, the more noise is reduced, but the image becomes blurrier.

Gaussian filter : example



Gaussian filter

Convolution with itself is another Gaussian



Convolving twice with a Gaussian kernel of size σ is equivalent to convolving once with a kernel of size $\sigma\sqrt{2}$

Outline

Median filter : non-linear filter

Principle

- To remove noise in an image, there are better options than the mean filter or the Gaussian filter.
- Median filter.
- Non-linear filter that cannot be implemented as a convolution product.
- The value of a pixel is replaced by the median value of its NxN neighborhood.
- Preserves contour information while removing impulse noise.

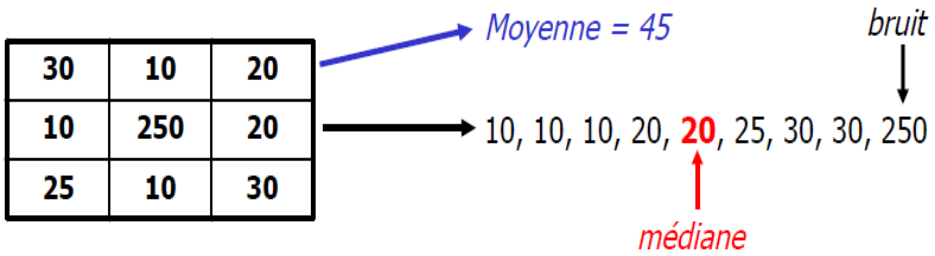
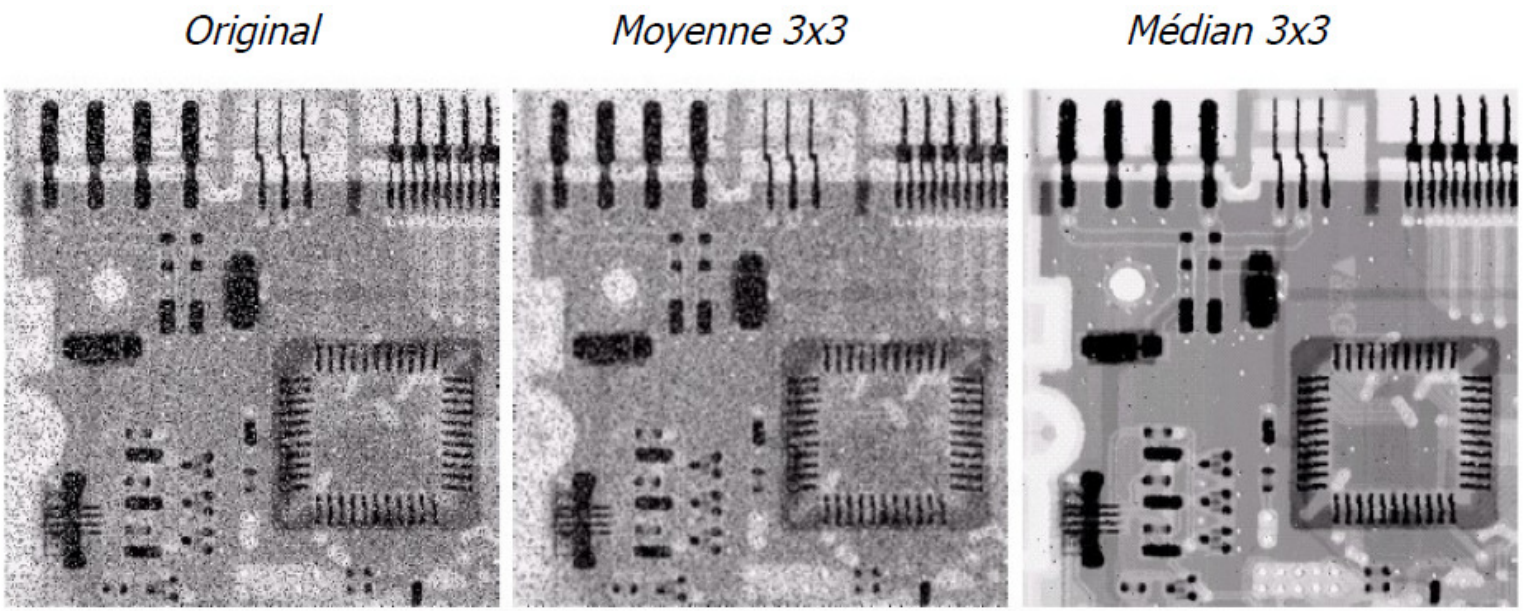


FIGURE – Source : Alain Boucher

Median filter : example



a b c

FIGURE 3.37 (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a 3×3 averaging mask. (c) Noise reduction with a 3×3 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

FIGURE – Source : Gonzales & Woods

Median filter : example

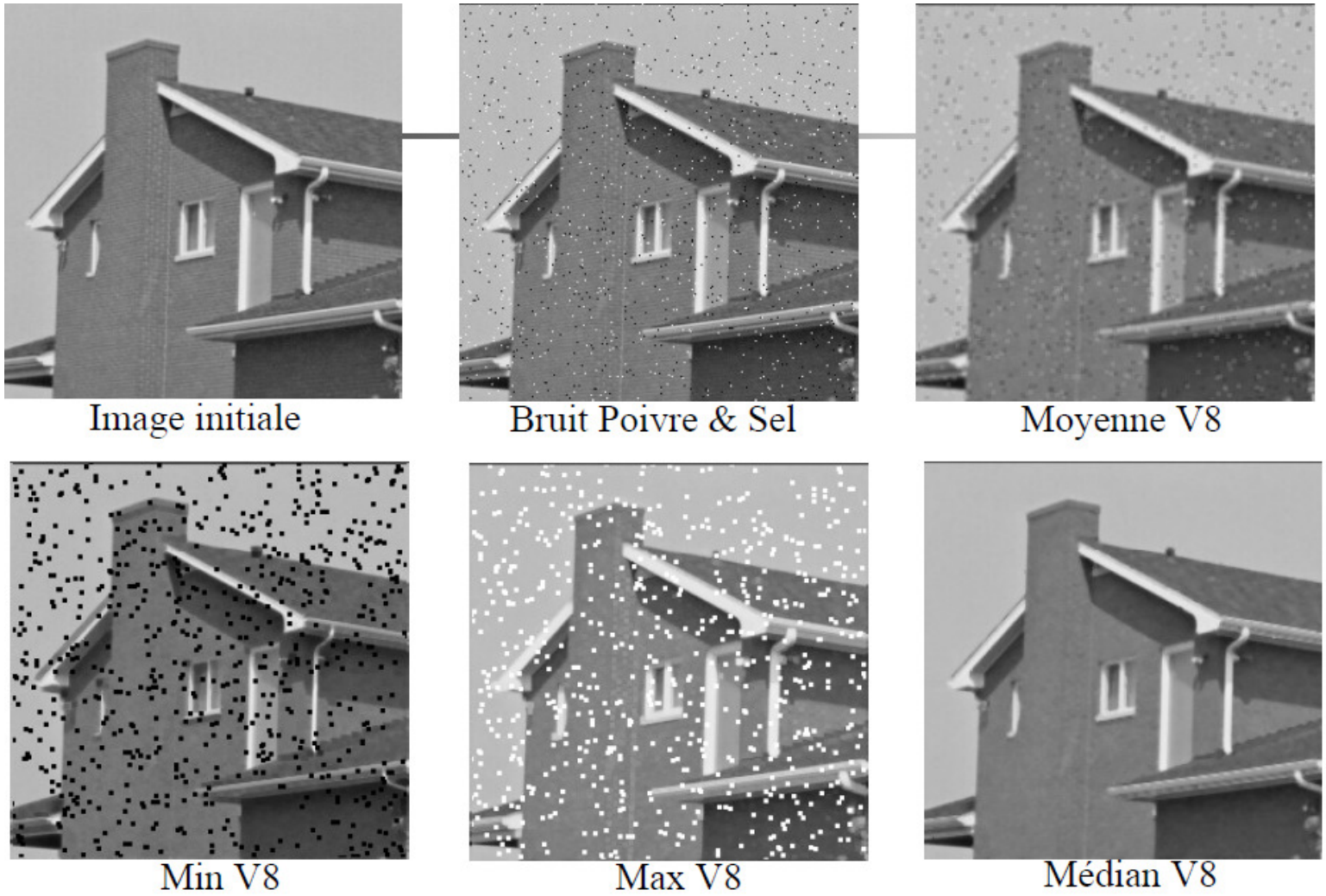


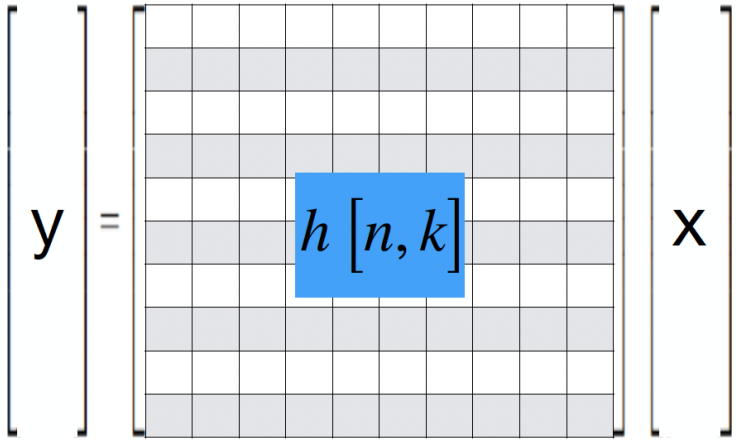
FIGURE – Source : Alain Boucher

A little parenthesis on the convolution operation

Convolutional networks.

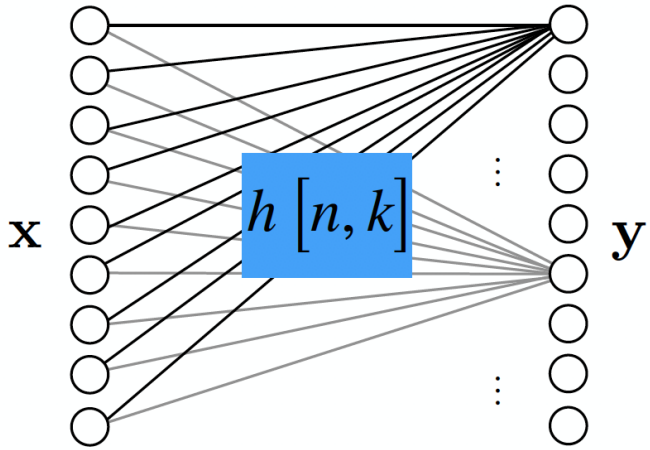
A little parenthesis on the convolution operation

A linear function f can be written as a matrix multiplication:



n indexes rows,
k indexes columns

It can also be represented as a fully connected linear neural network

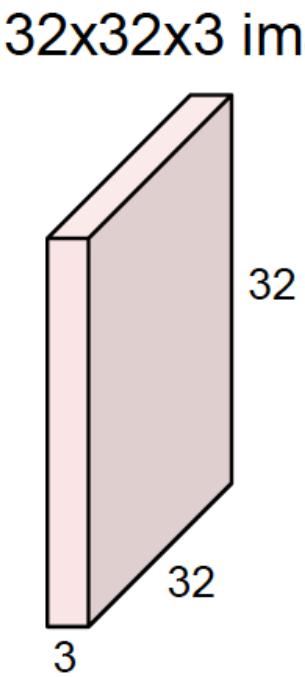


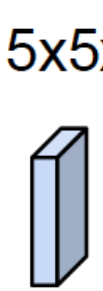
$h\,[n,k]$ Is the strength of the connection between x[k] and y[n]

FIGURE – Source : MIT course

Convolutional neural networks : a convolutional layer

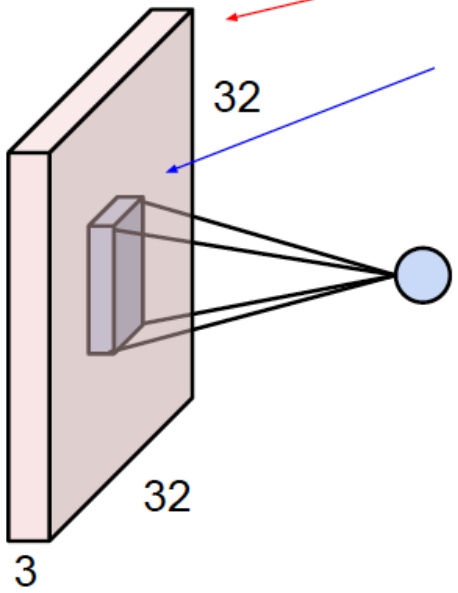
From the course by Li, Karpathy, and Johnson at Stanford





Convolve the filter with the image
i.e. “slide over the image spatially, computing dot products”

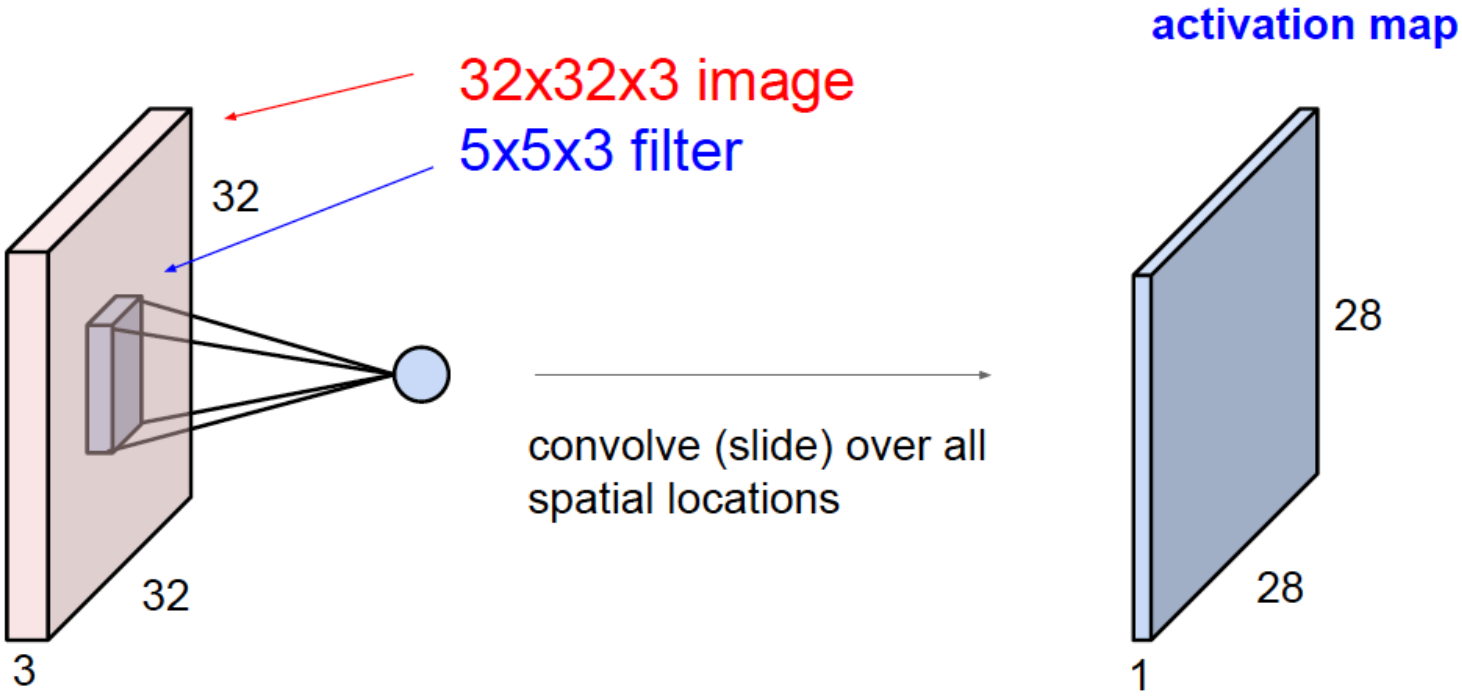
Convolutional neural networks : a convolutional layer



1 number:
the result of taking a dot product between the filter and a small 5x5x3 chunk of the image
(i.e. 5*5*3 = 75-dimensional dot product + bias)

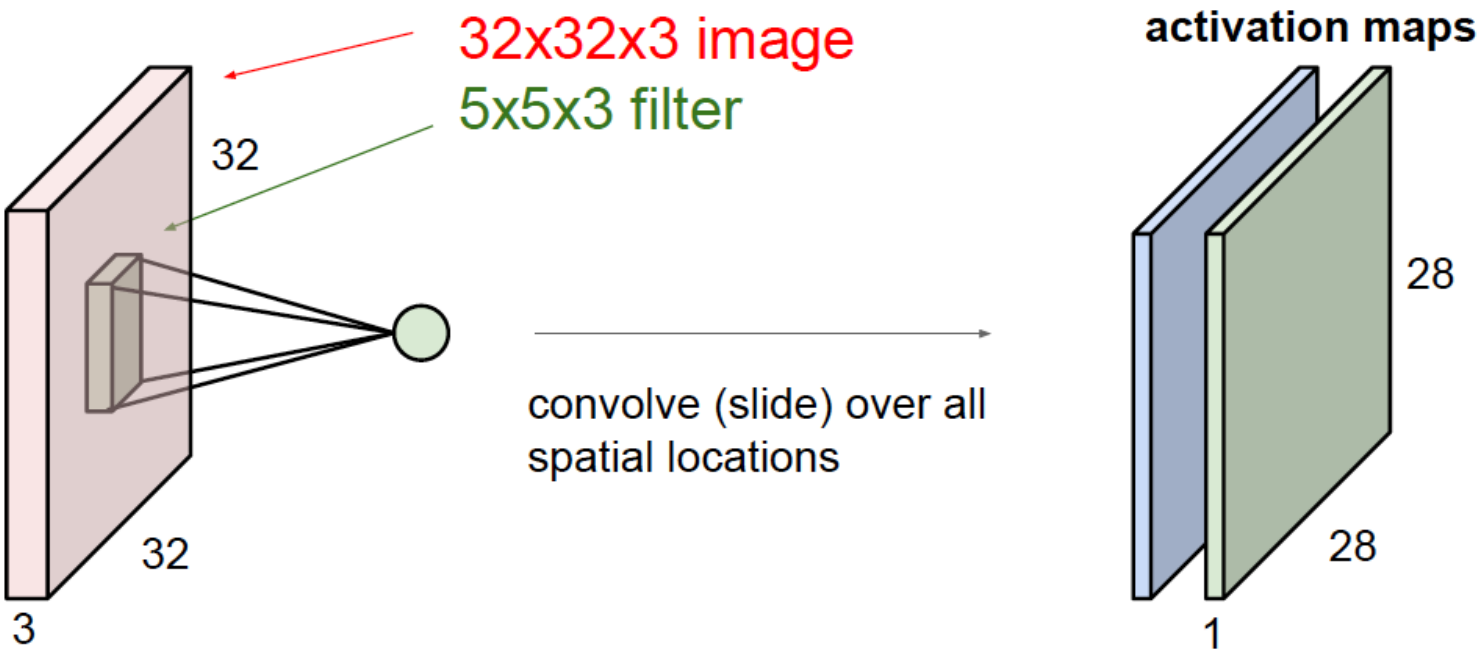
$$w^T x + b$$

Convolutional neural networks : a convolutional layer



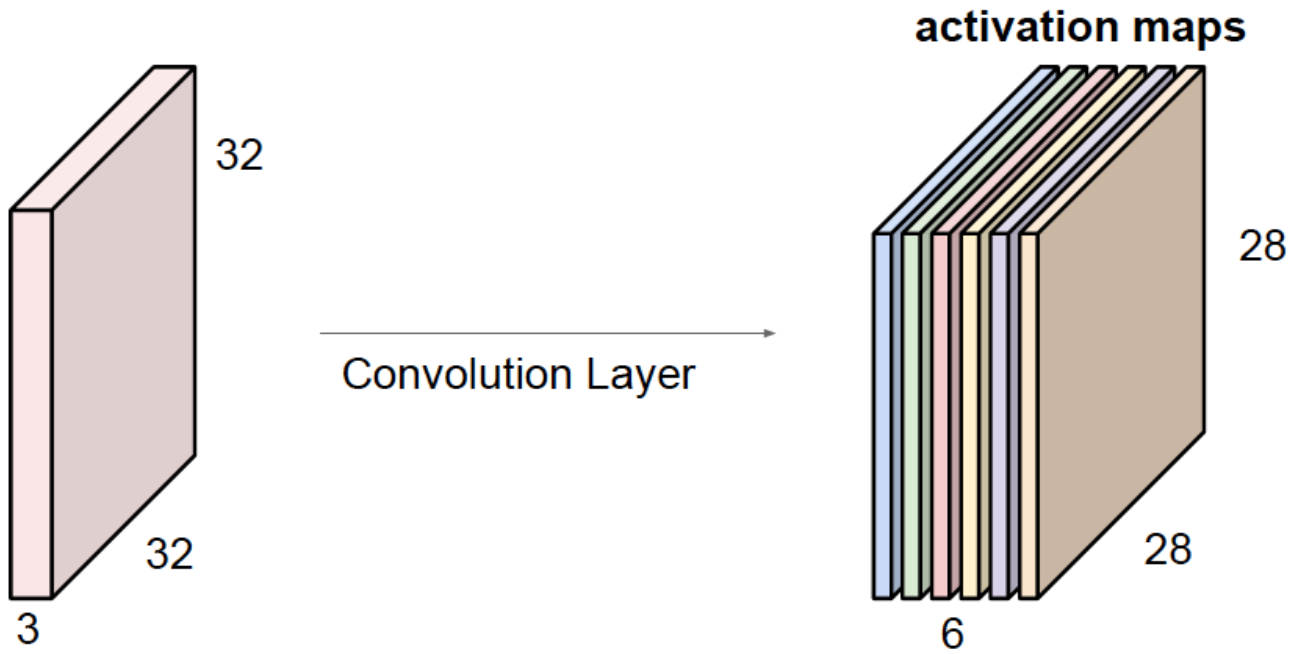
Convolutional neural networks : a convolutional layer

With another filter : the green one



Convolutional neural networks : a convolutional layer

For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:



We stack these up to get a "new image" of size 28x28x6!

Filtering

To remember

- **Correlation** : Sliding a filter over the image and comparing it with the dot product.
- **Convolution** : Rotating the filter rightward and downward, then performing the correlation.
- **Smoothing** or **blurring** an image with a Gaussian kernel : the larger the kernel σ, the blurrier it becomes.
- **Some** filters (like the Gaussian) are separable : efficient implementation by first applying a 1D convolution to each row, then a 1D convolution to each column.
- Applying a Gaussian filter with a kernel size σ_1, followed by another Gaussian filter with kernel size σ_2, is equivalent to applying a Gaussian filter with kernel size $\sigma = \sqrt{\sigma_1^2 + \sigma_2^2}$
- For more information : sections 3.2 and 3.3 of Sleziski's book.

Outline

Introduction

Why ?

- Preliminary step for many image analysis applications.
- Rich cues for scene interpretation.
- More compact representation : all the image's information summarized in the contours of different objects.
- Biologically plausible : one of the first steps in vision is to recognize contours.

http://perso.telecom-paristech.fr/~bloch/TDI/poly_contours.pdf

Introduction

A difficult task

What are the contours in this image ?

What is a contour ?

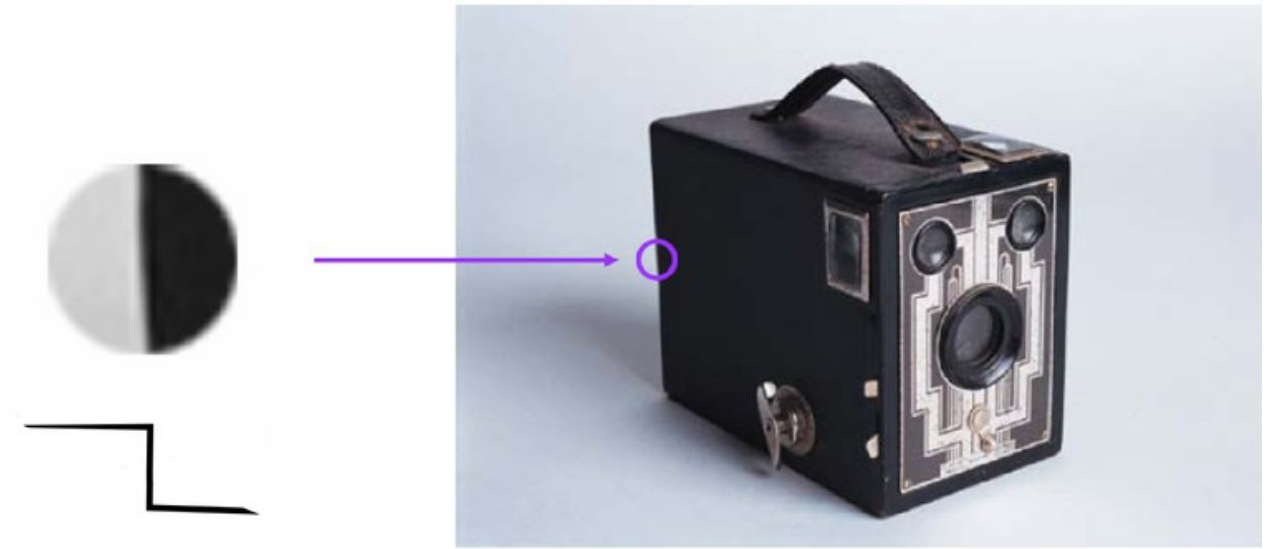


FIGURE – Source : Gluckman & Wong

A sudden change in intensity.

What is a contour ?

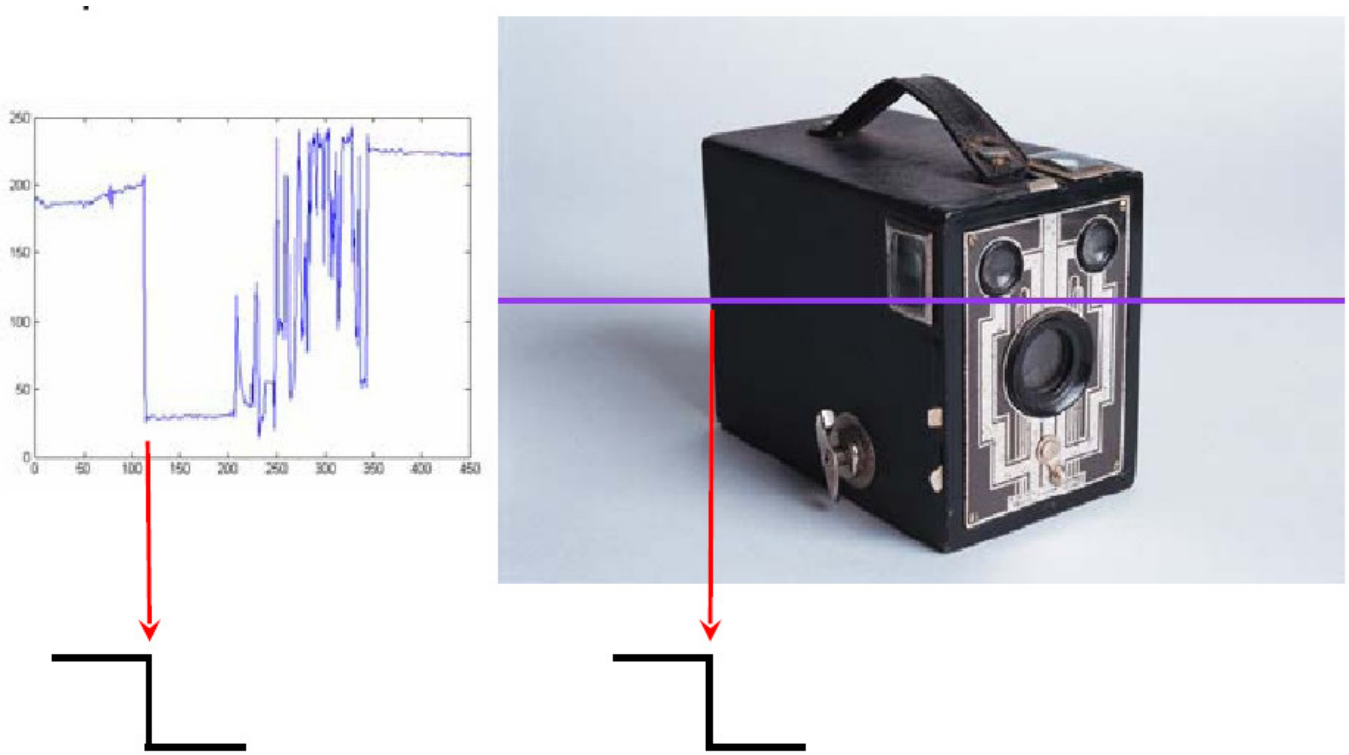


FIGURE – Source : Gluckman & Wong

What is a contour ?

FIGURE – Source : D. Hoeim

What is a contour ?

Surface orientation discontinuity

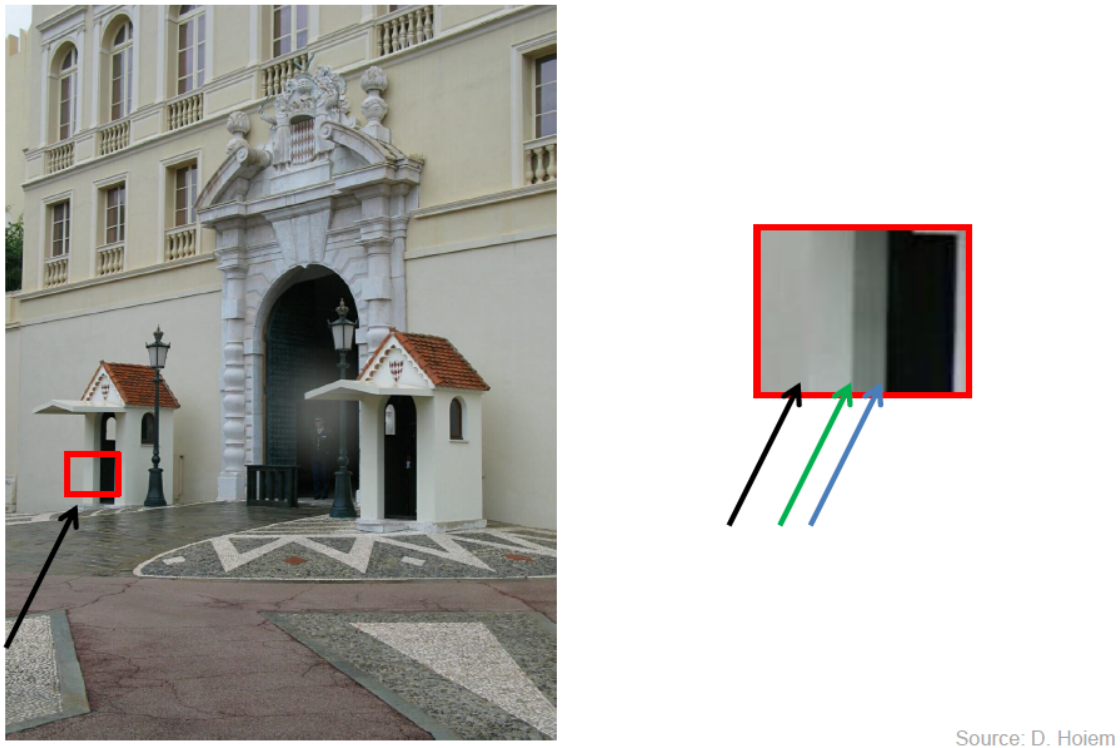


FIGURE – Source : D. Hoeim

What is a contour ?

Depth discontinuity

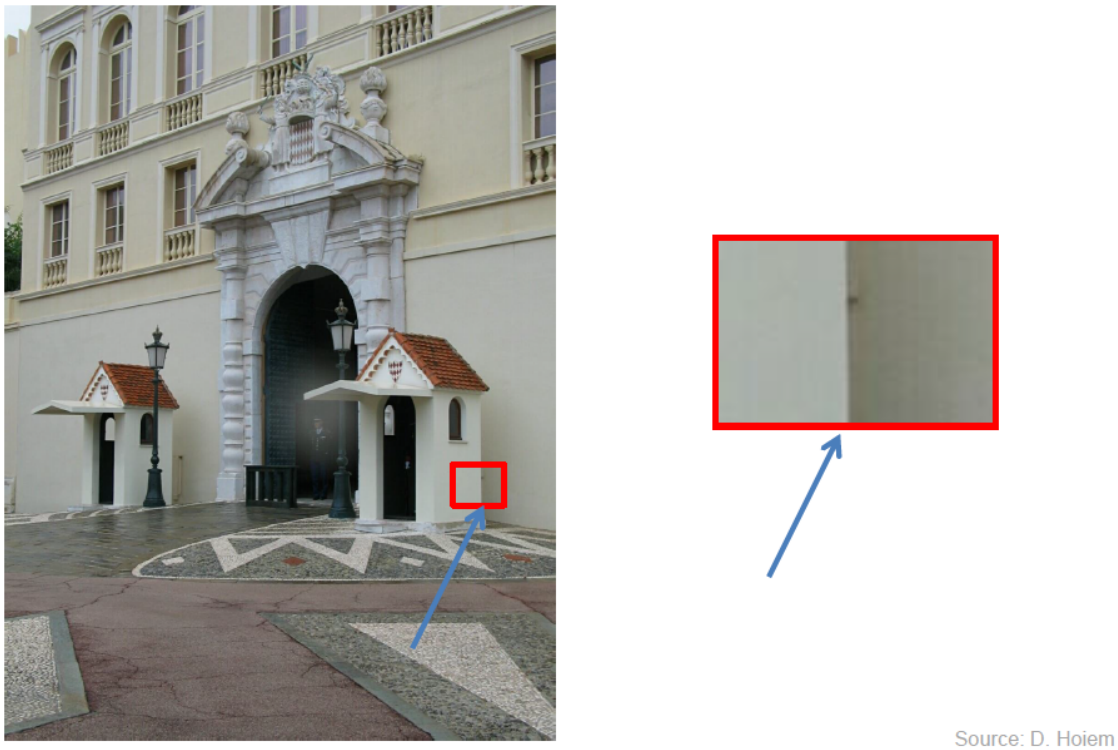


FIGURE – Source : D. Hoeim

What is a contour ?

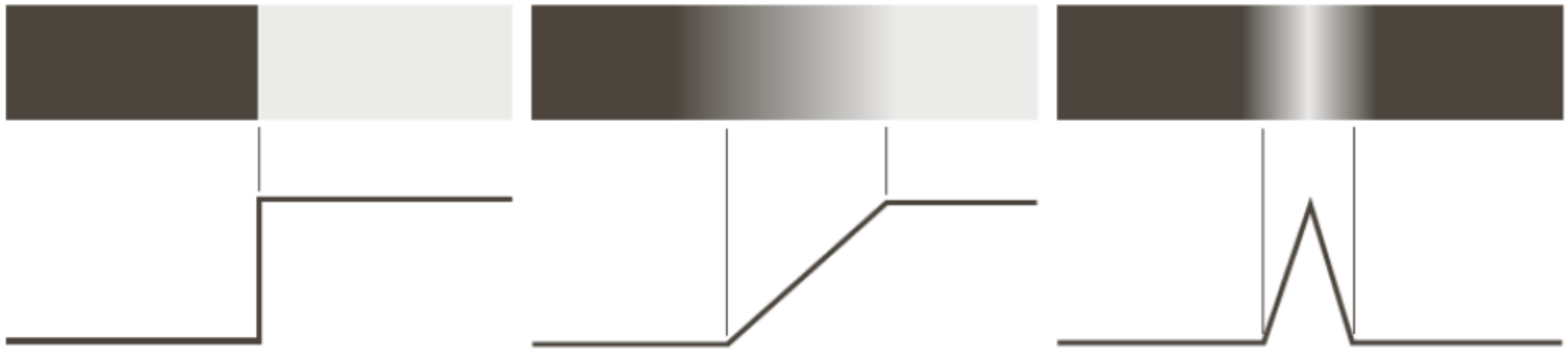
Surface color discontinuity

Source: D. Hoiem

FIGURE – Source : D. Hoeim

Types of contours

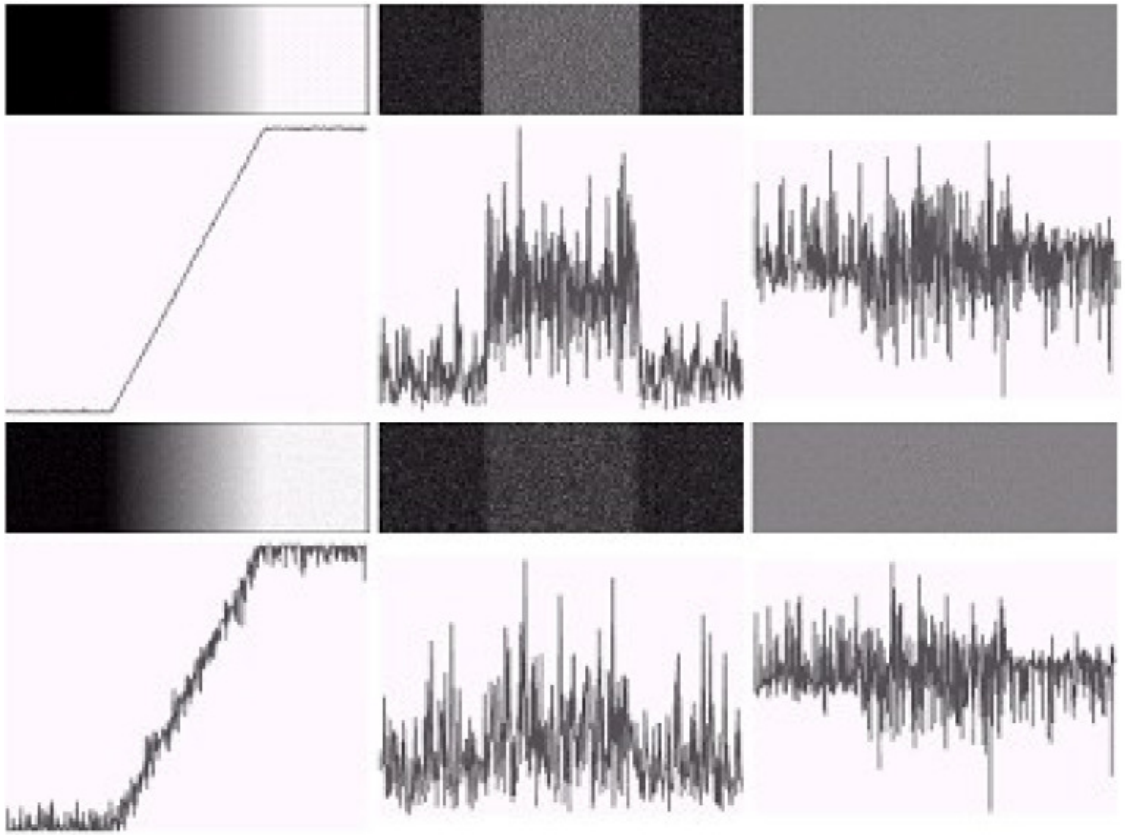
Characterized by discontinuities in the intensity function.



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

But in reality

A lot of noise



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

Edge extraction

Two-step method

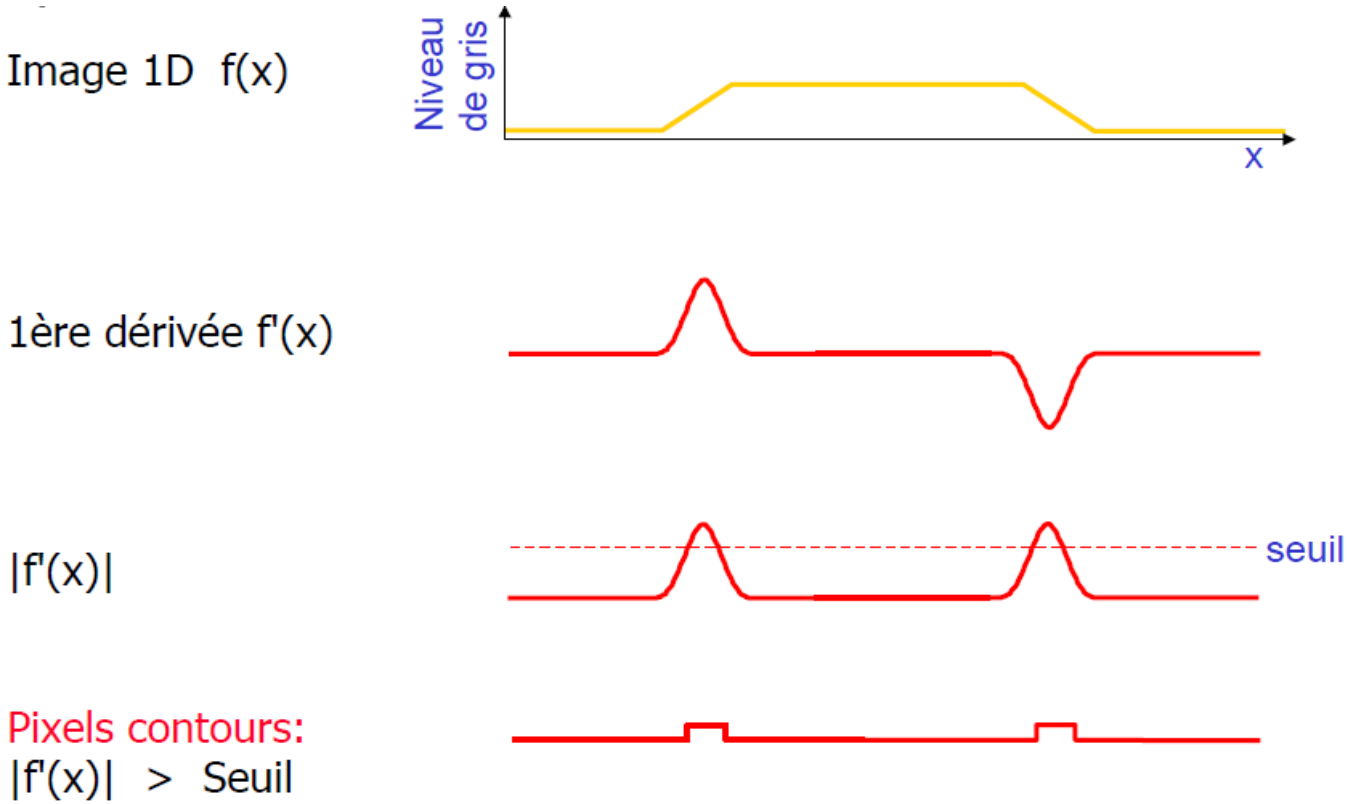
- First step : locate the edges by calculating the **gradient or Laplacian** in preferred directions while quantifying the importance of the contour.
- Second step : isolate the edges from the rest of the image using a **thresholding** technique.

Edge extraction

Principle : two steps

- High-pass filter
  - Gradient approach
  - Laplacian approach
- Thresholding
  - Simple thresholding
  - Hysteresis thresholding

Image derivative and edge



Source : Alain Boucher

Study of the derivatives of the intensity function in the image

Image gradient

Reminder

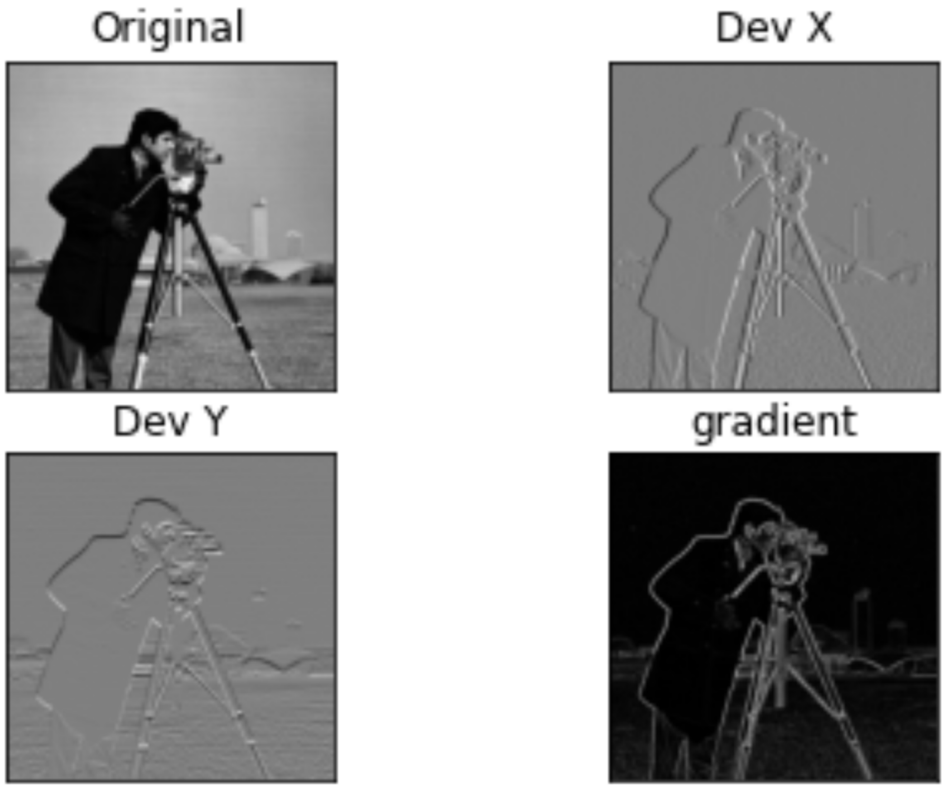
The image is a function :
$I : S \rightarrow \Omega$
$(x, y) \rightarrow I(x, y)$

First derivative of the image

In two dimensions, this is the gradient of the image, a vector representing the variation of intensity as a function of position :

$$\nabla I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right)$$

Image Gradient : Example



Plan

Image Gradient : Edge Detection

Thresholding of the gradient : edge points in an image are characterized by local extrema of the gradient.

Approach 1 : Principle

1. Discrete gradient approximations by finite differences (convolution)
2. Calculation of the gradient magnitude at every point in the image.
3. Local maxima in the gradient direction
4. Non-maximum suppression
5. Thresholding
6. Linking

First-order approach : gradient approximation

Image Gradient : Implementation - Discrete Derivative

Discrete Gradient Approximations by Finite Differences

$$\nabla_x I(x,y) = I(x,y) - I(x-n,y)$$

or :

$$\nabla_x I(x,y) = I(x+n,y) - I(x-n,y)$$

with, generally, $n = 1$

Calculation of derivatives by convolving the image with a difference mask.

Image Gradient : Implementation - Discrete Derivative

Gradient Approximation

- $\nabla_x I \approx I(x,y) - I(x-1,y)$ ($\nabla_y I \approx I(x,y) - I(x,y-1)$)
- $\nabla_x I \approx I(x+1,y) - I(x-1,y)$

-1	1

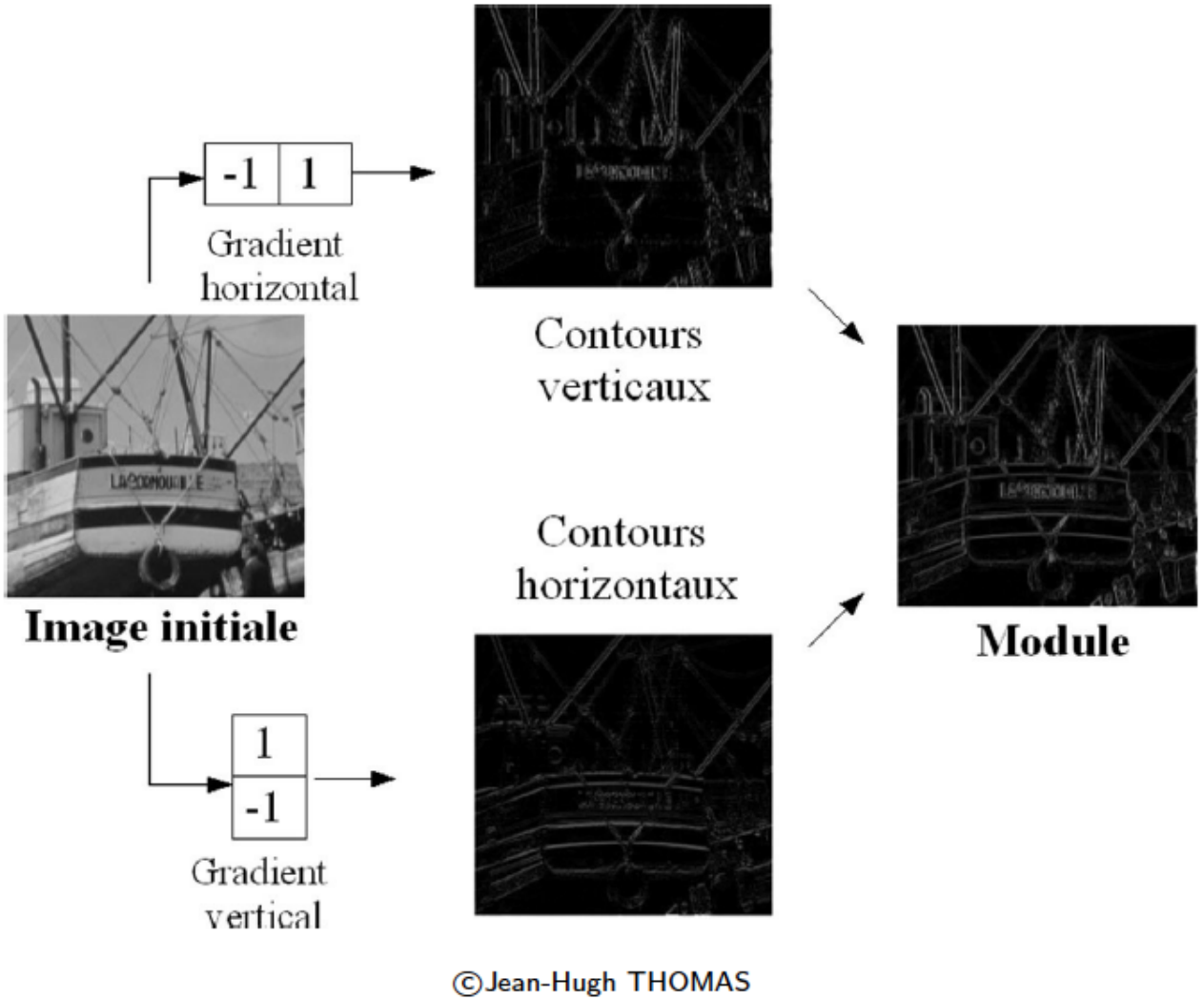
-1
1

- Or :

-1	0	1

-1
0
1

Image Gradient : Implementation - Discrete Derivative



Roberts Filter

Principle

- The first approximation of the first derivative of an image was proposed by Roberts in 1965.
- Gradients at $\frac{\pi}{4}$:
  - $\nabla_1 I \approx I(x+1, y+1) - I(x, y)$
  - $\nabla_2 I \approx I(x, y+1) - I(x+1, y)$
- Use of two convolution masks for the two gradient directions

1	0
0	-1

0	1
-1	0

- High sensitivity to noise due to the size of the masks.
  - The derivative filter is a high-pass filter.
  - Noise = high frequencies (sharp variations).

Other Filters for Edge Detection

Smoothing then differentiating the image

Prewitt Filter

Aaveraging filter + derivative

-1	-1	-1	-1	0	1
0	0	0	-1	0	1
1	1	1	-1	0	1

Sobel Filter (1972)

Gaussian filter + derivative

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

Edge detection is less sensitive to noise.

Edge Detection : Example

Roberts

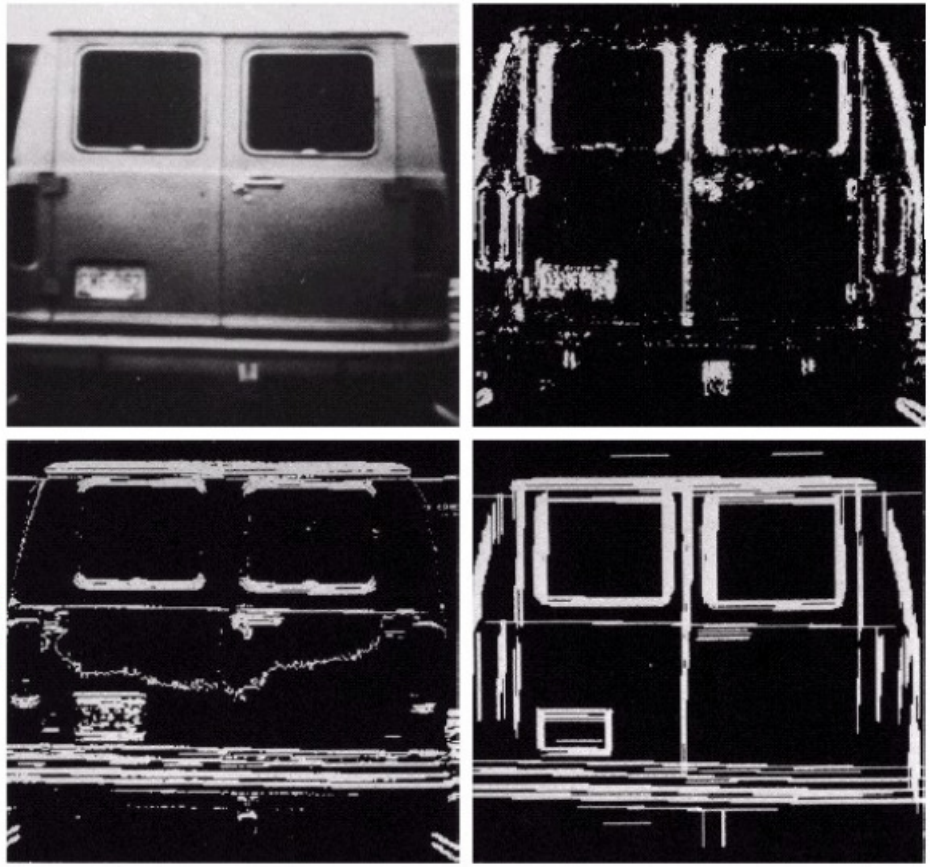
Prewitt

Sobel

Image Gradient : Example

a b
c d

FIGURE 10.16
(a) Input image.
(b) G_y component of the gradient.
(c) G_x component of the gradient.
(d) Result of edge linking. (Courtesy of Perceptics Corporation.)



$\frac{\partial f}{\partial x}$ Contours verticaux

$\frac{\partial f}{\partial y}$ Contours horizontaux

$\frac{\partial f}{\partial y} + \frac{\partial f}{\partial x}$ Norme

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

Edge Detection : Thresholding

Influence of thresholding on detection

Sans seuillage

Avec S = 25

Avec S = 60

Edge Detection : Thresholding

The threshold can be determined from the (cumulative) histogram of the gradient magnitude. Advantage : this threshold is *adapted* to the content of the image.

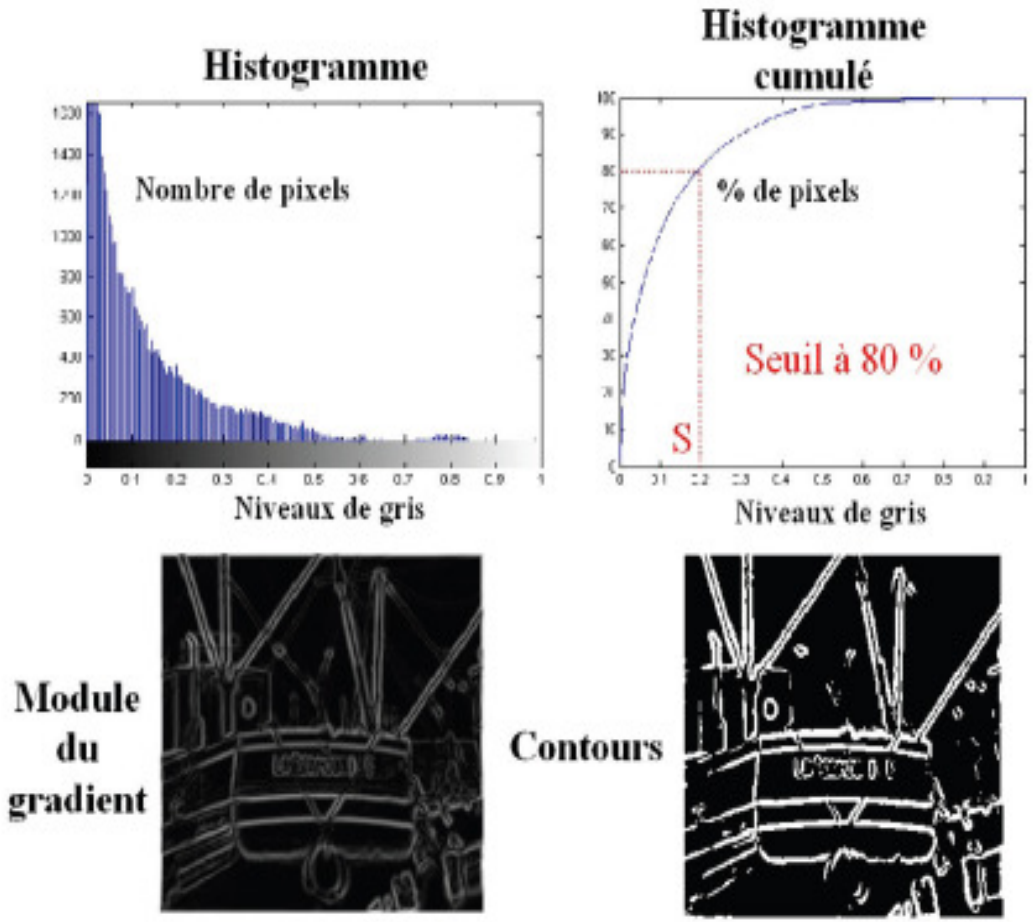


FIGURE – From E. Arnaud

Image Gradient : Edge Detection

Post-Processing

1. Elimination of non-local maxima of the gradient magnitude in its direction.
2. Thresholding by hysteresis.

Image Gradient : Edge Detection

Extraction of Local Extrema in the Gradient Direction

- For a given pixel p, determination of gradient values along the line passing through p in the gradient direction.
- Verify that the gradient at p is locally a maximum along this line.

Image Gradient : Edge Detection

Extraction of Local Extrema in the Gradient Direction

- We search for local maxima in the gradient direction.
- P is a local maximum $\Rightarrow ||G(\vec{P})|| > ||G(\vec{P}_1)||$ and $||G(\vec{P})|| > ||G(\vec{P}_2)||$

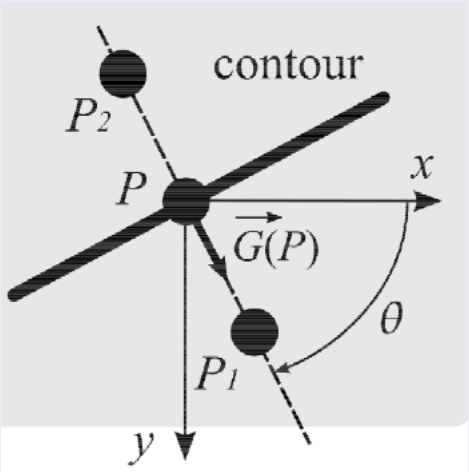
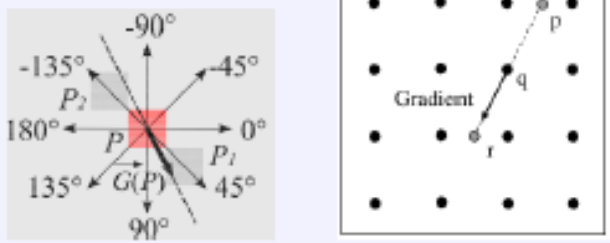


Image Gradient : Edge Detection

Extraction of Local Extrema in the Gradient Direction : In Practice

Non-maxima suppression

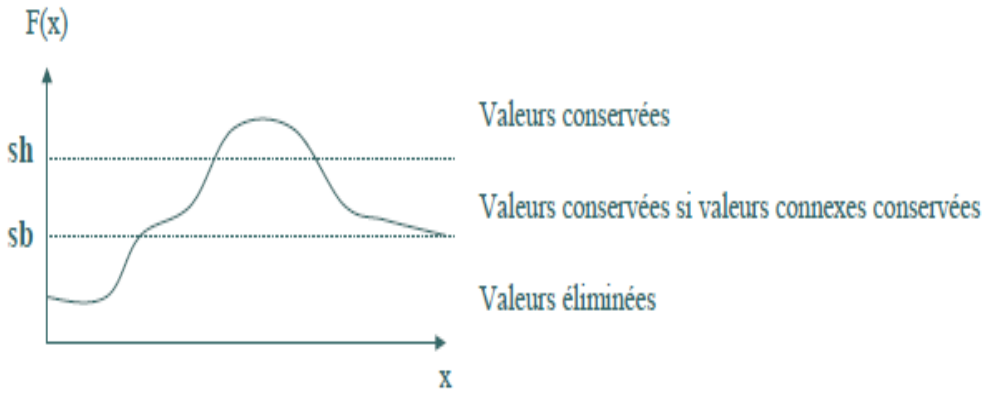
- Discretization of the direction
  - The direction is rounded to $\frac{\Pi}{4}$ (8 neighbors).
  - The two neighboring pixels in the gradient direction are found.
- Interpolation
  - The subpixel values for the two neighbors are calculated.
  - The gradient magnitude value is interpolated at these points.



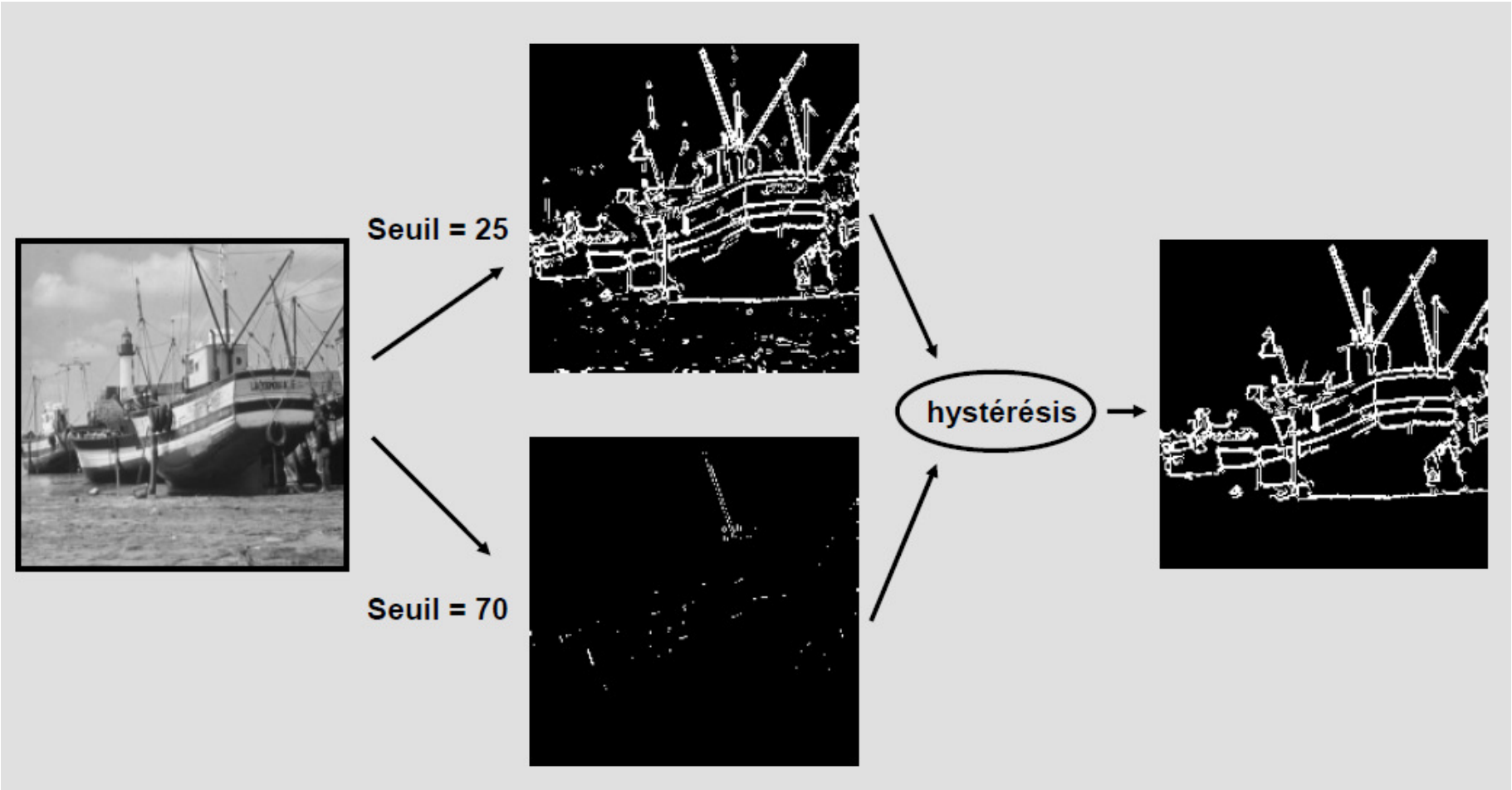
Hysteresis Thresholding

Idea

- Keep the strongest contours while trying to ensure their continuity.
- Use a low threshold (s_b) and a high threshold (s_h).
- The low threshold highlights weaker contours in the image.
- These weaker contours are kept only if they are near strong contours, i.e. if a given pixel is connected by a path of pixels with a gradient magnitude greater than s_b, to a pixel with a gradient magnitude greater than s_h.



Hysteresis Thresholding



Edge Detection : Laplacian

Principle

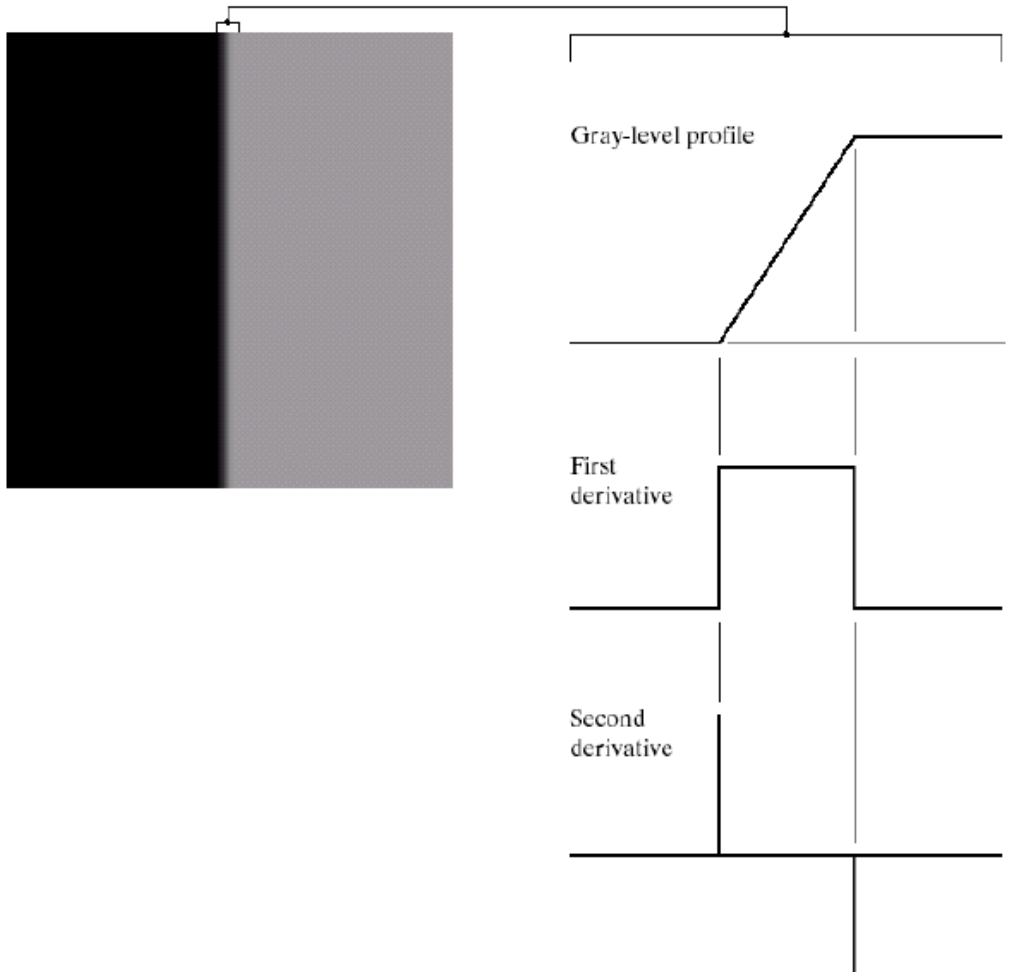
- Use of the second derivative of the image.
- The contours correspond to zero-crossings of the second derivative.
- Laplacian operator :

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

Edge Detection : Laplacian

a b

FIGURE 10.6
(a) Two regions separated by a vertical edge. (b) Detail near the edge, showing a gray-level profile, and the first and second derivatives of the profile.



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

Edge Detection : Laplacian

Laplacian Approximations by Finite Differences

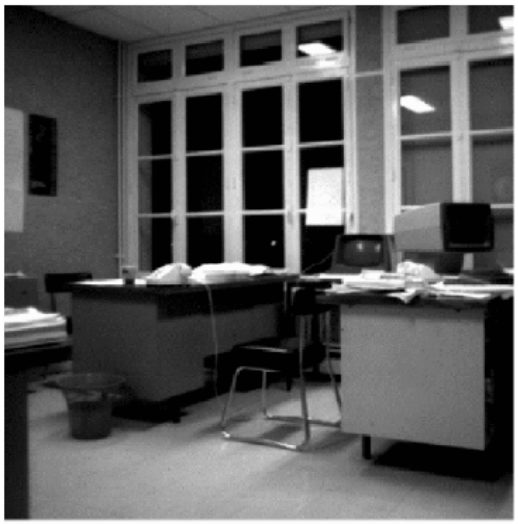
- Demonstration : Taylor series at $I(x \pm h, y \pm h)$
- $\Delta I \approx I(x+1,y) + I(x-1,y) + I(x,y+1) + I(x,y-1) - 4I(x,y)$
- $\Delta I \approx I(x+1,y) + I(x-1,y) + I(x,y+1) + I(x,y-1) + I(x+1,y+1) + I(x-1,y-1) + I(x+1,y-1) + I(x-1,y+1) - 8I(x,y)$
- Several discrete approximations of the Laplacian :

0	1	0	1	1	1
1	-4	1	1	-8	1
0	1	0	1	1	1

Conclusion

- The image gradient is a powerful tool for detecting edges and analyzing image features.
- Several approaches, such as gradient-based methods, use finite differences to approximate derivatives and detect local extrema.
- Post-processing steps like non-maximum suppression and hysteresis thresholding refine edge detection.
- The Laplacian operator offers an alternative method for edge detection by leveraging second-order derivatives.

Détection de contours : Laplacien



Edge Detection : Laplacian

Sensitive to noise $\Rightarrow$ Necessity to smooth by low-pass filtering.

Low-pass smoothing

- Convolution of the image with a Laplacian mask : $I * g * \Delta$
- Case of a Gaussian mask : $g(x, y, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}}$

$$I * g * \Delta = I * (\Delta * g) = I * \Delta g$$

Linear differential operators commute with convolution :

$$\Delta * g = \Delta g = \frac{4}{\sqrt{2\pi}\sigma}(\frac{x^2+y^2}{2\sigma^2} - 1)e^{-\frac{x^2+y^2}{2\sigma^2}}$$

- We directly convolve I with the mask defined by Δg : **Laplacian of Gaussian (LoG)**

Edge Detection : Laplacian

Laplacian of Gaussian

- LoG : filter for the human vision system.
- LoG can be approximated by DoG = Difference of Gaussians.

$$DoG(x, y, \sigma_1, \sigma_2) = g(x, y, \sigma_1) - g(x, y, \sigma_2)$$

Plan

Edge Detection : Optimal Approaches

Optimal Filtering

- In 1D : modeling the edge as a step function.
- With noise : $C(x) = A\Theta(x) + n(x)$, where n is white Gaussian noise.
- We seek the derivative filter h such that :

$$h * C = \Theta$$

Edge Detection : Optimal Approaches

In the one-dimensional case, we assume that detection is performed by convolving the signal with a filter with impulse response h. The edges are the extrema of the filter's output (step-like edges, white noise).

Canny's Criteria (1983)

Criteria for determining an optimal filter :

- Detection : the operator must guarantee good detection, i.e., a strong response even for weak edges. (criterion Σ)
- Localization : the edge must be localized accurately (criterion Λ).
- Single response : an edge must produce a single response from the operator.

We seek the derivative filter h that maximizes Σ and Λ under the constraint of uniqueness of the response.

Edge Detection : Optimal Approaches

Principle

- Expression of the criteria through the joint optimization of 3 functionals to define the optimal linear filter for detecting an ideal edge (step-like).
- Assumption of additive noise independent of the signal.
- Differential equation whose solution is of the form (finite impulse response filter) : $h(x) = a_1 e^{\alpha x} cos(\omega x) + a_2 e^{\alpha x} sin(\omega x) + a_3 e^{-\alpha x} cos(\omega x) + a_4 e^{-\alpha x} sin(\omega x)$

Edge Detection : Optimal Approaches

Solution of the differential equation to optimize Canny's criteria :

Different Approaches

- Gaussian filter and derivative (1980, approximate solution).
- Shen-Castan filter (1986, exact solution).
- Deriche filter (1987, exact solution).

Edge Detection : Optimal Approaches

Gaussian Filter

- Gaussian smoothing filter whose derivative provides an approximation of the optimal filter solution (Canny).
- Gaussian smoothing filter has the impulse response : $h(x) = c \exp^{-x^2/2\sigma^2}$
- Derivative : $h'(x) = -c\frac{x}{\sigma^2} \exp^{-x^2/2\sigma^2}$ = solution of the optimal filter (Canny).
- In 2D : action of 2 crossed filters :

$$f_x(x,y) = -x \exp^{\frac{-x^2}{2\sigma^2}} \exp^{\frac{-y^2}{2\sigma^2}}$$

$$f_y(x,y) = -y \exp^{\frac{-x^2}{2\sigma^2}} \exp^{\frac{-y^2}{2\sigma^2}}$$

J. Canny, A Computational Approach To Edge Detection, IEEE Trans. Pattern Analysis and Machine Intelligence, 8 :679-714, 1986.
`http://www.limsi.fr/Individu/vezien/PAPIERS_ACS/canny1986.pdf`

Edge Detection : Optimal Approaches

Canny Detector

- Filter the image with the derivatives of the Gaussian.
- Find the magnitude and orientation of the gradient.
- Non-maximum suppression (sliding window).
- Assembly and thresholding :
  - Define two thresholds : low and high.
  - The high threshold is used to start the edge and the low threshold to continue it.

Edge Detection : Optimal Approaches

Canny Detector

Effect of σ.

FIGURE – Source S. Seitz

- Large σ allows for the detection of thick edges.
- Small σ allows for the detection of fine edges.

Edge Detection : Optimal Approaches

Shen Castan Filter

- Optimization of a criterion that includes detection and localization.
- Solution of the form $h(x) = ce^{-\alpha|x|}$
- c is chosen to normalize the filter $c = \frac{1-e^{-\alpha}}{1+e^{-\alpha}}$
- α determines the width of the filter, the smaller α is, the more smoothing is applied.
- Derivative filter :

$$h'(x) = \begin{cases} de^{-\alpha|x|} & \text{if } x \geq 0 \\ -de^{-\alpha|x|} & \text{otherwise} \end{cases}$$

with $d = 1 - e^{-\alpha}$ chosen to normalize the filter.

Edge Detection : Optimal Approaches

Deriche Filter

- Smoothing filter whose derivative is the exact solution to the Canny equation extended to filters with infinite support.
- $h(x) = k(\alpha |x| + 1)e^{-\alpha|x|}$
- with $k = \frac{(1-e^{-\alpha})^2}{1+2\alpha e^{-\alpha} - e^{-2\alpha}}$
- $h'(x) = -k'xe^{-\alpha|x|}$ with $k' = \frac{(1-e^{-\alpha})^2}{e^{-\alpha}}$
- Directional (anisotropic) filters.

Edge Detection : Optimal Approaches

Deriche Filter : Example

Figure 1: (a) : image originale, (b) : gradient en x (filtre de Deriche), (c) gradient en y (filtre de Deriche), (d) extréma locaux de la norme du gradient dans la direction du gradient.

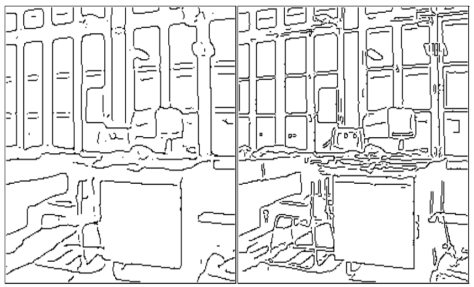


Figure 2: Seuillages des extrema locaux pour différentes valeurs du paramètre α du filtre de Deriche : (a) $\alpha = 0.5$, (b) $\alpha = 1.5$.

- Influence of α : scale factor.
- Low α : low robustness to noise, good localization.
- High α : good robustness to noise, poor localization.
- Adjust α according to the signal-to-noise ratio of the image.

Edge Detection

A first step only for segmentation.

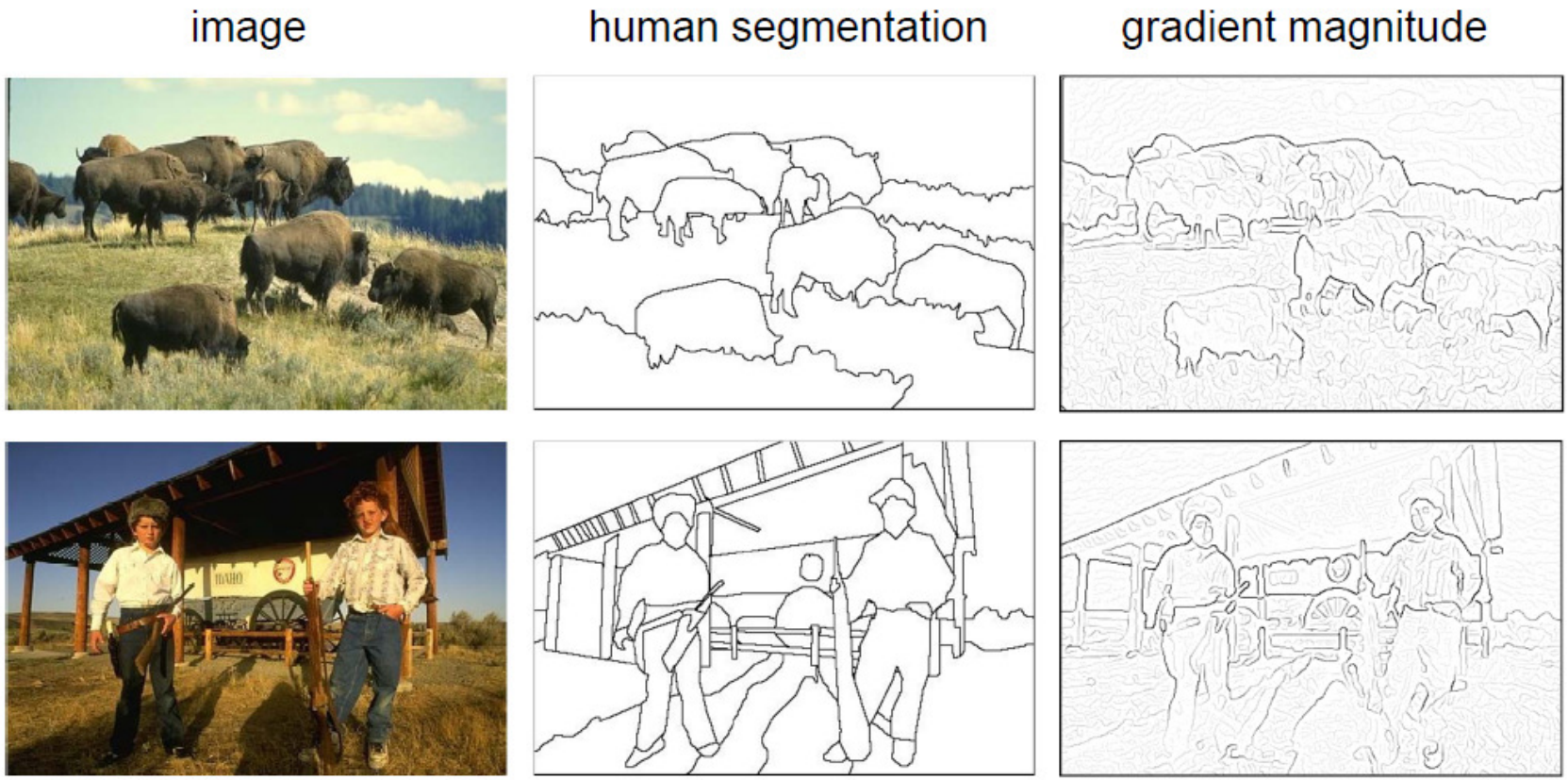


FIGURE – Source Fei Fei Li

Plan

Edge Detection : Hough Transform

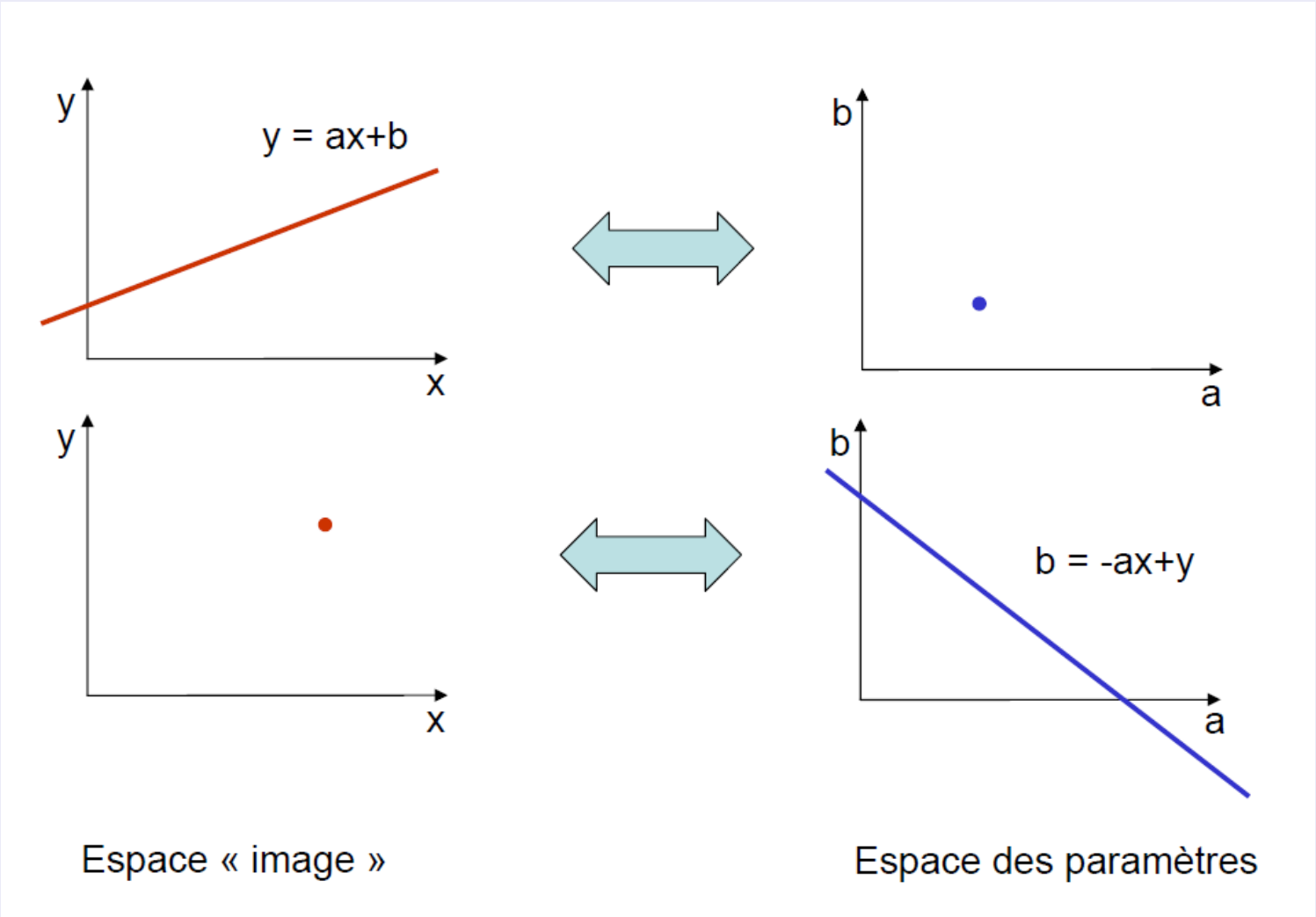
For example, for detecting lines :

Principle

- Shape recognition technique invented in 1962 by Paul Hough.
- Global approach to detect continuous edges, i.e., not just contour pixels but the complete contour.
- Transition from the x-y plane to the parametric a-b plane and voting procedure.

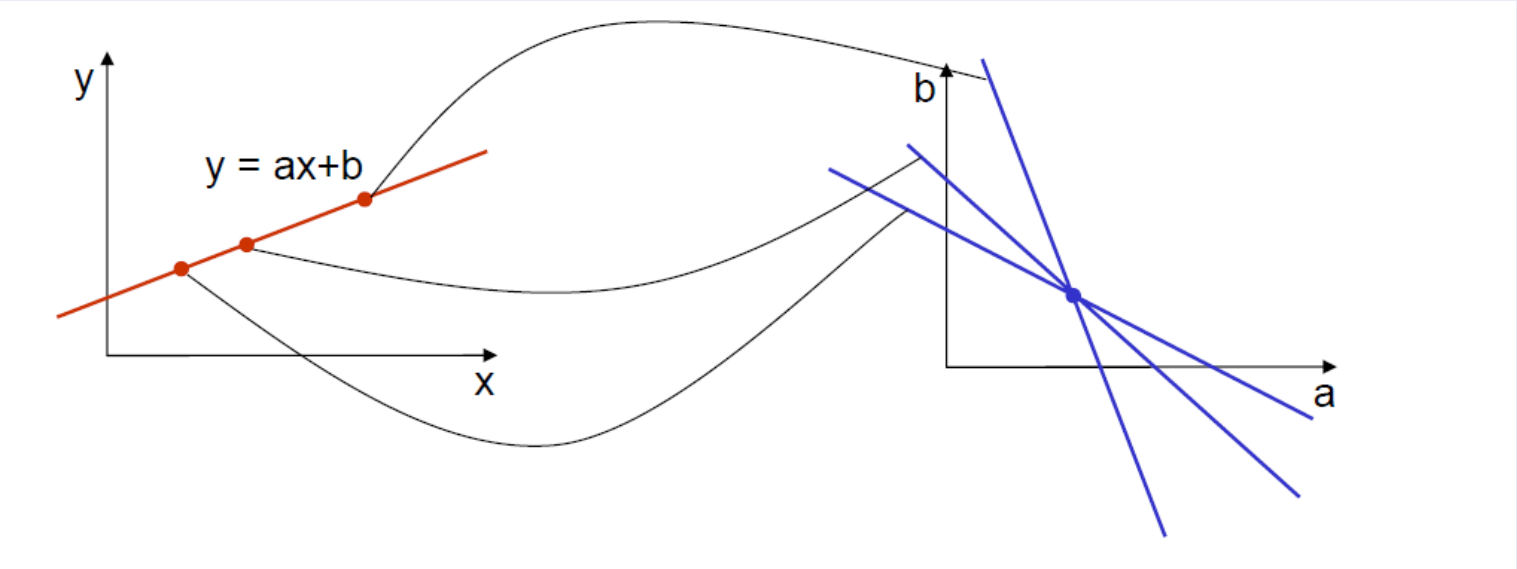
Edge Detection : Hough Transform

Principle



Edge Detection : Hough Transform

Principle



The images of all the points of a line intersect at (a, b) in the Hough domain.

Edge Detection : Hough Transform

Principle for Line Detection

1. Apply an edge detection.
2. Discretize the parameter space (a,b).
3. Initialize an accumulator.
4. For each edge point :
   - Determine its corresponding line in the parameter space.
   - Increment the accumulator at points along this line.
     - Search for maxima : parameters (a,b) of the line. The intersection points of lines in the a-b plane indicate the actual lines present in the x-y plane.

Edge Detection : Hough Transform

- Problem : The space (a, b) is not suitable (unbounded as $a \to \infty, b \to \infty$).
- Solution : Representation in polar form (ρ, θ), $\rho = x \cos\theta + y \sin\theta$.

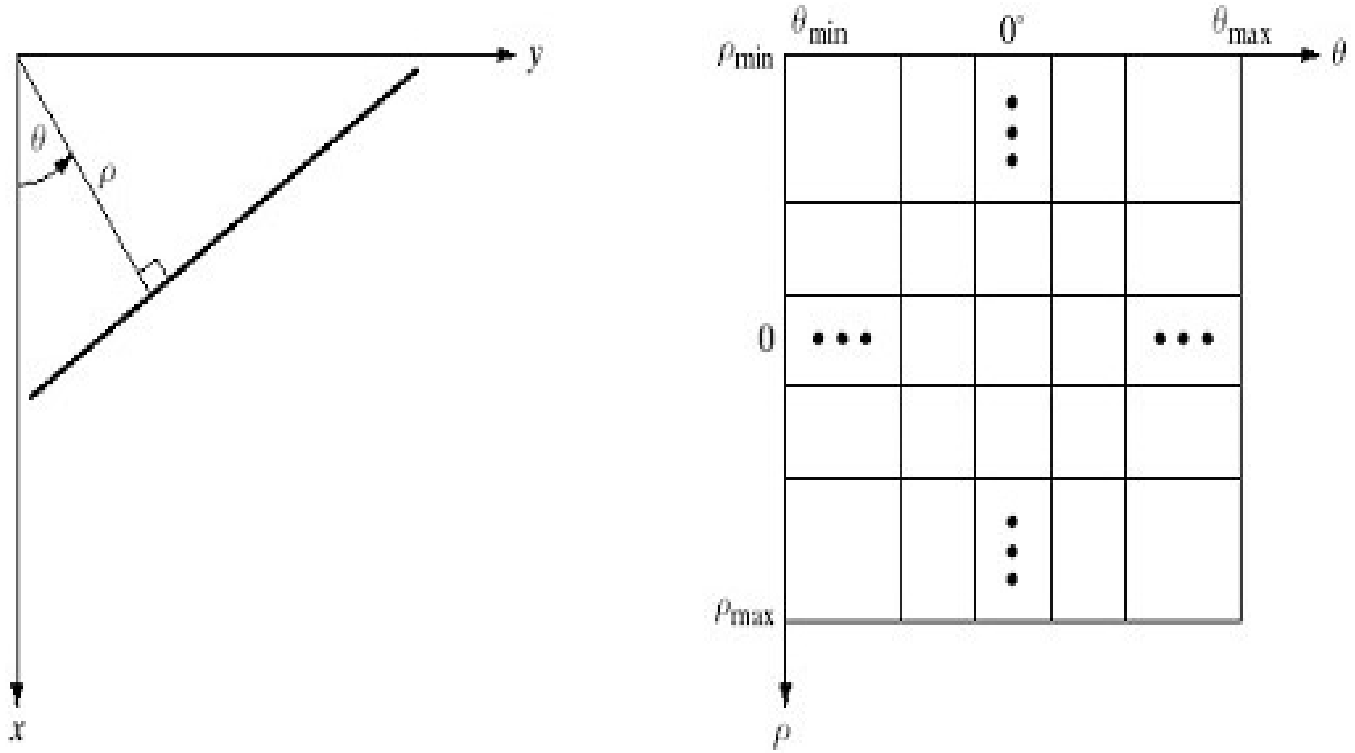


FIGURE – Source : A. Boucher

Edge Detection : Hough Transform

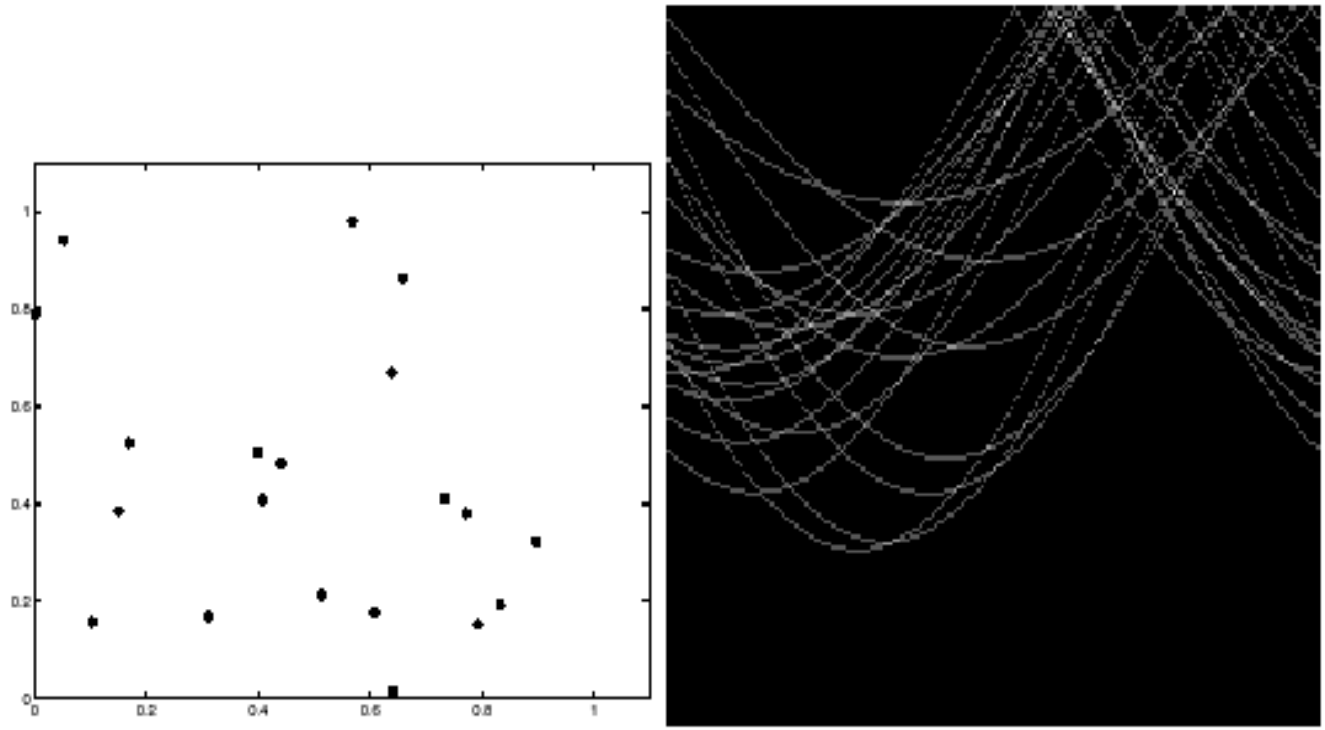


FIGURE – Source : A. Boucher

The transform of random points yields no precise result.

Edge Detection : Hough Transform

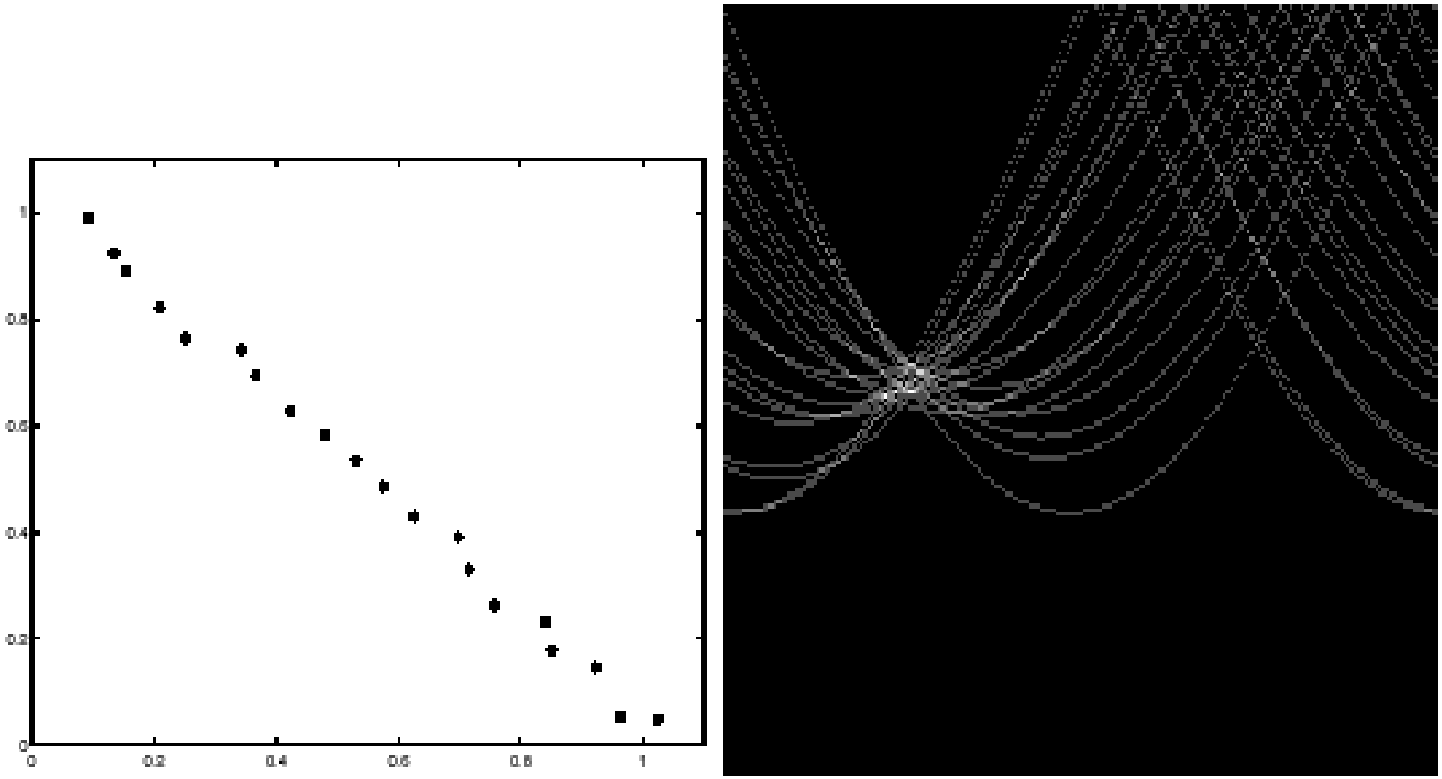


FIGURE – Source : A. Boucher

The transform of aligned points allows for retrieving the line.

Edge Detection : Hough Transform

Example

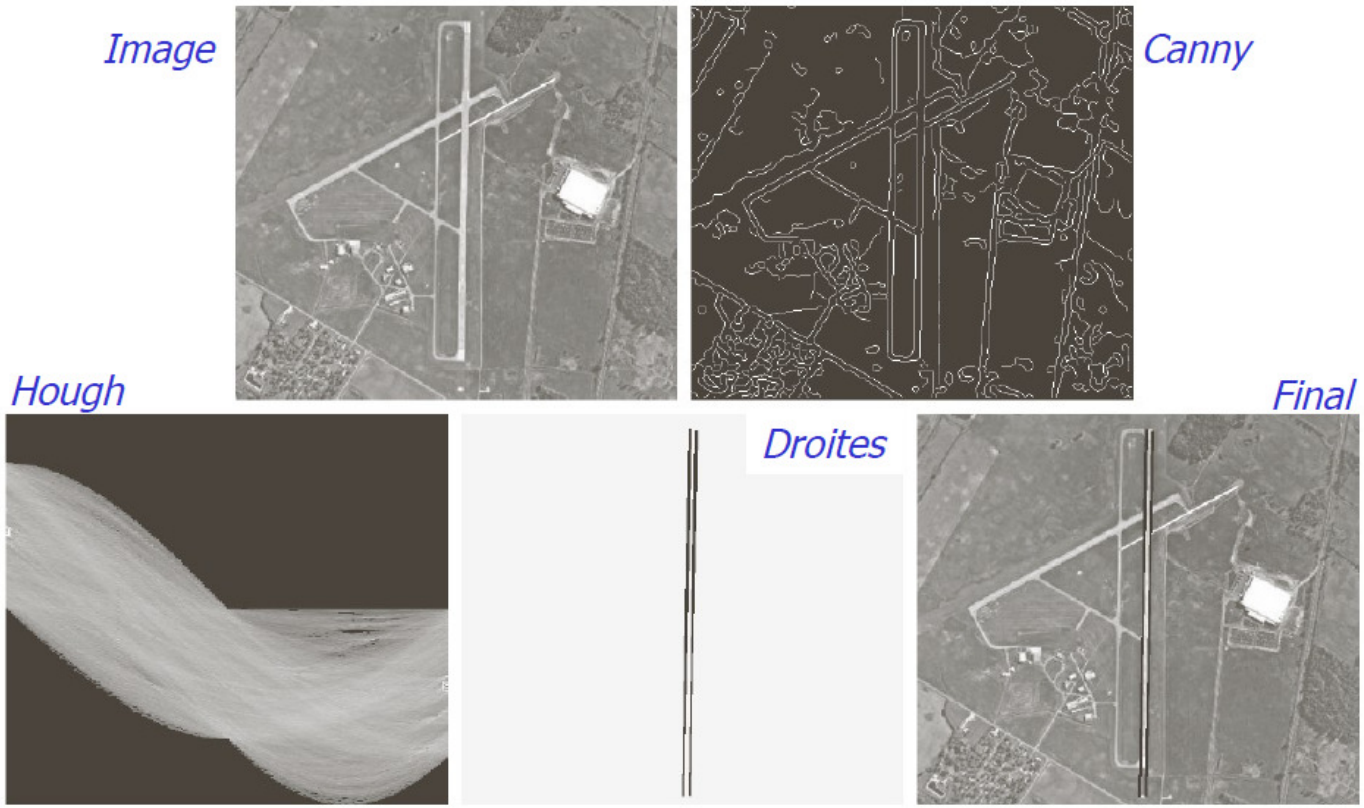
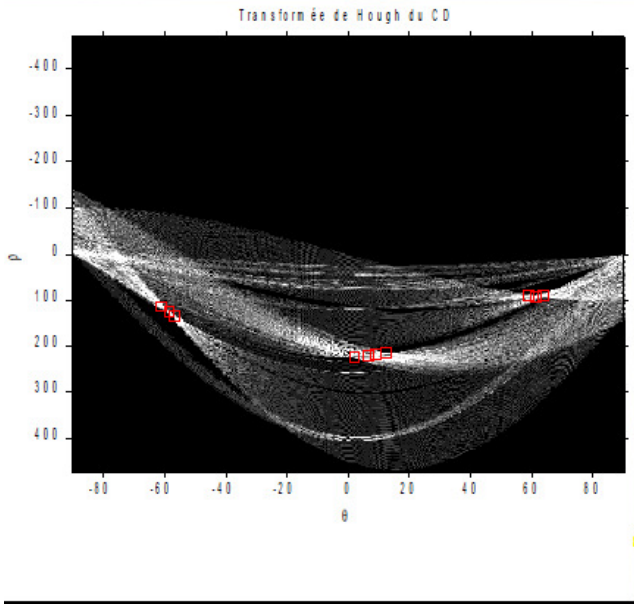
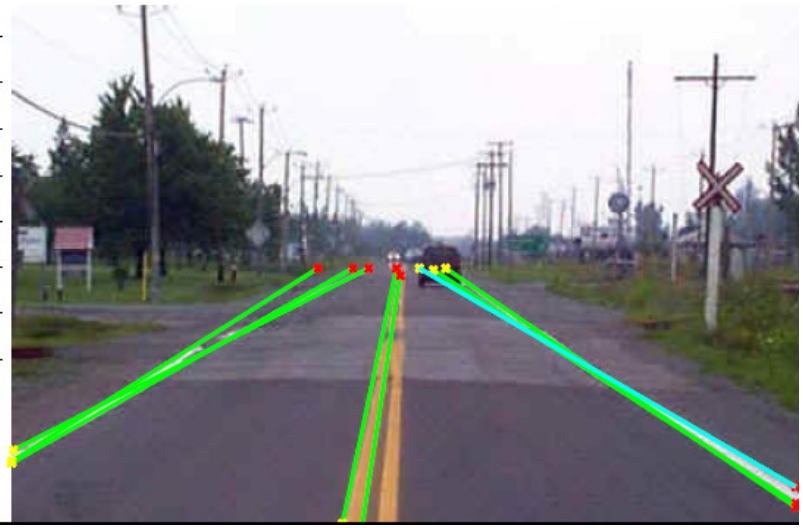


FIGURE – Source : Gonzales & Woods

Edge Detection : Hough Transform

Example





In Summary

- The histogram, a fundamental tool for many pre-processing tasks.
- Many pre-processing tasks are also modeled as filtering operations.
- Edge detection :
  - Search for discontinuities in the image.
  - Gradient and Laplacian of an image.
  - Optimal approaches.