Introduction to Image Edge Detection, Classification

Elsa Angelini PhD

TELECOM ParisTech
Dpt. IDS

Some slides from the IMAGE group @Telecom – Tupin et al

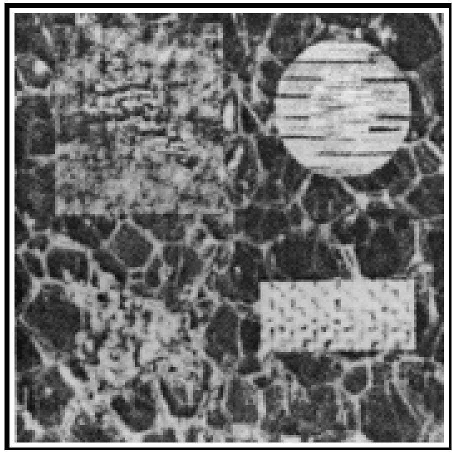
Outline

- Introduction
- Edge Detection
- Classification

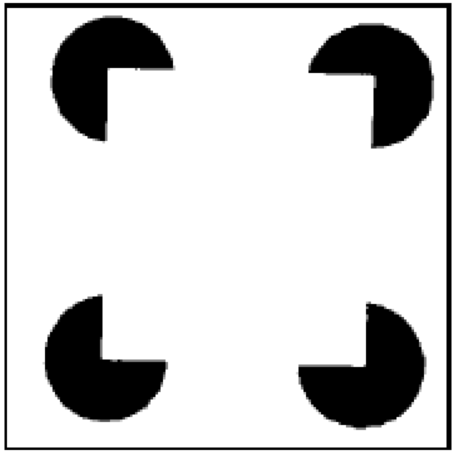
Introduction

- What is an object in an image?

Edges / Regions



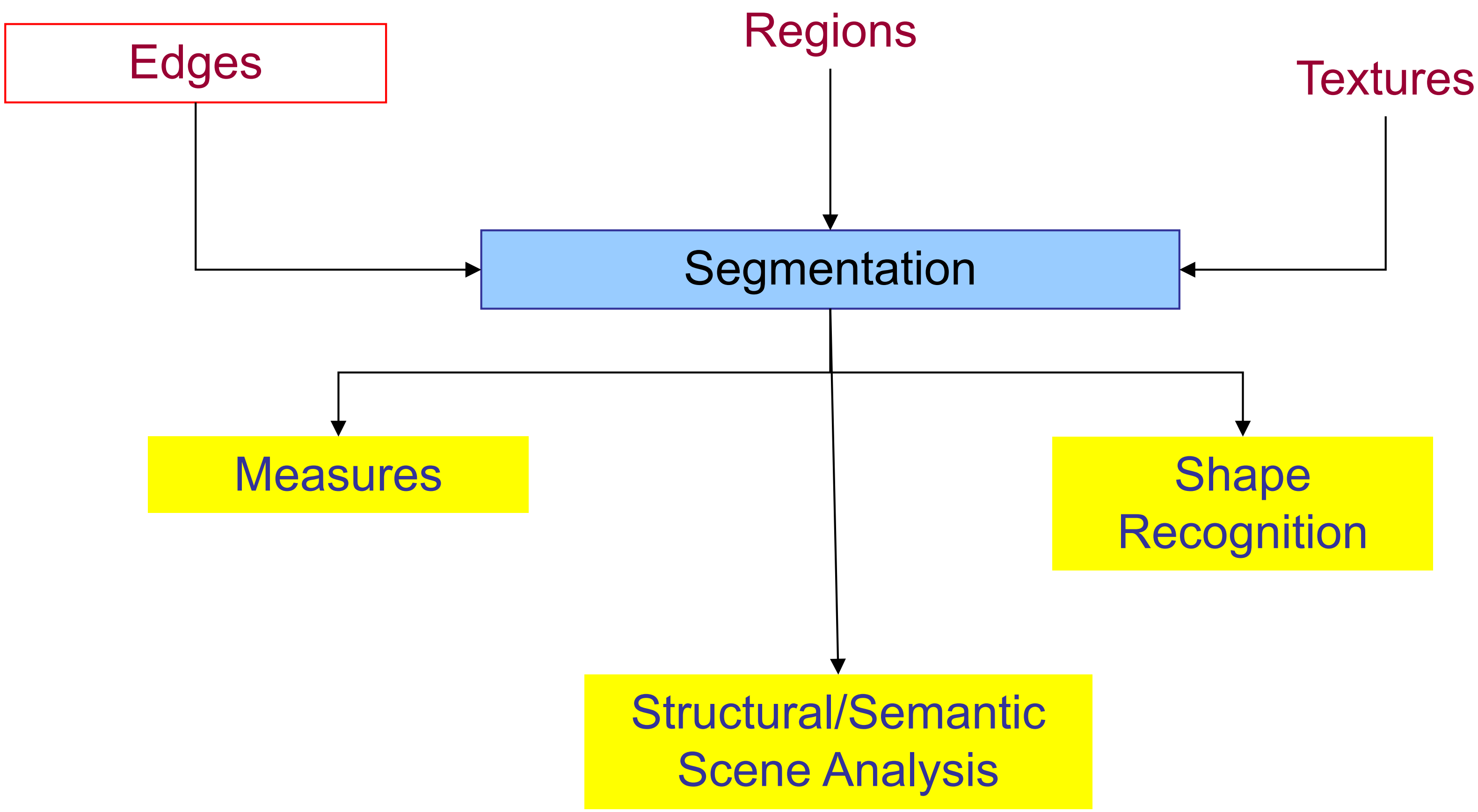
Texture / Regions



➔ Active contours

Introduction

- The image segmentation Problem

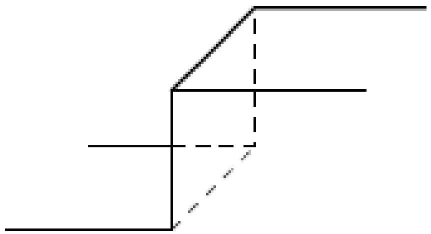


Introduction

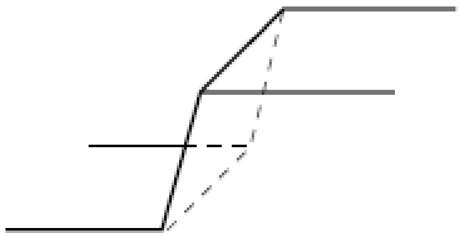
- The segmentation problem:
  - Partition an image into objects:
  - 2 approaches:
    - Region-based
    - Contour-based

Introduction

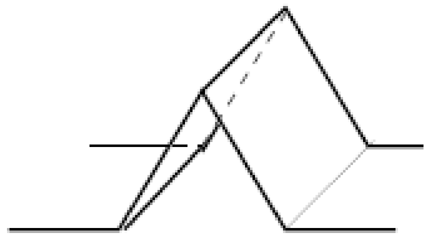
- Types of Contour profiles



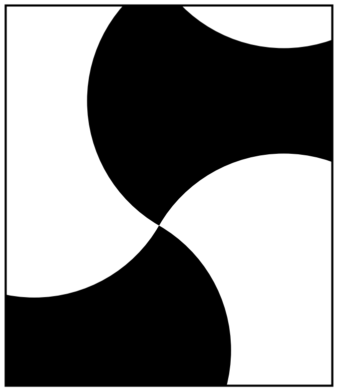
Staircase

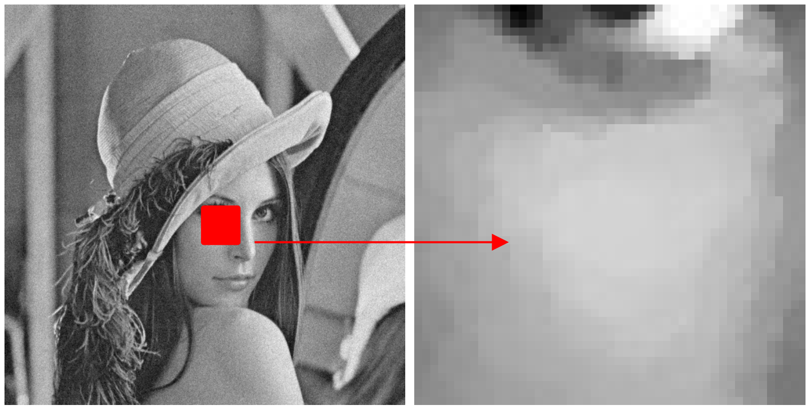


Ramp



Roof



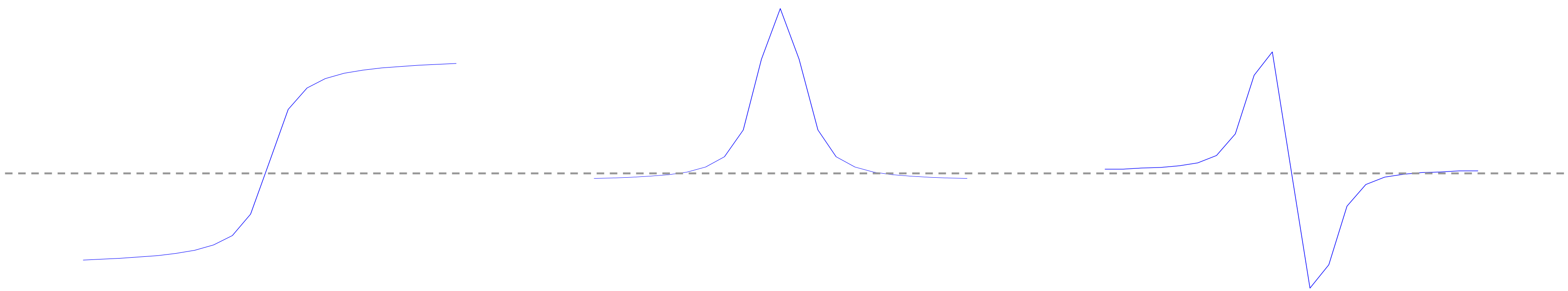


Outline

- Introduction
- Edge Detection
- Classification

Edge Detection

- **Gradient-based** Detection of « abrupt changes » in image gradient. Analysis of first and second derivatives of image gradients.



Edge　　1st derivative　　2nd derivative

Edges: location of gradient maxima, in the **direction** of the gradient.

Edge Detection

- Gradient-based

Image $I(x, y)$ with a continuous representation:

$$\vec{G} = \vec{\nabla} I(x, y) = \begin{bmatrix} \dfrac{\partial I(x,y)}{\partial x} \\ \dfrac{\partial I(x,y)}{\partial y} \end{bmatrix}$$

$$G = \left\| \vec{\nabla} I(x, y) \right\| = \sqrt{\left(\frac{\partial I(x,y)}{\partial x} \right)^2 + \left(\frac{\partial I(x,y)}{\partial y} \right)^2}$$

Edge Detection

- Gradient-based
  - Dedicated Gradient Filters

1. **Pre-processing**: filtering (Gaussian, Median).
2. **Segmentation** via thresholding or local maxima detection.
3. **Post-processing**: contour closing, curve fitting, smoothing.

Edge Detection

- Gradient-based Filters

Gradient

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

Roberts

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

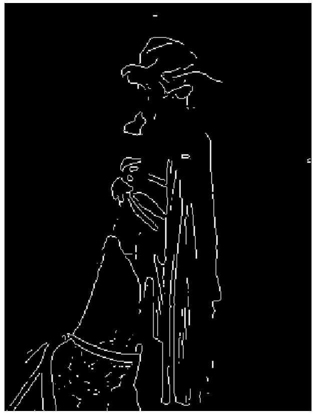
Prewitt

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

Sobel

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Edge map

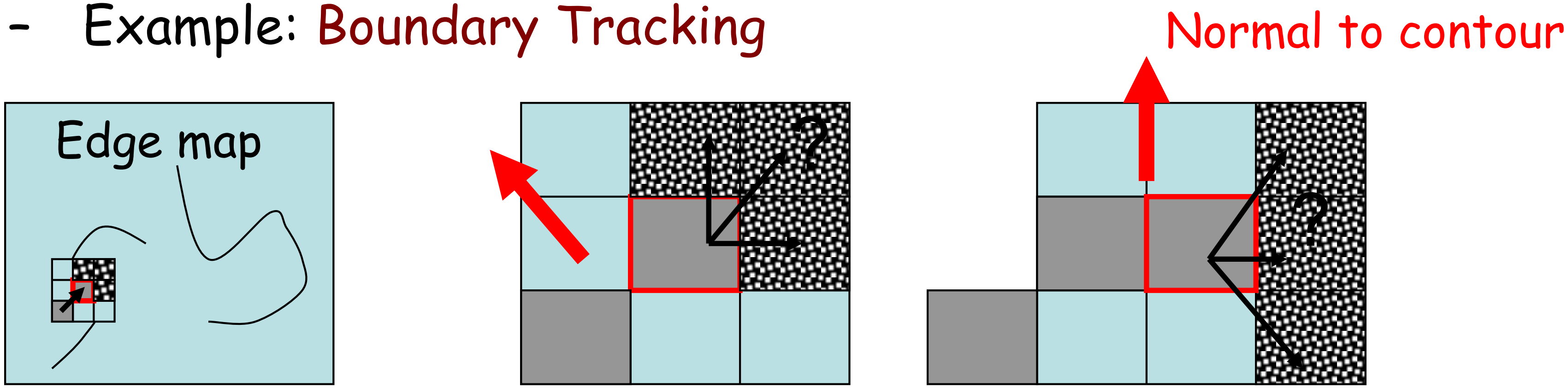


Edge maps

With oriented edges.

Edge Detection

- Post processing of Edge maps
  - Example: Boundary Tracking



Boundary tracking is very sensitive to noise ⇒

-Use of smoothing

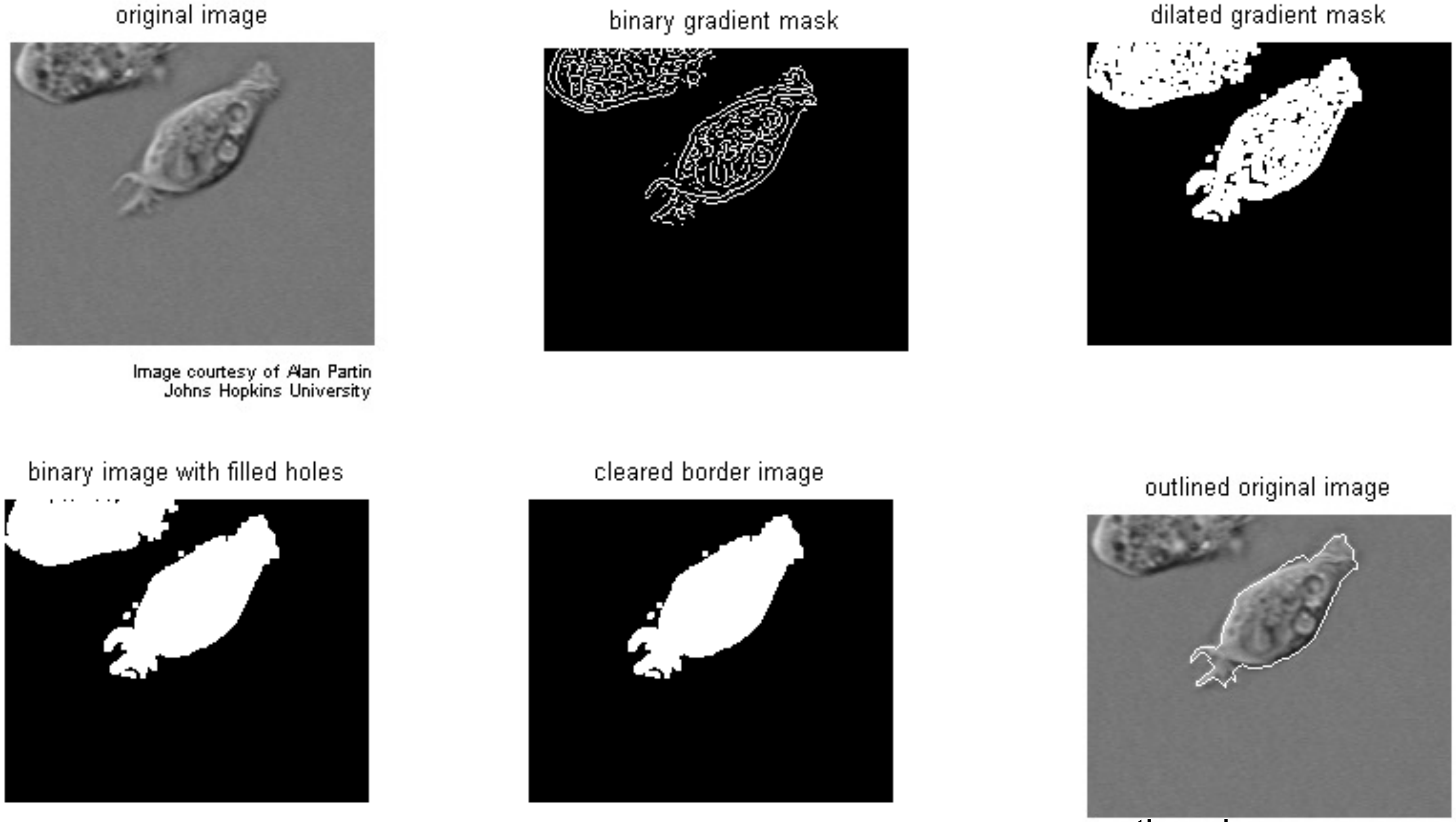
-Average gradient computation

-Use large « tracking » neighborhoods.

Edge Detection

Post processing of Edge maps

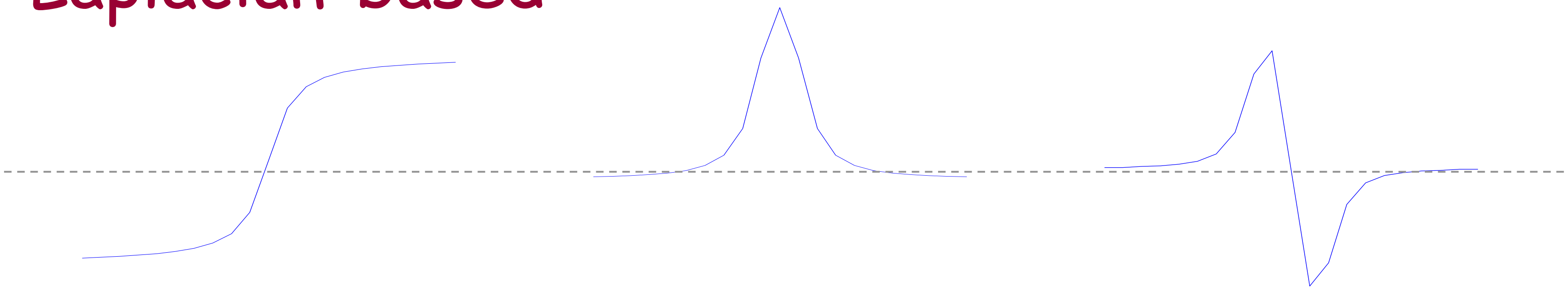
- Example: morphological post-processing



www.mathworks.com

Edge Detection

- Laplacian-based



Edge　　　　1st derivative　　　　2nd derivative

Zero Crossing

$$\Delta I(x,y) = \frac{\partial^2 I(x,y)}{\partial x^2} + \frac{\partial^2 I(x,y)}{\partial y^2}$$

Edge Detection

- Laplacian-based

  Laplacian operator on the image: Discrete implementation with convolution kernels:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \qquad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

-Set of closed connected contours…but…

Very sensitive to noise!

2 convolution kernels

Edge Detection

- Laplacian-based: Laplacian of Gaussian (LoG)

$$LoG\left(I(x,y)\right)=\frac{\partial^2\left(I\left(x,y\right)*G_\sigma\left(x,y\right)\right)}{\partial x^2}+\frac{\partial^2\left(I\left(x,y\right)*G_\sigma\left(x,y\right)\right)}{\partial y^2}$$

$$=I\left(x,y\right)*\frac{\partial^2\left(G_\sigma\left(x,y\right)\right)}{\partial x^2}+I\left(x,y\right)*\frac{\partial^2\left(G_\sigma\left(x,y\right)\right)}{\partial y^2}$$

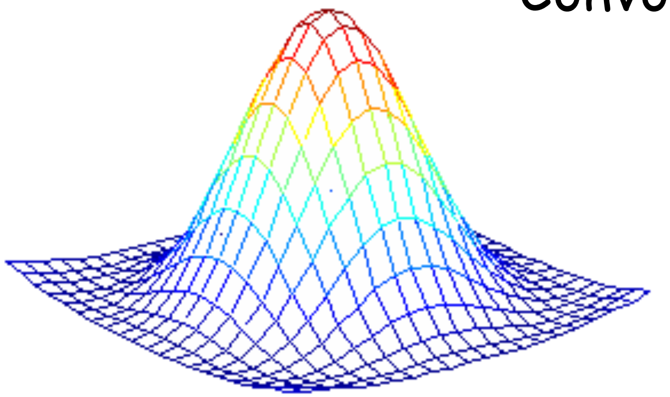
Convolution kernel?

Edge Detection

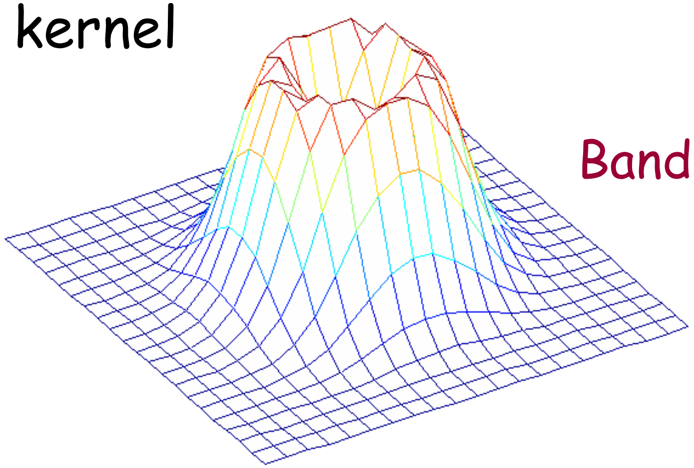
- Laplacian-based: Laplacian of Gaussian (LoG)

$$\Delta\left(G_\sigma\right) = \Delta\left(\frac{1}{2\pi\sigma^2} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}\right) = \frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}$$

Convolution kernel



Impulse response



Band-pass

Transfer function

Edge Detection

- Laplacian-based
  - Parameter σ controls the width of central peak = amount of smoothing.

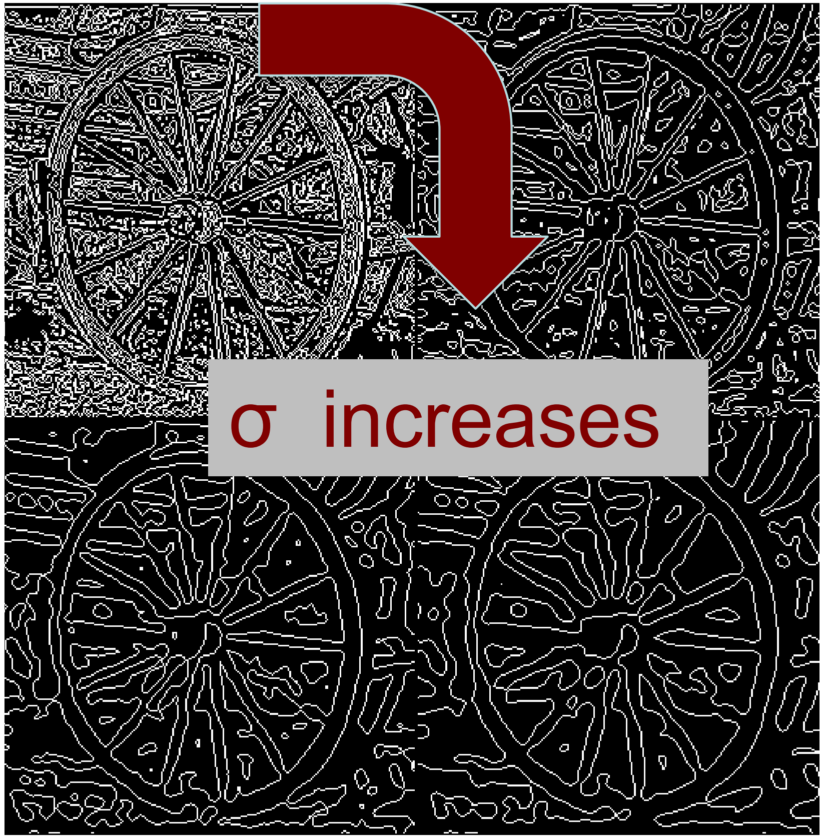


Illustration from Jain - Book

Edge Detection

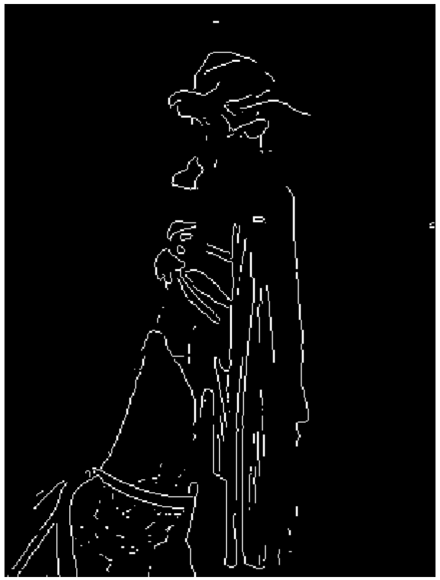
- Laplacian-based

  - Good approximation with a **Difference of Gaussians** (DoG), with a ratio $\sigma_2 / \sigma_1 = 1.6$.
  - DoG separable in x and y :

  => efficient implementation.

Edge Detection

- Laplacian of a Gaussian (LoG)



Gradient-based

LoG

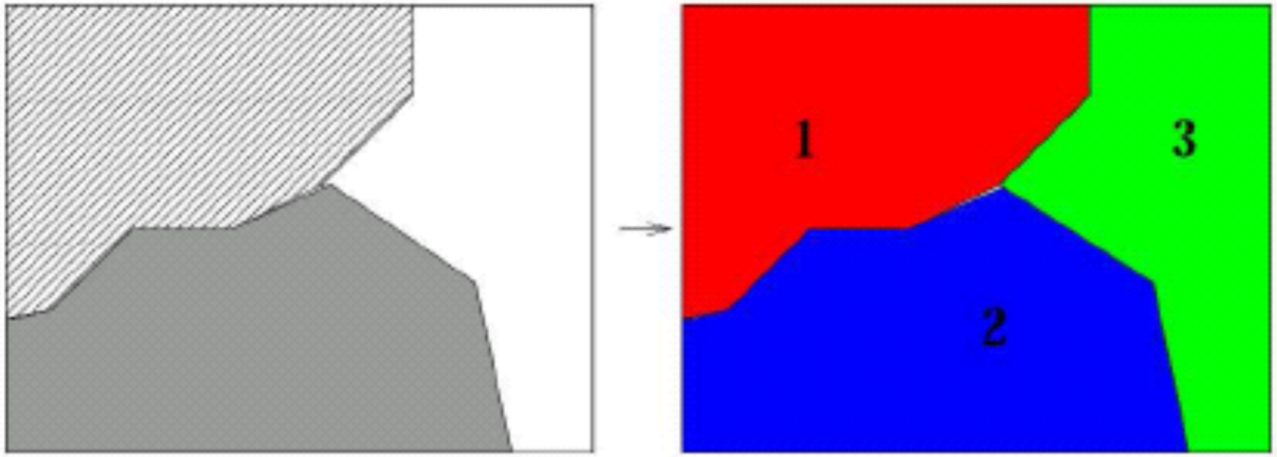
Outline

- Introduction
- Edge Detection
- Classification

Classification

Image modeling

- **Probabilistic model**



S set of sites (pixels = localization (i,j) in the image)

$$\begin{array}{ccc} \text{grey-level } x_s \in \{0,...,255\} = E & \rightarrow & \text{class } l_s \in \{1,...,K\} = \Lambda \\ \downarrow & & \downarrow \\ X_s \text{ random variable of grey-level} & & L_s \text{ random variable of label} \end{array}$$

Classification

Content from F. Tupin

Introduction

◦ **Classification objectives**

- identification of the different classes in the image
- preliminary step of pattern recognition methods (object detection)

◦ **Hypotheses**

- grey-level images
- classes = peaks in the histogram
- low grey-level variations in the same class
- punctual classification : each pixel is classified separatly
- supervised learning : samples of each class are available

◦ **Possible extensions**

- multi-channel images
- contextual classification : markovian framework

Classification

- Bayesian classification

  - **Maximum A Posteriori criterion**

you know a grey-level x_s for pixel s

$\Rightarrow$ take the "best" class l_s knowing x_s

$\Rightarrow$ find the i which maximizes $P(L_s = i | X_s = x_s)\ \forall i \in \Lambda$

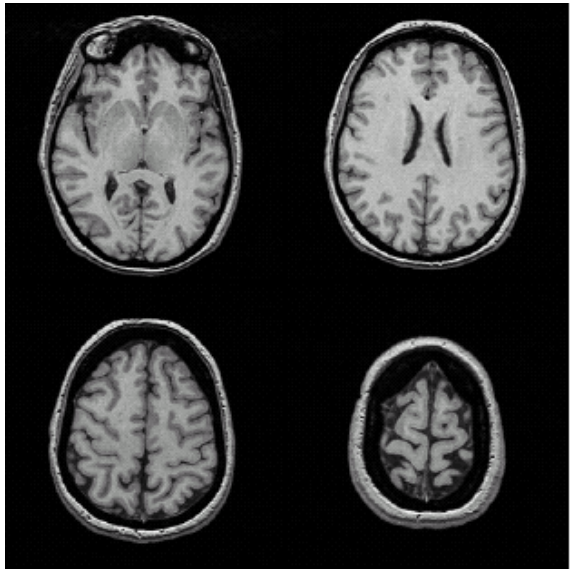
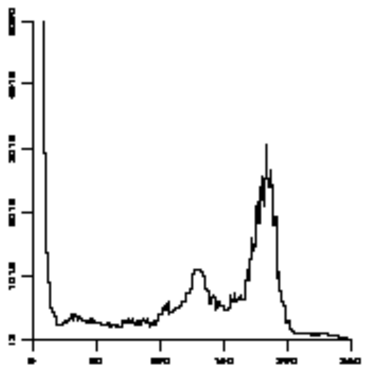
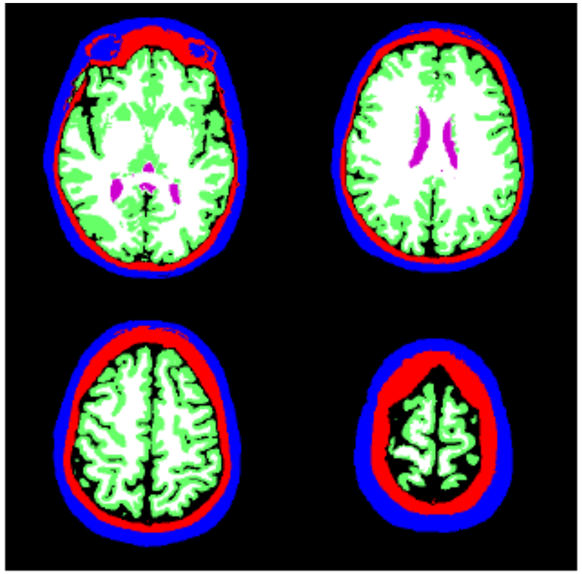
  - **Can we compute $P(L_s = i | X_s = x_s)$?**

Bayes rule

$$P(L_s = i | X_s = x_s) = \frac{P(X_s = x_s | L_s = i) P(L_s = i)}{P(X_s = x_s)}$$

$$l_s = argmax_{i \in \{1,\dots,K\}} P(X_s = x_s | L_s = i) P(L_s = i)$$

Classification

$$P(L_s = i | X_s = x_s) = \frac{P(X_s = x_s | L_s = i) P(L_s = i)}{P(X_s = x_s)}$$

◦ **Example of brain image**

$\Lambda = \{0 = \emptyset; 1 = skin; 2 = bone; 3 = GrayMatter; 4 = WhileMatter; 5 = LCR\}$

- $P(L_s = i)$ = apparition probability of region i
- $P(X_s = x_s | L_s = i)$ = grey-level distribution knowing that the pixels belong to region i

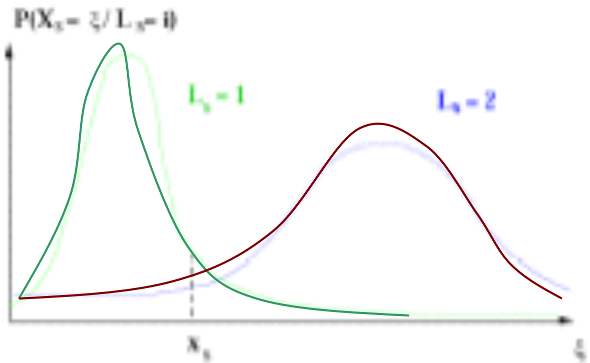
Classification

How can we learn these probabilities ?

- **Learning of $P(L_s = i)$**
  - frequencies of apparition for each class
  - no knowledge : uniform distribution $(P(L_s = i) = \frac{1}{Card(\Lambda)}) \Rightarrow$ Maximum Likelihood criterion

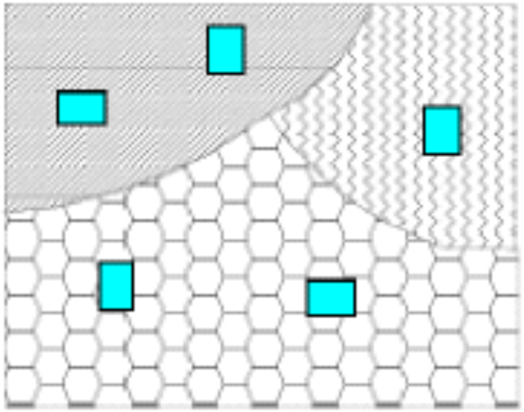
- **Learning of $P(X_s = x_s | L_s = i)$**

grey-level histogram for region i



Classification

- **Supervised learning**
  - samples selection in an image
  - histogram computation
  - histogram filtering



- **Parametric case**

If there exists a parametric model for the grey-level distribution, compute the model parameters !

Ex :

- Gaussian distribution : mean, standard deviation
- Gamma distribution : mean, knowledge of the sensor parameter

Classification

○ Case of a Gaussian distribution

each class $i \in \Lambda$ is characterized by (μ_i, σ_i)

- the conditional probability is :

$$P(X_s = x_s | L_s = i) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp(-\frac{(x_s - \mu_i)^2}{2\sigma_i^2})$$

- if the classes are equiprobable:

$$P(X_s | L_s = i) \quad maximum \Leftrightarrow \frac{(x_s - \mu_i)^2}{2\sigma_i^2} + \ln(\sigma_i) \quad minimum$$

- if the classes are equiprobable and have the same standard deviation (gaussian noise):

$$P(X_s | L_s = i) \quad maximum \Leftrightarrow (x_s - \mu_i)^2 \quad minimum$$

Classification

Contextual / punctual classification

- **Global classification**

$x = \{x_s\}_{s \in S}$ (observed image), $X = \{X_s\}_{s \in S}$ (random field)

$l = \{l_s\}_{s \in S}$ (searched classification), $L = \{L_s\}_{s \in S}$ (random field)

$$P(L = l | X = x) = \frac{P(X = x | L = l) P(L = l)}{P(X = x)}$$

- **Independance assumption for $P(X = x | L = l)$**

$$P(X = x | L = l) = \prod_{s \in S} P(X_s = x_s | L_s = l_s)$$

- **Independance assumption for $P(L = l)$**

$$P(L = l) = \prod_{s \in S} P(L_s = l_s)$$

$\Rightarrow P(L = l | X = x) \propto \prod_{s \in S} P(X_s = x_s | L_s = l_s) P(L_s = l_s) \Rightarrow$ punctual classif.!

Classification

Contextual / punctual classification

◦ **Prior knowledge on $P(L)$**

- independence assumption not verified in practice: images are smooth with strong spatial coherency (image description = smooth areas)
- BUT the coherency is at a local scale ⇒ introduction of contextual knowledge

Markov random fields ⇒ smoothness of the solution, local spatial coherency in the result !

Classification

- K means classification

Unsupervised case

◦ **K-means algorithm**

- choose $\mu_1^0, \mu_2^0, \ldots, \mu_K^0$

At iteration k:

- $\forall s \in S \quad l_s = argmin_{i \in \Lambda} ||x_s - \mu_i^k||^2$
- $\forall i \in \{1, \ldots, K\} \quad \mu_i^{k+1} = \frac{1}{card(R_i)} \sum_{s, l_s = i} x_s$
- if $\mu_i^k \neq \mu_i^{k+1}$ iterate

◦ **Drawbacks**

- no proof of convergence to the optimal solution
- influence of the initial means