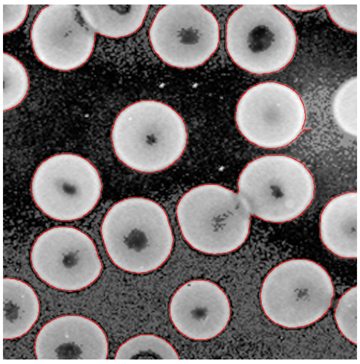
Image Processing

Chapter 5

Image Processing Tasks

Prof. Michael Unser, LIB



November 2023

CONTENT

- 5.1 Preprocessing
  - Histogram
  - Normalization
  - Combining images
  - Spatial averaging
- 5.2 Matching and detection
  - Correlation
  - Matched filtering
- 5.3 Feature extraction
  - Contour detection
  - Texture analysis
- 5.4 Segmentation
  - Variational thresholding
  - Connected component labelling

5.1 PREPROCESSING

- Histogram
- Normalization
- Combining images
- Spatial averaging (smoothing)
- Median filtering

Graylevel histogram

Input image: $r[\boldsymbol{k}]$ with $\boldsymbol{k} \in \Omega = \{0, \dots, K-1\} \times \{0, \dots, L-1\}$

Total number of pixels: $\#\Omega = K \times L$

- Graylevel distribution

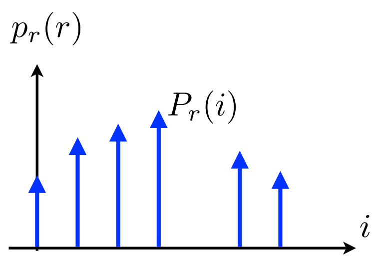
  Probability density function $p_r(r)$ with $\int_{-\infty}^{+\infty} p_r(r)\,\mathrm{d}r = 1$

- Histogram

  Quantized graylevels: $\{0, 1, 2, \dots, N_g - 1\}$

  n_i: number of pixels with graylevel i

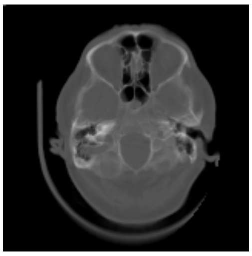
  $P_r(i) = \dfrac{n_i}{\#\Omega}$: relative occurrence of graylevel i

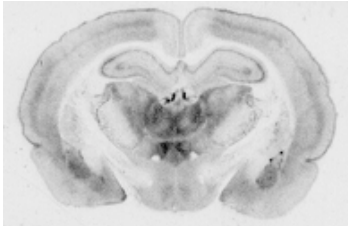


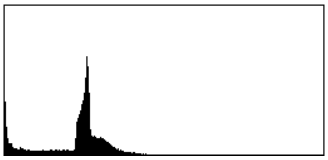
- Discrete probability density function

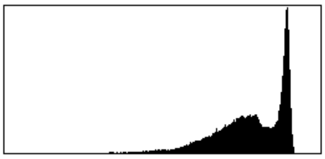
$$p_r(r) = \sum_{i=0}^{N_g-1} P_r(i)\delta(r-i)$$

Examples of histograms









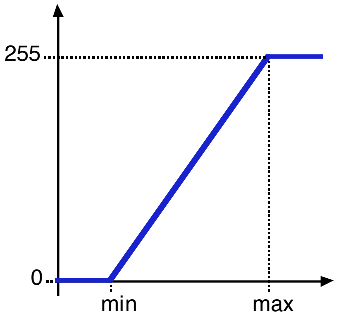
- Reading the histogram can tell us about
  - Dynamic range
  - Potential saturation problems
  - Average intensities of background and objects

Normalization : Linear contrast adjustment

Pointwise linear transformation: $T(f) = \alpha(f - \beta)$ with parameters $\alpha, \beta \in \mathbb{R}$

- Full dynamic-range contrast stretching

$$\beta = \min\{f\} \qquad \alpha = \frac{255}{\max\{f\} - \min\{f\}}$$



- Normalization

Average gray level

$$\mu = \frac{1}{\#\Omega} \sum_{\boldsymbol{k}\in\Omega} f[\boldsymbol{k}]$$

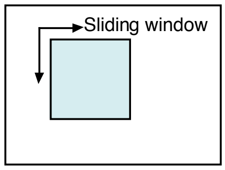
Variance

$$\sigma^2 = \frac{1}{\#\Omega} \sum_{\boldsymbol{k}\in\Omega} (f[\boldsymbol{k}] - \mu)^2$$

Normalized image statistics: $T(f) = a\left(\frac{f-\mu}{\sigma}\right) + b$

Local normalization

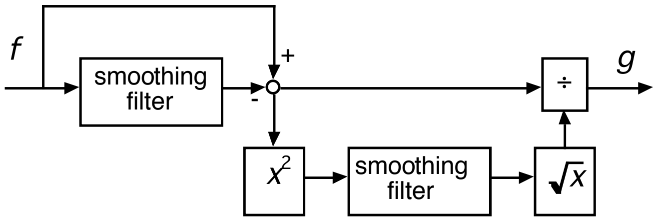
Compensation of non-uniformities across the image field;
e.g., shading, nonuniform background, changes in illumination



- Normalization over a sliding window

$$g[\boldsymbol{k}] = a\left(\frac{f[\boldsymbol{k}] - \tilde{\mu}[\boldsymbol{k}]}{\tilde{\sigma}[\boldsymbol{k}]}\right) + b$$

Weighted averaging: $\tilde{\mu}[\boldsymbol{k}_0] = \sum_{\boldsymbol{k}} w[\boldsymbol{k}] f[\boldsymbol{k} - \boldsymbol{k}_0]$ with $\sum_{\boldsymbol{k}} w[\boldsymbol{k}] = 1$



Online IP demo

Smoothing filter implements a local averaging window $\Rightarrow$ Estimation of local statistics

Combining images

- Averaging for noise reduction
  - Independent noisy observations: $f_i[\boldsymbol{k}] = s[\boldsymbol{k}] + n_i[\boldsymbol{k}] \qquad (i = 1, \dots, N)$
  - Hypotheses

    (a) $\mathbb{E}\{f_i[\boldsymbol{k}]\} = s[k, l] \quad \Rightarrow \quad \mathbb{E}\{n_i[\boldsymbol{k}]\} = 0$

    (b) i.i.d. noise at each location $\boldsymbol{k}$ $\Rightarrow$ $\mathrm{Var}\{f_i[\boldsymbol{k}]\} = \mathrm{Var}\{n_i[\boldsymbol{k}]\} = \sigma^2[\boldsymbol{k}]$

  - Noise reduction: $\bar{f}[\boldsymbol{k}] = \frac{1}{N}\sum_{i=1}^{N} f_i[\boldsymbol{k}]$

    Mean: $\mathbb{E}\{\bar{f}[\boldsymbol{k}]\} = s[\boldsymbol{k}]$

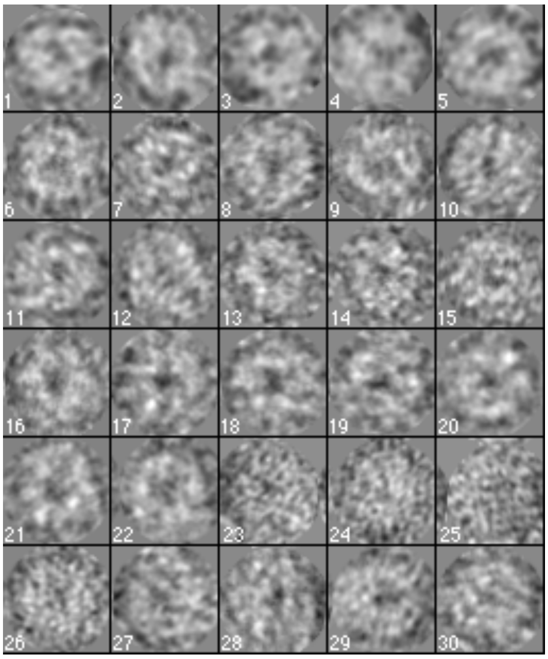
    Variance: $\mathrm{Var}\{\bar{f}[\boldsymbol{k}]\} = \frac{1}{N^2}\sum_{i=1}^{N} \mathrm{Var}\{f_i[\boldsymbol{k}]\} = \frac{\sigma^2[\boldsymbol{k}]}{N}$

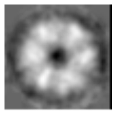
    $\Rightarrow$ Signal-to-noise ratio up by $\sqrt{N}$

    Central-limit Theorem: $\bar{f}[\boldsymbol{k}] \sim \mathrm{Gauss}\left(s[\boldsymbol{k}], \sigma^2/N\right)$

Example: noise reduction

Correlation-aligned Herpes Simplex Type 2 Capsomers (electron micrographs)



Result of averaging: 

Practical problems

- Registration
- Detection of outliers

Spatial averaging: smoothing

Linear smoothers = Lowpass filters $\Leftrightarrow$ $g = h * f$ with $\sum_{\boldsymbol{k}\in\mathbb{Z}^d} h[\boldsymbol{k}] = 1$

- Finite impulse response (FIR)

  Moving average

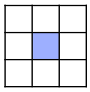
$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \boxed{\frac{1}{9}} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix} \qquad \begin{bmatrix} 0 & \frac{1}{8} & 0 \\ \frac{1}{8} & \boxed{\frac{1}{2}} & \frac{1}{8} \\ 0 & \frac{1}{8} & 0 \end{bmatrix}$$

- Infinite impulse response (IIR)
  - Symmetric exponential
  - Gaussian filter

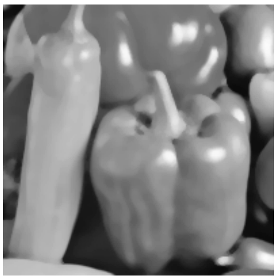
- Main uses
  - Image simplification
  - Noise reduction (high frequency)
  - Estimation of local statistics (mean, energy)
  - Multiscale processing

- Limitations
  - Blurring of edges and image details
    $\Rightarrow$ nonlinear operators

Spatial averaging: median filter

$g[\boldsymbol{k}] = \mathrm{Median}\,\{f[\boldsymbol{k}-\boldsymbol{i}], \boldsymbol{i} \in W\}$ W neighborhood: 

Input (200×200)

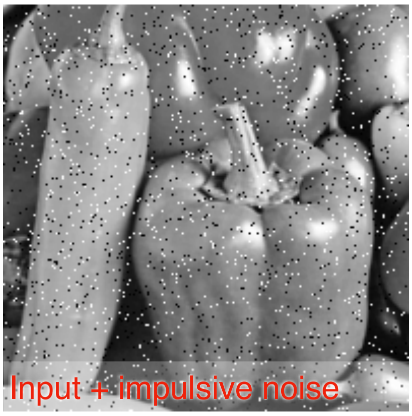


5×5 median filtered

- Advantages
  - Tends to preserve contours better than linear smoothers
  - Good for impulsive or heavy-tailed (non-Gaussian) noise (robust estimation)
- Limitations
  - Computationally costly for large size of neighborhood
  - Breaks down when there is a majority of noisy pixels

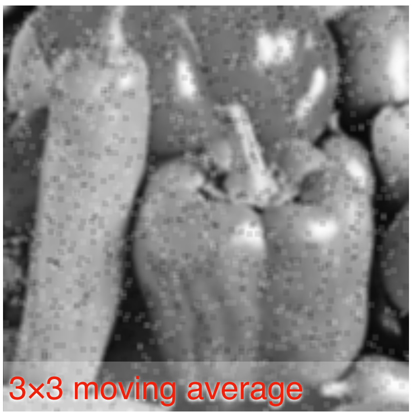
Impulsive noise reduction experiment

Input: 200×200



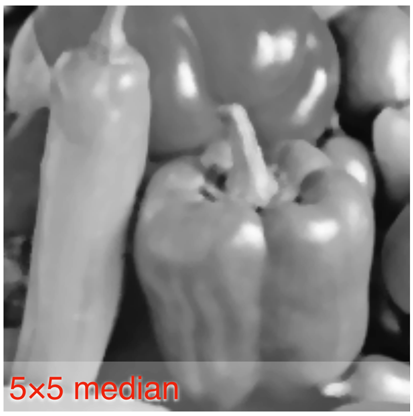
Input + impulsive noise

3×3 median



3×3 moving average

5×5 moving average



5×5 median

5.2 MATCHING AND DETECTION

- Template matching
  - Problem definition
  - Correlation
- Matched-filter detection
- *Application areas*
  - *Object detection*
  - *Automated inspection*
  - *Data fusion*
  - *Registration*
  - *Motion compensation*

Template matching

- Problem definition

Reference pattern, target, or template(s): $f_r[\boldsymbol{k}], \boldsymbol{k} \in \Omega_r$

Test image: $f[\boldsymbol{k}], \boldsymbol{k} \in \Omega_f$

Common support: $\Omega = \Omega_f \cap \Omega_r$

How do we decide that f and f_r are similar?

Given a collection of templates f_i (*e.g.*, shifted version of our reference), how do we select the best match?

Correlation measures

- Basic correlation (or $\ell_2(\Omega)$-inner product)

$$\langle f, f_r \rangle_{\ell_2} = \sum_{\boldsymbol{k} \in \Omega} f[\boldsymbol{k}]\, f_r[\boldsymbol{k}]$$

Relation with Euclidean distance

$$\|f - f_r\|_{\ell_2}^2 = \langle f - f_r, f - f_r \rangle_{\ell_2} = \|f\|_{\ell_2}^2 + \|f_r\|_{\ell_2}^2 - 2\, \langle f_r, f \rangle_{\ell_2}$$

Given a collection of templates with $\|f_r\|^2 \approx \mathrm{const}$

$\|f - f_r\|^2$ is minimum $\Leftrightarrow$ $\langle f_r, f \rangle$ is maximum

Correlation measures (Cont'd)

- Centered correlation

Motivation: invariance to a constant intensity offset b with $f = f_0 + b$

$$\left\langle f - \bar{f}, f_r - \bar{f}_r \right\rangle_{\ell_2} = \sum_{\boldsymbol{k} \in \Omega} \left(f[\boldsymbol{k}] - \bar{f}\right)\, \left(f_r[\boldsymbol{k}] - \bar{f}_r\right)$$

where the average value is $\bar{f} = \frac{1}{\#\Omega} \sum_{\boldsymbol{k} \in \Omega} f[\boldsymbol{k}]$

Note: $\left\langle f - \bar{f}, f_r - \bar{f}_r \right\rangle_{\ell_2} = \left\langle f - \bar{f}, f_r \right\rangle_{\ell_2} = \left\langle f, f_r - \bar{f}_r \right\rangle_{\ell_2}$

- Normalized correlation coefficient

Motivation: invariance to linear amplitude scaling $f = a\, f_0 + b$

$$-1 \leqslant \rho\{f, f_r\} = \frac{\left\langle f - \bar{f}, f_r - \bar{f}_r \right\rangle_{\ell_2}}{\|f - \bar{f}\|_{\ell_2}\, \|f_r - \bar{f}_r\|_{\ell_2}} \leqslant 1$$

Schwarz inequality: $\langle f, g \rangle \leqslant \|f\|\, \|g\|$

Matched-filter detection

- Measurement model (signal + noise): $f[\boldsymbol{k}] = s[\boldsymbol{k} - \boldsymbol{k}_0] + n[\boldsymbol{k}]$

  s: known deterministic signal or template

  n: additive white noise with zero mean and variance σ^2

  $\boldsymbol{k}_0$: unknown signal location $\qquad \mathbb{E}\{f[\boldsymbol{k}]\} = s[\boldsymbol{k} - \boldsymbol{k}_0]$

- Correlation-like detector

$$
\begin{aligned}
g[\boldsymbol{k}] &= (h * f)[\boldsymbol{k}] \\
&= \underbrace{\sum_{\boldsymbol{k}_1 \in \mathbb{Z}^d} h[\boldsymbol{k}_1]\, f[\boldsymbol{k} - \boldsymbol{k}_1]}_{\text{convolution}} = \underbrace{\sum_{\boldsymbol{k}_2 \in \mathbb{Z}^d} w[\boldsymbol{k}_2]\, f[\boldsymbol{k} + \boldsymbol{k}_2]}_{\text{correlation}}
\end{aligned}
$$

  where $w[\boldsymbol{k}] = h[-\boldsymbol{k}]$

Optimal matched filter

- Optimum detector: maximum SNR at $\boldsymbol{k} = \boldsymbol{k}_0$

  Solution: $\quad w[\boldsymbol{k}] = s[\boldsymbol{k}] \qquad$ (matched filter)

Proof:

Signal estimate at $\boldsymbol{k} = \boldsymbol{k}_0$

$$\mathbb{E}\{g[\boldsymbol{k}_0]\} = \sum_{\boldsymbol{k}_1 \in \mathbb{Z}} w[\boldsymbol{k}_1]\, s[\boldsymbol{k}_0 - \boldsymbol{k}_0 + \boldsymbol{k}_1] = \langle w, s \rangle_{\ell_2}$$

Residual-noise variance

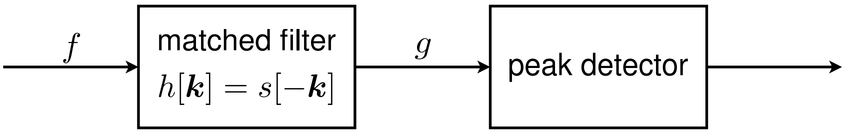
$$\mathrm{Var}\{g[\boldsymbol{k}]\} = \sum_{\boldsymbol{k}_1 \in \mathbb{Z}} w^2[\boldsymbol{k}_1]\, \mathrm{Var}\{n[\boldsymbol{k} + \boldsymbol{k}_1]\} = \|w\|_{\ell_2}^2\, \sigma^2$$

Signal-to-noise ratio at $\boldsymbol{k} = \boldsymbol{k}_0$: $\mathrm{SNR} = \dfrac{\langle s, w \rangle_{\ell_2}}{\|w\|_{\ell_2}\, \sigma}$

Cauchy-Schwarz inequality

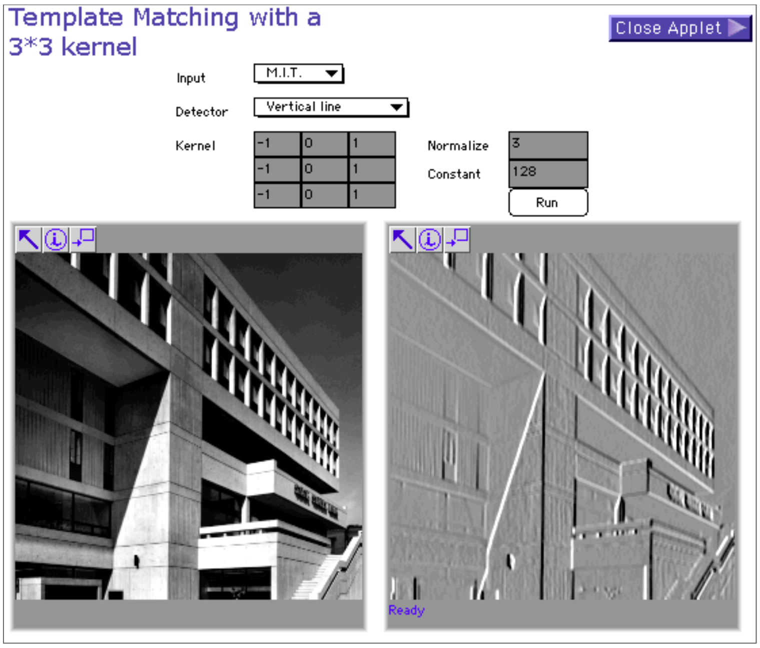
$$\langle s, w \rangle_{\ell_2} \leqslant \|s\|_{\ell_2}\, \|w\|_{\ell_2} \qquad \text{with equality iff. } w[\boldsymbol{k}] = \lambda\, s[\boldsymbol{k}]$$

Pattern detection by template matching



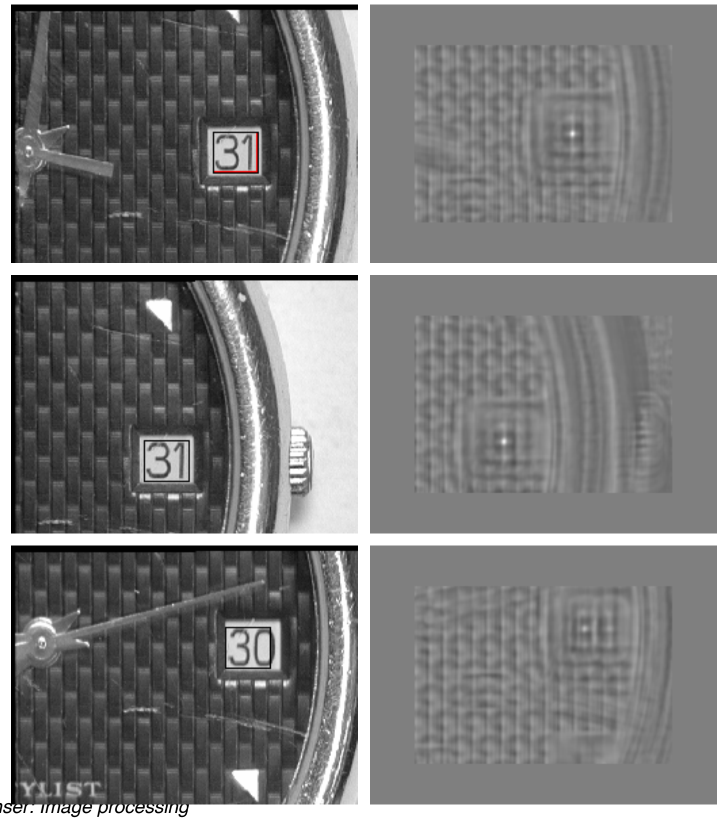
- Application

  Line detector



Template matching: example

Reference template (33×31 pixels) 



$x = 149,\, y = 95,\, \rho = 100\%$

$x = 98,\, y = 123,\, \rho = 88\%$

$x = 58,\, y = 61,\, \rho = 33\%$

Matched filtering: extension to colored noise

Make the noise white and you are back to the previous problem!

$$\Phi_f(\mathrm{e}^{\mathrm{j}\boldsymbol{\omega}}) \xrightarrow{f} \boxed{H(\mathrm{e}^{\mathrm{j}\boldsymbol{\omega}})} \xrightarrow{h*f} |H(\mathrm{e}^{\mathrm{j}\boldsymbol{\omega}})|^2\Phi_f(\mathrm{e}^{\mathrm{j}\boldsymbol{\omega}})$$

- Prewhitening filter: $\dfrac{1}{\sqrt{\Phi_n(\mathrm{e}^{\mathrm{j}\boldsymbol{\omega}})}}$

  where $\Phi_n(\mathrm{e}^{\mathrm{j}\boldsymbol{\omega}})$ is the spectral power density of the noise

- Prewhitened template: $P(\mathrm{e}^{\mathrm{j}\boldsymbol{\omega}}) = \dfrac{S(\mathrm{e}^{\mathrm{j}\boldsymbol{\omega}})}{\sqrt{\Phi_n(\mathrm{e}^{\mathrm{j}\boldsymbol{\omega}})}}$

$\Rightarrow$ Prewhitened matched filter: $H(\mathrm{e}^{\mathrm{j}\boldsymbol{\omega}}) = \dfrac{P^*(\mathrm{e}^{\mathrm{j}\boldsymbol{\omega}})}{\sqrt{\Phi_n(\mathrm{e}^{\mathrm{j}\boldsymbol{\omega}})}} = \dfrac{S^*(\mathrm{e}^{\mathrm{j}\boldsymbol{\omega}})}{\Phi_n(\mathrm{e}^{\mathrm{j}\boldsymbol{\omega}})}$

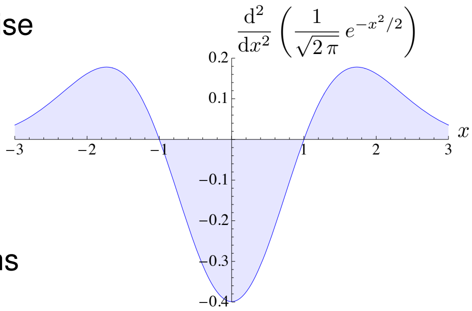
- Example

  Detection of a Gaussian blob φ in isotropic $1/\|\boldsymbol{\omega}\|^2$ noise

  Optimal detector (Mexican-hat filter)

  $$\Delta\varphi(\boldsymbol{x}) \quad \overset{\mathcal{F}}{\longleftrightarrow} \quad -\|\boldsymbol{\omega}\|^2\,\hat{\varphi}(\boldsymbol{\omega})$$

  Application: detection of μCA^{++} in digital mammograms



5.3 FEATURE EXTRACTION

- **Edge detection**

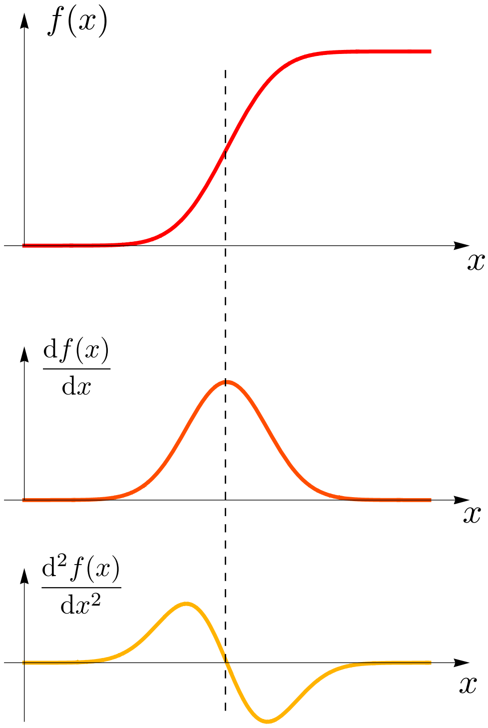
  Edges are important clues for the interpretation of images; they are essential to object recognition

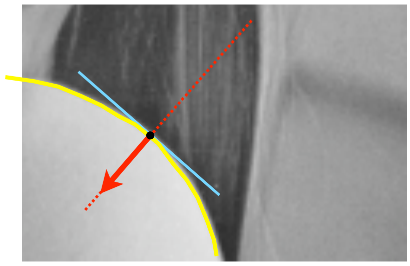
  - Edges: continuous formulation
  - Gradient-based edge detection

- **Texture analysis**
  - What is texture
  - Filterbank analysis
  - Towards texture segmentation

Edges: continuous-domain formulation

- Edge point: location of abrupt change in an image





Edge

Image value at location $\boldsymbol{x}$: $f(\boldsymbol{x})$

Normal vector: $\boldsymbol{n} = \dfrac{\boldsymbol{\nabla} f(\boldsymbol{x})}{\|\boldsymbol{\nabla} f(\boldsymbol{x})\|}$

$\Rightarrow$ direction of maximum change

Gradient and directional derivatives

- Gradient of f at $\boldsymbol{x} = (x_1, x_2)$: $\boldsymbol{\nabla} f(\boldsymbol{x}) = \left(\frac{\partial f(\boldsymbol{x})}{\partial x_1}, \frac{\partial f(\boldsymbol{x})}{\partial x_2}\right) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}))$

- Directional derivative of f along the unit vector $\boldsymbol{u}_\theta = (\cos\theta, \sin\theta)$

$$\begin{aligned}\mathrm{D}_{\boldsymbol{u}_\theta} f(\boldsymbol{x}) &= \lim_{\epsilon \to 0} \frac{f(\boldsymbol{x} + \epsilon\, \boldsymbol{u}_\theta) - f(\boldsymbol{x})}{\epsilon} \\ &= f_1(\boldsymbol{x})\ \cos\theta + f_2(\boldsymbol{x})\ \sin\theta\end{aligned}$$

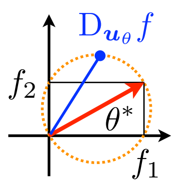
Taylor-series argument :
$f(\boldsymbol{x} + \epsilon \boldsymbol{u}) =$
$f(\boldsymbol{x}) + \epsilon \boldsymbol{u}^{\mathrm{T}} \boldsymbol{\nabla} f(\boldsymbol{x}) + \mathcal{O}(\epsilon^2)$

- Generalization to d-dimensions: derivative of f along the vector $\boldsymbol{u}$

$$\mathrm{D}_{\boldsymbol{u}} f(\boldsymbol{x}) = \lim_{\epsilon \to 0} \left(\frac{f(\boldsymbol{x} + \epsilon\, \boldsymbol{u}) - f(\boldsymbol{x})}{\epsilon\, \|\boldsymbol{u}\|} \right) = \left\langle \frac{\boldsymbol{u}}{\|\boldsymbol{u}\|}, \boldsymbol{\nabla} f(\boldsymbol{x}) \right\rangle$$

- Maximum of the directional derivative (Cauchy-Schwartz)

$$\max_\theta \{\mathrm{D}_{\boldsymbol{u}_\theta} f\} = \mathrm{D}_{\boldsymbol{n}} f = \|\boldsymbol{\nabla} f\| = \sqrt{f_1^2 + f_2^2}$$



- Direction of maximum deviation

$$\theta^* = \angle(\boldsymbol{\nabla} f) = \arctan\left(\frac{f_2}{f_1}\right) + k\,\pi,\ k \in \mathbb{Z} \qquad (\perp \text{ to edge})$$

General criteria for edge detection

- Maximum of the gradient
- Zero-crossings of the second-order (directional) derivative
- Combination of both

- Remarks
  - Gradient magnitude and Laplacian are rotationally invariant while gradient vectors and directional second-order derivatives are not
  - Derivatives are usually estimated on a smoothed version of the image to improve robustness and/or reduce the effect of noise or irrelevant details

  $\Rightarrow$ Multiscale approaches

Gradient-based edge detection

- Discretized gradient operators

  Horizontal derivative: $g_1[\boldsymbol{k}] = (h_1 * f)\,[\boldsymbol{k}]$

  Vertical derivative: $g_2[\boldsymbol{k}] = (h_2 * f)\,[\boldsymbol{k}]$

  $$g[k_1,k_2] = \sqrt{g_1^2[k_1,k_2] + g_2^2[k_1,k_2]}$$

  $$\theta_g[k_1,k_2] = \arctan\!\left(\frac{g_2[k_1,k_2]}{g_1[k_1,k_2]}\right) + n\,\pi, n \in \mathbb{Z}$$

Centered finite differences

$$\partial_x \approx \begin{array}{|c|c|c|}\hline \frac{1}{2} & 0 & -\frac{1}{2} \\ \hline \end{array}$$

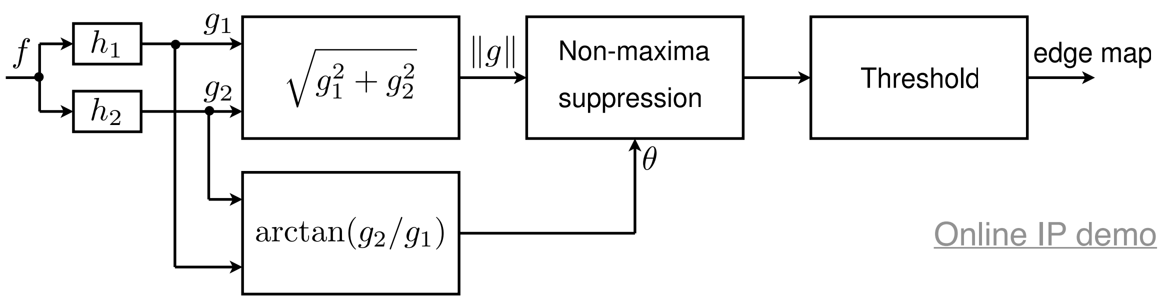
$$\partial_y \approx \begin{array}{|c|}\hline \frac{1}{2} \\ \hline 0 \\ \hline -\frac{1}{2} \\ \hline \end{array}$$

- Threshold-based edge detection

$$\mathrm{edge}[k_1,k_2] = \begin{cases} 1 & g[k_1,k_2] \geqslant T_1 \\ 0 & \text{otherwise} \end{cases}$$

Canny's edge detection algorithm

- Refinements
  - Non-maxima suppression: based on local search in the direction θ_g
  - Hysteresis threshold: contour segments above T_1 (high threshold) are grown such as to include all connected points with $g[k_1, k_2] \geqslant T_0$ (low threshold)

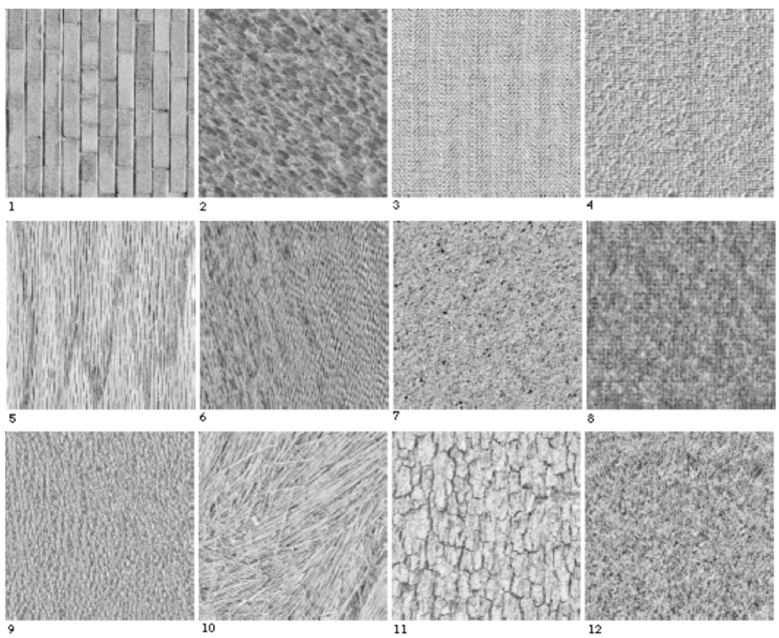


Online IP demo

What is texture ?

What is meant by texture

- Local order or pattern
- Neighborhood property
- Invariance by translation
- Homogeneity
- Subjective notion related to visual perception



Notation: $x[\boldsymbol{k}],\ \boldsymbol{k} \in \Omega$ (texture region)

Gaussian texture model

- Power spectral density function

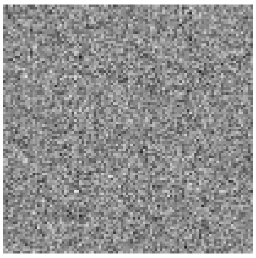
$$\Phi_x(e^{j\boldsymbol{\omega}}) = \sum_{\boldsymbol{k}\in\mathbb{Z}^2} a_x[\boldsymbol{k}]e^{-j\langle\boldsymbol{k},\boldsymbol{\omega}\rangle} \qquad \text{(Wiener-Khintchine relation)}$$

where $a_x[\boldsymbol{k}] = \mathbb{E}\{x[\cdot]x[\cdot+\boldsymbol{k}]\}$ (autocorrelation)

- LSI system

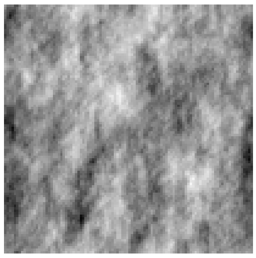
$$x[\boldsymbol{k}] = (h * n)[\boldsymbol{k}] \qquad \longleftrightarrow \qquad \Phi_x(e^{j\boldsymbol{\omega}}) = |H(e^{j\boldsymbol{\omega}})|^2 \cdot \Phi_n(e^{j\boldsymbol{\omega}})$$

- Gaussian texture generation model



Gaussian white noise $\longrightarrow$ $\boxed{H(e^{j\boldsymbol{\omega}}) = \sqrt{\Phi_x(e^{j\boldsymbol{\omega}})}}$ $\longrightarrow$ 2D stationary process

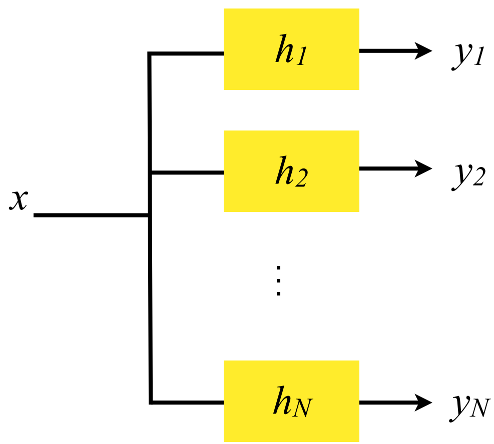
$\Phi_n(e^{j\boldsymbol{\omega}}) = 1$ $\qquad$ $\Phi_x(e^{j\boldsymbol{\omega}})$



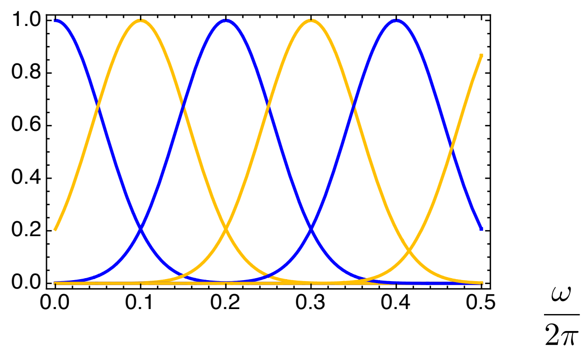
Filterbank analysis

- Multichannel filterbank

$$y_i[\boldsymbol{k}] = (h_i * x)[\boldsymbol{k}], \qquad i = 1, \dots, N$$



$H_i(\omega)$



Filterbank analysis (Cont'd)

- Channel statistics

Histograms: $P_i(a) = \mathrm{Prob}\{y_i = a\}$

Moments: $m_{i,p} = \mathbb{E}\{|y_i|^p\} = \sum_a |a|^p P_i(a)$

Texture energies: $\sigma_i^2 = \mathrm{Var}\{y_i\} = \begin{cases} m_{1,2} - \mu^2, & i = 1 \quad \text{(lowpass)} \\ m_{i,2}, & i \neq 1 \quad \text{(highpass)} \end{cases}$

- Spatial estimators over a texture region Ω

$$\hat{P}_i(a) = \frac{1}{\#\Omega} \sum_{\boldsymbol{k} \in \Omega} \delta_{y_i[\boldsymbol{k}]-a}$$

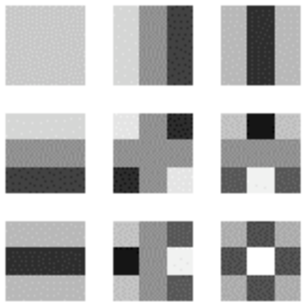
$$\hat{m}_{i,p} = \frac{1}{\#\Omega} \sum_{\boldsymbol{k} \in \Omega} |y_i[\boldsymbol{k}]|^p$$

Pratical issues

- Choice of the filterbank
  - Local linear transforms (Unser 1986)

    $\Rightarrow$ Sliding 3×3 DCT or DST

    Motivation: fast algorithms, good approximation of KLT



Filter masks for the 3x3 DCT

  - Gabor filters (Fogel 1989)

    Motivation: similarity with visual system

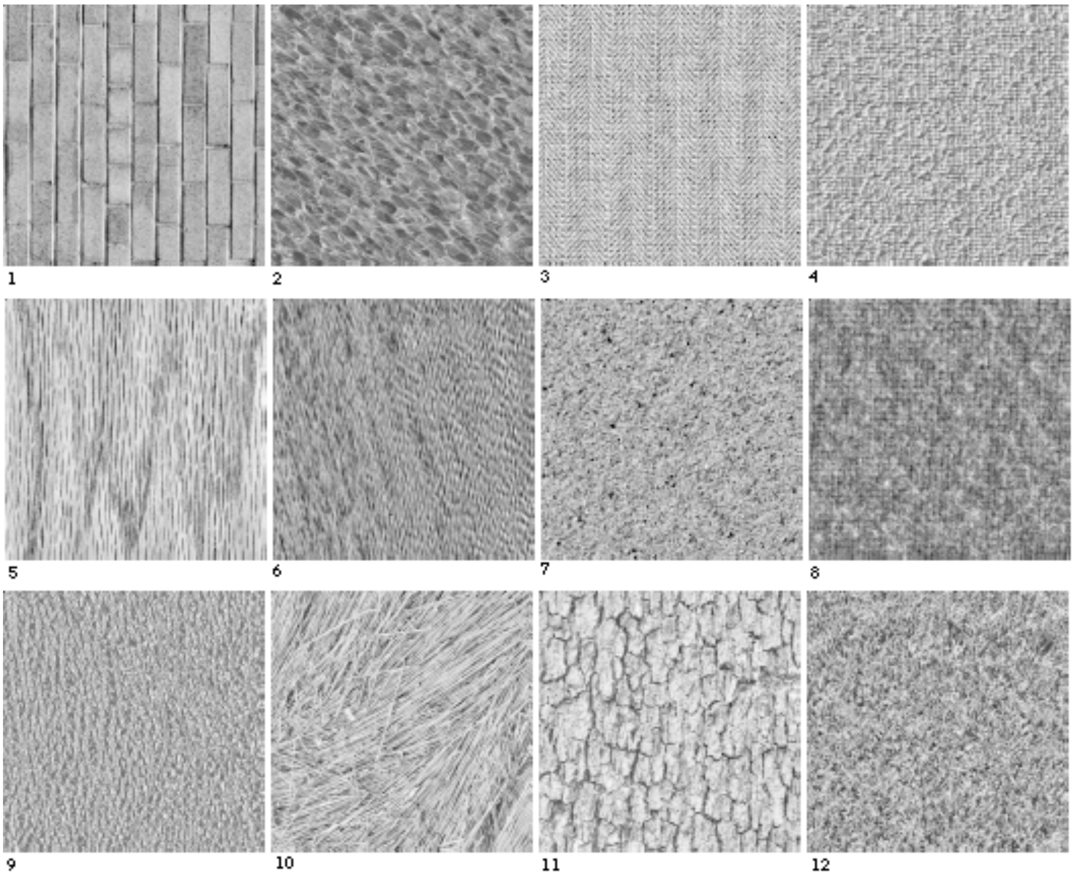
  - Wavelet filterbanks (with or without decimation) (Unser 1995)

    Motivation: fast algorithm; multiscale analysis

  - Convolutional neural networks

    Motivation: the "learning revolution" = data-driven design

Texture classification



Texture-classification results

- Test data
  - 12 Brodatz textures
  - Equalized histograms (32 levels)
  - (32×32) non-overlapping regions
- Training and classification
  - Maximum-likelihood estimation of

    $(\mathbf{m}_i, \mathbf{C}_i)$ for $i \in \{1, \ldots, K\}$
  - Leave-one-out method

- Confusion matrix (line: true class; column: assigned class)

	1	2	3	4	5	6	7	8	9	10	11	12
1	64	0	0	0	0	0	0	0	0	0	0	0
2	0	64	0	0	0	0	0	0	0	0	0	0
3	0	0	64	0	0	0	0	0	0	0	0	0
4	0	0	0	64	0	0	0	0	0	0	0	0
5	0	0	0	0	64	0	0	0	0	0	0	0
6	0	0	0	0	0	64	0	0	0	0	0	0
7	0	0	0	0	0	0	62	0	0	0	0	2
8	0	0	0	0	0	0	0	64	0	0	0	0
9	0	0	0	0	0	0	0	0	64	0	0	0
10	0	0	0	0	0	0	0	0	0	64	0	0
11	0	0	0	0	0	0	0	1	0	0	63	0
12	0	0	0	0	0	0	5	0	0	1	0	58

Number of features: 9 texture energies (3x3 DCT)

Number of errors: 9 out of 768

Total score: 98.83%

Towards texture segmentation

- Basic principle

  Define a local feature map $\boldsymbol{f}[\boldsymbol{k}]$ associated to a window centered on current pixel

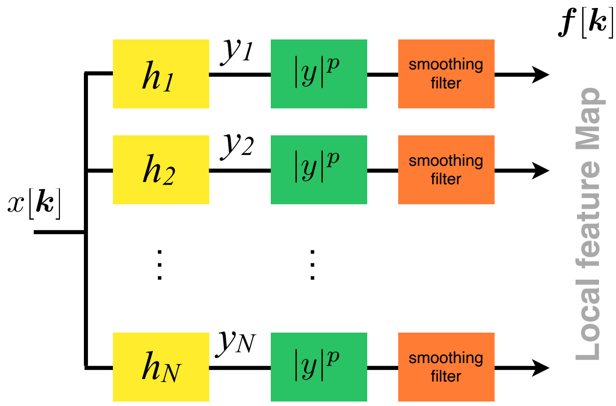
- Efficient multichannel implementation

  Smoothing filter implementing a local-averaging window

  $\Rightarrow$ Estimation of local statistics (moments)

  Gaussian smoother

  - isotropic weighting window
  - optimal space/frequency localization

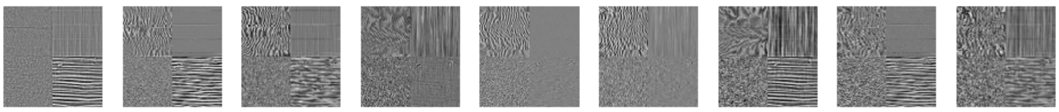


- Additional processing steps
  - Feature reduction; *e.g.*, Karhunen-Loève transform
  - Classification or clustering

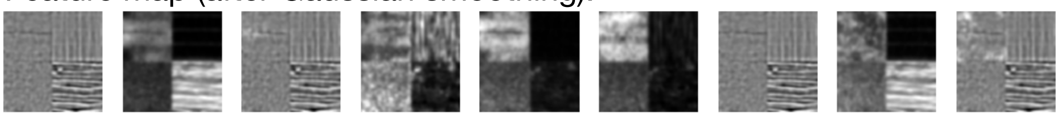
Example of filterbank analysis

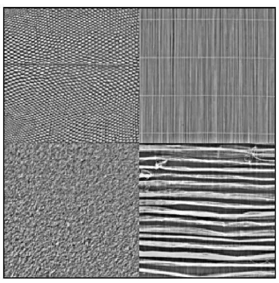
DCT filters (3x3)

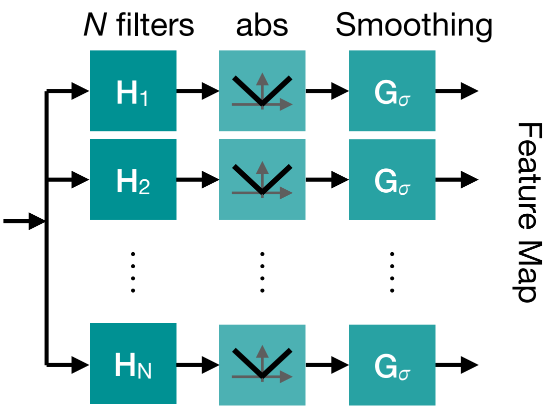
Outputs of filterbank:



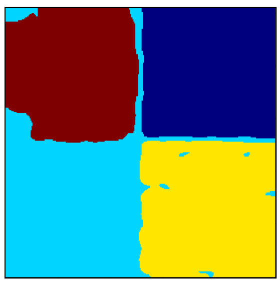
Feature map (after Gaussian smoothing):







K-means clustering (pixelwise)



5.4 IMAGE SEGMENTATION

- Segmentation: art or science?
- Amplitude thresholding
  - Variational thresholding
  - Statistical thresholding
- Binary segmentation techniques

Segmentation problem

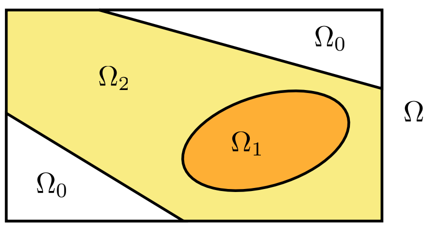
- Definition

Image $f[\boldsymbol{k}]$, with $\boldsymbol{k} \in \Omega$

Image segmentation: Find a partition of the support Ω of the image f, with

$$\Omega = \bigcup_i \Omega_i \text{ with } \Omega_i \cap \Omega_j = \emptyset \text{ for } i \neq j$$

such that the regions Ω_i satisfy some homogeneity (and connectivity) criterion.



The total number of regions I is not necessarily known

- Three main approaches
  - Pixel classification
  - Region-based segmentation
  - Boundary-based segmentation $\Rightarrow$ Edge detection

Segmentation: art or science?

Problem: lack of a universal definition of homogeneity
⇨ many application-specific approaches

- Approaches for specifying homogeneity
  - Empirical (e.g., similar graylevels; feature maps)
  - Statistical, based on some a priori model
    (e.g., constant mean + additive white noise)
- Approaches for enforcing connectivity (if required)
  - Prior information about object size or shape
  - Joint probability model for class labels
  - Contour length

Segmentation as an optimization problem

Variational *vs.* **Markov-random-field** approaches

Principle: maximize the quality of any candidate segmentation, as measured by a functional that incorporates all problem-specific knowledge

Criterion = Data term + Regularization term

homogeneity connectivity

Amplitude thresholding

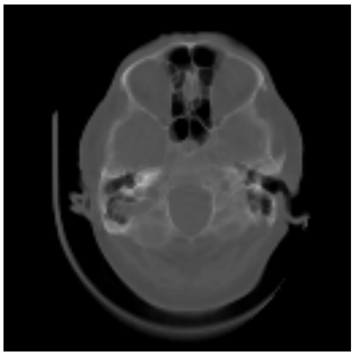
- Empirical approach

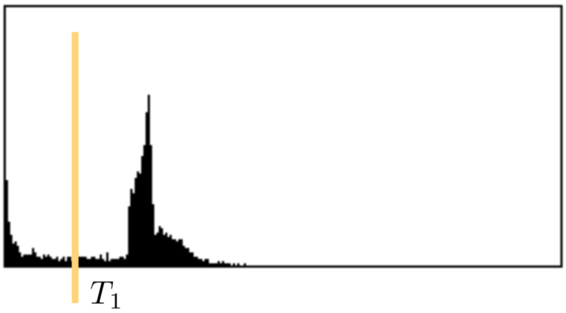
Based on the histogram, select a collection of thresholds

$T_0 < \cdots < T_i < \cdots < T_I$

and use the following rule to assign regions:

$(k,l) \in \Omega_i$ for $T_i \leqslant f[k,l] < T_{i+1}$





Variational thresholding

Principle: minimize an appropriate goodness-of-fit criterion

- Variational formulation

Constant-mean model: $f[\boldsymbol{k}] = \mu_i$, $\boldsymbol{k} \in \Omega_i$

Find μ_i and Ω_i s. t. $\displaystyle\sum_i \sum_{\boldsymbol{k}\in\Omega_i} \left(f[\boldsymbol{k}] - \mu_i\right)^2$ is minimum

$\Rightarrow$ Same problem as **Max-Lloyd quantization** (K-means)

Simple iterative two-step optimization scheme

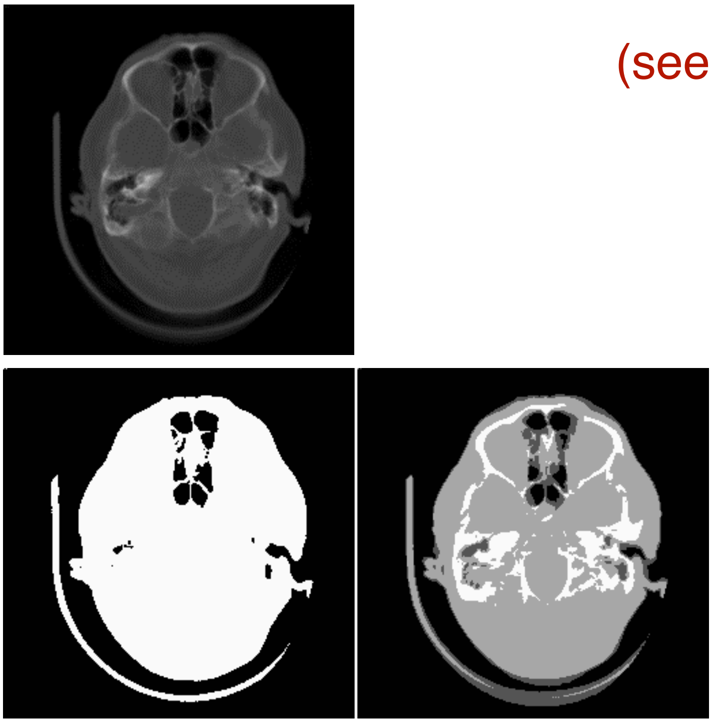
1. Given Ω_i, compute region means μ_i
2. Given μ_i, compute optimal partitions $\Omega_1, \ldots, \Omega_I$ $\Rightarrow$ $T_{i+1} = \frac{1}{2}\left(\mu_i + \mu_{i+1}\right)$

Note: all computations can be done from the histogram

Segmentation by minimum-error quantization

Search for the "optimal" threshold to segment images

(see Chap 2)



Minimum-error solution: *K*=2 *K*=4

Statistical thresholding

Principle: Find the most "likely" segmentation model

- Statistical formulation with labels $x[\boldsymbol{k}] = i \Leftrightarrow \boldsymbol{k} \in \Omega_i$

  Standard "statistics" notation

  $X = \{x[\boldsymbol{k}] : \boldsymbol{k} \in \Omega\}$: unknown labels

  $Y = \{\boldsymbol{f}[\boldsymbol{k}] : \boldsymbol{k} \in \Omega\}$: observed data = image or feature map to segment

  $\boldsymbol{f} : \Omega \to \mathbb{R}^N$ (scalar=graylevel, RGB or feature map)

  - Class-conditional probability at location $\boldsymbol{k}$, assuming i.i.d. Gaussian feature channels

$$p(\boldsymbol{f} | x = i) = \prod_{n=1}^{N} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(f_n - \mu_{i,n})^2}{2\,\sigma^2}\right) = \frac{1}{(\sigma\sqrt{2\pi})^N} \exp\left(-\frac{\|\boldsymbol{f} - \boldsymbol{\mu}_i\|^2}{2\,\sigma^2}\right)$$

Feature vector: $\boldsymbol{f} = \boldsymbol{f}[\boldsymbol{k}] = (f_n) \in \mathbb{R}^N$

Parameters: $\sigma \in \mathbb{R}$, $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_I \in \mathbb{R}^N$

Statistical thresholding (Cont'd)

- Region labels: $x[\boldsymbol{k}] = i \Leftrightarrow \boldsymbol{k} \in \Omega_i$

- Joint probability density function (i.i.d. Gaussian components)

$$p(Y|\,X; \boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_I, \sigma) \propto \prod_{\boldsymbol{k}\in\Omega} \exp\Big(-\frac{\|\boldsymbol{f}[\boldsymbol{k}] - \boldsymbol{\mu}_{x[\boldsymbol{k}]}\|^2}{2\sigma^2}\Big)$$

  Log-likelihood

$$\log p(Y|\,X; \boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_I, \sigma) = C_0 + \sum_{\boldsymbol{k}\in\Omega} -\frac{\|\boldsymbol{f}[\boldsymbol{k}] - \boldsymbol{\mu}_{x[\boldsymbol{k}]}\|^2}{2\sigma^2}$$

- Maximum-likelihood estimate

  Find $x[\boldsymbol{k}]$ and $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_I \in \mathbb{R}^N$ such that

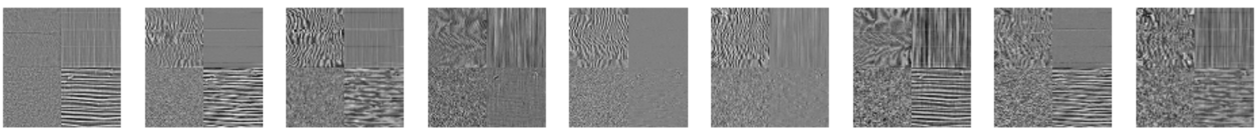
$$-\sum_{\boldsymbol{k}\in\Omega} \|\boldsymbol{f}[\boldsymbol{k}] - \boldsymbol{\mu}_{x[\boldsymbol{k}]}\|^2 = -\sum_i \sum_{\boldsymbol{k}\in\Omega_i} \|\boldsymbol{f}[\boldsymbol{k}] - \boldsymbol{\mu}_i\|^2 \text{ is maximum}$$

  $\Rightarrow$ Equivalent to minimum-error vector (Max-Lloyd) quantization (I-means)

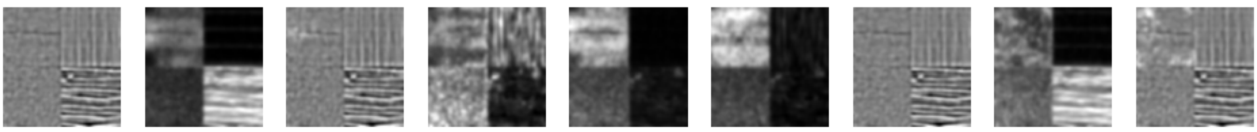
Texture segmentation by vector quantization

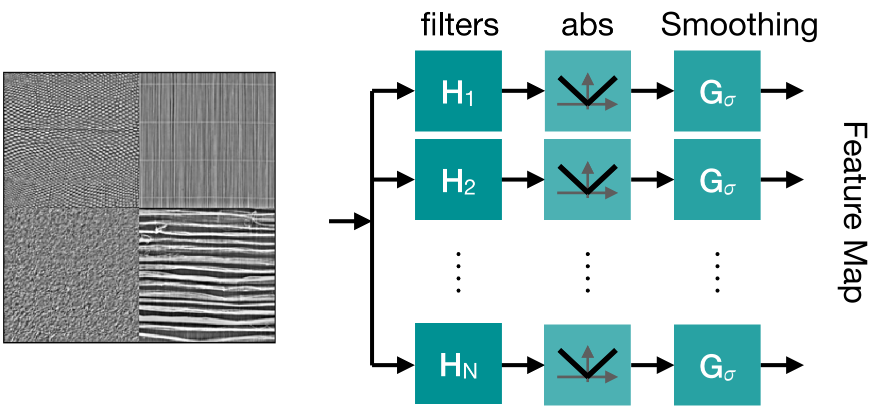
(see Section 5.3)

Outputs of filterbank (3x3 DCT)

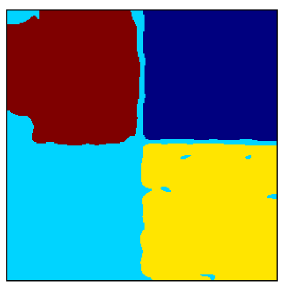


Feature map (after Gaussian smoothing):





K-means clustering
(pixelwise)



Binary-segmentation techniques

Objects or regions: set of points in $\mathbb{Z}^2$ (bitmap)

- Distance measures with $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{Z}^2$

  City-block distance: $D_4(\boldsymbol{a}, \boldsymbol{b}) = |a_1 - b_1| + |a_2 - b_2|$

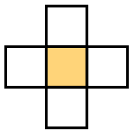
  Chessboard distance: $D_8(\mathbf{a}, \boldsymbol{b}) = \max(|a_1 - b_1|, |a_2 - b_2|)$

  distance $\Rightarrow$ ε-neighborhood $N(\mathbf{a})$ of a point $\mathbf{a}$ (with $D(\boldsymbol{a}, \boldsymbol{b}) \leqslant \varepsilon$)

- Connectivity

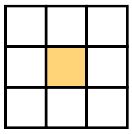
  4-connect neighborhood

  $$N_4(\boldsymbol{a}) = \{\boldsymbol{b} | \, \boldsymbol{b} \in \mathbb{Z}^2, D_4(\boldsymbol{a}, \boldsymbol{b}) \leqslant 1\}$$



  8-connect neighborhood

  $$N_8(\boldsymbol{a}) = \{\boldsymbol{b} | \, \boldsymbol{b} \in \mathbb{Z}^2, D_8(\boldsymbol{a}, \boldsymbol{b}) \leqslant 1\}$$



Binary-segmentation techniques (Cont'd)

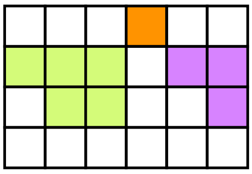
Objects or regions: set of points in $\mathbb{Z}^2$ (bitmap)

- Path

  List $\{\boldsymbol{a}_i : i \in [1 \dots N]\}$ of N connected pixels such that $\boldsymbol{a}_i \in N(\boldsymbol{a}_{i-1})$

- Connected components

  Maximum set of connected pixels



Binary-segmentation techniques (Cont'd)

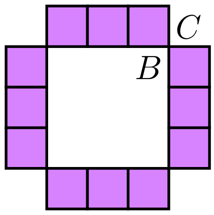
Background/foreground connectivity ambiguity

B and C are separated by an 8-connected contour; yet they are themselves 8-connected

Solution

8-connectivity for foreground

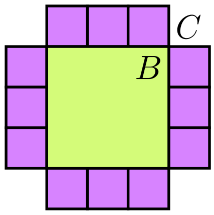
4-connectivity for backgroud



OR

4-connectivity for foreground

8-connectivity for backgroud



In 3-D: 6-connected *vs.* 18-connected *vs.* 26-connected

Connected-component labeling

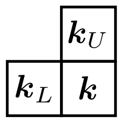
- Connected-component labeling (blob coloring algorithm)

$\mathbb{F}$: foreground $\square$

$\mathbb{B}$: background $\blacksquare$

$\mathbb{Z}^2 = \mathbb{F} \cup \mathbb{B}; \mathbb{F} \cap \mathbb{B} = \emptyset$

4-connected scanning window:



Start with color equivalences $\mathbb{E} = \emptyset$ and initial color $i = 1$

`Scan image from left to right, then top to bottom`

`if` $\boldsymbol{k} \in \mathbb{F}$ `then` $\{$

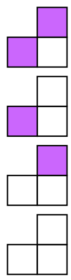
`if` $(\boldsymbol{k}_U \in \mathbb{B} \wedge \boldsymbol{k}_L \in \mathbb{B})$ `then` $\{\mathtt{color}(\boldsymbol{k}) = i;\ \mathbb{E} = \mathbb{E} \cup \{(i,i)\};\ i = i+1;\}$

`if` $(\boldsymbol{k}_U \in \mathbb{F} \wedge \boldsymbol{k}_L \in \mathbb{B})$ `then` $\mathtt{color}(\boldsymbol{k}) = \mathtt{color}(\boldsymbol{k}_U);$

`if` $(\boldsymbol{k}_U \in \mathbb{B} \wedge \boldsymbol{k}_L \in \mathbb{F})$ `then` $\mathtt{color}(\boldsymbol{k}) = \mathtt{color}(\boldsymbol{k}_L);$

`if` $(\boldsymbol{k}_U \in \mathbb{F} \wedge \boldsymbol{k}_L \in \mathbb{F})$ `then` $\{$

$\mathtt{color}(\boldsymbol{k}) = \mathtt{color}(\boldsymbol{k}_L);\ \mathbb{E} = \mathbb{E} \cup \{(\mathtt{color}(\boldsymbol{k}_U), \mathtt{color}(\boldsymbol{k}_L))\};\}$



$\}$

Blob coloring (Cont'd)

- Post-processing: Resolve color equivalences

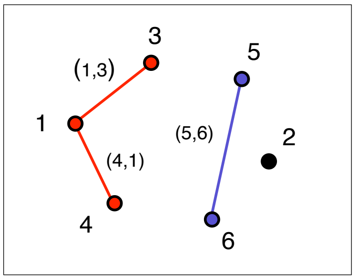
  For $j = 1$ to i do $\{$

  $\mathbb{U} = \mathit{getEquivalentColors}(j, \mathbb{E})$; $\forall u \in \mathbb{U} : \mathit{equivalenceTable}[u] = j$;$\}$

  For all pixels $\boldsymbol{k}$ set $\mathit{color}(\boldsymbol{k}) = \mathit{equivalenceTable}[\mathit{color}(\boldsymbol{k})]$;

  Graph representation of color-equivalence list

  $\mathbb{E} = \{(1,1), (2,2), (3,3), (1,3), (4,4), (4,1), (5,5), (6,6), (5,6)\}$



- Main applications
  - Finding image regions following an edge detection
  - Counting objets (cytology)
  - Modification for *region growing*: $\boldsymbol{k}, \boldsymbol{k}_U \in \mathbb{F} \Leftrightarrow |f(\boldsymbol{k}) - f(\boldsymbol{k}_U)| < T$