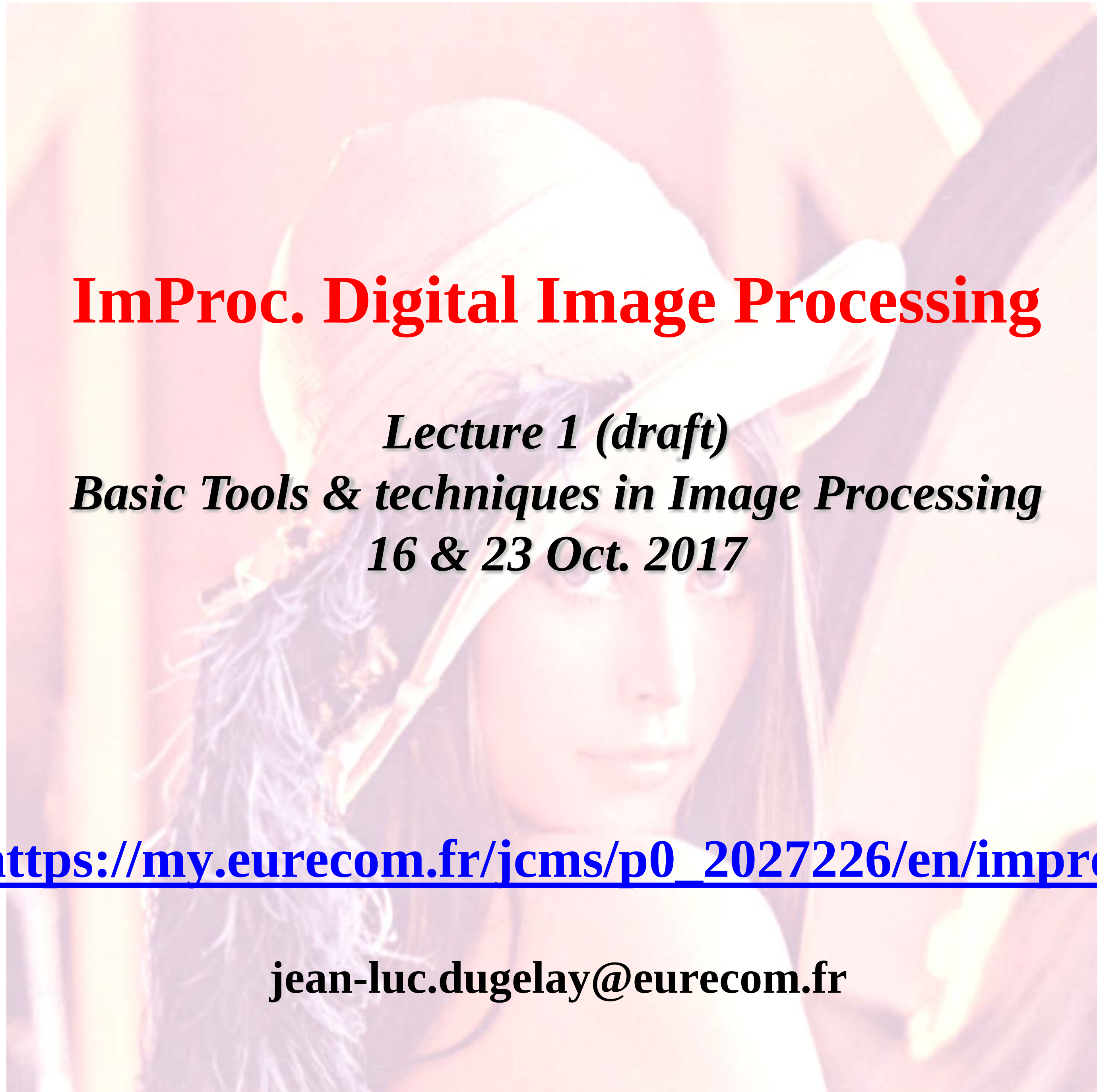


ImProc. Digital Image Processing

Lecture 1 (draft)
Basic Tools & techniques in Image Processing
16 & 23 Oct. 2017

https://my.eurecom.fr/jcms/p0_2027226/en/improc

jean-luc.dugelay@eurecom.fr

Typical Sequence

- **Pre-processing**
  - Noise reduction: low-pass Filtering
- **Processing**
  - Gradient maps + Threshold
  - Gradient based descriptor (hand-crafted *vs. learned* features)
- **Post-processing**
  - Edge thinning, closing
  - Outlier removal
- **Representation & Description**
  - Hough Transform

Spatial Masks

$$\begin{bmatrix} w_1 & w_2 & w_3 \\ (x-1, y-1) & (x-1, y) & (x-1, y+1) \\ w_4 & w_5 & w6 \\ (x, y-1) & (x, y) & (x, y+1) \\ w_7 & w_8 & w_9 \\ (x+1, y-1) & (x+1, y) & (x+1, y+1) \end{bmatrix}$$

$$T\left[f(x,y)\right] = w_1 f(x-1, y-1) + w_2 f(x-1, y)$$
$$+ w_3 f(x-1, y+1) + w_4 f(x, y-1)$$
$$+ w_5 f(x, y) + w_6 f(x, y+1) + w_7 f(x+1, y-1)$$
$$+ w_8 f(x+1, y) + w_9 f(x+1, y+1)$$

W1,...,W9: mask coefficents
8-neighbors of (x,y)

- ***size of the template***
- ***values of coefficients***

Direct Low pass filtering by averaging

averaging 2x2

$$\begin{bmatrix} \mathbf{\frac{1}{4}} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} \end{bmatrix}$$

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & \mathbf{1} & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

averaging 3x3

Image smoothing
+ noise reduction
- edge smoothing

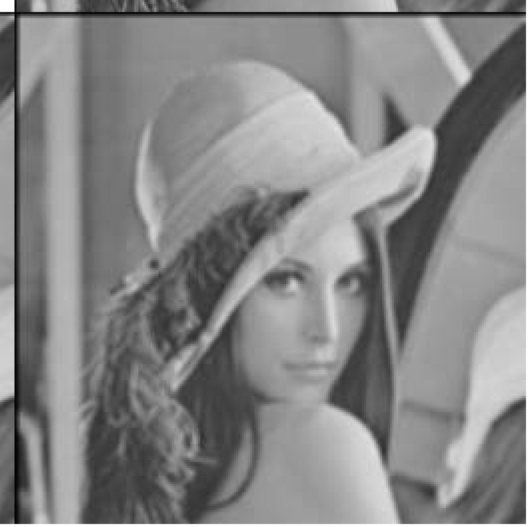
$$\sum_{p=0}^{k-1}\sum_{q=0}^{k-1} h(p,q) = 1$$

Low pass filtering by averaging

3x3

5x5

7x7



Median Filtering

Algorithm

1. $Classify\ S = \left\{ f(x_j, y_j), (x_j, y_j) \in W \right\}$

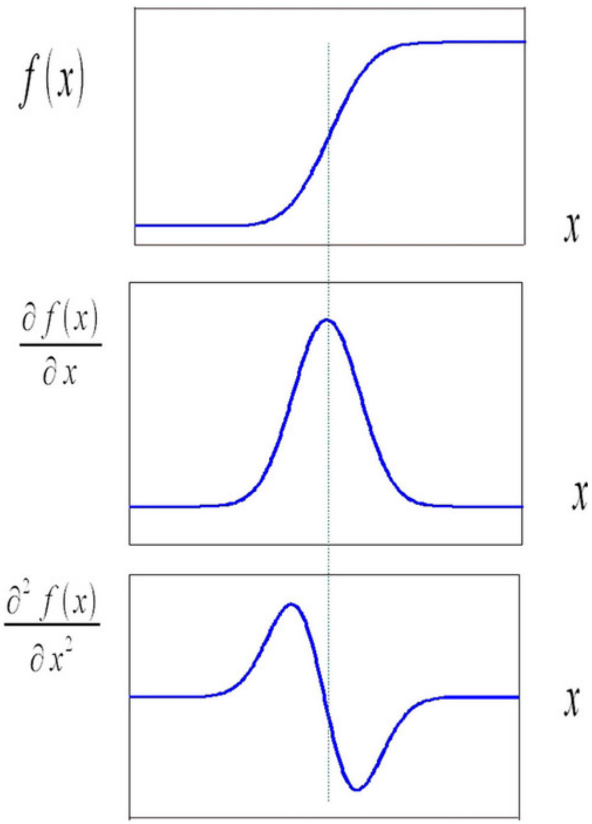
2. $f'(x_i, y_i) = med(S)$

Non linear,

$$med(\alpha.I_1 + \beta.I_2) \neq \alpha.med(I_1) + \beta.med(I_2)$$

Useful in preserving edges while reducing noise (image smoothing)

Edges : Gradient & Laplacian



$$f(x, y)$$

$$\nabla f(x, y) = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{pmatrix}$$

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

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

Image Sharpening

Gradient Pattern

- Gradient

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

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

Prewitt

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

Sobel

$if\ \left|\nabla f(x,y)\right| > Threshold\ at\ (x_0, y_0)$

$then\ (x_0, y_0)$ is an edge point

$else\ (x_0, y_0)$ is not an edge point

jean-luc.dugelay@eurecom.fr

Gradient Pattern

$$\begin{bmatrix} 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 0 & 0 & 0 & 0 & 0 \\ -10 & -10 & -10 & -10 & -10 \\ -10 & -10 & -10 & -10 & -10 \end{bmatrix}$$

0°

$$\begin{bmatrix} 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 6 & 2 \\ 8 & 4 & 0 & -4 & -8 \\ -2 & -6 & -10 & -10 & -10 \\ -10 & -10 & -10 & -10 & -10 \end{bmatrix}$$

22.5°

$$\begin{bmatrix} 10 & 10 & 10 & 10 & 0 \\ 10 & 10 & 10 & 0 & -10 \\ 10 & 10 & 0 & -10 & -10 \\ 10 & 0 & -10 & -10 & -10 \\ 0 & -10 & -10 & -10 & -10 \end{bmatrix}$$

45°

Questions

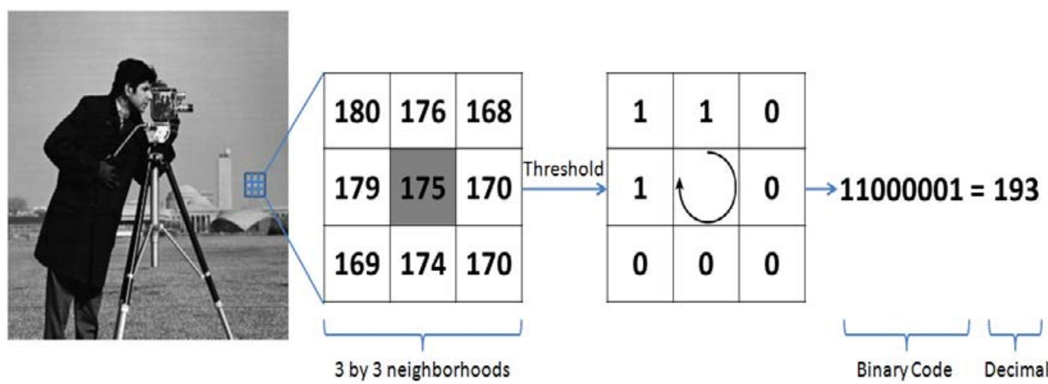
$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & \mathbf{1} & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & \mathbf{4} & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

If we apply it two times?

2D filtering or 2 times 1D?

Gradient based descriptor: SURF, FAST, DAISY, SIFT, etc.
Local binary Pattern (LBP)



$$LBP = \sum_{p=0}^{p-1} s(g_p - g_c)2^p$$

$$s(x) = \begin{cases} 1\, if \ \ x \geq 0 \\ 0, otherwise \end{cases}$$

Original Image

125	156	158	123
71	86	169	186
54	62	33	41
205	154	73	60

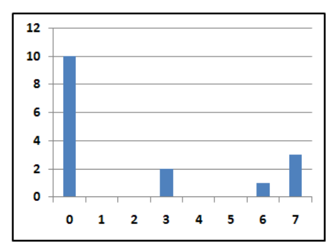
LBP codes

00010000	00011000	00001100	00000111
01110000	11110000	00010000	00000000
01111100	11101110	11111111	11000110
00000000	00000001	00000001	00000001

Quantization

000	000	000	000
011	111	000	000
011	111	111	110
000	000	000	000

Histogramization



$$\chi^2(S,M) = \sum_i^n \frac{(S_i - M_i)^2}{S_i + M_i}$$

Widely used to compare two face images

Laplacian Pattern

- Laplacian

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

Edges: zero-crossing points

$$\nabla^2 \mathrm{f}\left[\left(\mathrm{x,y}\right)\right] \approx \mathrm{f}(\mathrm{x}+1,\mathrm{y}) + \mathrm{f}(\mathrm{x}-1,\mathrm{y}) + \mathrm{f}(\mathrm{x},\mathrm{y}+1) + \mathrm{f}(\mathrm{x},\mathrm{y}-1) - 4\mathrm{f}(\mathrm{x},\mathrm{y})$$

$$\frac{\partial f}{\partial x}(x, y) \cong \frac{f(x+a, y) - f(x-b, y)}{a+b}$$

Basic question on Filtering

- *What are the differences between averaging and median filters (e.g. of size 3x3) in terms of implementation and impact?*

jean-luc.dugelay@eurecom.fr

Hough Transform

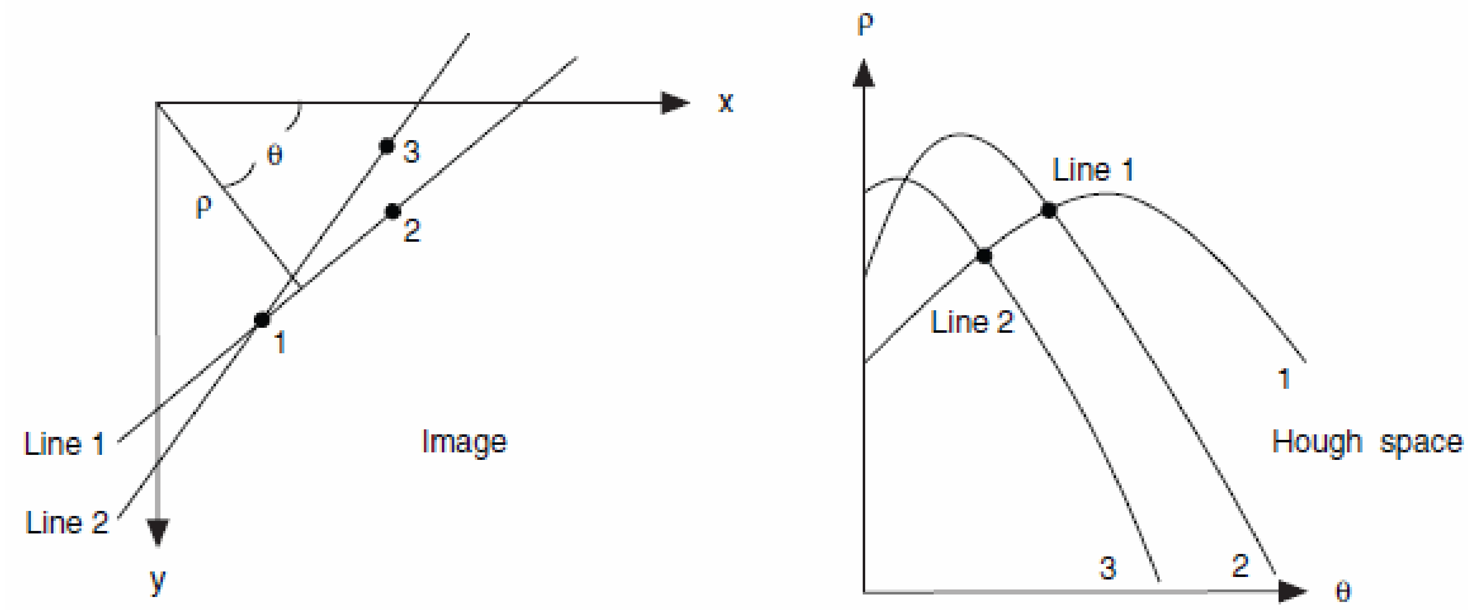
Originally designed for Line detection
but can also be used for any analytical curve (circle, ellipse, etc.)

$$\boxed{\text{IMAGE } \boldsymbol{J}} \rightleftarrows \boxed{\text{HOUGH } \boldsymbol{H}}$$

Image Space defined by $\{M_i\ (x_i, y_i)\}$
A set of pixels defines a curve described by a parametric equation,
$f(a_1, a_2, ..., a_n, x, y) = 0$
Ex.

Line $\{y = a.x + b$ or $\rho = x.\cos(\theta) + y.\sin(\theta)\}$: 2 parameters (a,b) or (ρ, θ)
Circle $\{(x - a)^2 + (y - b)^2 = c^2\}$: 3 parameters (center (a,b) and radius c, or (a, w, R))

Hough Transform (Cont.)



A point in H corresponds to a straight line in I
A point in I corresponds to a sinusoid in H

Points on the same line in I give curves passing through a common point in H
Points on the same curve in H give lines passing through a common point in I

Hough Transform (Cont.)

Several kind of transformations:

- **From m to 1**

  Any "m-uplets" from the image space is associated to a parametric curve $\{a_i\}$ in the Hough space;

- **From 1 to n**

  Any pixel (x_i,y_i) from the image space is associated to m parametric curves $\{a_i\}$ in the Hough space;

- **From m to m'**

  Any "m-uplets" from the image space is associated to m' parametric curves $\{a_i\}$ in the Hough space;

Hough Transform (Cont.)

Line detection : from m to 1

- m=2

$M_i(x_i,y_i)$ and $M_j(x_j,y_j)$

$$\rho_k = \frac{\left| x_i y_j - x_j y_i \right|}{\sqrt{(y_j - y_i)^2 + (x_j - x_i)^2}} \qquad \theta_l = -\mathrm{Arctg}\frac{x_j - x_i}{y_j - y_i}$$

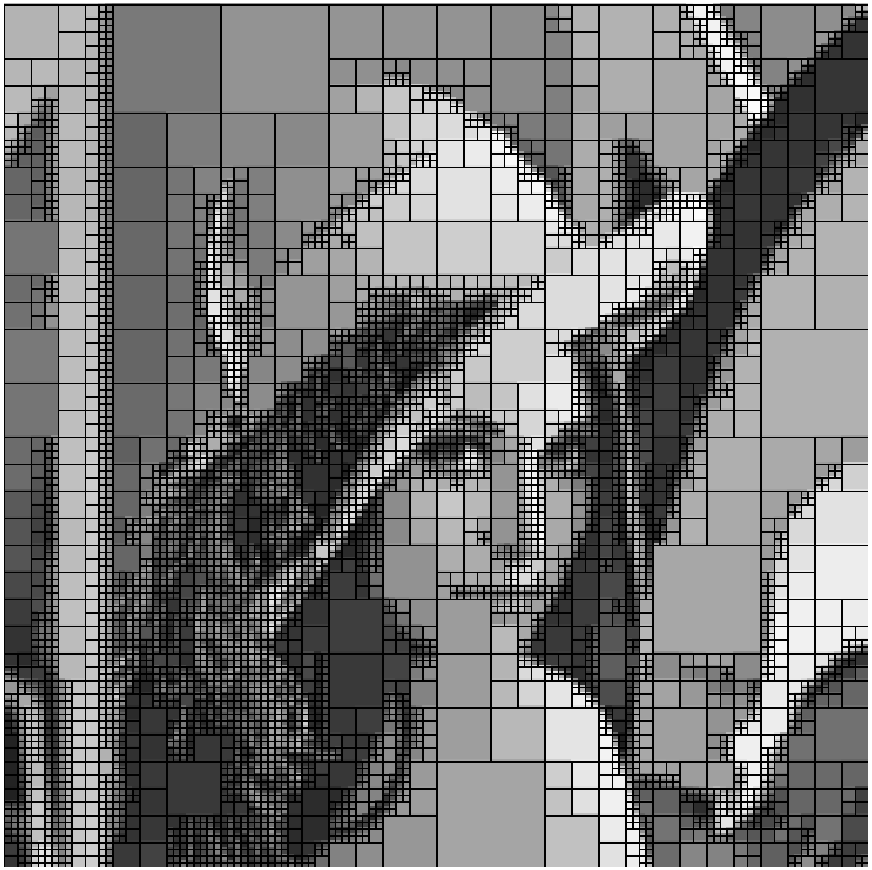
Hough Transform (Cont.)

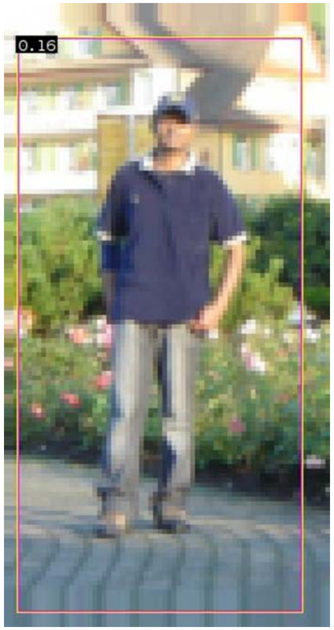
Line detection : from 1 to m

Algo.

A- Partition H into cells $ACC(\rho_k, \theta_l)$ initialized to 0;
B- For each pixel $M_i(x_i,y_i)$ in I,
do $ACC(\rho_k, \theta_l) \leftarrow ACC(\rho_k, \theta_l) + 1$ if $f(\rho_k,\theta_l,x_i,y_i) \approx 0$ is verified
C- Hence, for a given cell, that is to say $ACC(\rho_m, \theta_n)$,
its value is incremented as many times as a pixel is on the straight line (ρ_m, θ_n) ;
D- If $ACC(\rho_m, \theta_n) >$ Threshold in H, then a straight line (ρ_m, θ_n) in I is detected.

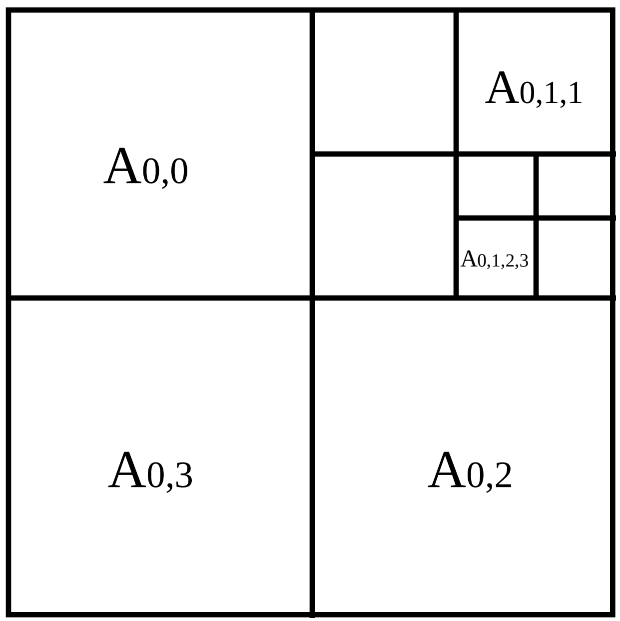
Quad tree

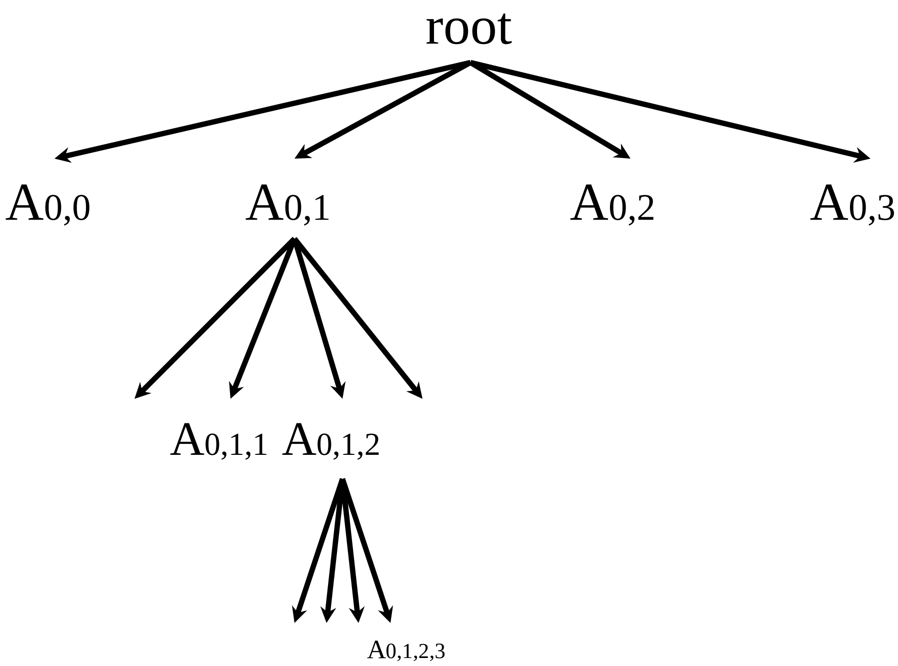




Bounding box

Quad tree





Split & Merge segmentation

- Split:

Split the picture into smaller and smaller areas until reaching a given uniformity criterion;

- Merge:

Merge neighboring areas according to a similarity criterion (mean, variance, etc.);

- Split & Merge:

Combination of both previous approaches

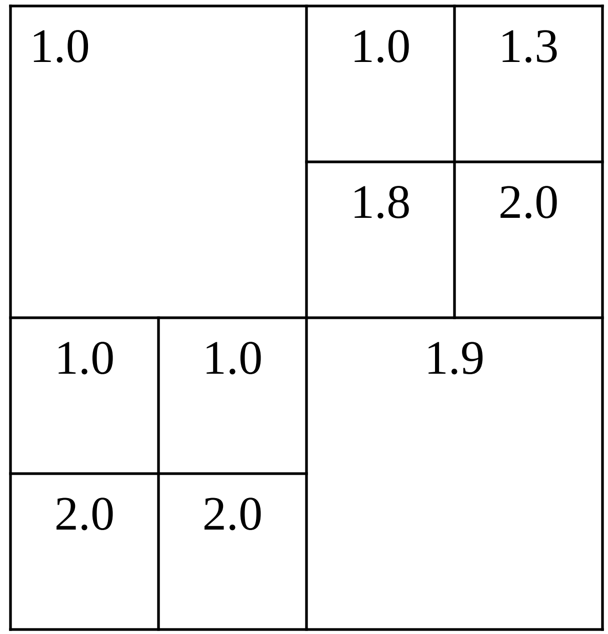
jean-luc.dugelay@eurecom.fr

Split

1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	2
1	1	1	1	1	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	1	2	2	2
1	1	1	1	1	2	2	2
2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2

$$TEST : E(R_l) = \frac{1}{card[R_l]} \sum_{x_i \in R_l} \left(g(x_i) - m(R_l) \right)^2 < Threshold\_1$$

Split



$$TEST : E(R_l) = \frac{1}{card[R_l]} \sum_{x_i \in R_l} \left(g(x_i) - m(R_l)\right)^2 < Threshold\_1$$

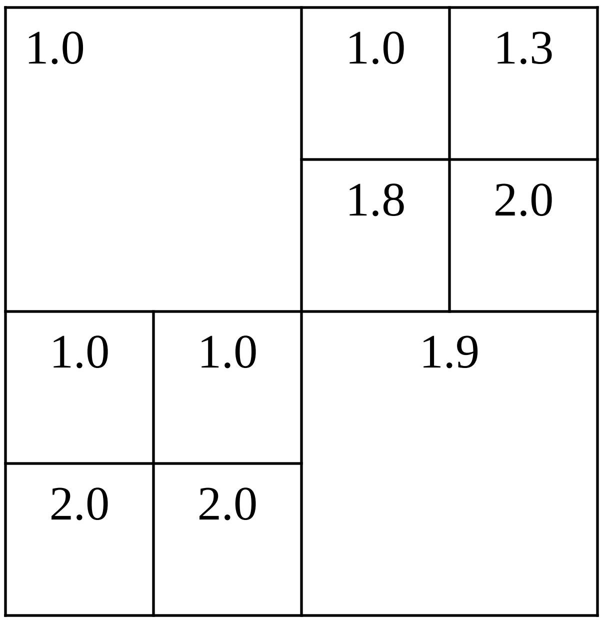
Split

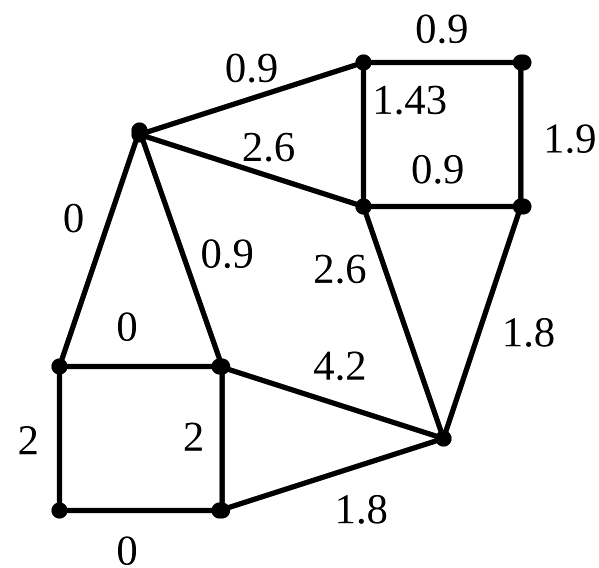
0	0	0	0	0	0	.3	.3
0	0	0	0	0	0	.3	.7
0	0	0	0	.8	.2	0	0
0	0	0	0	.2	.2	0	0
0	0	0	0	.9	.1	.1	.1
0	0	0	0	.9	.1	.1	.1
0	0	0	0	.1	.1	.1	.1
0	0	0	0	.1	.1	.1	.1

$$TEST : E(R_l) = \frac{1}{card[R_l]} \sum_{x_i \in R_l} \left(g(x_i) - m(R_l) \right)^2 < Threshold\_1$$

Merge

Region adjacency graph





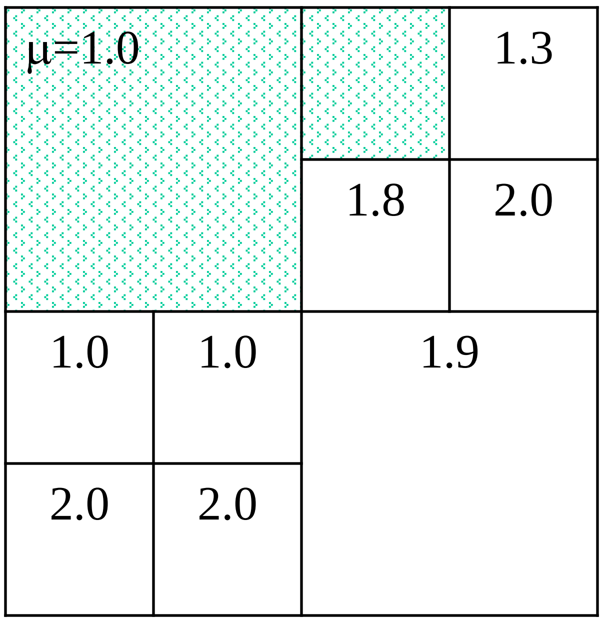
$$E(R_l, R_k) = \sum_{x_i \in R_l \cup R_k} \left(g(x_i) - \mu(R_l, R_k) \right)^2$$

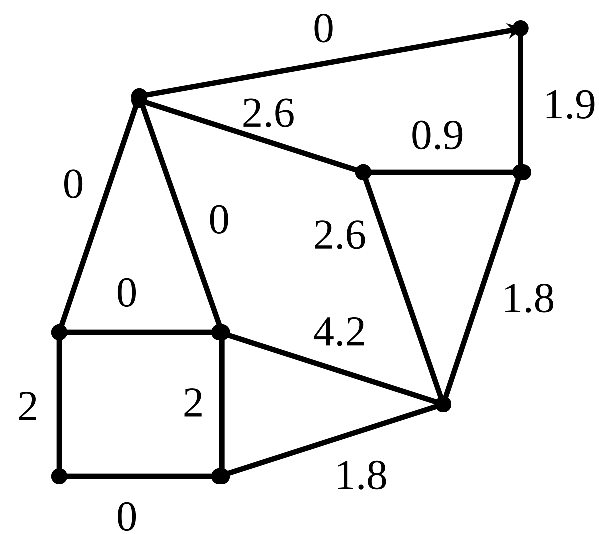
$$\mu(R_l, R_k) = \frac{1}{\mathrm{card}\{R_l \cup R_k\}} \sum_{x_i \in R_l \cup R_k} g(x_i)$$

jean-luc.dugelay@eurecom.fr

Merge

Region adjacency graph





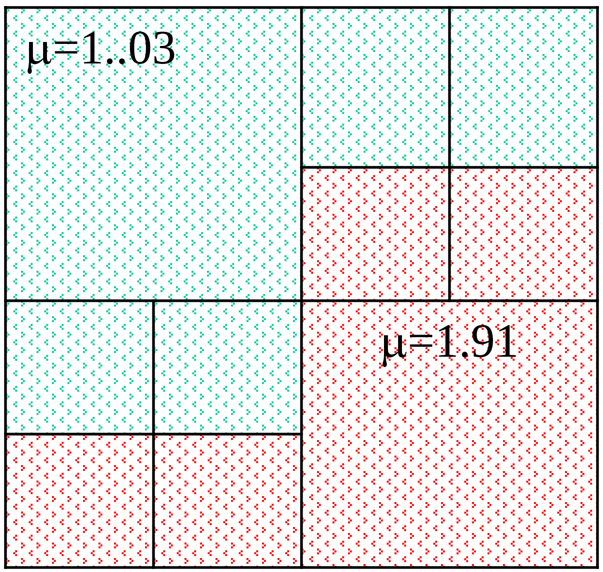
$$E(R_l, R_k) = \sum_{x_i \in R_l \cup R_k} \left(g(x_i) - \mu(R_l, R_k) \right)^2$$

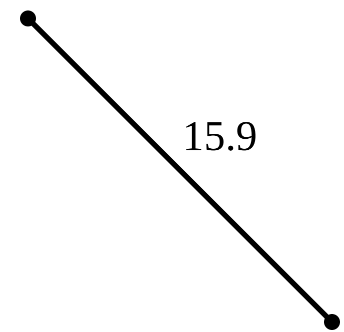
$$\mu(R_l, R_k) = \frac{1}{\mathrm{card}\{R_l \cup R_k\}} \sum_{x_i \in R_l \cup R_k} g(x_i)$$

$$SEQ(L) = \sum_{l=1}^{L} \sum_{x_i \in R_l} \left(g(x_i) - \mu(R_l) \right)^2 > Threshold\_2$$

Merge

Region adjacency graph

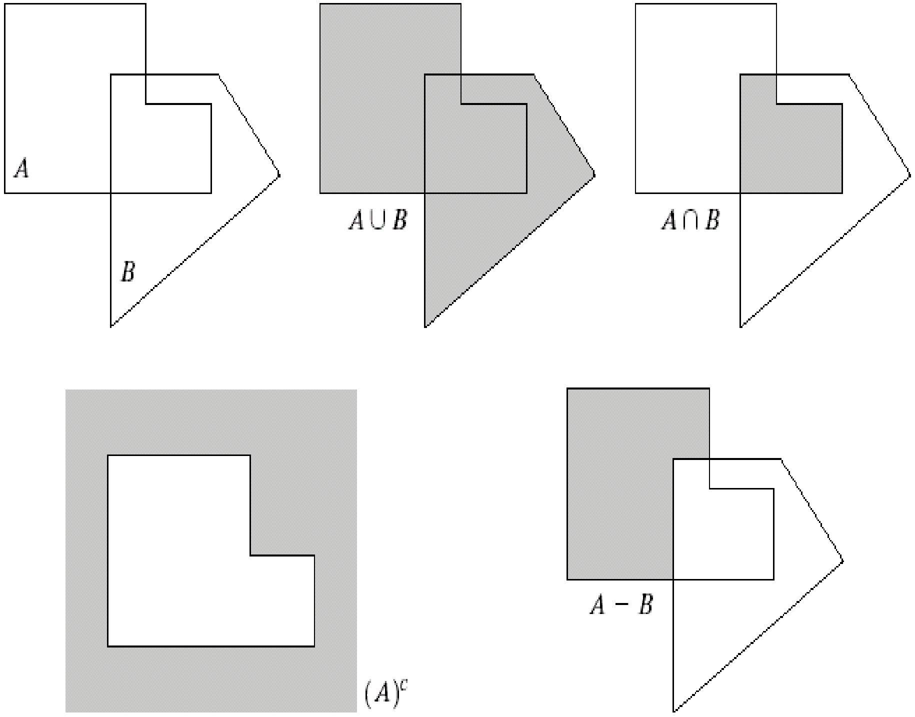




$$E(R_l, R_k) = \sum_{x_i \in R_l \cup R_k} \left(g(x_i) - \mu(R_l, R_k) \right)^2$$

$$\mu(R_l, R_k) = \frac{1}{card\{R_l \cup R_k\}} \sum_{x_i \in R_l \cup R_k} g(x_i)$$

Math. Morpho.: Intro.



union

intersection

complementary

inclusion

difference

Math. Morpho.: Basic operations

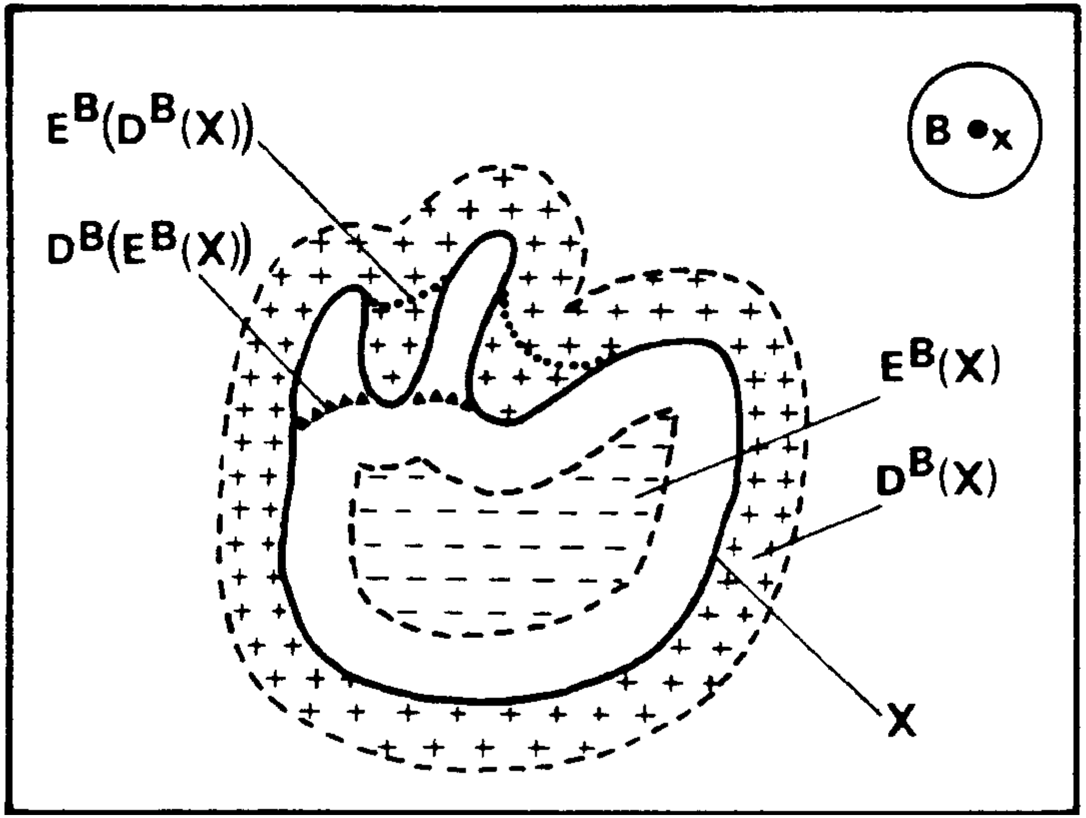
- Object A
- Structuring element B_p

- EROSION $er(A, B_p) \equiv \{p \mid B_p \subset A\}$
- DILATATION $dil(A, B_p) \equiv \{p \mid B_p \cap A \neq \varnothing\}$

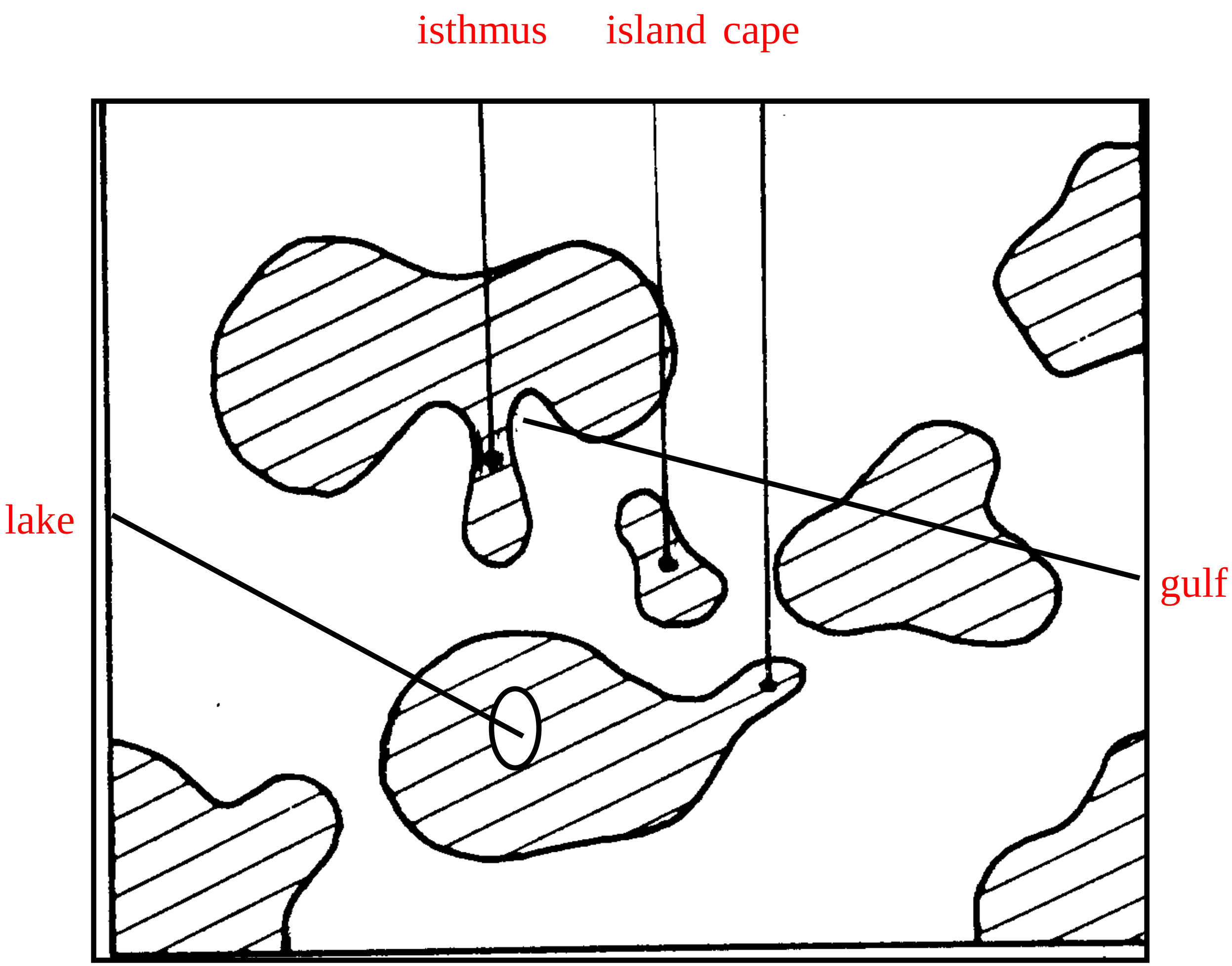
- OPENING dil(er(A,B_p),B_p)
- CLOSING er(dil(A,B_p),B_p)

jean-luc.dugelay@eurecom.fr

Math. Morpho.: Some properties (Cont.)



Math. Morpho.: Intro.(Cont.)



Math. Morpho.: Some properties

- Translation invariant
- Erosion & dilatation are not inverses of each other

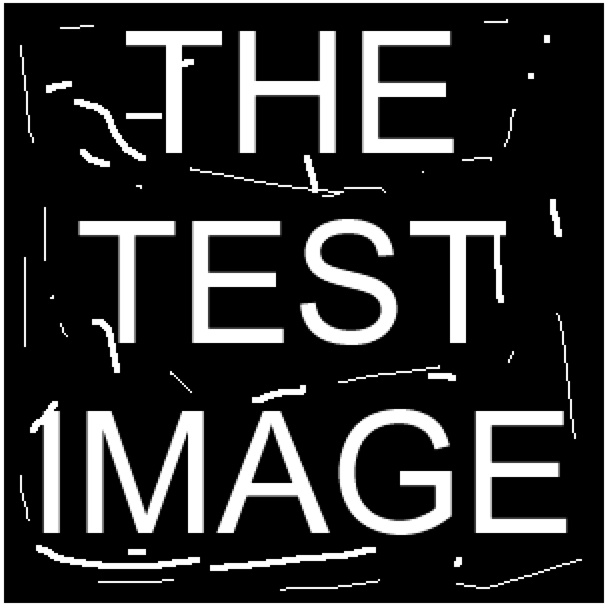
Duality: $\mathrm{dil}(A, B_p) = \left[\mathrm{er}(A^c, B_p)\right]^c$

- Increasing: X ⊂ X’ => T(X) ⊂ T(X’)
- ...

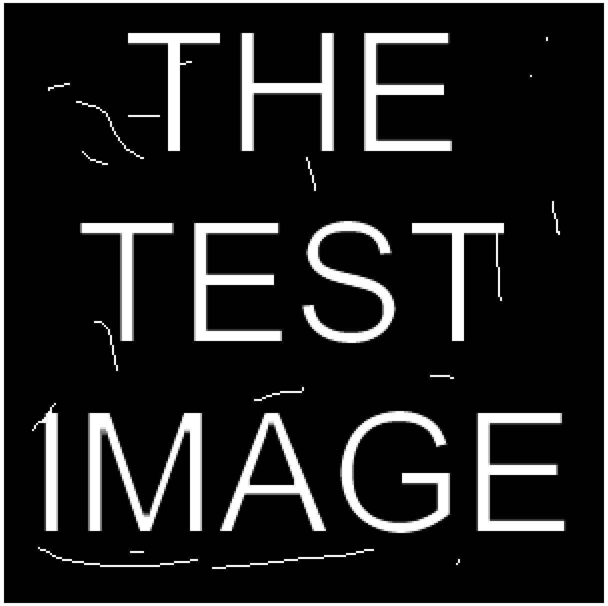
Opening: cuts narrow isthmus;
removes small islands and narrow capes.
Closing: fills narrow canals;
removes small lakes and narrow gulfs.

It is more severe to erode prior to dilate than the contrary.

Erosion

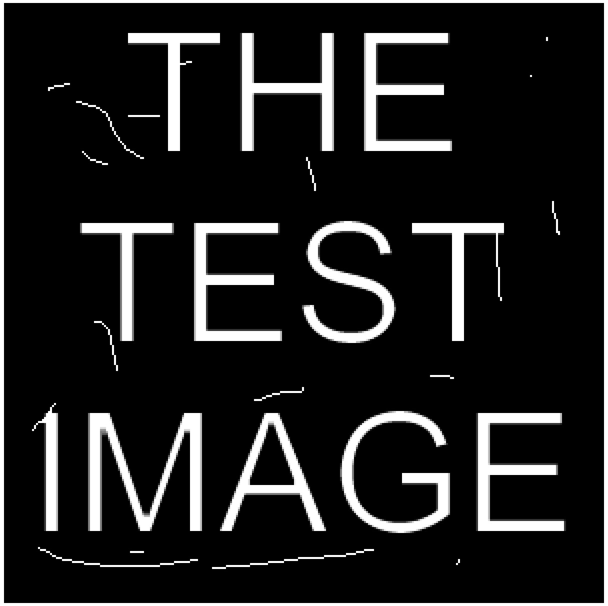


Original image



Eroded image

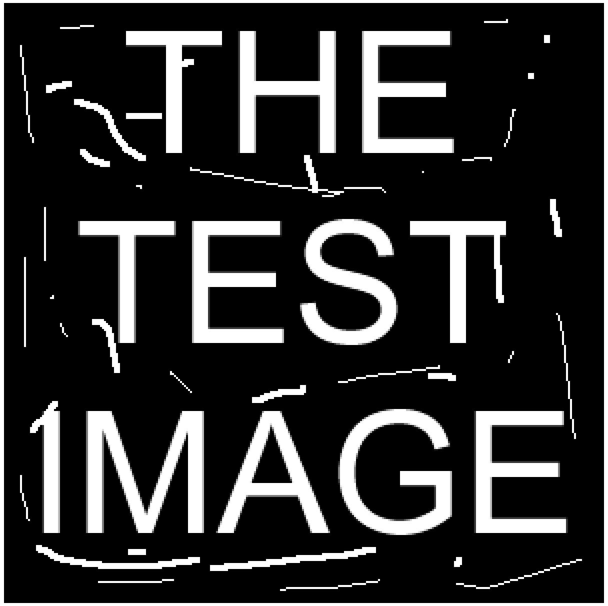
Erosion

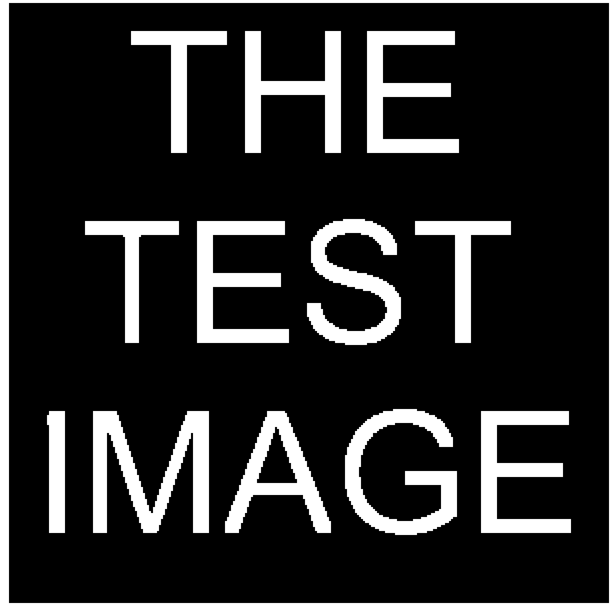


Eroded once

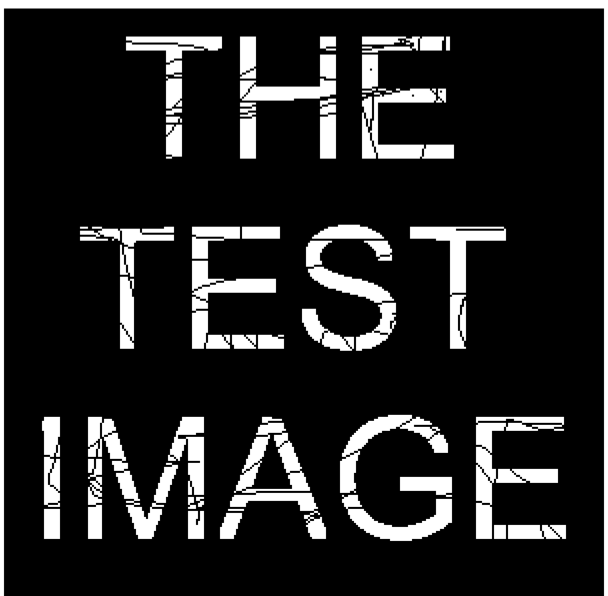
Eroded twice

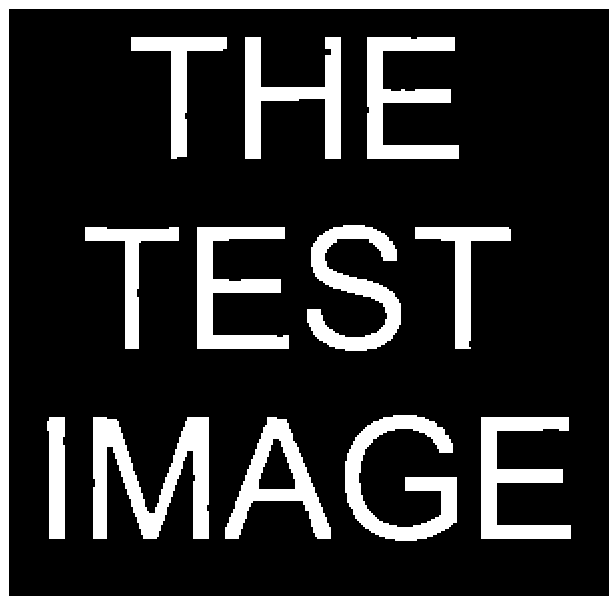
Opening and Closing





OPENING: The original image eroded twice and dilated twice (opened). Most noise is removed



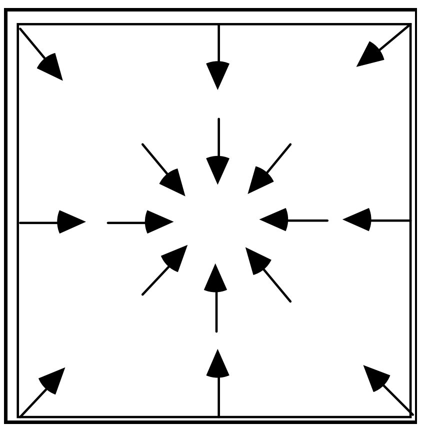


CLOSING: The original image dilated and then eroded. Most holes are filled.

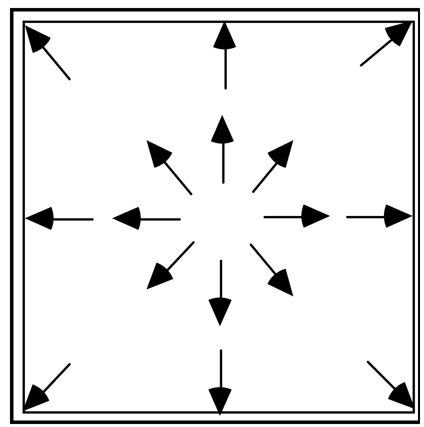
Optical Flow, or apparent motion field

relative motion between an observer (an eye or a camera) and the scene

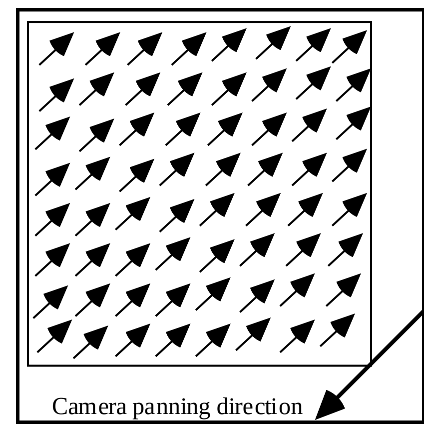
Motion vector pattern
resulting from camera panning and zooming



zoom out



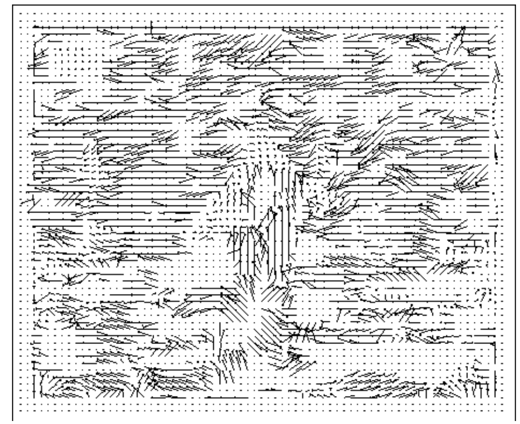
zoom in



Camera panning and tilting

jean-luc.dugelay@eurecom.fr

http://extra.cmis.csiro.au/IA/changs/motion/



jean-luc.dugelay@eurecom.fr

Apparent motion

Optical flow

between two consecutive image frames taken at t and t'(t +δt)

STEP 1. Local Estimation:

displacement: $d_i\left(d_{x_i}, d_{y_i}\right)$ *for each pixel:* $p_i(x_i, y_i)$

$$\begin{cases} d_{x_i} \equiv x'_i - x_i \\ d_{y_i} \equiv y'_i - y_i \end{cases}$$

Apparent motion (Cont.)

$$\{z\_f, pan_x, pan_y\}:$$

STEP 2. Global Interpretation

$$\forall i, \begin{pmatrix} x'_i \\ y'_i \end{pmatrix} = z\_f. \begin{pmatrix} x_i \\ y_i \end{pmatrix} + \begin{pmatrix} pan_x \\ pan_y \end{pmatrix}$$

parameter	**name**	**interpretation**
Z_f	***zoom factor***	IF z_f is closed to 0 THEN no zoom ELSE Z_f>> 1 => backward zoom Z_f<< 1 => forward zoom
t_x	***vertical pan***	mean vertical displacement in pixels of the whole image
t_y	***horizontal pan***	mean horizontal displacement in pixels of the whole image

jean-luc.dugelay@eurecom.fr

Estimation of the Optical Flow

Hypothesis: the luminance of a pixel is constant over time.

$$I(x, y, t) = I(x+\delta x, y+\delta y, t+\delta t)$$

Assuming that δx and δy are smalls

I(x,y,t) with Taylor series can be developed to get:

$$I(x+\delta x, y+\delta y, t+\delta t) = I(x, y, t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t + H.O.T.$$

$$\frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t = 0 \qquad \frac{\partial I}{\partial x}Vx + \frac{\partial I}{\partial y}Vy = -\frac{\partial I}{\partial t}$$

- Technique based on a relation between spatial and temporal gradients

(i.e. derivatives of the image at (x,y,t)

- 1 equation – 2 unknowns
- Compute V_x and V_y via an iterative process:

$$V^{(i)} = V^{(i-1)}(p,t) - \varepsilon.DFD(p,t,V^{(i-1)})\nabla I(x - V_x^{(i-1)}, y - V_y^{(i-1)}, t-1)$$

jean-luc.dugelay@eurecom.fr

$$d^{(i)} = d^{(i-1)}(p,t) - \varepsilon.sign\{DFD(p,t,d^{(i-1)})\}.sign\{\nabla I(x-d_x^{(i-1)}, y-d_y^{(i-1)}, t-1)\}$$

Optical flow: basic illustration

(iteration #1)

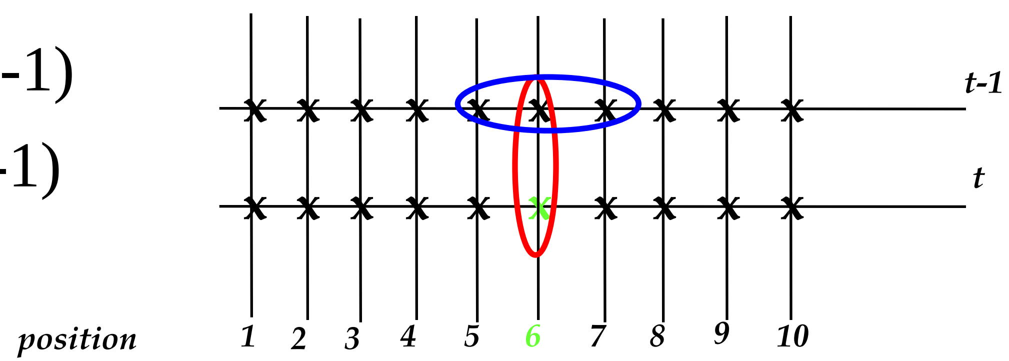
Pixel #6 at time t (from which position at t-1 does this pixel come from ?)

Assumption: $d^{(0)} = 0$

Gradients

Temporal I(6,t) - I(6,t-1)

Spatial I(7,t-1) - I(5,t-1)



Gain: $\varepsilon = 1$

Temporal > 0

Spatial > 0

$$d^{(i)} = d^{(i-1)}(p,t) - \varepsilon.\mathrm{sign}\left\{DFD(p,t,d^{(i-1)})\right\}.\mathrm{sign}\left\{\nabla I(x - d_x^{(i-1)}, y - d_y^{(i-1)}, t-1)\right\}$$

Optical flow: basic illustration (Cont.)

(iteration #2)

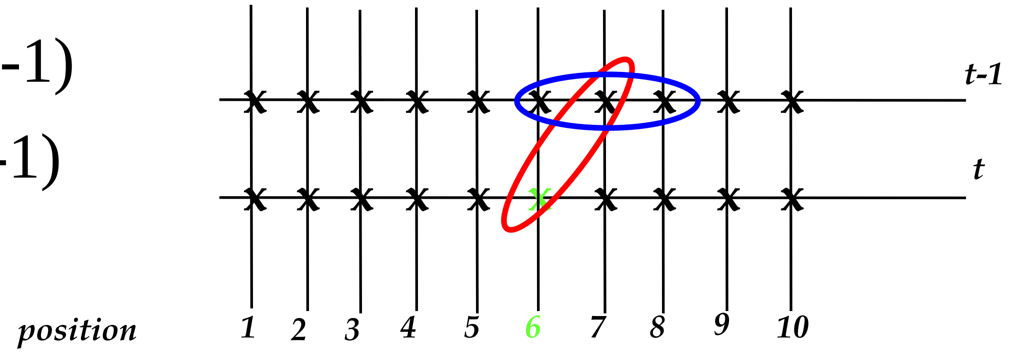
Pixel #6

$d^{(1)} = d^{(0)} - 1 = -1$

Gradients

Temporal I(6,t) - I(7,t-1)

Spatial I(8,t-1) - I(6,t-1)



jean-luc.dugelay@eurecom.fr