Image descriptors

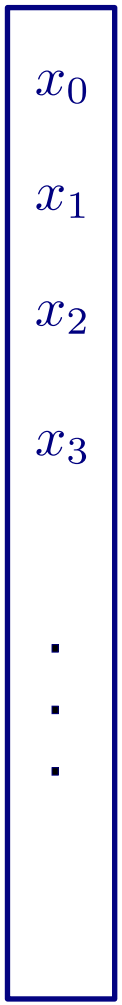
Diane Lingrand

UNIVERSITÉ
CÔTE D'AZUR
Master Data Science M1

2020 - 2021

Problematic

how to describe an image



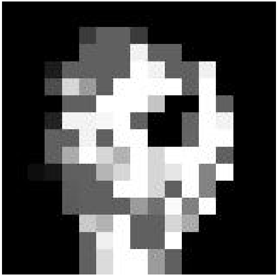
classification algorithm

class

Menu

- Basics on image processing
  - color, edges, filtering, points
- Image descriptors
  - SIFT, SURF, HOG
- Deep features

Very basic idea : image as a pixel array



0.alexandre.hiltcher.009.txt

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 11, 30, 30, 11, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 30, 30, 30, 227, 30, 30, 0, 0, 0, 0, 0, 0, 0, 0, 3, 30, 30, 30, 30, 255, 227, 30, 30, 223, 0, 0, 0, 0, 0, 0, 19, 171, 83, 30, 255, 255, 255, 255, 30, 255, 0, 0, 0, 0, 0, 0, 0, 30, 30, 255, 255, 255, 141, 0, 0, 255, 30, 0, 0, 0, 0, 0, 4, 227, 227, 237, 255, 30, 0, 0, 32, 255, 227, 0, 0, 0, 0, 0, 30, 30, 227, 30, 255, 255, 0, 0, 33, 255, 255, 0, 0, 0, 0, 0, 30, 104, 255, 171, 114, 255, 171, 255, 255, 255, 30, 0, 0, 0, 0, 1, 2, 30, 30, 118, 171, 255, 171, 202, 255, 58, 255, 0, 0, 0, 0, 0, 0, 27, 30, 30, 255, 255, 171, 30, 255, 58, 0, 0, 0, 0, 0, 0, 0, 26, 30, 30, 30, 83, 41, 244, 55, 255, 0, 0, 0, 0, 0, 0, 0, 0, 149, 114, 255, 30, 30, 171, 104, 0, 0, 0, 0, 0, 0, 0, 0, 0, 30, 223, 255, 30, 30, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 30, 181, 255, 255, 84, 30, 0, 0, 0, 0, 0]

Color images

- RGB = Red, Green, Blue. Usually : 1 byte per component. The most common.
- YUV and variants (linear transformation)
- HSV = Hue Saturation Value
- CIE LAB

Composite representation of video

$$Y = 0.299\ R + 0.587\ G + 0.114\ B$$

- Betacam (Y pb pr)
  - pb = 0.5 (B – Y) / (1 – 0.114)
  - pr = 0.5 (R – Y)/ (1 – 0.299)
- Digital Video System (Y Cb Cr)
  - Cb = 128 + 112 pb
  - Cr = 128 + 112 pr
- YUV
  - U = 0.493 (B – Y)
  - V = 0.877 (R – Y)
- NTSC YIQ
  - I = 0.6 R – 0.28 G – 0.32 B
  - Q = 0.21 R – 0.52 G + 0.31 B

- Hunter color space(1948) : Lab
- CIELAB (1976) : L*a*b*
  - L* : lightness (cubic root of relative luminance)
  - a* and b* : distance to grey
  - a* : from green to red
  - b* : from blue to yellow

$$\begin{aligned} L^\star &= 116 f(Y/Y_n) - 16 \\ a^\star &= 500(f(X/X_n) - f(Y/Y_n)) \\ b^\star &= 200(f(Y/Y_n) - f(Z/Z_n)) \end{aligned}$$

with

$$f(t) = \begin{cases} t^{1/3} & \text{if } t > \left(\frac{6}{29}\right)^3 \\ \frac{1}{3}\left(\frac{29}{6}\right)^2 t + \frac{4}{29} & \text{else} \end{cases}$$

Using illumination D65, coordinates of white :
$X_n = 95.047, Y_n = 100.000, Z_n = 108.883$

HSV model : *Hue Saturation Value*

$$V = max(R, G, B)$$

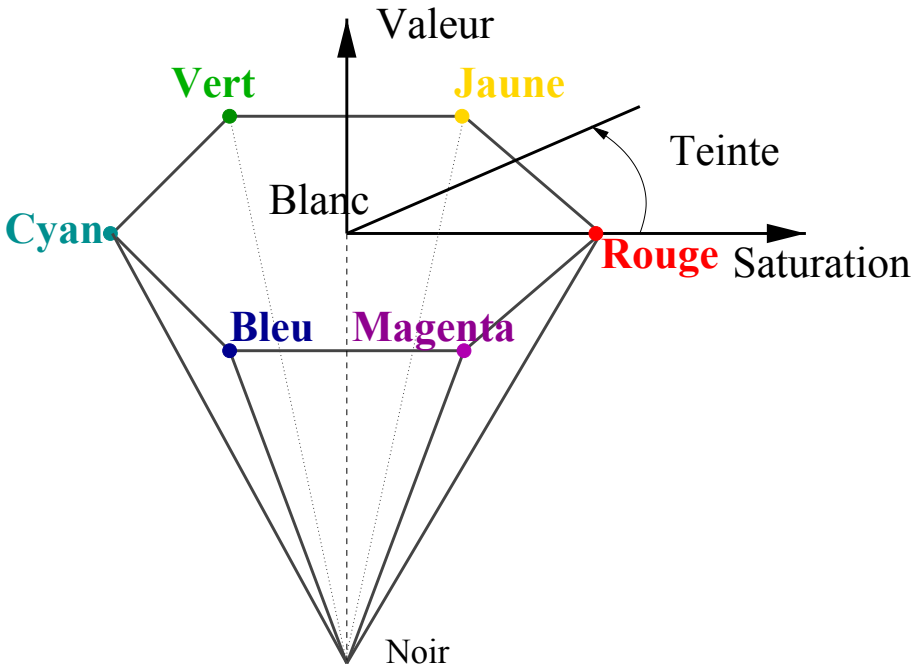
$$S = \begin{cases} \frac{V - min(R,G,B)}{V} & \text{if } V \neq 0 \\ 0 & \text{else} \end{cases}$$

$$H = \begin{cases} \frac{60(G-B)}{V - min(R,G,B)} & \text{if } V = R \\ \frac{120 + 60(B-R)}{V - min(R,G,B)} & \text{if } V = G \\ \frac{240 + 60(R-G)}{V - min(R,G,B)} & \text{if } V = B \end{cases}$$

If $H < 0$ then $H = H + 360$.

Output :$0 \leq V \leq 1$, $0 \leq S \leq 1$,

$0 \leq H \leq 360$

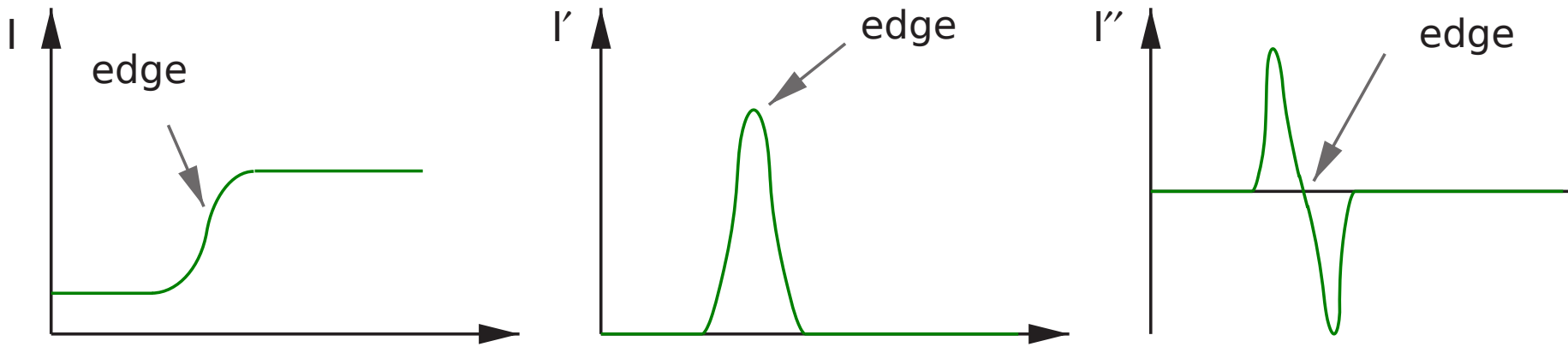


How to describe a region

- main color
- color histogram
- texture (Gabor filters, Local Binary Pattern)
- edges, corners

Edge detectors

- maximum of derivatives
- derivatives sensitive to noise $\Rightarrow$ need to smooth



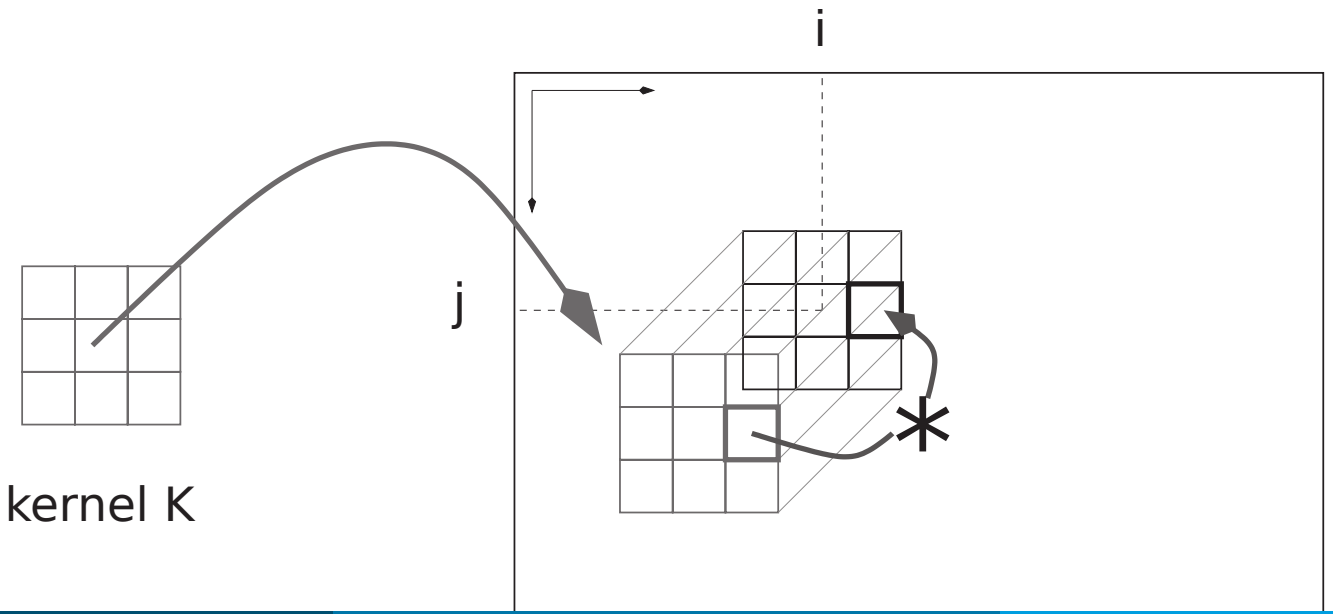
Convolution

- For f and g discrete functions :

$$f * g(x, y) = \sum_{u=-\infty}^{\infty} \sum_{v=-\infty}^{\infty} f(x, y) g(u - x, v - y)$$

- Convolution of an image I_1 by a kernel K of dimensions $(2p + 1)x(2q + 1)$:

$$I_2[i][j] = \sum_{k=0}^{2p} \sum_{l=0}^{2q} I_1[i - k + p][j - l + q] K[k][l]$$



Average filter smoothing

Dimension 3x3 :

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

Dimension (2p+1)x(2p+1) :

$$\begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ & . & & .. \\ 1 & 1 & \dots & 1 \end{bmatrix} /(2p+1)^2$$

3x3

5x5

7x7

9x9

Gaussian smoothing

$$G_\sigma(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{(-\frac{|x|^2}{2\sigma^2})}$$

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix} /8 \quad \text{or} \quad \begin{bmatrix} 1 & 1 & 1 \\ 1 & 8 & 1 \\ 1 & 1 & 1 \end{bmatrix} /16$$

3x3

5x5

7x7

Sobel detection (1970)

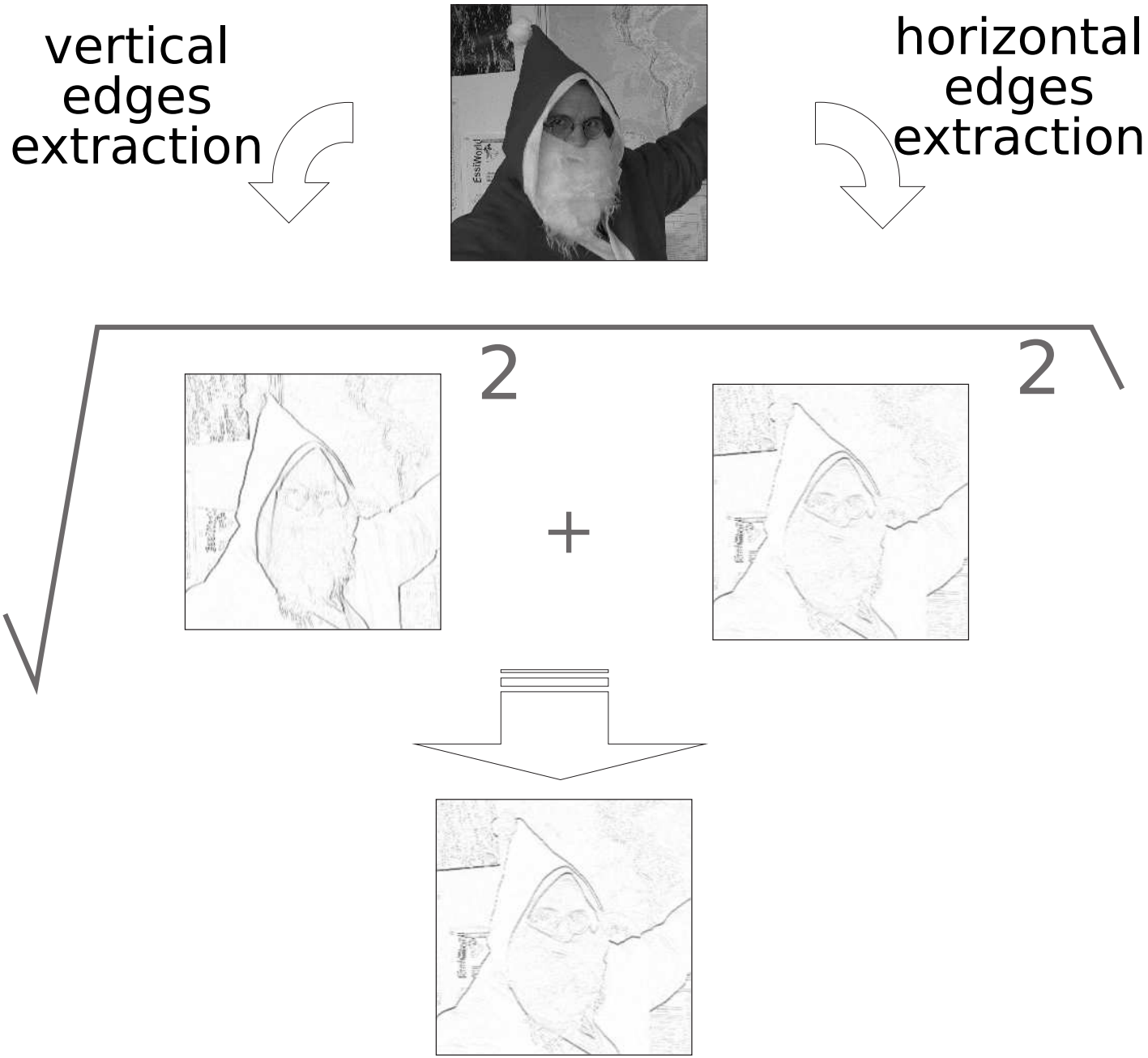
combines smoothing in one direction $[1, 2, 1]$ and derivative in a perpendiculare direction using $[1, 0, -1]$

$$\frac{dI}{dx} = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} /4 \quad \frac{dI}{dy} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} /4$$

Angle between horizontal and normal vector :

$$\arctan\left(\left(\frac{dI}{dy}\right) / \left(\frac{dI}{dx}\right)\right)$$

Example of Sobel detection



Prewitt edge detector(1970)

$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} /3, \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{bmatrix} /3, \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} /3, \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix} /3$$

Kirsch edge detector(1971)

$$\begin{bmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{bmatrix} /15, \begin{bmatrix} 5 & 5 & -3 \\ 5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} /15, \begin{bmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} /15,$$

$$\begin{bmatrix} -3 & 5 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & -3 \end{bmatrix} /15, \begin{bmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{bmatrix} /15, \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & 5 \\ -3 & 5 & 5 \end{bmatrix} /15,$$

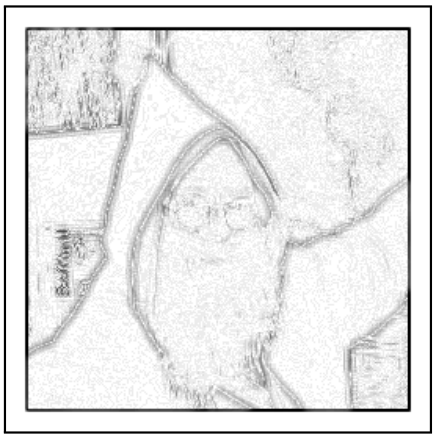
$$\begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & -3 \\ 5 & 5 & 5 \end{bmatrix} /15, \begin{bmatrix} -3 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & 5 & -3 \end{bmatrix} /15$$

Laplace by example

Laplace connexity 4

Laplace connexity 8

Laplace connexity 8
after gaussian smoothing (7x7)



Zeros crossings

Laplacian as difference between two gaussians.

Difference between gaussian smoothing 7x7 and 5x5 :

Corners detection

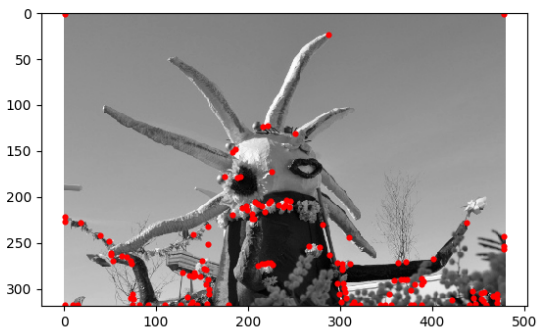
- many approaches (edges points linkage, curvature, models, ...)
- the more famous : Harris and Stephens, 1988

$$\mathcal{O} = \det(\widehat{C}(x,y)) - k\,(\text{trace}(\widehat{C}(x,y)))^2$$

$$\widehat{C}(x,y) = \begin{pmatrix} \widehat{(\frac{\partial I(x,y)}{\partial x})^2} & \widehat{\frac{\partial I(x,y)}{\partial x}\frac{\partial I(x,y)}{\partial x}} \\ \widehat{\frac{\partial I(x,y)}{\partial x}\frac{\partial I(x,y)}{\partial x}} & \widehat{(\frac{\partial I(x,y)}{\partial y})^2} \end{pmatrix} \text{ with } k = 0.04$$

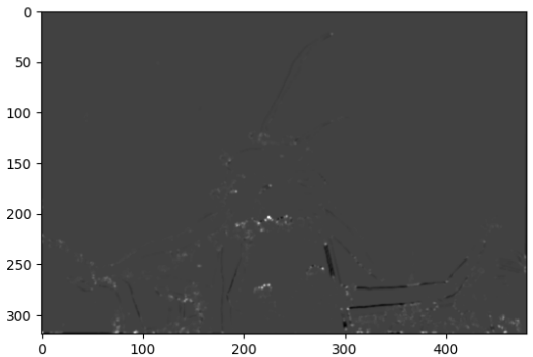
$\widehat{\ }$: gaussian smoothing

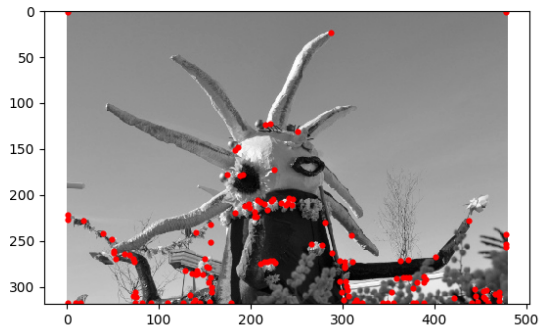
- computation of first derivatives with Gaussian smoothing
- compute $\mathcal{O}$
- thresholding



Harris example

```
from skimage.feature import corner_harris, corner_peaks
img = skimage.io.imread('carnaval4.jpg', as_gray=True)
pts=corner_peaks(corner_harris(img), min_distance=1)
print( pts.shape[0], ' points found')
import matplotlib.pyplot as plt
plt.imshow(img, cmap='gray')
plt.scatter(y=pts[:,0],x=pts[:,1],c='r',s=10)
plt.show()
```



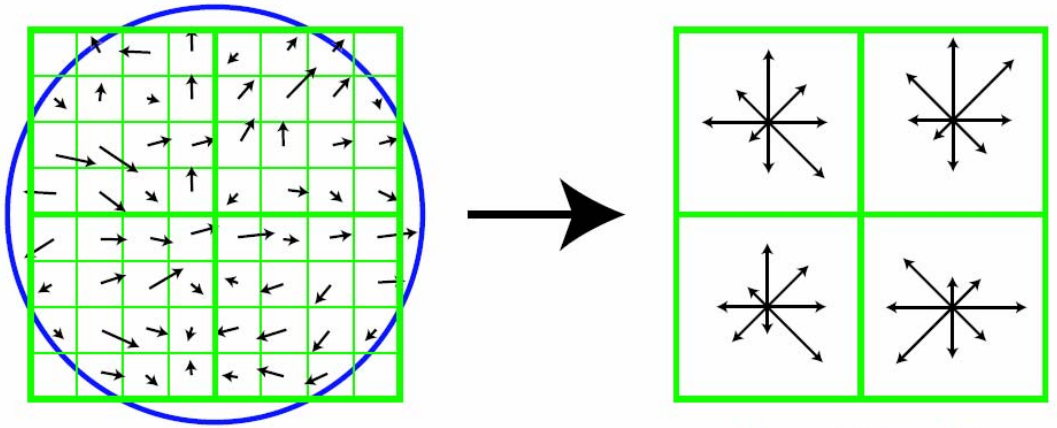


UNIVERSITÉ CÔTE D'AZUR

SIFT = Scale Invariant Feature Transform

- Detector
  - multi-scale
  - DOG laplacian (Difference Of Gaussians)
- Descriptor
  - edges orientations in the neighborhood

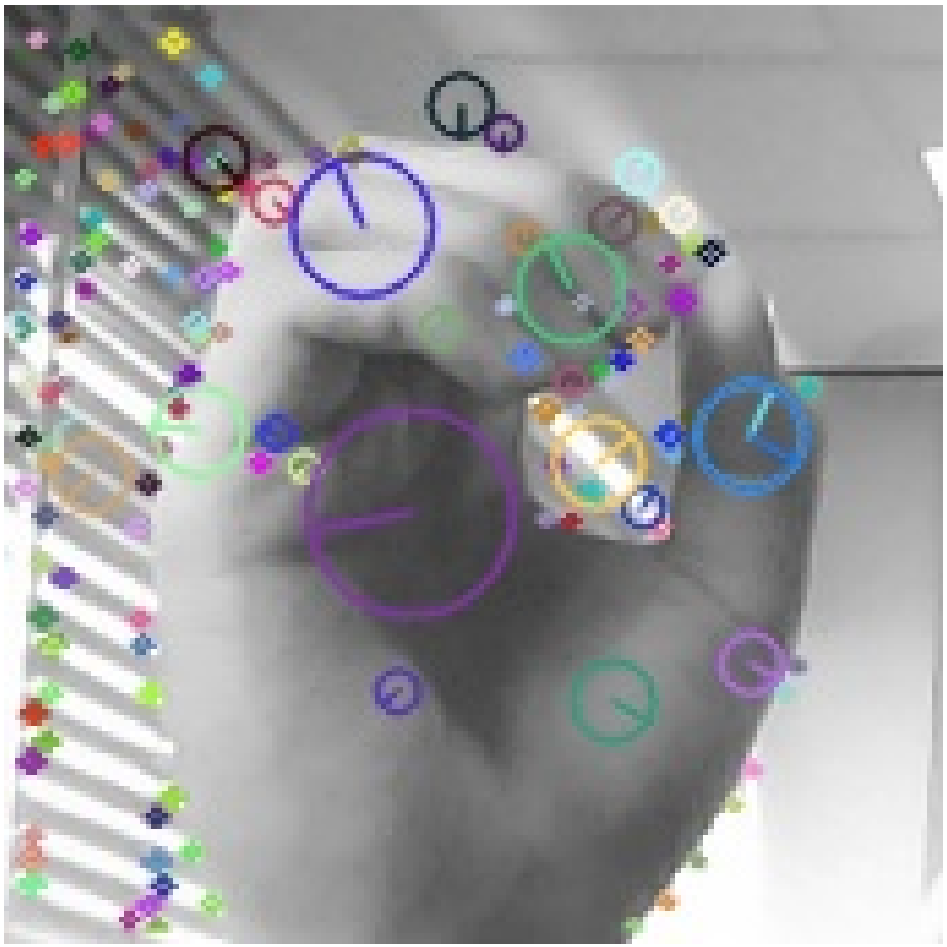
SIFT descriptor



- vectors of 128 integers
- 4 steps :
  - interest points detection
  - gradients orientation in the neighborhood (16x16 pixels divided in 16 blocks of size 4x4)
  - orientation histogram (quantified on 8 values in blocks of 4x4 pixels)
    - $8x4 * 4 = 128$
  - normalisation

SIFT descriptor

`0.alexandre.hiltcher.006.png`
252 descriptors



`0.alexandre.hiltcher.009.png`
182 descriptors

SIFT implementation in OpenCV

UNIVERSITÉ CÔTE D'AZUR

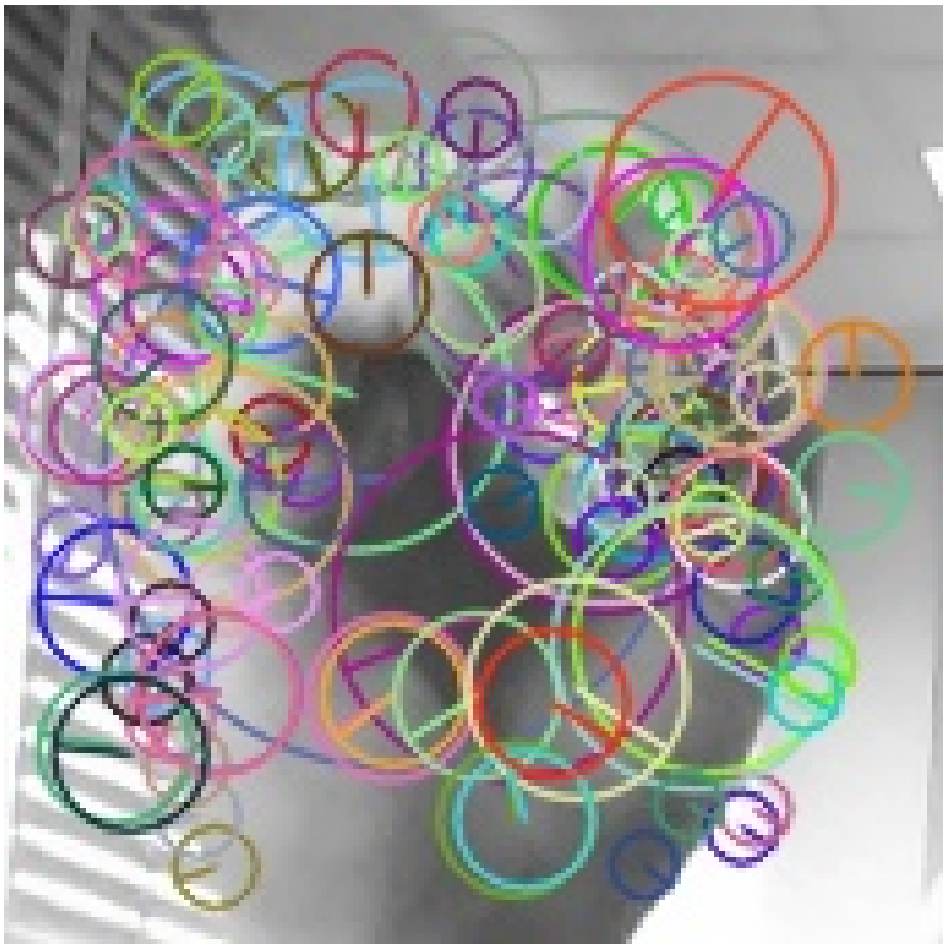
```
import cv2
import numpy as np
img = cv2.imread('carnaval.jpg')
gray= cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
sift = cv2.xfeatures2d.SIFT_create()
kp = sift.detect(gray,None)
cv2.drawKeypoints(gray,kp,img,\\
    flags=cv2.DRAW_MATCHES_FLAGS_DRAW_RICH_KEYPOINTS)
cv2.imwrite('carnavalSIFT.jpg',img)
```

UNIVERSITÉ CÔTE D'AZUR

SURF = Speeded-Up Robust Features

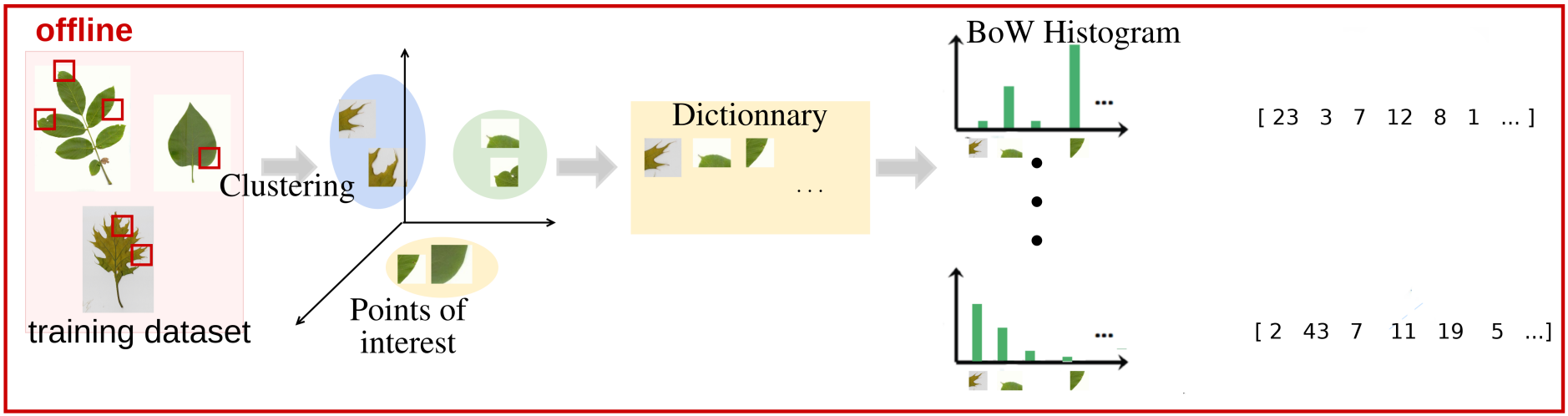
default size : 64

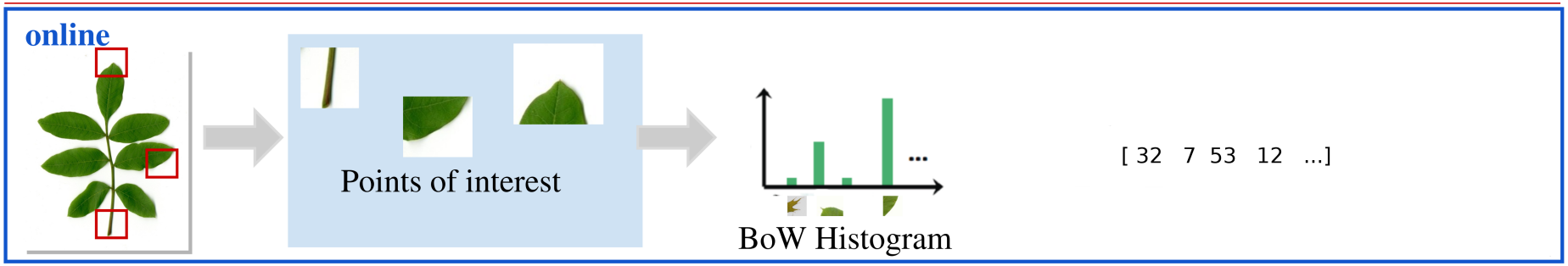
`0.alexandre.hiltcher.006.png`
184 descriptors



`0.alexandre.hiltcher.009.png`
121 descriptors

Bag Of Words (BOW)



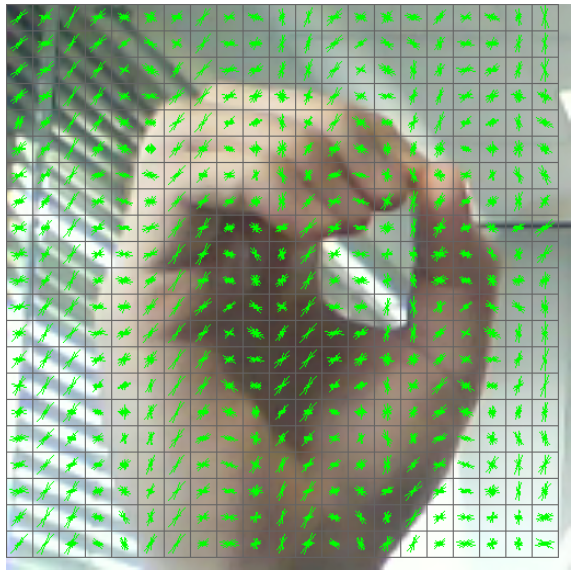


UNIVERSITÉ CÔTE D'AZUR

HOG : Histogram of Gradients

- gradient computation ([-1 0 1] et [-1 0 1] T)
- histogram construction
  - squared cells (from 4x4 to 12x12 pixels)
  - discretisation on 9 angle values
  - pixel votes proportional to gradient amplitude
- blocks construction
  - 1 block = several cells
  - normalisation of blocks
- HOG = concatenation of histograms

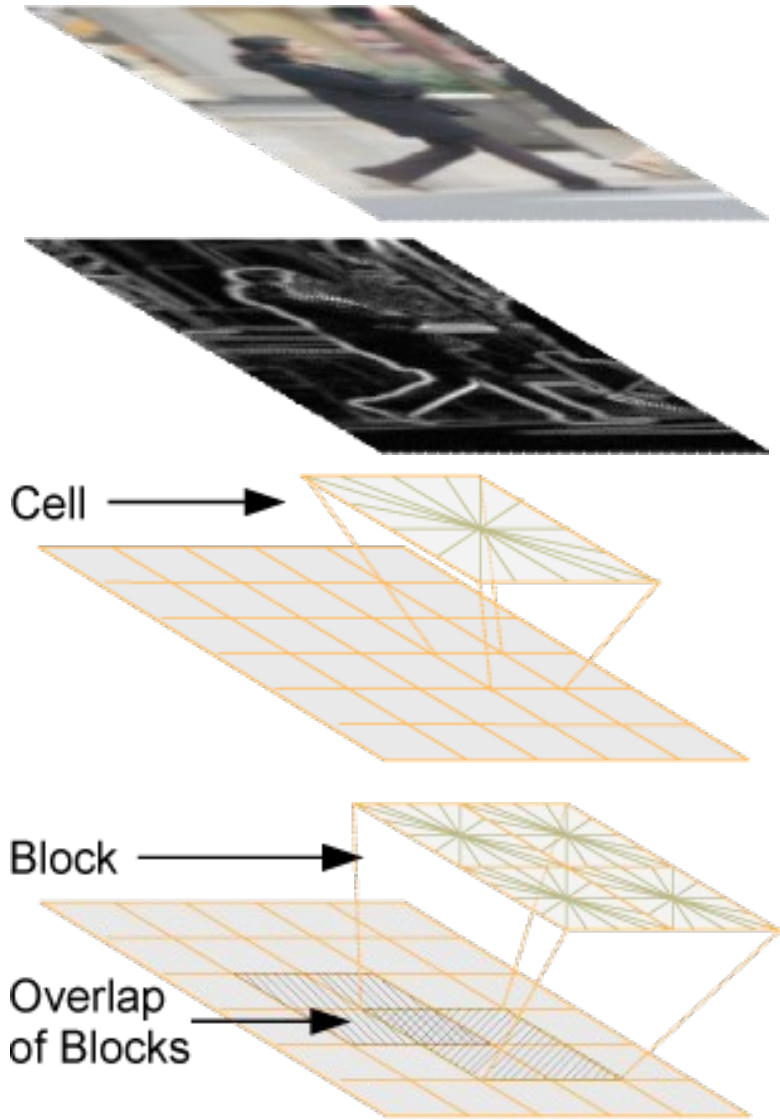
Next slides from presentation by Seeman.



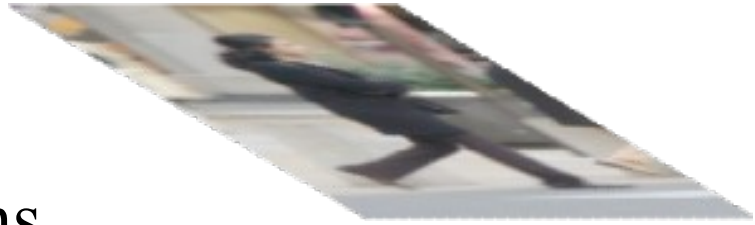
0.alexandre.hiltcher.009.png

Descriptor

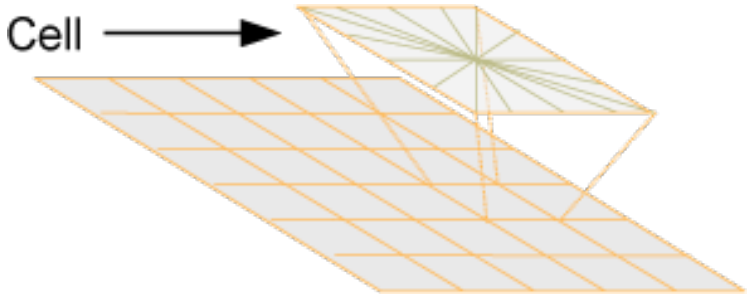
1. Compute gradients on an image region of 64x128 pixels
2. Compute histograms on ‘cells’ of typically 8x8 pixels (i.e. 8x16 cells)
3. Normalize histograms within overlapping blocks of cells (typically 2x2 cells, i.e. 7x15 blocks)
4. Concatenate histograms



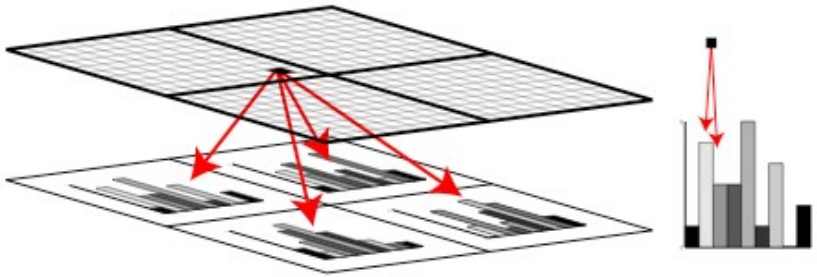
Cell histograms



- 9 bins for gradient orientations (0-180 degrees)
- Filled with magnitudes

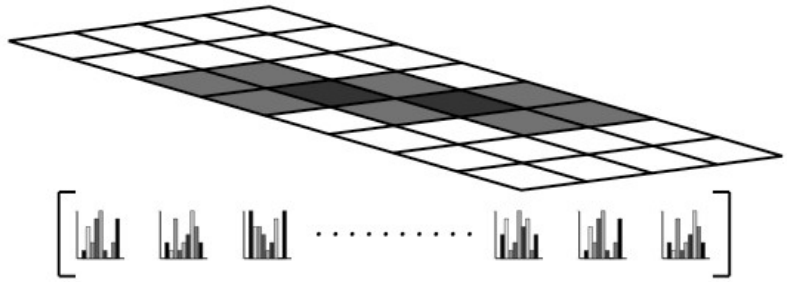


- Interpolated trilinearly:
  - Bilinearly into spatial cells
  - Linearly into orientation bins

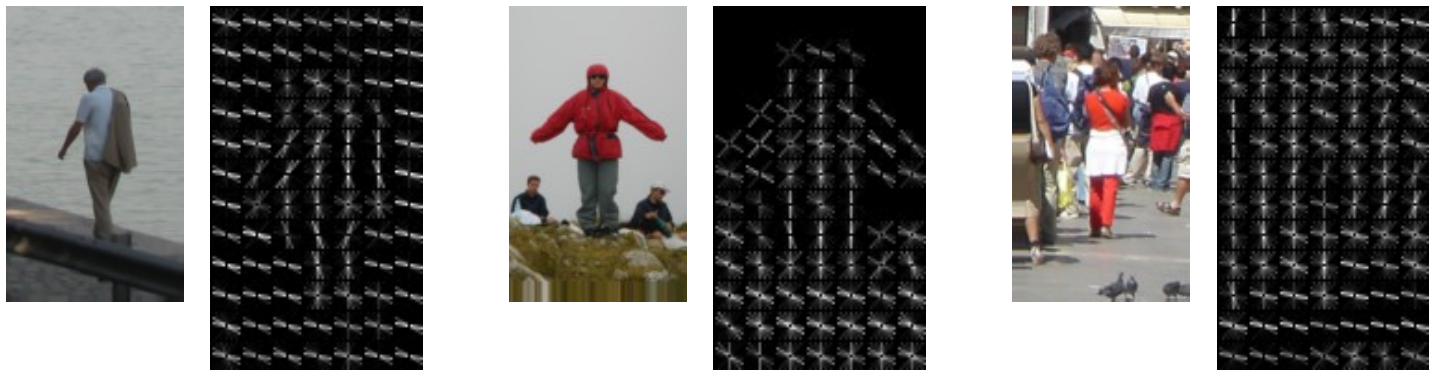


Final Descriptor

- Concatenation of Blocks



- Visualization:



From Sobel to CNN

- many convolution filters :
  - many edge detectors (Roberts, Sobel, Kirch, Prewitt, Laplace)
  - smoothing (mean, Gaussian, ...)
  - edge boosters
- which ones to choose ? thresholds ?
- other approaches :
  - Canny, Deriche
  - Deformable models (parametric curves, levelsets)
- CNN is learning the filters coefficients ... and gets the best results

Deep descriptors

- using already trained CNN
  - Xception (2016)
  - VGG16, VGG19 (2014)
  - ResNet, ResNetV2, ResNeXt (2015-2016)
  - InceptionV3 (2015)
  - InceptionV4, InceptionResNetV2 (2016)
  - MobileNets (2017)
  - DenseNet (2017)
- different options
  - tensor
  - pooling (reducing the size) : average or max
- try them using keras : `https://keras.io/applications/`