Image descriptors

Diane Lingrand

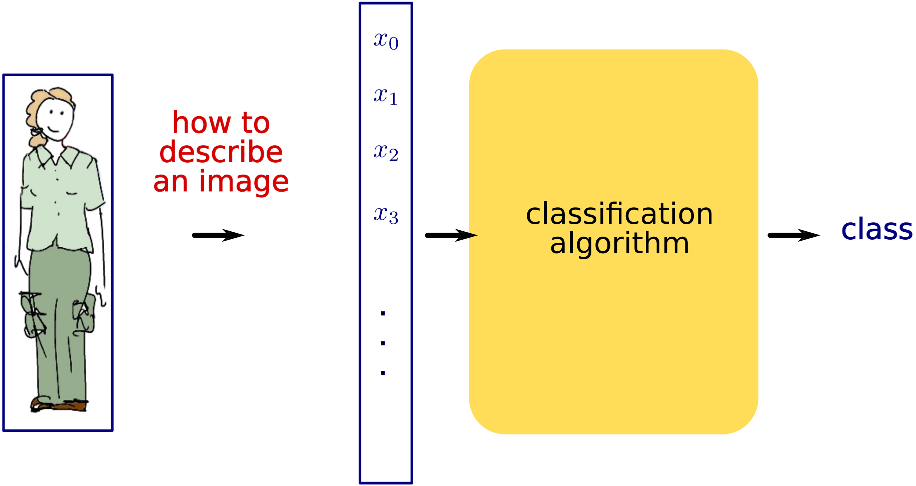
POLYTECH NICE SOPHIA

UNIVERSITÉ CÔTE D'AZUR

SI4

2020 - 2021

Problematic



how to describe an image

x_0

x_1

x_2

x_3

$\vdots$

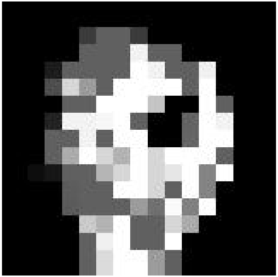
classification algorithm

class

Menu

- Bag of Words / Bag of Features
    - using SIFT, SURF, ...
- HOG
- Deep features

Very basic idea : image as a pixel array



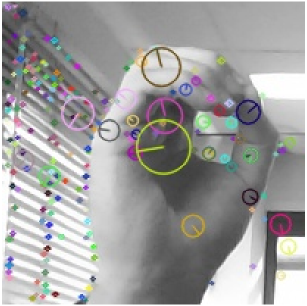
0.alexandre.hiltcher.009.txt

[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]

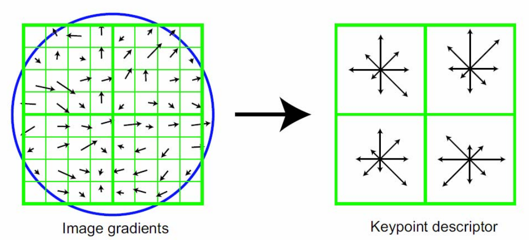
SIFT (Lowe 1999)

SIFT = Scale Invariant Feature Transform

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

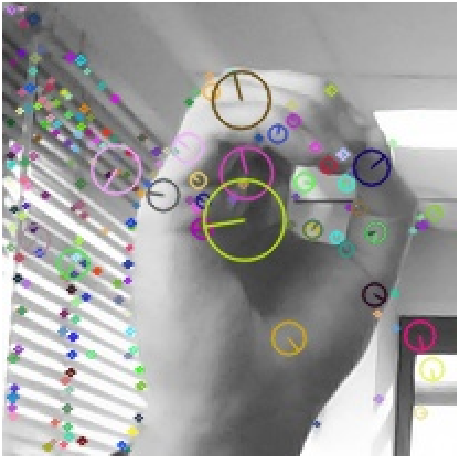


SIFT descriptor

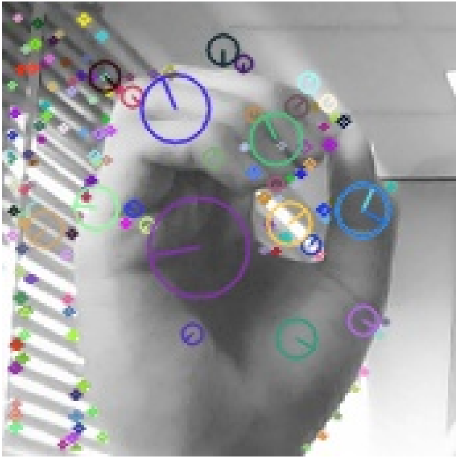


- vectors of 128 integers
- 4 steps :
  - interest points detection
  - gradients orientation in the neighborhood (16x16 pixels divided in 16 blocks of size 4x4)
  - orientation histogram (quantified on 8 values in blocks of 4x4 pixels)
    - $8x4 * 4 = 128$
  - normalisation

SIFT descriptor



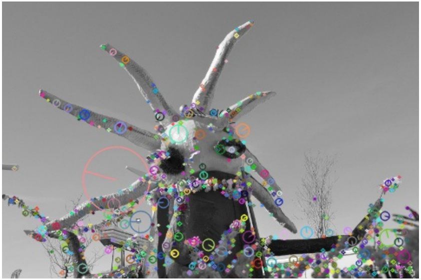
`0.alexandre.hiltcher.006.png`
252 descriptors



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

SIFT implementation in OpenCV

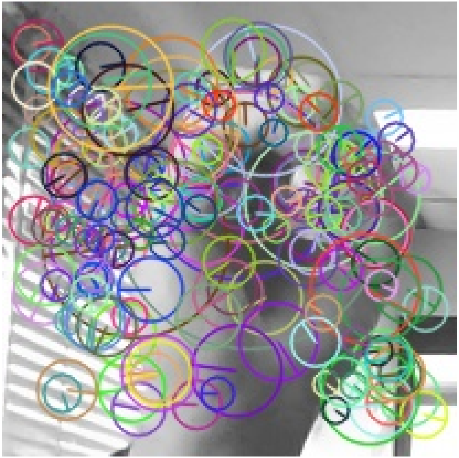
```
import cv2
import numpy as np
img = cv2.imread('carnaval.jpg')
gray= cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
sift = cv2.xfeatures2d.SIFT_create()
kp = sift.detect(gray,None)
cv2.drawKeypoints(gray,kp,img,\\
   flags=cv2.DRAW_MATCHES_FLAGS_DRAW_RICH_KEYPOINTS)
cv2.imwrite('carnavalSIFT.jpg',img)
```



SURF descriptor (Bay etal 2006)

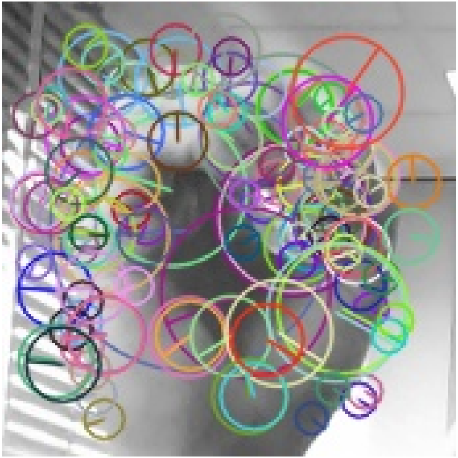
SURF = Speeded-Up Robust Features

default size : 64



`0.alexandre.hiltcher.006.png`

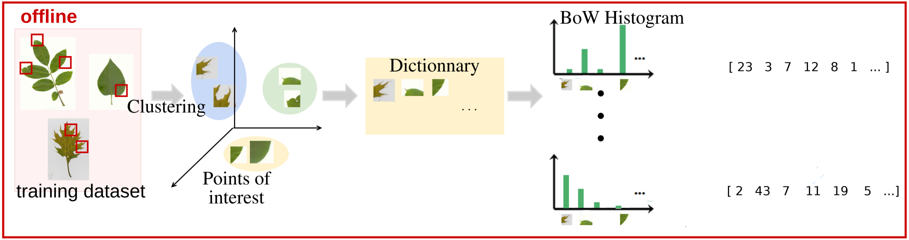
184 descriptors



`0.alexandre.hiltcher.009.png`

121 descriptors

Bag Of Words (BOW)



offline

training dataset

Clustering

Points of interest

Dictionnary

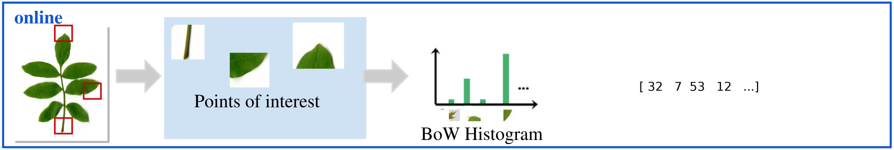
. . .

BoW Histogram

...

[23 3 7 12 8 1 ...]

[2 43 7 11 19 5 ...]



online

Points of interest

BoW Histogram

...

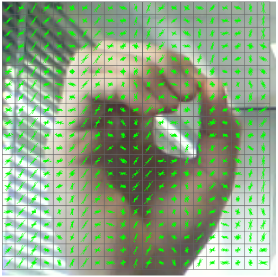
[32 7 53 12 ...]

HOG descriptor (Dalal and Triggs, 2005)

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`

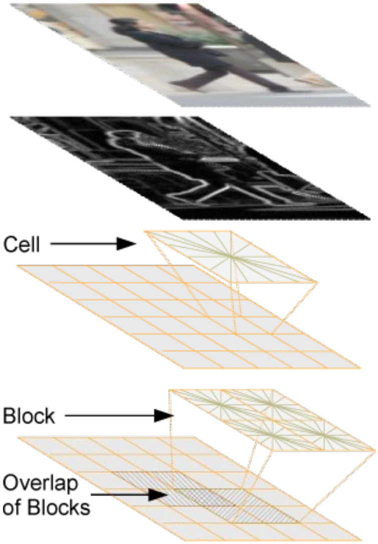
Descriptor

1. Compute gradients on an image region of 64x128 pixels

2. Compute histograms on ‘cells’ of typically 8x8 pixels (i.e. 8x16 cells)

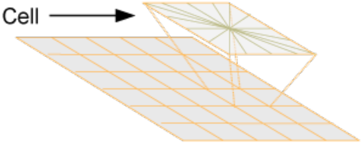
3. Normalize histograms within overlapping blocks of cells (typically 2x2 cells, i.e. 7x15 blocks)

4. Concatenate histograms

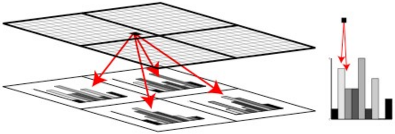


Cell histograms

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

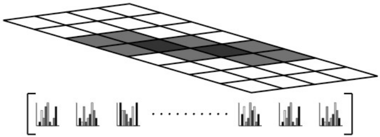


- Interpolated trilinearly:
  - Bilinearly into spatial cells
  - Linearly into orientation bins

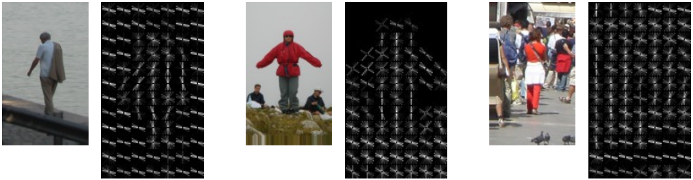


Final Descriptor

- Concatenation of Blocks



- Visualization:



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
  - flatten (or reshaped) : same number of elements
  - pooling (reducing the size) : average or max
- try them using keras : `https://keras.io/applications/`
- autoencoder