Master 2 Paris-Saclay
AIC - Data & Knowledge - Image Mining
Lab 1: Feature Detection, Description and Matching

2019-2020

The goal of this practical work is (1) to get familiar with the image processing library python-opencv, and (2) to experiment, criticise and implement different *representation* methods for matching local structures in images, by following the different steps:

- DETECTOR: How to reduce the representation support.
- DESCRIPTOR: What information to attach to every point of the support.
- METRICS: What measure to use for matching points.
- SEARCH: How to browse the set of descriptors and match the points.

You are expected to provide a report in pdf format, that will include the answers to the questions, a justification of the chosen methods and, as much as possible, illustrations with your own experiments.

1 Software preliminaries

A software basis is provided, that works with the standard opencv-python package(without opencv-contrib, tested version is 4.1.0) and Python3. The software basis can be downloaded on the following link:
`http://perso.ensta-paristech.fr/~manzaner/Cours/Masters_ParisSaclay/Image_Mining/TP1_Features_OpenCV.zip`

All provided codes are editable Python3 scripts that can be run with commands like:

```
$ python3 Convolutions.py
```

Some scripts need arguments, see Python code for more details.

A few test image pairs can be found in the same directory as the software:
`http://perso.ensta-paristech.fr/~manzaner/Cours/Masters_ParisSaclay/Image_Mining/TP1_Image_Pairs.zip`

OpenCV is a very popular image processing library, and many guides and tutorials can be found on line (pay attention to Python and OpenCV versions used in the examples). We will only mention here the official page and the reference guide for functions and classes:
`https://docs.opencv.org/4.1.0/`
and the tutorial :
`https://opencv-python-tutroals.readthedocs.io/en/latest/`

2 Image Format and Convolutions

Q1 Experiment the convolution code given as example in *Convolutions.py*. Note the difference between the direct calculation by scanning the 2d array and the calculation using function *filter2d* from OpenCV. Try to decrypt the OpenCV functions used for image reading and copying, and the MatPlotLib function used for image display.

Q2 Explain why the convolution kernel provided as example realises a contrast enhacement with respect to the original image.

Q3 Modify the code to compute, with the two methods, the convolution that approximates the partial derivative $I_x = \frac{\partial I}{\partial x}$, then the gradient modulus $||\nabla I|| = \sqrt{I_x^2 + I_y^2}$. What precautions should be taken in order to get a correct display (i.e. a correct interpretation of the grayscales)?

3 Detectors

Q4 Complete the code in the *Harris.py* script to compute the interest function of Harris (at one single scale), and the corresponding interest points. Explain how the provided code, that uses morphologic dilation, allows to calculate the local maxima of the interest function *Theta*.

Q5 Comment the results obtained with your Harris detector and the effect of the used parameters, in particular the size of the summing window and the value of α. How is it possible to extend this computation on several scales?

Q6 Experiment and compare the two detectors ORB and KAZE by running the script *Features_Detect.py*. Find the principle of each detector. Explain the main parameters intrinsic to each detector and their effect on the detection. How is it possible to visually evaluate the repeatability of each detector applied on a pair of images?

4 Description and Matching

Q7 Explain the principle of the descriptors attached to points ORB and those attached to points KAZE. What properties of detectors and/or descriptors (distinguish the two aspects in your answer), allow to make the matching scale and rotation invariant?

Q8 Explain and compare qualitatively the effects of the three different point matching strategies performed in the three scripts *Features_Match_CrossCheck.py*, *Features_Match_RatioTest.py* and *Features_Match_FLANN.py*. Explain why the used distances are different for the two descriptors. Explain why the strategy FLANN (and, in a lesser extent the strategy RatioTest) does not work well with ORB points.

Q9 Propose a strategy to evaluate quantitatively the matching by using an image with a known geometric transformation (OpenCV functions like *cv2.warpAffine* can be used).