Practical block I: Landmark detection and descriptors

Objective of the practical session

Learn how to use different landmark detection method, analyze which one is the most suited, and why. Critically analyze the performances of the detectors applied to different exemplar images, and learn how to tune optimally the landmark detection and matching pipeline.

1. Introduction

Image matching is a fundamental aspect of many problems in computer vision including object and scene recognition. During the development of this practice we aim at analyzing the process for extracting distinctive invariant features from images to perform reliable matching among different views of an object or scene. The extracted feature set should be highly distinctive in order to provide robust matching across a substantial range of affine distortions, changes in 3D viewpoint, addition of noise, and changes in illumination. The aim of the lab is to analyze the characteristics of image descriptors and the matching process for detecting a certain pattern. This practical block is expected to be performed in three sessions.

The method

The image descriptors extract from an image a set of significant points in a way that it is consistent with variations of illumination, viewpoint and other viewing conditions The descriptor associates to such points a signature or descriptor which identifies their appearance in a discriminative and compact way, see figure 1a.

The descriptor vectors extracted from different images can be then matched. The matching is often based on a distance between these vectors, e.g. the Euclidean or Mahalanobis distance. A simple way to proceed is to compute the distance between each descriptor in for each point. Then a matched pair from the best score (minimal distance), see figure 1b, is constructed. To increase robustness matches are rejected in relation to nearest neighbor heuristic approaches.

Although the distances approaches described above discard many of the false matches arising from background clutter, we will still may have (false) matches belonging to different objects. Therefore, it is necessary to exclude those points which do not maintain a geometric consistency. For planar scenes this is easy to manage: each possible transformation can be expressed with a homography. A homography (in 3D) is a mapping of a plane to another plane. Mathematically, a homography can be computed by establishing a correspondence between 4 points in each plane. Homographies between pairs of images can be estimated using RANSAC (RANdom SAmple Consensus), see [3]. In case of feature point matching, RANSAC removes outliers in an iterative procedure by randomly selecting a subset of the data and estimating the consensus score based on a classification hypothesis in the remaining set of data.

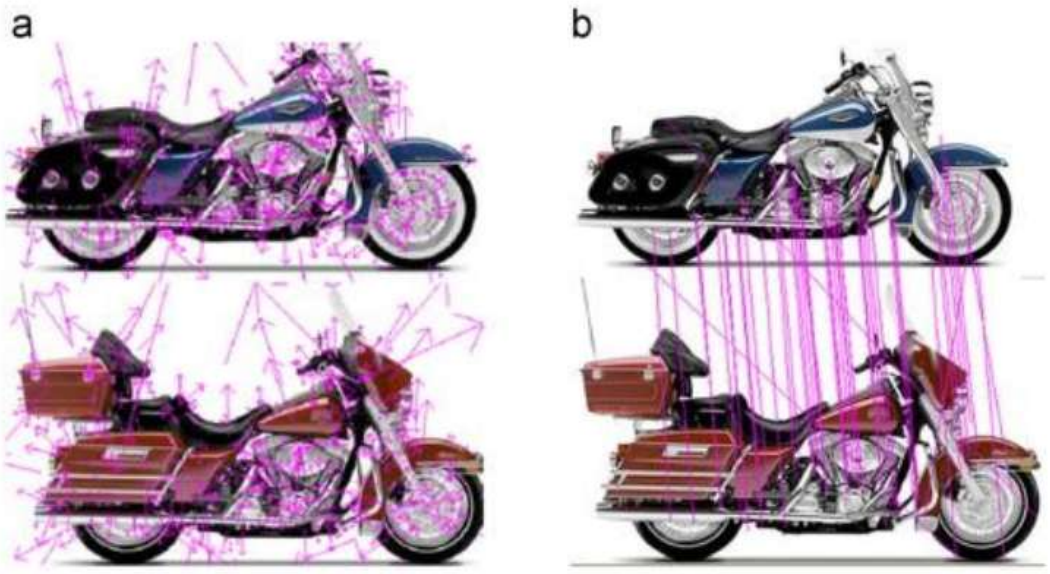


Figure 1: a) 2D detected keypoints and b) matched keypoints.

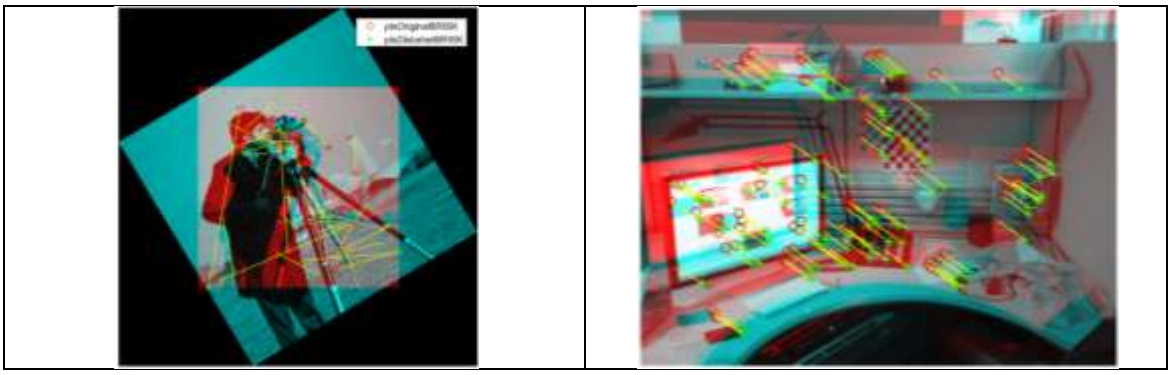
2. Practical session

2.1. Introduction

The practical session of the block I consist in coding a landmark detection and aligment between a pair of images. The pipeline consist in extracting relevant landmarks using a feature descriptors and then matching them using RANSAC.
In order to show what it is expected at the end of the practical session, please have a look to the following two matlab examples:

1) https://es.mathworks.com/help/vision/ug/local-feature-detection-and-extraction.html
openExample('vision/HowToUseLocalFeaturesExample')
this first script illustrates how the image descriptors are used to recover the rotation of an image.

2) https://es.mathworks.com/help/vision/ref/matchfeatures.html
openExample('vision/UseSURFFeaturesToFindCorrespondingPointsExample')
this second script illustrate how the image descriptors are use to recover the alignment between stereoscopic images

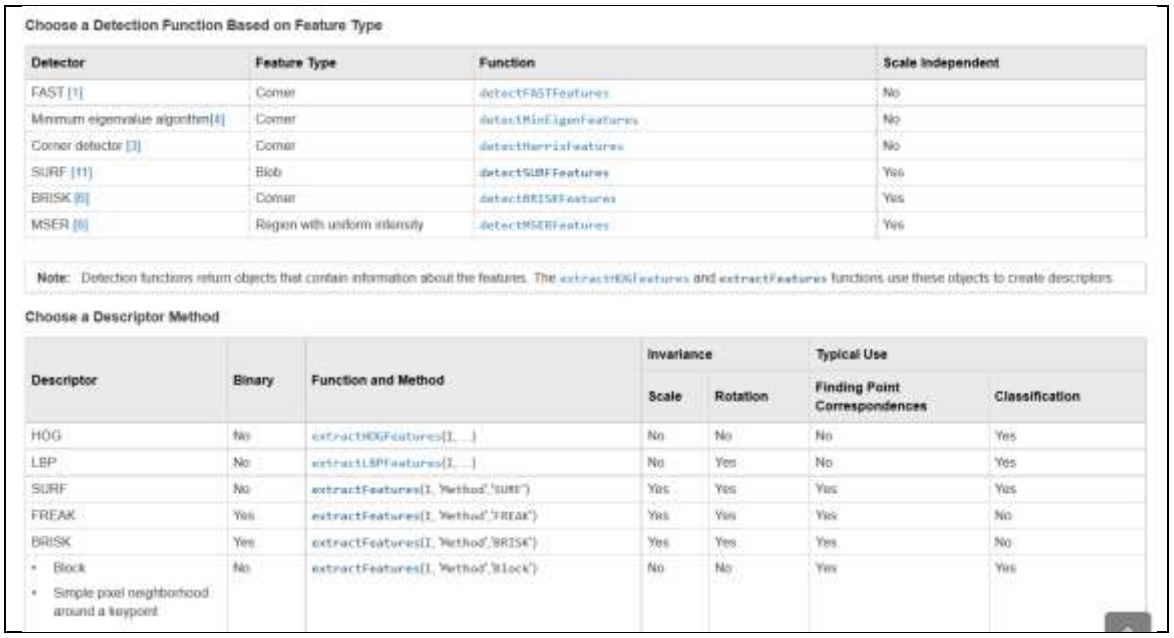


IMPORTANT, these examples will not be used during the practical session. They are meant to illustrate exemplar pipelines that the student will reproduce with their own code during the practical session.

The examples are based on concept described into the the three referenced papers at the end of this document. It is recommended to read them before the first session.
Several point detector and image descriptors are described in the paper [4], others are listed in one of the Matlab examples.

Feature Detection and Feature Extraction

Feature detection selects regions of an image that have unique content, such as corners or blobs. Use feature detection to find points of interest that you can use for further processing. These points do not necessarily correspond to physical structures, such as the corners of a table. The key to feature detection is to find features that remain locally invariant so that you can detect them even in the presence of rotation or scale change.
Feature extraction involves computing a descriptor, which is typically done on regions centered around detected features. Descriptors rely on image processing to transform a local pixel neighborhood into a compact vector representation. This new representation permits comparison between neighborhoods regardless of changes in scale or orientation. Descriptors, such as SIFT or SURF, rely on local gradient computations. Binary descriptors, such as BRISK, ORB or FREAK, rely on pairs of local intensity differences, which are then encoded into a binary vector.
The next two tables provide details on the detectors and descriptors available in Computer Vision Toolbox.



Choose a Detection Function Based on Feature Type

Detector	Feature Type	Function	Scale Independent
FAST [1]	Corner	detectFASTFeatures	No
Minimum eigenvalue algorithm[4]	Corner	detectMinEigenFeatures	No
Corner detector [3]	Corner	detectHarrisFeatures	No
SURF [11]	Blob	detectSURFFeatures	Yes
BRISK [6]	Corner	detectBRISKFeatures	Yes
MSER [8]	Region with uniform intensity	detectMSERFeatures	Yes

Note: Detection functions return objects that contain information about the features. The extractHOGFeatures and extractFeatures functions use these objects to create descriptors.

Choose a Descriptor Method

Descriptor	Binary	Function and Method	Invariance		Typical Use	
			Scale	Rotation	Finding Point Correspondences	Classification
HOG	No	extractHOGFeatures(I, ...)	No	No	No	Yes
LBP	No	extractLBPFeatures(I, ...)	No	Yes	No	Yes
SURF	No	extractFeatures(I, 'Method','SURF')	Yes	Yes	Yes	Yes
FREAK	Yes	extractFeatures(I, 'Method','FREAK')	Yes	Yes	Yes	No
BRISK	Yes	extractFeatures(I, 'Method','BRISK')	Yes	Yes	Yes	No
• Block • Simple pixel neighborhood around a keypoint	No	extractFeatures(I, 'Method','Block')	No	No	Yes	Yes

2.2. Goal and tasks:

Identify the optimal pipeline (combining a pair of feature detector and descriptor) considering the criteria of your application and the nature of your data, quantify the performance of the pipeline when image distortions are introduced, and comment the obtained results.

- Create a test data-set:

Select several sample images provided by Matlab ('cameraman.tif', 'lena', 'viprectification_deskLeft.png',) and others that the student may consider relevant. Choose the images in order to show the limitation of each specific feature detector and descriptor. For instance a corner detector might have good performances on images containing corners, a blob detector might perform better on images having rounded objects.
Apply distortions to the image

- Apply image distortions to the data-set:

Image distortions (rotation, scaling, projections, blurring, intensity and contrast changes) might affect the performances of the pipeline. The amount of distortion can be increased progressively, in order to see the performance reduction. For instance, the more the blurring, the less point correspondences are expected to be found. In such way the original image can be compared against a second one having a distortion, and the number of matches can be quantified.

- Test and compare several pipelines

Define several pipelines combining the feature detection and feature extraction functions available in matlab. Create a script that executes several meaningful combinations of such functions on the data set. Note that it is not necessary to compare all of them. The score of the practical session will depend most of the analysis of the descriptor and not on the amount of descriptors chosen.
Critically compare the different image descriptors to scale, rotation, change in 3D viewpoint, change in illumination invariance and computational efficiency. Which details of the image are identified by each detector? Which one is the most suited for the selected image? The behavior is the one expected (for instance a corner detector, is still robust, when a blurring makes the corners smoother?)
Critically analyze each of them in the PDF report.

- Quantitative comparison

The RANSAC algorithm identifies the matches between the features and discards the pairs having low probability. In general RANSAC is used to compute the homography between pair of images. However, in our case, RANSAC can be also employed to quantitatively compare the performances of a certain pipeline. For instance, RANSAC can be used to quantify the amount of matches found between the original image and the distorted one. Consequently, the more distortion is applied, the lower the number of matches expected.

Note that the Ransac algorithm depends on some parameters (outlier rejection threshold etc..) which might be previously set, for different scenarios (different images: blurred, rotated, rescaled...etc).

In case it is not available in your matlab distribution you can download it from the following link:
https://es.mathworks.com/matlabcentral/fileexchange/30809-ransac-algorithm-with-example-of-finding-homography

Alternative metrics that allow quantifying the results (for instance implementing the metrics detailed in [4]) are also accepted.

Add further examples in order to show the limitations of the algorithm (example panorama stitching, object identification (in a Bag of Word fashion) etc....

3. Deliverable

A **SHORT** report in PDF format containing around five pages must be delivered for the first practical block. Please do not include "all" the results, only the meaningful images, comments and tables required for your analysis. The matlab code of the session should be delivered as well. Code everything in a single ZIP file.

IMPORTANT, only files delivered using the campus virtual will be evaluated. Please upload in the corresponding task your file before the dead line
IN CASE YOU HAVE NO ACCCESS TO THE CAMPUS VIRTUAL please contact the professor of the practical course to solve the issue.

References:

[1] Lowe, D.G. Distinctive Image Features from Scale-Invariant Keypoints, International Journal of Computer Vision, November 2004.

[2] Bay, H. and Ess, A. and Tuytelaars, T. and Van Gool, L. Speeded-Up Robust Features (SURF), Computer Vision and Image Understanding, June 2008.

[3] Fischler, M.A. and Bolles, R.C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM. June 1981.
[4] S Gauglitz, T Höllerer, M Turk "Evaluation of interest point detectors and feature descriptors for visual tracking" International journal of computer vision, 2011 - Springer