

**Assignment #2 [100 points]**  
**Feature Detection, Description, and Matching**

The goal of this assignment is to implement a simple feature detection, description, and matching pipeline that can be used for variety of applications including image registration to object retrieval.

**A) Feature detection [30pts]**

**Step-A1:** As described in class and in lecture notes compute the autocorrelation matrix:

$$A = \begin{bmatrix} G(\sigma) \otimes I_x^2 & G(\sigma) \otimes I_x I_y \\ G(\sigma) \otimes I_x I_y & G(\sigma) \otimes I_y^2 \end{bmatrix}$$

You can use built in Python and OpenCV functions for convolution/filtering function (i.e. `cv2.filter2D`).

**Step-A2:** Compute a scalar feature detection measure using one of the formulas discussed in class:

- Shi-Tomasi,
- Harris-Stephens, or
- Harmonic mean.

**Step-A3:** Find local maxima above a certain threshold and report them as detected feature point locations.

Show the detected feature points on the image and include them to the report.

**B) Feature matching and evaluation [70pts]**

**Step-B1:** Define and describe a feature descriptor. You can simply use square patches around the detected feature points.

**BONUS [1-10 points]** Or you can design and implement a better descriptor that is invariant to one or more of the followings:

- scale
- rotation
- brightness-contrast

Specify in the report which invariance you were targeting and whether you were successful.

**Step-B2:** Perform feature matching and report results for the three matching strategies described in class and lecture notes:

1. distance thresholding,
2. nearest match, and
3. nearest neighbor distance ratio.

**Step-B3:** Use the transformation/homography matrices (H1to2p, H1to3p...) provided with the data and evaluate feature matching performance by reporting

- TPR (True Positive Rate) and
- FPR (False Positive Rate) for the three matching strategies.

Process:

1. Apply the given transformation matrix to a point to estimate its location in the second image.
2. Compute pixel distance between the estimated position and the matched point's position.
3. If the distance is less than k pixels consider a true positive otherwise label as false positive.

Show the matched points on the input images and include the pictures to your report.

### **Program Structure and Deliveries:**

Your program needs to include the following functions:

**1. function [R]=computeFeatureMeasure(I,parameters)**

I: an NxM image grayscale measure

Parameters: additional parameters needed

R: an NxM image, one of the three feature measures described in lecture notes (ShiTomasi/Harris/HarmonicMean).

**2. function [x,y,mask]=FeatureMeasure2Points(R,npoints)**

R: NxM feature measure matrix

npoints: number of feature points that will be returned

x,y : coordinates of the feature points (npointsx1)

mask : binary mask showing the location of the feature points

**3. function [Dlist]=generateFeatureDescriptors(I,x,y)**

I : input image

x,y : coordinates of feature points (npointsx1)

Dlist: npointsxDlength descriptor list, where Dlist(i,:) is the feature descriptor vector for the feature point x(i),y(i).

**3. function [Dist]=computeDescriptorDistances(Dlist1,Dlist2)**

Dlist1: descriptor list from first image

Dlist2: descriptor list from second image

Dist: distance matrix, Dist(i,j): distance between ith descriptor in first image and jth descriptor in second image.

4. **Function [MatchList1]=Distance2Matches\_DistThresh(Dist,Th1)**  
**Function [MatchList2]=Distance2Matches\_NearestMatch(Dist,Th2)**  
**Function [MatchList3]=Distance2Matches\_NearestRatio(Dist,Th3)**  
 Dist: distance matrix, Dist(i,j): distance between ith descriptor in first image and jth descriptor in second image.  
 Th1,Th2,Th3: threshold values needed for the methods  
 MatchList: nx2 matrix of matches where MatchList(i,1) is the index of a feature points in first and and MatchList(i,2) is the index of the matching feature point in the second image.

### **Test Data**

Download the file hw2.zip for test data. Your program should get reasonable matches for at least bikes\img1 and bikes\img2. With appropriate descriptor design you can get good results for the other images. This will get you bonus points.

### **Submission instructions:**

1. **Code:** Code should be organized into functions as described above. Use specific names and inputs/outputs. Non-compliant code will result in reduced points even if the outputs are correct.
2. **Report:** The report should include the following items for at least two image pairs. Include more images to demonstrate how well your pipeline works, particularly if you worked on bonus section.
  - a. Detected feature points marked on the input images
  - b. For three matching strategies:
    - a. Matched points connected with a line on the input images
    - b. Matching statistics as a table

	<b>Distance threshold</b>	<b>Nearest neighbor</b>	<b>Nearest neighbor distance ratio</b>
Number of detected feature points			
Number of matched points			
TPR (True Positive Rate)			
FPR(False Positive Rate)			

