Computer Vision
Lab Exercise 2
Harris Corner Detector and SIFT Descriptor

Students should work on this lab exercise in groups of two people. In this assignment, you will be implementing the scale invariant Harris Corner Detector. You will then use this detector to match images with the SIFT descriptor.

1 Harris Corner Detector

In this part, you will be implementing a scale invariant Harris Corner Detector. In the first step you need to complete the code in the harris function that is provided on Brightspace. All the necessary steps are indicated in the code with the hint for the completion. You need to compute the entries of the structure tensor matrix which you will use to form your 'cornerness'(R)). (review your lecture slides for this task!)

The corner points are the local maxima of R. Therefore, in your function you should check for every point in R, if it is greater than all its neighbours (in an (n x n) window centered around this point) and if it is greater than the user-defined threshold, then it is a corner point.

In the second step, make your detector scale invariant. In order to build a scale invariant detector you need to work with different scales. All you

need to do is to compute scale normalized laplacians with different sigma values and find corresponding corner points in each level. This time you do an additional check also within levels. Instead of checking local maxima only within neighbours in the same image, you need to find the maximum of same corner points in different levels (you can use 3X3 searching window in order to match corner points). Now, the points which returned as local maxima are scale invariant. Your function should return the rows of the detected corner points r, and the columns of those points c (the first corner is given by (r(1),c(1))). Your function should also plot the original image with the corner points plotted on it.

2 Image Matching with SIFT

In this part, you will be using your corner points detected in the previous section. You will need to build a descriptor for each of these corner points in the given image-a. And try to find closest descriptor in the other given image-b (Euclidean distance). You can define your own threshold for matches to be accepted as correct based on the effect of the ratio-of-the-second-best-match, as mentioned in the lecture(try different possibilities and describe).

You can find an implementation of SIFT at `http://www.vlfeat.org/index.html`. Download and setup the vlfeat package for Matlab.