Dpto. Ingeniería de Sistemas y Automática
Universidad de Málaga

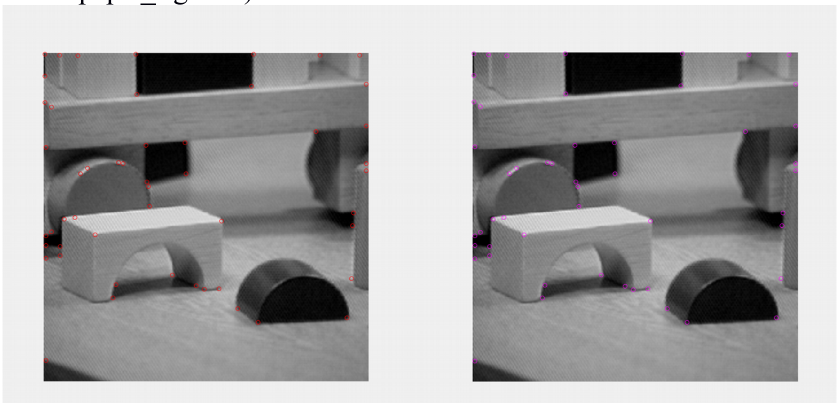
COMPUTER VISION

Exercise 3: Keypoint detection using Harris and matching

Concepts: Harris operator, keypoint matching, normalized correlation

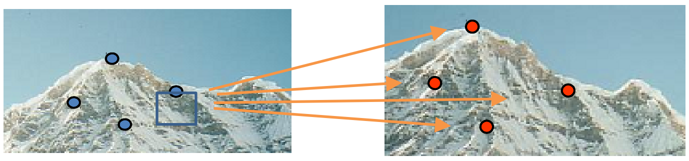
Implement a Matlab script to:

1. Use the **harris()** function to detect keypoints in two images from the same scene (for example pepsi_left.tif and pepsi_right.tif).



In the previous image I highlighted the keypoints in both photos. One can clearly see how the perfectly coincide.

2. Show in the same figure both images and draw the detected keypoints in them.



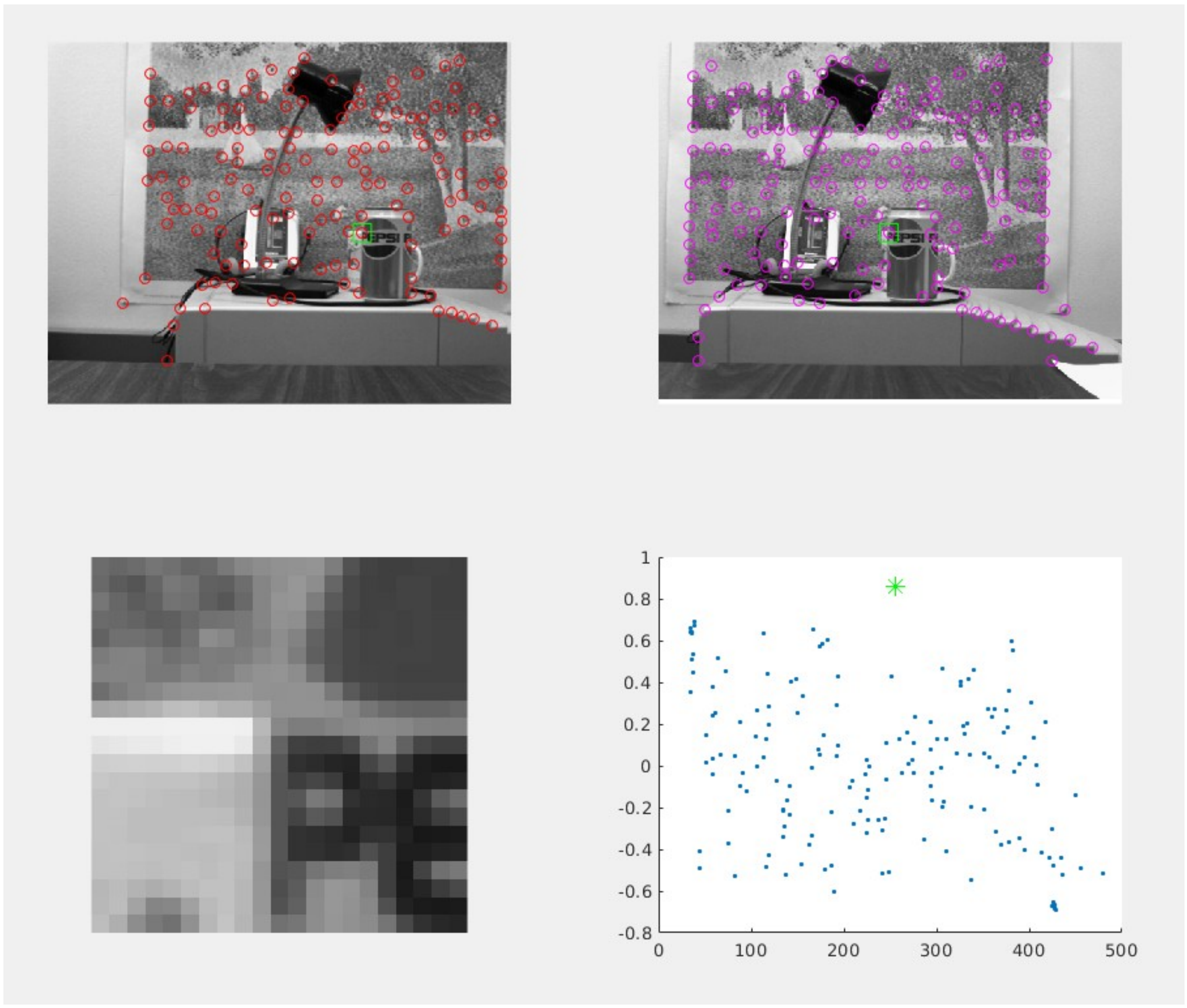
3. Now, try to match points employing Normalized Cross-Correlation (NCC) (see figure in next page). Implement a loop iterating over the keypoints in the left image, an in each iteration:
   a) Crop a squared region surrounding the keypoint, the size of the resulting template is up to you! *Note: you should consider only the keypoints that are not in the image border; this depends on the window size that you have chosen.*
   b) Perform correlation with the cropped image (the template) and the right image using the command **normxcorr2**. *Note: the resultant matrix of correlation coefficients has a different size than the right image, so you have to remove the unnecessary-generated border for next steps.*
   c) Obtain the NCC values for all the Harris keypoints in the right image and plot them: in the x-axis, the index of the point attending to how they are returned by the **harris()** function, in the y-axis, the NCC value (take a look at the result image). Draw an asterisk in the position of the keypoint with the highest correlation value.
   d) Draw a window surrounding both: the current keypoint in the left image and the keypoint with the highest correlation value in the right window. They are the pair of matched keypoints!. Employ the Matlab command **rectangle()**. *Note: you have to clean some stuff from the figure before starting the next loop iteration, so it does not accumulate plots.*

Useful commands:

[cim, r, c] = harris(im, sigma, thresh, radius);	Detect corners with the Harris operator in the image im.

C = normxcorr2(patch,im);	Computes NCC between the patch and the image in im.
rectangle('Position',[x,y,w,h],' 'EdgeColor','r')	Draws a red rectangle in the given position.
pause;	Pauses the execution until the user press a key in the command window.
delete(handler)	Deletes the graphic with a given handler. You can get the handler of a graphic just assigning to it the value returned by any draw function, for example: handler = plot(x,y);
isempty(handler)	Check if a handler is empty or not.

Results



The previous image shows the results of the combination of Exercise 2 and 3, with some minor changes. Firstly, I added a green rectangle that would show the pair of keypoints that are related. To find which were the pairs related, I had to use the Harris function provided and the normxcorr2 function with a cropped section of the original image.

The principal idea was to get the cropped section and compare it with the value of the correlation in each of the keypoints in the other photo. Then, I select the one with the maximum value as the correct one.

This procedure has some drawbacks though. One of them is that it always tries to couple two keypoints, even if they don't have any partner at all. This issue could be solved using a threshold to pair two keypoints.

In order to make the graph, I just stored in an array the value that each keypoint in the right image had with the one that I was checking. Afterwards I used an scatter plot in order to plot all the points and plotted on top the maximum value to highlight it.