# 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). *Note: To debug the code you can use a simple image like piezas.tiff. For obtaining the same results as in the figure, use the parameters sigma=2, thresh=1000, and radius=10.*
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:
   1. 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.*
   2. 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.*
   3. 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.
   4. Draw a window surrounding both: the current keypoint in the left image and the keypoint with the highest correlation value in the right one. 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.*
   5. Take a look at the first 20 matched keypoints. How many of them are correctly matched? Could this performance be improved?

**Useful commands:**

|  |  |
| --- | --- |
| **[cim, r, c] = harris(im, sigma, thresh, radius);** | Detect corners with the Harris operator in the image im. Introduce help harris in the Matlab command window to get information about the returned variables. |
| **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 (x,y). |
| **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**

