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). *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.

```
imL = imread('pepsi_left.tif');
imR = imread('pepsi_right.tif');
[ibL,Ly,Lx] = harris(imL,2,1000,10);% y coordinates = row and x
coordinates = columns
imshow(imL);
scatter(Lx,Ly);% To plot the points
[ibR,Ry,Rx] = harris(imR,2,1000,10);% y coordinates = row and x
coordinates = columns
imshow(imR);
scatter(Rx,Ry,'r');
```

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.*

```
len=10;%the half length of the rectangle of the template
wit=10;%the half width of the rectangle of the template
rec=[Lx(i)-len Ly(i)-wit len*2 wit*2];% the rectangle of the
template
im=imcrop(imL,rec);%Cropping the image for the template
```

   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=normxcorr2(im, imR);
```

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.

```
Cp=C(len:size(C,1)-len,wit:size(C,2)-wit);%Eliminating the border of
the NCC
%Searching the probabilities of the NCC of each keypoints of the
right
%image
for i=1:length(Rx)
    x=Ry(i);
    y=Rx(i);
    prob(i)=Cp(x,y);
end
scatter(Ry,prob,5,'r.');%Plotting the probability along the row of
the image
[m,p]=max(prob);%Calculating the maximum probability and its
position
scatter(Ry(p),m,'g*');%Plotting the maximum probability
```

d) 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.*

```
subplot(2,2,1);
rectangle('Position',rec, 'EdgeColor','r');
subplot(2,2,2);
rec=[Rx(p)-len Ry(p)-wit len*2 wit*2];
handler2=rectangle('Position',rec, 'EdgeColor','g');
```

e) Take a look at the first 20 matched keypoints. How many of them are correctly matched? Could this performance be improved?

For a square of 20x20 pixels, they are matched correctly 13 of the 20 keypoints. The misses are due to it has not the other image that keypoint detected or the correlation of the two points is bigger that the correct ones. However, if we tried a square of 30x30 or 40x40 pixels, the matches increase up to 15. And thus, it has been improved in increasing the size of the taken template. Other way is in

Results

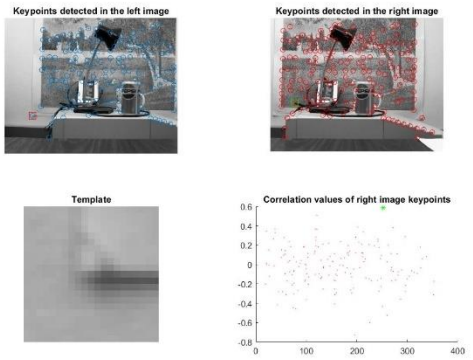

Image 1

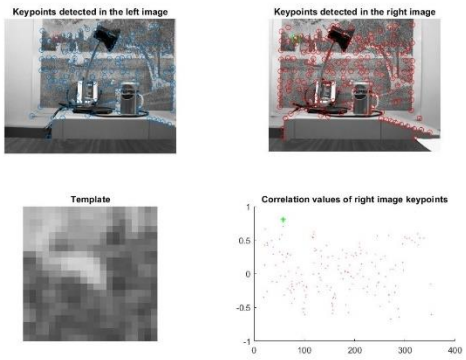

Image 2

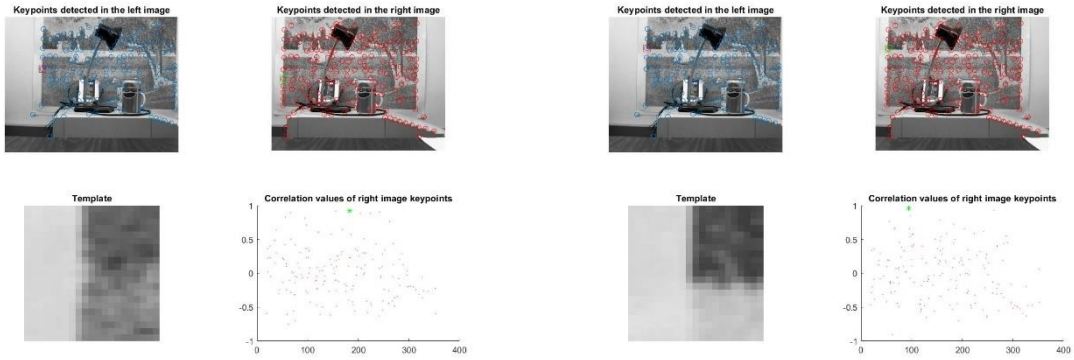

Image 3

Image 4

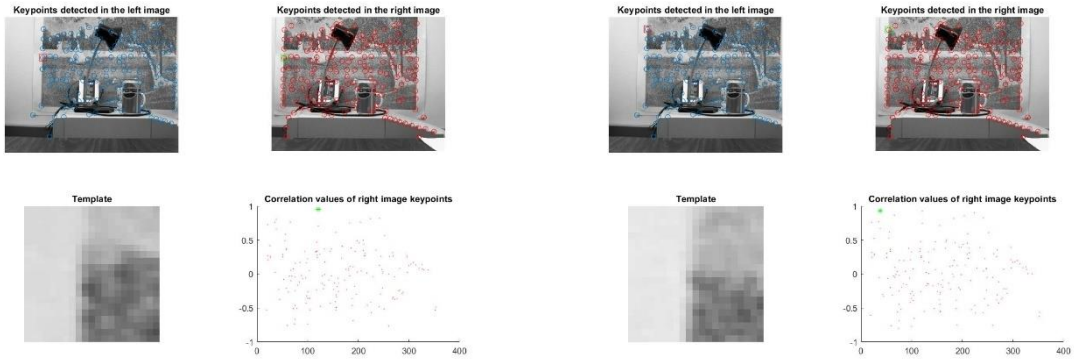

Image 5

Image 6

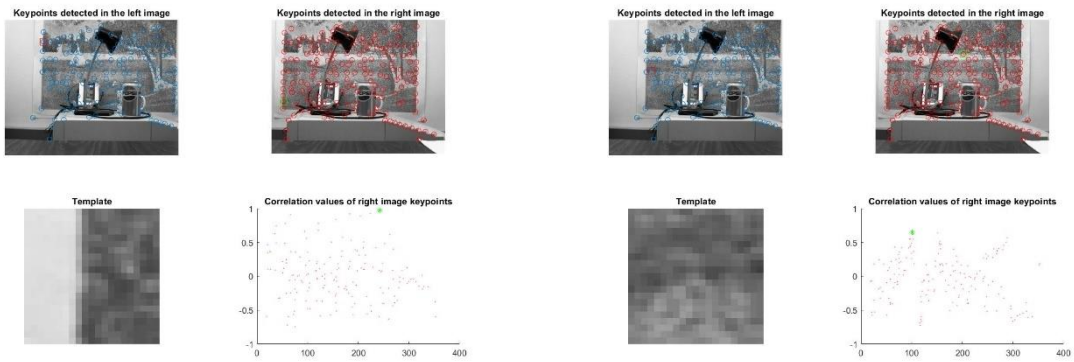

Image 7

Image 8

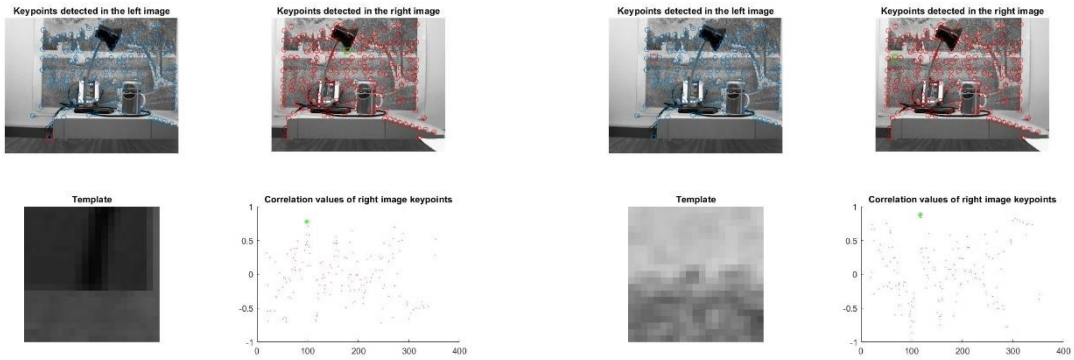

Image 9

Image 10

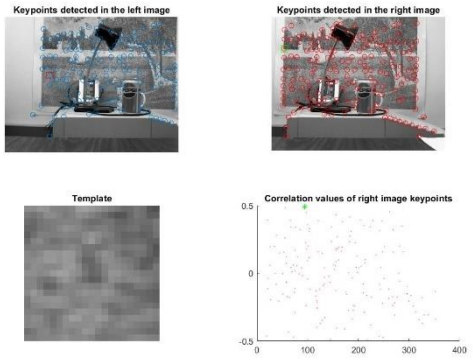

Image 11

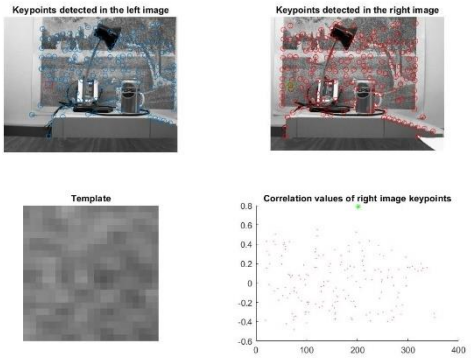

Image 12

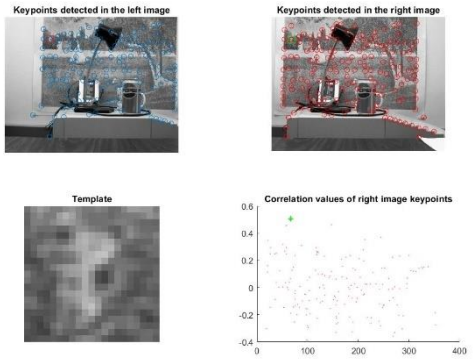

Image 13

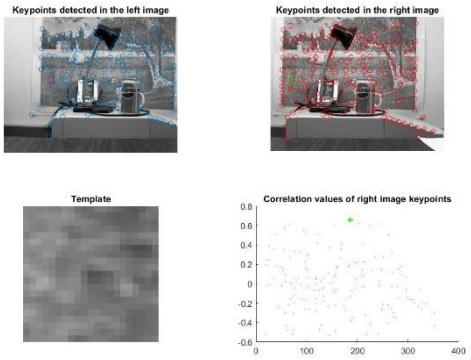

Image 14

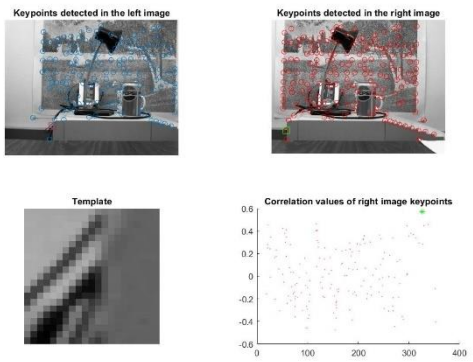

Image 15

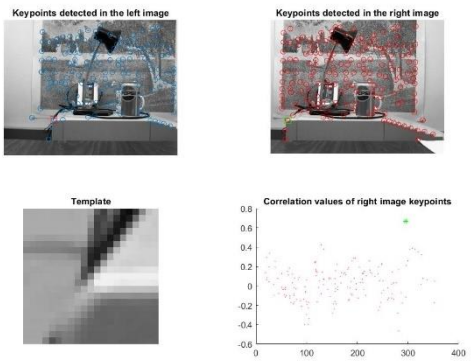

Image 16

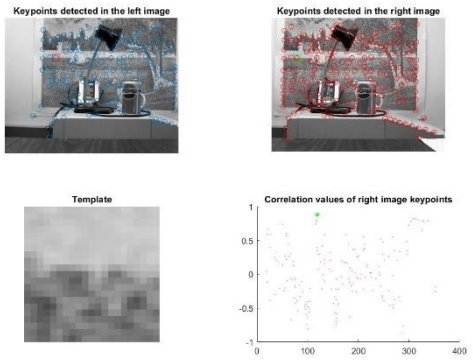

Image 17

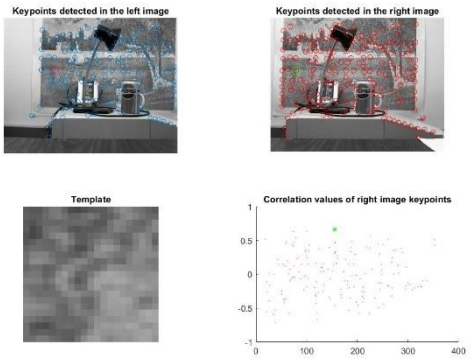

Image 18

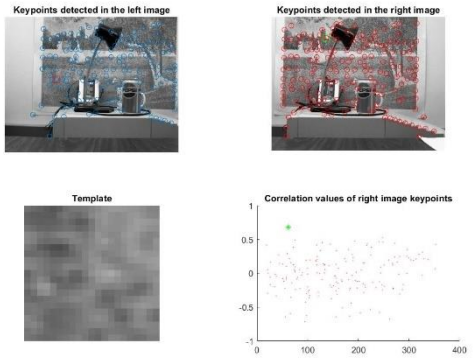

Image 19

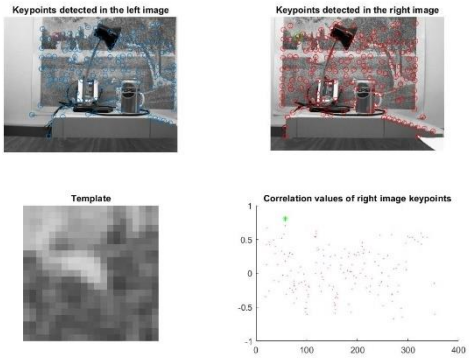

Image 20