LAB 5

NCC-BASED SEGMENTATION AND HARRIS CORNER DETECTION

Matteo Dicenzi
Marco Demutti

1. Introduction

The aim of this experiment is to perform **Normalized-Cross-Correlation (NCC)-based segmentation** and **Harris corner detection** on different images.
NCC-based segmentation is a technique that allows us to choose a small patch (i.e. an arbitrary - specific area) in an image and use it as a template to be detected in other images. In particular, we are going to work on a set of 6 images: after extracting two different templates (a red and a black car) from the first image, we are going to try to find them in all the other images. Specifically in this algorithm, **Normalized Cross-Correlation (NCC)** is computed and used as a robust way of measuring the similarity between template and image.
Then, the results obtained from this experiment (i.e. using NCC-based segmentation) are compared with the ones obtained using ***color-based segmentation*** in the previous Lab and analysed in terms of computation time and accuracy of detection.
In the second part of the experiment, we are going to implement the **Harris corner detector**, an algorithm which allows us to find corners in an image. Indeed, **corners** are distinctive key points in an image, since they correspond to regions where there is a significant change **of intensity** in all the directions.
Specifically, we are going to use this algorithm to detect the corners in the ***"image i235.png"***, and we are going to display all the relative results.

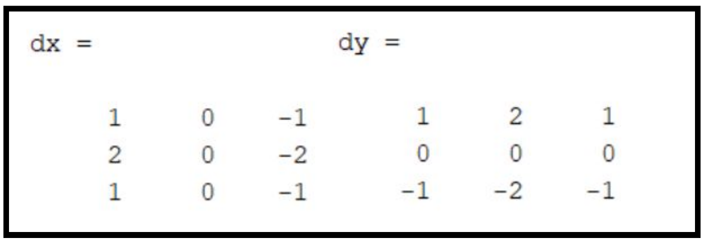
2. Procedure

In the first part of the experiment the six images are loaded, and their grayscale version is stored in a cell.
Then we select four patches in the first image: the first corresponds to the red car, while the other three are all centered around the black car, but with different sizes (i.e. number of pixels).
In order to correctly find the red and the black cars in all the other images, we wrote the function ***ComputeNCC***, which takes as input a cell containing all the six images and the template, that is the window around the car that we want to find in the images. A third input can also be passed to the function, in order to describe the type of the template used (e.g. in the case of the dark car, for which we want to discuss the results of three different-in-size templates). The function returns the **score maps**, (i.e. matrices containing correlation coefficients) in a cell, corresponding to the NCCs between the template and all the images.
Inside the body of ***ComputeNCC*** function, we carry out the *Normalized two-dimensional cross-correlation* between each one of the six images and the template, with the MATLAB function ***normxcorr2***, and display the obtained results. In the case

of the dark car template, we also measured (through MATLAB function *clock)* the computational time required to compute the NCC for all the six images, and then we estimated each single computational time by taking the average:

$$t = \frac{t_f - t_i}{N}$$

where the numerator is the total computational time expressed in seconds, and N is the number of NCC performed (one for each image).
Finally, we wrote a function called ***displayResults,*** which takes as input the NCC returned by ***ComputeNCC***, the template, and all the gray-images, and displays all the six images along with the car correctly identified. In order to do so, the position of the maximum of the score map was used and a rectangle with the size of the template was drawn around the car using MATLAB function ***rectangle***.
As regards the second part of the experiment, at first we loaded the *i235.png* image and we computed separately the x and y time derivatives with MATLAB function ***conv2;*** this was possible by convolving the image with *dx* and then *dy (Sobel Kernel).*
After that, we computed the square of the time derivative of the image.



```
dx =                         dy =

     1     0    -1              1     2     1
     2     0    -2              0     0     0
     1     0    -1             -1    -2    -1
```

figure 2.1: structure of dx (left), structure of dy (right)

Then, we designed a Gaussian filter through the MATLAB function *fspecial* and used it as the window function. Therefore, we convolved such filter with the products of derivatives, and we obtained a weighted sum of products of derivatives.
In order to carry out the corner detection, we wrote the function ***corner_detection,*** which takes as input the weighted sum of products of derivatives computed before and returns the ***R score map (cornerness)*** along with the ***corner regions map***. More specifically, this function uses its input to build the *M-matrix*, computes the R score map with the following formula:

$$R = det[M(\sigma_I, \sigma_D)] - \alpha[trace(M(\sigma_I, \sigma_D))]^2$$

and thresholds the corner response of all the points in R using the following relation:

$$R(x,y) > 0.3 * m$$

with m corresponding to the max value of R, thus obtaining the corner regions map.
Finally, we used the MATLAB function ***regionprops*** in order to show the centroids of the blobs in the corner regions map overlapped to the image itself.

3. Results

Figure 3.1 shows all the six original images, in their RGB version, where it is possible to notice the original positions of the red and the dark car that need to be identified.
Figure 3.2 and *figure 3.3* show the templates of the red and dark cars used in the NCC; more specifically, as shown in *figure 3.2*, we used only one template in order to identify the red car in the six images, and the results are shown in *figures 3.4-3.5*; while in case of the dark car, we used different templates (larger or smaller than the original one, *figures 3.2-3.3*). As regards the identification of the dark car, only the results in image 1 are included (*figure 3.8*, *figure 3.9* and *figure 3.10)*, since the principle is the same for all the other 5 images left.
Figure 3.11 shows the computational time (in seconds) required to perform the Normalized Cross-Correlation: it increases as the dimension of the template increases; this is a reasonable result, since having larger templates means having more pixels to analyze.

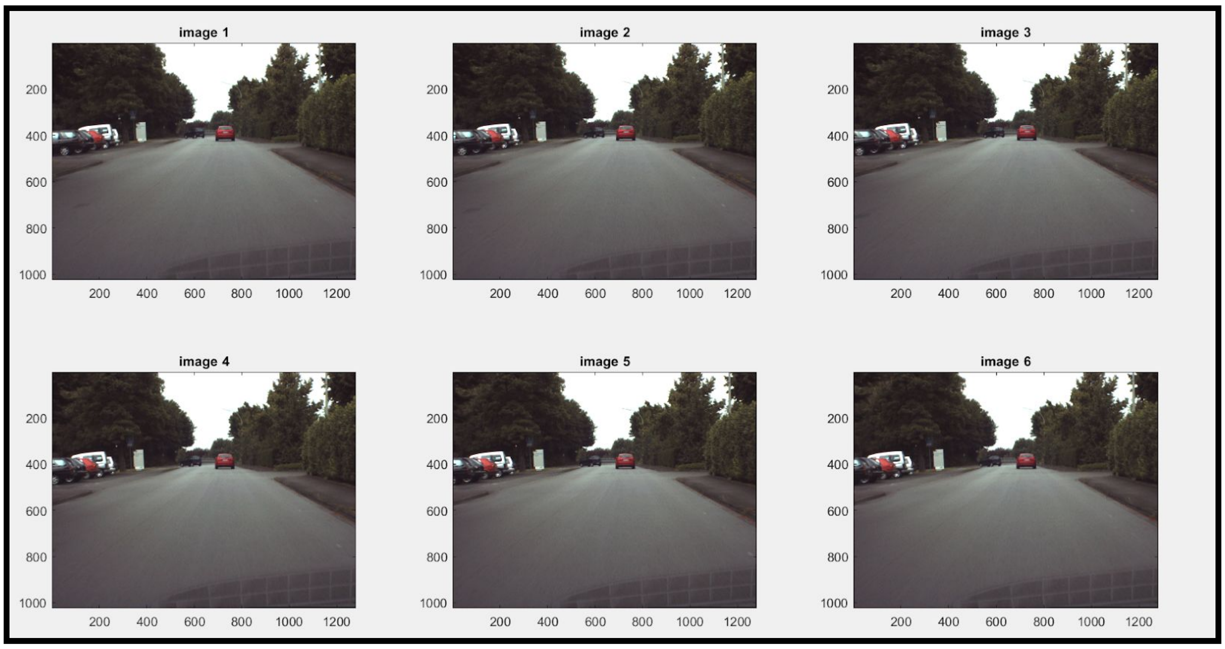


Figure 3.1 - Original RGB images

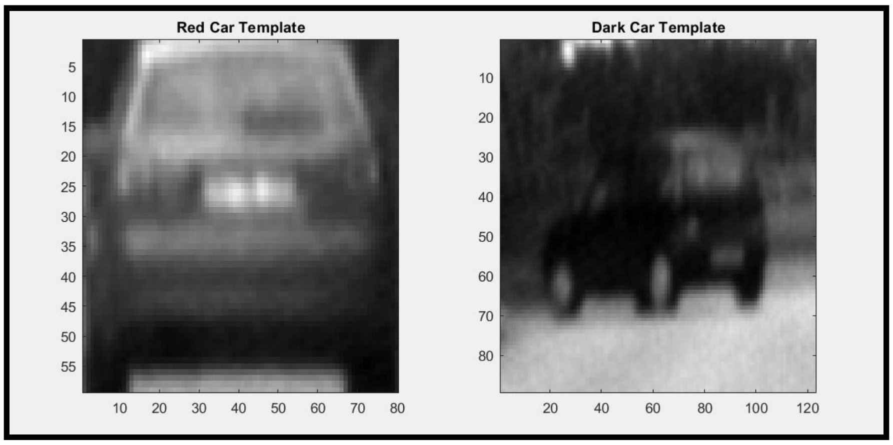


Figure 3.2 - Template of red car (left) and dark car (right)

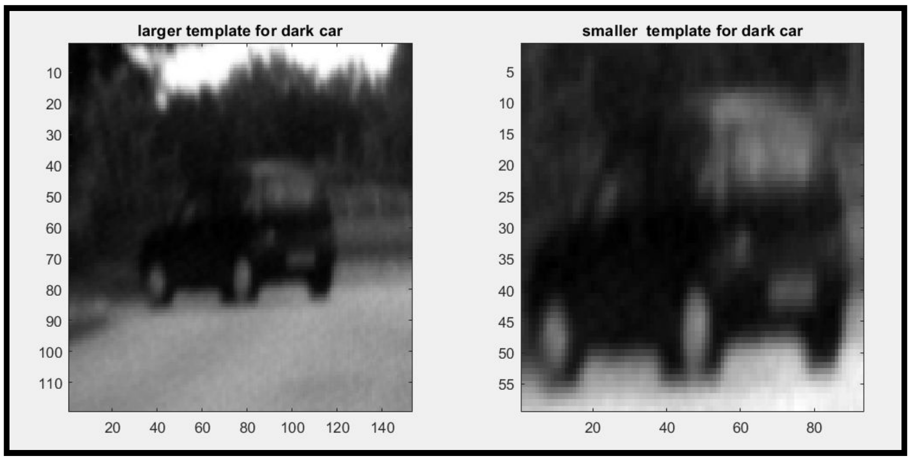


Figure 3.3 - Different-in-dimensions templates of the dark car

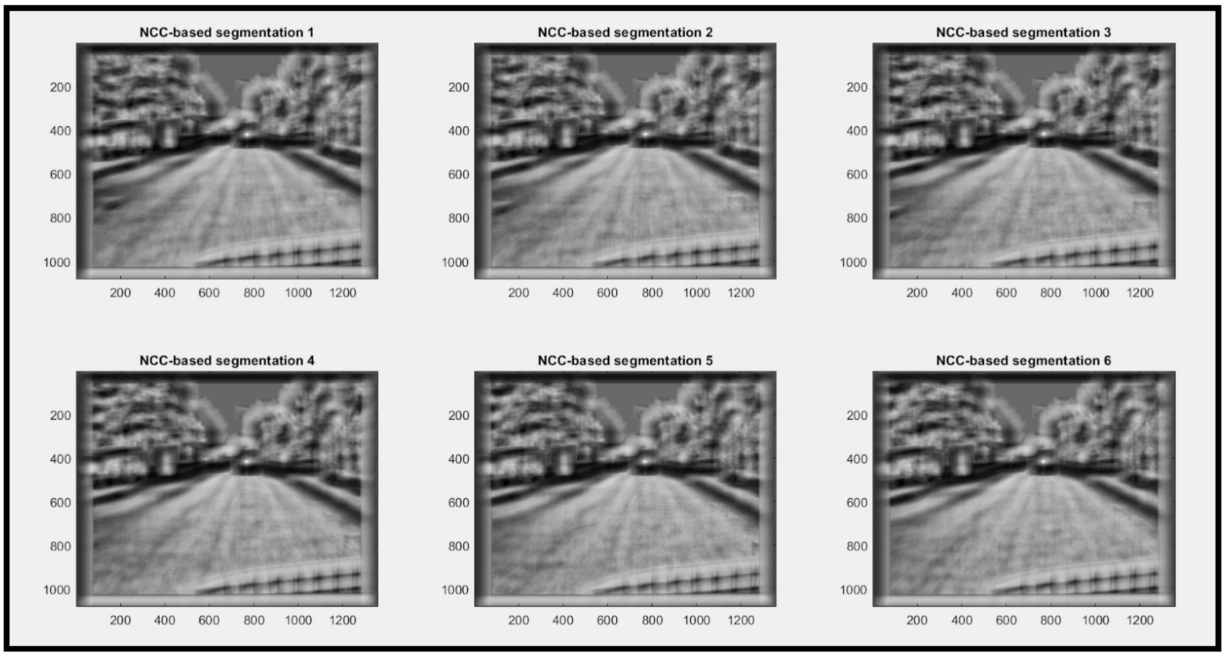


Figure 3.4 - NCC-based-segmentation of all six images with the red car (score map)

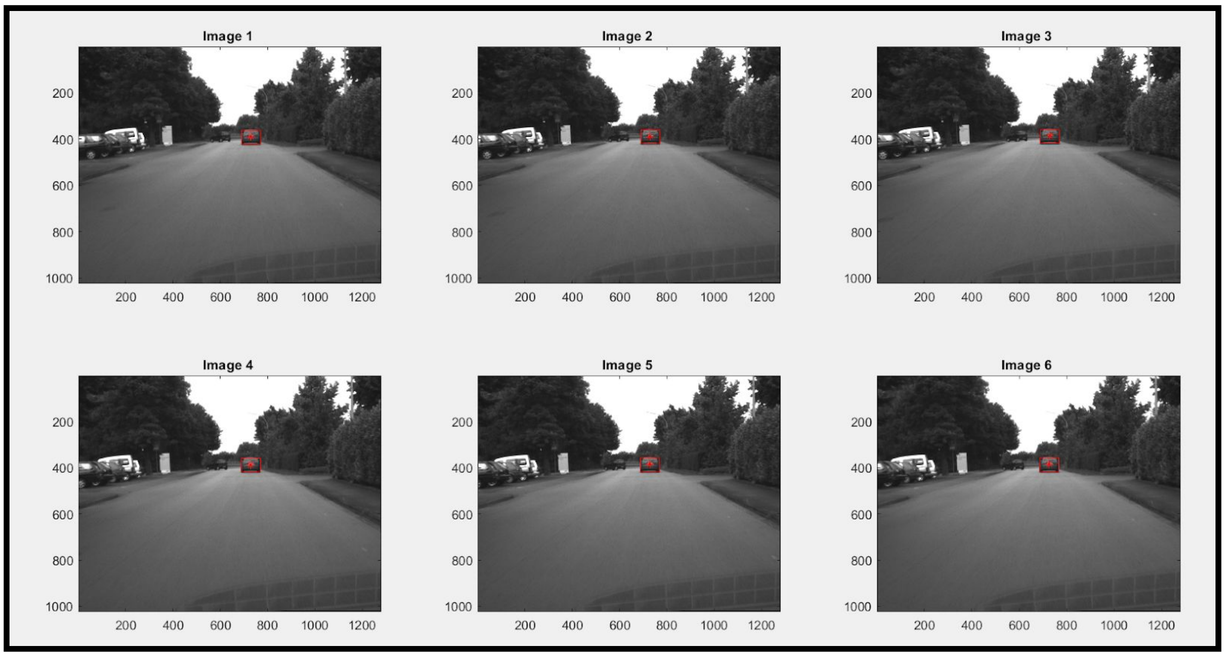


Figure 3.5 - Red car correctly detected in the six images

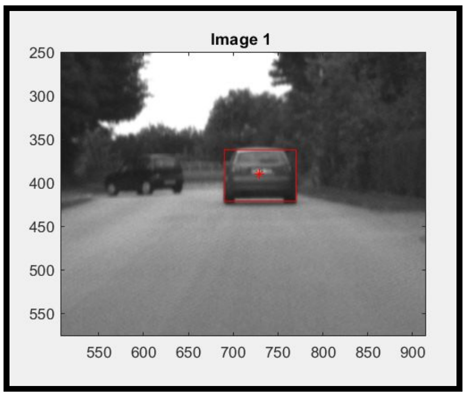


Figure 3.6 - Focus on image 1 with red car correctly detected

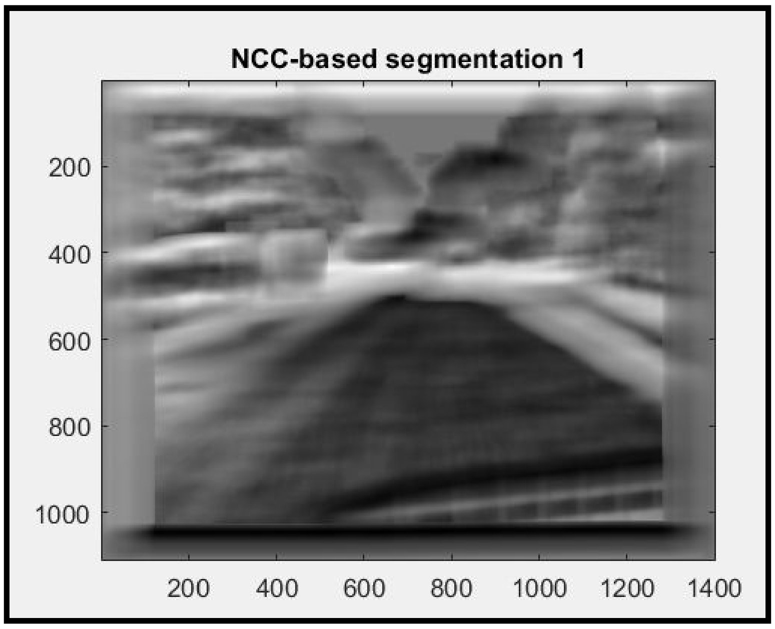


Figure 3.7 - NCC based segmentation of image 1 with the dark car (score map, standard template)

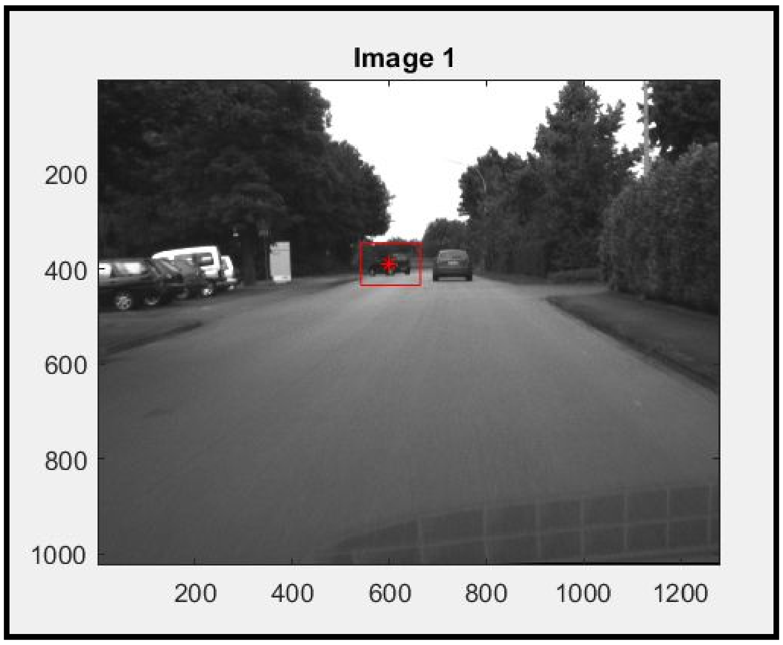


Figure 3.8 - image 1 with the dark car correctly detected (standard template)

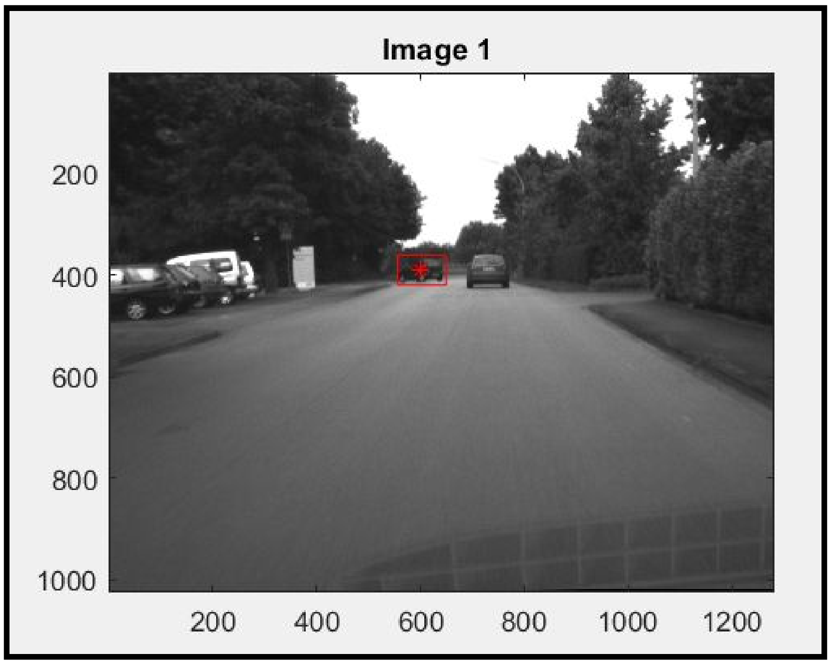


Figure 3.9 - image 1 with the dark car correctly detected (smaller template)

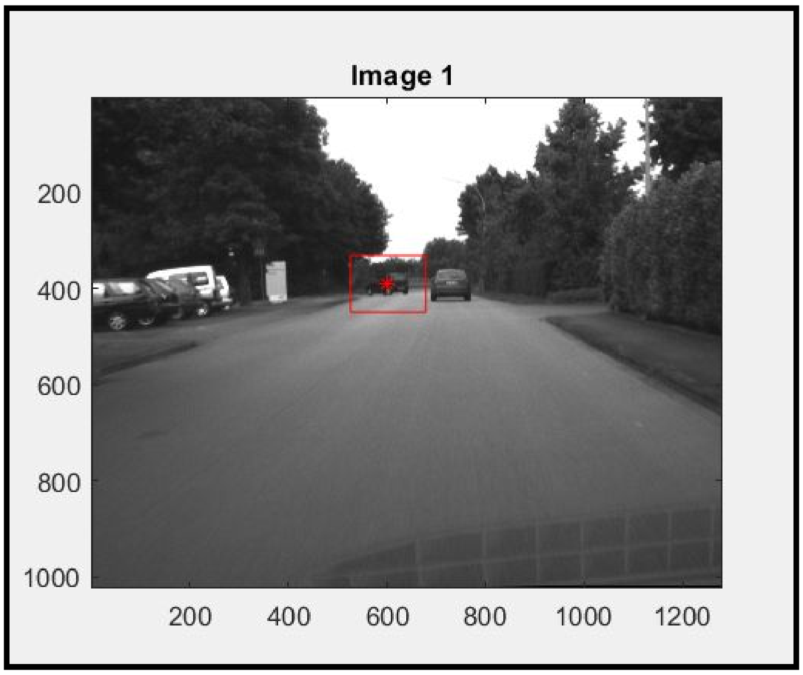


Figure 3.10: image 1 with the dark car correctly detected (larger template)

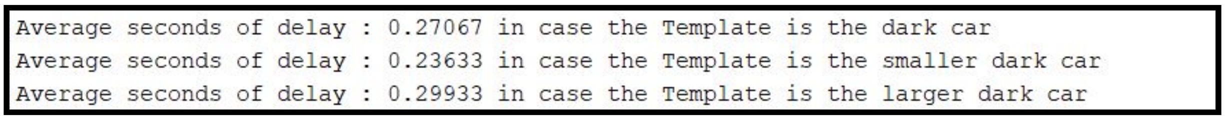

```
Average seconds of delay : 0.27067 in case the Template is the dark car
Average seconds of delay : 0.23633 in case the Template is the smaller dark car
Average seconds of delay : 0.29933 in case the Template is the larger dark car
```

Figure 3.11: computational time required to perform NCC

As regards the second part of the experiment, *figure 3.12* shows the original image, which will be subject to the corner detection.

In *figure 3.13* and *figure 3.14* we can see respectively the partial derivatives of the image and the square of partial derivatives, later used to compute the corner regions and the R score matrix; these results are shown in *figure 3.15.*
In *Figure 3.16* we can see the final result, after overlapping the image to the centroids of the blobs of all the detected corners.

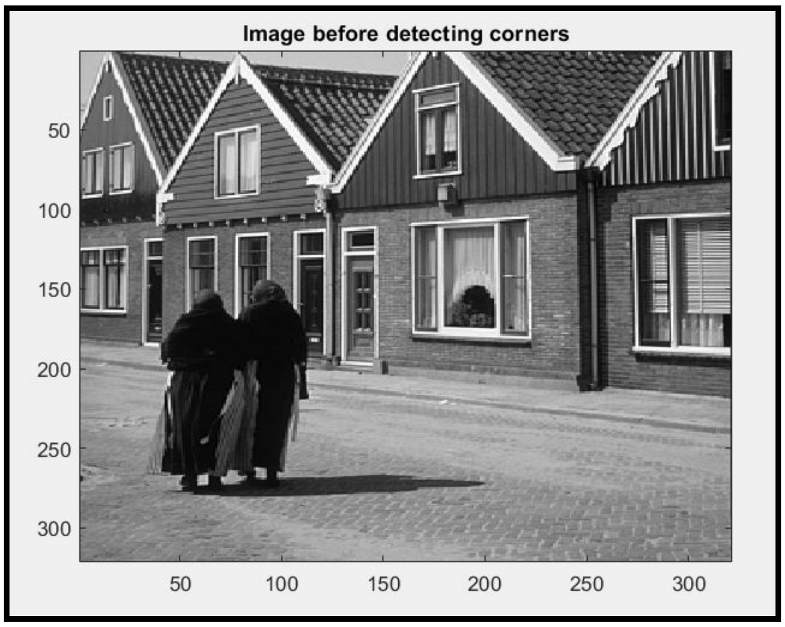


Figure 3.12 - Original image before detecting corners

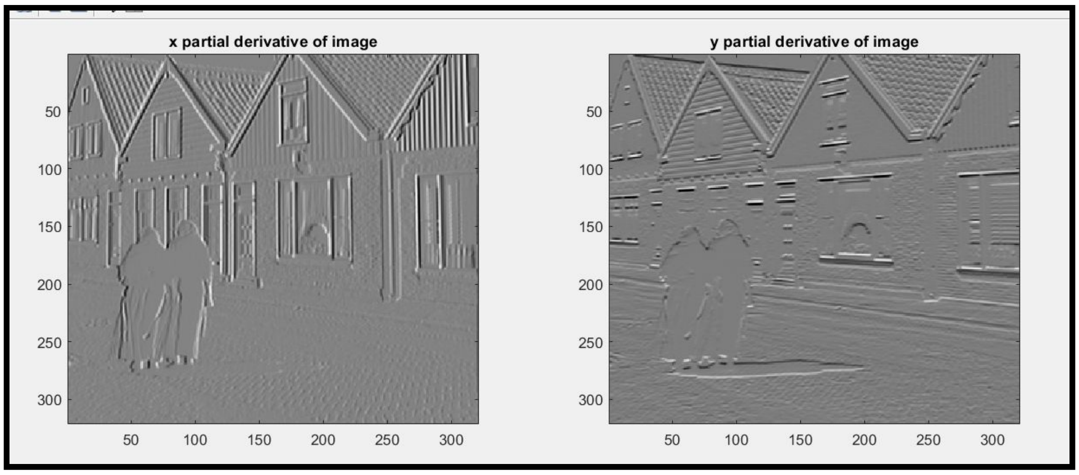


Figure 3.13 - Partial derivatives of the image

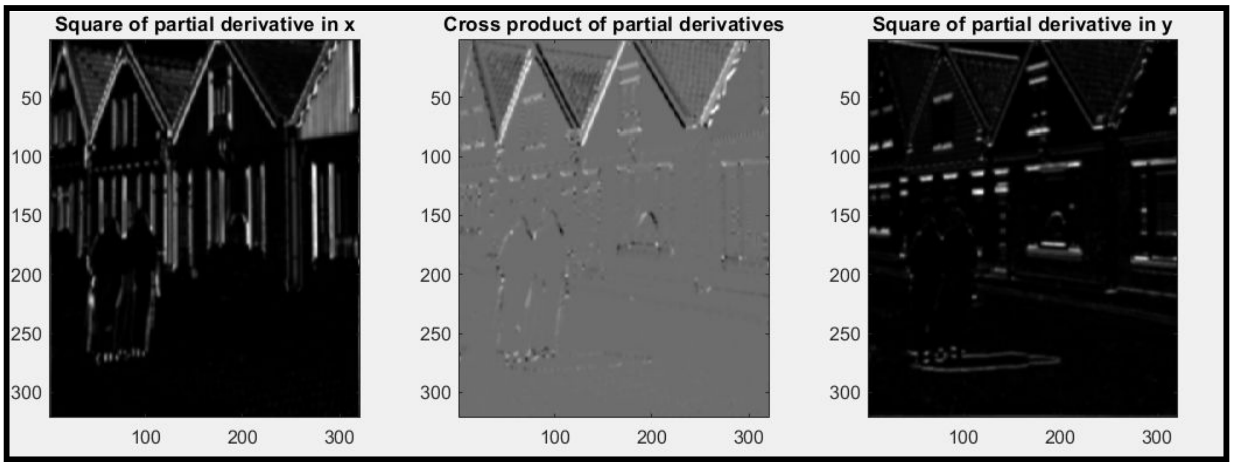


Figure 3.14 - Square of partial derivatives (left and right), cross product of partial derivative (middle)

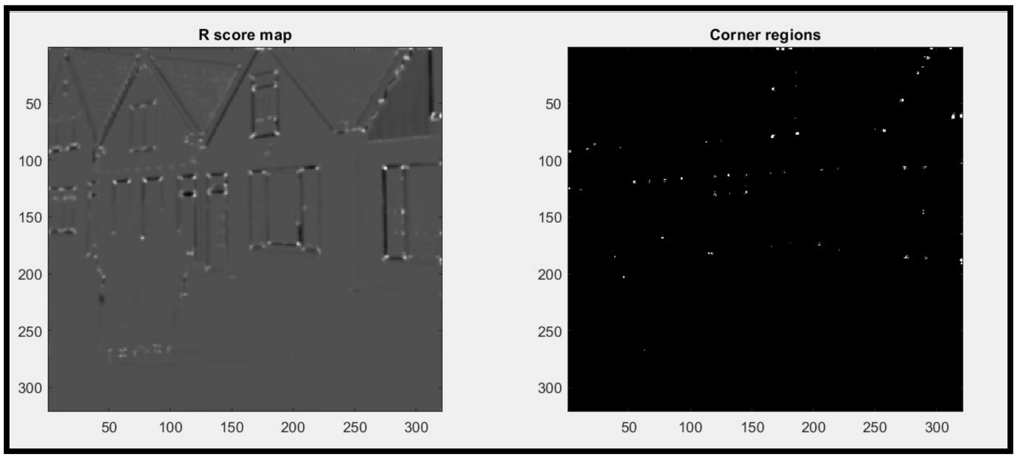


Figure 3.15 - R score map (left), Corner region (right)

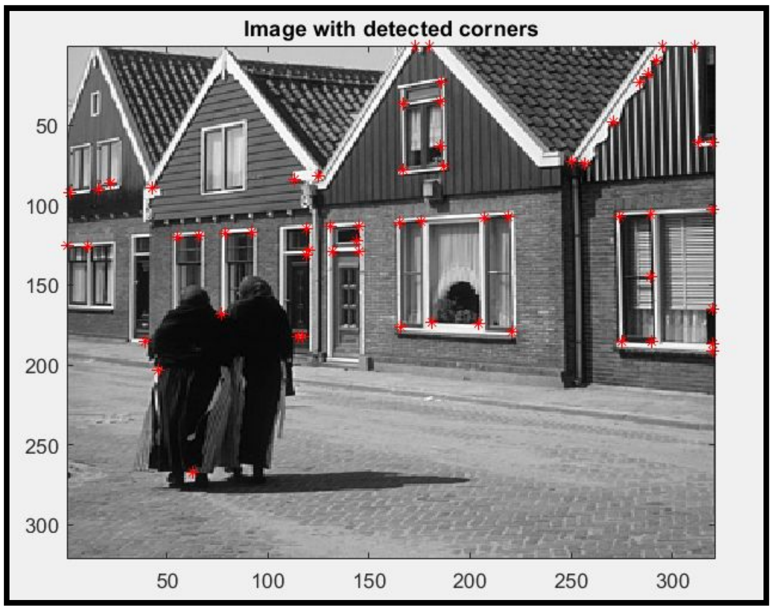


Figure 3.16 - Image with detected corners

4. Conclusions

The results in the first part of the experiment were coherent with the ones we expected: using an arbitrary template we were able to find both cars in all the six images.
Then, comparing the results related to the different-in-dimensions templates, we noticed how the computational time of the NCC is strictly related to the size of the template.
In particular, the NCC turns out to be a quite fast operation if we want to find a small area in the image, whereas, as we can observe from our experiments, the computational time quickly increases when increasing the size of the window.
As for the accuracy, the dark car was correctly found in all three cases, but of course we can deduce that by reducing too much the size of the window several problems may occur in the identification of template.
Therefore we can conclude that, especially in *real-time applications*, a *trade-off* between accuracy and computational time must be found for the choice of the size of the area.
Finally, by comparing the results we obtained using the NCC with the ones obtained in the previous Lab using the *color-based segmentation*, we can state that they are very similar. However, unlike the case of *NCC-based segmentation*, in the *color-based segmentation*, where the the red car had been found by using the color information in the Hue component of the image, a specific **threshold** had to be chosen and applied.
This may make the procedure used in the actual lab more precise and robust than the previous one, although perhaps more complex.
In the second part of the experiment, we can consider the final result as quite satisfactory, since the corners (i.e. the regions which have a significant change in all the directions) were correctly detected in the input image.
However, we can also notice that the algorithm was not able to detect all the corners.
This may be due to the complexity in the image, as well as to the choice of the threshold.