Laboratory Session 4

Computer Vision - Università Degli Studi di Genova

Filip Hesse
S4889393
filip_hesse@yahoo.de

Justin Lee
S4885003
leej1996@gmail.com

Steven Palma Morera
S4882385
imstevenpm.study@gmail.com

***Abstract*—Image segmentation based on the normalized cross correlation (NCC) of an image with an image patch is performed to detect two cars in a scene and a Harris corner detector is used to detect corners by finding the max R values.**

***Index Terms*—Image segmentation, normalized cross correlation, NCC, Gaussian Filter, Harris Corner Detector, Partial Derivatives**

I. Introduction

The purpose of this lab [2] is to implement NCC-based segmentation and a Harris corner detector. Task 1 had six different versions of Fig 1 , which were all very similar to each other. Cross-correlation of two patches containing one car each was applied with the original images in order to obtain the position of the patch in the image. Task 2 involved using a Harris corner detector derived from lecture [1] to detect the corners in the image, superimpose them on the image and also show a map of the R values.

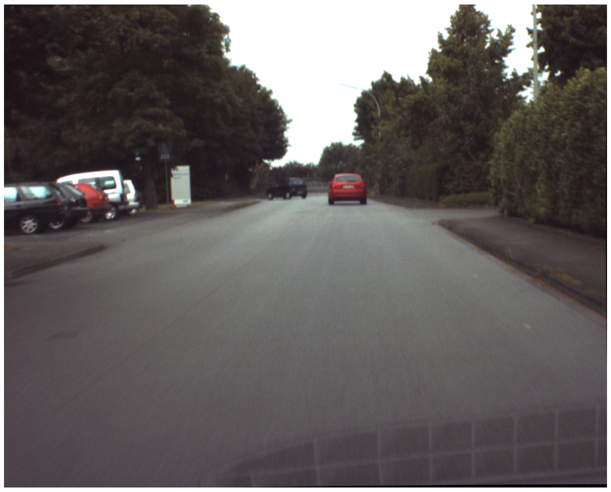


Fig. 1: Test image of cars for segmentation. Source: [2]

II. Procedure

A. Task 1

From the first image of the series (Fig 1), patches of cars were extracted. The function normxcorr2 from MATLAB was used to perform normalized cross correlation of the patches with all six images. This function first subtracts the mean of the patch intensities and then divides them by their standard deviations. This prevents the algorithm from obtaining high cross-correlation scores for areas of similar shapes but different contrasts or shifted intensities. The result of normxcorr2 is a scoremap, which contains high values for areas with high similarity with the patch and low values for low similarities. The maximum of the obtained score map shows the position of the patch in the original image. Accounting the different sizes of the scoremap and the original image due to padding by the normxcorr2 algorithm, the position of the maximum of the scoremap is transferred to the original images. A central point and a bounding box for the detected patch are displayed in the according positions. The entire procedure is repeated with many different sizes of patches of the dark car while measuring the time required to perform the normxcorr2-function. This way, some insight about appropriate patch sizes can be found.

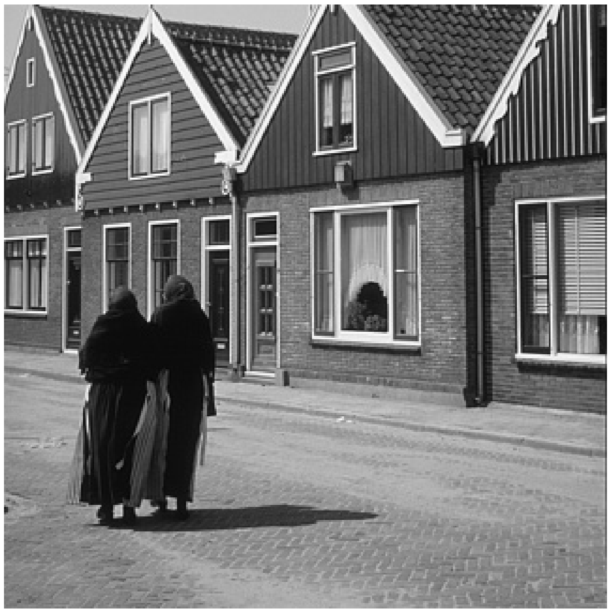


Fig. 2: Test image of village for corner detection. Source: [2]

B. Task 2

The Harris corner detector used works as follows. First of all, it computes the partial derivative in x and y directions for each pixel of the image. These values are then multiplied, summed together, convolved with a Gaussian filter and arranged in matrix form to compute the structure tensor (M) as seen in Fig. 3. The Gaussian filter is required in order to get a representative value of the gradient around a point.

$$M(\sigma_I, \sigma_D) = g(\sigma_I) * \begin{bmatrix} I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\ I_x I_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

Fig. 3: Second moment matrix (M) computed from image derivatives. Source: [1]

Once the M matrix is calculated, the corner response measure (R) value for each pixel is found using the formula below, where M is the second moment matrix and α is a constant value, 0.05 in this case.

$$R = det(M) - \alpha \times trace(M)^2$$

Non maxima suppression is then used to find the corner regions of the image. The regionprops() function from MATLAB is used to find the centroids of these areas and then they are superimposed on the image.

III. Results and Analysis

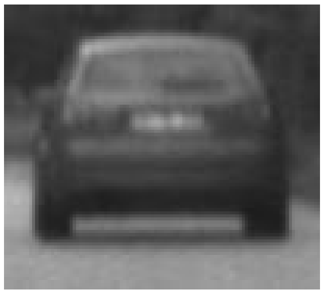
A total of 195 images were obtained after running the Lab5.m script in the MATLAB workspace as follows:

Lab5();

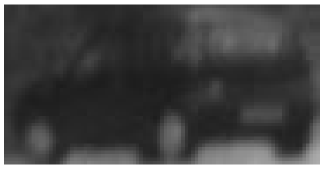
The following images and analysis shown are the most useful ones for understanding the effect of each task from the procedure. However, the reader can access the full set of images in the folder called "plots", automatically generated when the script is executed.

A. Task 1

Fig. 4 shows the two patches of the cars, which were used for the NCC-based segmentation in the first place.



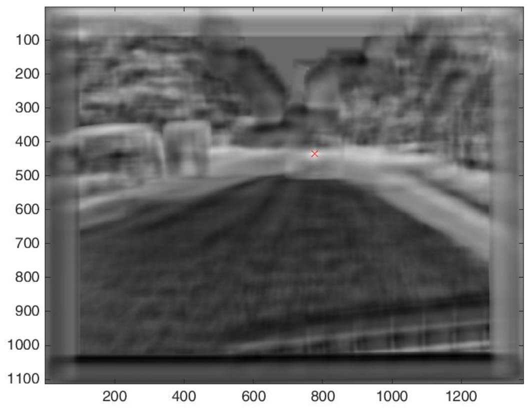
a) Red Car



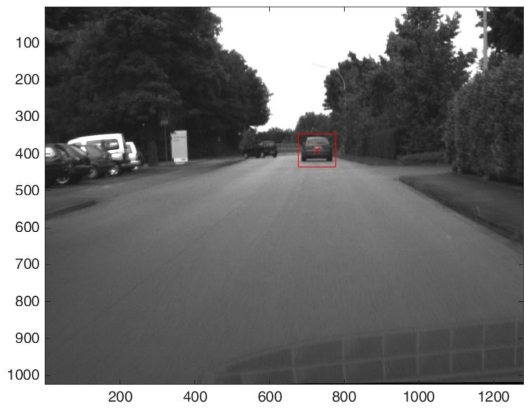
b) Dark Car

Fig. 4: Patches for NCC-Segmentation

Obviously, when correlating with a patch obtained from the image itself, the algorithm certainly finds the cars in the correct place, because there is a 100% match of the pixels in the patch and some pixels in the image and the exact same patch does not exist twice in the image. For this case, it is almost irrelevant, how big the patch is, as long as it is bigger than a few pixels. Consequently, the interesting part is the application of the patches on images which slightly change. The two first



a) Scoremap red car

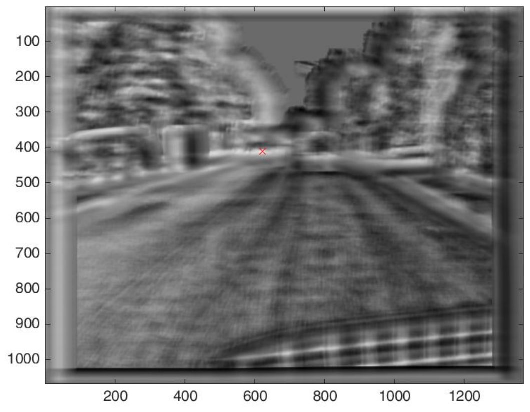


b) Segmentation of red car

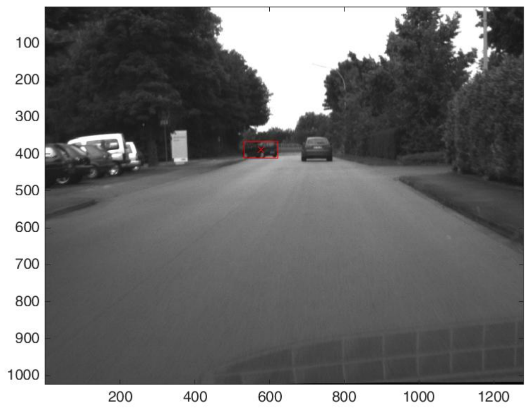
Fig. 5: Scoremaps and Segmentation of red car. Source: Own Elaboration

patches shown in Fig. 4 are applied on all the images from the series and the cars are always found in the correct place. The scoremaps are displayed in the figures 5a and 6a. The maximum value is marked with a red x. The images 5b and 6b show the original images with bounding boxes around the detected objects.

Comparing this method to color-based segmentation, like it was performed in the previous assignment, it can be stated, that in our case both approaches lead to a satisfying solution. The difference between the methods is, that color-based segmentation only considers to color of an image and completely ignores contrasts or gradients in the image. NCC-Segmentation on the other hand completely ignores the colors and only explores the contrasts of an image. Color based segmentation is covariant to scaling of the desired object, while NCC would need to be adjusted to different sizes by correlating with patches of different sizes, which is computationally more complex. For a more robust solution, the combination of the two approaches could be used: Apply both algorithms and



a) Scoremap dark car



b) Segmentation of dark car

Fig. 6: Scoremaps and Segmentation of dark car. Source: Own Elaboration

find more candidates for possible solutions with assigned probabilities (e.g. not only find one maximum of the scoremap, but multiple local maxima). Another algorithm could decide, which of the two solutions is the correct one based on the probabilities of their correctness.

In order to discuss the size of the window which is used for cross correlation in terms of computation time and accuracy of the detection, many correlations with different window sizes are performed. The aspect ratio of the window remains constant: $width/height = 2$. Starting from a very small window of 8 by 4 pixels, the window is increased in a loop by 8 pixels width and 4 pixels height 44 times. All the obtained windows are used to perform the normalized cross correlation and the computational time is measured using the Matlab functions *tic* and *toc*.

Surprisingly, the duration of the function normxcorr2 grows very slowly when increasing the filtersize. The relation can be shown in the figure 7. When the width of the window is eight-folded, the computational time is not even the double time.

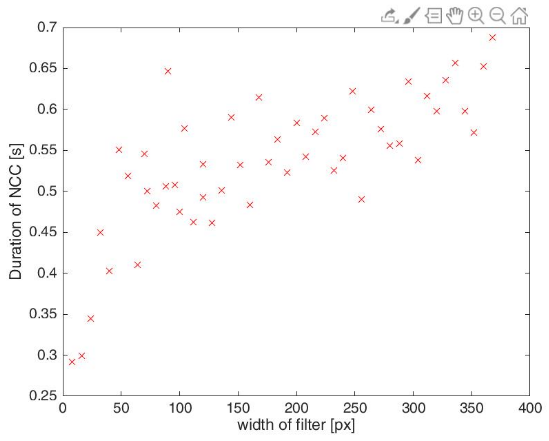
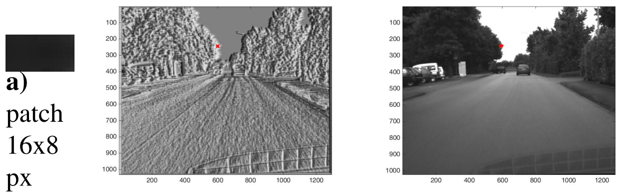


Fig. 7: Computational time over width of window. Source: [2]

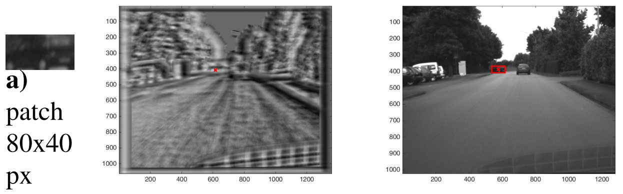
Consequently the cost to increase the window size is not very high.



a) patch 16x8 px

b) Scoremap **c)** Segmentation

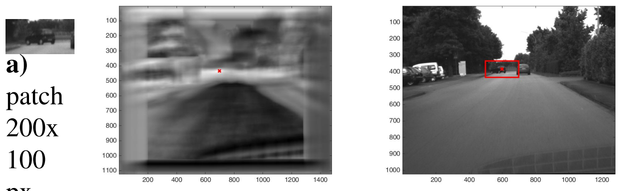
Fig. 8: Too small window for cross correlation



a) patch 80x40 px

b) Scoremap **c)** Segmentation

Fig. 9: Good size for cross correlation



a) patch 200x 100 px

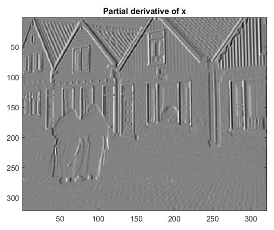
b) Scoremap **c)** Segmentation

Fig. 10: Too big window for cross correlation

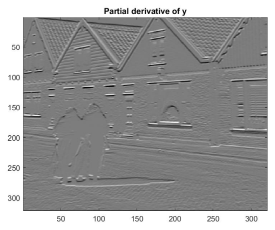
The figures 8, 9, 8 show, that the window size matters for obtaining a good segmentation result. The window for cross correlation can not be too small, or the patch will be recognized in other areas of the image, because it has too few characteristic features. If it is too big (e.g 100x200 px), then the background behind the car gains more importance with respect to the car, so the algorithm won't find the car in the image but the are with the same background. Consequently, when the car moves, the algorithm will always find the same are in the image. For these reasons, it is important to choose an appropriate window size like 80x40 pixels, which contains a lot of the car but not much of the background.

B. Task 2

Below, different image will be shown to display the process the Harris corner detector uses in order to detect corners. In Fig. 11 the partial derivatives in the x and y direction of each pixel are shown.



a) x direction



b) y direction

Fig. 11: Partial derivative in x and y direction

Fig. 12 is the Gaussian filter used to generate the M matrix for each pixel.

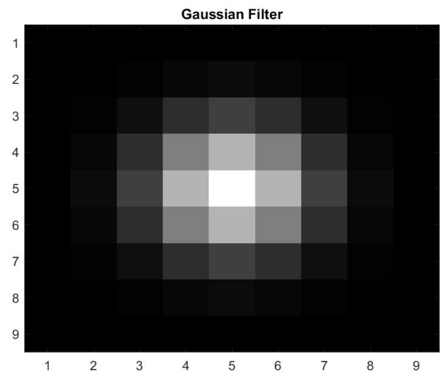


Fig. 12: Gaussian Filter used to find M matrix. Source: Own Elaboration

Fig. 13 shows the R values of each pixel on the image displayed with the Jet colormap. To move from this to Fig. 14, non maximum suppression was done to get the maximums. The threshold value was $0.3 \times R_m$, where R_m is the max R value in the whole image.

As can be seen in Fig. 15, almost all corners were detected in the image. Unfortunately, some were missed, for example in the top left corner, the peaks of the houses are not detected.

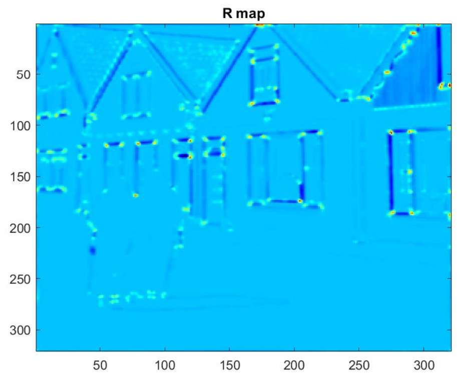


Fig. 13: Rmap of image. Source: Own Elaboration

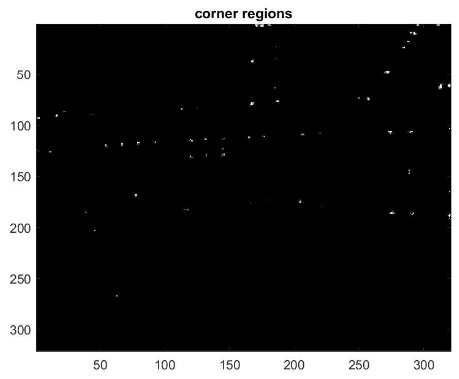


Fig. 14: Corner regions of image. Source: Own Elaboration

To detect these corners that were missed, the threshold value could be lowered but this would run the risk of including regions which are not corners.

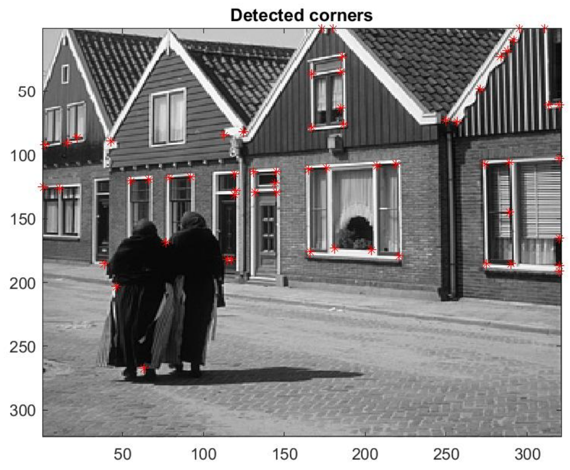


Fig. 15: Centroids of corners superimposed on image. Source: Own Elaboration

IV. Conclusions

- It is not very computationally expensive to increase the window size NCC-segmentation
- Appropriate size of the window for NCC-segmentation (neither too small nor too large) is crucial for good results.
- The Harris corner detector works well for detecting corners. Main problem is choosing proper thresholding value.

References

[1] Solari, F. Part 5 - corners, 2020. Retrieved from: https://2019.aulaweb.unige.it/pluginfile.php/209267/mod_resource/content/3/lecture5_corner.pdf

[2] Solari, F. Lab Session n.5, 2020. Retrieved from: https://2019.aulaweb.unige.it/pluginfile.php/209269/mod_resource/content/2/undefined/Lab5.pdf