Exercise 2: Harris Corner Detection

1 Harris Corner Detection Summary

A corner in an image can be defined as the intersection of two or more edges. Corners are features with high repeatability.

1.1 The basic concept of corner detection

One of the earliest corner detectors was invented by Moravec [3]. He defined a corner as a point where there is a large intensity variation in every direction. An intuitive explanation of his corner detection algorithm is given in Figure 1. Intuitively, one could recognize a corner by looking through a small window centered on the pixel. If the pixel lies in a "flat" region (i.e., a region of uniform intensity), then the adjacent windows will look similar. If the pixel is along an edge, then adjacent windows in the direction perpendicular to the edge will look different, but adjacent widows in a direction parallel to the edge will result only in a small change. Finally, if the pixel lies on a corner, then none of the adjacent windows will look similar. Moravec used the Sum of Squared Differences (SSD) as a measure of the similarity between two patches. A low SSD score indicates more similarity. If this number is locally maximal, then a corner is present.

1.2 The Harris corner detector

Harris and Stephens [2] improved Moravec's corner detector by considering the partial derivatives of the SSD score instead of using shifted windows.

Let I be a grayscale image. Consdering taking an image patch centered on (x, y) and shifting it by $(\Delta x, \Delta y)$. The Sum of Squared Differences (SSD) between these two patches is given by:

$$SSD(x, y) = \sum_{x,y \in P} (I(x, y) - I(x + \Delta x, y + \Delta y))^2. \tag{1}$$

$I(x + \Delta x, y + \Delta y)$ can be approximated by a first-order Taylor expansion. Let I_x and I_y be the partial derivatives of I, such that

$$I(x + \Delta x, y + \Delta y) = I(x, y) + I_x(x, y)\Delta x + I_y(x, y)\Delta y. \tag{2}$$

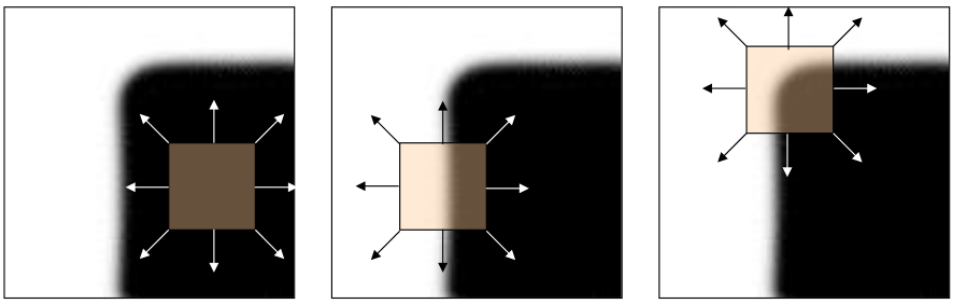
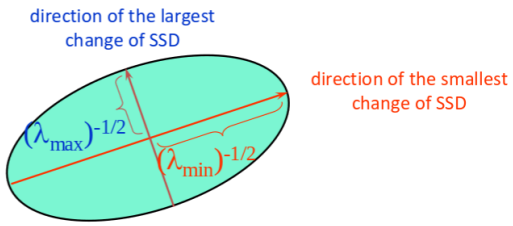
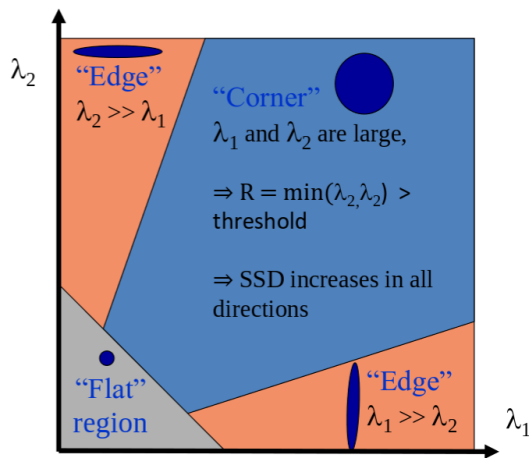


Figure 1: Illustration of flat regions (left), edge regions (middle) and corner region (right).



(a) This ellipse is built from the second moment matrix and visualizes the directions of fastest and lowest intensity change.



(b) The classification of corner and edges according to Harris and Stephens.

This produces the approximation

$$SSD(x,y) \approx \sum_{x,y \in P} (I_x(x,y)\Delta x + I_y(x,y)\Delta y)^2, \tag{3}$$

which can be written in matrix form:

$$SSD(x,y) \approx \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} M \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}, \tag{4}$$

where M is the second moment matrix

$$M = \sum_{x,y \in P} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \tag{5}$$

and since M is symmetric, we can rewrite M as

$$M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R, \tag{6}$$

where λ_1 and λ_2 are the eigenvalues of M.

As mentioned before, a corner is characterized a by a large variation of the SSD score in all directions of the vector $(\Delta x, \Delta y)$. The Harris detector analyzes the eigenvalues of M to decide if we are in presence of a corner or not. Let us first give an intuitive explanation before showing the mathematical expression.

Using equation (4) we can visualize M as an ellipse (Figure (2a)) of equation

$$\begin{bmatrix} \Delta x & \Delta y \end{bmatrix} M \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \text{const.} \tag{7}$$

The axis length of this ellipse are determined by the eigenvalues of M and the orientation is determined by R. Base on the magnitudes of the eigenvalues, the following inferences can be made based on this argument:

- If both λ_1 and λ_2 are small, the SSD score is almost constant in all directions (i.e., we are in presence of a flat region).
- If either $\lambda_1 \gg \lambda_2$ or $\lambda_2 \gg \lambda_1$, we are in presence of an edge: the SSD score has a large variation only in one direction, which is the one perpendicular to the edge.

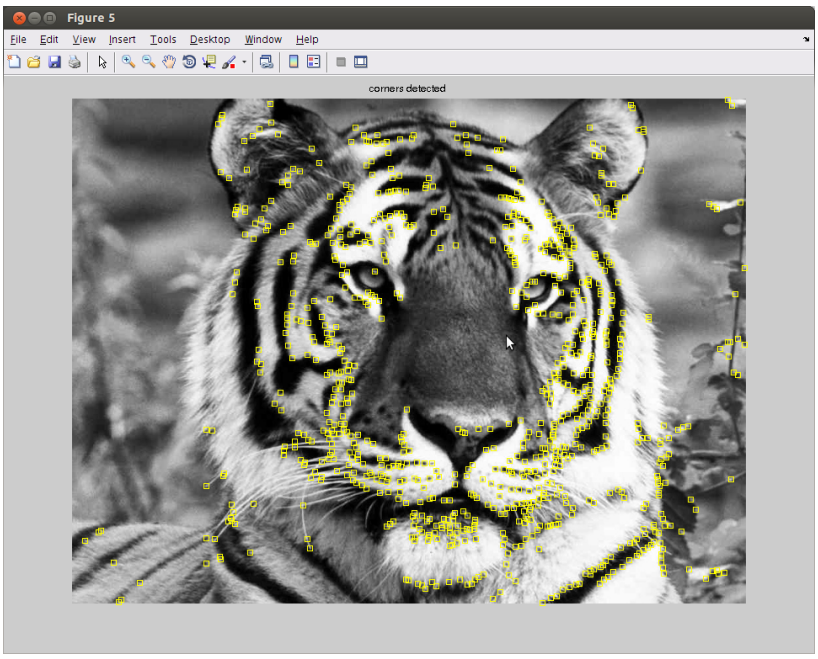


Figure 2: Result of Harris detector.

- If both λ_1 and λ_2 are large, the SSD score has large variations in all directions and then we are in presence of a corner.

The three situation mentioned above are pictorially summarized in Figure 2b.

Because the calculation of the eigenvalues is computationally expensive, Harris and Stephens suggested the use of the following "cornerness function" instead:

$$R = \lambda_1\lambda_2 - \kappa(\lambda_1 + \lambda_2)^2 = \det(M) - \kappa \text{ trace}^2(M), \tag{8}$$

where κ is a tunable sensitivity parameter. This way, instead of computing the eigenvalues of M, we just need to evaluate the determinant and trace of M. The value of κ has to be determined empirically. In the literature, values are often reported in the range of $0.04 - 0.15$.

The last step of the Harris corner detector consists in extracting the local maxima of the cornerness function, using "nonmaxima suppression". Nonmaxima suppression involves revisiting every pixel of the cornerness function and determining whether or not it is at a local maximum.

2 Exercises

In these exercises, we will implement the Harris corner detector from scratch in MATLAB. Therefore, download the zip file from the website http://rpg.ifi.uzh.ch/docs/teaching/Lab_Exercise_2_Harris.zip.

2.1 Implement Harris corner detector

Open the file `run_harris.m`. You can see that this file calls the function `harrisCorners.m`; however, this function is not completely implemented and it is your task to fill in the missing lines. The comments in the code will guide you. The final result should look like in Figure 2. Try to understand *every line of code* and use the `help` command if you don't know what a function does.

To better understand the algorithm described in Section 1, you can run your code on the provided `checkerboard.png` and try to understand the intermediate output for different kinds of regions (i.e., flat regions, edges and corners). Note that you should be able to observe that the corners detected do not lie exactly on the checkerboard crossings and there are multiple corners detected at nearby positions. Try to understand why this would happen based on the characteristics of the checkerboard pattern (Figure. 4). We will come back to this in Section 2.4.

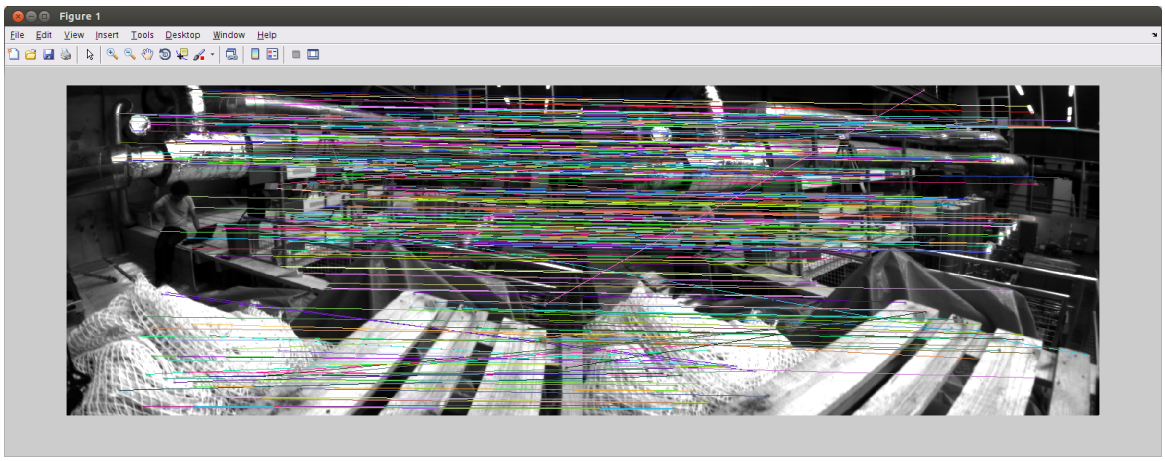


Figure 3: Result of the matching function.

2.2 Change the parameters

There are two important parameters for the Harris detector: the patch size and the κ. Try to change these parameters in `run_harris.m` and observe the changes in the output.

The parameter κ is also known as the sensitive factor. With a smaller κ, the the algorithm is more likely to detect corners (i.e., weak corners will also be detected). The window size defines the area in which the image gradients are considered. To correctly detect corners, the window size must be properly set according to the scales of the corners. Thus, the Harris detector is not scale invariant. We will have a look at scale invariant detectors in Section 2.5.

Note that for the checkerboard pattern, the corners detected are somehow not affected by the window size. Why are these crossings, to some degree, robust to the window size change?

2.3 Run matching using detected corners

As soon as your Harris detector works, we can use it to find point correspondences in two images. Therefore, run the script `run_matching.m`. This script uses your detector to find corners in two images that are recorded with a stereo camera. For every corner in the left image, it tries to find the corresponding corner in the right image. The result should look like in Figure 3. Read the code and try to understand how it works. In the next lecture, you will learn how to identify the outliers in your matches.

2.4 Achieve subpixel accuracy

Digital images are represented by individual pixels. Therefore, a corner does not necessarily locate on a specific pixel and, instead, it may lie between the discrete postions of several pixels. For example, in the checkerboard pattern used in this exercise, pixels near a crossing would look like Figure.4. This is the reason for which our implementation of Harris detector can only find values that are near the exact positions of the corners.

In many computer vision tasks such as camera calibration, it is desired to achieve a higher accuracy and this can be done by subpixel optimization. Uncomment the last section in `run_harris.m` and run the script again on the checkerboard pattern. You should be able to observe that now the corners lie exactly on the crossings. Try to understand `cornerSubPix.m` if you have time.

2.5 Scale invariant detectors

As can be observed from the exercise 2.2, the Harris detector is not scale invariant. Fortunately there are several detectors that can handle features of different scales. Run `compare_detectors.m` to observe the output of the Harris detector and the SURF (Speeded Up Robust Features) detector

0	0	1	1
0	0	1	1
0	0	1	1
0	0	1	1
0	0	1	1
1	1	0	0
1	1	0	0
1	1	0	0
1	1	0	0
1	1	0	0

(a) Pixel pattern

(b) Without subpixel optimization (c) After subpixel optimization

Figure 4: Subpixel optimization for a checkerboard crossing.

Figure 5: Output of SURF detector.

[1]. The result should look like Figure. 5. The circles around the detected features stand for the scales of the features.

Matlab also provides the implementations of several other detectors (BRISK, FAST, etc.). Try to implement the code to detect features with these detectors.

References

[1] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer vision–ECCV 2006*, pages 404–417. Springer, 2006.

[2] Chris Harris and Mike Stephens. A combined corner and edge detector. In *In Proc. of Fourth Alvey Vision Conference*, pages 147–151, 1988.

[3] Hans Moravec. Towards automatic visual obstacle avoidance. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, page 584, August 1977.