ELE510 Image Processing with robot vision: LAB, Exercise 3, Histogram and point transformations

Purpose: *To learn about the image histogram, histogram equalization and image noise.*

The theory for this exercise can be found in chapter 3 of the text book [1]. Supplementary information can found in chapter 1, 2 and 3 in the compendium [2]. See also the following documentations for help:

- OpenCV
- numpy
- matplotlib

IMPORTANT: Read the text carefully before starting the work. In many cases it is necessary to do some preparations before you start the work on the computer. Read necessary theory and answer the theoretical part frst. The theoretical and experimental part should be solved individually. The notebook must be approved by the lecturer or his assistant.

Approval:

The current notebook should be submitted on CANVAS as a single pdf file.

To export the notebook in a pdf format, goes to File -> Download as -> PDF via LaTeX (.pdf).

Note regarding the notebook: The theoretical questions can be answered directly on the notebook using a *Markdown* cell and LaTex commands (if relevant). In alternative, you can attach a scan (or an image) of the answer directly in the cell.

Possible ways to insert an image in the markdown cell:

```
![image name]("image_path")
```

```
<img src="image_path" alt="Alt text" title="Title text" />
```

Under you will find parts of the solution that is already programmed.

You have to fill out code everywhere it is indicated with ...

The code section under $\#\#\#\#\#\#\#\#a)$ is answering subproblem a) etc.

Problem 1

The histogram for an image of a **black** and **white** football and **grey** background is $[0, 520, 920, 490, 30, 40, 5910, 24040, 6050, 80, 20, 80, 440, 960, 420, 0]$, where 16 gray levels are used.

The diameter of the football is $d = 230\text{mm}$.

Use these information to find the pixel size, $\Delta x = \Delta y$.

Describe the steps to arrive to the solution.

▶ **Click here for an optional hint**

Step 1 what we know

4 left side of the histogram is black. 4 right side is white. In between is the football.

number_of_pixels = 0 + 520 + 920 + 490 + 440 + 960 + 420 + 0 = 3750

d = 230 mm

$\Delta x = \Delta y$

The area of the ball is given by: $A = \pi \cdot (d/2)^2$ == (number_of_pixels · area_single_pixel)

Step 2 Calculation

$A = \pi \cdot (d/2)^2$ == (number_of_pixels · area_single_pixel)

$A = \pi \cdot (d/2)^2$

$$\Delta x = \sqrt{\frac{\pi \cdot (\frac{d}{2})^2}{numberofpixels}}$$

$$\Delta x = \sqrt{\frac{\pi \cdot (\frac{230mm}{2})^2}{3750}} = 3.33mm = \Delta y$$

The solution of problem 1 should be:
Δx = Δy = 3.33mm or, if you have taken different level of gray into consideration, Δx = Δy = 3.22mm

Problem 2

For images, such as `./images/christmas.png` , some processing is normally desired to improve the contrast. The simplest approach for doing this is called histogram stretching. For a given image, where the used pixel values range from g_{min} to g_{max} we can spread these so they cover the entire $[0, G-1]$ range. The formula for histogram stretching is given by:

$$g_{\text{new}} = \left\lfloor \frac{g_{\text{old}} - g_{\text{min}}}{g_{\text{max}} - g_{\text{min}}} G + 0.5 \right\rfloor$$

Where g_{old} is an old pixel value, and g_{new} a new pixel value.

a) Make a small Python function, taking an image as input, perform histogram stretching using the previous equation, and giving the increased contrast image as output. Use an 8-bit grayscale range (G=255). Show and explain the result using `./images/christmas.png` as an example.

In [109…
```
# Import the packages
```

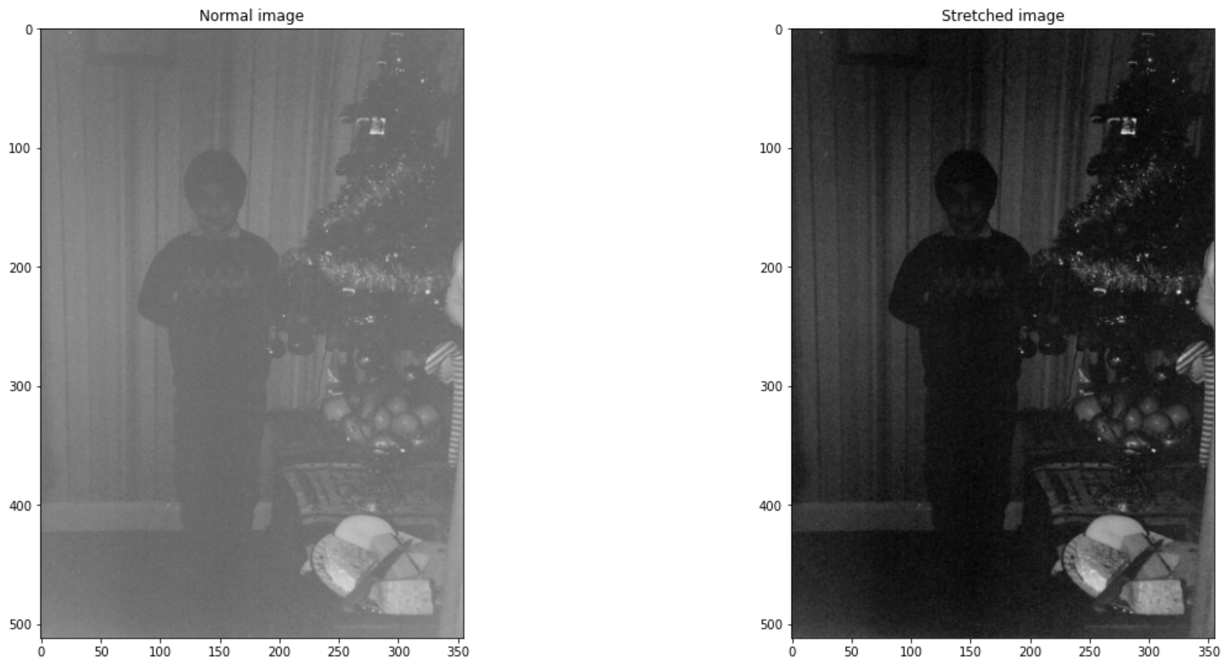
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

In [110…

```
"""
Function that takes in input an image and return the same image stretched.
"""

g_min = np.min(img)
g_max = np.max(img)

def histogram_stretch(img):

    g_new = np.array(((img-g_min)/(g_max-g_min))*255+0.5).astype(np.uint8)

    return g_new
```

In [111…

```
# Read the image and convert it to RGB channels
img = cv2.imread('christmas.png', 0)
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

# Use the function to improve the contrast of the image
img_stretch = histogram_stretch(img)

plt.figure(figsize=(20,20))
plt.subplot(221)
plt.imshow(img)
plt.title('Normal image')
plt.subplot(222)
plt.imshow(img_stretch)
plt.title('Stretched image')
plt.show()
```



Problem 3

In this experiment we use **four** images. The two first are gray level images, `./images/pout.jpg` and `./images/tire.jpg` . The other two are colour images captured with a standard digital camera. We want to study image enhancement with histogram equalization.

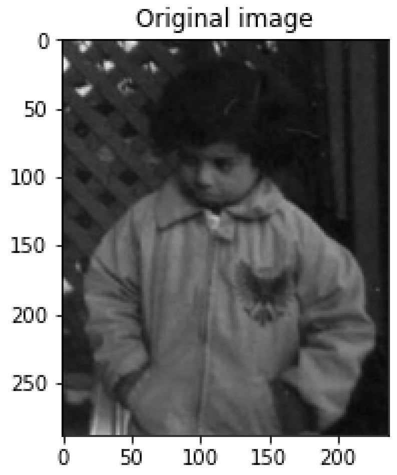
We simplify by using only gray level images. Therefore, the colour images are first read to gray level; a grey level image can be imported using the flag (`cv2.IMREAD_GRAYSCALE`). The colour images are `./images/waterfall2.jpg` and `./images/restaurantSpain.jpg` , available from CANVAS.

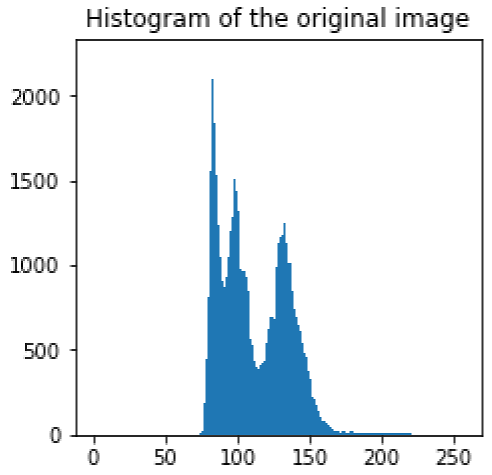
▶ **Click here for optional hints**

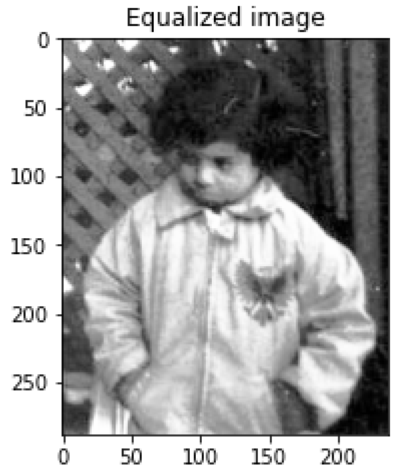
a) Use Python and find the histograms for the images.

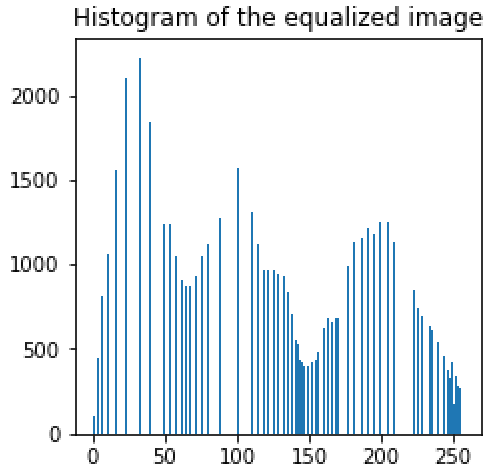
b) Perform histogram equalization of these images and find the new histograms.

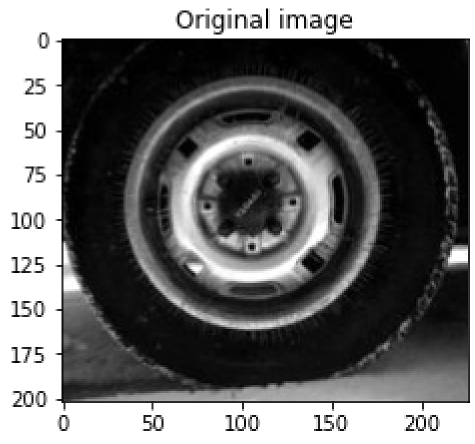
In [101…
```python
img1 = cv2.imread('pout.jpg', 0)
img2 = cv2.imread('tire.jpg', 0)
img3 = cv2.imread('waterfall2.jpg', 0)
img4 = cv2.imread('restaurantSpain.jpg', 0)

def histogram_equalization(img):
    # computing the histogram of the input image and output image and display
    hist = cv2.calcHist([img], [0], None, [256], [0, 256])
    # Equalizing the histogram
    img_equalized = cv2.equalizeHist(img)
    # Computing the histogram of the equalized image
    hist_equalized = cv2.calcHist([img_equalized], [0], None, [256], [0, 256]
    # Displaying the images and histograms
    plt.figure(figsize=(8, 8))
    plt.subplot(2, 2, 1)
    plt.title('Original image')
    plt.imshow(img, cmap='gray')
    plt.subplot(2, 2, 2)
    plt.title('Histogram of the original image')
    plt.hist(img.ravel(), 256, [0, 256])
    plt.subplot(2, 2, 3)
    plt.title('Equalized image')
    plt.imshow(img_equalized, cmap='gray')
    plt.subplot(2, 2, 4)
    plt.title('Histogram of the equalized image')
    plt.hist(img_equalized.ravel(), 256, [0, 256])
    plt.show()

# Applying the function to the 4 images
histogram_equalization(img1)
histogram_equalization(img2)
histogram_equalization(img3)
histogram_equalization(img4)
```

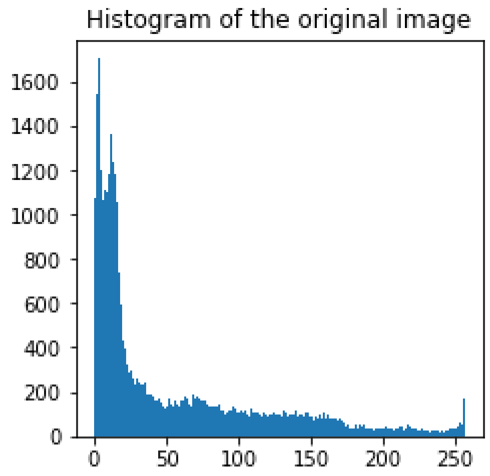

Original image

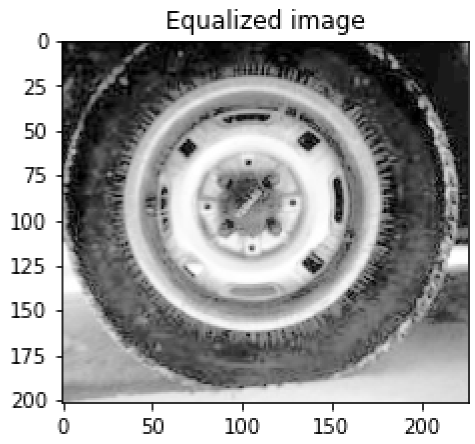

Histogram of the original image

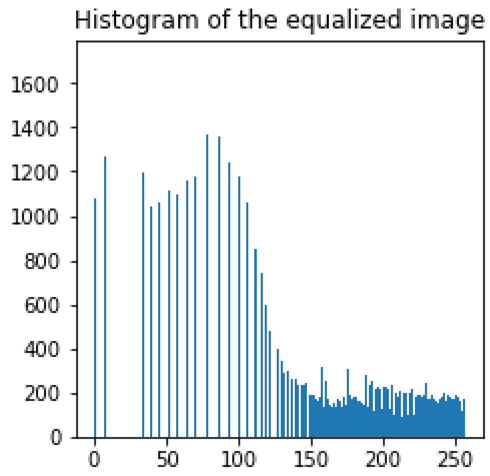

Equalized image

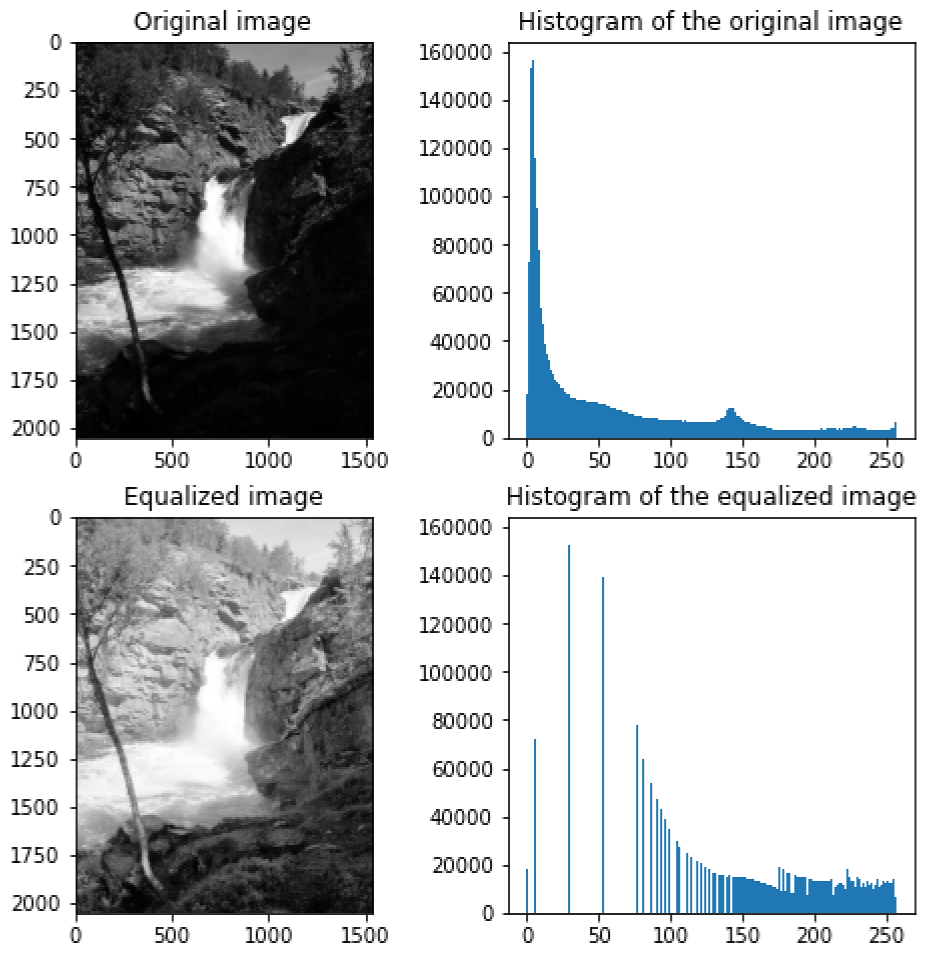

Histogram of the equalized image

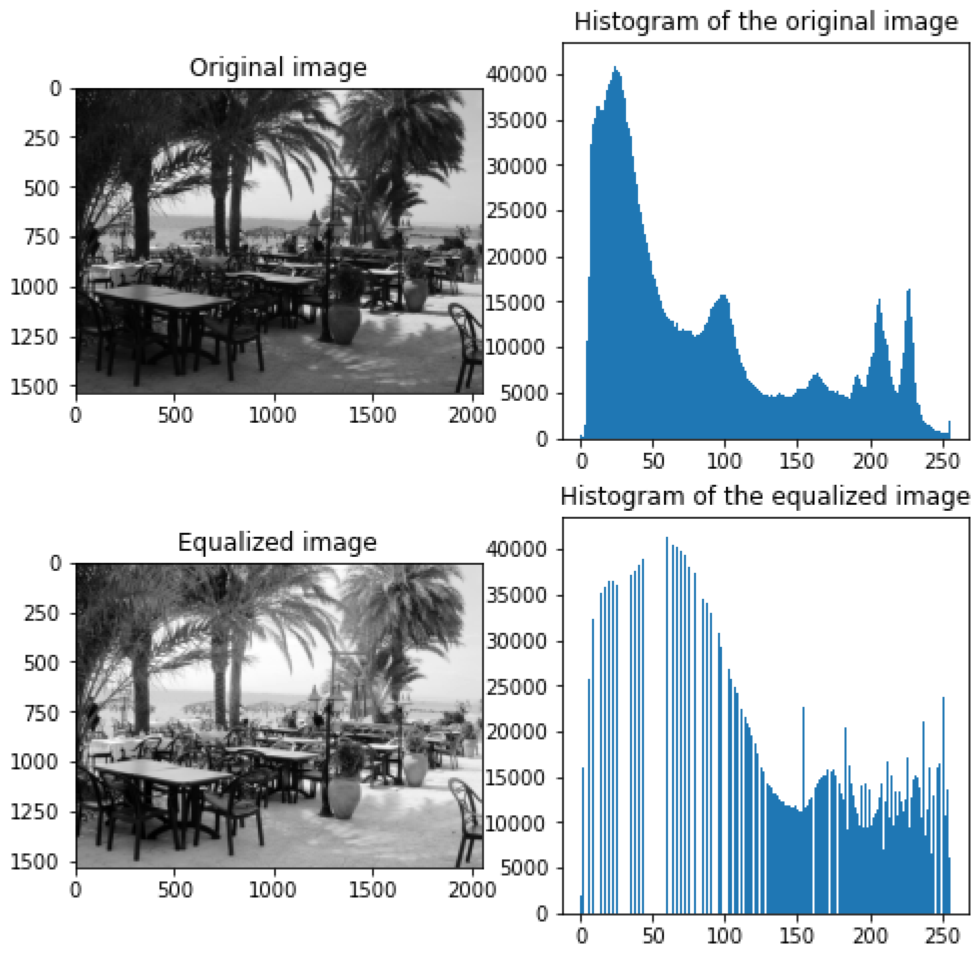

Original image


Histogram of the original image


Equalized image


Histogram of the equalized image





c) Discuss the results. Are the results as expected?

By comparing the original images and their histograms, with their equalized image and its equalized histogram, we see that the histograms behave diferently. This is because with histogram equalization we increase the contrast by spreading the pixel values of the original images more evenly accross the range of allowable values. The first orignal image of the girl has a pretty low contrast. We can see this in the original histogram because the histogram values are quite concentrated in a small space - whilst for the equalized image, we see that the histogram values are distributed more evenly and contrast is increased. In general, we can also note that more of the equalized histogram values are on the right side since the images become brighter. Histogram equalization first converts the PDF (from the normalized histogram) to a CDF. The CDF ensures that equal number of pixels contribute to an equally spaced interval in the output (this, when gray levels are continous).

d) Explain why the discrete histogram equalization usually do not give a completely flat histogram.

Discrete histogram equalization only produces an approximately flat output because of discretization effects. In other words it is an approximation of continous PDF, and when equalizing it doesn't produce new intesity levels.

Problem 4

Noise is a common problem in digital images. In this problem we want to study estimation of camera noise. We have a set of K images. The only difference between the images is the noise value at each pixel.

Assume that the noise is additive and uncorrelated with the gray values of the image such that for each image point we have $g_k = f + \eta_k,\ k = 1, 2,\ \cdots\ , K$, where g_k is image number k with noise η_k.

The image without noise, f, is unchanged (here we have not shown the indexes (x, y)). The mean image is given by the average value:

$$\overline{g(x,y)} = \frac{1}{K}\sum_{k=1}^{K} g_k(x,y). \qquad (1)$$

Then it can be shown that

$$E\{\overline{g(x,y)}\} = f(x,y) \qquad (2)$$

and

$$\sigma^2_{\overline{g(x,y)}} = \frac{1}{K}\,\sigma^2_{\eta(x,y)}. \qquad (3)$$

a) Show how to derive these two results using the first equation and the information given in the text.

$g_k = f + \eta_k,\ k = 1, 2,\ \cdots\ , K$

$\overline{g(x,y)} = E(f + \eta_k) = E(f) + E(\eta_k) - > 0 = E(f) = f(x,y)$

$VAR\overline{g(x,y)} = VAR(\frac{1}{K} \cdot \sum_{k=1}^{K}(f + \eta_k))$

$\frac{1}{K^2} \cdot VAR(\sum_{k=1}^{K} \eta_k)$

$\frac{1}{K} \cdot \sigma^2 \eta_{(x,y)}$

Delivery (dead line) on CANVAS: 23.09.2022 at 23:59

Contact

Course teacher

Professor Kjersti Engan, room E-431, E-mail: kjersti.engan@uis.no

Tomasetti Luca, room E-401 E-mail: luca.tomasetti@uis.no

Teaching assistant

Tomasetti Luca, room E-401 E-mail: luca.tomasetti@uis.no

Saul Fuster Navarro, room E-401 E-mail: saul.fusternavarro@uis.no

Jorge Garcia Torres Fernandez, room E-401 E-mail: jorge.garcia-torres@uis.no

References

[1] S. Birchfeld, Image Processing and Analysis. Cengage Learning, 2016.

[2] I. Austvoll, "Machine/robot vision part I," University of Stavanger, 2018. Compendium, CANVAS.