SIT789 – Robotics, Computer Vision and Speech Processing

Credit Task 1.2: Image histograms

Objectives

The objectives of this lab include:

- Calculating and plotting image histograms
- Applying histogram equalisation technique to improve image contrast

---

Tasks

1. Calculating and plotting histograms of an image

In this task, we will calculate and plot histograms for the image in Lenna.png (supplied in Task 1.1P). We will use cv.calHist() to calculate the histograms. Details of cv.calHist() can be found at Histograms - 1: Find, Plot, Analyze

We first load the image Lenna.png into computer memory by doing:

```
import numpy as np
import cv2 as cv
img = cv.imread('Lenna.png')
```

To calculate the histogram for the Blue/Green/Red channel of img, we perform:

```
hist_blue = cv.calcHist([img],[0],None,[256],[0,256]) #[0] for the blue channel
```

Note that the second parameter in cv.calHist is set to [0] for the blue channel. The green and red channels can be set to [1] and [2] respectively. For gray-scale images, only [0] is used. To plot hist_blue, we can use the following commands:

```
from matplotlib import pyplot as plt
plt.plot(hist_blue, color = 'b')
plt.xlim([0,256])
plt.show()
```

You can also calculate and plot the histogram for the red and green channels similarly.

2. Histogram equalisation

Histogram equalisation is an image processing technique that aims to improve the contrast of an image by avoiding inequality in the intensity distribution of that image, i.e., preventing large portions of the same intensity. In this task, we will apply the histogram equalisation technique to the images in img2.jpg and img3.jpg provided in OnTrack. The image img3.jpg is from the Chest X-ray Screening System: Segmentation Module – v3 dataset.

To better observe the effect of histogram equalisation, we will work on the grayscale versions of these images. For example, suppose that img is a colour image, we first convert img to its grayscale version by doing:

```
img_gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
```

We then calculate and plot the histogram of img_gray as follows:

```
hist_gray = cv.calcHist([img_gray],[0],None,[256],[0,256])
plt.plot(hist_gray)
plt.xlim([0,256])
plt.show()
```

We can also calculate the cumulative distribution of intensity of img_gray by defining the function getCumulativeDis as follows,

```
def getCumulativeDis(hist):
    c = [] #cummulative distribution
    s = 0
    for i in range(0, len(hist)):
        s = s + hist[i]
        c.append(s)
    return c
```

Next, we call this function and plot the returned cumulative distribution,

```
c = getCumulativeDis(hist_gray)
plt.plot(c, label = 'cummulative distribution', color = 'r')
plt.legend(loc="upper left")
plt.xlim([0,256])
plt.show()
```

We then apply histogram equalisation on img_gray by calling:

```
img_equ = cv.equalizeHist(img_gray)
```

We now show the histogram of img_equ,

```
hist_equ = cv.calcHist([img_equ],[0],None,[256],[0,256])
plt.plot(hist_equ)
plt.xlim([0,256])
plt.show()
```

and calculate and plot the cumulative distribution of intensity of img_equ,

```
c_equ = getCumulativeDis(hist_equ)
plt.plot(c_equ,label='cummulative distribution after histogram equalisation',color='r')
plt.legend(loc="upper left")
plt.xlim([0,256])
plt.show()
```

The cumulative distribution should have a line-like shape. You can also visually compare img_gray and its result after applying the histogram equalisation technique for both img2.jpg and img3.jpg.

We can verify the effect of histogram equalisation via an OCR (Optical Character Recognition) system. Specifically, you can pass img2.jpg and its histogram equalised version to the free online OCR engine at https://www.onlineocr.net/ and visually check their corresponding OCR results.

Note: cv.equalizedHist applies histogram equalisation on the whole image and thus may lose the contrast of several local image regions. To overcome this situation, adaptive histogram equalisation can be used. You are referred to Histograms - 2: Histogram Equalization for more details.

Finally, to measure how much change the histogram equalisation technique can make on an image, we can measure the difference between the intensity histograms of an input image and its histogram equalised version. There exist several metrics to measure such difference. In this task, we experiment with two common distance metrics: $\chi 2$ distance and Kullback–Leibler (KL) divergence. In particular, let h_1 and h_2 be two histograms. The $\chi 2$ distance between h_1 and h_2 is denoted as $\chi 2(h_1,h_2)$ and defined as,

$$\chi^2(h_1, h_2) = \sum_{i=0}^{n-1} \frac{(h_1[i] - h_2[i])^2}{h_1[i] + h_2[i]}$$

where n is the number of bins in the histogram h_1 (e.g., n=256) and $h_1[i]$ denotes the i-th bin of h_1.
The KL divergence $KL(h_1,h_2)$ is defined as,

$$KL(h_1, h_2) = \sum_{i=0}^{n-1} h_1[i] \log \frac{h_1[i]}{h_2[i]}$$

Since KL divergence is defined on probability distributions, the histograms h_1 and h_2 need to be normalised and the KL divergence is then applied on the normalised histograms. To normalise h_1 (h_2 can be normalised in the same way), we first calculate the sum of all elements in h_1, e.g., $s = \sum_{i=0}^{n-1} h_1[i]$, and then perform $h_1[i] \leftarrow \frac{h_1[i]}{s}$.

Your task here is to calculate the difference between the original images (in img2.jpg and img3.jpg) and their histogram equalised versions using both $\chi 2$ distance and KL divergence.

Note

- To calculate log, you need to "import math" and use "math.log".
- KL divergence is not a symmetric metric, i.e., $KL(h_1,h_2) \neq KL(h_2,h_1)$. Therefore, to overcome this issue, one may consider $KL(h_1,h_2)+KL(h_2,h_1)$ as a measure of the similarity between h_1 and h_2.
- In the implementation of $\chi 2$ distance and KL divergence, to avoid "divide by zero" error, one may add a very small value (e.g., 1e-10) to each $h_1[i]$ and $h_2[i]$.

Submission instructions

1. Calculate
   a. The histograms for the green and red channels of the image Lenna.png.
   b. The intensity histogram and cumulative intensity distribution of **img_gray** for img2.jpg and img3.jpg.
   c. The result (i.e., **img_equ**) of the histogram equalisation method when applied to img2.jpg and img3.jpg.
   d. The intensity histogram and cumulative intensity distribution of **img_equ** for img2.jpg and img3.jpg.
2. Calculate the $\chi 2$ distance and KL-divergence between the original image (i.e., **img_gray**) and its histogram equalised version (i.e., **img_equ**) for img2.jpg and img3.jpg.
3. Complete the supplied answer sheet with the results achieved in Instruction 1 and 2.
4. Convert the answer sheet to pdf and submit the pdf to OnTrack.
5. Save your notebook into a python (.py) file and submit the .py file to OnTrack.