EECS 332 Introduction to Computer Vision

Machine Problem 3: Histogram Equalization

Name: Runlong Lin   ID:3075255

1. Introduction

The image's histogram is the graphic representation of the gray level (or other attributes) distribution of the digital image. It plots the number of pixels for each gray level value. Usually, we should obtain a gray-level image with high contrast before we implement further image segmentation. So we use histogram equalization techniques to re-distribute the gray level value to each pixel. This is accomplished by spreading out the high frequent gray level value. In this way, we can obtain a new histogram in which the gray level value would be better distributed. Thus we can obtain a image with high contrast. At the same time, we also use linear and quadratic methods to implement light correction in the image.

2. Algorithm Description

In Machine Problem 3, we use MATLAB scripts to implement three funcitons:

```
function[output_img]=HistoEqualization(input_img)
function[output_img]=linear_correction(input_img)
function[output_img]=quadratic_correction(input_img)
```

where input_img is the input image and output_img is the output image.

- ### HistoEqualization

Here are the main steps of HistoEqualization function:

1) Convert the test image into gray level image.
2) Sum up the numbers of pixels for each gray level, using for-loop
3) Calculate the probability density of the numbers of the pixels. Then we use cumulative density function to implement a transform to the new distribution of the numbers of pixels for each gray level. The code below is the implementation of the transfer function.

```
for i=1:256
        p_h(1,i)=h(1,i)/(256^2);
    end
    Tr=zeros(1,256);
for i=1:256
    Tr(1,i)=round(sum(p_h(1:i))*256);
end
```

4) Since we have the transformation map, we can update the gray level for each pixel in the input image using for-loop.
5) Therefore we obtain the output image and we can plot its histogram as well.

- ### linear_correction

After the histogram equalization processing, we can obtain a image with high contrast. But it's imperfect as if the light was uneven. So we should correct the "light". The idea is that we fit a 3D plane using the gray level and pixel data to compensate our original image. The first method is the linear correction. In MP 3, we use generalized fitting method.

Here are the main steps of linear correction:

1) We use the linear model $a_1 + a_2 u + a_3 v = I(u, v)$ to fit our dataset $\{(\mathrm{u}, \mathrm{v}, \mathrm{I}(\mathrm{u}, \mathrm{v}))\}_{p=1}^{p}$,

where u and v define the pixel's position and I(u,v) is the gray level of the pixel.

2) We use the matrix form to express our model

$$\begin{bmatrix} u_1 & v_1 & 1 \\ u_2 & v_2 & 1 \\ \vdots & \vdots & \vdots \\ u_N & v_N & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} I(u_1, v_1) \\ I(u_2, v_2) \\ \vdots \\ I(u_N, v_N) \end{bmatrix}$$

3) Then we use can solve the equation using the pseudo-inverse method. It's easily implemented by Matlab calculation.

$$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} u_1 & v_1 & 1 \\ u_2 & v_2 & 1 \\ \vdots & \vdots & \vdots \\ u_N & v_N & 1 \end{bmatrix}^{\dagger} \begin{bmatrix} I(u_1, v_1) \\ I(u_2, v_2) \\ \vdots \\ I(u_N, v_N) \end{bmatrix}$$

4) Therefore we obtain the fitted plane. We can use the plane to compensate our original image. Our purpose is to increase the gray level of image part with low gray level, and decrease the gray level of the image part with high gray level. So we should apply some operations in the fitted plane and our input image. In this example, we can simply calculate the average of the transpose of the fitted plane and the input image. In this way, the output image would be the corrected image.

- **quadratic_correction**

The idea of quadratic correction is similar to linear correction. But we use quadratic model to fit the dataset instead of the linear model. Most of the steps are the same as linear correction so we skip the steps here.

3. Result Analysis

Here are the result figures.

1) Histogram Equalization

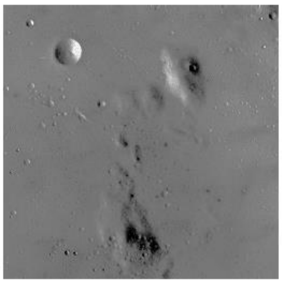

Fig.1. original image

Fig.2. image after equalization

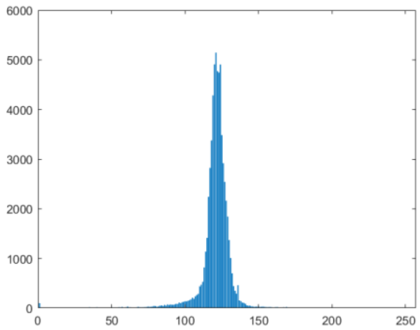

Fig.3. histogram of Fig.1.

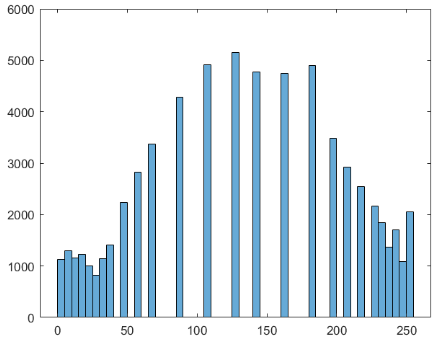

Fig.4. histogram of Fig.2.

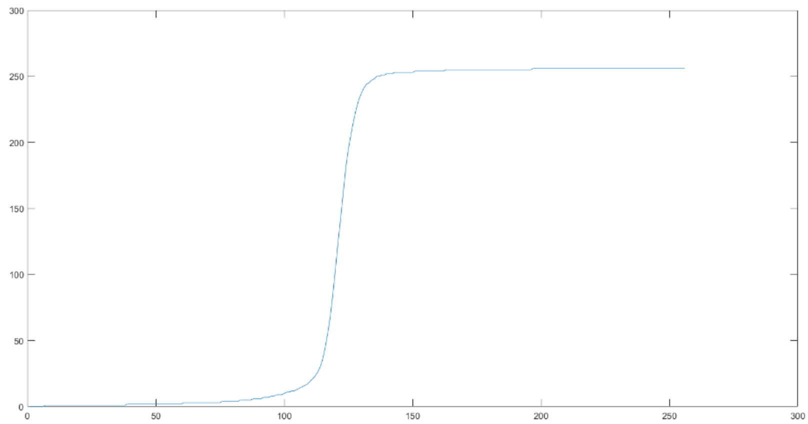

Fig.5. curve of transform function

Analysis: After histogram equalization, we can see more details in the output image because the contrast of the image is higher than before. We can also see in Fig.4 that the gray level is redistributed to each pixels. This is the reason why we can obtain a high contrast image. However, we can also see that the light in the output image is unevenly distributed.

2) linear correction

Fig.6. linear fitted plane

Fig.7. transpose of fitted plane

Fig.8. image after linear correction

Analysis: We use Fig.2 as the input image for linear correction. Fig.6 is the fitted plane. We can see that the upper right part is bright while the lower left part is dark. In order to compensate our input image, we need to adjust the light intensity by adjusting the gray level, which means that we should increase the value of low-gray-level region and decrease the value of high-gray-level region. In this example, we use the transpose of fitted plane to implement compensate. We calculate the average value of the transpose of fitted plane and input image. In this way, we can keep the total gray level of the image unchanged after the correction operation. Then we obtain Fig.8. Compared to Fig.2, we can see that light is evenly distributed in Fig.8.

3) quadratic correction

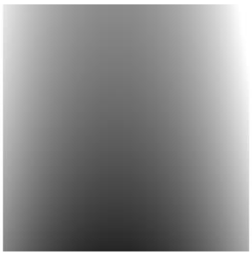

Fig.9.quadratic fitted plane

Fig.10. adjusted fitted plane

Fig.11. image after quadratic correction

Analysis: Similarly, we can obtain a fitted plane in quadratic correction. We also need to use this tilted plane to compensate our input image. Here we let 256 subtract each gray level value in fitted plane and obtain Fig.10. Then we can obtain our quadratic corrected image.