HW2: Histogram and Spatial Filtering

13331314 叶嘉祺

1.1 Histogram Equalization (15 Points)
Suppose that you have performed histogram equalization on a digital image. Will a second pass of histogram equalization (on the histogram-equalized image) produce exactly the same result as the first pass?Prove your answer.
It is equivalent to all of the same number of pixels on the gray level, no matter how many times you have a number of histogram equalization, he has not changed.
Prove:
Assume that the first pass of histogram equalization result in gray level s.

$$s = T_1(r) = \int_0^r p_r(\omega)d\omega\ , 0 \le r \le 1$$

$$P_s(s) = \frac{dT(r)}{ds} = \frac{dT(r)}{dr} \cdot \frac{dr}{ds}$$

$$= \frac{dT_1(r)}{dr} \cdot \frac{dr}{ds} = p_r(r) \cdot \frac{1}{p_r(r)} = 1$$

assume that the first pass of histogram equalization result in gray level t.

$$t = T_2(s) = \int_0^s p_s(\omega)d\omega\ , 0 \le s \le 1$$

In the same way,

$$p_t(t) = P_s(s) \cdot \frac{ds}{dt} = 1, 0 \le t \le 1$$

To sum up ，a second pass of histogram equalization produce exactly the same result as the first pass.

1.2 Spatial Filtering (20 Points)
Consider a 4 × 4 gray image and a 3 × 3 filter:
Image :

$$\begin{matrix} 85 & 13 & 29 & 83 \\ 169 & 8 & 243 & 28 \\ 17 & 155 & 27 & 33 \\ 91 & 25 & 173 & 79 \end{matrix}$$

Filter :

$$\begin{matrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{matrix}$$

1. Convolve the gray image with the given filter with zero-padding, and show your result (whose size should be 4 × 4). (7 Points)

$$\begin{matrix} -177 & -420 & -279 & -271 \\ -74 & -72 & -90 & 52 \\ 61 & 131 & 2 & 19 \\ 172 & 199 & 1215 & 60 \end{matrix}$$

2. Discuss the meanings of positive values and negative values in your convolution result respectively. (8 Points)
Negative values indicates that, the new gray value is very small and it may not be the border image.
Positive values indicates that, the new gray value is very big and may be bigger than a average, and it could be the border image.

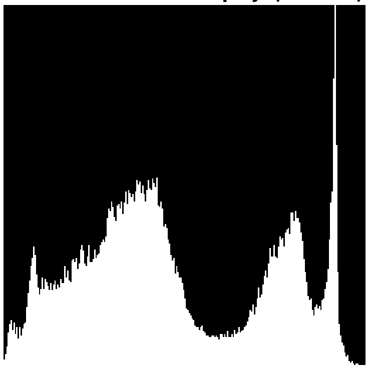
3. Describe some applications of the given filter based on your own knowledge. (5 Points)
This spatial filter can be used for edge detection, remove the part of the pseudo edge, make a smooth effect for the noise.

Programming

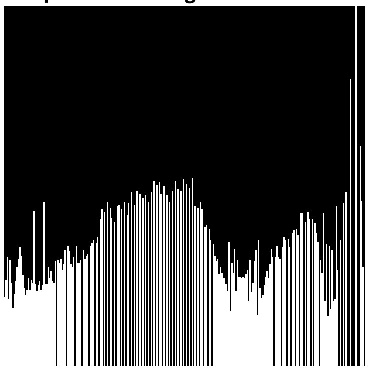
2.2 Histogram Equalization (35 Points)

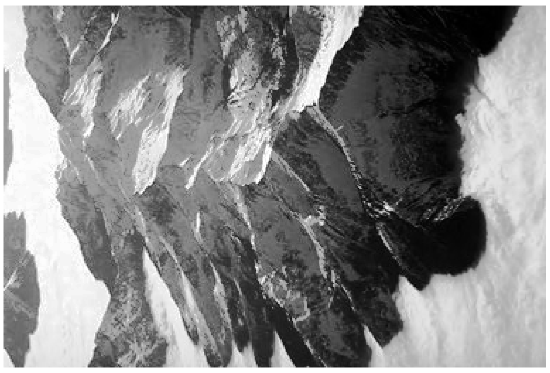
Write a function that applies histogram equalization on a gray scale image. The function prototype is "equalize hist(input img) → output img", returning a gray scale image whose histogram is approximately flat. You can modify the prototype if necessary. For the report, please load your input image and use your program to:

1. Compute and display its histogram. Manually paste the histogram on your report. Note:You must compute the histogram by yourself, but existing APIs can be used for display. (5 Points)



2. Equalize its histogram. Paste the histogram-equalized result and the corresponding histogram on your report. (10 Points)





3. Analyze your histogram-equalized result in less than 1 page. (8 Points)

The result in fact looks better than the origin one.

By this method, the brightness can be better distributed on the histogram. This can be used to enhance the contrast of the contrast without affecting the overall contrast, histogram equalization by effectively expanding the common brightness to achieve this function.

Let's put the origin image and the output image together.

It strongly shows that the first image is much better than the second one since after the histogram equalization processing, the original relatively small pixel gray level will be allocated to other gray, the pixel is relatively concentrated, after processing gray range, contrast, large, clarity, so can effectively enhance the image.
Histogram equalization is a method to adjust the contrast in the field of image processing. This method is usually used to increase the local contrast of many images, especially when the contrast of the useful data of the image is very close. By this method, the brightness can be better distributed on the histogram. This can be used to enhance the contrast of the contrast without affecting the overall contrast, histogram equalization by effectively expanding the common brightness to achieve this function.

4. Detailedly discuss how you implement the histogram equalization operation, i.e., the "equal-ize hist" function, in less than 2 pages. Please focus on the algorithm part. Don't widely copy/paste your codes in the report, since your codes are also submitted. (12 Points)
The target to get a histogram equalization is the same to get a corresponding relation between `gray_value_old` and `gray_value_new` .

The core algorithm of histogram equalization is to calculate such a table(array) for the image.

$$s = T(r) = \sum_{0}^{r} p_r(i), 0 \leq r \leq 255$$

```
fx[i] = 255 * sum([hist[j] for j in range(i+1)]) / (width * height)
```

Note that `fx` indicates the corresponding relation table(array) for the origin image. And sum is function to calculate the formula above.

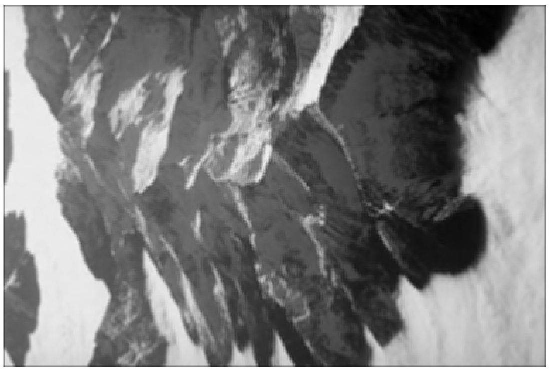
And also note that I implicitly do a scaling in the calculating.

Next it's very easy, just traverse the array and then assign the table value for the new image.

```
new_img[x, y] = fx[old_img[x, y]]
```

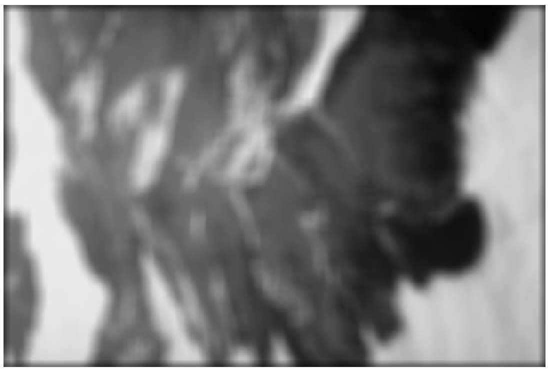
Happily , we got the histogram equalization result and then we can draw the histogram again and the result.

2.3 Spatial Filtering (30 Points)
Write a function that performs spatial filtering on a gray scale image. The function prototype is "filter2d(input img, filter) → output img", where "filter" is the given filter. Modify the prototype if necessary.For the report, please load your input image and use your "filter2d" function to:
1. Smooth your input image with 3 × 3, 7 × 7 and 11 × 11 averaging filters respectively. Paste your three results on the report. (9 Points)
3*3



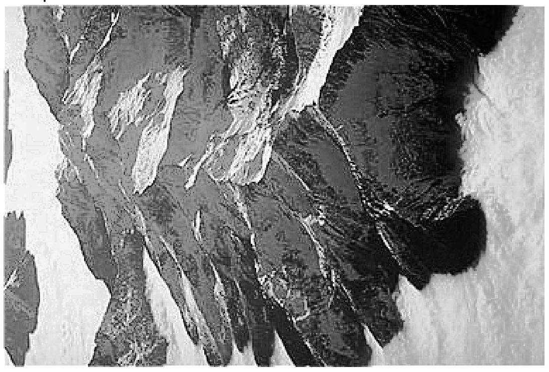
7*7

11*11



22. Sharpen your input image with a 3 × 3 Laplacian filter (There are four variants of Laplacian in Fig. 3.37 of the textbook. Pick any one you like.) and then paste the result. In addition, briefly discuss why Laplacian filter can be used for sharpening. (6 Points)

sharpen result.



The function of image sharpening is to make the gray contrast enhanced, so that the fuzzy image becomes more clear. The essence of the image is that the image is subjected to an average operation or an integral operation, so it can be carried out on the inverse operation, such as differential operation can highlight the image details, so that the image becomes more clear.

$$\begin{matrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{matrix}$$

Since Laplasse is a differential operator, it is applied to enhance the gray level of the image in the region, and weaken the slow change of gray area. Therefore, the sharpening processing can choose the original image of the Laplasse operator to deal with the original image, and then the Laplasse image and the original image superimposed to produce a sharpening image.

$$\nabla f^2 = [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) + f(x+1,y+1) + f(x+1,y-1) + f(x-1,y+1) + f(x-1,y-1)] - 8f(x)$$

3. Perform high-boost filtering (i.e., g(x, y) = f (x, y) + k ∗ g mask (x, y), see Eq. (3.6-9) of the textbook for other details) on your input image. The averaging part of the process should be done using the filter in Fig. 3.32(a) of the textbook. Choose a k (the weight in Eq. (3.6-9)) as you see fit. Write down the selected k and paste your result on the report. (5 Points)

k = 0.8

4. Detailedly discuss how you implement the spatial filtering operation, i.e., the “filter2d” function, in less than 2 pages. (10 Points)

For a selected filter and a selected pixel, we need to calculate the as we want the convolution result of the image. We defined a function to calculate the gray value for the exactly one pixel.

```
Convolution(x, y, filter) → new_gray_value
```

```
def convolution(arguments):
    for i in range(filter_size):
        for j in range(filter_size):
            if out_of_range:
             center_value += 0
            else:
             center_value += filter[i][j]*image_p
    center_value /= filter_size*filter_size
```

Then we can calculate the center_value for one pixel in the origin image. Then we can have a nested for loop to calculate the whole image.

```
def filter2d(arguments):
    for i in range(height):
        for j in range(width):
          new_image[i,j] =
            convolution(i,j, filter)
```

Notice that, when I am doing laplacian_filter, the result is very wired. As a result, we need to compelete scaling after operation. What’s more, I did a gama correction as wil to have a better view.
scaling method:

$$s = |S(x,y)| = |\frac{f(x,y) - min(f(x,y))}{max(f(x,y)) - min(f(x,y))}| \times 255$$

I use the same method in histogram equalization that to set up a table for gama function.

```
    for i in range(256):
        val = pow(float(i)/255.0 ,scale) * 255.0
        if val>255:
                val = 255
        elif val<0:
                val = 0
        table[i]= val
```

I meet a problem when using the laplacian method. The noise of the image is also enhanced.
So I adjust the method with:

$$g = G(x,y) = k * f(x,y) + L(x,y)$$

For high_boost_filtering:
First we should calculate the Fuzzy image:

```
    for i in range(height):
        for j in range(width):
            sub_img[i ,j] = img[i, j][0] - fuzzy_img[i][j]
```

And we get the mask `sub_img` , and then we calculate the output image:

```
for i in range(height):
    for j in range(width):
        enhanced_img[i, j] = img[i, j][0] + k * sub_img[i][j]
```

And then we get the enhanced_img by doing the formula.

$$g(x,y) = f(x,y) + k * g_m(x,y)$$