CLab-1 Report

ENGN6528

Zhiyuan Huang

u6656110

21/8/2020

1 Task 1 Python Warm-up

(1) $a = np.array([[2, 4, 5], [5, 2, 200]])$

This function creates a 2×3 matrix (2D array).

(2) $b = a[0, ;]$

Extracts the first row from a.

(3) $f = np.random.randn(500, 1)$

Creates a 500×1 matrix from the standard normal distribution.

(4) $g = f[f < 0]$

Finds all the values in the matrix f that is less than 0.

(5) $x = np.zero((1, 100)) + 0.55$

Creates a 1×100 matrix with all the values are 0.55.

(6) $y = 0.4 * np.ones([1, len(x.transpose())])$

This creates a 1×100 matrix with all the values are 0.4.

(7) $z = x - y$

This computes the result mtrix of x - y.

(8) $a = np.linspace(1, 200, 200)$

This function creates an array with 200 elements from 1 to 200 with same intervals. That is {1, 2, 3, ... 200}.

(9) $b = a[:: -1]$

b is the reversed array of a.

(10) $b[b < 50] = 0$

Assign 0 to all elements in b that is less than 50.

2 Task 2: Basic coding practice

1.

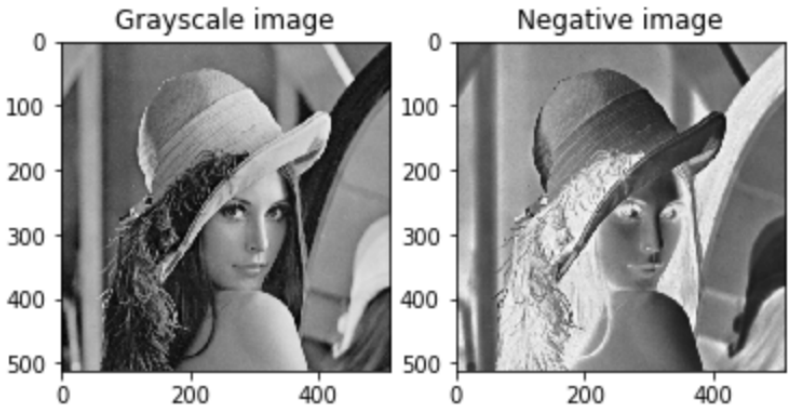

Figure 1: A grayscale image with its negative image

2.

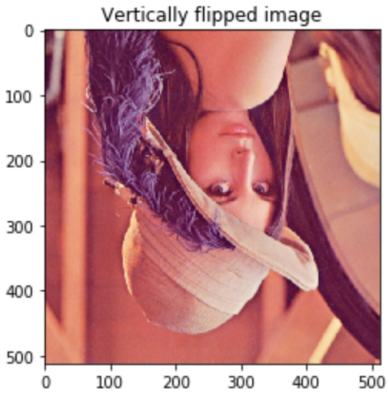

Figure 2: Flip the image vertically

3.

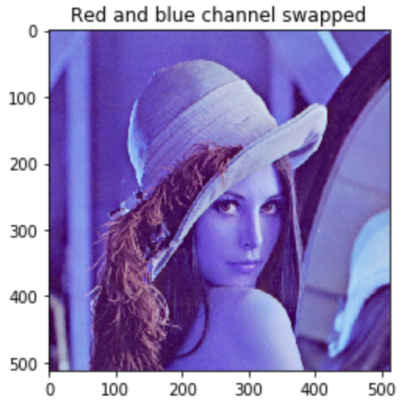

Figure 3: Swap red and blue value

4.

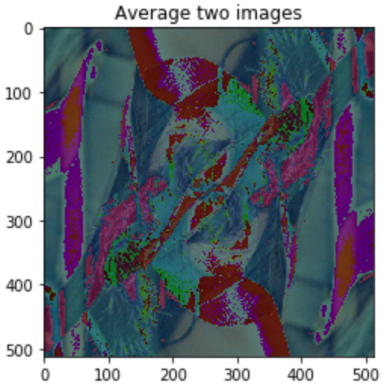

Figure 4: Average two images

5.

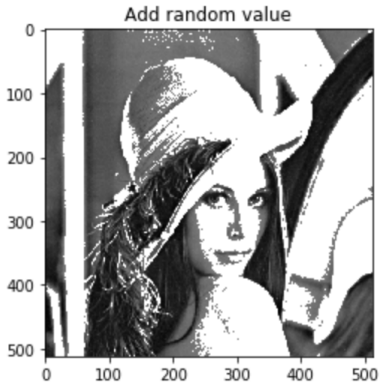

Figure 5: Add a random value to the image

3 Task 3 Basic image i/o

1. Take one of my face image and resize the image to 720 rows * 480 columns.

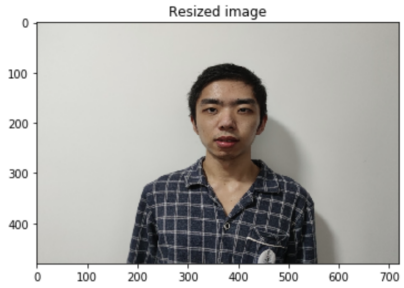

Figure 6: Resize the image to 720 * 480

2. Extract red, green, blue grayscale values and plot them into images.

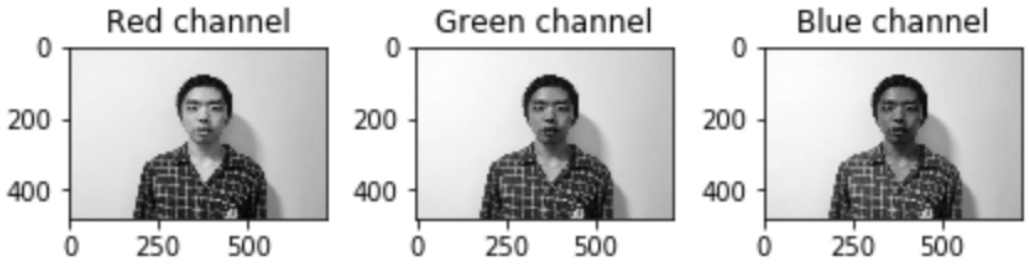

Figure 7: Three channel grayscale image

3.

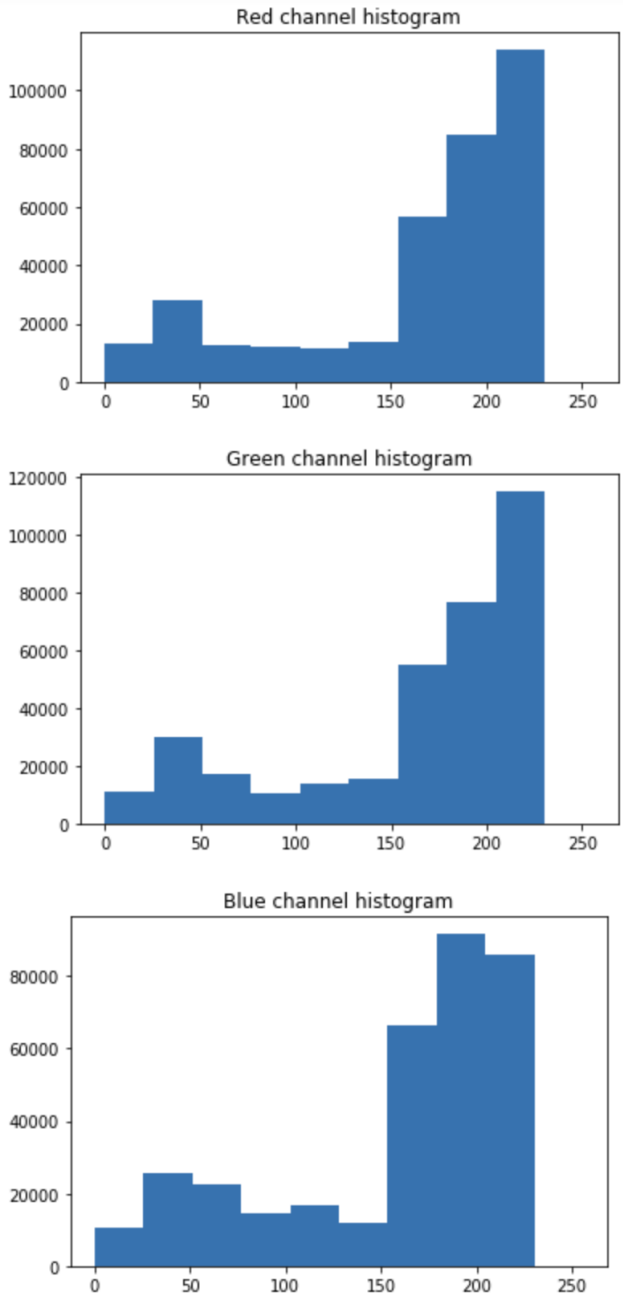

Figure 8: Three histogram of each channel

4. Plot the equalized single channel image and then combine them together to form the final histogram equalization image.

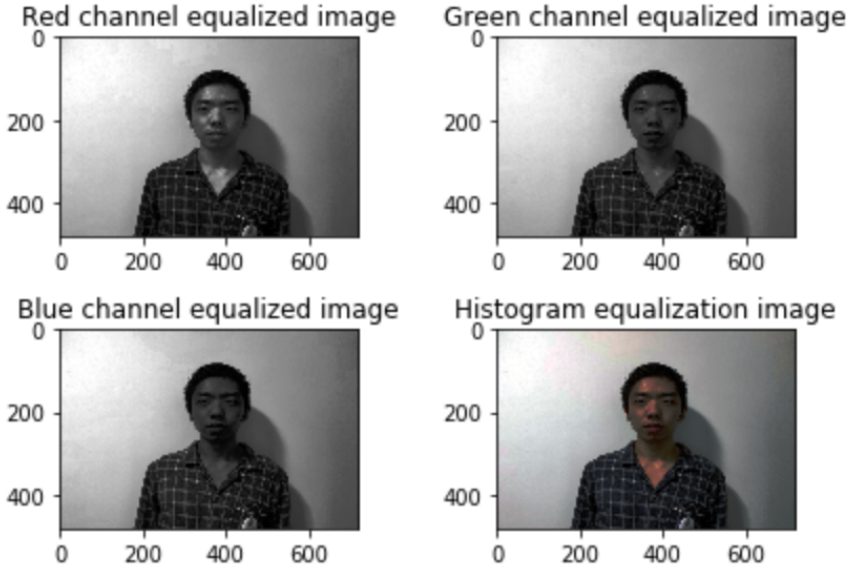

Figure 9: Histogram equalization images

4 Task 4 Image denoising

1.

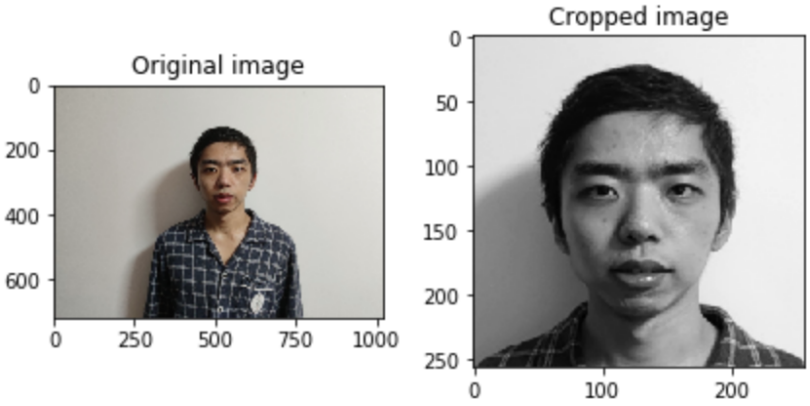

Figure 10: The original image and the cropped image with size of 256 * 256

2.

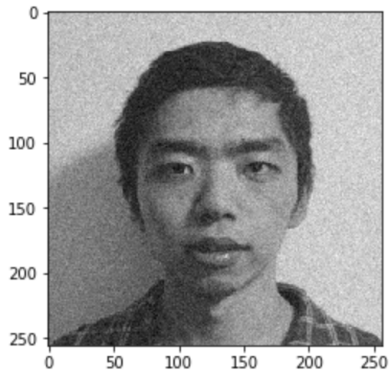

Figure 11: Image after adding gaussian noise

3.

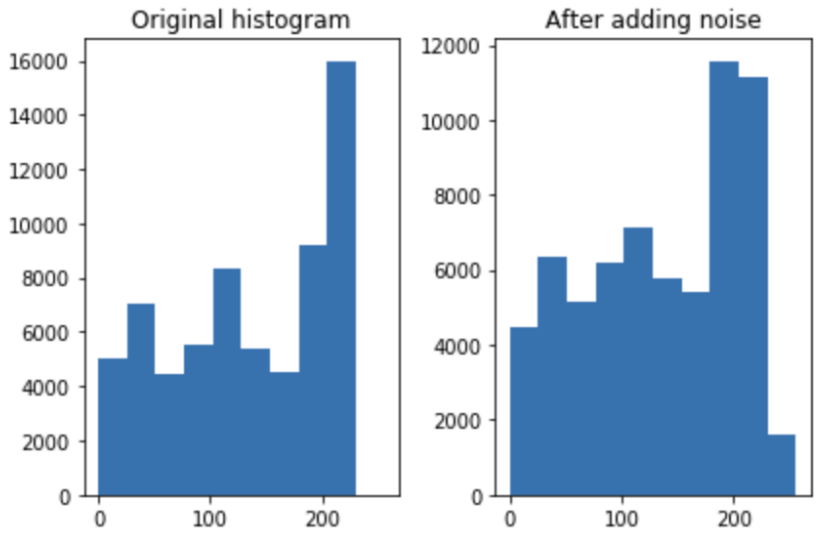

Figure 12: Two histograms

4. Use my own Gaussian filter to process the noisy image

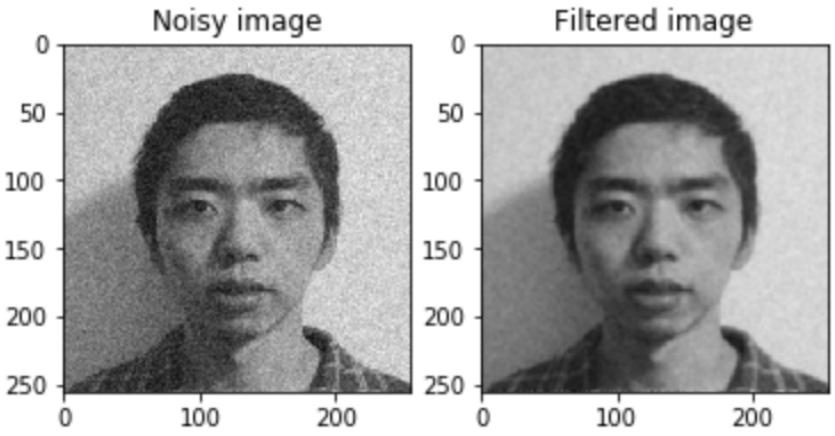

Figure 13: Filter the noisy image with my own gaussian filter

5. The image in the left is the image processed with the inbuilt filter. The

image on the right is filtered by my own function. From the picture we can see that both images have less noise than before, demonstrating that my Gaussian filter could reduce noise. These two images are nearly identical in terms of the smooth and noise-removal effects, which is within our expectations. However, the image uses my filter function is has low contrast than the image using inbuilt filter.

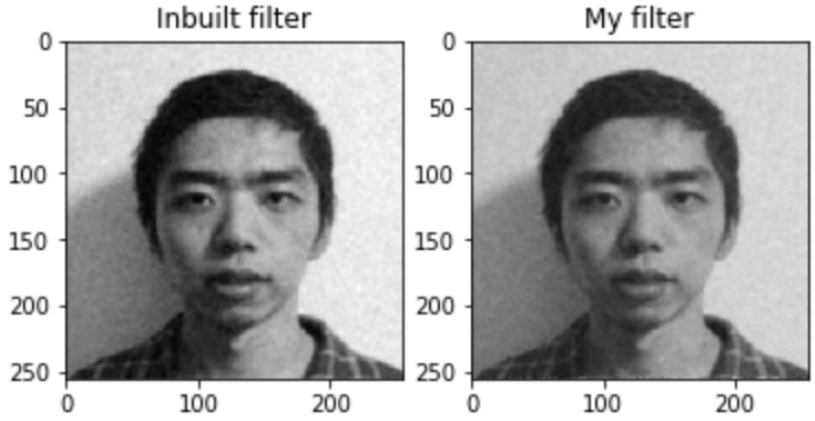

Figure 14: Compare the inbuilt gaussian filter with my own implementation

5 Task 5 Implement Sobel filter

The implementation of Sobel filter is pretty straightforward. There are two kernels, which corresponds to x-axis and y-axis, respectively. I just iterate through the image and multiply the two filters in different direction and then calculate the summation of the square root of the two values, and then assign the new value to the image.

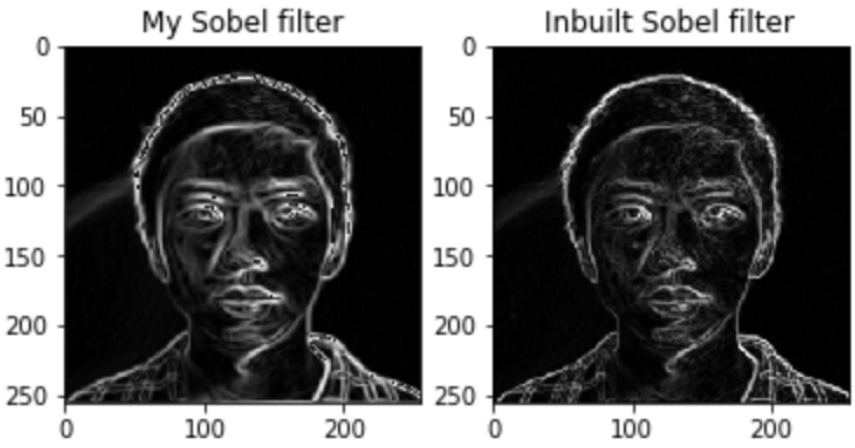

Figure 15: Compare the inbuilt Sobel filter with my own implementation

6 Task 6 Image rotation

1.

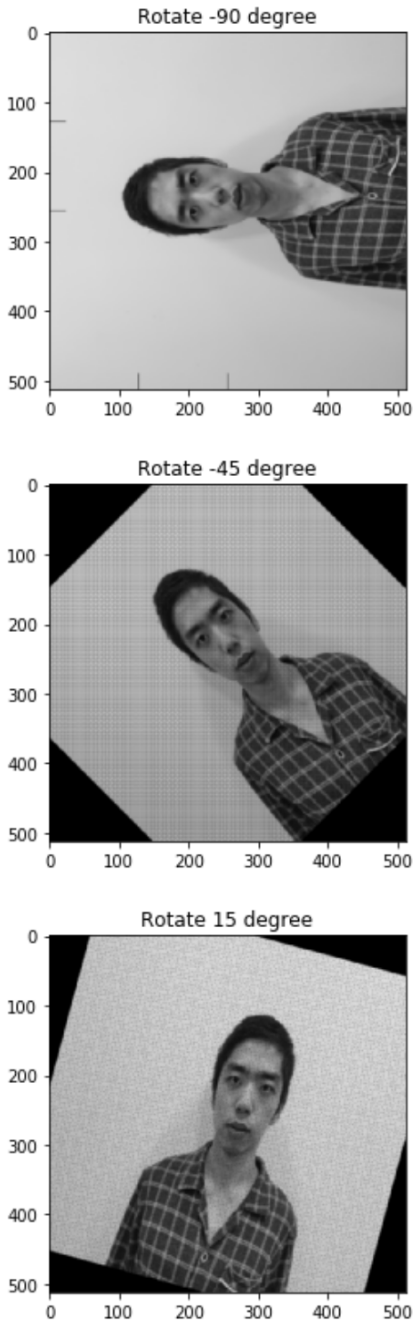

Figure 16: Rotate the image with different angles

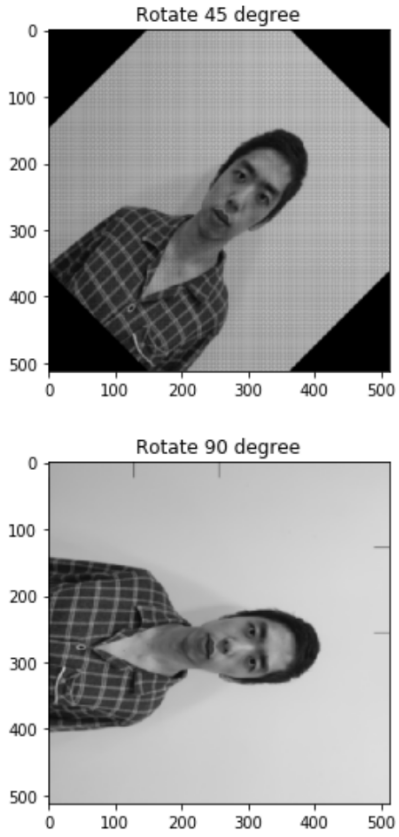

Figure 17: Rotate the image with different angles

2. Forward mapping just map the original pixel value to its new position. Since its new position may not be an integer, so there might be holes after mapping, so we have to use splat technic to reconstruct the holes, which is complicated and time consuming.

Backward mapping doesn't have this problem. However, it maybe difficult to calulate the invertible warp function.