HW 1 Report

Name: 陳翰雯

ID：B08902092

Problem 0: WARM UP

(a) Horizontal Flipping

This is done by flipping the location of the pixel located on the left side to the position of the pixel located on the mirrored right side.

The main code for this is: `final_img[y][w-x-1] = img[y][x]`

Fig 1. sample1.png

Fig 2. result1.png

(b) Grayscale Image Transformation

We read the colored image in 3 layers with the ordering of R,G,B

The image is transformed to grayscale using the formula: `0.2989 * r + 0.5870 * g + 0.1140 * b`

`r`, `g`, `b` here refers to the intensity at each respective layer of the RGB image. The formula is applied to every pixel through looping.

Fig 3. sample1.png

Fig 4. result2.png

Problem 1: Image Enhancement

(a) Decrease Brightness

Every intensity value is divided by 2

Fig 5. sample2.png

Fig 6. result3.png

(b) Increase Brightness

Every intensity value is multiplied by 3. However in cases where the multiplied value is bigger than 255, we will round it to 255.

Fig 9. result3.png

Fig 10. result4.png

(c) Histogram Observation

When the brightness of the image is decreased, we could see that the bars in the histogram only fills up to half the space. The histogram does have the same pattern or shape however, result3.png's is more like of a compressed version of the sample2.png's and does not go over the intensity value of approximately 125.

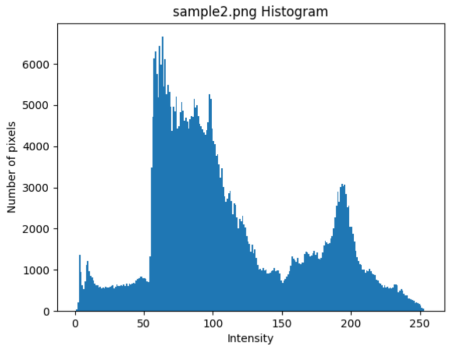

Fig 7. Histogram of sample2.png

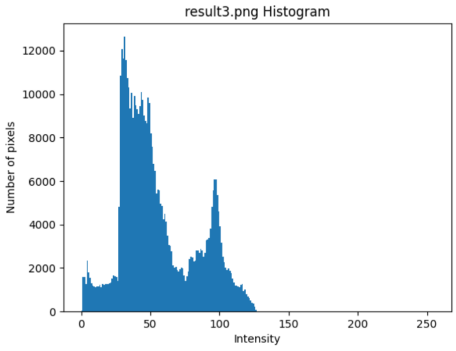

Fig 8. Histogram of result3.png

When the brightness is increased the we can see that the pixels are centered at value 255. This is because most of the intensity values, when multiplied by 3,

exceeds 255. We could also see that the histogram of result4.png has some gaps in between. The image (result4.png) itself of shows lots of white areas in comparison to the original image (sample2.png)

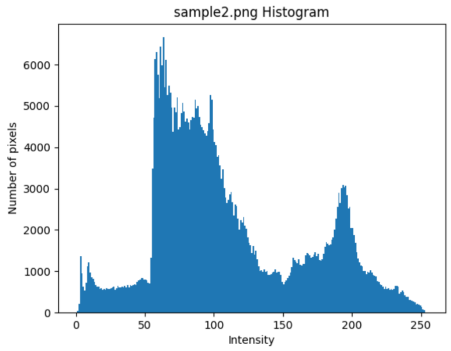

Fig 11. sample2.png

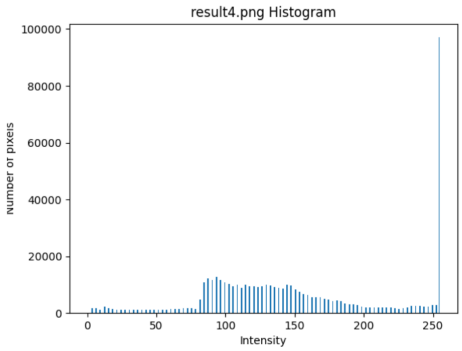

Fig 12. result4.png

(d) Global Histogram Equalization

After performing global histogram equalization on result3.png, we could see that the image got brighter and more details are shown clearly. On the sky part, the separation of colors become even more visible. It goes from darker to lighter shade. The histogram of result3.png shows that the bars were packed into the left side of the space, however, after performing global equalization we could now see that the bars on the histogram of result5.png is now evenly spread. Both histograms still have similar pattern or shape.

Fig 13. result3.png

Fig 14. result5.png

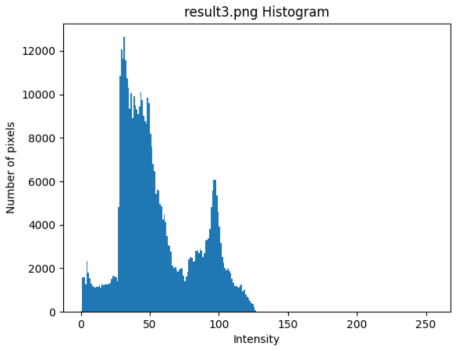

Fig 15. Histogram of result3.png

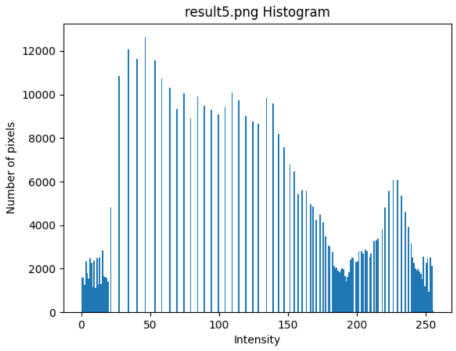

Fig 16. Histogram of result5.png

After performing global histogram equalization on result4.png, unlike result3.png, the image got overall darker. However similarly, the separation of the shades on the sky part has become even more visible too. Regarding the clouds, we could see that it is rougher and we could clearly see the outline of the clouds. The colors are more well blended in result3.png. The histogram of result6.png shows more clustered bars rather than unlike result4.png which are more evenly spread.

Fig 17. result4.png

Fig 18. result6.png

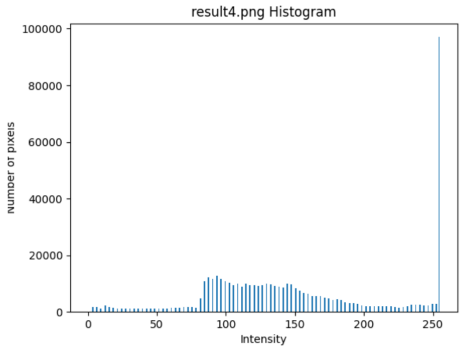

Fig 19. Histogram of result4.png

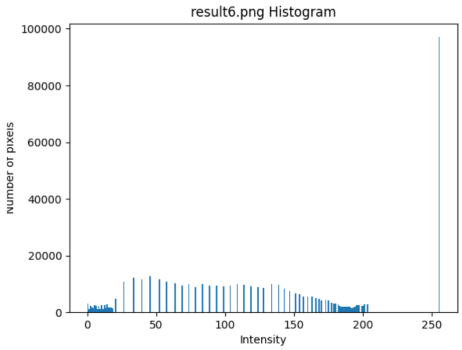

Fig 20. Histogram of result6.png

(e) Local Histogram Equalization

For both images result3.png and result4.png, we use window size of 200.

In case of result3.png, we could see that after performing the local histogram equalization, some parts of the images such as the mountain area got brighter, however the sky part of the image got darker. The details of the image still can be seen even after equalization. In the histogram we could see that instead of having high peaks like that of result3.png, we got a smoother bowl like pattern in the histogram of result7.png and instead of being compressed to the left side, the bars of

the local histogram equalization are more evenly spread through out the intensity values.

Fig 21. result3.png

Fig 22. result7.png

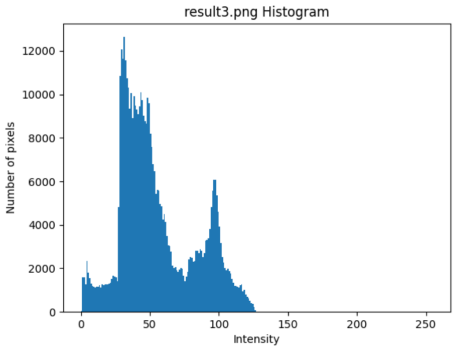

Fig 23. Histogram of result3.png

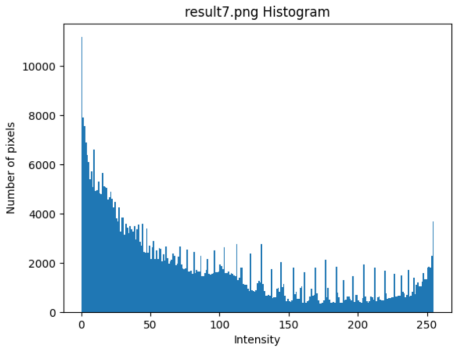

Fig 24. Histogram of result7.png

In case of result4.png, we could see that in the mountain part of the image, the brightness does not have significant change, however the sky got a lot darker. In the cloud part we could also see that the cloud and the sky does not blend well. We could clearly see the outlines of the clouds. In the histogram we could see that the bars are now connected instead of having gaps (result4.png) with a smooth decreasing pattern. However, The high peak of intensity level 255 still remains the same.

Fig 25. result4.png

Fig 26. result8.png

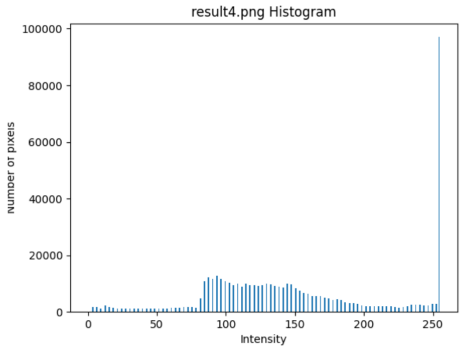

Fig 27. Histogram of result4.png

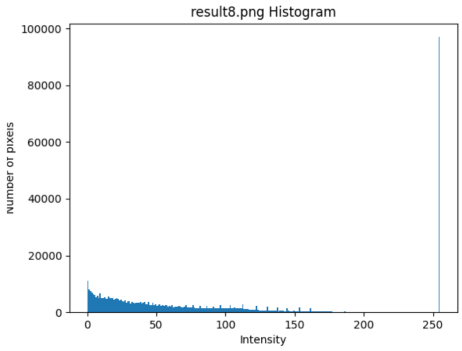

Fig 28. Histogram of result8.png

(f) Transfer Function

The transformation I used here is power-law transformation with $p = 1/2$. The resultant image got a lot more brighter without degrading the details of the image. In the histogram of sample2.png we could see that a great increase happens around the intensity of 60. However, in the histogram of result9.png we could see that it has somehow shifted to the right. The pixels number suddenly increase on the intensity of approximately 120.

Formula for power-law transformation: $G(j,k) = [F(j,k)]^p \quad 0 \leq (j,k)] \leq 1$

Formula implementation in code: `final_img = 255*(img/255)**(1/2)`

Fig 29. sample2.png

Fig 30. result9.png

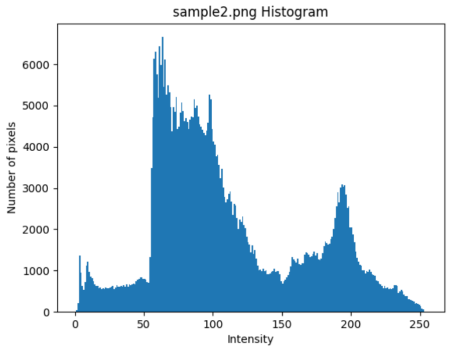

Fig 31. Histogram of sample2.png

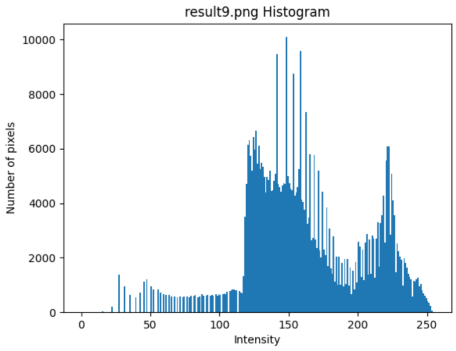

Fig 32. Histogram of result9.png

Problem 2: Noise Removal

(a) Remove Noise

The noise on the sample4.png seems similar to uniform noise. Hence, low-pass filtering is used. The general form matrix is used with b=2. Here, 5 rounds of low-pass filtering with b=2 is done. As the number of rounds increases, the noises in the image also decreases. However, the blurriness of the image also increases.

Formula used:

$$H = \frac{1}{(b+2)^2}\begin{bmatrix} 1 & b & 1 \\ b & b^2 & b \\ 1 & b & 1 \end{bmatrix} \text{ with } b = 2$$

Fig 33. sample4.png

Fig 34. result10.png

Fig 35. first round: result10.png

Fig 36. second round: result10.png

Fig 37. third round: result10.png

Fig 38. fourth round: result10.png

Fig 39. fifth round: result10.png

The noise on sample5.png seems more like impulse noise. Hence outlier detection of window=3 and ε=60 is used. Another 5 rounds of outlier detection is done. On every round the epsilon is decreased by 10 (epsilon-=10). In the histogram we could see that at first, the most left (intensity value = 0) and the most right (intensity value = 255) stand out the most. Which really shows that there are lots of slat and pepper. However, as the number of rounds increases, we could see that the number of pixels at intensity values 0 and 255 decreases over time.

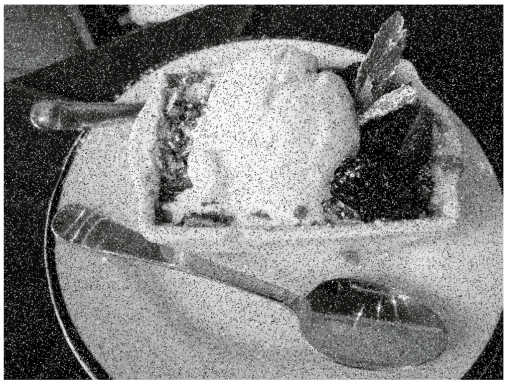

Fig 39. sample5.png

Fig 40. result11.png

Fig 42. first round: result11.png

Fig 43. second round: result11.png

Fig 44. third round: result11.png

Fig 45. fourth round: result11.png

Fig 46. fifth round: result11.png

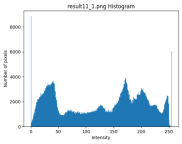

Fig 47. first round: Histogram of result11.png

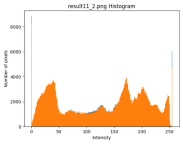

Fig 48. second round: Histogram of result11.png

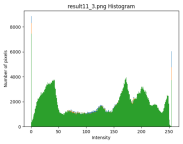

Fig 49. third round: Histogram of result11.png

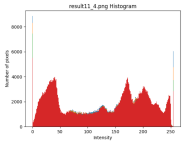

Fig 50. fourth round: Histogram of result11.png

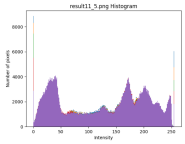

Fig 51. fifth round: Histogram of result11.png

(b) PSNR values

Here are the PSNR values of sample3.png and result10.png for 5 rounds:

1. $18.469622616299684db$
2. $18.483988826826085db$
3. $18.4209229178387 02db$
4. $18.350095045723 67db$
5. $18.283404865239 69db$

The value shows a decreasing trend overall, however the difference is very small. The reason PSNR decreases even if the image has less noise is maybe due to the increasing blur after each round. Even though

Here are the PSNR values of sample3.png and result11.png for 5 rounds:

1. $24.663174862660 37db$
2. $26.524057113450183db$
3. $27.76462542551307db$
4. $28.5438221469143 05db$
5. $29.084904889197208db$

The value shows an increasing trend overall, however the difference is not significant.