Intro. Image Processing
LAB 2

謝侑哲

112550069

1. Method

A. Histogram Equalization

(1) Preliminary

對人類來說，頻率 Uniform 的圖片，在對比度上比較好，也就是對人類比較好看的圖片。Histogram Equalization 就是要使得圖片的分布變成 Uniform 的。

(2) Method

Histogram Eqalization 基於 s = T(r) = (L-1)$\sum_{j=0}^{k} P_{r(r_j)}$ 的這個從原圖的頻率 r 轉換成頻率 s 會符合 unform distribution (在連續時)這個觀察。所以有了以下方法:

1. 把所有 pixel 歸類到 N 個區間中，並計算每個區間出現的機率
2. 把機率累加變成得到 Transform 後的 r
3. 因為映射到的值做四捨五入，確保映射到整數序的頻率區間
4. 把原本的 pixel 加上移動的頻序區間，就可以得到近似 Uniform 分布的圖

```
def generate_hist(img, n):
    interval = 256 / n
    levels = (img / interval).astype(np.uint8)
1.  hist = np.bincount(levels.flatten(), minlength=n).astype(np.float32)
    hist = hist / (img.shape[0] * img.shape[1])# hist.sum() # make hist probability
    return hist

def hist_equalization(img, hist, n):
2.  s = np.cumsum(hist)
    s = s * (n - 1)
3.  s = (s + 0.5).astype(np.uint8)
    interval = 256 / n
    levels = (img / interval).astype(np.uint8)
4.  img = img + (s[levels] - levels.astype(np.float32)) * interval
    return img.astype(np.uint8)
```

(code 部分，用紅色數字標記對應哪一的步驟)

B. Histogram Specification

(1) Method

在 Histogram Equalization 我們可以把一個 Distribution 轉成 Uniform，相對來說，就可以用他的反函式把 Uniform 轉成另一個 Distribution。所以當要把 Distribution 1 (D1)轉成 Distribution 2 (D2)，可以用$T_2^{-1}(T_{1(img)})$的方式，把 D1 轉成 Uniform 再轉成 D2。

在實作上，因為 Histogram Equalization 的函式是 mapping，所以只要把 mapping 的值當成參數映射到原輸入，就可以得到反函式，至於沒有一一對應的部分，則設定直接往頻率小的地方找。

步驟為:

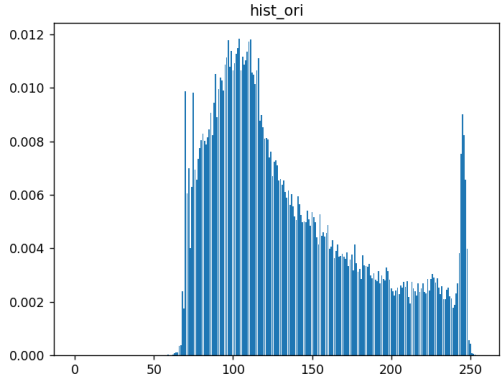
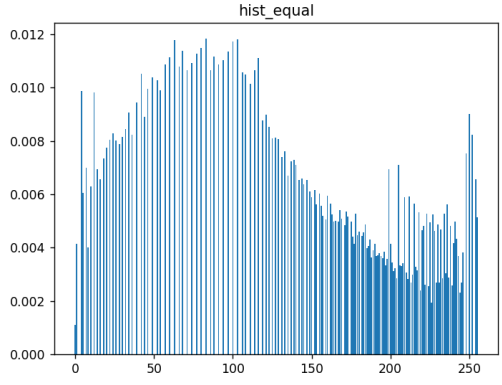
1. 計算兩張圖的 Equalization mapping
2. 計算 inverse mapping
3. 做 mapping，並把反函數中有空缺的部分往小頻率找填值

```python
def hist_specification(img1, img2, n): # img1 -> img2
1.  hist1 = generate_hist(img1, n)
    s = np.cumsum(hist1)
    s = s * (n - 1)
    s = (s + 0.5).astype(np.uint8)
    hist2 = generate_hist(img2, n)
    z = np.cumsum(hist2)
    z = z * (n - 1)
    z = (z + 0.5).astype(np.uint8)
2.  inv_z = np.zeros(n)
    for i in range(0, n):
        idx = np.where(z == i)[0] # fine first i in z
        idx = idx[0] if idx.size > 0 else -1
        inv_z[i] = idx
3.  canva = np.zeros(img1.shape)
    interval = 256 / n
    for i in range(img1.shape[0]):
        for j in range(img1.shape[1]):
            level = (img1[i][j] / interval).astype(np.uint8)
            to_level = s[level]
            while inv_z[to_level] == -1:
                to_level -= 1
            to_level = inv_z[to_level].astype(np.float32)
            canva[i][j] = img1[i][j] + (to_level - level.astype(np.float32)) * interval
    return canva.astype(np.uint8)
```

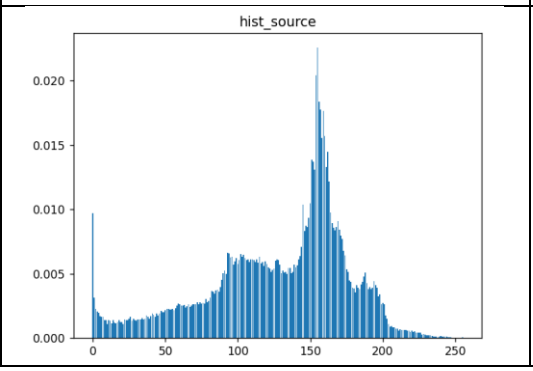
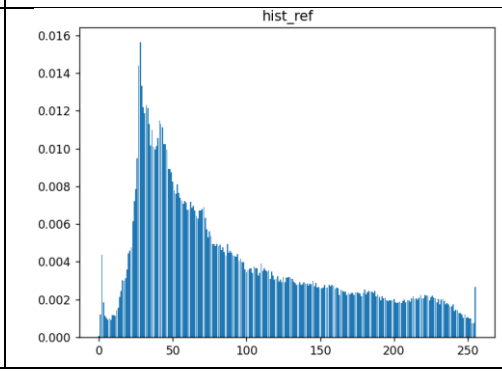
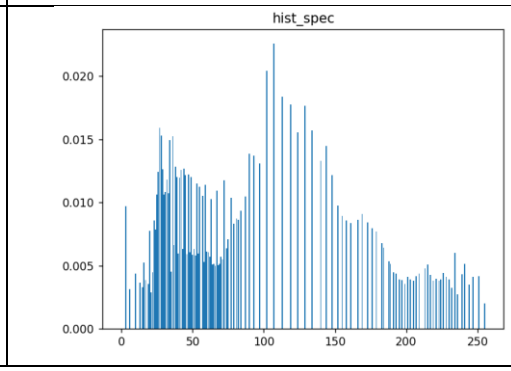
(code 部分，用紅色數字標記對應哪一的步驟)

2. Result

A. Histogram Equalization

Origin	Histogram Equalization
	
	

B. Histogram Specification

Origin	Reference	Histogram Specification
		
		

從 Equalization 的結果中，可以看出做過 Equalization 的圖片對比度更好，更能看出圖片中的細節。並且明暗的 Histogram 的分布也更平均而不這麼集中在偏高的地方

在 Specification 的結果中，可以看到圖片在對比度上，變得跟 Reference 更加接近，分布的變得更偏暗，但細節比較少的狀態。可以看到明暗的 Histogram 從原本的偏高變的邊低。

3. Feedback

本次課程中，讓我了解到怎麼透過調整照片的明暗度頻率，讓圖片的對比度變得更好看，或是變成想要的效果。感覺可以延伸做成 Style Transfer 之類的任務，可以有很多的 extend 應用。十分有趣！