

11/6/2026 黃廷睿

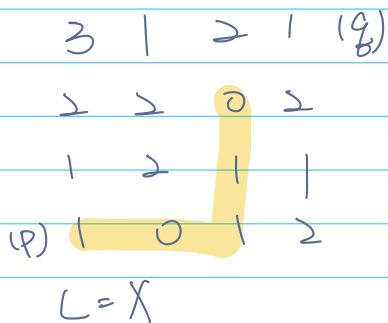
2.12

$$f(x,y) = i(x,y) \cdot r(x,y), \text{ where } r = | \\ = i(x,y) \\ = k e^{-[x-x_0]^2 + [y-y_0]^2}, \quad k = 255$$

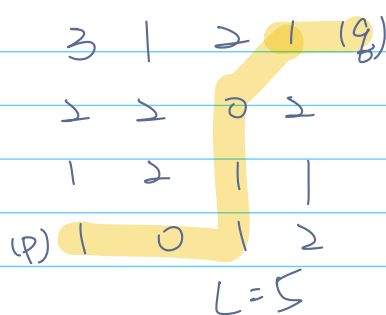
2.16

(4)

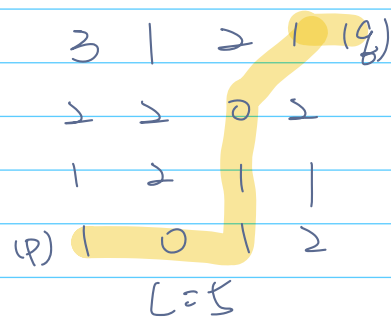
2.18



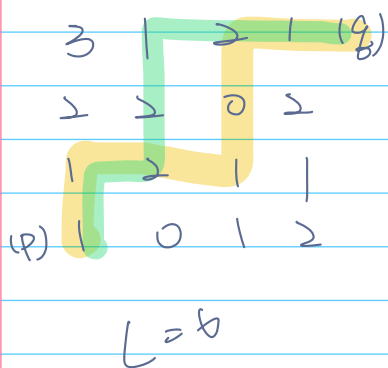
(8)



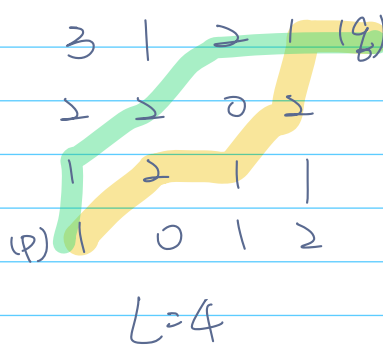
(m)



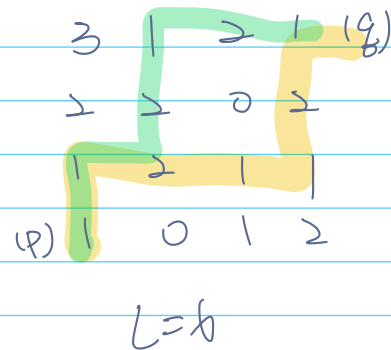
(4)



(8)



(m)



2.37

a.  $A = \begin{bmatrix} G_x & 0 & 0 \\ 0 & G_y & 0 \\ 0 & 0 & 1 \end{bmatrix}, A^{-1} = \begin{bmatrix} \frac{1}{G_x} & 0 & 0 \\ 0 & \frac{1}{G_y} & 0 \\ 0 & 0 & 1 \end{bmatrix}$

b.  $A = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}, A^{-1} = \begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix}$

c.  $A_v = \begin{bmatrix} 1 & s_v & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, A_v^{-1} = \begin{bmatrix} 1 & -s_v & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$A_h = \begin{bmatrix} 1 & 0 & 0 \\ s_h & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, A_h^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -s_h & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

d.  $A = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}, A^{-1} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$

e.  $A = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$

3.12.  $\int_0^r P(w) dw.$

$$P_r(r) = -r+2 = \int_0^r (-2w+2) dw$$

$$= -r^2 + 2r$$

$$P_z(r) = 2r = \int_0^r 2r dw = r^2$$

$$\Rightarrow z = G^{-1}(w) = \pm \sqrt{w} = \sqrt{r+2r}$$

3.18  $w = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}, f = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$

(a)  $g(x,y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s,t) f(x+s, y+t)$

$$w = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \Rightarrow 4 \times 1 + 2 \times 1 = 6$$

(b) convolution

padding  $\rightarrow$  repeat  $\rightarrow$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \xrightarrow{\text{padding}} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \xrightarrow{\text{repeat}} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 3 & 6 & 3 & 0 & 0 \\ 0 & 0 & 4 & 8 & 4 & 0 & 0 \\ 0 & 0 & 3 & 6 & 3 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

w 翻轉後仍為 w

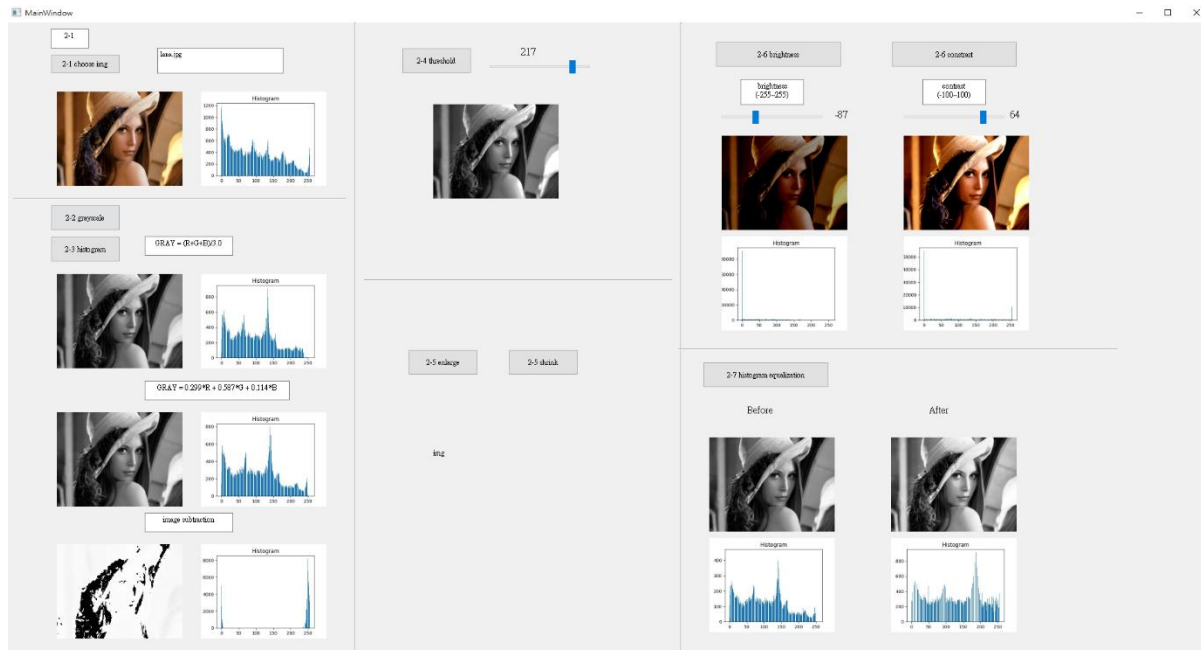


# Principles and Applications of Digital Image Processing

## HW2 Report

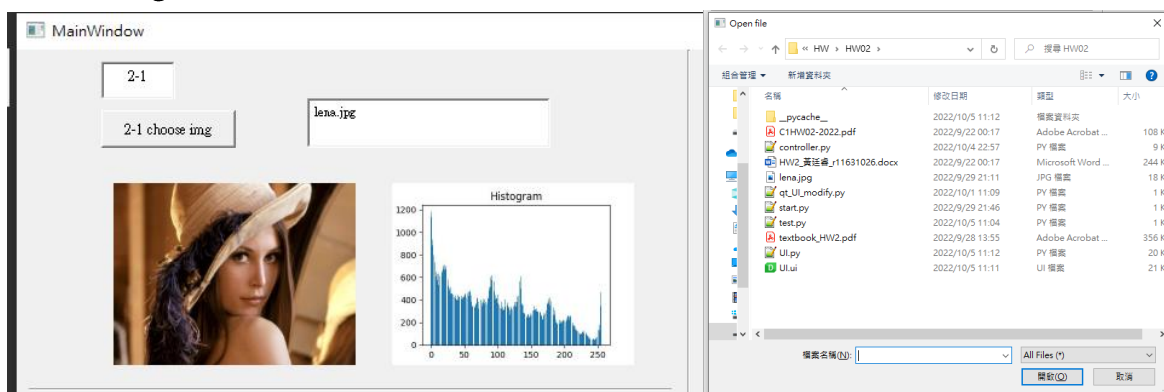
R11631026 黃廷睿

### Part 2: Image File Reading, Display and Basic Processing



整體介面如上圖，各區域分別對應不同小題的功能。

#### 1. Read image

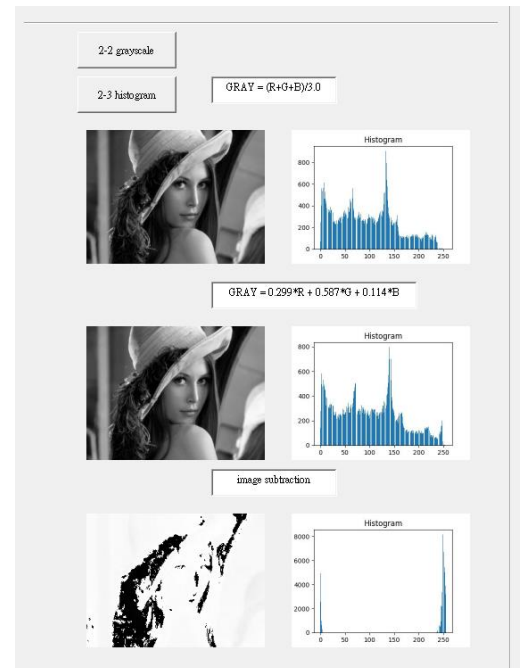


透過檔案選取功能，讓使用者可透過檔案總管選取所需要的圖片，並將其讀入 QT 中，利用 QLabel 做顯示，並同時顯示直方圖以便後續觀察變化。

2. Convert a color image into a grayscale image
3. Display the histogram of a grayscale image

此處兩題一起進行，透過將圖片各通道/3 或是乘上彼此不同的權重，雖然從肉眼判斷並無太大差距，但在直方圖表示即可發現不同權重圖的高光部分有個較尖的突起，這是/3 所沒有的。

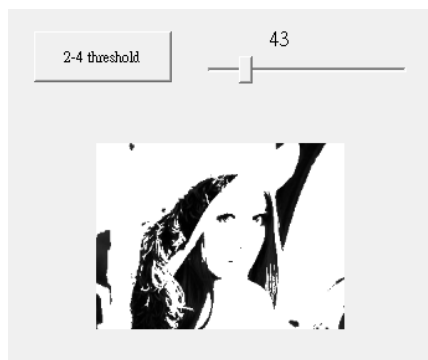
並且將兩張照片相減可發現，其實彼此不同的區域比想像的還要多，大致都分布在最亮或是最暗處。



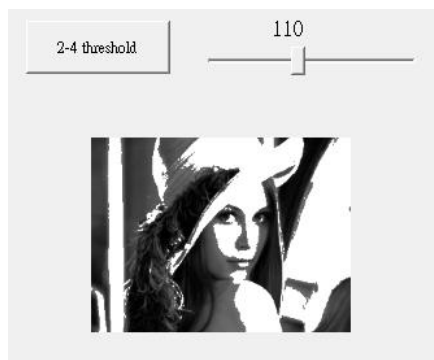
並且同時也與攝影界常用的修圖軟體 Lightroom(LR)做比較，可發現利用 LR 轉成的黑白照片較/3 的照片亮部過渡較為平滑，推測是軟體內還是有針對相片做特定的優化算法。

4. Manual threshold function

此處設計是利用 Slider 做滑桿，當調整到所要的數值時按下 2-4 button，照片便會做相對應的轉換，目前是設計當照片中的像素值大於滑桿設定的 threshold 便會將其轉為 255，亦即為白色，以下便是不同 threshold 對應的結果



Threshold = 43



Threshold = 110

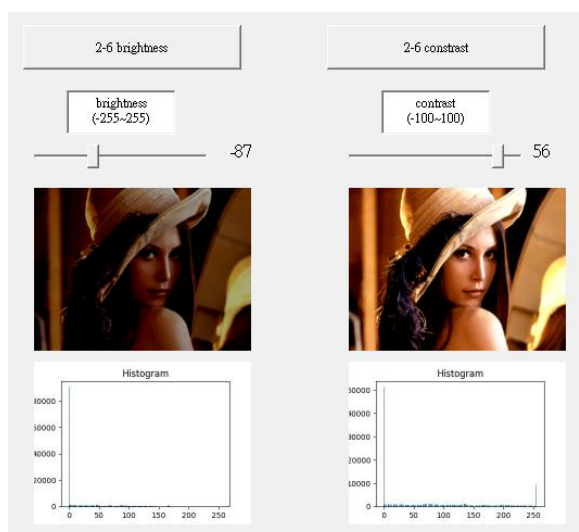


Threshold = 207

## 5. Adjust the spatial resolution

此題目目前尚未做完

## 6. Adjust the brightness and contrast



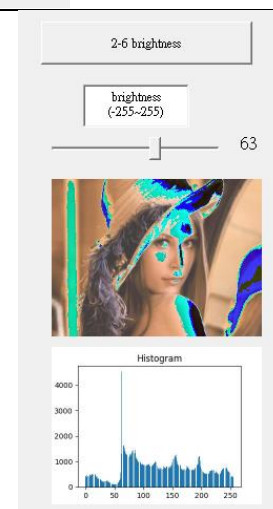
原先打算利用單純的矩陣加上常數作為調整亮度功能：

```
self.brightness_img = self.img + self.brightness
```

然而此方法會產生相當奇怪的照片，圖片顏色會錯亂，原因並不清楚。

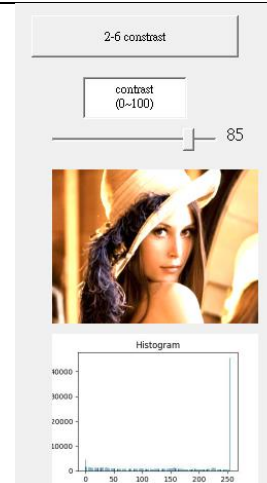
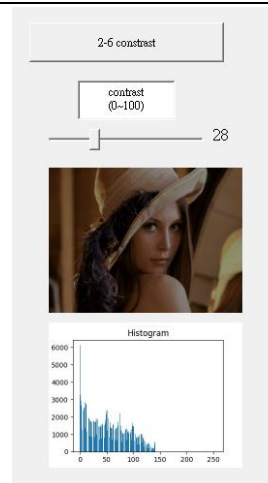
但後來發現若將原始圖片\*1.0 便可以正常顯示了  
(僅只有\*1 還不行)

```
self.brightness_img = self.img * 1.0 + self.brightness
```



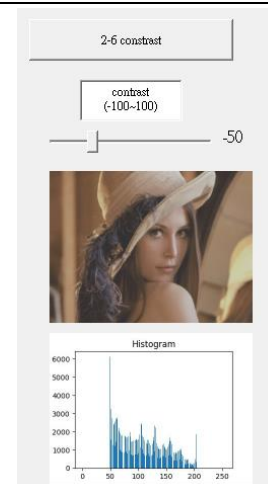
原先利用矩陣乘上特定比例做為調整對比度，然此方法效果不佳，僅像是將照片調亮調暗而已，並不像認為中的增加明暗對比。

```
self.contrast_img = self.img *  
(self.contrast/50)
```

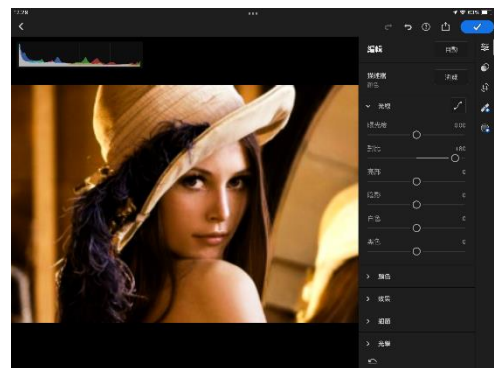
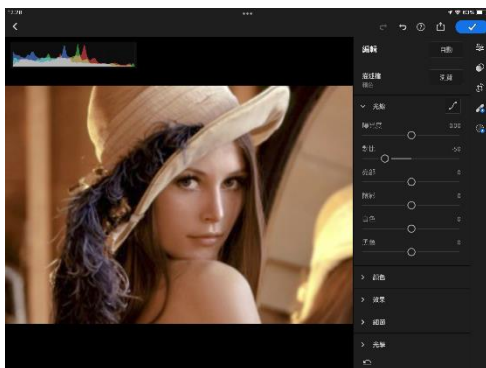


後來在下方文章中發現另一個計算方法：  
$$\text{new\_image} = (\text{old\_image}) \times (\text{contrast}/127 + 1) - \text{contrast} + \text{brightness}$$
，經實際測試，確實比原先的效果好多了，高對比也有較濃的顏色。

[How do I adjust brightness, contrast and vibrance with opencv python? - Stack Overflow](#)



並且這邊也與 LR 的對比調整功能做比較，可發現即使用了較合理的計算方法，但調整出來的顏色風格仍較 LR 更為濃烈，因此可判斷兩者內部的算法應不相同。





## 7. Histogram equalization function



可發現目前設計的算法確實可以將直方圖平均的分散到整個 X 軸上。