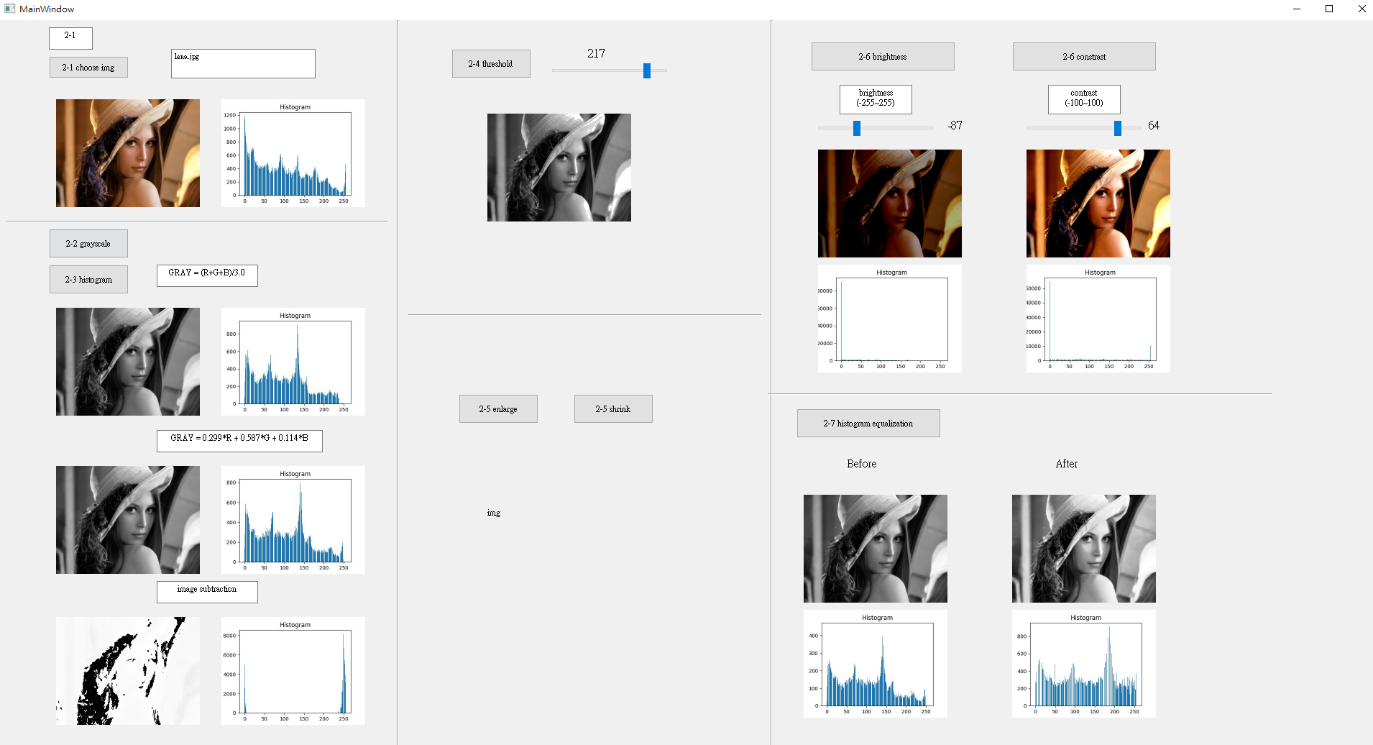
Principles and Applications of Digital Image Processing

HW2 Report

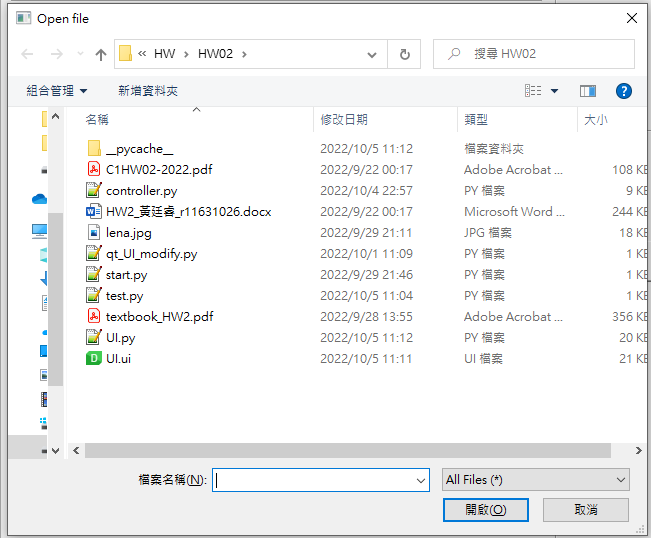
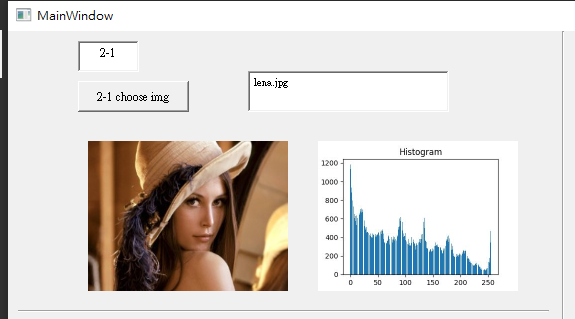
R11631026 黃廷睿

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。

1. Convert a color image into a grayscale image
2. 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。 | |

1. 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 |

1. Adjust the spatial resolution

此題目前尚未做完

1. 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個計算方法：new\_image = (old\_image) × (contrast/127 + 1) - contrast + 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](https://stackoverflow.com/questions/50474302/how-do-i-adjust-brightness-contrast-and-vibrance-with-opencv-python) |  |  |

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。

|  |  |
| --- | --- |
|  |  |

1. 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。