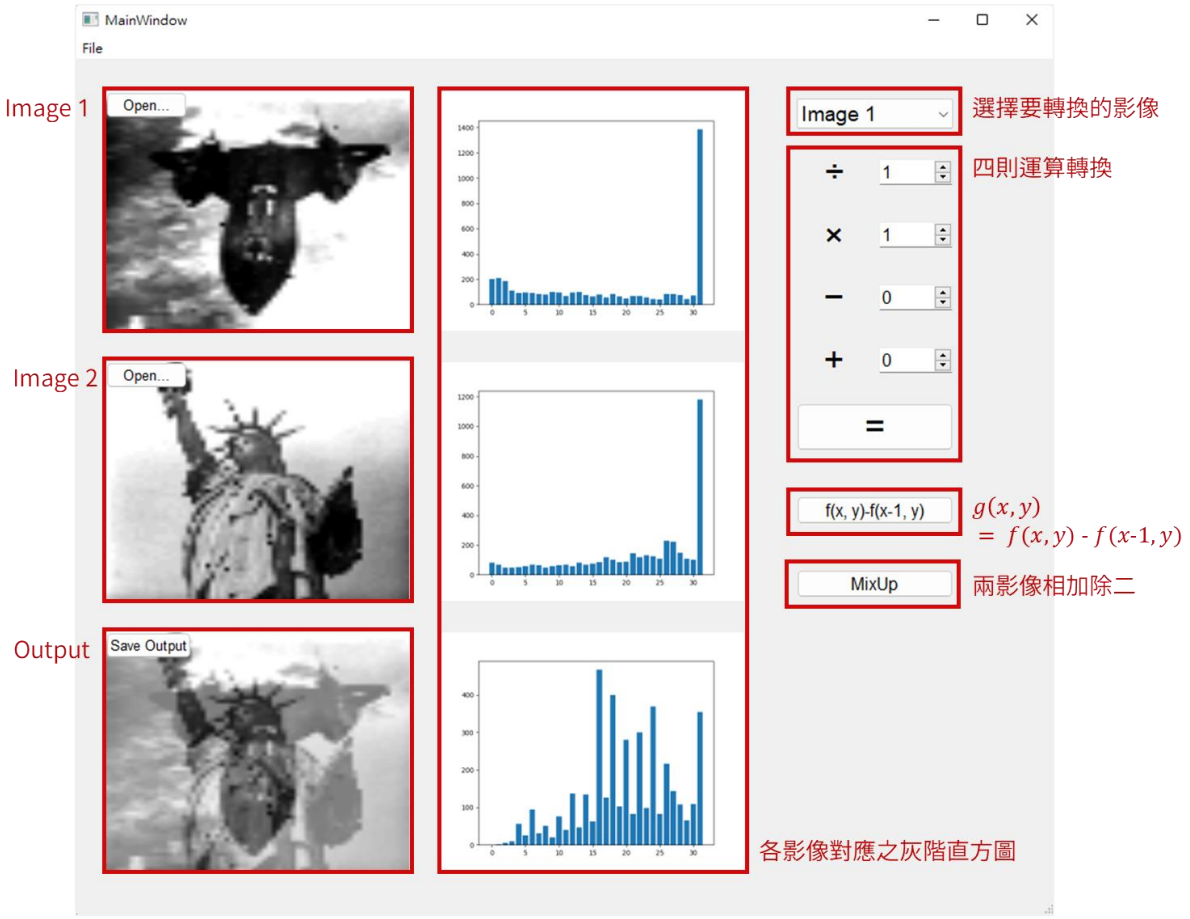
Principles and Applications of Digital Image Processing

Homework 1

GUI

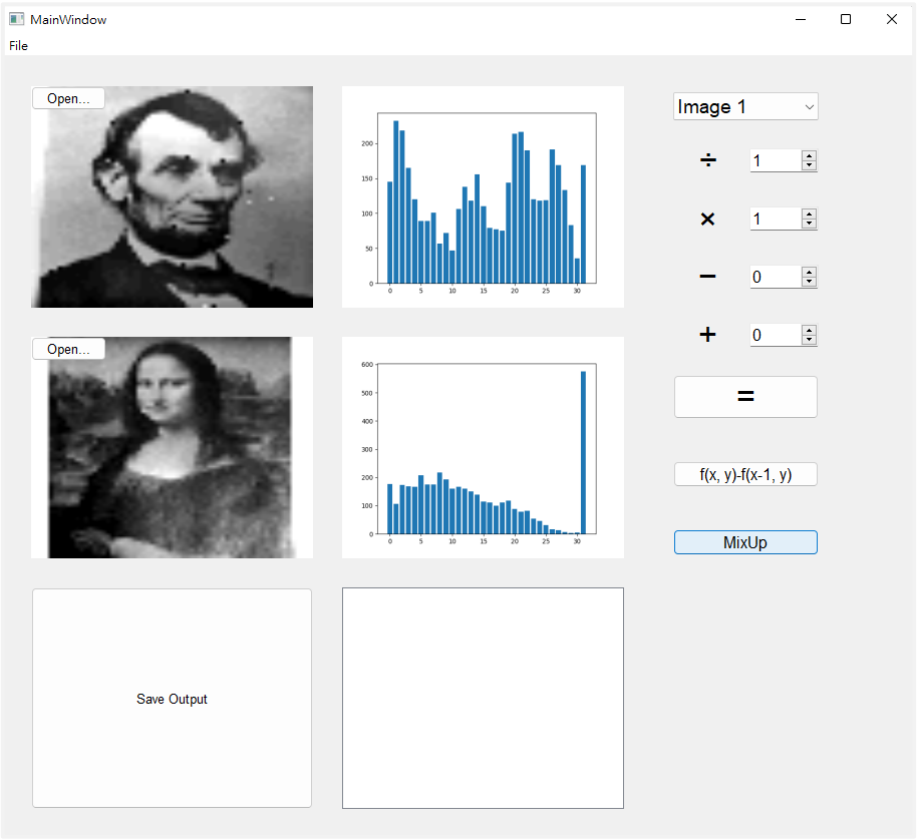


圖一、介面使用說明。

Part 1: (50%) Histogram of an Image

需要讀取.64 檔案並顯示該影像及其灰階直方圖。輸入之.64 檔，透過 dictionary 將 0-9, A-V 轉換成對應的 0-31 個灰階值。為了符合一般 256 階輸出，將影像矩陣值*255/31。統計該影像在 0-31 值的分佈次數以繪製灰階直方圖，結果如圖一。

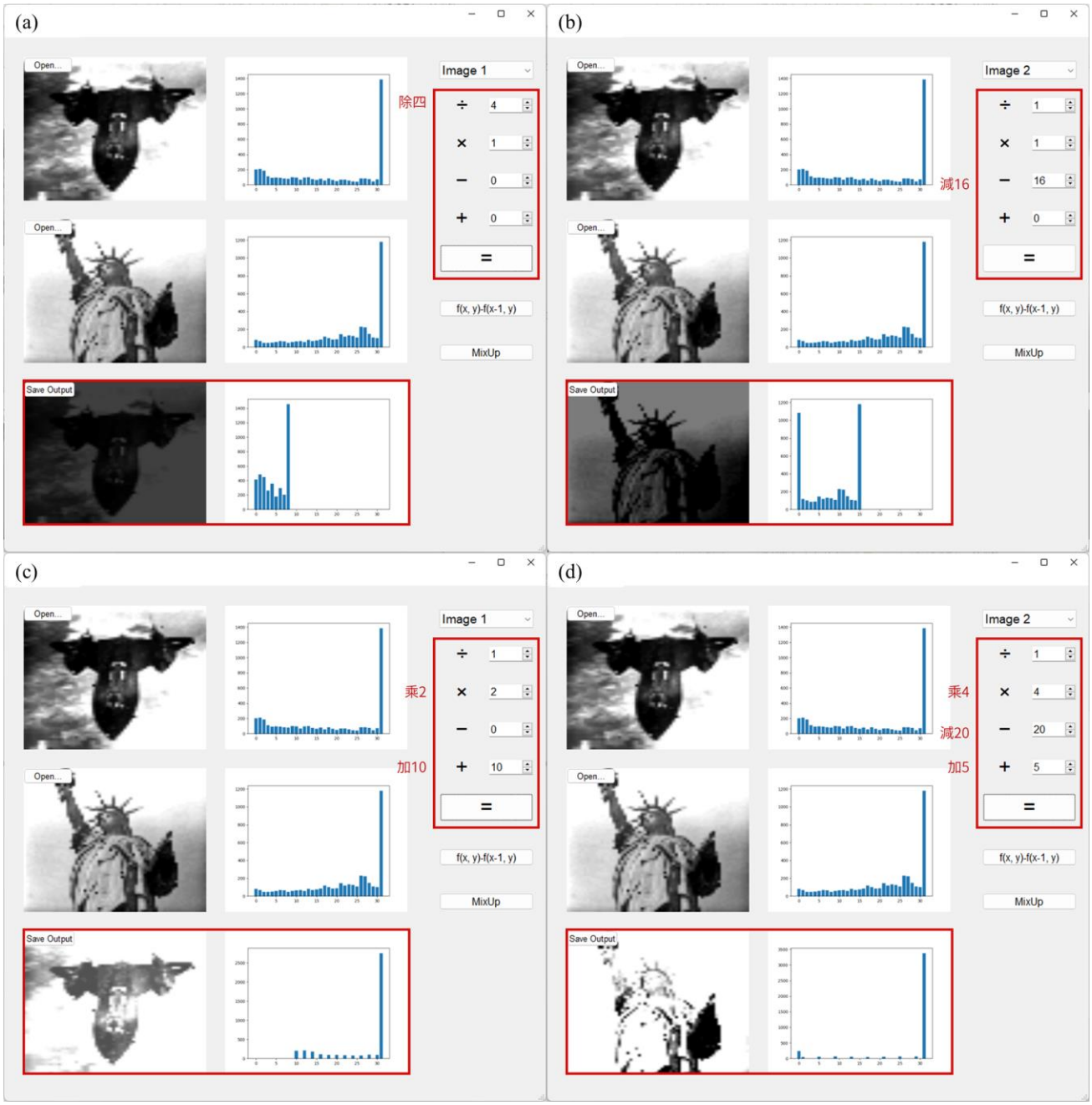
```
# Transfer .64 file to np array

lst = [[b64_dict[x] for x in row] for row in b64]

img = np.array(lst, dtype=float)*255/31

# Grayscale value distribution.

labels, counts = np.unique(img, return_counts=True)
```



圖二、讀取.64 檔之影像輸出與灰階值方圖。

Part 2: (50%) Arithmetic Operations of an Image Array

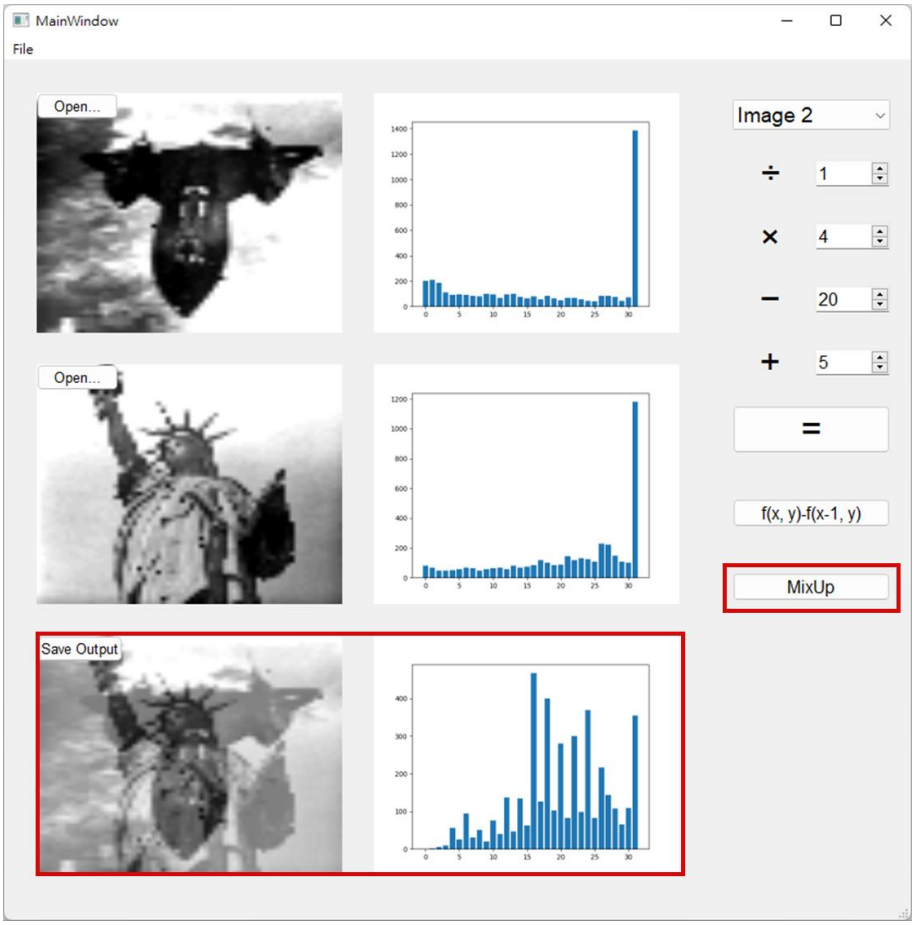
影像的四則運算轉換可透過 SpinBox 決定數值，加減運算被限制在(0, 31)，乘除運算被限制在(1, 31)，按下等號後將執行轉換，運算先乘除後加減，輸出結果如圖三。JET.64 原先之高峰值落在 31，除四以後新高峰值落在 7。LIBERTY.64 原先之高峰值落在 31，減十六以後新高峰值約落在 15。影像數值下降後接近 0，整體色階下降。在圖三(c)(d)影像數值經過乘法後，灰階值區間改變，影像變明亮。



圖三、經四則運算之輸出影像。(a) JET.64 除四，(b) LIBERTY.64 減十六，(c) JET.64 乘二加十，(d) LIBERTY.64 乘四減二十加五。

將兩張影像重疊。輸出結果如圖四，影像保留兩張圖的輪廓，影像交疊呈現半透明。在使用此 MixUp 轉換時，將使用 Image 1 和 Image 2 之原始數值，會忽略四則運算。

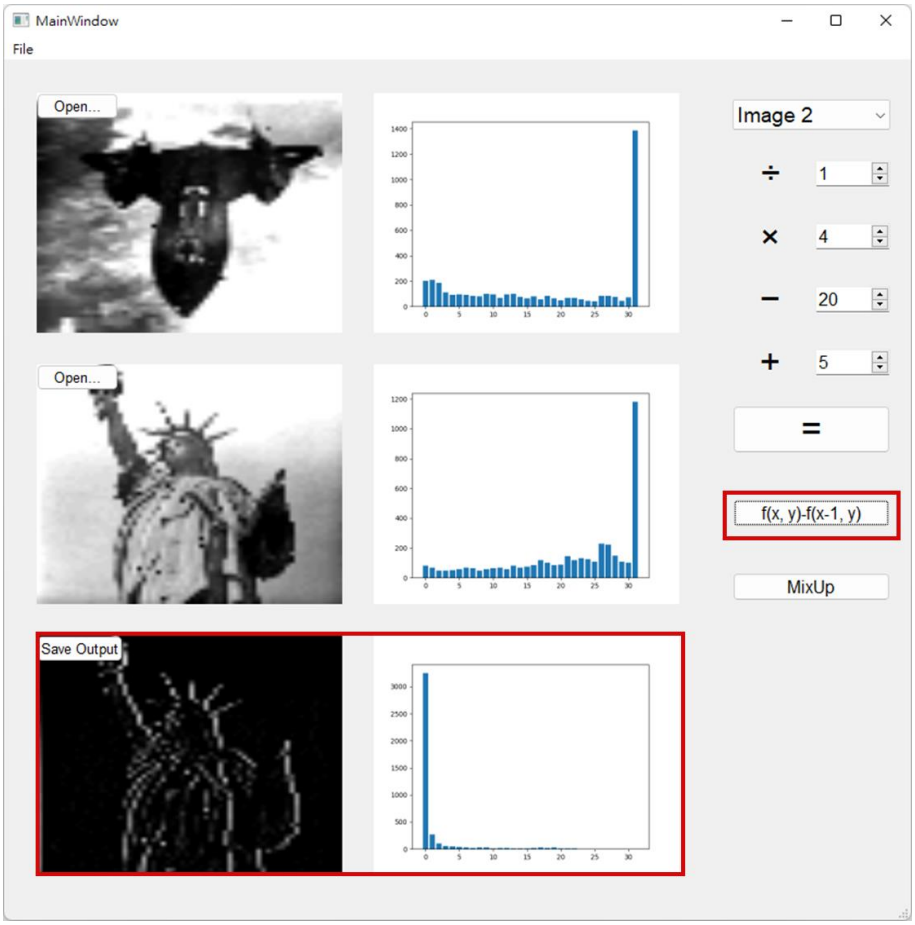
```
self.output_arr = (self.img1_arr+self.img2_arr)/2
```



圖四、重疊兩張影像之輸出。

新的影像為原始影像減去其向右平移一像素之影像，輸出結果保留遠使影像的右側輪廓（圖五）。

```
for i in range(self.output_arr.shape[0]):
    for j in range(self.output_arr.shape[1]):
        left_image[i, j] = self.output_arr[i, j-1]
```



圖五、$g(x,y) = f(x,y) - f(x-1,y)$。