

國立成功大學

資訊工程學系

影像處理

作業一

學生：黃仁鴻

授課教授：孫永年

中華民國一百零九年十一月

一、 開發環境

使用 C/C++ 編寫核心功能並使用 emcc 將其編譯為 Webassembly，使其可以由 H5 (javascript+html+css) 編寫的 GUI 調用，而達成跨平台運行的優勢。

表 1 開發環境

作業系統：	Linux mint 20
編輯器：	vscode
程式語言：	C/C++(核心功能)、javascript+html+css(使用者介面)
C/C++編譯器：	emcc

二、 問題與方法

1. 將輸入的彩色圖片分離成 R、G、B 三個通道的灰階圖，以及將原圖轉為灰階。
 - (1) 分離出對應通道的強度值並複製到 RGB 三個通道使其轉成圖片。
 - (2) 將 RGB 三個通道的強度值分別以 0.299、0.587 與 0.114 的比例相加，得到灰階強度。
2. 實做 mean filter 與 median filter 並比較其去噪結果。
 - (1) 實作出 convolution 功能，並使用值為 1/9 的 3x3 卷積核對原圖做 convolution。

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9
 - (2) 將 3x3 範圍的強度值做排序並取出其中位數作為中心點的數值。
3. 實做直方圖均衡化，將影像明暗進行校正。
 - (1) 計算出各個強度出現的次數後，將其每個強度值依據累積次數進行轉換映射，盡量讓出現次數均勻分散開來。
4. 利用使用者自定義的 threshold，將影像二值化。
 - (1) 依照使用者給定的 threshold 與亮度做比對，如果亮度大於等於 threshold 就將其調整為最高(255)，否則就設為最低(0)
5. 使用 Sobel Edge Detection 過濾出影像的邊緣特徵。
 - (1) 使用問題 2-1 實作出的 convolution，並使用 Sobel operator 計算出圖像梯度並做絕對值，便可得到 X 與 Y 方向的影像邊緣。

G_x

1	0	-1
2	0	-2
1	0	-1

G_y

1	2	1
0	0	0
-1	-2	-1

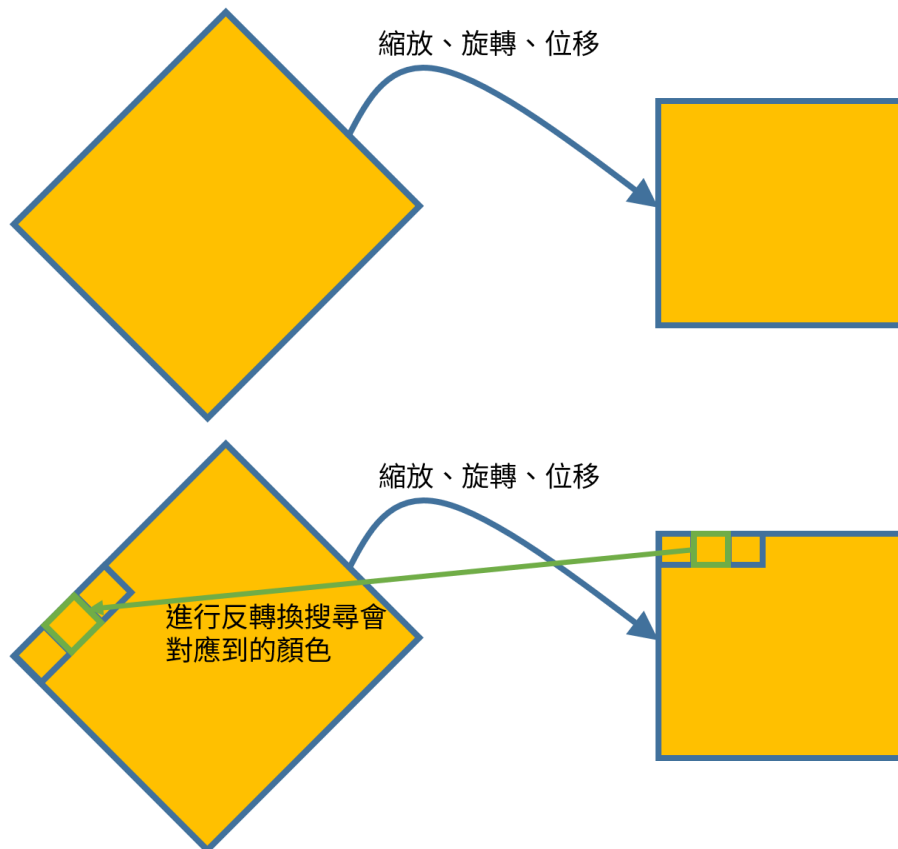
- (2) 將 X 與 Y 方向的影像邊緣計算 $\sqrt{\frac{x^2+y^2}{2}}$ 得出整體邊緣。

6. 將影像的邊緣特徵二值化後與原圖相疊，並以綠色突顯其邊緣。

(1) 利用 5-2 求出的邊緣特徵通過 4-1 的 threshold 去除不明顯的邊緣訊號，之後再將其值為 255 之處轉成綠色(0,255,0)取代原圖。

7. 利用使用者標記的多個點將影像對準。

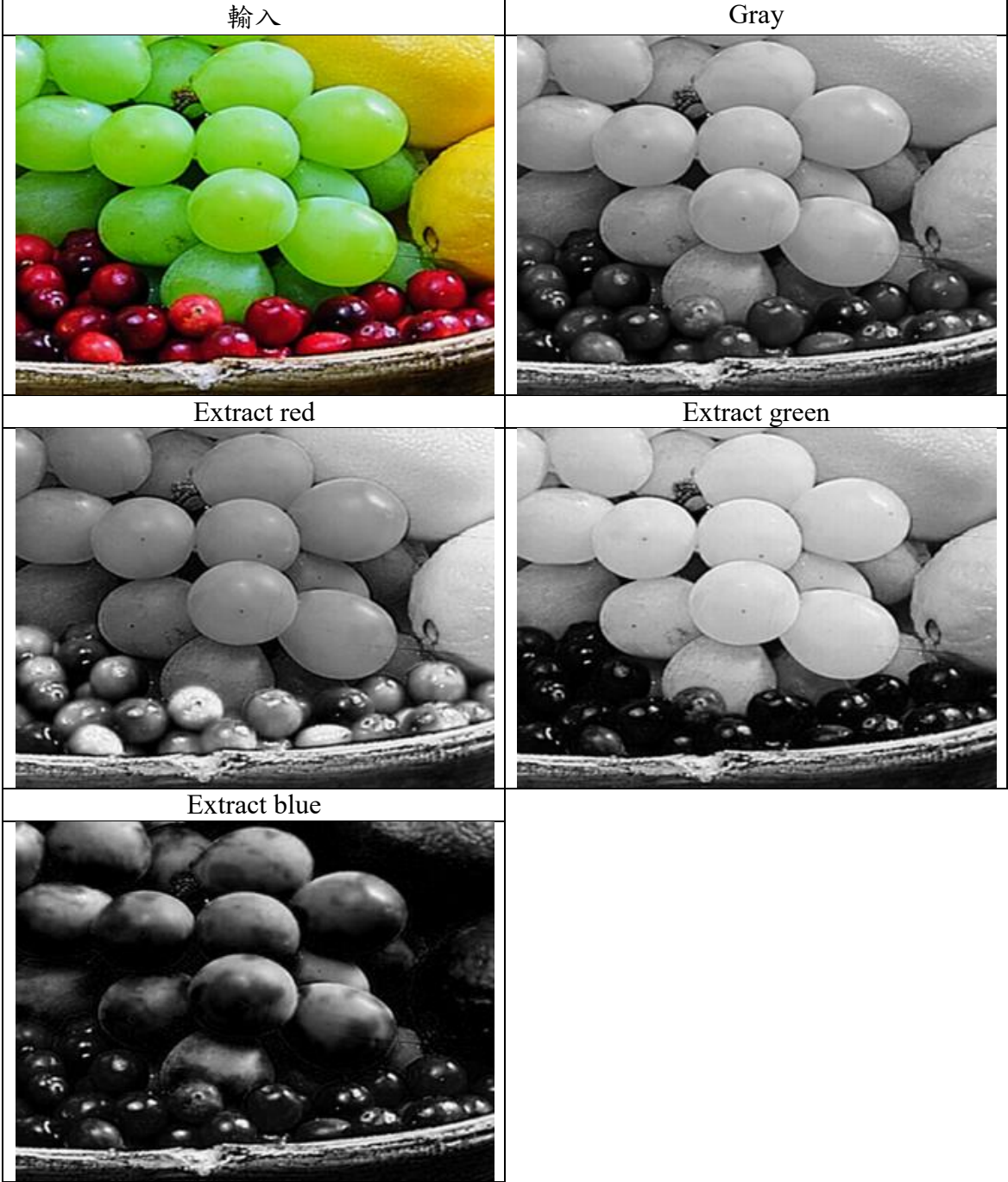
(1) 由使用者在圖像上各點兩點，便可以得出兩張圖的向量與偏移，再由向量計算出旋轉角度和縮放量。有了這些參數後，便可藉由下面的方法計算出轉換後的圖片。



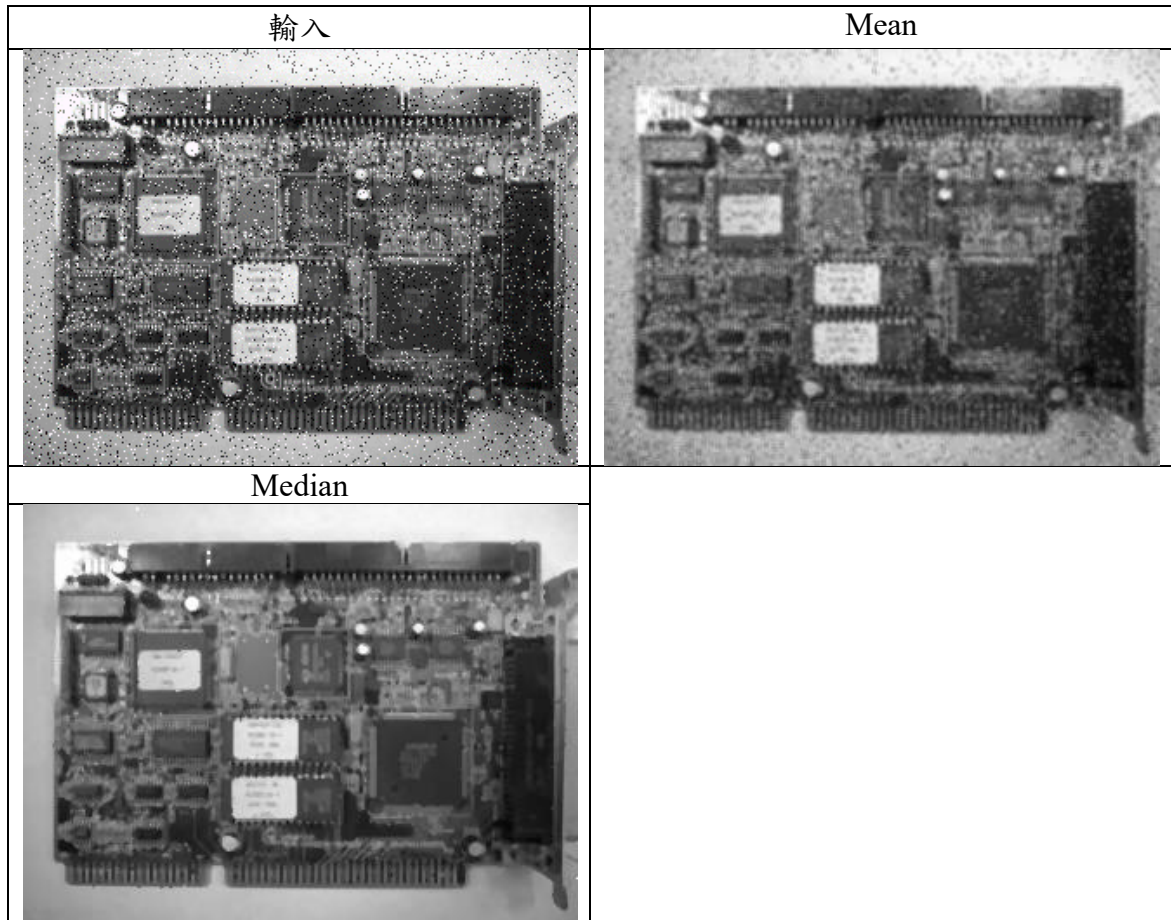
計算時要從轉換完的位置去反向求得原圖位置，才能避免有些位置沒有被映射的情況。

三、 結果

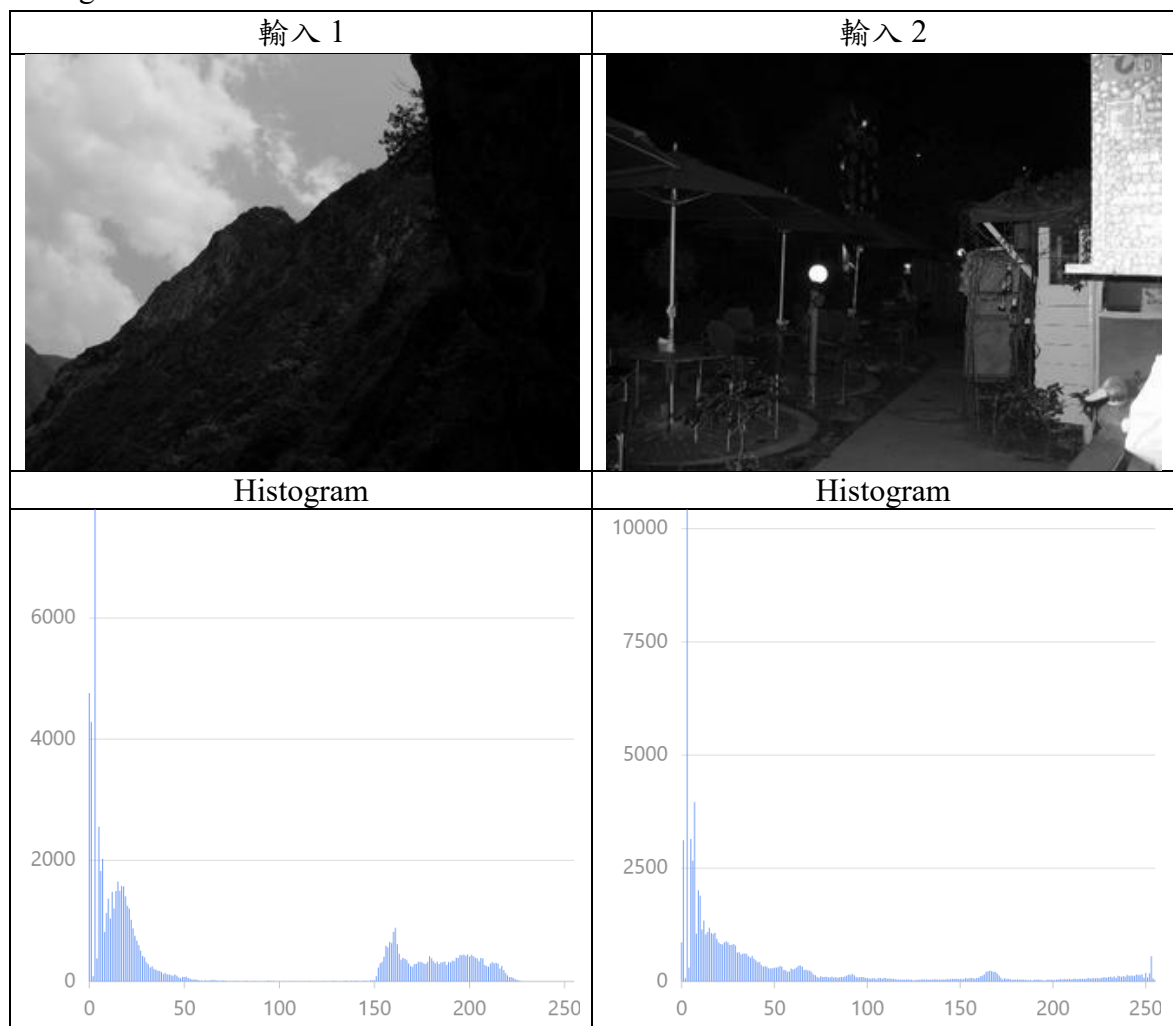
1.



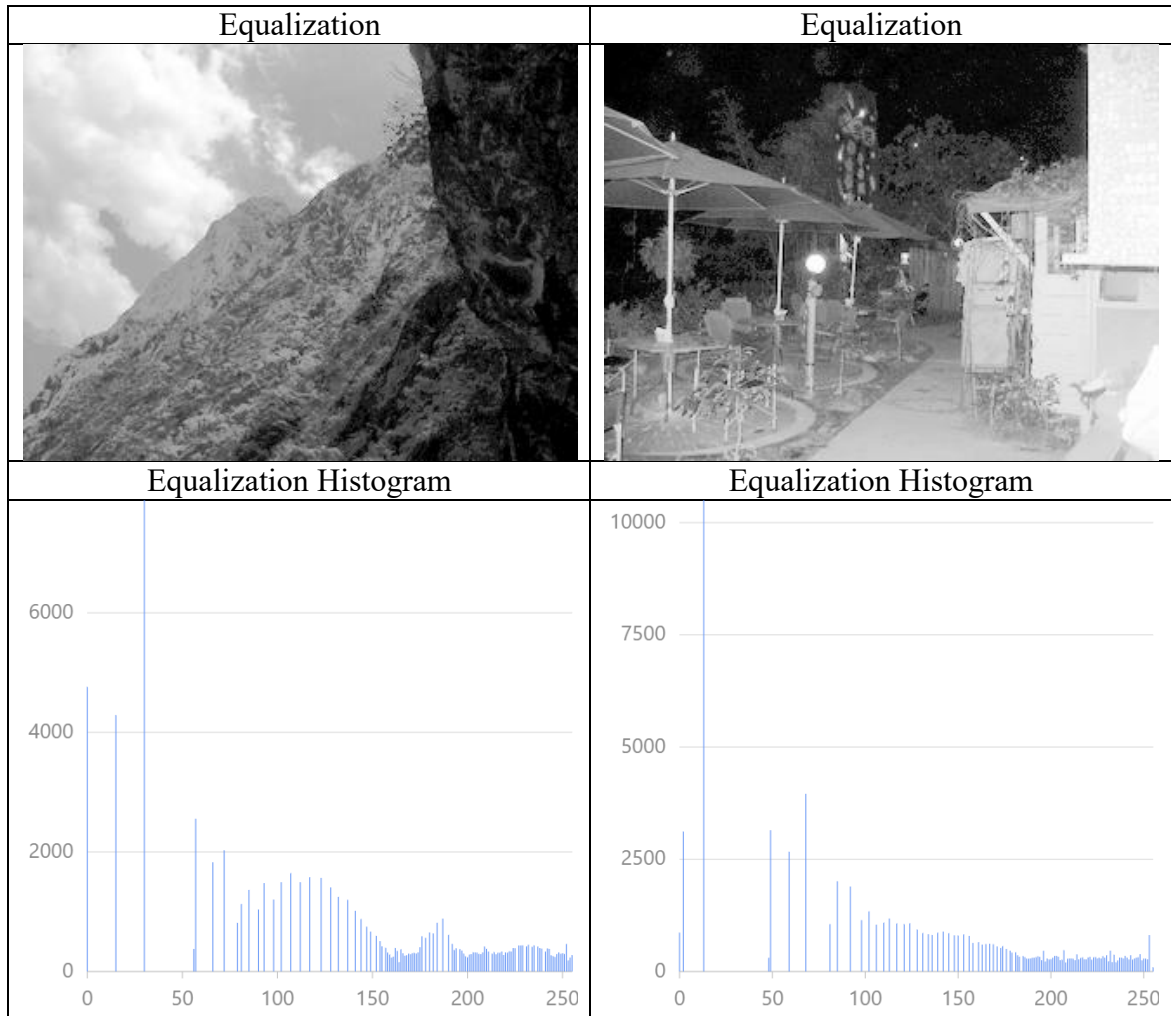
2.



3. Histogram



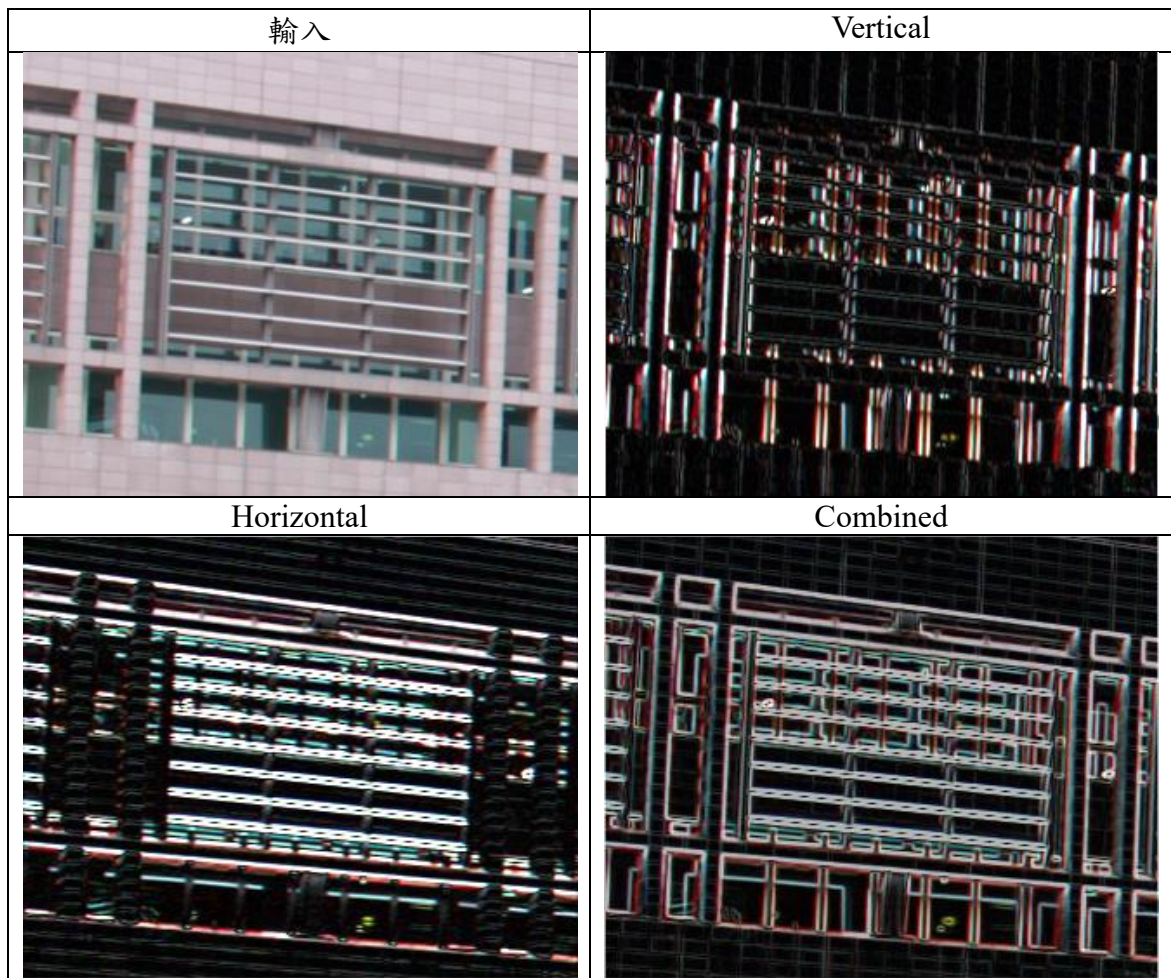
Equalization



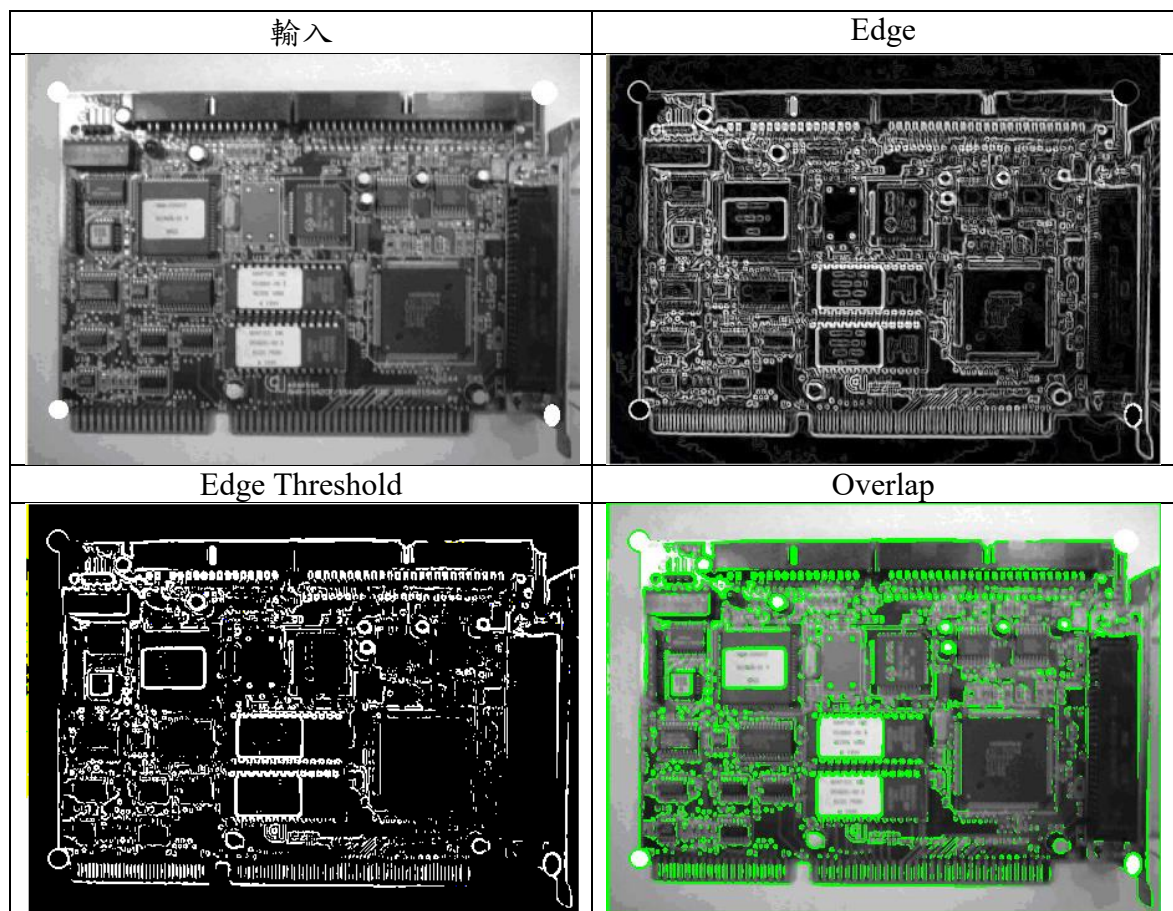
4.



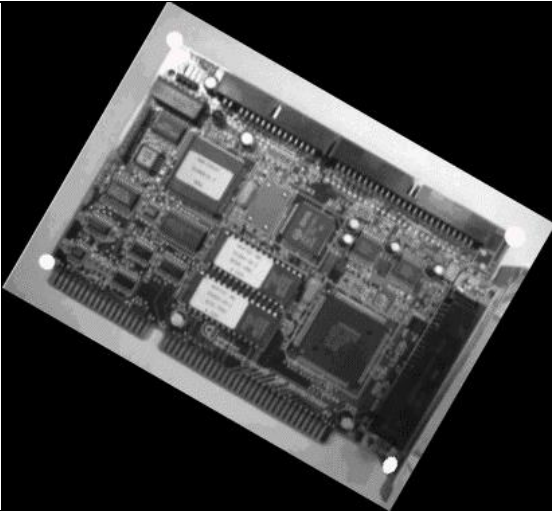
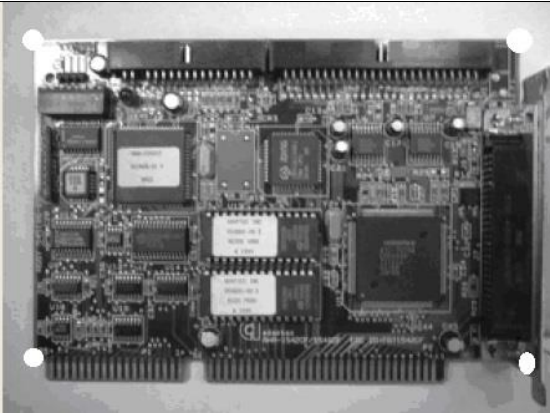
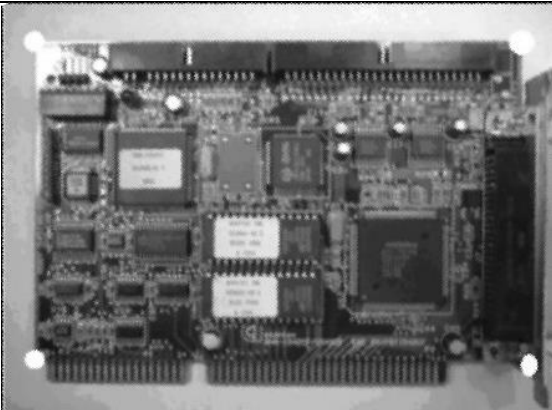
5.



6.



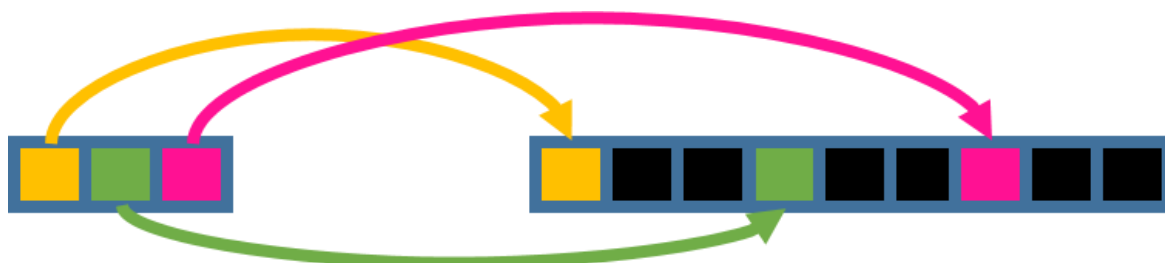
7.

輸入	對應圖片
	
變換後的結果	
	

- 偏移：-103, 4
- 旋轉：30°
- 縮放：1.4
- Intensity Difference：13.17

四、 討論

在第七題時做圖片變換功能時，並不是計算原圖座標轉換後對應到的位置，因為這樣可能會造成對應過去的圖片稀疏破碎的情形。



為了避免這種狀況發生，要改成計算轉換後的座標應該會對應至原圖的哪個位置，如此一來便能保證每個像素都能對計算出相應的數值。此外，還可以藉由雙線性內插法來降低轉換完之後的顆粒感。



另外，相比由電腦視覺搜尋特徵點的方式，人工標記的特徵點反而會出現少許誤差，使得 Intensity Difference 變大。

五、 結論

1. 在影像處理中，實作濾波器或是 Histogram Equalization 等等方法反而比對影像進行旋轉縮放來的簡單。要對影像作偏轉時，還需要顧慮到使用哪個定點為中心，將定點轉換到(0,0)後才能進行選轉的操作是在最開始實作時忽略掉的重點。
2. 與均值濾波器相比，中位數濾波器很明顯更適合用在 salt and pepper 的雜訊上面。