Digital Image Processing HW02 Report

生機碩一 R09631007 吳乙澤

一、課本習題

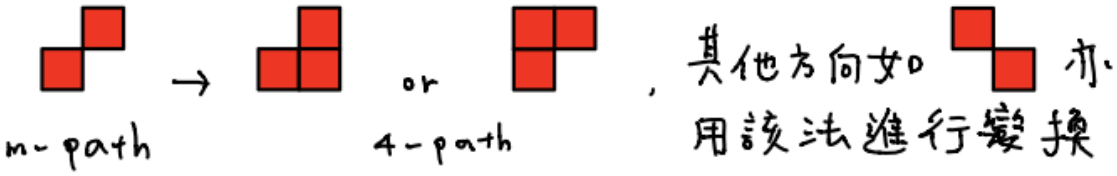
2.12

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

eye can detect abrupt change of eight intensity $\Rightarrow 8 = (255+1)/2^K \quad K=5$

量化至32種灰階後，圖片中任二像素值至少差 8

2.16

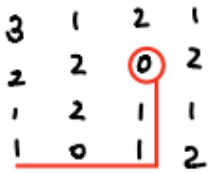


m-path → 4-path or , 其他方向如 亦用該法進行變換

2.18

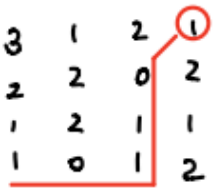
(a) $V=\{0,1\}$

4-path

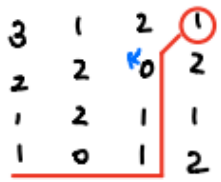


走到0後不能再前進，因4-path內不符 V set

8-path



m-path

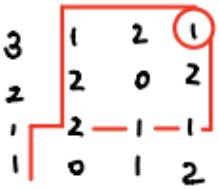


走到K後可以再前進，因符合 $N_4(K) \cap N_4(q)$ 沒有0或1

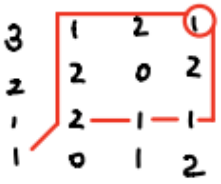
(b)

$V = \{1, 2\}$

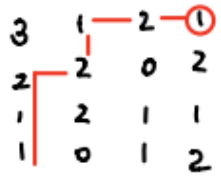
4-path



8-path



m-path



m-path 只會有一條路可走

2.37

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = A \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

(a) inverse scaling

$$\begin{bmatrix} \frac{1}{Cx} & 0 & 0 \\ 0 & \frac{1}{Cy} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

(b) inverse translation

$$\begin{bmatrix} 1 & 0 & -tx \\ 0 & 1 & -ty \\ 0 & 0 & 1 \end{bmatrix}$$

(c) inverse shearing

$$\begin{bmatrix} 1 & -Sv & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$ vertical

$$\begin{bmatrix} 1 & 0 & 0 \\ -Sh & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$ horizontal

(d) inverse rotation

$$\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

(e) composite inverse translation / rotation

$$(b) \times (d) = \begin{bmatrix} \cos\theta & \sin\theta & -tx \\ -\sin\theta & \cos\theta & -ty \\ 0 & 0 & 1 \end{bmatrix}$$

3.12

$(1,0),(0,2) \rightarrow p_r(r) = -2r+2$

$$s = T(r) = (L-1)\int_0^r p_r(w)\,dw = \int_0^r (-2w+2)\,dw = -r^2+2r$$

$$z = T^{-1}(s) = \sqrt{s(L-1)} = \sqrt{-r^2+2r}$$

3.18

(a)

$$w = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad f = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

(b)

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} = 1\times0 + 2\times0 + 1\times0 + 2\times0 + 4\times1 + 2\times0 + 1\times0 + 2\times1 + 1\times0 = 6$$

$$w \star f = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \star \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 2 & 1 & 0 \\ 0 & 3 & 6 & 3 & 0 \\ 0 & 4 & 8 & 4 & 0 \\ 0 & 3 & 6 & 3 & 0 \\ 0 & 1 & 2 & 1 & 0 \end{pmatrix}$$

(c)

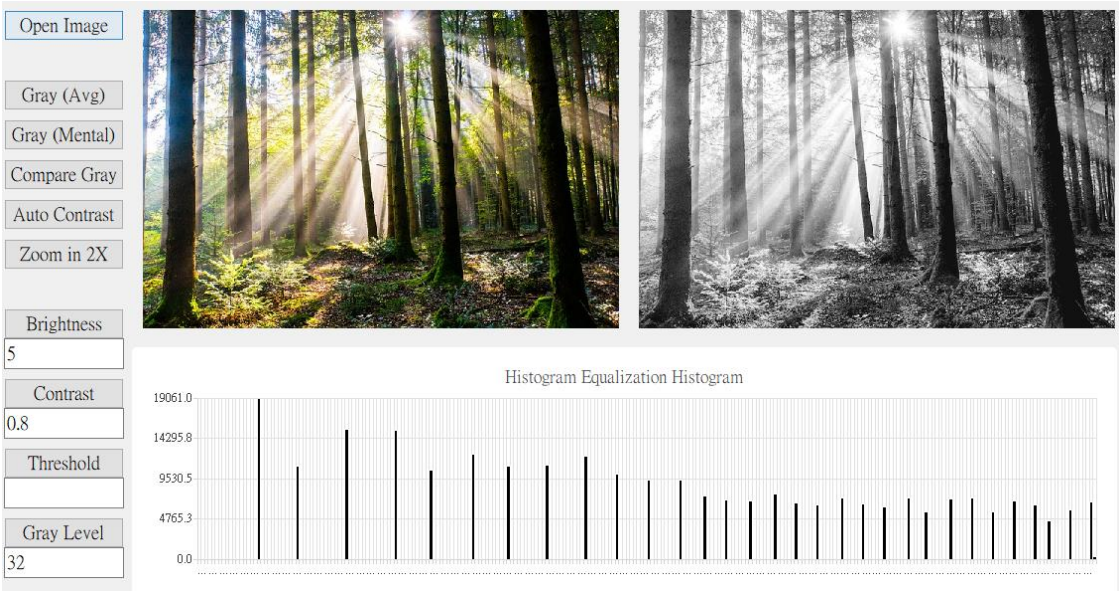
$$w \star f = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \star \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 2 & 1 & 0 \\ 0 & 3 & 6 & 3 & 0 \\ 0 & 4 & 8 & 4 & 0 \\ 0 & 3 & 6 & 3 & 0 \\ 0 & 1 & 2 & 1 & 0 \end{pmatrix}$$

由於w為對稱矩陣，故correlation及convolution出來的結果是相同的

二、軟體簡介

以下簡單介紹本次作業所寫出的軟體，該軟體的使用方法，以及該軟體與市面上影像處理軟體的比較。

2-1 使用方法



上圖為本作業的軟體介面。以下簡單介紹使用方法。首先，需開啟圖片，圖片大小建議大於 480*320，否則顯示會有扭曲或變色現象。按鈕部分，Compare Gray 可把灰階化後的圖片相減，以方便觀察兩種灰階方法的差異。之後，Brightness、Contrast、Threshold、Gray Level 下方有空白框可輸入數字，輸入完畢後，按下按鈕便能對圖片進行各種調整。值得注意的是，Contrast 可接受小數型的數字，輸入多少便代表像素要乘多少。其他如 Brightness、Threshold、Gray Level 則只能輸入整數型的數字，Brightness 輸入多少表像素要加多少，Gray Level 輸入多少表要量化成多少位階。另外，Gray Level 的灰階量化只能往下調，如 128 → 64 → 32 → 16，若 16 → 32 → 64 → 128 則會出錯無法運作成功。最後，Contrast 建議只輸入正數，否則可能有意料之外的問題產生。

2-2 與商業影像處理軟體的比較

本軟體除空間解析度的變化較不自由，只能放大兩倍以外，其他如亮度、對比都能自由調整，不論輸入正數負數，該軟體大致都能處理正確。直方圖等化的效果也很成功。再來，灰階量化可自由調整，不限於 2 的指數倍，且量化後仍能做亮度與對比的調整。最後，經過任何運算後，本軟體都能同步將直方圖顯示在介面中，供使用者參考該圖片的特性。總結，至少在亮度、對比、直方圖等化、灰階量化等功能上，本軟體不輸 PhotoShop 太多。

三、演算法說明與結果討論

本作業有開檔讀檔、灰階化、顯示直方圖、二值化、空間解析度變化、灰階量化、亮度與對比調整、直方圖等化.....等多個功能需完成。以下依序說明這些功能的演算法並對結果做討論。

3-1 開檔讀檔

利用 OpenCV 函數讀取圖片，並將讀取結果存在 Mat 型的資料結構中。之後將該 Mat 變數轉成 QImage 並做縮放後，顯示在 QLable 便能成功顯示圖片。

3-2 灰階化

本次作業用到的灰階演算法有兩種，以下表示為公式 3.1 與 公式 3.2。從數學來分析，可把公式 3.1 展開後化為小數，意即 R、G、B 這三個變數的係數皆為 0.333。以下將公式 3.1 展開後的公式表示為 3.3 。將 3.2 式與 3.3 式做比較，發現公式 3.2 中，綠色的係數最大，紅色和藍色的係數較小。因此，經過 3.2 式的灰階化後，原本綠色的區域會較亮，紅藍色的區域會較暗。經過 3.3 式的灰階化則沒有該特性。見下圖，我取原圖中為紅色的部分進行比較。注意紅色框框所在的地方，對比後可以發現亮度有所區別。

左圖經過 3.2 式的灰階化，右圖經過 3.3 式的灰階化

用上法對兩種灰階運算做分析，我認為不太直觀。因此，我將公式 3.2 減去公式 3.3 並將運算結果表示為公式 3.4。

$$GRAY = (R+G+B)/3.0 \quad (3.1)$$

$$GRAY = 0.299*R + 0.587*G + 0.114*B \quad (3.2)$$

$$GRAY = 0.333*R + 0.333*G + 0.333*B \quad (3.3)$$

$$GRAY = -0.034*R + 0.254*G - 0.219*B \quad (3.4)$$

照公式 3.4 將二圖片相減。相減後，我再做直方圖等化，使對比更加明顯。跑出的結果如下方右圖所示。原本紅色的部分經過相減及直方圖等化後，呈現黑色。這是因為 3.4 式的紅色係數已變負數。另一方面，原本綠色的部分經過相減及直方圖等化後，顏色偏亮。這是因為 3.4 式的綠色係數為正數。

左圖為原圖，右圖為兩圖相減後做直方圖等化

3-3 顯示直方圖

同上次作業，利用 QChart 畫出直方圖。在此不加贅述。

3-4 二值化

在QLineEdit 輸入閥值後，QT 有函數能將該閥值轉為變數。之後，利用 If/Else 將大於該變數的像素區域轉為 255，低於該閥值的區域轉為 0。便能完成手動的二值化。

```
int threshold = ui->threshold->text().toInt();
```

海鷗。手動找閥值，使海鷗與背景分離開來

3-5　空間解析度變化

利用鄰近內插法將原圖放大兩倍，即放大後圖片之長寬為原來的兩倍。我使用兩個 for 迴圈，並在裡層 for 迴圈的 setPixel 處操作 index，因此每跑一次便能填滿一個 2*2 方塊。成果如下圖所示。拉大該圖片後，可以發現樹木的邊緣有明顯的鋸齒狀。若之後的作業以雙線性內插去做放大的話，肯定會平滑許多。

```
for (int i = 0 ; i < imgCols*2 ; i+=2)
{
    for (int j = 0 ; j < imgRows*2 ; j+=2)
    {
        int pixel = imgOutPixel[index];
        pixel = checkPixel(pixel);
        QimgEnlarge.setPixel(i, j, qRgb(pixel, pixel, pixel));
        QimgEnlarge.setPixel(i+1, j, qRgb(pixel, pixel, pixel));
        QimgEnlarge.setPixel(i, j+1, qRgb(pixel, pixel, pixel));
        QimgEnlarge.setPixel(i+1,j+1, qRgb(pixel, pixel, pixel));
        index ++;
    }
}
```

左圖為放大兩倍的程式碼。右圖為放大兩倍後的結果

3-6　灰階量化

將原圖片乘以要轉變的灰階數後，再除以原圖片中最大的灰階數。舉例，quantization 為 16，imgOutPixel[index] 為 178，imgOutMax 為 255。經過第一行程式碼的運算，結果為 11。之後，我利用第二行的程式碼，拉大像素間距使圖片顯示清楚。經過運算，像素值從 11 到 175。

```
int pixel = int(quantization*imgOutPixel[index]/imgOutMax);
pixel*=(imgOutMax/quantization);
```

右圖為量化後的 16 灰階圖片。如教科書所說有 False Contouring 現象。

3-7　亮度與對比調整

對原圖片中的像素做加減乘除，並限制像素在0~255間，便能成功調整圖片的亮度與對比。以下秀出一些經過該軟體調整後的圖片。

樹林。在 Contrast 處輸入負數，使圖片有黑白倒反的效果

胡蘿蔔。灰階量化後調亮度跟對比

3-8　直方圖等化

照著直方圖等化的公式，獲得像素關係轉換表。該表為一維矩陣，矩陣名稱為 equalizationTable。舉例，equalizationTable[2] 中儲存的元素，即為原圖片中像素值為 2 的區域所要轉換成的像素值。利用該表即可完成直方圖等化。見下圖，直方圖等化能調整對比，使圖片變得較為清晰，暗部細節可視化。

```
equalizationTable[i] = (equalizationTable[i]-minCDF)/ (maxCDF -
minCDF)*255;
pixel = int(equalizationTable[checkPixel(imgOutPixel[index])]);
```

烤肉，右圖為經過直方圖等化後的結果