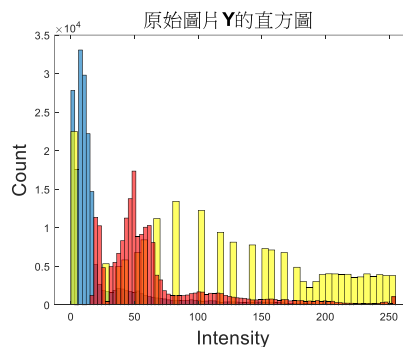
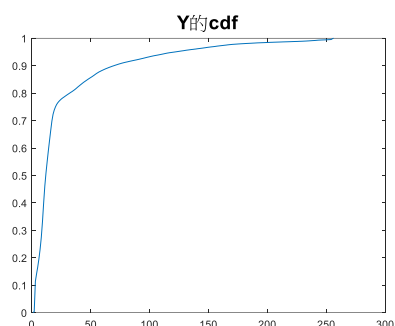
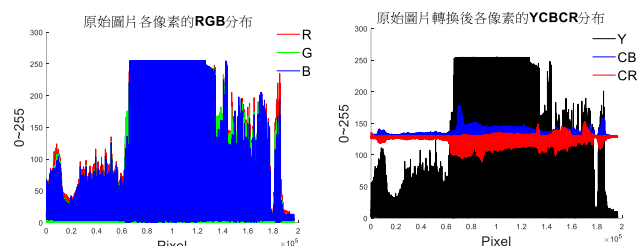


Digital Image Processing (2024) Hw2 Report

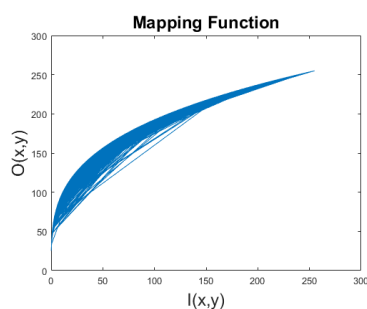
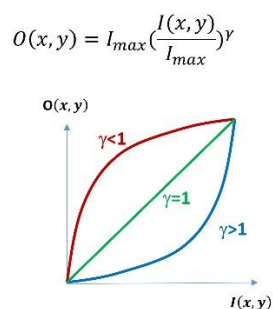
電子碩二 312510198 黃育豪



1. Low-luminosity Enhancement

這題我試了兩種方法，並且兩種方法都是如右圖將 RGB 轉換成 YCBCR 進行處理後再轉換回 RGB 域輸出，首先使用了 Histogram Equalization 的方式，可以看到尚未處理的 Intensity 統計直方圖是藍色的波型並集中在左半邊，這代表整體亮度偏暗，較為理想的分佈為在保有原始波型樣貌的前提下將其平均拉寬，以使得暗的地方不會太暗亮的地方不會太亮。

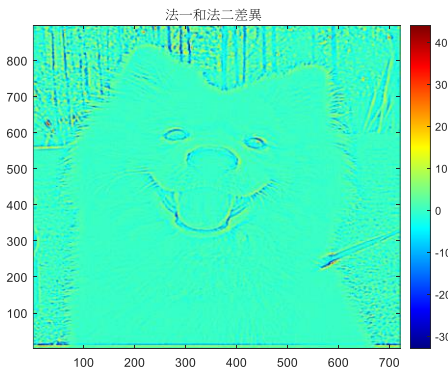


因此我算出原始 Y(0~255)的機率分佈並將其累加成為機率累積函數 CDF，而利用上圖中的 CDF 再乘上 255 當作 Mapping Function 的好處是它在機率高的地方斜率較大，而斜率大就會把轉換後的 Intensity 拉寬，轉換後為黃色的直方圖，成功將圖片亮度提升，但這個方法我看起來亮部有點太亮了於是試了另一個方式 Gamma Correction，最後測試 $r=0.3$ 時整體色彩還原度最高，如紅色的直方圖所示。






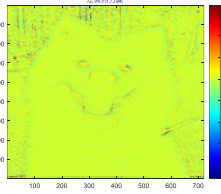
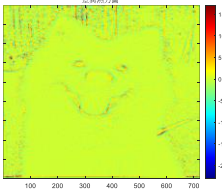
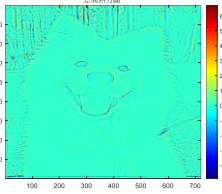
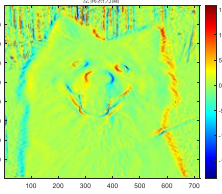
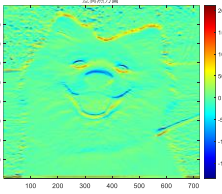


	Histogram Equalization	Gamma Correction($r=0.3$)
效果		

2. Sharpness Enhancement



這題我總共試了 5 種不同的 Filter，因為原圖看起來很模糊所以我們必須對它做銳化以
提升每個像素的邊緣，而銳化的方式就是決定好 Mask size 後選擇一個較適合的 Mask 並對圖像
的每個像素進行 Convolution, 像是第一個 Laplacian Filter 就是利用像素點的邊緣去調整中
央像素的值，而 Unsharp Masking 則是只利用中央像素的上下左右來調整，兩種方式調整出來
的色彩差異如上圖所示，另外我還有試了 Sobel_H 和 Sobel_V 的 Filter, Sobel_H 是會增強圖中
垂直方向的銳化，而 Sobel_V 是增強圖中水平方向的銳化，像是狗狗的雙眼皮就變得很清晰，
這幾種 Filter 看下來銳化效果最好的是 Sobel_V，但銳化最好不代表影像處理的最好，看起來
會沒那麼真實，我覺得整體來說 Laplacian 看起來比較自然。

Filter	Laplacian	Unsharp Masking	Gaussian	Sobel_H	Sobel_V
Mask	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -7 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 1 & 2 \\ -1 & 0 & -1 \end{bmatrix}$	$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 1 & 0 \\ 1 & 2 & 1 \end{bmatrix}$
效果					
原圖差異					

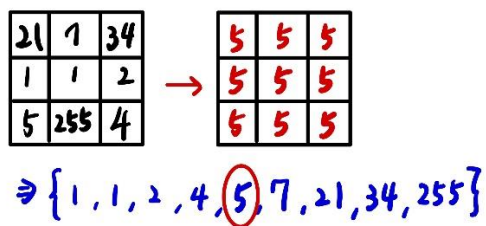
3. Denoise



這題我總共使用了三種不同的濾波器來線性組合去除圖像中的噪點，分別為 3x3 Gaussian Filter 和 3x3 Median Filter 以及 5x5 Median Filter，我使用的 Gaussian Filter Mask 如下

$$\frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

它可以用來消除噪點的原理是從中心像素往外擴散的權重慢慢遞減，這樣當 Convolution 的時候就可以降低噪點對於整張圖的影響。而 Median Filter 的原理在於取出 Mask size 中的數值進行排序，排序完後以中位數取代 Mask Size 中的所有值，做法如下圖，這樣做的好處是可以去除周圍像素中的離群值 outlier。



● Input3

Input3	Gaussian Filter	Median Filter(3x3)	Median Filter(5x5)
效果			
原圖差異			
SSIM	0.34	0.85	0.78

對於 Input3 來說使用 3x3 的 Median Filter 效果最好，因為其中的噪點都是黑色(0)或是白色(255)，所以誠如前面所說離群值被濾波器濾掉了，而原本以為使用 5x5 的 Median Filter 效果又會更好，但試了之後發現因為 Mask size 過大導致了原本的訊號失真，產生了像是馬賽克的效果，雖然邊緣的噪點變少了但是也失去了原本的圖像。





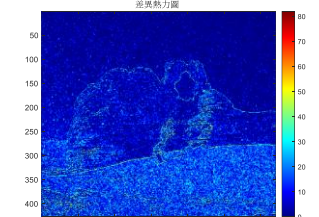
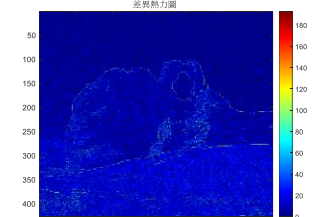
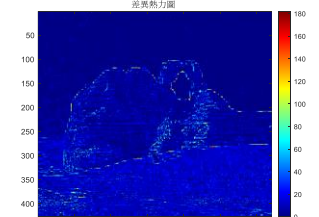
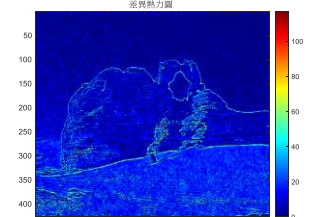
但並不是什麼噪點都適合用 Median Filter 消，像是 Input4 的噪點色彩都不一樣，Median Filter 的效果就沒那麼好，經過我的測試最後經過五次的 Gaussian Filter 效果最佳。但可以看到我有使用 Matlab 畫出濾波後跟原始沒有噪點的 Input4_org.bmp 做比較，雖然第四個方法的噪點看起來比 Median Filter 還多但 SSIM 卻比較好，因此我問了 chatgpt 以下是他的回答。

1. SSIM 對結構的強調
SSIM 的計算方式特別關注圖像的局部結構特徵。若「方法二」保留了原始圖像的整體結構和細節，而只是有些額外的雜訊或小範圍的偏差，這些對 SSIM 的影響會較小，因而 SSIM 仍可能接近 1。方法一雖然在視覺上可能更清晰，但如果改動了較多的局部結構（比如邊緣細節），可能會導致 SSIM 分數降低。

2. SSIM 不完全考慮單點的像素差異
SSIM 更注重局部區域的對比度和結構，而不完全計算單點像素的準確相似度。換句話說，即便方法二看起來噪點較多，但只要其區域內的整體結構、亮度和對比度與原圖接近，SSIM 可能依然較高。

3. 可能的均值偏移
如果「方法二」的處理方式使得整體灰階值更接近原圖，這也會使 SSIM 分數更高，因為 SSIM 包含亮度和對比度的比較。這樣，即使在個別區域存在偏差，SSIM 可能仍會視為與原圖更相似。

4. SSIM 不適合所有噪點型態
SSIM 對於大範圍、結構性失真的評估較為準確，但對小範圍的隨機噪點則不一定敏感。方法一可能消除了更多隨機噪點，而方法二的消噪方式在結構性上更接近原圖，因此 SSIM 更高。

Input4	Gaussian Filter	Median Filter(3x3)	Median Filter(5x5)	Gaussian*5
效果				
原圖差異				
SSIM	0.69	0.56	0.63	0.75

● 使用 Matlab 的 SSIM 函數計算 SSIM

```
% 載入圖片
img1 = imread('Input4_org.bmp');
img2 = imread('output4_1.bmp');

% 轉為灰度圖
img1 = rgb2gray(img1);
img2 = rgb2gray(img2);

% 使用 MATLAB 內建 ssim 函數
ssim_value_builtin = ssim(img1, img2);
fprintf('內建 SSIM: %.4f\n', ssim_value_builtin);
```

• A common setting: $\alpha = \beta = \gamma = 1, C_3 = \frac{C_2}{2}$

$$\Rightarrow SSIM(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \frac{2\sigma_{xy} + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}$$

參考資料

<https://blog.csdn.net/hedgehog/article/details/107257755>