

# Digital Image Processing Homework 2 Report

109511207 蔡宗儒







## 1. Low-luminosity Enhancement

本題中我使用了兩種方式來調整亮度，第一個是簡單粗暴地將每個 pixel 的灰階加上一個數值，如果計算結果灰階超出 255 的話就讓此 pixel 的灰階等於 255，這樣就能很簡單地將照片整體亮度提高。




第二種方式是透過 Gamma Correction 來調整。公式為  $O(x,y) = I_{max} \left( \frac{I(x,y)}{I_{max}} \right)^\gamma$ 。  $\gamma < 1$  的話圖像會變亮；  $\gamma > 1$  的話圖像則會變暗。雖然將每個 pixel 的灰階增加或減少就能夠輕易做到調整亮度，但假如每個 pixel 都被調高 20 好了，那原本灰階落在 235~255 的值就都會變成 255，會導致這張圖片失去一些細節，如果調的更亮的話就會丟失更多細節了，而因為 Gamma Correction 是嚴格遞增函數，所以可以解決丟失細節的問題。

實作結果如下，可以觀察到兩種方式皆有效地將圖像調亮，  $\gamma$  越小也的確讓圖像看起來更亮。

### Result 1: Brute Force (I choose factor = 20 & 40)

	Original	Factor = 20	Factor = 40
i n p u t 1			
L e n a			

### Result 2: Gamma Correction (I choose $\gamma = 0.75$ & $0.5$ )

	Original	$\gamma = 0.75$	$\gamma = 0.5$
i n p u t 1			



## 2. Sharpness Enhancement

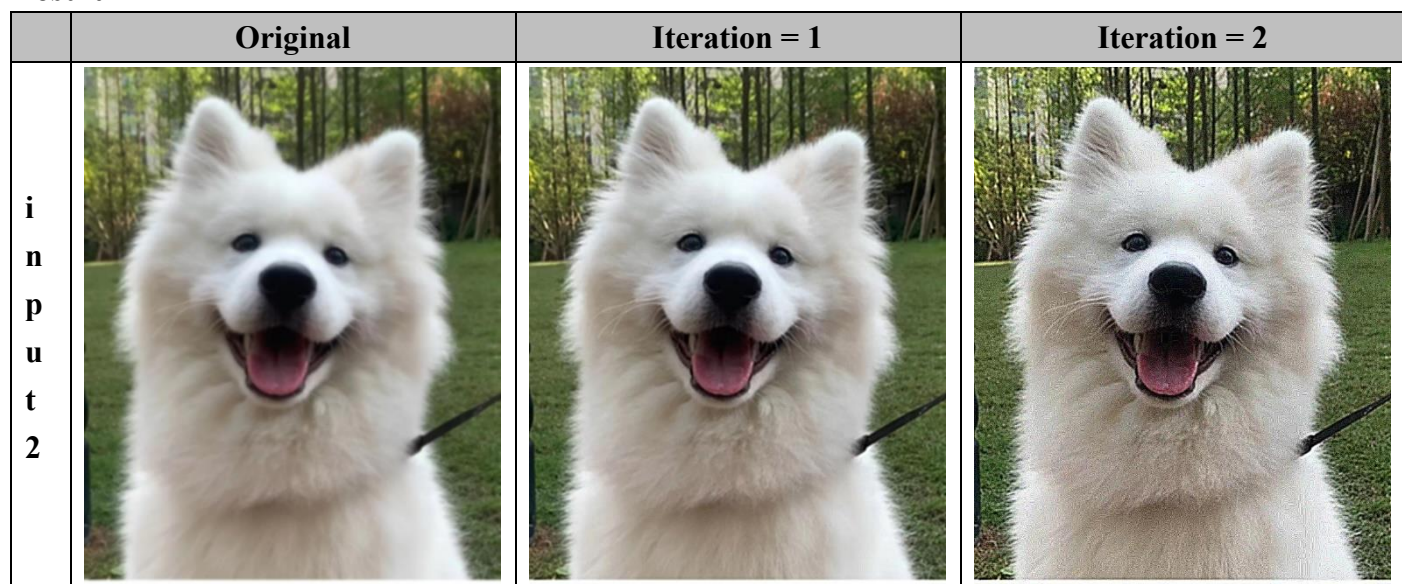
Sharpness enhancement 的原理是增強圖像邊緣的細節，也就是加強高頻成分，這樣能讓圖像中的邊緣處看起來更加清晰，但同時也會放大圖像中的小波動和雜訊。而我在本題中使用了 kernel size 為 3 的 Composite Laplacian Mask 對影像做 convolution 運算。我實作 convolution 的作法是將每次需要計算的 9 個 pixel 存入一個矩陣中，再傳入 filter function 中逐項相乘後累加，最後再回傳 convolution 計算後的結果。而如果遇到邊界的 pixel 的話，我使用了 replication padding 來填充矩陣，我發現如果使用 zero padding 的話會在影像周圍產生一圈黑框，會顯得有點突兀。最後我設置了不同迭代次數來產生兩個結果。

-1	-1	-1
-1	9	-1
-1	-1	-1

Composite Laplacian Mask

實作結果如下，由 input2.bmp 確實可以觀察到狗狗的邊緣和背景的樹林變得更加銳利清晰了，然而觀察 Lena 迭代一次的圖會發現，儘管帽子上的毛線確實看起來銳利不少，線條看起來更加明顯，但卻也同時放大了背景部分的小雜訊，讓整個影像看起來一點一點，變得較尖銳的感覺。

## Result







### 3. Denoise

Denoise 的原理是用低通濾波器把圖像中的高頻部分去除掉，而因為 Gaussian noise 為 zero mean，所以用平均的方式可以抑制每個 pixel 附近有高有低的 noise，然而這樣也會降低細節層次，讓圖像邊緣變得沒那麼銳利，會看起來更為模糊。而 filter 的 kernel size 也會影響 denoise 的效果，size 越大的話，denoise 效果會更好，但圖像也會越模糊。本題中我分別用了 kernel size 為 3 的 mean filter、Gaussian filter 以及 median filter 對影像做 convolution 計算。實作方式與第二題雷同，將每次需要計算的 9 個 pixel 存入一個矩陣中，再傳入 filter function 中做 convolution 計算或是 sort 取中位數。最後我同樣使用了 replication padding 來填充邊緣的 pixel，並設置不同迭代次數來產生兩個結果。





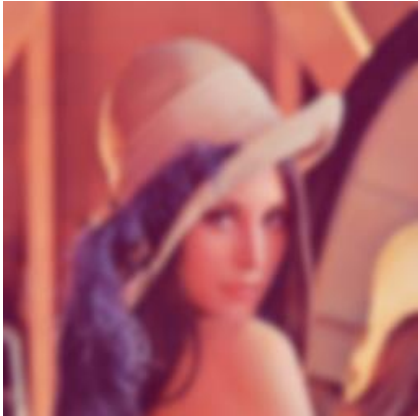
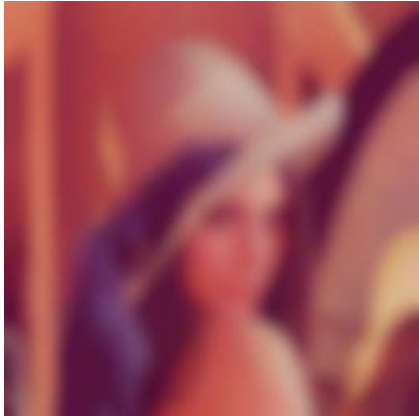
1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Mean Filter

1/16	1/8	1/16
1/8	1/4	1/8
1/16	1/8	1/16





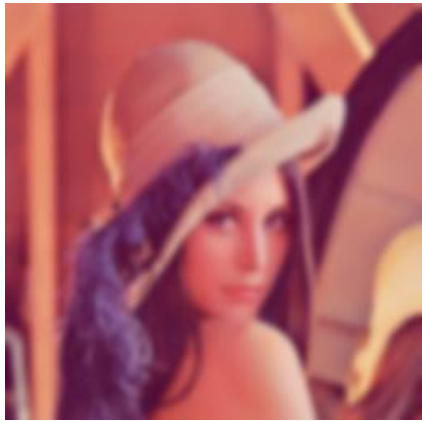
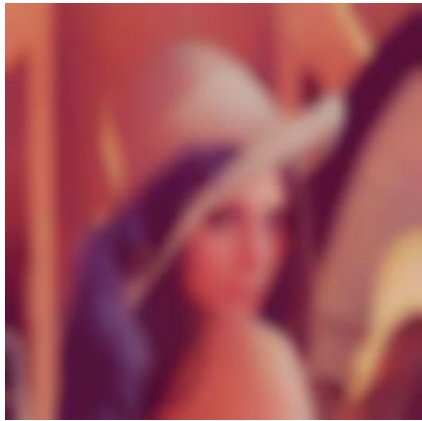
Gaussian filter

Result 1: Mean Filter





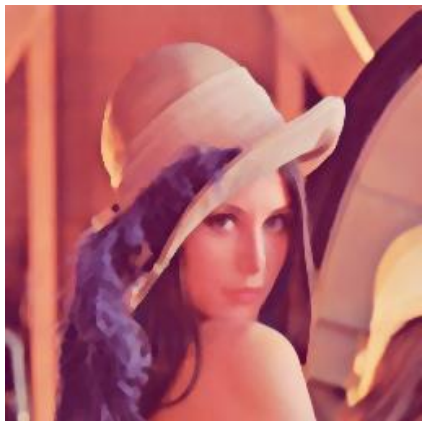
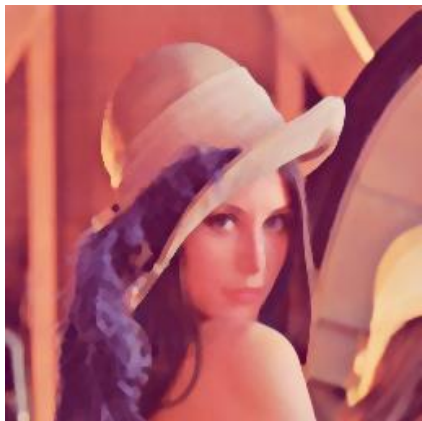
	Original	Iteration = 10	Iteration = 50
i n p u t 3			
L e n a			



## Result 2: Gaussian Filter

	Original	Iteration = 10	Iteration = 50
i n p u t 3			
L e n a			

## Result 3: Median Filter

	Original	Iteration = 10	Iteration = 50
i n p u t 3			
L e n a			

實作結果如上，觀察 input3.bmp，看起來 mean filter 和 Gaussian filter 的 denoise 效果差不多，迭代越多次 noise 越少，但圖像也越模糊，可能是 kernel size 設置太小，Gaussian filter 才沒有較顯著的效果。而 median filter 在迭代 10 次的情況下明顯有比 mean filter 和 Gaussian filter 有更好的效果。再往下繼續迭代至 50 次後，denoise 效果則無明顯提升，但圖像相比 mean filter 和 Gaussian filter 更為清晰。觀察 Lena 則會感覺使用 median filter 的圖象雖較清晰，但圖像中的人和背景變得較不立體。

## 4. Reference

[1] [銳度](#) [2] [Edge enhancement](#) [3] [伽瑪校正 Gamma Correction](#)