# CV Homework 8

R02922124 葉信良

$ gcc –o hw8 hw8.cpp

Part 1 Generate additive white Gaussian noise

我直接使用c++讀入bmp file lena.bmp，呼叫Read\_image來將header 1078個字元直接寫入到output，並將image的數值0~255存到unsigned char array BBMap中。接著根據輸入的amplitude來呼叫Generate\_Gauss\_Noise來算出result image，存到unsigned char array OMap中。

在Generate\_Gauss\_Noise中，我將每個pixel都加上amplitude乘以Gauss\_rand產生出來的亂數，來當作result image pixel的數值。在Gauss\_rand中，我應用了Central Limit Theorem來產生Gauss亂數。

$ ./hw8 g\_gauss 10  
圖片檔名是Gauss\_10.bmp，SNR為13.5878  


$ ./hw8 g\_gauss 30  
圖片檔名是Gauss\_30.bmp，SNR為4.16048  


Part 2 Generate Salt-and-Pepper noise

我直接使用c++讀入bmp file lena.bmp，呼叫Read\_image來將header 1078個字元直接寫入到output，並將image的數值0~255存到unsigned char array BBMap中。接著根據輸入的threhold來呼叫Generate\_Salt\_Pepper來算出result image，存到unsigned char array OMap中。

在Generate\_Salt\_Pepper中，我將每個pixel都做了幾個判斷，先隨機gen出一個0~1的數值，如果這個數值小於threhold的話，那就以0來當作result image pixel的數值；如果這個數值大於1-threhold的話，那就以255來當作result image pixel的數值；其他的情況則是用input image原來的數值。

$ ./hw8 g\_sandp 0.05  
圖片檔名是SandP\_0.05.bmp，SNR為0.954258  


$ ./hw8 g\_sandp 0.1  
圖片檔名是SandP\_0.1.bmp，SNR為-2.09278  


Part 3 Run Box-filter(3x3, 5x5) on all noisy image

我直接使用c++讀入bmp file lena.bmp，呼叫Read\_image來將header 1078個字元直接寫入到output，並將image的數值0~255存到unsigned char array BBMap中。接著根據輸入的matrix size來呼叫Box\_Filter來算出result image，存到unsigned char array OMap中。

在Box\_Filter中，我將每個pixel的周圍3x3(5x5)矩陣內的數值都加在一起，除以matrix size，作為那個pixel result image的值。

$ ./hw8 box 3 Gauss\_10.bmp  
圖片檔名是Box\_3\_Gaussian\_10.bmp  


$ ./hw8 box 3 Gauss\_30.bmp  
圖片檔名是Box\_3\_Gaussian\_30.bmp  


$ ./hw8 box 3 SandP\_0.05.bmp  
圖片檔名是Box\_3\_SandP\_0.05.bmp  


$ ./hw8 box 3 SandP\_0.1.bmp  
圖片檔名是Box\_3\_SandP\_0.1.bmp  


$ ./hw8 box 5 Gaussian\_10.bmp  
圖片檔名是Box\_5\_Gaussian\_10.bmp  


$ ./hw8 box 5 Gaussian\_30.bmp  
圖片檔名是Box\_5\_Gaussian\_30.bmp  


$ ./hw8 box 5 SandP\_0.05.bmp  
圖片檔名是Box\_5\_SandP\_0.05.bmp  


$ ./hw8 box 5 SandP\_0.1.bmp  
圖片檔名是Box\_5\_SandP\_0.1.bmp  


Part 4 Run Median-filter(3x3, 5x5) on all noisy image

我直接使用c++讀入bmp file lena.bmp，呼叫Read\_image來將header 1078個字元直接寫入到output，並將image的數值0~255存到unsigned char array BBMap中。接著根據輸入的matrix size來呼叫Median\_Filter來算出result image，存到unsigned char array OMap中。

在Median\_Filter中，我將每個pixel的周圍3x3(5x5)矩陣內的數值都放進一個int array num中，然後呼叫algorithm裡的sort function來對num做排序，接著選出正中間的數值來作為那個pixel result image的值。

$ ./hw8 median 3 Gaussian\_10.bmp  
圖片檔名是Median\_3\_Gaussian\_10.bmp  


$ ./hw8 median 3 Gaussian\_30.bmp  
圖片檔名是Median\_3\_Gaussian\_30.bmp  


$ ./hw8 median 3 SandP\_0.05.bmp  
圖片檔名是Median\_3\_SandP\_0.05.bmp  


$ ./hw8 median 3 SandP\_0.1.bmp  
圖片檔名是Median\_3\_SandP\_0.1.bmp  


$ ./hw8 median 5 Gaussian\_10.bmp  
圖片檔名是Median\_5\_Gaussian\_10.bmp  


$ ./hw8 median 5 Gaussian\_30.bmp  
圖片檔名是Median\_5\_Gaussian\_30.bmp  


$ ./hw8 median 5 SandP\_0.05.bmp  
圖片檔名是Median\_5\_SandP\_0.05.bmp  


$ ./hw8 median 5 SandP\_0.1.bmp  
圖片檔名是Median\_5\_SandP\_0.1.bmp  


Part 5 Run Opening-Closing and Closing-Opening on all noisy image

我直接使用c++讀入bmp file lena.bmp，呼叫Read\_image來將header 1078個字元直接寫入到output，並將image的數值0~255存到unsigned char array BBMap中。接著根據輸入的function name來決定先呼叫Opening還是Closing，接著呼叫Writeback把OMap寫回BBMap以便當作下一個function的input，最後再呼叫另一個function來算出result image，並存到unsigned char array OMap中，輸出OMap。

在Opening跟Closing function中，我直接使用之前作業寫的opening and closing，並把它們寫成function，輸入跟輸出分別是BBMap and OMap。

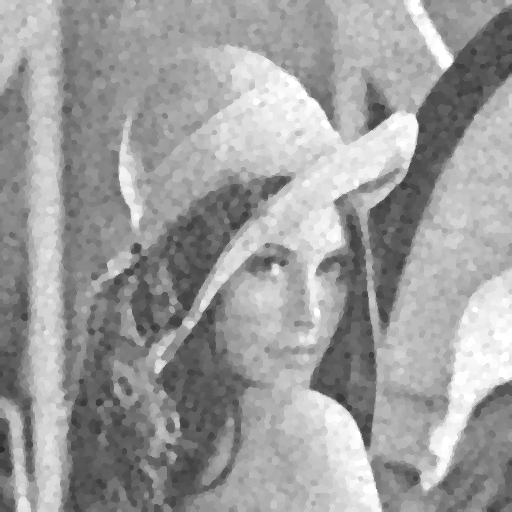
$ ./hw8 open\_clos Gaussian\_10.bmp  
圖片檔名是Open\_Clos\_Gaussian\_10.bmp  


$ ./hw8 open\_clos Gaussian\_30.bmp  
圖片檔名是Open\_Clos\_Gaussian\_30.bmp  


$ ./hw8 open\_clos SandP\_0.05.bmp  
圖片檔名是Open\_Clos\_SandP\_0.05.bmp  


$ ./hw8 open\_clos SandP\_0.1.bmp  
圖片檔名是Open\_Clos\_SandP\_0.1.bmp  


$ ./hw8 clos\_open Gaussian\_10.bmp  
圖片檔名是Clos\_Open\_Gaussian\_10.bmp  


$ ./hw8 clos\_open Gaussian\_30.bmp  
圖片檔名是Clos\_Open\_Gaussian\_30.bmp  


$ ./hw8 clos\_open SandP\_0.05.bmp  
圖片檔名是Clos\_Open\_SandP\_0.05.bmp  


$ ./hw8 clos\_open SandP\_0.1.bmp  
圖片檔名是Clos\_Open\_SandP\_0.1.bmp  


🡪以上的source code、執行檔、result image皆附在壓縮檔中。