Homework 2

姓名：楊哲旻
學號：413511003
學系：電機工程學系博士一年級
日期：2024.11.07

1. Low-luminosity Enhancement (20%). Using C++ or C, improve the luminosity of the given input image. **[Code Demo Video]**

常見的亮度調整方法有「線性增強法(Linear Brightness Enhancement)」、「對比度增強與亮度增強結合(Brightness and Contrast Adjustment)」與「伽馬校正(Gamma Correction)[1]」等，以下如表 1 所整理。

表 1、各亮度增強的方法之比較整理表

方法名稱	數學公式	優點	缺點
線性增強法	$V_{New} = V_{Original} + B$	簡單易實現，計算快速	可能導致過曝或細節流失，對比度提高有限
對比與亮度調整結合	$V_{New} = V_{Original} \times C + B$	可以同時調整亮度和對比度，靈活控制	增強效果受限，需調整參數來達到預期效果
Gamma 校正[1]	$V_{New} = 255 \times (V_{Original} / 255)^{\gamma}$	模擬人眼對亮度的感知 避免過曝或過暗 適合細緻調整	γ 值選擇不當會導致影像過亮或過暗，計算稍複雜

其中 V_{New} 為調整後的像素值；$V_{Original}$ 為原始的像素值；B 為亮度偏移量用來調整影像亮度，正值增亮度，負值減亮度；C 為對比度因子，當 C>1 時，對比度增加；當 0<C<1 時，對比度減少。γ 值用於調整影像亮度。當 γ<1 時，影像變亮；當 γ>1 時，影像變暗。以上比較方法之影像如圖 1 所示。

原圖

線性增強法(B=50)

對比與亮度調整結合(C=1.2, B=30)

Gamma 校正(γ = 0.5)

Gamma 校正(γ = 0.3)

圖 1、各亮度增強的方法之影像比較

備註：另外還有直方圖均衡化 (Histogram Equalization)[2]方法，它主要目的是增強影像的對比度，而不是直接調整影像的亮度，它與亮度的關係是間接的，其他也有如自適應直方圖均衡化 (Adaptive Histogram Equalization, AHE) [3]，對比受限自適應直方圖均衡化 (Contrast Limited Adaptive Histogram Equalization, CLAHE) [4] 等方法。因此，這裡我們選擇「Gamma 校正」作為我們這題亮度提高的方法，而其他兩種方法 (線性增強法，對比與亮度調整結合) 程式碼我放在「Other」資料夾。

從輸入文件中讀取一行與每一列像素數據，並將像素值除以 255 標準化到 [0, 1] 範圍，接著使用 Gamma 校正公式後重新映射到 [0, 255] 並轉回 uint8_t 類型，將處理後的像素數據寫入輸出文件，如圖 2 所示。

```
55      // 逐行處理影像像素資料
56      for (int i = 0; i < header.height; i++) {
57          fread(pixelData, 1, rowPadded, input); // 從輸入檔案讀取每行像素資料
58
59          for (int j = 0; j < header.width * 3; j++) { // 對每個像素的 R、G、B 分量進行處理
60              float normalized = pixelData[j] / 255.0; // 將像素值標準化到 [0,1] 範圍
61              pixelData[j] = (uint8_t)(255 * pow(normalized, gamma)); // 根據 gamma 值調整亮度，並重新映射到 [0,255]
62          }
63
64          fwrite(pixelData, 1, rowPadded, output); // 將處理後的像素資料寫入輸出檔案
65      }
```

圖 2、 Gamma 校正之程式碼

參考文獻

[1] Wikipedia Understanding Gamma Correction https://www.cambridgeincolour.com/tutorials/gamma-correction.htm

[2] Wikipedia Histogram Equalization https://en.wikipedia.org/wiki/Histogram_equalization

[3] Wikipedia Adaptive Histogram Equalization https://en.wikipedia.org/wiki/Adaptive_histogram_equalization

[4] K. Zuiderveld. “Contrast limited adaptive histogram equalization,” Graphics gems IV. Academic Press Professional, Inc., USA, 474–485, 1994.

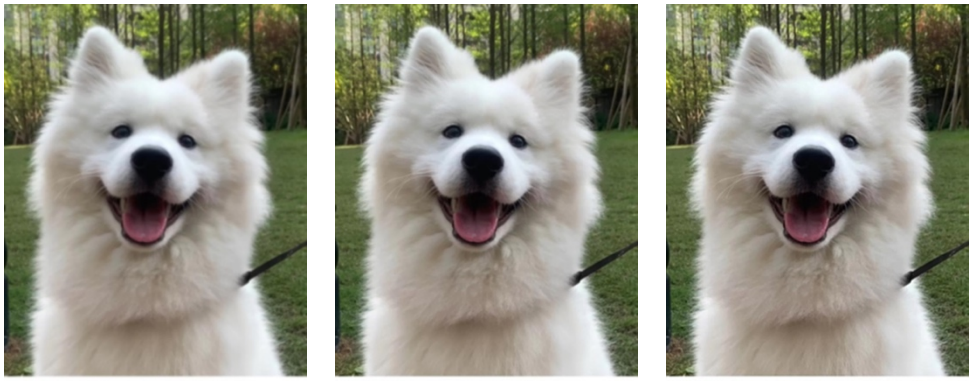
[5] Homework 2-1 Code Demo Video https://youtu.be/ku9iqMumiFs

2. Sharpness Enhancement (30%). Using C++ or C, perform sharpness enhancement on the given image. You should output images with 2 different degrees of modification. **[Code Demo Video]**

銳化增強 (Sharpness Enhancement) 是一種影像處理技術，通過強化影像中像素值變化較大的區域（如邊緣）來達到效果，銳化增強可以讓影像中對比度低的邊緣更加突出，從而使整體影像看起來更清晰，常見的有拉普拉斯濾波器、Sobel 銳化、非銳化遮罩(Unsharp Masking)等。而**拉普拉斯濾波器**[1] 是用於影像銳化的常見濾波器之一。它是一種高通濾波器，專注於檢測影像中的快速變化（如邊緣）。拉普拉斯濾波器的作用是強化高頻成分，從而突出影像細節。拉普拉斯濾波器的數學表達式如下：

$$L(x,y) = I(x,y) \times \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

其中 $I(x,y)$ 表示原始影像中的像素值，$L(x,y)$ 是濾波後的影像中像素的結果。銳化的基本過程可以表示為：$I_{sharp}(x,y) = I(x,y) + \alpha \cdot L(x,y)$，α 是銳化強度的參數，用於控制拉普拉斯濾波結果對原始影像的影響程度。圖 4 為影像使用拉普拉斯濾波器的銳化之比較。



原圖　　銳化 α=1　　銳化 α=2

圖 4、影像使用拉普拉斯濾波器的銳化之比較

在我程式碼如圖 5 中，使用雙重迴圈將拉普拉斯濾波器應用於每個像素及其周圍像素，以計算 sum 來獲得濾波後的高頻成分。laplacianKernel 變數儲存 3x3 的濾波器的值，迴圈將 sum 與原始像素值 originalVal 相加，並乘以銳化強度 strength。然後使用條件運算符將像素值限制在 0 到 255 之間，避免超出範圍導致色彩失真。程式碼中設 strength 為 1.0 和 2.0，用來產生兩種不同程度的銳化效果，調整此參數可以控制影像的銳化程度。

```
27  // 銳化濾波器（拉普拉斯濾波器），用於強化圖像邊緣
28  int laplacianKernel[3][3] = {
29      { 0, -1,  0 },
30      {-1,  4, -1 },
31      { 0, -1,  0 }
32  };
33
34  // 應用拉普拉斯濾波器進行影像銳化
35  // imageData: 輸入圖像數據
36  // outputData: 銳化後的輸出圖像數據
37  // width: 圖像寬度
38  // height: 圖像高度
39  // rowPadded: 每行的實際位元組數（包含填充）
40  // strength: 銳化強度
41  void applySharpening(uint8_t* imageData, uint8_t* outputData, int width, int height, int rowPadded, float strength) {
42      for (int y = 1; y < height - 1; y++) { // 跳過圖像邊界
43          for (int x = 1; x < width - 1; x++) {
44              for (int c = 0; c < 3; c++) { // 處理每個顏色通道 (B, G, R)
45                  int sum = 0;
46                  // 使用拉普拉斯核計算當前像素的銳化值
47                  for (int ky = -1; ky <= 1; ky++) {
48                      for (int kx = -1; kx <= 1; kx++) {
49                          int pixelVal = imageData[(y + ky) * rowPadded + (x + kx) * 3 + c];
50                          sum += pixelVal * laplacianKernel[ky + 1][kx + 1];
51                      }
52                  }
53                  // 原始像素值
54                  int originalVal = imageData[y * rowPadded + x * 3 + c];
55                  // 銳化後的像素值，使用指定的銳化強度
56                  int newVal = (int)(originalVal + strength * sum);
57                  // 確保像素值在 [0, 255] 範圍內
58                  outputData[y * rowPadded + x * 3 + c] = (newVal > 255) ? 255 : (newVal < 0) ? 0 : newVal;
59              }
60          }
61      }
62  }
```

圖 5、使用拉普拉斯濾波器銳化的程式碼

參考文獻

[1] Wikipedia Edge detection https://en.wikipedia.org/wiki/Edge_detection#Laplacian_of_Gaussian

[2] Homework 2-2 Code Demo Video https://youtu.be/zvY4ryS7Qhc

3. Denoise (30% + 20%). Using C++ or C, remove the noise in the given two input images. You should output images with 2 different degrees of modification. You must write a SSIM (Structural Similarity Index) code to evaluate the results after denoising. The formula for SSIM is in the next page (20% of the grade will be based on your SSIM score for image2) File names with a suffix _1 will be the image we use for evaluation: **[Code Demo Video]**

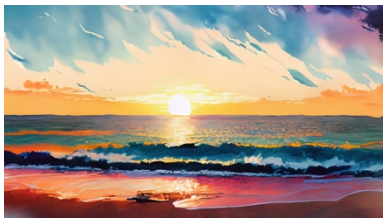
中值濾波器(Median Filter)[1]是一種非線性濾波方法，用於去除影像中的噪聲，特別是椒鹽噪聲。其基本原理是在影像中選擇一個大小窗口，將窗口內所有像素的值排序，並用中間值替換中心像素的值，這樣可以有效地去除尖銳型噪聲，並保持邊緣細節，如下公式。

$$\hat{I}(x,y) = median\{I(x+i, y+i)\}, \quad (i,j) \in W$$

其中$\hat{I}(x,y)$是濾波後的像素值，$I(x+i,y+i)$是在窗口W內的像素值。W是一個預定義的窗口，可為 3x3 或 5x5 等。

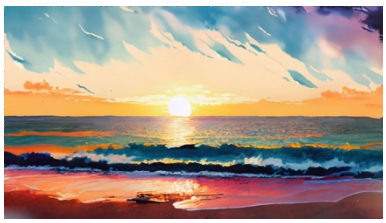
＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

雙邊濾波器(Bilateral Filter)[2]是一種非線性、邊緣保護的濾波方法，它不僅考慮空間上的鄰域關係，還考慮像素亮度的相似性。這樣做的目的是在去除噪聲的同時保留影像邊緣。雙邊濾波器使用兩個權重來計算濾波後的像素值：



原圖

加入雜訊的圖



中值濾波器去雜訊的圖

雙邊濾波器去雜訊的圖

圖 6、 input3.bmp 去雜訊影像比較

- 空間權重 (σ_s)：根據像素距離計算，決定距離中心像素越遠的像素權重越小。
- 強度權重 (σ_r)：根據像素值的差異計算，決定顏色或亮度差異較大的像素權重越小。

雙邊濾波器公式如下：

$$\hat{I}(x,y) = \frac{1}{W_P} \sum_{(i,j)\in W} I(i,j) \cdot \exp\left(-\frac{\|(i,j)-(x,y)\|^2}{2\sigma_s^2}\right) \cdot \exp\left(-\frac{\|(i,j)-(x,y)\|^2}{2\sigma_s^2}\right)$$

從作業的「input3.bmp」為圖 6，發現此圖像的雜訊應該為椒鹽噪聲，因為使用中值濾波器有良好的去噪效果，而在雙邊濾波器下則效果很差。而在作業的「input4.bmp」為圖 7，使用中值濾波器無法完整去噪，而在雙邊濾波器則效果一般。本作業我中值濾波器都使用 3x3 窗口大小，而雙邊濾波器則 7x7 窗口大小，在σ_s 小 σ_r 大時，適合保護邊緣細節，但同時進行適度平滑。在σ_s 大σ_r 小時，影像變得平滑，其細節和邊緣保護較少，噪聲去除能力強，本作業σ_s =45 與σ_r =55。

SSIM (結構相似性指數) 是用來衡量兩張影像之間的相似度。SSIM 越接近 1 表示兩張影像越相似，而 SSIM 越小則表示兩張影像的差異越大。以下為用課堂中的公式，並使用 Grayscale 與窗口大小為 11x11 對兩張影像與兩種方法進行評估，如表 2。

兩個方法程式碼在 Homework_2_3.c 分別為第 73-98 行與第 108-139 行；而 SSIM 程式碼在 SSIM.c。

原圖

加入雜訊的圖

中值濾波器去雜訊的圖

雙邊濾波器去雜訊的圖

圖 7、 input4.bmp 去雜訊影像比較

表 2、SSIM 評估結果

	input3.bmp	input4.bmp
中值濾波器	0.9416	0.6140
雙邊濾波器	0.1278	0.7128

參考文獻

[1] Wikipedia Median Filter https://en.wikipedia.org/wiki/Median_filter

[2] Wikipedia Bilateral Filter https://en.wikipedia.org/wiki/Bilateral_filter

[3] Homework 2-3 Code Demo Video https://youtu.be/AL1e-3WbNRY