

Computer Vision HW2

612415013 電機碩一 蕭宥羽

一、 程式碼及方法

```
import numpy as np
import cv2
import matplotlib.pyplot as plt
```

Import python libraries

```
def RGB2GRAY(image):
    return np.dot(image[...,:3], [0.21, 0.72, 0.07]).astype(np.uint8)
```

將原圖片轉為灰階圖片

```
def mean_filter(image,kernel_size):
    height, width = image.shape[:2]
    matrix=np.zeros((height-kernel_size+1,width-kernel_size+1))
    for i in range(height-kernel_size+1):
        for j in range(width-kernel_size+1):
            total=0
            for a in range(kernel_size):
                for b in range(kernel_size):
                    total+=image[i+a,j+b]
            matrix[i,j]=total/(kernel_size**2)
    return matrix.astype(np.uint8)
```

做 mean_filter 的動作，將 kernel 中的直先相加再除以 kernel 的大小

```

def quicksort(data, left, right):
    if left >= right :
        return
    i = left
    j = right
    key = data[left]
    while i != j:
        while data[j] > key and i < j:
            j -= 1
        while data[i] <= key and i < j:
            i += 1
        if i < j:
            data[i], data[j] = data[j], data[i]
    data[left] = data[i]
    data[i] = key
    quicksort(data, left, i-1)
    quicksort(data, i+1, right)

```

實現快速排序的算法，將第一個元素設為基準值，然後派兩個代理人分別從資料的兩邊開始往中間找，一個人從前面往後搜尋比基準值大的值，一個人從後面往前搜尋比基準值小的值。找到後值互換，一直下去直到兩人相遇。

這樣可以以基準點為分界，左邊是比基準值小的，右邊是大的。在用地回的方式，就可以做出排序

```

def median(sorted_arr):
    quicksort(sorted_arr, 0, len(sorted_arr)-1)
    n=len(sorted_arr)
    if n%2==1:
        return sorted_arr[n//2]
    else:
        middle1=sorted_arr[n//2]
        middle2=sorted_arr[n//2-1]
        return (middle1+middle2)/2

```

得到中位數，如果 array 是基數，就可以直接取中位數；如果是偶數則將中間兩數相加後除 2

```
def median_filter(image,kernel_size):
    height, width = image.shape[:2]
    matrix=np.zeros((height-kernel_size+1,width-kernel_size+1))
    for i in range(height-kernel_size+1):
        for j in range(width-kernel_size+1):
            arr=[]
            for a in range(kernel_size):
                for b in range(kernel_size):
                    arr.append(image[i+a,j+b])
            matrix[i,j]=median(arr)
    return matrix.astype(np.uint8)
```

做 median filter 的動作，取 kernel 中的中位數，作為新 matrix 的元素

```
def his(image,title,path):
    hist, bins = np.histogram(image.flatten(), bins=256, range=[0,256])
    plt.title(title)
    plt.bar(bins[:-1], hist, width=1.0)
    plt.savefig(path)
    plt.show()
```

繪製直方圖

```
noise_img=cv2.imread("test_img/noise_image.png")
cv2.imshow('noise_image', noise_img)
gray_img = RGB2GRAY(noise_img)
his(gray_img,"gray_img","result_img/noise_image_his.png")
cv2.waitKey(0)
cv2.destroyAllWindows()
```

對 noise_image 取灰階圖
並且繪製直方圖

```

mean_img=mean_filter(gray_img,3)
cv2.imshow('mean_image', mean_img)
cv2.imwrite("result_img/output1.png", mean_img)
his(mean_img,"mean_img","result_img/output1_his.png")
cv2.waitKey(0)
cv2.destroyAllWindows()




median_img=median_filter(gray_img,3)
cv2.imshow('median_image', median_img)
cv2.imwrite("result_img/output2.png", median_img)
his(median_img,"median_img","result_img/output2_his.png")
cv2.waitKey(0)
cv2.destroyAllWindows()

```

做 mean filter 跟 median filter 的動作，並將其顯示到畫面中及儲存
接著繪製直方圖

二、結果

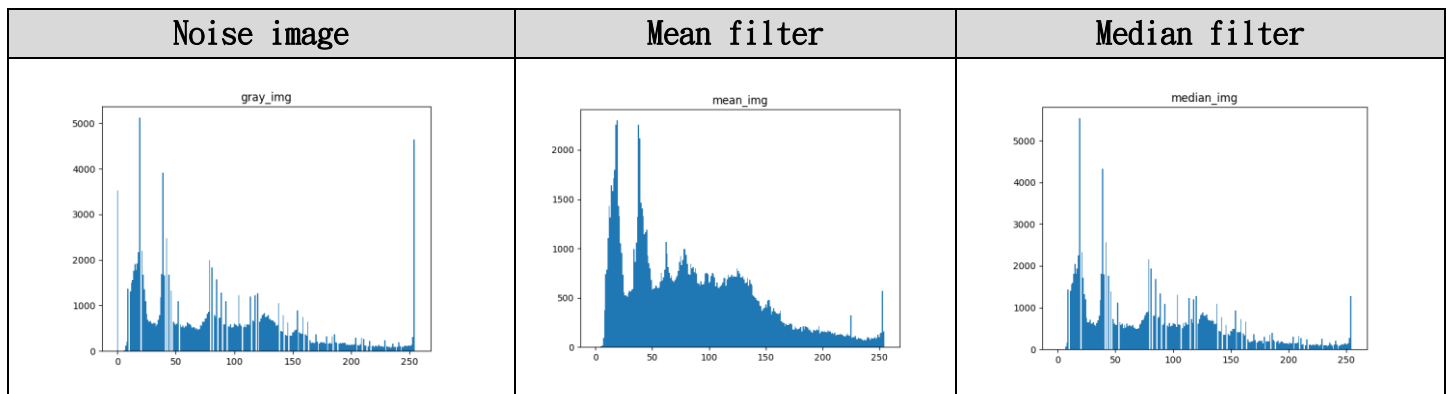
1. 圖片

Noise image	Mean filter	Median filter
		

Mean filter:還是會有雜訊點，但是相較於 Noise image 雜訊被模糊掉了，因為 mean filter 是對整個區域做平均處理

Median filter:效果很好，雜訊通常會是脈衝值，用取中間值的方式可以很好過濾掉雜訊。

2. 直方圖



Mean filter 圖像的值方圖扯以看出分布的更加平均，而且極端值有所減少。
但可能是因為平均的關係所以圖像會較模糊

Median filter 圖像的跟 noise image 的值方圖更加地接近，但是極端值都被
濾掉了，所以可以有效減少噪音，且圖像依舊清楚