機器視覺作業報告

Homework #1

姓名: 吳秉宸
學號: 00957202
日期: 2022/04/13

方法

- 想法，演算步驟。

  這次的報告最一開始是構思要做哪些 image operation 的組合達成老師所要求的項目，如下：

  1. 第一張圖片採用
     Gauss + different color coordinate + tensorflow + 雪山 youtube 影片
  2. 第二張圖片採用
     Gray-level mapping + 老師的測試影片
  3. 第三張圖片採用
     Histogram equalization + 老師的測試影片
  4. 第四張圖片採用
     High-pass + 老師的測試影片

  每張圖片都 define 一個獨立的 function。

  其中，第一張圖是先轉換到 HSV 的座標系中，因為其很適合調整顏色飽和度，進行顏色飽和度的提升，再轉回 BGR 座標系，用 opencv 生成 Gauss kernel，並將其 kernal 帶入 tensorflow 做卷積運算。

  第二張圖是先將圖片轉到灰階圖再處理，自己實作出 Gray-level mapping (gamma correction of gamma = 0.5)，先建 Table 再進行對應，此處若將 Table 建置在 function 外可以減少每次 for loop 都要建 Table 的運算與時間 (因為 mapping 的 function 是固定的) ，但為了程式的易讀性寫在裡面。

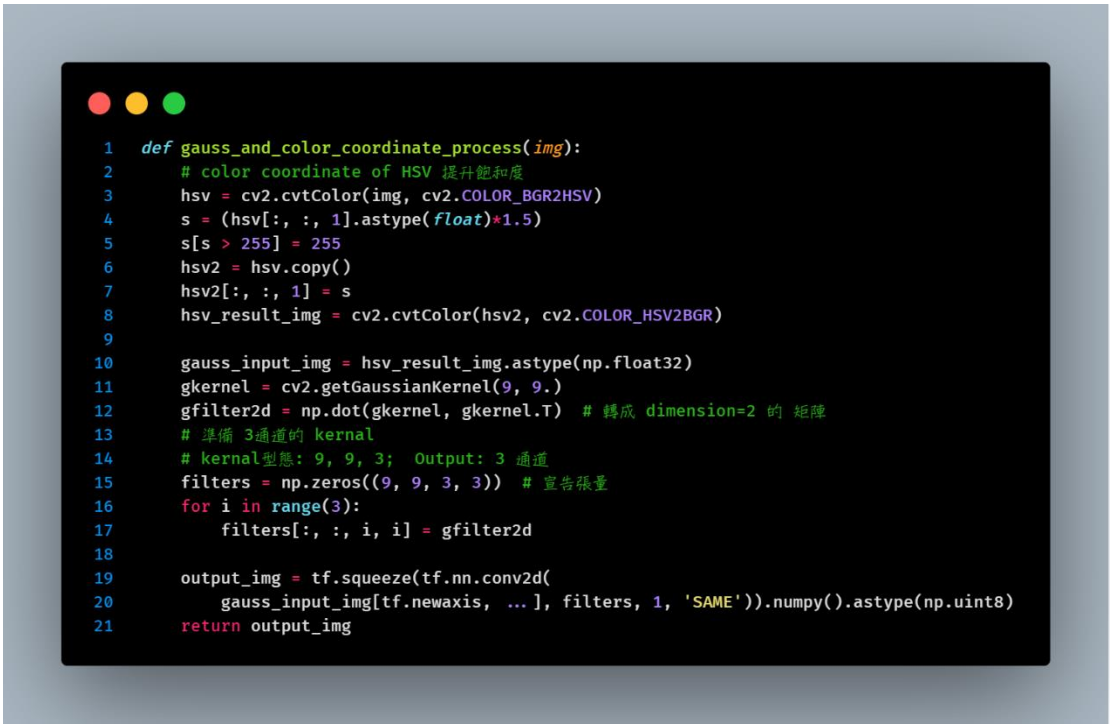
  第三張圖也先將圖片轉到灰階圖再處理，而原本是自己實作出 histogram equalization，雖然有成功，不過發現程式真的跑太慢了，因為其要依據 input 的 img 才能決定 mapping 的曲線，所以最後去 Opencv 的網站上看到了 cv2.equalizeHist() function，並使用之。

  第四張圖是先將圖片轉到灰階圖再處理，採用的是 high_pass 的 img operation 來處理，先用傅立葉轉換轉換到適合操縱細節的 Fourier domain，並做上下左右的區塊顛倒，以中心點為圓心，將 Frequency 為 50 以下的去掉，再做逆傅立葉轉換回原本 domain 呈現。

  最後再將圖片都 copy 到一張大圖中，做 VideoWriter 得結果的影片 。

- 重要程式片段說明。

1. Gauss + different color coordinate + tensorflow + 雪山 youtube 影片



```python
def gauss_and_color_coordinate_process(img):
    # color coordinate of HSV 提升飽和度
    hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
    s = (hsv[:, :, 1].astype(float)*1.5)
    s[s > 255] = 255
    hsv2 = hsv.copy()
    hsv2[:, :, 1] = s
    hsv_result_img = cv2.cvtColor(hsv2, cv2.COLOR_HSV2BGR)

    gauss_input_img = hsv_result_img.astype(np.float32)
    gkernel = cv2.getGaussianKernel(9, 9.)
    gfilter2d = np.dot(gkernel, gkernel.T)  # 轉成 dimension=2 的 矩陣
    # 準備 3通道的 kernal
    # kernal型態: 9, 9, 3;  Output: 3 通道
    filters = np.zeros((9, 9, 3, 3))  # 宣告張量
    for i in range(3):
        filters[:, :, i, i] = gfilter2d

    output_img = tf.squeeze(tf.nn.conv2d(
        gauss_input_img[tf.newaxis, ...], filters, 1, 'SAME')).numpy().astype(np.uint8)
    return output_img
```

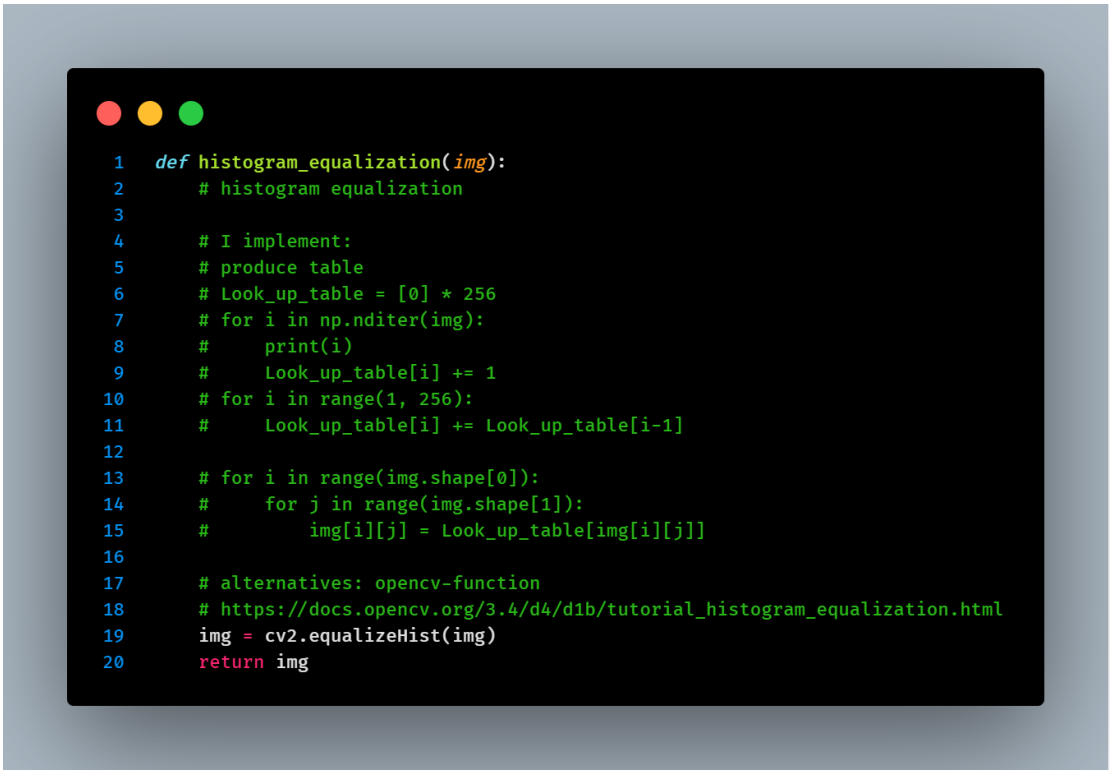
先做色彩座標系的轉換，轉換到 HSV，提升飽和度，再轉回 BGR 座標系，利用 OpenCV 得到高斯 kernal，利用 tensorflow 把該 kernal 與原圖片進行卷積運算。

2. Gray_level_mapping function

```python
def gray_level_mapping(img):
    # Gamma correction (gamma is 0.5)

    Look_up_table = [(i / 255.0) ** 0.5 * 255 for i in range(0, 256)]
    # https://pyimagesearch.com/2015/10/05/opencv-gamma-correction/ 看別人如何正規化

    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
            img[i][j] = Look_up_table[img[i][j]]

    return img
```

自己實作出 gray_level_mapping 的 function，因為上課在講解常數 c 的時候沒有很懂，所以上網看別人如何正規化。

3. Histogram_equalization



```python
def histogram_equalization(img):
    # histogram equalization

    # I implement:
    # produce table
    # Look_up_table = [0] * 256
    # for i in np.nditer(img):
    #     print(i)
    #     Look_up_table[i] += 1
    # for i in range(1, 256):
    #     Look_up_table[i] += Look_up_table[i-1]

    # for i in range(img.shape[0]):
    #     for j in range(img.shape[1]):
    #         img[i][j] = Look_up_table[img[i][j]]

    # alternatives: opencv-function
    # https://docs.opencv.org/3.4/d4/d1b/tutorial_histogram_equalization.html
    img = cv2.equalizeHist(img)
    return img
```

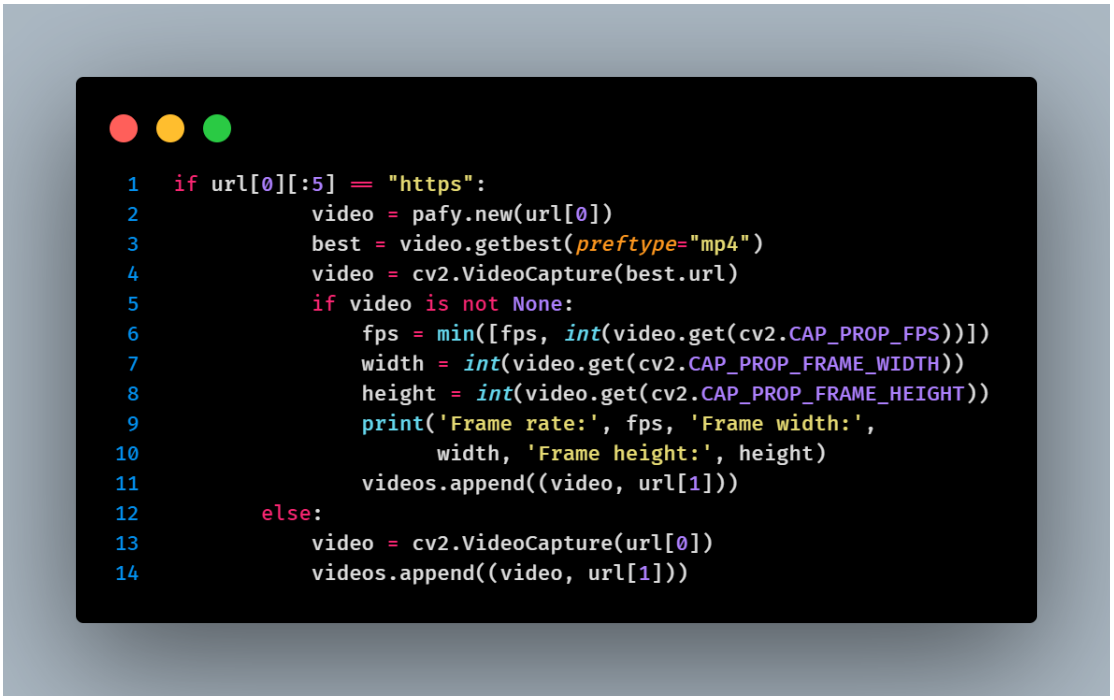
使用 cv2.equalizeHist()，進行直方圖等化，增加對比。

4. High_pass

```python
def high_pass(img, freq):  # 頻率高的通過
    f = np.fft.fft2(img)
    fshift = np.fft.fftshift(f)
    cy, cx = fshift.shape[0]/2, fshift.shape[1]/2
    h = np.arange(fshift.shape[0]).reshape((-1, 1)) - cy
    w = np.arange(fshift.shape[1]).reshape((1, -1)) - cx
    freq = freq**2
    fshift = np.where(h**2+w**2 ≥ freq, fshift, 0)
    ifft_f = np.fft.ifft2(np.fft.fftshift(fshift))
    return ifft_f
```

透過傅立葉轉換到 Fourier domain，並將四角對調，使低頻在中間，將低頻部分小於 50 frequency 的移除，再四角對調做逆傅立葉轉換，回到原本 domain。

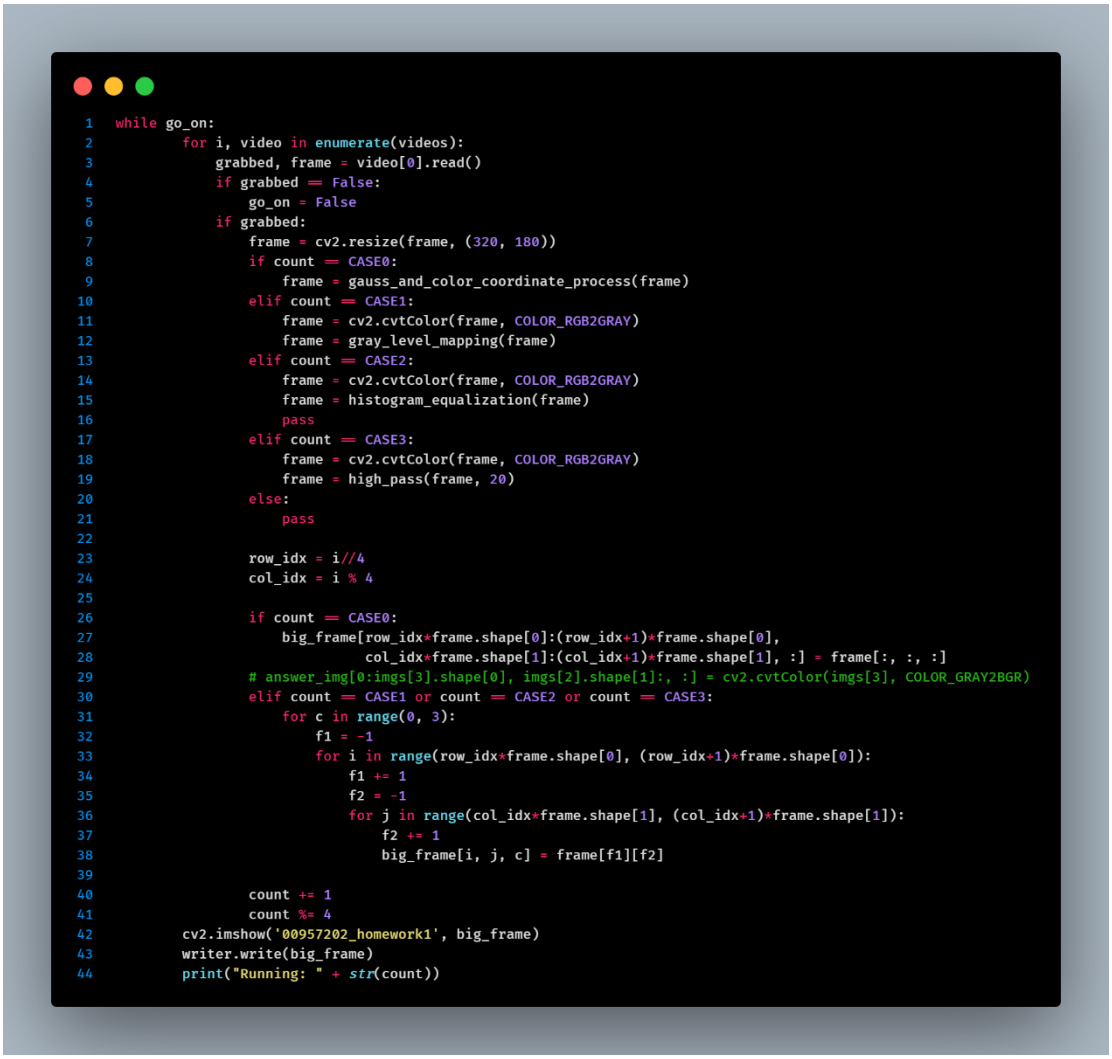
5. Video 的抓取與 youtube 影片的抓取



```
if url[0][:5] == "https":
            video = pafy.new(url[0])
            best = video.getbest(preftype="mp4")
            video = cv2.VideoCapture(best.url)
            if video is not None:
                fps = min([fps, int(video.get(cv2.CAP_PROP_FPS))])
                width = int(video.get(cv2.CAP_PROP_FRAME_WIDTH))
                height = int(video.get(cv2.CAP_PROP_FRAME_HEIGHT))
                print('Frame rate:', fps, 'Frame width:',
                      width, 'Frame height:', height)
                videos.append((video, url[1]))
        else:
            video = cv2.VideoCapture(url[0])
            videos.append((video, url[1]))
```

改寫張老師的檔案，將 url 前面字串與”https”判斷是否為 video 或是 youtube 影片網址。

6. 主要程式碼



```
while go_on:
        for i, video in enumerate(videos):
            grabbed, frame = video[0].read()
            if grabbed == False:
                go_on = False
            if grabbed:
                frame = cv2.resize(frame, (320, 180))
                if count == CASE0:
                    frame = gauss_and_color_coordinate_process(frame)
                elif count == CASE1:
                    frame = cv2.cvtColor(frame, COLOR_RGB2GRAY)
                    frame = gray_level_mapping(frame)
                elif count == CASE2:
                    frame = cv2.cvtColor(frame, COLOR_RGB2GRAY)
                    frame = histogram_equalization(frame)
                    pass
                elif count == CASE3:
                    frame = cv2.cvtColor(frame, COLOR_RGB2GRAY)
                    frame = high_pass(frame, 20)
                else:
                    pass

                row_idx = i//4
                col_idx = i % 4

                if count == CASE0:
                    big_frame[row_idx*frame.shape[0]:(row_idx+1)*frame.shape[0],
                              col_idx*frame.shape[1]:(col_idx+1)*frame.shape[1], :] = frame[:, :, :]
                # answer_img[0:imgs[3].shape[0], imgs[2].shape[1]:, :] = cv2.cvtColor(imgs[3], COLOR_GRAY2BGR)
                elif count == CASE1 or count == CASE2 or count == CASE3:
                    for c in range(0, 3):
                        f1 = -1
                        for i in range(row_idx*frame.shape[0], (row_idx+1)*frame.shape[0]):
                            f1 += 1
                            f2 = -1
                            for j in range(col_idx*frame.shape[1], (col_idx+1)*frame.shape[1]):
                                f2 += 1
                                big_frame[i, j, c] = frame[f1][f2]

                count += 1
                count %= 4
        cv2.imshow('00957202_homework1', big_frame)
        writer.write(big_frame)
        print("Running: " + str(count))
```

主要程式碼即將圖片進行轉換再 input 進去各自處理的 function，做完後再統整成一張大圖輸出，最後的 31 行 for 迴圈是因為採用的是 channel last 結構，無法使用 python 簡潔的語法插入灰階(由 2 通道插入到 3 通道)，所以自己實作 for 迴圈將灰階得的 RGB 三通道都填相同數字。

結果

成果影片由左而右為

1. Gauss + different color coordinate(提升飽和) + tensorflow + 雪山 youtube 影片
2. Gray-level mapping + 老師的測試影片
3. Histogram equalization + 老師的測試影片
4. High-pass + 老師的測試影片

YouTube 成果連結:
https://www.youtube.com/watch?v=p40P__Xq9cs
Github 程式連結:
https://github.com/evan20010126/code/tree/main/python/Machine_vision/Homework1
<裡面的 homework_gogogo.py>

結論

在這次的作業中第一次碰到影像處理相關的問題，雖然處理的流程一開始就很清楚，而自己從頭到尾寫了一個檔案，不過在最後的排版上出了很多問題，所以最後改用老師的講義來改寫。
另一個問題是在資料結構的不熟悉，因為 opencv 灰階圖為 2 通道，原本查了很多資料看要怎麼寫成三通道的，網路上有看到可以用轉座標系的方式轉到 BGR，不過應該是因為我的 python 版本關係，所以這個方法會報錯，於是自己寫了一個簡易的複製陣列元素的 for 迴圈。

覺得再這次的作業中對影像的資料結構與處理有更進一步的認識，希望以後也可以繼續努力！

參考文獻

https://www.w3schools.com/python/numpy/trypython.asp?filename=demo_numpy_array_iterating4
https://docs.opencv.org/3.4/d4/d1b/tutorial_histogram_equalization.html
https://shengyu7697.github.io/python-opencv-gray-to-rgb/
https://docs.opencv.org/3.4/d4/d1b/tutorial_histogram_equalization.html
https://ithelp.ithome.com.tw/articles/10266785
https://stackoverflow.com/questions/68144185/cv2-error-opencv4-5-2-error-5bad-argument-in-function-cvtcolor