|  |
| --- |
| 機器視覺作業報告 |
| HW1 |
|  |
|  |
| 姓名:岳紀伶  學號:01057113  日期:2024/3/25 |
|  |

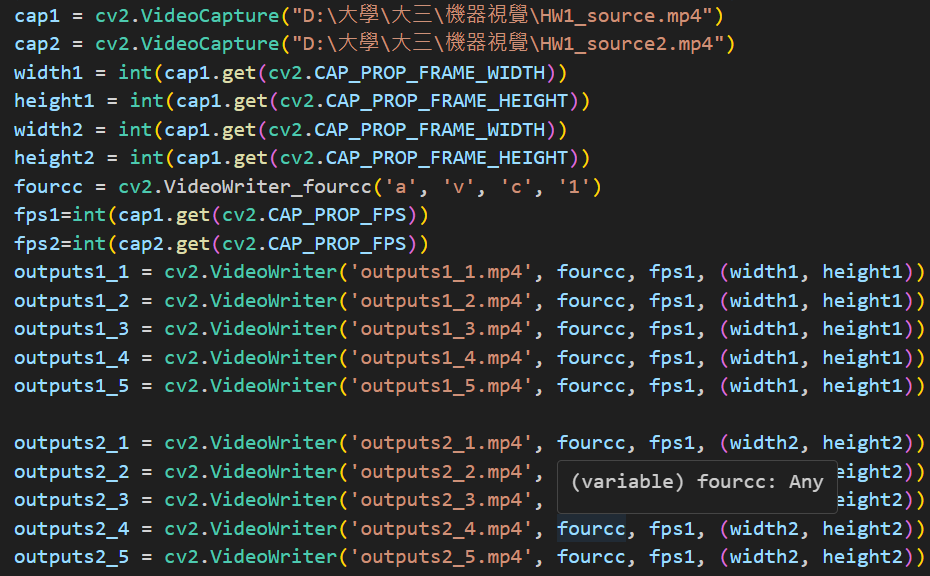
|  |
| --- |
|  |

# 方法

步驟:

這個程式主要可以分成開檔、編輯、輸出、合併影片、加上字幕和聲音五個部分，再開檔的部分，透過OpenCV的VideoCapture讀取檔案，再透過while loop切成一幀一幀處理。編輯影像的部分，我選擇了grayscale、hsv、high pass filter 和 histogram equalization，並透過OpenCV的putText在每一幀左上角加入說明。完成編輯後透過OpenCV的write把每一幀寫入輸出的影像檔中，儲存完全部後release掉所有檔案資源。完成十個影片後(原始影片\*2+四種變化\*2)，將這些影片讀入，改變這些影片的尺寸並加上邊界，完成後再次輸出最後透過老師提供的字幕旁白範例將影片加上字幕、旁白、和背景音樂。

重要程式片段說明:



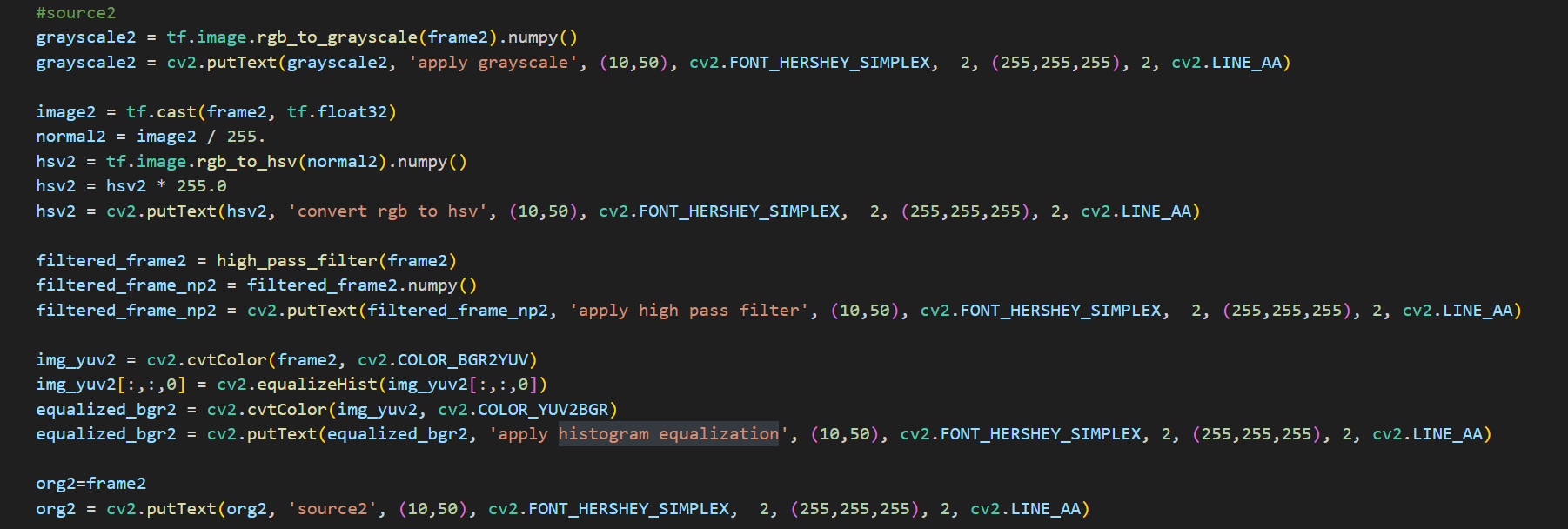
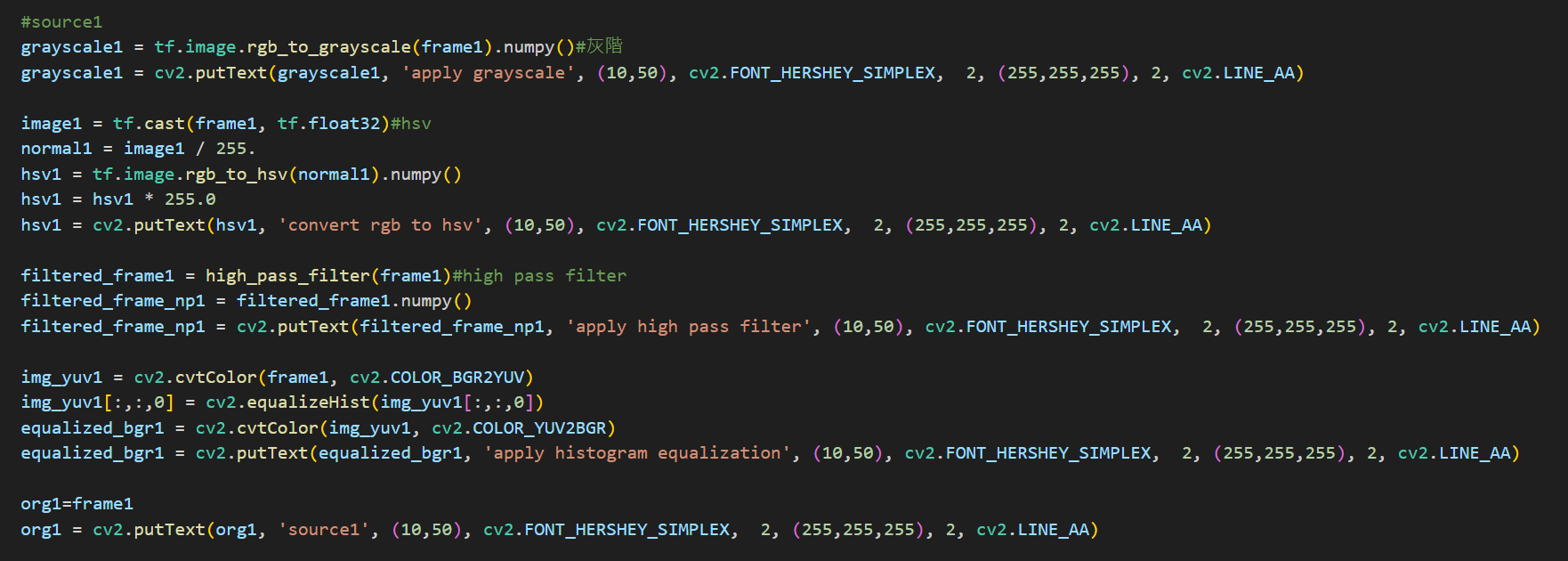
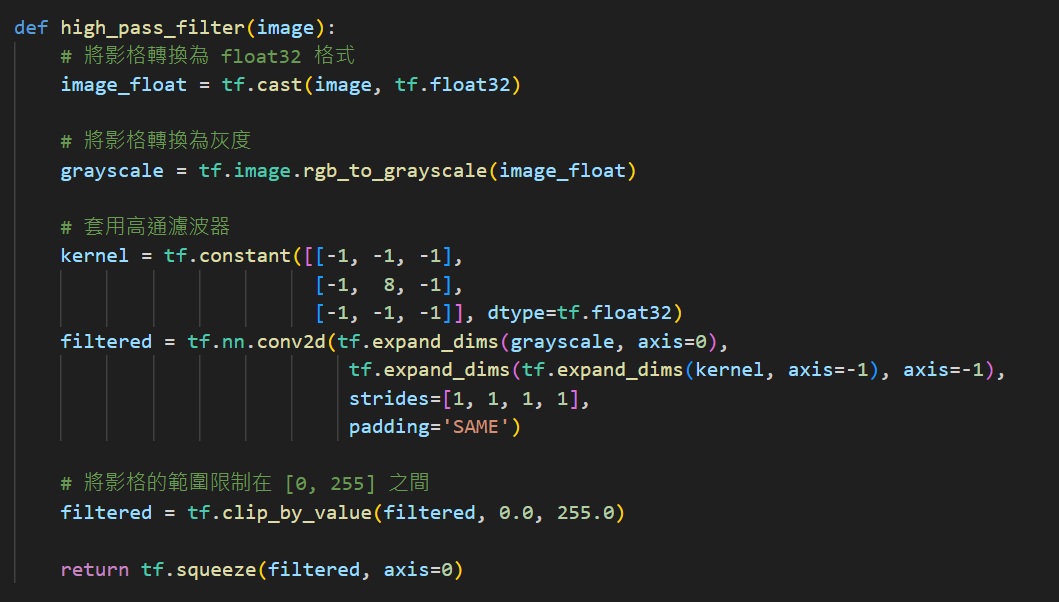
cap1,cap2:透過OpenCV 的VideoCapture讀入資料夾中的兩個原始檔。

width1, width2, height1, height2:影片的長和寬。

fourcc:由於電腦不支援MP4V格式，因此設為avc1。

fpd1,fps2:為影片的幀數，用來設定輸出影片的幀數，使結果長度和原始影片長度相同。

outputs1\_1 ~ outputs2\_5: 透過OpenCV 的VideoWriter儲存編輯後的影片，括號中分別為輸出檔名、輸出格式、幀數和長寬。



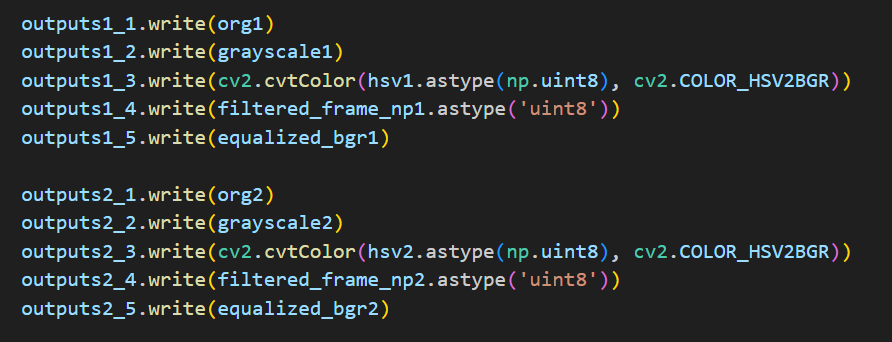
cv2.putText:在編輯完的圖片左上角加入轉換說明。由於OpenCV要對numpy數組進行編輯，因此每個編輯完的影片都需要完成後加上.numpy()。

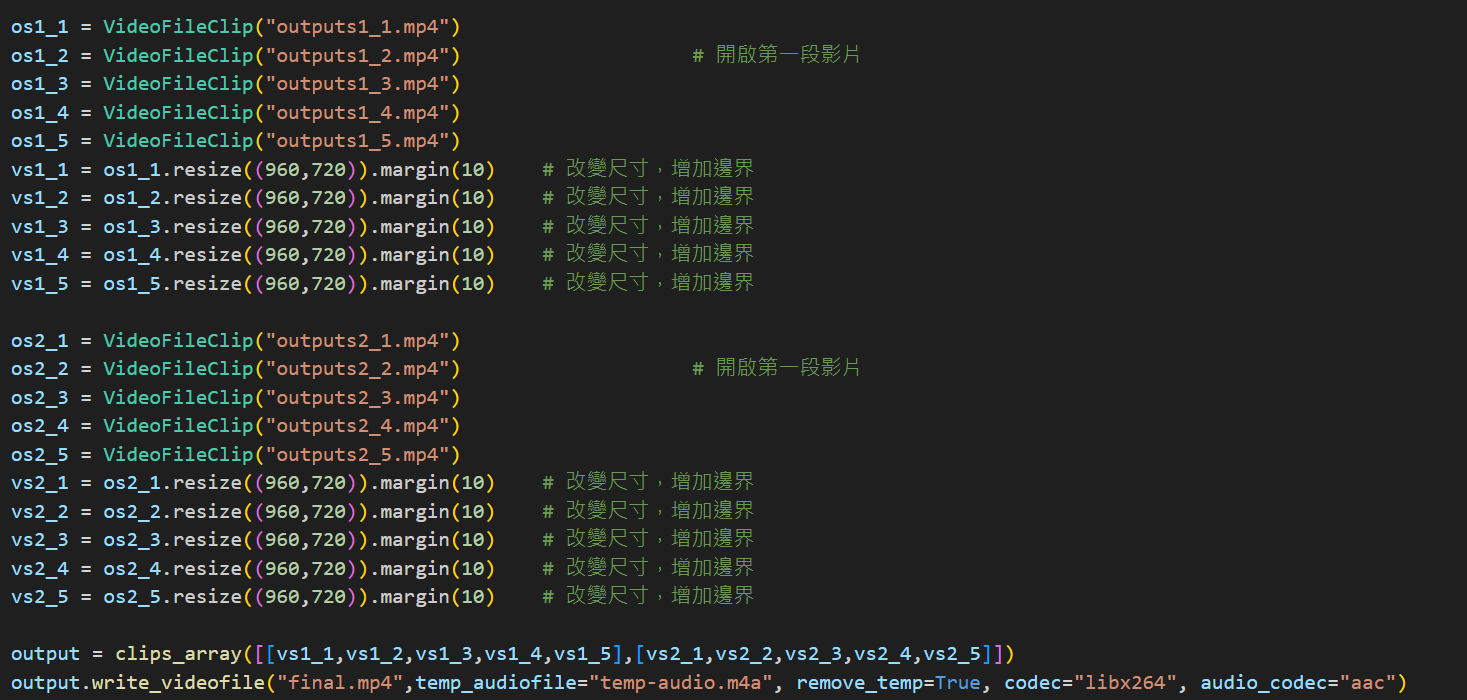
grayscale1, grayscale2:透過tensorflow的rgb\_to\_grayscale將圖片轉為灰階。

hsv1, hsv2:將影片轉為hsv格式。因為圖片讀入為uint8，但rgb\_to\_hsv不接受這個格式，圖片必須在 [0,+1] ，因此圖片要先轉成float32並除以255，使其符合格式，完成編輯後再將它乘以255回到uint8。

filtered\_frame1, filtered\_frame2:透過high pass filter(image)對影片用high pass filter進行轉換。High pass filter為一個3x3的矩陣，中間為8，其餘為-1。

equalized\_bgr1, equalized\_bgr2:對影片使用histogram equalization。由於沒辦法將影片原始檔直接做histogram equalization，因此需要先轉換坐標系。先將影片轉到YUV坐標系，再對Y(亮度)座標進行histogram equalization，最後再把他轉回RGB坐標系。

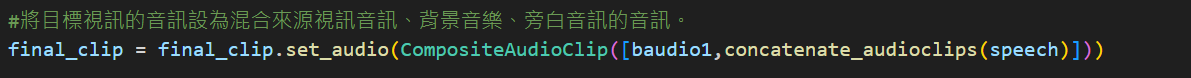
org1,org2:為去除背景音樂的原始影片，因為要加上自己的背景音樂因此需要對原始影片進行處理，透過把每一幀讀出再寫入可以拿掉原始音樂。

把編輯好的每一幀照片寫入outputs1\_1~outpu2s2\_5，每一幀都寫入後輸出影片。由於outputs1\_3和outputs2\_3為hsv坐標系，輸出時轉為rgb坐標系。

把輸出的每個影片載入，改變他們的長寬並加上邊界後，將它們合併成兩列輸出。

output = clips\_array([[vs1\_1,vs1\_2,vs1\_3,vs1\_4,vs1\_5],[vs2\_1,vs2\_2,vs2\_3,vs2\_4,vs2\_5]])

製作一個二維陣列，第一列為「聖稜-雪山的脊樑©」的變化，第二列為「《看見台灣III》預告片」的變化。



加入字幕、旁白和背景音樂的部分幾乎和老師的範例相同，唯有混合所有來源的部分稍作修改，在合併時拿掉了clip的部分。

# 結果

在輸出結果中，grayscale、hsv 和 histogram equalization都還能辨識出原始影片的資訊，但high pass filter在處理「聖稜-雪山的脊樑©」時，整部影片幾乎都處於全黑的狀態，極難進行判斷，「《看見台灣III》預告片」在經過處理後，也只能依稀辨識出輪廓。另外，經過histogram equalization處理後，影片和原始影片相差不大，只能明顯感覺出整體畫面變亮。

影片連結: <https://youtu.be/0RTo_auutsw>

# 結論

在這個作業中，我知道了tensorflow提供那些可以幫助處理影像的API，也學會如何透過編輯每一幀影像來處理一整部影片。在做histogram equalization的過程中，我原本是透過grayscale來進行histogram equalization，但在輸出時想要輸出乘RGB格式，卻發現只有灰色，經過查詢才發現原來製作成grayscale後，會將原本的三通道變為單一通道，所以完成後不管怎麼處理，都會是灰的，由於結果和只做grayscale的結果太過相似，幾乎無法區別，加上老師的範例是彩色的，因此我改為先轉成YUV坐標系再轉回RGB，使其可以保持彩色。

# 參考文獻

<https://python-ecw.com/2023/02/23/videocapture/#toc7>

<https://github.com/bhattbhavesh91/portrait-mode-tensorflow/blob/master/background-blur-notebook.ipynb>

<https://gist.github.com/syphh/943c67ce98d73c5abf4bfc34d4408278>

<https://pillow.readthedocs.io/en/stable/handbook/concepts.html#concept-filters>

<https://colab.research.google.com/github/tensorflow/io/blob/master/docs/tutorials/colorspace.ipynb?authuser=1#scrollTo=kLEdfkkoK27A>

<https://ithelp.ithome.com.tw/m/articles/10275137>

<https://claire-chang.com/2023/01/04/tensorflow%E7%9A%84%E5%9C%96%E5%83%8F%E6%93%8D%E4%BD%9C%E5%8A%9F%E8%83%BD%E7%AD%86%E8%A8%98/>

<https://ithelp.ithome.com.tw/articles/10294436>

<https://www.askpython.com/python-modules/opencv-puttext>

<https://shengyu7697.github.io/python-opencv-gray-to-rgb/>

<https://stackoverflow.com/questions/31998428/opencv-python-equalizehist-colored-image>

chatgpt