**電腦網路導論**

**期末Project Report**

**基於RTSP (Real Time Streaming Protocol)的**

**串流媒體播放器**

組員

B06901031 楊宗賢

B02204048 莊奇凡

B05502090 曾奇鈞

**目錄**

1. **主題介紹**
2. **操作環境**
3. **操作指南**
4. **源碼分析**
5. **實作困難**
6. **未來改進**
7. **主題介紹**

RTSP協定是基於其他串流傳送網路協定上的網路協定，其主要功能在於遠端控制串流影片的播放，目前在市場上已經有許多軟體是基於這個協定所開發的。而選擇這個協定來實作的原因是因為這個協定並不負責傳輸的部分，而是基於RTP來進行傳輸，以此為主題將更有機會去操作底層傳輸，利用其來學習跟思考如何實作此協定去達成遠端控制的串流媒體播放器，並改進其行為。

1. **操作環境**

主要的操作環境是建立在Windows 10的Python 3.6上，所使用的Python Package有：numpy、cv2、tkinter、PIL、enum、mod、Pillow。目前只能夠在Windows上操作，在Linux上因為Package相容性的問題需要再做修改。

1. **操作指南**

因為我們環境是建置在Windows10上，以下的安裝以及皆以Windows10為主。

完整檔案包含:

主要程式

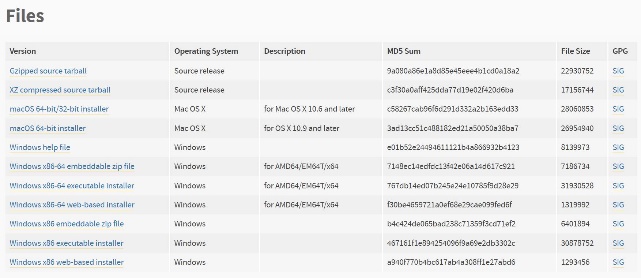
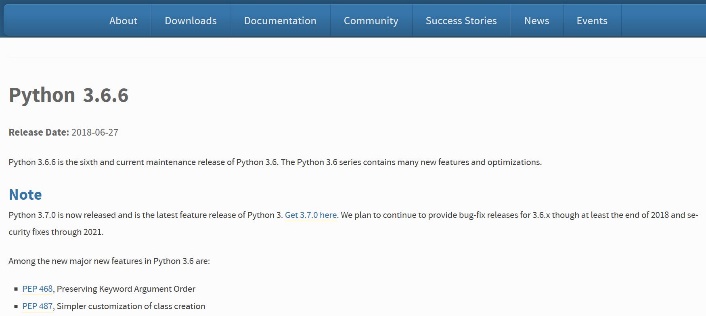
rtppacket.py、s\_client\_UDP.py、s\_server\_UDP.py。

安裝批次檔

b\_install\_opencv\_and\_Pillow.bat、b\_install\_dlib.bat、b\_download\_face-landmarks\_90MB.bat。

* 1. 安裝Python3.6

1. 先至Python的官網，到Download頁面後選擇Python3.6.6 (<https://www.python.org/downloads/release/python-366/>)
2. 接下來會進入右圖的頁面，向下滑到Files部分



1. 根據電腦是64位元或32位元來選擇安裝檔，如果是64位元電腦就選擇Windows x86-64 executable installer
2. 下載完成後就啟動安裝檔，跟著指示做就好了。
   1. 安裝套件(在CMD下輸入)
3. OpenCV(cv2)安裝

py -m pip install opencv-python

1. Pillow安裝

py -m pip install Pillow

1. Dlib安裝

py -m pip install src\dlib-19.8.1-cp36-cp36m-win\_amd64.whl

1. Face\_landMark辨識檔下載

直接點選檔案中的b\_download\_face-landmarks\_90MB.bat

* 1. 程式執行

1. Server端

在cmd上執行py s\_server\_UDP.py，啟動server端

1. Client端

在cmd上執行py s\_client\_UDP.py後會跳出播放器(下圖)，

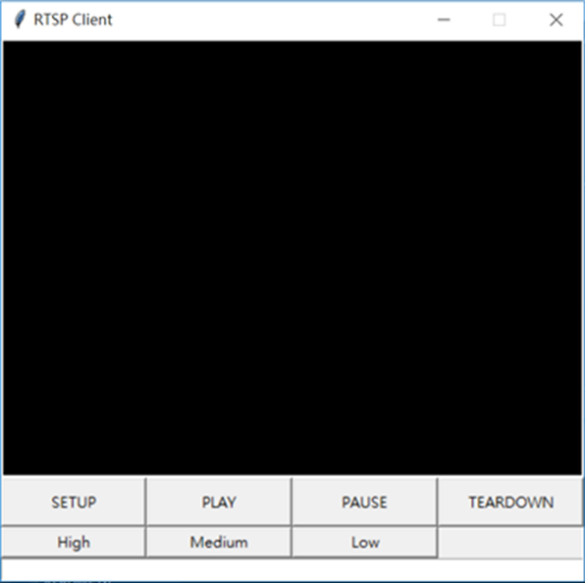
播放器的操作欄中有：

SETUP將啟動Client與Server間的會話，

而PLAY、PAUSE、TEARDOWN對影片串流本身做操作，

High、Medium、Low可以切換影片品質，

下方輸入欄可以根據使用者需求來選擇影片性質。

1. 操作流程
   * + 1. 先按SETUP與Sever建立連線
       2. 再根據所需要的影片性質在輸入欄輸入以下Command
2. Movie or Stream

播放Server端影片檔

1. Camera

啟動Server端電腦前置鏡頭，播放即時影像

1. Face

人臉偵測

1. Menu

功能介紹目錄

* + - 1. 使用PLAY、PAUSE、TEARDOWN操作影片

**肆、 源碼分析**

**s\_server\_UDP.py:**

1. **主要功能**

s\_server\_UDP.py是作為RTP跟RTSP的server端，接收client端的要求，提供傳送影片、Camera即時影像、人臉辨識的功能。

1. **使用的網路協定**

基於UDP的RTP網路協定來實作使用者端能夠控制影片的RTSP協定。

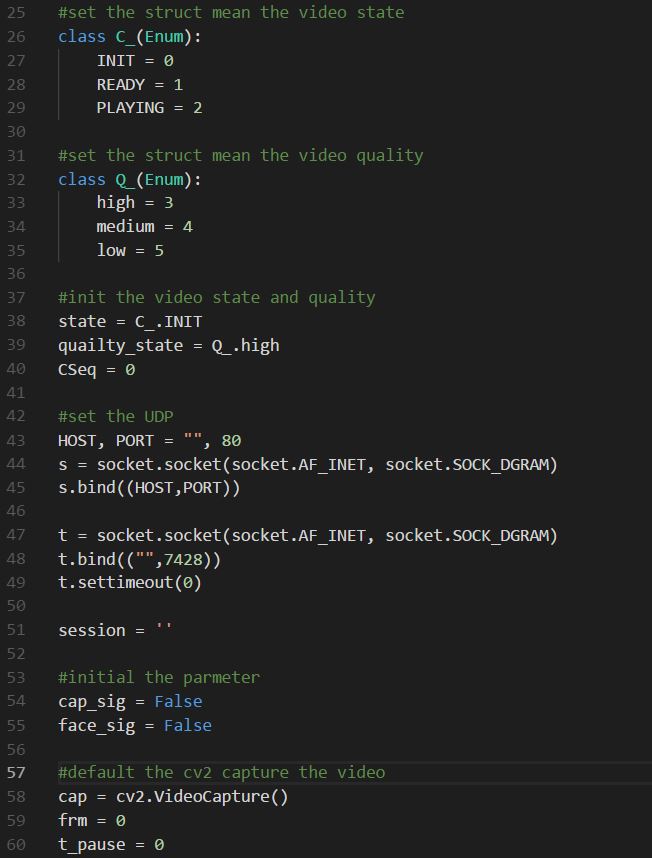
1. **運作流程**

**參數初始化()**

影片操作的State default為INIT

影片品質default為High Quality

建立UDP連線



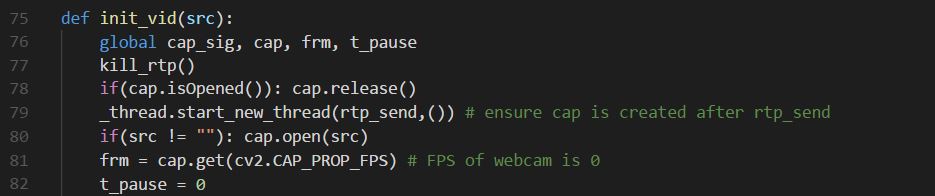
**操作初始化()**

將舊有連線刪除

釋放過去影片擷取參數

新開對傳輸函數的執行緒確認cap已經啟用

如果有影片來源就使用cap開啟影片



**進入無限迴圈()**

等待請求

If 收到setup的請求

在主迴圈中會收到來自Client端的請求並完成連線

將State設成Ready

Else if 收到TEARDOWM

將State設成INIT

Else if 收到 Play

將State設成Play，並開啟執行緒進行傳送

Else if 收到Pause

State還是Play，主要影片暫停還是由Client做操作

Else if 收到High

將影片的Quality調整至dpi=1920,1080 jpeg壓縮品質80%

Else if 收到Medium

將影片的Quality調整至dpi=1280,720 jpeg壓縮品質50%

Else if 收到Low

將影片的Quality調整至dpi=640,480 jpeg壓縮品質 30%

Else if 收到 Set\_Parameter(Client從空格處輸入要求)

根據以下要求來選擇來源檔

1.Movie/Stream(播放影片檔)

2.Camera(播放從Server前置鏡頭拍出去的畫面)

3.Menu(回到初始畫面)

4.Face(從Server前置鏡頭拍出去的畫面處理後將人臉

辨識的結果回傳)

**傳輸資料()---跟迴圈函數是在不同的執行緒上兩者是平行運行**

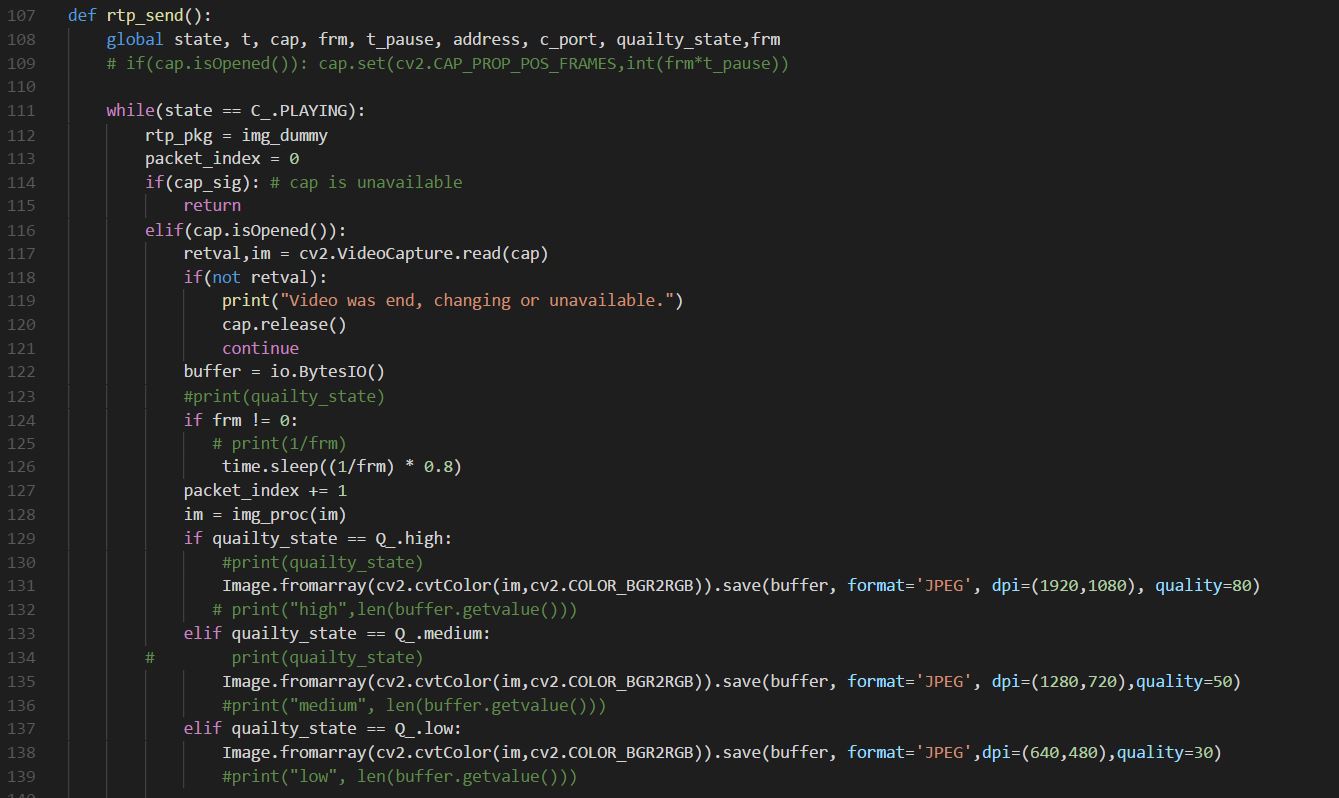
判斷State是Play才會進入傳輸檔案的迴圈

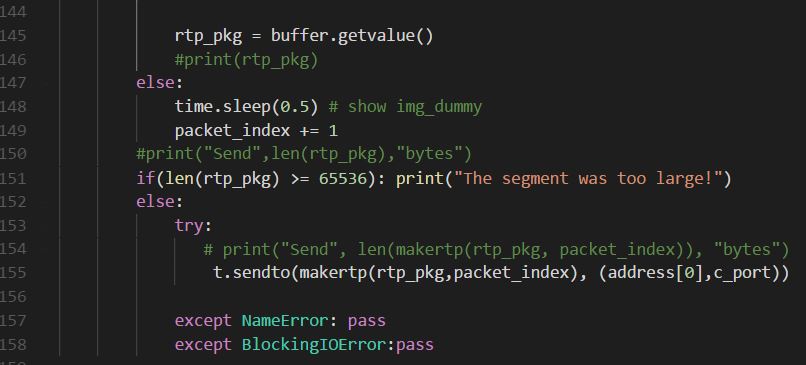
判斷Frame rate是否為0

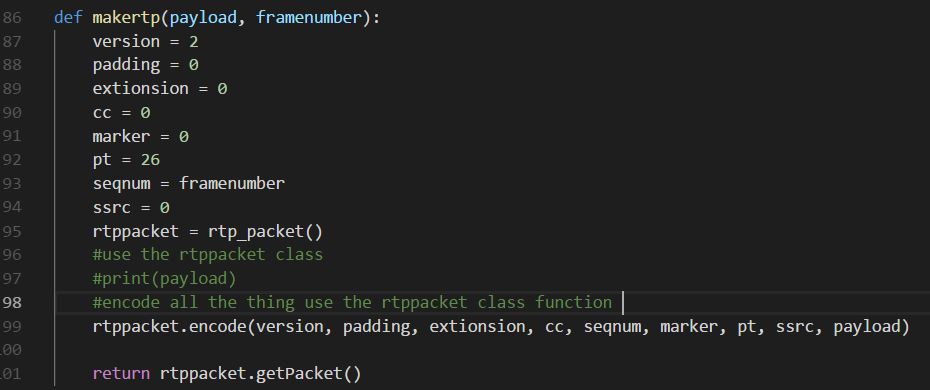
(Frame rate為0代表是來自即時串流的Camera，如

果是一般影片播放，為了避免再傳到Clinet端的速度太快，所以這邊利用Sleep來減緩送Frame的速度)

根據Client端來決定要播哪種畫質的影片



 將影片的Frame轉成二進位型態後和Frame\_number傳入做成Packet的Function，將轉成二進位Frame跟符合RTP的Header結合成一個Packet，再將其傳到指定位址，每傳送一次就將Frame number加1。

**製作Packet的函數():**

**s\_client\_UDP.py:**

1. **主要功能**

s\_client\_UDP.py是作為RTP跟RTSP的client端，實現串流控制播放器，提供Play、Pause、Setup、Teardown還有Quality選擇給使用者端操作。

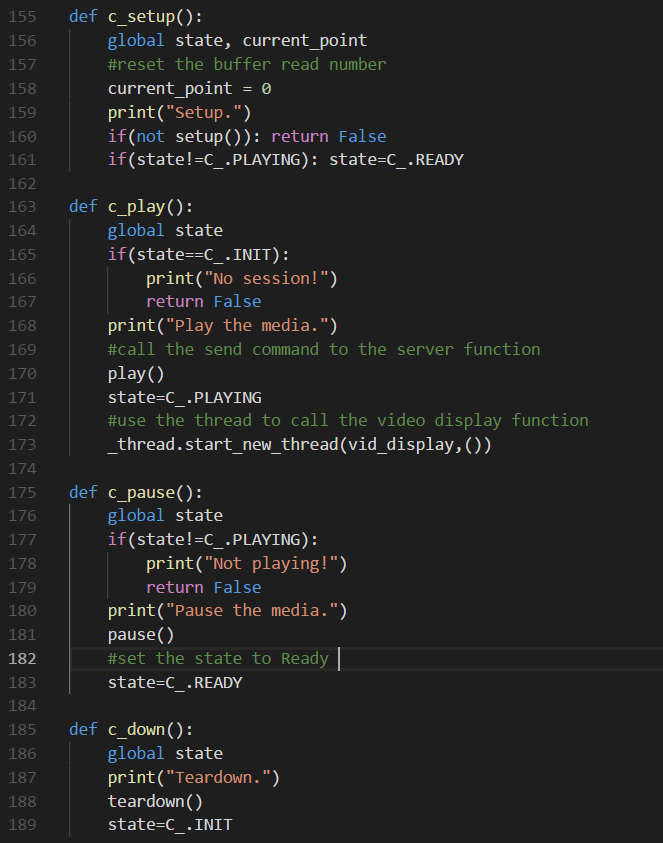
1. **使用的網路協定**

基於UDP的RTP網路協定來實作使用者端能夠控制影片的RTSP協定。

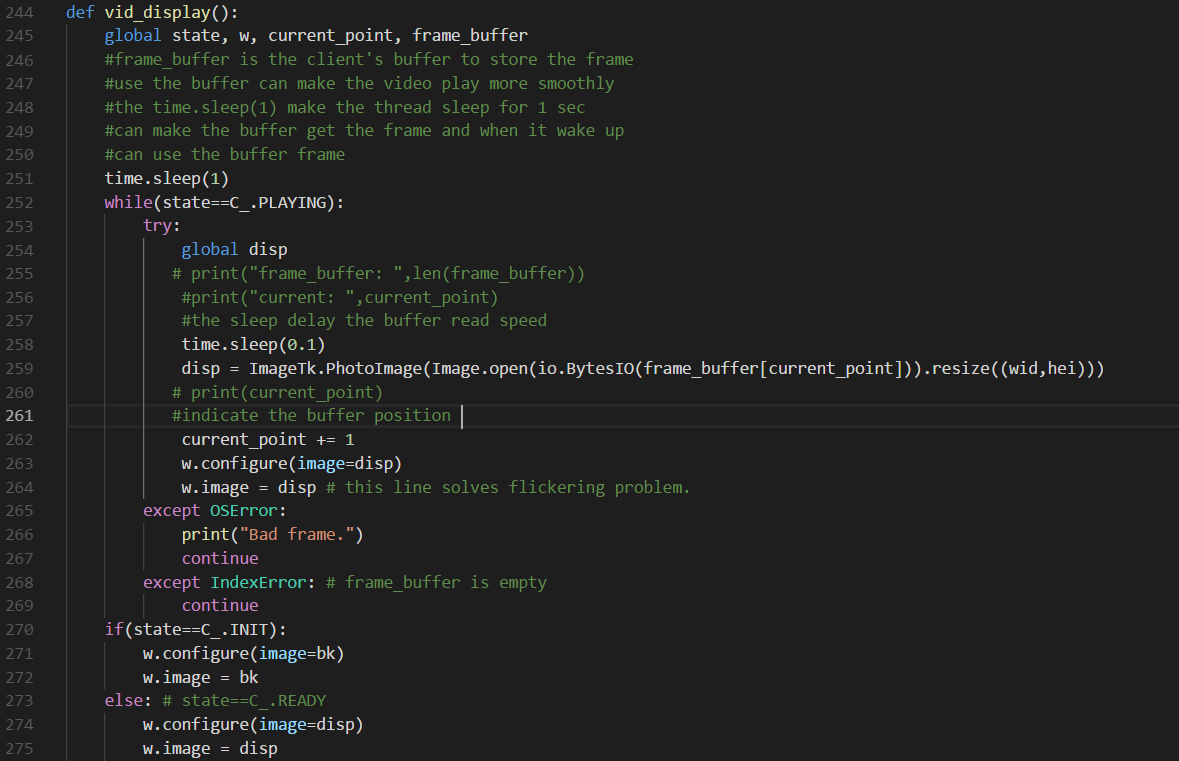
1. **運作流程**

產生GUI，不同按鈕對應到不同Function，當使用不同Function時都會傳相對應的需求給Server端





****



**伍、 實作困難**

1. 我們之前在Python上沒有開發過較大的程式，因此首先遇到的困難就是如何將Package引入並使用其功能。之後我們設計了RTSP的架構，為了使Request與Response合致，必須去維護兩邊的state。
2. RTSP控制、RTP封包收發、Server影片讀取、Client影片顯示四者必須同時平行進行，一開始我們毫無頭緒。我們後來把RTSP放入mainloop，其他三者都放入thread中的迴圈，彼此透過state、flag以及buffer來溝通，就不會互相阻擋干擾了。
3. 在引入人臉辨識時，需要佈署Dlib和臉部辨識檔，在Windows和Python下並不容易。我們查閱了許多技術文件後，終於作出了現行的方案，用已編譯檔迴避掉CMake編譯。

**陸、 未來改進**

* + 1. 在RTP header中有timestamp的資訊，我們在Client中還沒有使用到。未來可以用此資訊來實現Fixed Playout Delay，這樣Client的播放內容就不會跟Server的串流差太久。
    2. RTSP Pause後Server跟Client間還在傳輸資料，這樣重返Play時Client仍會把Buffer中的內容播完。

由於我們有Movie(預製)跟Camera(即時)兩種性質的串流內容，希望Movie暫停時Server能告訴Client，讓Client記住Movie的暫停位置，重返Play時Client再告訴Server上次中斷位置，Server直接從該處繼續播放。

* + 1. 我們在RTSP中預留了Audio Play的指令，希望未來我們掌握音訊片段化再重組的技術後，可以加入音訊串流的功能。
    2. 希望未來加入影片彈幕、彈幕語音等有趣的功能。
    3. 希望可以換掉Windows專屬的Package，增加平台相容性。