

# ESLab Final Project Report

b08901164 電機三 林霽瑀

b08901053 電機三 張苾慈

b08901051 電機三 王濰紳

## 一、動機


遠程呈現 (telepresence) 是時下十分熱門的技術, 加上這一年受到疫情的影響, 我們想要結合課堂上學到的知識, 透過遠端操控車子以及網頁的結合, 讓使用者有完整的功能與體驗, 不用出門也能身歷其境參與展覽。

## 二、作法

### 呈現方式

展覽場內放置三個展覽品 (以 STM32 作為代表) 以及我們的車子, 使用者有專屬的網頁與一個 STM32 遙控器。使用者可以在網頁看到車上的相機直播畫面, 並透過手勢操控車子的移動。同時, 車子會偵測距離最近的展覽品編號, 並在網頁上顯示對應的展覽品資訊, 使用者也能看到目前車子的移動狀態。網頁的呈現如下圖:

### ESLab Final




Controls

Car Direction : stable

しろくま

北はもう  
だめだ...  
ずるずる...



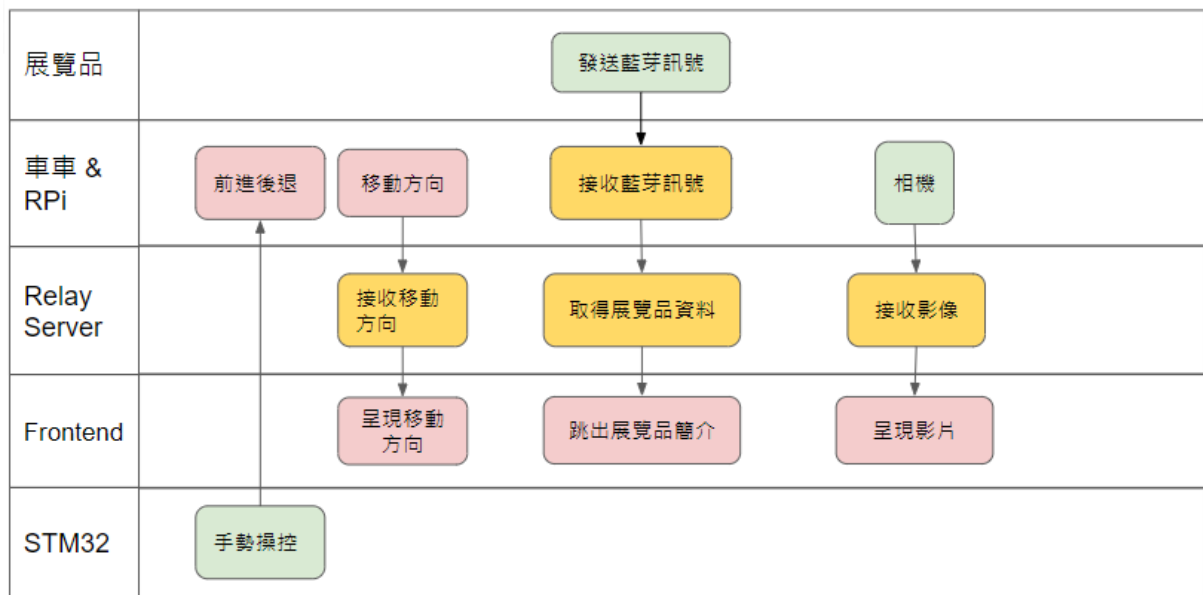
北からにげてきた、さむかりで  
ひとみしりのくま。あったかい  
お茶をすみっこでのんでる  
ときがおちつく。

Exhibit Information

白熊

超級害羞的白熊，專長是畫畫，興趣是喝茶。因為北方實在太寒冷，剛好他又非常怕冷，所以毅然收拾包袱逃往南方。應對寒冷的方法就是用包袱巾包裹自己。

流程圖：



詳細作法：

## STM32 – 遙控器

### (一) 手勢偵測

寫判斷式判斷偵測到的三軸加速度數據，一次判斷三十筆資料的平均，以 moving average filter 過濾掉一些無意義抖動，將 STM32 的狀態分成五類，穩定平放 (stable)、前傾(forward)、後傾(backward)、左傾(left)和右傾(right)。

### (二) 資料傳送

這邊使用的是 mbed-os-example-sockets 的架構去改的。我們加了一個傳送 HTTP POST Request 的函式，裡面傳送的就是當下判斷的控制信號。

## STM32 – 展覽品

我們以三個STM32作為展覽品的象徵，其功能主要就是持續發送藍芽訊號，程式的部分使用 mbed-os-example-ble 進行修改，需要注意的是因為我們另外借的 STM32 型號有所不同，因此要在 mbed\_app.json 內新增型號資訊。

```
"B_L4S5I_IOT01A": {
  "target.components_add": ["ism43362"],
  "ism43362.provide-default": true,
  "target.network-default-interface-type": "WIFI",
  "target.macros_add" : ["MBEDTLS_SHA1_C"]
}
```

## RPi 車子

### (一) 控制車子

接收從 STM32 傳送過來的字串以決定方向和速度，利用 Adafruit\_PCA9685 和 Adafruit\_GPIO 的 I2C 調整 pwm 在不同 address 上的參數去控制車子前輪伺服馬達的角度和後輪直流馬達的速度。

### (二) 展覽品偵測

RPi 會持續偵測三個展覽品藍芽訊號的 RSSI 值，每次選出最大者，考慮最近七次結果的眾數作為最終選擇，再利用 requests post 到網頁。

### (三) 影片直播

影片直播的實作方式是利用 Gstreamer 的 command line tool。這次我們選擇使用的協定是 HLS，也就是一個 HTTP based 的協定。選擇這個協定的原因就是因為他能夠讓我們比較容易的顯示在網頁上，呈現一個完整的使用者介面，而且延遲也可以壓在1~2秒之間。

原本我們也有考慮要使用延遲較低的 UDP based 的 WebRTC 技術，但是因為複雜度較高，因此我們認為可以規劃成未來展望的優化。

### (四) 資料接收與傳送

我們在 Port 3000 架設了一個 Flask Server，是一個 python based 的輕量化的後端架構。利用 HTTP 的協定來和 STM32 以及網頁的後端溝通。我們會從 STM32 接收到控制信號，並且將控制信號傳回給網頁的後端。

## 使用者介面

### (一) 網頁前端

這邊是利用 HTML、Javascript 完成介面的呈現。設計的部分，我們使用了 Bootstrap 的 stylesheet。

### (二) 後端

這邊也是使用 Flask，在 Port 5000上跑我們的後端。這邊主要的功能是需要達成不斷的接收到 RPi 回傳目前車子移動的方向，並且不斷地更新前端的網頁。這邊比較困難的是要做到server主動的更改畫面(server-sent event)，而不是在使用者觸發之下更改畫面，這樣就可以利用 HTTP 協定，達到 websocket 類似的效果。

```
@app.route("/listen")
def listen():
    def respond_to_client():
        toggle = 0
        while True:
            global data
```

```
_data = json.dumps({
    "camera" : data.camera,
    "direction" : data.direction,
    "name" : data.name,
    "info" : data.info,
    "img":data.img
})
time.sleep(0.1)
toggle = not toggle
yield f"id: 1\ndata: {_data}\nevent: online\n\n"
return Response(respond_to_client(), mimetype='text/event-stream')
```

除此之外，我們使用了 Video.js 協助我們可以向 RPi 拿取 HLS playlist、segment.ts 檔案，並且不斷的呈現在前端。(補充：我們在 RPi 的 Port 8080 上架了一個 Simple HTTP Server，可以允許我們拿到在目錄底下的檔案)

## 三、成果

### 功能介紹

#### (一) STM32 手勢控制車子前進

利用 STM32 的手勢，可以控制在展覽場的車子移動。我們總共分成五種移動方式：停下(stable)、前進(forward)、後退(backward)、左轉彎(left)、右轉彎(right)。當 RPi 接收到 STM32 所發送的指令，就會執行上述的五種的其中一項，並且將目前車子收到的指令回傳給使用者看到的前端網頁。

#### (二) RPi 直播展覽場的現場畫面到使用者的前端

使用者可以透過前端網頁看到 PiCamera 所拍攝到的直播畫面，並且由此觀賞展覽。這個畫面也是使用者操縱車子移動的依據。

#### (三) RPi 偵測展覽品的藍芽訊號與展覽品簡介呈現

每一個展覽品都會不斷的發射藍芽訊號，RPi 也可以透過偵測訊號的RSSI值判斷現在應該距離哪一個展覽品比較近，將最近的展覽品名稱回傳給使用者，前端也會跳出展覽品的相關資訊。

## 四、問題與解決方法

### ● 影片延遲問題

因為選用的協定 HLS 具有固有延遲，也就是 playlist 裡面的 segment 數量乘上每個 segment 的長度，因此我們有試著優化到最低的延遲。調整的方式是將 encode 的速度設定成 speed-preset=ultrafast，將 playlist 長度調成1，每一個片段的長度也設定成1，這樣固有的延遲就會是1秒，而且 encode 的速度也會比較快。檢視最後的結果可以看出延遲只會在1~2秒之間。

```
#!/bin/bash
echo "Launching gst"
gst-launch-1.0 -v v4l2src device="/dev/video0" ! videoconvert !
clockoverlay ! \ videoscale ! video/x-raw,width=640,height=360 !
x264enc bitrate=256 speed-preset=ultrafast !
video/x-h264,profile="high" ! \ mpegtsmux ! hlssink
playlist-root=http://192.168.0.156:8080/home/pi/eslabfinal/rpi/camera
location="/home/pi/eslabfinal/rpi/camera/segment_%05d.ts"
playlist-location="/home/pi/eslabfinal/rpi/camera/playlist.m3u8"
target-duration=1 max-files=10 playlist-length=1
```

- 控制信號延遲問題

一開始我們是將 STM32 的控制信號傳給網頁的後端，再由網頁的後端把控制信令傳給 RPi。但是這樣會造成車子在收到控制信令之前經過兩次的延遲，因始最後我們把架構改成：STM32 的控制信令直接傳給 RPi，RPi 接下來再把自己現在的狀態回傳給網頁。這樣使用者的控制信令可以更快的抵達車子，操作上也比較順暢。

- RPi上機運行問題

我們有發現如果想要在開機的時候使用到藍芽模組，我們必須要先等待個 10~20秒左右才可以。

```
/etc/rc.local
sudo /home/pi/eslabfinal/rpi/camera/camera.sh &
sudo service bluetooth start &
sleep 20s
sudo python3 /home/pi/eslabfinal/rpi/ble.py &
sudo python3 /home/pi/eslabfinal/rpi/rpi_car.py
```

- 藍芽RSSI值不穩定

我們有發現藍芽的 RSSI 值很容易受其他信號或是環境的干擾，而造成很大的波動。原本我們使用的方法是取當下 RPi 量測到的最大值，但是這樣展覽品資訊會不斷的更換，顯示的也經常不是最近的展覽品。

後來，我們嘗試設定一個 RSSI 值門檻，假如超過這個門檻值才會被比較、選用，但是我們發現這樣受絕對距離的影響很大，只要距離一遠，就不會有任何展覽品的資訊跳出來。另外，我們也試過規定連續k次最大者都相同才更新結果，但由於訊號強度的不穩定，很可能使沒有任何一個答案可以連續出現 k 次，造成結果無法更新。最後我們選定的方法是採取七次量測值當中的眾數當作我們的結果，這樣在切換不同展覽品資訊的時候雖然會有一些延遲，但是可以確保顯示的結果會穩定下來。

## 五、課程知識的運用

### (一) Input/Output Devices

這次我們使用到許多 Input/Output Devices, 例如 : STM32 讀取三軸加速度感應器資料、操控車子的移動、讀取 PiCamera 回傳的影片片段等等。

### (二) Multi-tasking

這次的專題當中, 我們也大量使用了 multi-threading 的技術。主要原因就是因為車子必須要同時做許多事情。像是: 讀取收到的控制信令, 並且將該信令再傳給網頁的後端。這邊因為必須要頻繁讀、寫同一個物件, 所以我們也使用了 mutex acquire/release 的技術, 這樣可以保護該物件不會在一個 thread 進行寫的同時, 導致另一個 thread 讀取到錯誤的值。但是我們並沒有將所有同時進行的程式都放進同一個檔案, 因為有些平行執行的程式其實是不需要資源的共享的, 這時我們就會選擇把它拆成另一個獨立的檔案, 保持程式的可讀性。

```
@app.route("/controlrpi", methods=['POST'])
def index():
    global current_direction
    input_json = request.get_json(force=True)

    if input_json:
        if "direction" in input_json:
            car.get = True
            data.direction = input_json.get("direction")
            mutex.acquire()
            current_direction = data.direction
            mutex.release()
        else:
            car.get = False
            if "camera" in input_json:
                data.camera = input_json.get("camera")
    print(data.direction)
    car.get_command(data.direction)
    return Response(status=200)
```

```
def sendDirection():
    while(True):
        global current_direction
        mutex.acquire()
        info_dict = { "direction" : current_direction }
        mutex.release()
        res = requests.post('http://192.168.0.131:5000/direction',
        json=info_dict)
        time.sleep(0.1)
```

```
if __name__ == '__main__':  
    threading.Thread(target=lambda: app.run(host="0.0.0.0", port=3000,  
debug=True, use_reloader=False)).start()  
    thd4 = threading.Thread(target = sendDirection())  
    thd4.start()
```

### (三) Wireless Connection

這次我們也使用了許多無線連接的技術，像是 STM32、RPI、網頁後端、網頁前端之間都是利用無線網路進行溝通。而 RPI 和展覽品之間則使用了 BLE 的技術。

## 六、未來展望

- 即時串流：如果能應用 WebRTC，應該會比用 HLS 有更好的即時串流效果。
- 遠端遙控：我們的 project 目前只能在區網下執行，但遠端遙控對實務上應該會比較有幫助。
- 規劃路線：希望之後可以讓車子自動往最近的展覽品移動，這樣可以省去使用者自己操控，而且也可以解決使用者對於展覽場空間上不熟悉的問題。
- 精準選擇展覽品：目前我們使用取眾數的方式減少 RSSI 值不穩定的影響，但這個方法會有一定的延遲，無法針對改變即時做出反應，若能搭配監督式學習 (supervised learning) 的方法，也許可以讓這個功能更穩定且有效率。
- 超音波防撞功能：利用超音波感測器偵測障礙物，在障礙物前一定的距離停下，提高車子的安全性。尤其是我們的相機只能看到前方，若對場地不是十分熟悉，倒車時使用者很難避免撞到障礙物。

## 七、心得

做這次project讓我們學到了以下幾件事情：

### (一) 資料搜尋與問題解決能力

Debug 的時候會發現，我們遇到的問題常常也是很多人曾經的問題，在 google 搜尋 error message 往往會對 error 有更深入的了解，更快、更容易解決問題。特別是有時候 error message 是出自於使用的套件本身，在對於 source code 不熟悉的狀況下，上網查詢問題可以妥善的利用他人的經驗。



## (二) 簡單的方法未必無效

有些做法雖然簡單但卻實際，例如本來手勢偵測我們都以為要利用 ML train 一個 model 才能有效偵測，但其實利用 accelerator 的資料取平均加上幾個判斷式就可以簡單且準確的判斷 STM32 傾斜方向。

## (三) 多方嘗試的能力

有些問題需要多方嘗試才會得到解答，例如關於如何開機就運行的問題，ble.py 一開始一直無法在開機時就運行，和其他檔案比較過後，我們將可能的問題聚焦在藍芽模組，試過許多可能的解法之後，最終發現加了20秒的等待時間竟然就成功了！我們也有嘗試過等待10秒，卻是無法成功的，由此也能看出多次嘗試與實驗的重要性，若在10秒失敗時就放棄，我們可能就錯過了解決的方法。

## (四) 學會簡化原來的問題

我們學會在有限的時間內如何取捨，保留最核心、完整的功能。有些技術其實需要更多的網路知識基礎，例如：我們本來期望能遠端遙控車車，但最後和助教討論過後，認為和課程重點較無關聯，而且難度較高，因此最後選擇在區網下面進行，也成功做出一個完整的架構。

最後，很開心一開始的發想最後都大致有實作出來，包括：STM32 手勢控制車子前進、RPI 直播展覽場的現場畫面到使用者的前端、RPI 偵測展覽品的藍芽訊號與展覽品簡介呈現。

# 八、程式碼與Demo影片

- Github Repository : <https://github.com/linpeiyu164/eslab-final.git>
- Video Demo : <https://youtu.be/eYAhpLyBEtk>

# 九、參考文獻與資料

## 現成資源的使用

- Donkey car 是向實驗室借用的
- eslab-final/rpi/car.py : 使用了 Donkey car 提供的 PCA9685, PWMSteering, PWMThrottle 的 class

## 參考資料

1. 如何開機運行：  
<https://hms5232.medium.com/raspberry-pi-%E9%96%8B%E6%A9%9F%E8%87%AA%E5%8B%95%E5%9F%B7%E8%A1%8C-shell-script-e5b60781bfa0>,  
<https://askubuntu.com/questions/500071/running-a-sh-to-execute-multiple-commands>
2. Gstreamer documentation : <https://gstreamer.freedesktop.org>
3. Donkey car documentation : <https://docs.donkeycar.com/>
4. 在瀏覽器上利用gstreamer和pi camera直播影片:  
<http://4youngpadawans.com/stream-live-video-to-browser-using-gstreamer/>