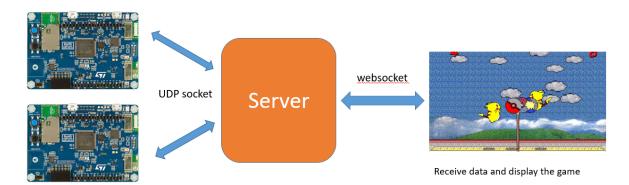
ESLAB Final Project Pikachu-Volleyball B06901162 電機四 梁瀚中 B06901166 電機四 陳杰民

1. Motivation

- NTUEE 的大家都喜歡皮卡丘打排球。
- 想讓這個遊戲變得更有挑戰且更需要操作
- 於是我們利用 STM32 裡面的三軸加速計和陀螺儀來控制皮卡丘移動

2. Implementation

Work flow



Detect and send player movement

STM32

我們主要使用 STM32 來偵測玩家的行動並且透過 UDP socket 將偵測到的動作傳送給 server。皮卡丘打排球總共擁有上下左右和一個殺球鍵,而由於皮卡丘的殺球方向也和上下左右有關,為了方便控制皮卡丘的殺球方向,我們將上鍵分開成 up 和 jump 分別判斷。各個按鍵的判斷方式如下

- ◆ 左右鍵:透過讀取 STM32 三軸加速度計 X 軸的值來判斷。
- ◆ 上下鍵:透過讀取 STM32 三軸加速度計 Y 軸的值來判斷,不過 由於我們在玩的過程之中發現手拿著 STM32 很難注意到自己是 否拿的是平的,所以我們會記錄過去平均的值來做為 threshold, 以便在殺球的時候可以做出上下的指令來控制殺球的方向。
- ◆ Jump 鍵:透過讀取 STM32 陀螺儀來獲取角速度,積分角速度來 得到角位移,以此來判斷皮卡丘是否起跳。
- ◆ 殺球鍵:透過外接一個按鈕來做為殺球鍵的輸入。
- ◆ STM32 藍色按鈕:設定 Button interrupt 讓 STM32 重新連接和重 新校準。

- ◆ Eventqueue: 我們透過設定 eventqueue 來確保 sample 和傳送 資料在相同的時間間隔進行,每 5ms 傳送資料,每 2ms 做 sample。
- Python server
 - ◆ 用 udp server 接收後用 websocket 將資料傳到遊戲端
- Pikachu Volley
 - ◆ Reference from https://github.com/gorisanson/pikachu-volleyball. 所做的方法把原始的執行檔經過 decompile 之後再將其改成 javascript 的版本,我們有更改其中關於 keyboard 和遊戲輸入的設定,使遊戲可以照我們傳送訊息的方式遊玩。

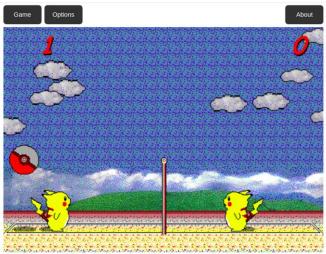
3.Results

- STM32
 - ◆ 偵測和傳送玩家做出的指令。
 - ◆ 按下藍色按鈕可以重新校準和重新連接網路。
- Python Server
 - ◆ 接收來自 STM32 的訊息。
 - ◆ 建設 websocket 傳送玩家的指令給遊戲。
- Pikachu volleyball
 - ◆ 接收來自 websocket 的訊息。
 - ◆ 1 player and 2 player mode ∘
 - ◆ 可以成功打贏電腦。
- Screenshots

```
"right":0, "left":0, "up":0, "down":0, "hit":0, "jump":0)
"right":0,
```









- Demo video
 - https://youtu.be/HzImpa4oTNo
 - https://youtu.be/Qfkzt2Bb2ZQ

4.References

- https://os.mbed.com/docs/mbed-os/v6.6/apis/udpsocket.html
- https://github.com/NTUEE-ESLab/2019-AirHockey/tree/master/RPi
- https://github.com/gorisanson/pikachu-volleyball
- ESLAB hw7