

F i n a l P r o j e c t

協力遙控車推球遊戲

Team3

王承翰

盧昱愷

C o n t e n t s

- 遊戲規則
- 系統架構圖
- 實作細節
 - 遙控車
 - 球門server
 - 玩家操作
- 問題解決
- DEMO

PART 01

遊戲規則

遊 戲 規 則

1. 限時2分鐘，玩家控制並利用自走車將球推进球門，進球即得分。
2. 支援**多位玩家同時存取**自走車控制權，共同決定速度與方向。
3. 自走車上配有鏡頭可供觀看。
4. 球門上樹莓派，負責接收連線，顯示分數與遊戲剩餘時間。
5. 球門上方架設鏡頭偵測球門內的球數以記錄分數。
(怕紅外線或超音波感測器會因為球卡在門上而偵測錯誤)
6. 提供分數紀錄，輸入姓名開始遊玩，也提供排行榜功能。

示意圖



PART 02

系統架構圖

軟體架構

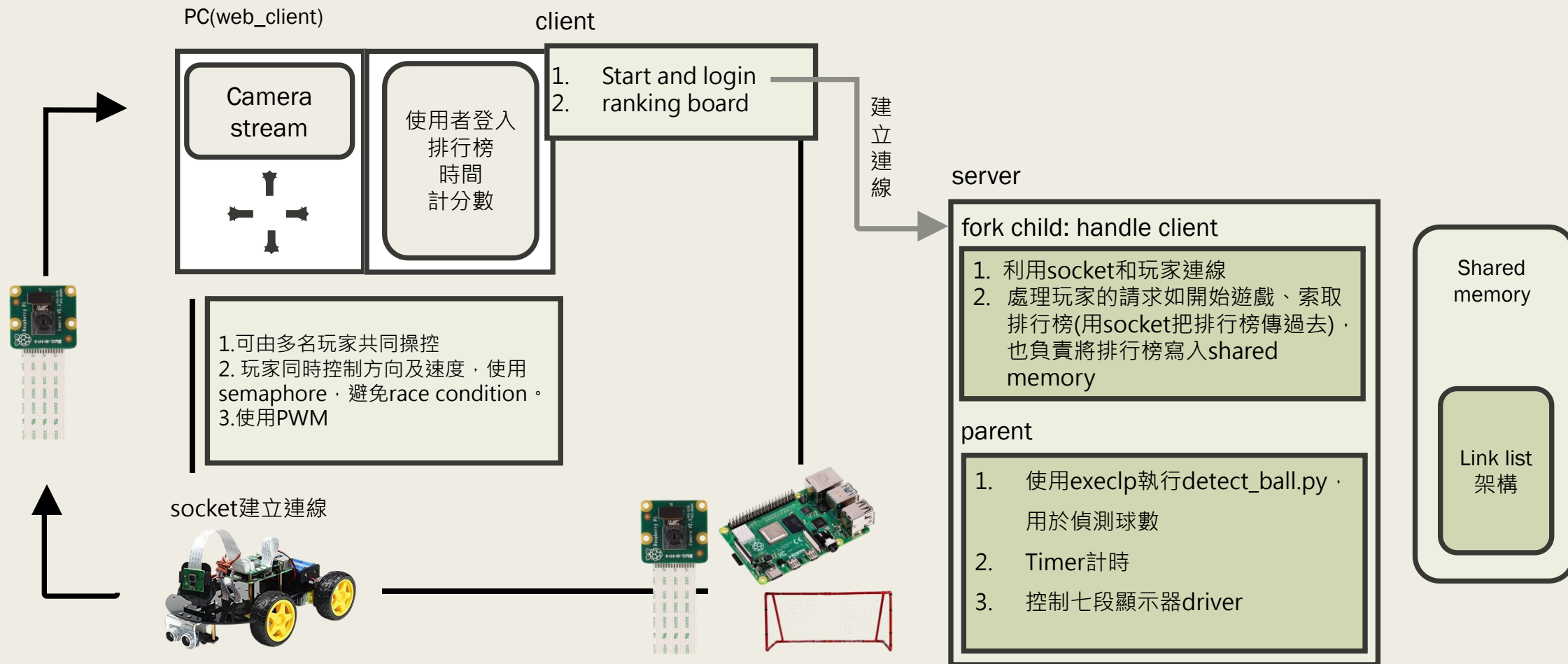
■ 遙控車

- **car_driver.c**
 - 控制GPIO及PWM
- **car_server.c**
 - 接收多個web指令
 - 統整車輛移動指令
 - **web_server.py**
 - **web.html**
 - 傳送即時影像
- **start_car_server.sh**
 - 開機自動啟動

■ 球門

- **goal_server.c**
 - 接收多個web指令
 - 控制遊戲流程
 - 分數紀錄及排序
 - **detect_ball.py**
 - 偵測進球數量
- **RUN.sh**

系統架構圖



PART 03

實作細節

玩家操作介面
Web


遙控車

球門主機

玩家操作介面


請輸入玩家名稱

player1



目前速度：Left:0 Right:0

=====Ranking Board=====		
ASD	6	2024-12-26 17:42
ADw	6	2024-12-26 17:43
444	5	2024-12-26 17:48
player1	5	2024-12-26 17:48
333	5	2024-12-26 17:48
444	4	2024-12-26 17:46
333	4	2024-12-26 17:46
222	3	2024-12-26 17:42
ASD	3	2024-12-26 17:42
111	3	2024-12-26 17:42

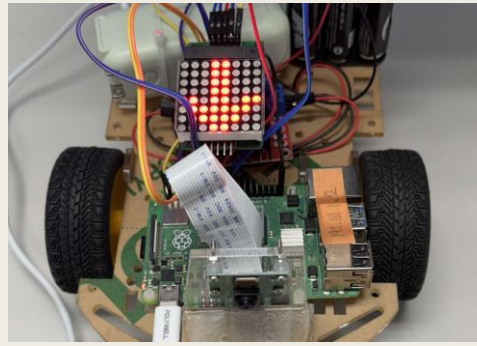


目前速度：Left:0 Right:0

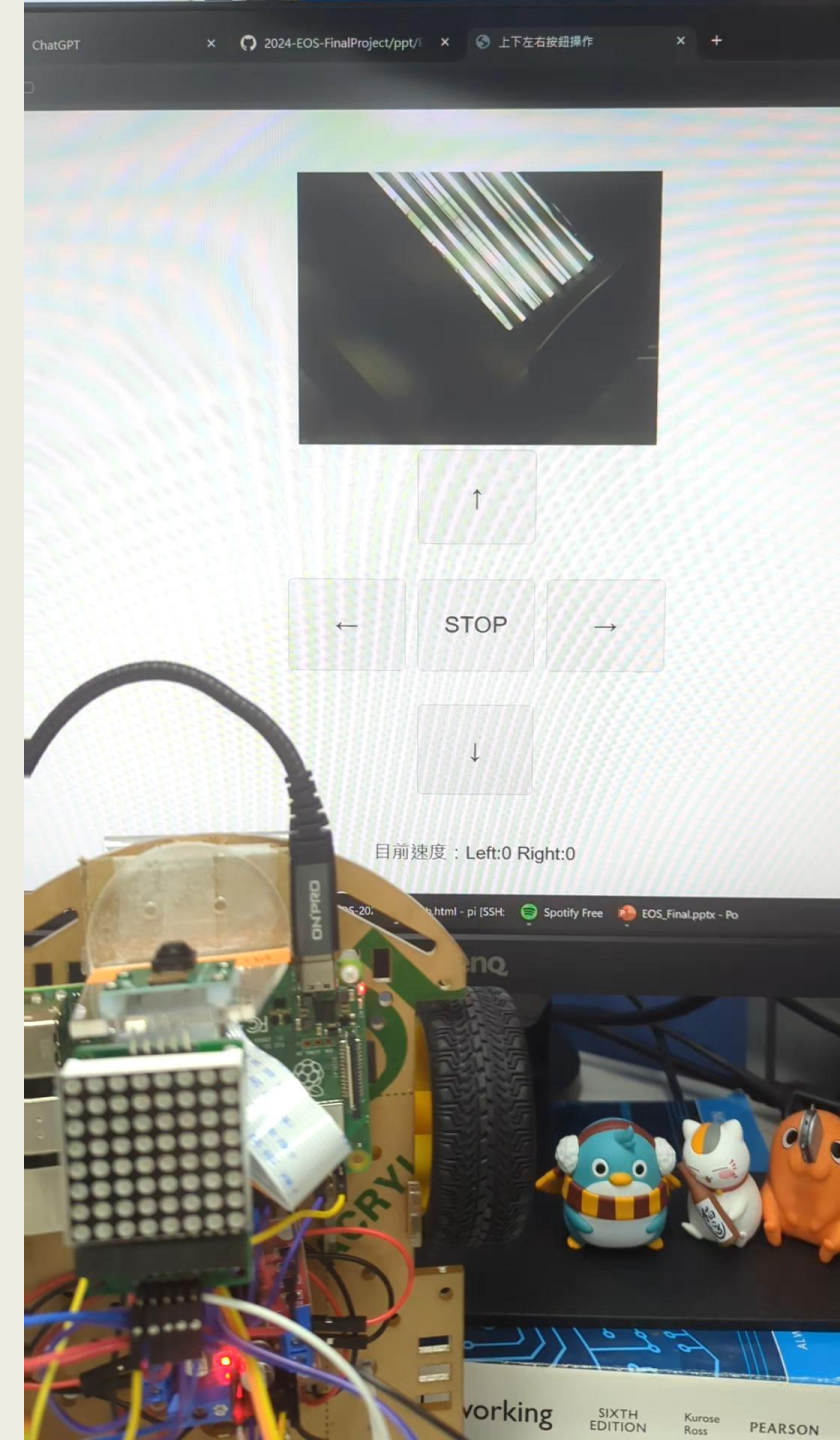
以鍵盤按壓或點擊按鈕控制，將指令以 **socket** 傳送至車及球門。

時間到，回到主畫面，並更新排行榜。

遙控車



- 利用shared memory儲存左右輪速度，配合semaphore避免race conditions，
- 在driver以PWM控制速度，GPIO控制轉向。
- 利用SPI介面控制8*8 LED矩陣，以箭頭顯示移動方向。
- Socket回傳鏡頭畫面。
- 架設 Flask web server。



協力操控

Car driver中定義PWM操作

```
void set_left_gpio(int lb, int lf){
    gpio_set_value(LEFT_BACKWARD_PIN, lb);
    gpio_set_value(LEFT_FORWARD_PIN, lf);
}

void set_right_gpio(int rb, int rf){
    gpio_set_value(RIGHT_BACKWARD_PIN, rb);
    gpio_set_value(RIGHT_FORWARD_PIN, rf);
}

void set_pwm_duty_cycle(struct pwm_device *pwm, int duty_cycle){
    struct pwm_state state;
    pwm_get_state(pwm, &state);
    if (state.period == 0) {
        state.period = 2000000; // 默認 20ms 週期
    }

    state.duty_cycle = state.period * duty_cycle / 100; // 設置占空比
    // 禁用 PWM
    if (pwm_is_enabled(pwm)) {
        pwm_disable(pwm);
    }

    pwm_apply_state(pwm, &state); // 應用設置
    pwm_enable(pwm); // 啟用 PWM
}

if (copy_from_user(rec_buf, buf, len) != 0) {
    pr_err("ERROR: Not all the bytes have been copied from user\n");
    return -EFAULT;
}

rec_buf[len] = '\0';

// 解析指令格式 · 例如 "w 80 60"
sscanf(rec_buf, "%s %d %d", command, &duty_cycle_left, &duty_cycle_right);
pr_info("Write Function : %s %d %d\n", command, duty_cycle_left, duty_cycle_right);

if (strcmp(command, "rf") == 0) { // 右前進
    set_right_gpio(0, 1);
} else if (strcmp(command, "rb") == 0) { // 右後退
    set_right_gpio(1, 0);
} else if (strcmp(command, "lf") == 0) { // 左前進
    set_left_gpio(0, 1);
} else if (strcmp(command, "lb") == 0) { // 左後退
    set_left_gpio(1, 0);
} else if (strcmp(command, "stop") == 0) { // 停止
    set_left_gpio(0, 0);
    set_right_gpio(0, 0);
} else if (strcmp(command, "pwm") == 0) { // 設置 PWM
    set_pwm_duty_cycle(pwm_left, duty_cycle_left);
    set_pwm_duty_cycle(pwm_right, duty_cycle_right);
}
```

8*8點矩陣

```
uint8_t arrow_left[8] = { 0x18, 0x3C, 0x7E, 0xDB, 0x18, 0x18, 0x18, 0x18 };
uint8_t arrow_right[8] = { 0x18, 0x18, 0x18, 0x18, 0xDB, 0x7E, 0x3C, 0x18 };
uint8_t arrow_down[8] = { 0x10, 0x30, 0x60, 0xFF, 0xFF, 0x60, 0x30, 0x10 };
uint8_t arrow_up[8] = { 0x08, 0x0C, 0x06, 0xFF, 0xFF, 0x06, 0x0C, 0x08 };
uint8_t arrow_stop[8] = { 0x18, 0x18, 0x18, 0x18, 0x18, 0x18, 0x18, 0x18 };
uint8_t arrow_lower_left[8] = { 0xFC, 0xF8, 0xF0, 0xF8, 0xDC, 0x8E, 0x07, 0x03 };
uint8_t arrow_upper_left[8] = { 0x3F, 0x1F, 0x0F, 0x1F, 0x3B, 0x71, 0xE0, 0xC0 };
uint8_t arrow_lower_right[8] = { 0x03, 0x07, 0x8E, 0xDC, 0xF8, 0xF0, 0xF8, 0xFC };
uint8_t arrow_upper_right[8] = { 0xC0, 0xE0, 0x71, 0x3B, 0x1F, 0x0F, 0x1F, 0x3F };
uint8_t none[8] = { 0, 0, 0, 0, 0, 0, 0, 0 };
```

Car_server定義左輪右輪
操作使用semaphore保護

KerryYK_Lu, 38 minutes ago | 1 author (KerryYK_Lu)

```
typedef struct {
    int right;
    int left;
} shared_data_t;
```

```
// write to shared memory
if(left != 0){
    P(sem_id, 0);
    if (shm_ptr->left + left > 50){
        shm_ptr->left = 50;
    }else if(shm_ptr->left + left < -50){
        shm_ptr->left = -50;
    }else{
        shm_ptr->left += left;
    }
    left = shm_ptr->left;
    V(sem_id, 0);
}

if(right != 0){
    P(sem_id, 1);
    if (shm_ptr->right + right > 50){
        shm_ptr->right = 50;
    }else if(shm_ptr->right + right < -50){
        shm_ptr->right = -50;
    }else{
        shm_ptr->right += right;
    }
    right = shm_ptr->right;
    V(sem_id, 1);
}
```

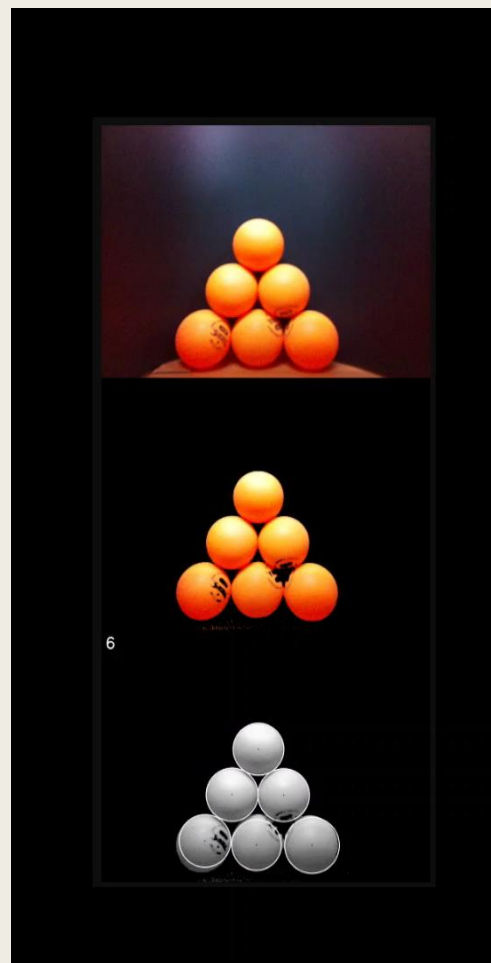



球門主機

- 排行榜利用[share memory](#)供各child存取，資料為link list結構，使用offset偏移量實作。
- 排行榜只取前十名顯示。
- 用[semaphore](#)做同步，讓遊戲時間內進來的所有人同時離開遊戲，並且分數相同。
- 遊戲時間利用4digit七段顯示器顯示。
- 用[signal](#)防止[zombie process](#)產生。
- 用[multitask](#)接收玩家連線。
- fork一個process利用[execlp](#)在背景執行"detect_ball.py"
- 寫一個[.service](#)檔案在systemd中，用於開機直接執行script，script則呼叫程式的執行與紀錄log。

球偵測

- detect_ball.py
 - 將橘色濾出
 - 霍夫曼圓檢測
 - 給定半徑範圍及重疊率
 - 得出數量
 - 以4-digit 七段顯示器顯示



其他細節

Link list架構排行榜

```
typedef struct Node{
    char name[NAME_SIZE];
    int score;
    char time[20];
    int offset; // offset from first NODE
}Node_t;

typedef struct list{
    int head;
    int size; // node amount
    Node_t nodes[CAPACITY+1]; // array
}List_t;
List_t *list;
```

execlp執行betect_ball.py

```
// call python detect_ball
python_pid = fork();
if (python_pid == 0){
    execlp("python3", "python3", "detect_ball.py", (char *)NULL);
}else{
    printf("python pid:%d\n",python_pid);
}
```

Semaphore做同步

```
// Set semaphore initial value = 0 for sync when
int val = 0;
if (semctl(s, 0, SETVAL, val) < 0){
    perror("Semaphore set value to 1 failed");
    exit(EXIT_FAILURE);
}

P(s);
FILE *file = fopen("game_score.txt", "r");
if (file == NULL) {
    perror("fopen");
}
fscanf(file, "%d", &game_score);
fclose(file);
add(list, name, game_score);
V(s);
```

script file

```
#!/bin/bash

# Will auto run this bash when boot
cd /home/pi/Goal
echo "$(date) - Starting goal_server..." >> /home/pi/Goal/log.txt
./goal_server 8888 >> /home/pi/Goal/log.txt 2>&1
echo "$(date) - goal_server finished." >> /home/pi/Goal/log.txt
```

PART 04

問題解決

問題解決

- OpenCV檢測球數值抖動

解決:

- (1) 將球門底部貼黑，增加顏色對比，濾出橘色降低誤差
- (2) 以過去10個值做平均(Moving Average)

- 影像串流檢測球Loading太重

解決:

- (1) 從影像串流改成定時拍照檢測球，降低計量。

- 車輛速度控制

- (1) 以硬體PWM減少軟體開發複雜度。
- (2) 硬體PWM腳位無作用，各版本腳位不同，以raspi-gpio get確認。

PART 05

D E M O

D E M O



感 謝 聆 聽