

國立虎尾科技大學
機械設計工程系
cd2023 2a-pj1ag4 分組報告

網際足球泡泡機器人場景設計

**Web-based bubbleRob Football
Scene Design**

指導教授： 嚴 家 銘 老 師
班 級： 四 設 二 甲
學 生： 李 學 淵 (41023125)
林 呈 恩 (41023128)

中華民國 112 年 4 月

摘要

由於矩陣計算、自動求導技術、開源開發環境、多核 GPU 運算硬體等這四大發展趨勢，促使 AI 領域快速發展，藉由這樣的契機，將實體機電系統透過虛擬化訓練提高訓練效率，再將訓練完的模型應用到實體上。

此專案是運用 w3 作業所做的泡泡機器人，將其導入 CoppeliaSim 模擬環境並給予對應設置，將其機電系統簡化並導入 Lua 及 python 程式，再到 CoppeliaSim 模擬環境中進行測試演算法在實際運用上的可行性。並嘗試透過架設伺服器將 CoppeliaSim 影像串流到網頁供使用者觀看或操控。

Abstract

Due to the four major development trends of multidimensional arrays computing, automatic differentiation, open source development environment, and multi-core GPUs computing hardware. The rapid development of the AI field has been promoted. In view of this development, the physical mechatronic systems can gain machine learning efficiency through their simulated virtual system training process. And afterwards to apply the trained model into real mechatronic systems.

This project involves taking the bubbleRob created for the W3 assignment and importing it into the CoppeliaSim simulation environment with corresponding settings, simplifying its electromechanical system, and incorporating Lua and Python programming for testing algorithms in the CoppeliaSim simulation environment to determine their feasibility in practical application. Additionally, the project aims to stream CoppeliaSim images to a webpage for users to view or manipulate by setting up a server.

目 錄

摘 要.....	i
Abstract	ii
第一章 前言	1
1.1 檔案說明.....	1
1.2 遊戲說明.....	2
1.3 規則說明.....	3
第二章 製作歷程	4
2.1 移動方式.....	4
2.2 重製位置.....	4
2.3 記分板與計時	5
2.4 歡迎與恭喜.....	6
2.5 RemoteApi	7
第三章 缺陷與可改進點.....	9
第四章 心得	10
相關資料	11

圖 目 錄

圖 1.1	遊戲檔案	1
圖 1.2	目前電腦 ip	2
圖 1.3	gameip	2
圖 1.4	歡迎	2
圖 2.1	lua move	4
圖 2.2	紀錄初始位置	4
圖 2.3	重製位置	5
圖 2.4	記分板與計時	5
圖 2.5	記分板與計時程式	6
圖 2.6	歡迎	6
圖 2.7	恭喜	7
圖 2.8	歡迎與恭喜 lua	7
圖 2.9	開啟埠號	8
圖 2.10	python 操控	8

表 目 錄

第一章 前言

1.1 檔案說明

共經歷了八個版本改版

前七個版本: bubbleRob 紀錄.7z

最終版: bubbleRob.7z

裡面包含了











 bubbleRob2.py	2023/4/14 下午 10:23	Python 來源檔案	3 KB
 my_ip.txt	2023/4/14 下午 10:06	文字文件	1 KB
 simpleTest_19997.py	2023/4/14 下午 09:13	Python 來源檔案	3 KB
 ip.bat	2023/4/14 下午 09:10	Windows 批次檔案	1 KB
 ip.py	2023/4/14 下午 08:15	Python 來源檔案	1 KB
 bubbleRob.py	2023/4/14 下午 08:00	Python 來源檔案	3 KB
 bubbleRob_scenes.ttt	2023/4/14 下午 07:57	TTT 檔案	656 KB
 __pycache__	2023/4/12 下午 05:21	檔案資料夾	
✓ 很久以前			
 remoteApi.dll	2021/10/8 下午 02:17	應用程式擴充	76 KB
 sim.py	2021/10/8 下午 02:17	Python 來源檔案	74 KB
 simConst.py	2019/11/12 下午 02:25	Python 來源檔案	43 KB

圖. 1.1: 遊戲檔案

點擊 ip.bat 就能直接獲取到目前電腦的 ipv4 位置

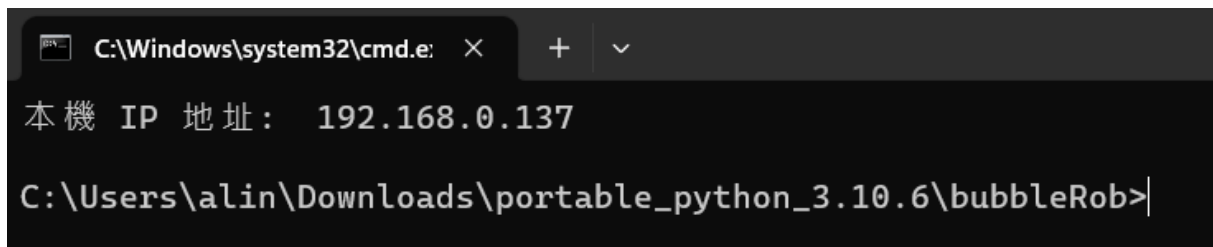


圖. 1.2: 目前電腦 ip

加入遊戲所在電腦 ip 只需輸入 ip (遊戲所在電腦 ip)

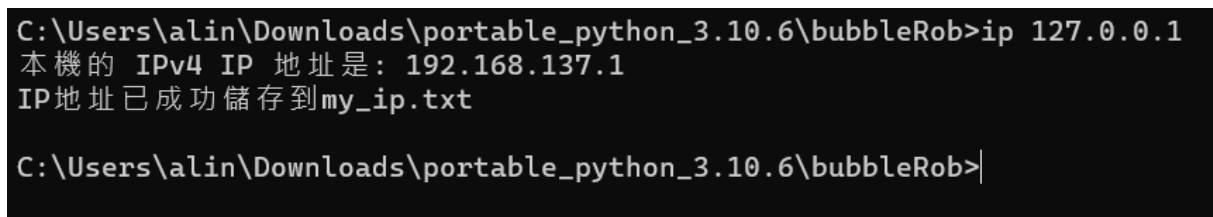


圖. 1.3: gameip

1.2 遊戲說明

開啟場景後便可以使用 19997 進行連線, 如果成功連線便會顯示 welcome to play

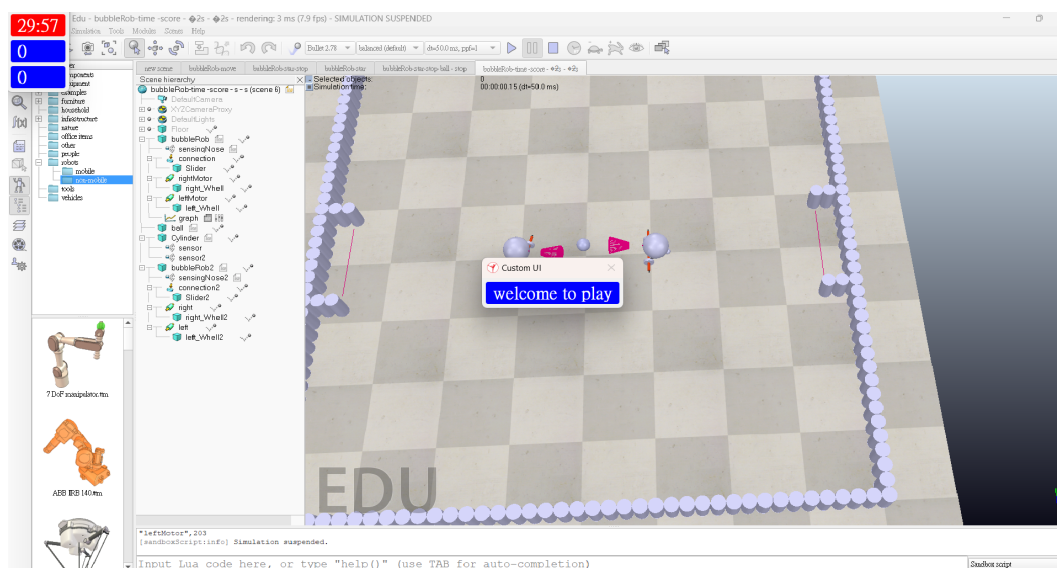


圖. 1.4: 歡迎

這時 23020 與 19998 埠號便會開啟, 第二位玩家便可利用 19998 加入遊戲, 加入成功便會立即開始, 觀戰者可以利用 23020 埠號進行觀戰

1.3 規則說明

遊戲規則簡單, 透過操控 bubbleRob 將球打入對方球門即得一分, 時間內其中一方分數高獲勝。

遊戲規則如下：

1. 利用 wasd 操控移動方向。
2. 將球打入敵方球框即得一分。
3. 一方得分後便會暫停遊戲並還原場地。
4. 按下 p 則繼續遊戲。
5. 時間內進球數多的一方獲勝。
6. 時間到即停止遊戲。

第二章 製作歷程

2.1 移動方式

在原有的設計上, 將自動移動改成了使用鍵盤操控
w 向上, a 向左, s 向下, d 向右
觸發不同條件來改變兩 joint 的速度達到變動方向的效果

```
function sysCall_actuation()
    local message, auxiliaryData, auxiliaryData2 = sim.getSimulatorMessage()
    result=sim.readProximitySensor(noseSensor) -- Read the proximity sensor
    -- If we detected something, we set the backward mode:
    if (message == sim.message_keypress) then
        if auxiliaryData[1] == string.byte("w") then ----w
            print("w")
            sim.setJointTargetVelocity(leftMotor, speed)
            sim.setJointTargetVelocity(rightMotor, speed)
        elseif auxiliaryData[1] == string.byte("s") then
            print("s")
            sim.setJointTargetVelocity(leftMotor, -speed)
            sim.setJointTargetVelocity(rightMotor, -speed)
        elseif auxiliaryData[1] == string.byte("a") then
            print("a")
            sim.setJointTargetVelocity(leftMotor, speed/4)
            sim.setJointTargetVelocity(rightMotor, speed)
        elseif auxiliaryData[1] == string.byte("d") then
            print("d")
            sim.setJointTargetVelocity(leftMotor, speed)
            sim.setJointTargetVelocity(rightMotor, speed/4)
        else
            sim.setJointTargetVelocity(leftMotor, 0)
            sim.setJointTargetVelocity(rightMotor, 0)
        end
    end
end
```

圖. 2.1: lua move

2.2 重製位置

添加了只要觸發特定條件就能使 bubbleRob 雙輪車回到原位
先記錄初始位置

```
initialBubbleRobPosition = sim.getObjectPosition(bubbleRobBase, -1)
initialBubbleRobOrientation = sim.getObjectOrientation(bubbleRobBase, -1)
initialNoseSensorPosition = sim.getObjectPosition(noseSensor, -1)
initialNoseSensorOrientation = sim.getObjectOrientation(noseSensor, -1)
```

圖. 2.2: 紀錄初始位置

在觸發特定條件時使用來達成重製效果

```
sim.setObjectPosition(bubbleRobBase, -1, initialBubbleRobPosition)
sim.setObjectOrientation(bubbleRobBase, -1, initialBubbleRobOrientation)
sim.setObjectPosition(noseSensor, -1, initialNoseSensorPosition)
sim.setObjectOrientation(noseSensor, -1, initialNoseSensorOrientation)
```

圖. 2.3: 重製位置

2.3 記分板與計時

增加倒數計時與分數的面板, 遊戲開始後開始倒數計時, 時間到則結束遊戲

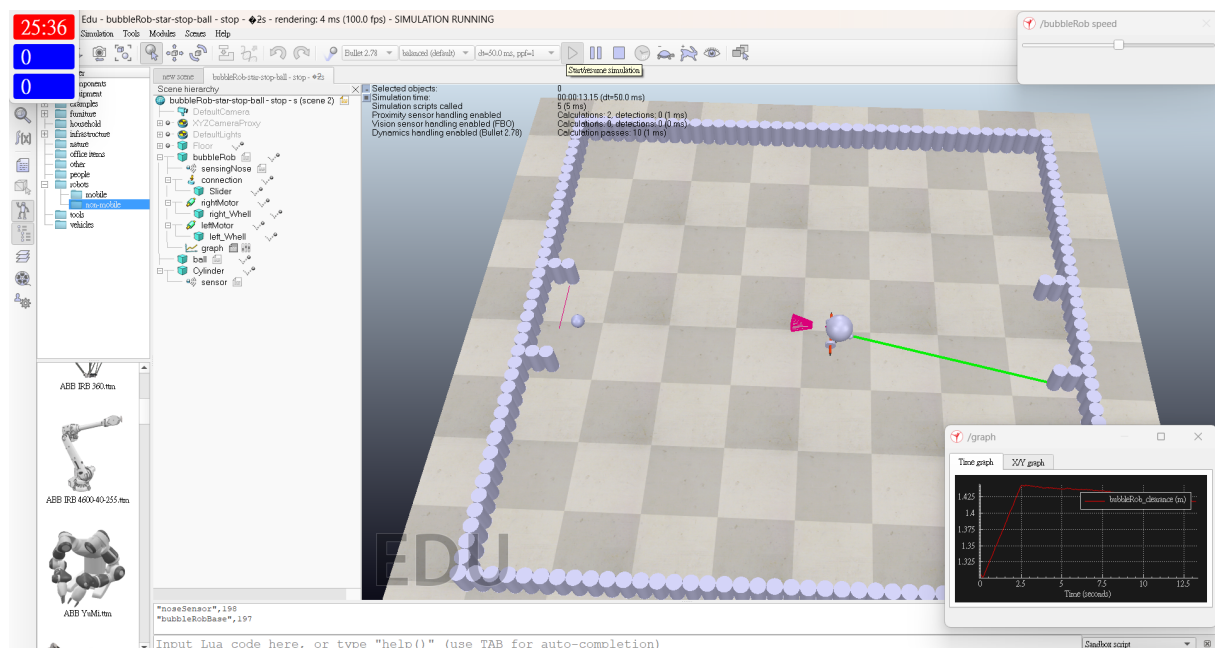


圖. 2.4: 記分板與計時

使用了 xml 進行編寫, 再利用 simUI.create 顯示出來

設定在不同的條件下, 更改顯示的數字

倒數結束時則利用 sim.stopSimulation() 來停止模擬

```
function sysCall init()
    count = 1800 -- 3300????????????
    score1 = 0 -- ?????
    score2 = 0
    xml = []
    <ui closable="false" resizable="false" activable="false">
        <label text="30:00" style="* (background-color: #F00; color: #FFF; font-size: 32px; font-weight: bold; padding: 4px; border-radius: 4px);">
        <label text="0" style="* (background-color: #00F; color: #FFF; font-size: 32px; font-weight: bold; padding: 4px; border-radius: 4px);">
        <label text="0" style="* (background-color: #00F; color: #FFF; font-size: 32px; font-weight: bold; padding: 4px; border-radius: 4px);">
    </ui>
    []
    ui = simUI.create(xml)
    simUI.setPosition(ui, 0,0, true)
end

function sysCall actuation()
    if count > 0 then
        count = count - 1
        local minutes = math.floor(count / 60)
        local seconds = count % 60
        local timeStr = string.format("%d:%02d", minutes, seconds)
        simUI.setLabelText(ui, 10, timeStr)
    else
        sim.stopSimulation()
    end
    --if count == 1700 then
        --sim.callScriptFunction(206, sim.scripttype_childscript, "updateScore", {1,2})
    --end
end
```

圖. 2.5: 記分板與計時程式

2.4 歡迎與恭喜

加入了歡迎與得分的 ui 提升觀賞性,一樣使用 xml 編寫即可使用同一個 id,只是更改顯示的內容,開始時隱藏,暫停時顯現

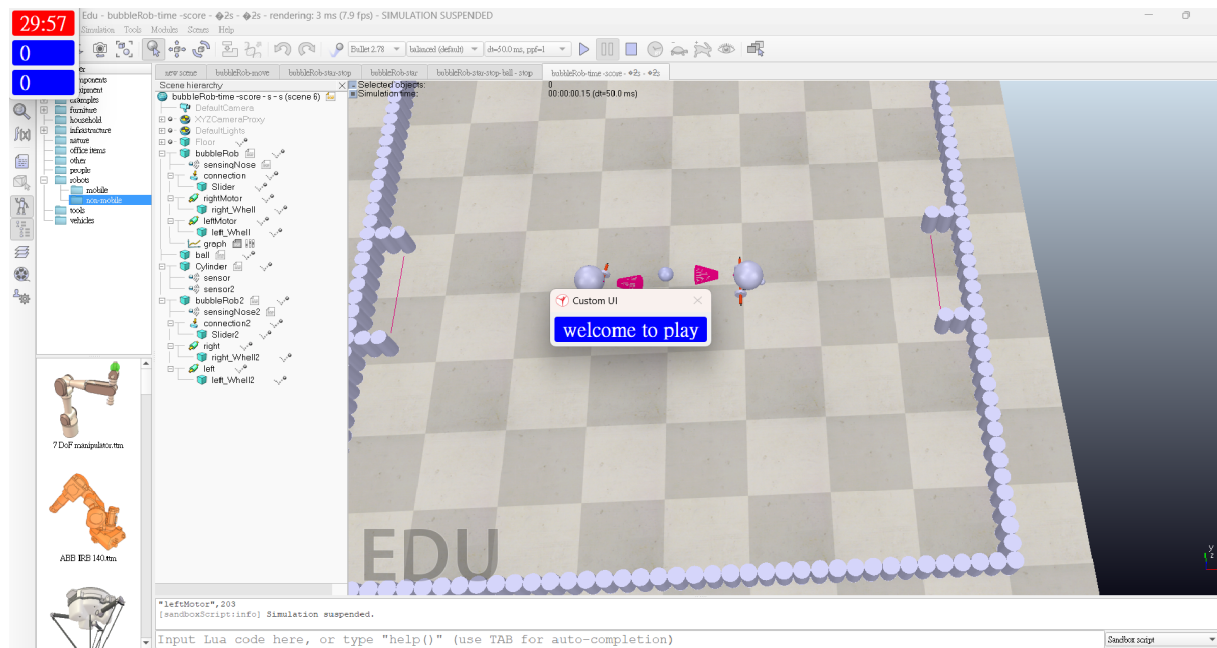


圖. 2.6: 歡迎

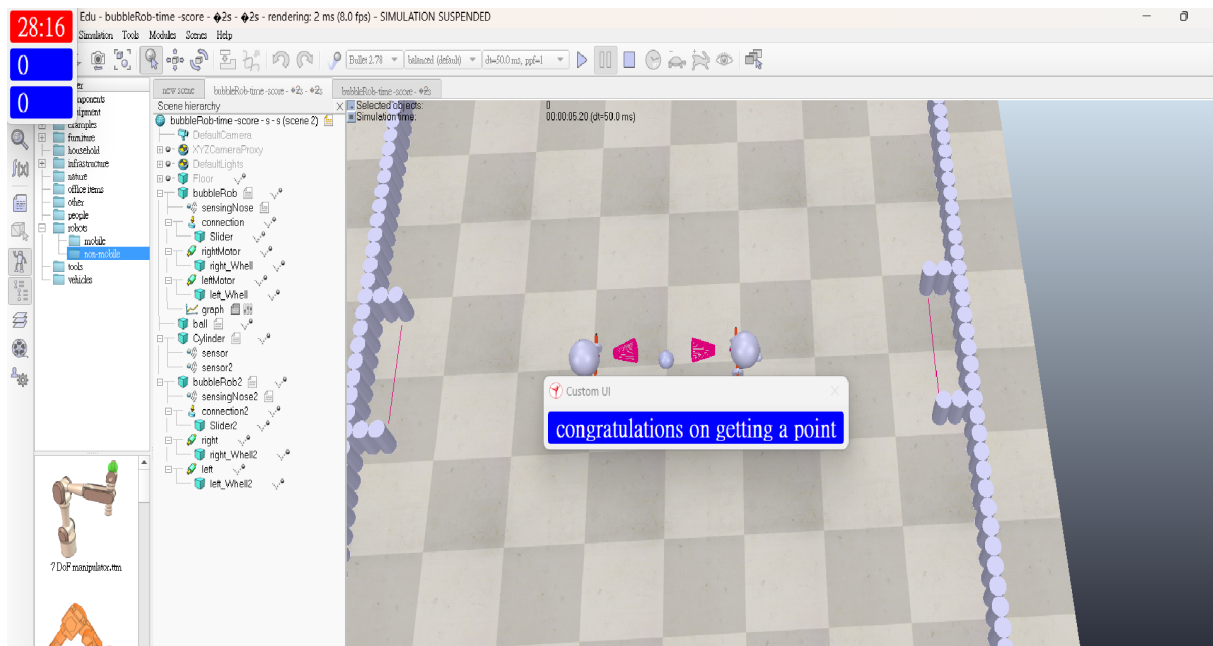


圖. 2.7: 恭喜

```
function sysCall_init()
    xml1 = {}
    ui1:closeable="false" resizeable="false" activatable="false"
    <ui class="GUISystem" style="background-color: #00FF00; color: #00FF00; font-size: 32px; font-weight: bold; padding: 4px; border-radius: 10px;">
    <label text="welcome to play" style="background-color: #00FF00; color: #00FF00; font-size: 32px; font-weight: bold; padding: 4px; border-radius: 10px;">
    </label>
    </ui>
    ui1 = simUI.create(xml1)
    add = true
    -- Pause simulation on the first run
    sim.pauseSimulation(true)
end

function sysCall_actuation()
    simUI.hide(ui1)
end

function sysCall_suspend()
    simUI.show(ui1)
    --simUI.setLabelText(ui1, 1, "good game")
    if add == false then
        simUI.setLabelText(ui1, 1, "congratulations on getting a point")
    end
    add = false
end
```

圖. 2.8: 歡迎與恭喜 lua

2.5 RemoteApi

開啟 19998 與 23020 埠號來提供玩家使用 RemoteApi 連線

將前面使用 lua 編寫的控制移動程式轉為使用 python 編寫, 並使用 RemoteApi 進行連線在模擬未開啟前, 可以使用 19997 默認埠號進行連線, 須注意連線時 coppeliasim 畫面必須開啟


```

1 function sysCall_init()
2 -- 場景模擬開始時開啟19998與23020埠號
3 simRemoteApi.start(19998)
4 simRemoteApi.start(23020)
5 end

```

圖. 2.9: 開啟埠號

```

1 #bubbleRob.py
2 import sim
3 import sys, math
4 import keyboard
5 import time
6
7 with open('my_ip.txt', 'r') as f:
8     my_ip1 = f.readlines()
9     for line in my_ip1:
10         if 'game' in line: #game #myip #local
11             ip = line.split(':')[1].strip()
12             print(ip)
13
14 # 連接到 CoppeliaSim simulation
15 sim.simxFinish(-1)
16 clientID = sim.simxStart(ip, 19997, True, True, 5000, 5)
17 sim.simxStartSimulation(clientID, sim.simx_opmode_oneshot_wait)
18
19 if clientID != -1:
20     print("已連接到遠端 CoppeliaSim 伺服器")
21 else:
22     print('連線失敗')
23     sys.exit('無法連接到 CoppeliaSim 伺服器')
24
25 # 取得遙控與感測器的 handles
26 errorCode, leftMotor = sim.simxGetObjectHandle(clientID, 'leftMotor', sim.simx_opmode_oneshot_wait)
27 errorCode, rightMotor = sim.simxGetObjectHandle(clientID, 'rightMotor', sim.simx_opmode_oneshot_wait)
28 errorCode, sensingNose = sim.simxGetObjectHandle(clientID, 'sensingNose', sim.simx_opmode_oneshot_wait)
29
30 # 設定一些參數
31 deg = math.pi/180
32 paused = False
33 if errorCode == -1:
34     print('找不到左右馬達')
35     sys.exit()
36
37 def jointspeed(left,right):
38     errorCode4=sim.simxSetJointTargetVelocity(clientID,leftMotor,left, sim.simx_opmode_oneshot)
39     errorCode5=sim.simxSetJointTargetVelocity(clientID,rightMotor,right, sim.simx_opmode_oneshot_wait)
40     errorCode, number2 = sim.simxLoadModel(clientID, 'number2.ttm', 0, sim.simx_opmode_oneshot_wait)
41 while sim.simxGetConnectionId(clientID) != -1:
42     event = keyboard.read_event()
43     if event.event_type == 'down':
44         print('The "' + event.name + '" key was pressed.')
45         if event.name == 'a' :
46             jointspeed(-3,5)
47         elif event.name == 'w' :
48             jointspeed(5,5)
49         elif event.name == 's' :
50             jointspeed(-5,-5)
51         elif event.name == 'd' :
52             jointspeed(5,-3)
53         if event.name == 'p':
54             if not paused:
55                 print('Paused')
56                 sim.simxPauseSimulation(clientID, sim.simx_opmode_oneshot_wait)
57                 paused = True
58                 time.sleep(0.1)
59             else:
60                 print('Resumed')
61                 sim.simxStartSimulation(clientID, sim.simx_opmode_oneshot_wait)
62                 paused = False
63                 time.sleep(0.1)

```

圖. 2.10: python 操控

第三章 缺陷與可改進點

1. 在 23020 埠號的觀戰者無法查看到分數與時間，
只能顯示在場景的主畫面上
2. 機器人的機體不適合推球
3. 遊戲結束後可以加入哪方勝利字樣
4. 計分可以拆分成兩方並標示 (藍方紅方等) 以便了解比分情況
5. 翻車時目前沒有自救空間
可以增加按鍵來達成將機器擺正
6. 按鍵指令只能觸發一個
造成動作不夠連貫，
希望能做到同時按兩個鍵，
將會有更高的操作性

第四章 心得

這次的專案從中學習到了 CoppeliaSim 的 remoteApi 實際運作方式
也對 CoppeliaSim 的 lua 與 python 的寫法有更深得了解
對於 CoppeliaSim 的句柄用法了解透徹能夠靈活運用
不過對於 sim.py 函式庫的使用依舊有加強空間
許多函式不熟悉只能使用替代方案執行
編寫程式時可以多利用 print
不管是除錯或者理解暫存的內容都好用
lua 的寫法與習慣的 python 略有不同
對於 latex 也有一定的認識
但還有需多不了解的地方依舊有待加強

相 關 資 料

[1] 2a-pj1ag4 小組網站

[2] bubbleRob 紀錄.7z

[3] 近端 tex 轉為 pdf

[4] apj1 統整網站

[5] bubbleRob.7z