國 立 虎 尾 科 技 大 學

機 械 設 計 工 程 系

cd2023 2a-pj1ag4 分組報告

網際手足球機器人場景設計

Web-based Robot Soccer Scene Design

指導教授： 嚴 家 銘 老 師

班 級： 四 設 二 甲

學 生： 洪 偉 陞 (41023146)

夏 進 源 (41023148)

紀 閔 翔 (41023147)

施 建 昌 (41023143)

李 承 翰 (41023121)

林 建 維 (41023134)

呂 佳 柔 (41023104)

王 啟 騰 (41023112)

中華民國 112 年 5 月

摘 要

本學期採取個人及團體分組來學習，團體實習目標為開發一款能在 web-based CoppeliaSim 場景中雙方或多方對玩的遊戲。pj3 為八人一組,，根據自選產品在期限內完成產品開發，在 w16 現場發表八人協同四週後所完成的產品，在 w17 各組採 OBS + Teams 以影片發表所完成的協同產品。

接續 pj2，各組須對雙輪車進行設計改良，以提升行進與對戰效率。採 CAD 進行場景與多輪車零組件設計後，轉入足球場景中以鍵盤 arrow keys 與 wasd 等按鍵進行控制，對陣雙方每組將有四名輪車球員，且每兩人在同一台電腦上操作，完成後各組須在分組網站中提供所有相關檔案下載連結，且提供線上分組簡報與分組 pdf 報告連結。

此專題是雙方利用各四台 BubbleRob 多輪車在一足球場景中進行對戰，雙方球門分別設有感測器。在規定時間內，每進一球即透過程式重與新往球場內隨機發球，接續賽局。模擬場景中還須配置 LED 計分板顯示比賽剩餘時間與比分，還需另外建立以機械轉盤傳動計分系統。在 CoppeliaSim 模擬環境中進行測試運用上的可行性並嘗試透過埠號供使用者觀看。

Abstract

This semester adopts both individual and group learning approaches. The group project involves developing a web-based game that allows two or more players to interact in a CoppeliaSim simulation environment. For Project 3 (pj3), the group consists of eight members who will collaborate to develop a product of their choice within a given deadline. In Week 16, the group will present the product they have developed over four weeks of collaboration. In Week 17, each group will use OBS + Teams to present their collaborative product through a video presentation.

Continuing from pj2, each group is required to design improvements for a two-wheeled vehicle to enhance its mobility and combat efficiency. Using CAD software, the groups will design the scene and components of the multi-wheel vehicle. The project will then transition to a soccer field scenario, where the vehicle will be controlled using keyboard arrow keys and WASD keys. Each group will have four vehicle players, with two players operating on the same computer. Upon completion, each group must provide download links for all relevant files on the group's website, as well as links to online group presentations and a PDF report.

This project involves a two-player battle using four BubbleRob multi-wheel vehicles in a soccer field scenario. Each player has their own goal equipped with sensors. Within a specified time frame, each goal scored triggers the program to reset and randomly kick off a new ball into the field, continuing the game. The simulation scene also includes an LED scoreboard to display the remaining time and score of the match. Additionally, a mechanical turntable-driven scoring system needs to be created. Feasibility testing and user observation will be conducted in the CoppeliaSim simulation environment, with the option for users to observe through port numbers.

目　錄

圖　目　錄

表　目　錄

第一章　前言

1.1 設計架構

此次 pj3 專題目標有建立場景中的計時器、球員外型及移動優化、添加球員擊球和翻車再起技能、進球後收集並隨機投下新的一顆球、建立以機械式轉盤傳動計分系統。由於目標繁多，需要組員間分工負責，在每個禮拜的協同中逐步完成 pj3 專題。

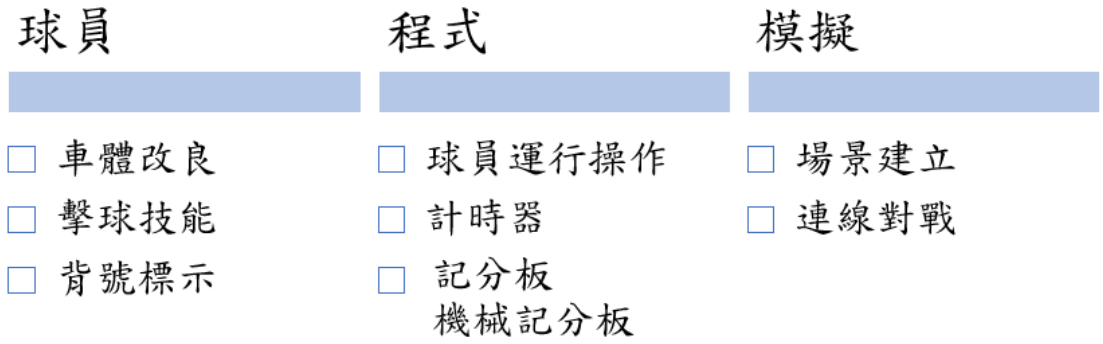

圖. 1.1: 設計目標圖

1.2 規則說明

類似於足球遊戲，一開始時球會置於場中央，遊戲開始後雙方即可以鍵盤操控機器人，透過防守敵方以及與隊友間的傳球推球至已方的球門得分。

遊戲規則如下：

1. 球觸碰到球門感測器即算得分。
2. 在十分鐘的鼻賽時間內，獲得最多分數的隊伍即獲勝。
3. 任一方進球得分後，隨機在場內投下新的球，雙方接續進行比賽。

第二章　球員製作過程

2.1 車體改良-運行

原本的車體為球形 bubbleRob，雖然造型簡單，卻會有容易翻車的問題。因此改為磚塊型，在前後添加兩顆輪子保持平衡。也將前進原理改為四輪驅動，使轉彎更為順暢合理。

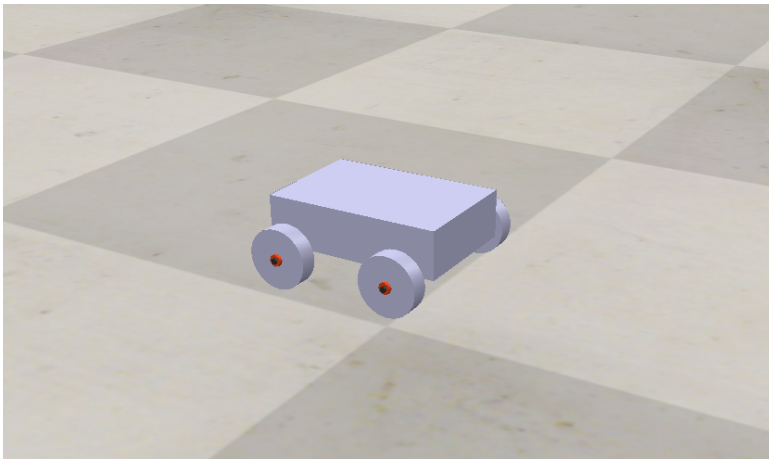

圖. 2.1: 磚塊型車體

2.2 車體改良-擊球

球員前端添加凸出的手部，球員本體在 CoppeliaSim 中用導入的開啟方式會產生抖動，因此改為加入物件 skin 並將本體隱藏。

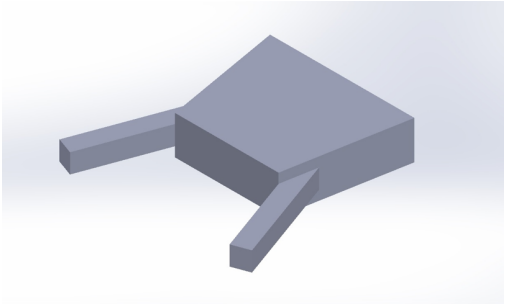

圖. 2.2: 球員 skin

2.3 車體改良-背號

由於兩隊各有四名球員，場上總共八名球員，為了能更清楚觀看及辨別球員，除了透過顏色區分隊伍，也需要讓每個球員添加背號。第一版的背號是直立式置於球員上方，實際遊玩時發現會有影響重心的問題。

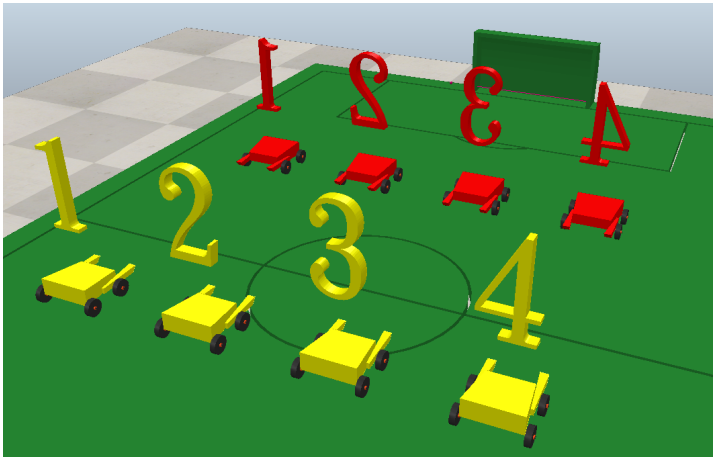

圖. 2.3: 第一版球員背號

於是第二版做了調整，參考實際賽車都將編號繪於車身，我們將背號改為平貼於車頂，解決了影響重心的問題。

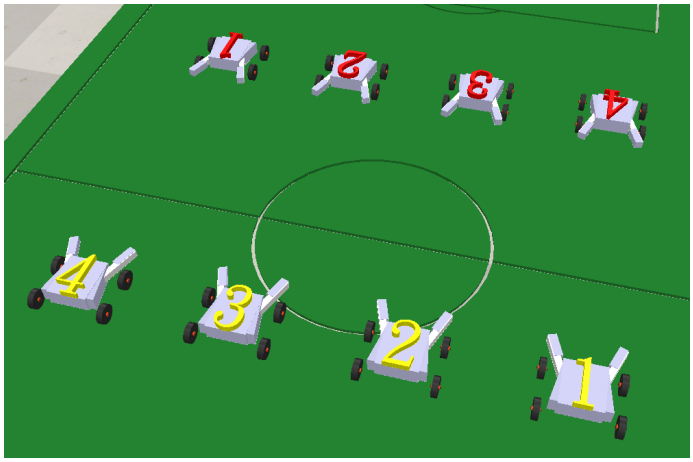

圖. 2.4: 第二版球員背號

第三章　球員程式碼

3.1 車體改良-操作

為了增加對戰的刺激性及操作的方便性，我們對球員的程式碼新增可以前後移動並同時左右移動，及添加倒地翻身的功能。

3.2 移動操作-輪子旋轉

```
def setBubbleRobVelocity(leftWheelVelocity1, rightWheelVelocity1,
leftWheelVelocity2, rightWheelVelocity2):
    leftMotor1 = sim.getObject('/a_player1/joint_lf')
    rightMotor1 = sim.getObject('/a_player1/joint_rf')
    leftMotor2 = sim.getObject('/a_player1/joint_lb')
    rightMotor2 = sim.getObject('/a_player1/joint_rb')
    sim.setJointTargetVelocity(leftMotor1, leftWheelVelocity1)
    sim.setJointTargetVelocity(rightMotor1, rightWheelVelocity1)
    sim.setJointTargetVelocity(leftMotor2, leftWheelVelocity2)
    sim.setJointTargetVelocity(rightMotor2, rightWheelVelocity2)
    #輸入四個變數分別給四個軸速度
```

圖. 3.1: 設定球員輪子旋轉

(圖.??) 設定一個 setVelocity 函數，接受四個參數：leftWheelVelocity1，rightWheelVelocity1，leftWheelVelocity2，rightWheelVelocity2，分別為左右前後輪速度。在函數內部使用 sim.getObject() 函數獲取左右前後輪的關節 joint ，並使用 sim.setJointTargetVelocity() 函數將各關節的目標速度設置為傳入的參數值，這樣做及可控制球員輪子速度，從而使球員移動。

3.3 移動操作-前輪方向

```
def setBubbleRobangel(a):
    brickRob= sim.getObject('/a_player1')
    angel = [-90*math.pi/180, a*math.pi/180, 0*math.pi/180]
    leftMotor = sim.getObject('/a_player1/joint_lf')
    rightMotor = sim.getObject('/a_player1/joint_rf')
    sim.setObjectOrientation(leftMotor, brickRob, angel)
    sim.setObjectOrientation(rightMotor, brickRob, angel)
    #輸入一個變數改變前輪方向
```

圖. 3.2: 設定球員前輪方向

(圖.??) 設定一個 setBubbleRobange1 函數，接受一個設置角度值得參數"a"，以函數 sim.getObject 獲取一個代表"a player" 模型的對象"brickRob"。計算角度值並傳入參數"a" 轉換為弧度值，將另兩個角度分別為固定角度值。在使用 sim.getObject() 獲取左右前輪的關節，以 sim.setObjectOrientation() 將關節地朝向設置為與 brickRob 相對應的角度，便可以給定角度旋轉並改變球員地朝向位置。

3.4 移動操作-左右轉向

```
def controlangel(y):
    if keyboard.is_pressed('a'):
        setangel(-y)
    elif keyboard.is_pressed('d'):
        setangel(y)
    else:
        setangel(0)
```

圖. 3.3: 控制球員左右轉向

(圖.??) controlange1 函數接受控制角度的函數"y"，檢查按鍵輸入，按下"a" 調用 setangel(-y) ，將傳入函數取相反值設成角度；按下"d" 則用 setangel(y)，傳入函數直接設成角度；無按下任何鍵則調用 setangel(0)，將角度設為零。

此程式碼根據按鍵輸入來控制角度值輸入，可以分別控制向左向右角度值。

3.5 移動操作-控制球員移動

```
def playercontrol(x,y):
    if keyboard.is_pressed('w'):
        setVelocity(x,x,x,x)
        controlangel(y)
    elif keyboard.is_pressed('s'):
        setVelocity(-x,-x,-x,-x)
        controlangel(y)
    elif keyboard.is_pressed('a'):
        setVelocity(-x,x,-x,x)
    elif keyboard.is_pressed('d'):
        setVelocity(x,-x,x,-x)
    elif keyboard.is_pressed('q'):
        # stop simulation
        sim.stopSimulation()
    else:
        setVelocity(0, 0, 0, 0)
        setangel(0)
```

圖. 3.4: wasd 移動球員

(圖.??) 以 playcontrol 函數定義參數"x" 和"y"，代表速度及角度。檢查按鍵輸入，如果按下"w" 將調用 setVelocity(x, x, x, x) 設置輪子速度、調用 controlangel(y) 控制角度。若為 setVelocity(-x, -x, -x, -x) 則設置為反向速度，按下"q" 則停止模擬。

根據鍵盤輸入 w, a, s, d 來控制球員前進、後退、左轉和右轉。

3.6 移動操作-維持速度

```
while True:
    if keyboard.is_pressed('shift'):
        playercontrol(v+4,a-20)
    else:
        playercontrol(v,a)
```

圖. 3.5: 控制球員速度

(圖.??) 此程式碼使用無限循環"while True" 迴圈，如果按下"shift" 會調用 playcontrol(v+4, a-20) 函數，使原有速度增加 4，角度減去 20。使用無線循環迴圈檢測鍵盤輸入，調整速度與角度。

3.7 球員改良-倒地翻身

```
elif keyboard.is_pressed('e'):
      floor= sim.getObject('/Floor')
      player = sim.getObject('/a_player1')
      a=sim.getObjectOrientation(player,floor)
      b=sim.getObjectPosition(player,floor)
      a[0]=0
      a[1]=0
      b[2]=b[2]+0.2
      sim.setObjectPosition(player,floor,b)
      sim.setObjectOrientation(player,floor,a)
```

圖. 3.6: 控制球員翻身

(圖.??) 定義如果按"e" 就執行，以 sim.getObject('/Floor') 獲取地板句柄、sim.getObject('/a player1') 獲取球員句柄。利用 sim.getObjectOrientation(player,floor) 獲取相對於地板的球員方向、sim.getObjectPosition(player,floor) 獲取相對於地板的球員位置，將數值存於變量"a" 及"b" 中。a[0]=0 將球員的 x 角度為 0，a[1]=0 將球員的 y 角度為 0，b[2]=b[2]+0.2 將球員的 z 位置上升 0.2。最後通過 sim.setObjectPosition(player,floor,b) 設定球員相對於地板的位置、sim.setObjectOrientation(player,floor,a) 設定球員相對於地板的方向。此段程式碼根據按下"e" 來執行將指定對象地朝向與位置進行修改，使球員翻倒後能夠再次翻身站起，繼續進行比賽。