

國 立 虎 尾 科 技 大 學

機 械 設 計 工 程 系

cd2023 2a3-pj3ag1 分組報告

網際足球對戰場景設計

Web-based Foosball Scene Design

指導教授： 嚴 家 銘 老 師

班 級： 四 設 二 甲

學 生： 王 樟 皓 (41023114)

吳 政 憲 (41023118)

呂 承 劍 (41023119) 組長

呂 昕 穎 (41023120)

李 彥 廷 (41023122)

李 茂 廷 (41023124)

卓 桓 琮 (41023126)

林 敬 燐 (41023138)

中華民國

112 年 6 月

目 錄

第一章 場景建立	1
1.1 前言	1
1.2 建立球員	1
1.3 建立記分板	2
1.4 建立球場	3

圖 目 錄

圖 1.1 球員建立	1
圖 1.2 球員建立	2
圖 1.3 記分板建立	2
圖 1.4 匯入記分板	3
圖 1.5 球場繪製	3
圖 1.6 建立球場	3

第一章 前言

1.1 規則

遊戲規則如下：

球打入敵方球門即得一分。

時間內進球數多的一方獲勝。

球進入球框後會回到原位

第二章 討論與分工

2.1 分工

5/18

1. 場地設置 41023118 41023138
2. 車子設計與組裝 41023122 41023124
3. 輪盤記分板繪製 41023126 41023114
4. 輪盤記分板程式 41023119 41023120
5. 車子控制程式與設置 41023119 41023120
6. 會議記錄 41023126
7. 整體組合 41023119 41023120

2.2 討論紀錄

5/22

- 41023124: 車子設計大略完成剩下背號
41023120: 程式研究中
41023118: 場地大致完成剩下球場線條
41023119: 機械式記分板程式正在編寫中球員程式正在編寫中
41023126: 機械記分板改第二版
41023138: 場地大致完成剩下球場線條
41023122: 球員除錯

5/29

- 41023122: 車子剩下背號
41023119: 球員程式正在修改中
41023138: 場景完成
41023120: 嘗試整體組合
和組長做模擬場景連線

第三章 場景建立

3.1 前言

老師有規定球場及球員的大小，重量

足球規格 (ball): 白色，直徑 0.1m，重量 0.5kg

足球場地 (field): 長 4m x 寬 2.5m

球門規格 (goal[0] and goal[1]): 長 0.6m, 高 0.3m, 寬 0.1m

球員尺寸範圍 (player[0]-player[7]): 長寬高各 0.2m, 重量 5kg。

3.2 建立球員

我們使用 CoppeliaSim 來製作車子。

後發現在移動左右轉彎時會分解，因此直接在 CoppeliaSim 內修改了定位，且在後背加上號碼。如 (圖.1.2)

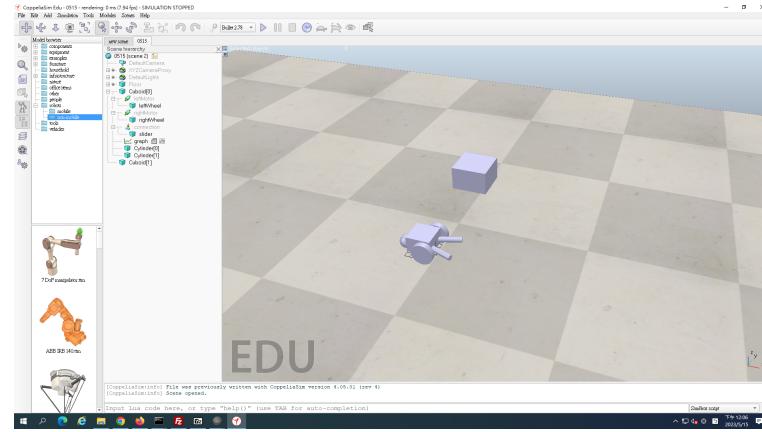


圖. 3.1: 球員建立

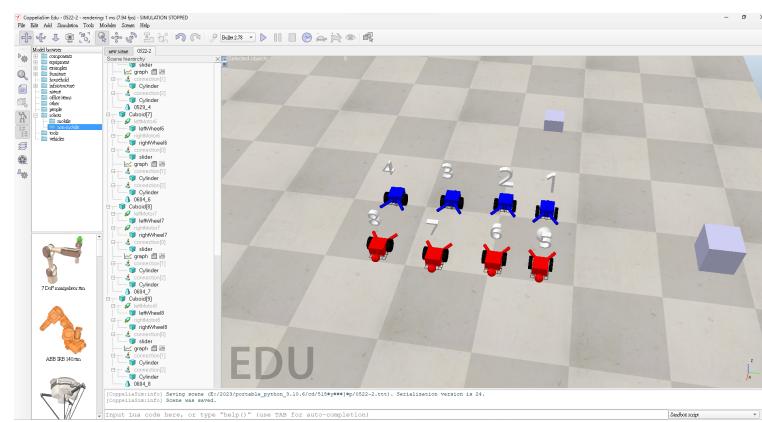


圖. 3.2: 球員建立 2

3.3 建立記分板

我們使用 Onshape 重新繪製了機械式記分板，如 (圖.1.3)

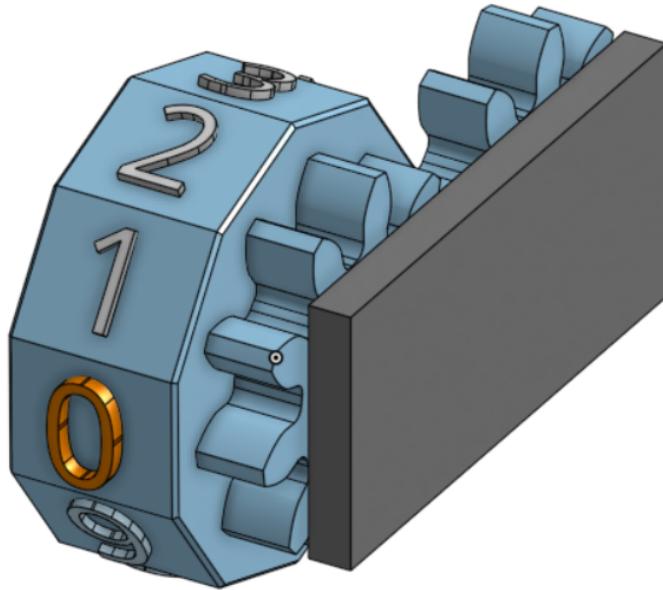


圖. 3.3: 記分板建立

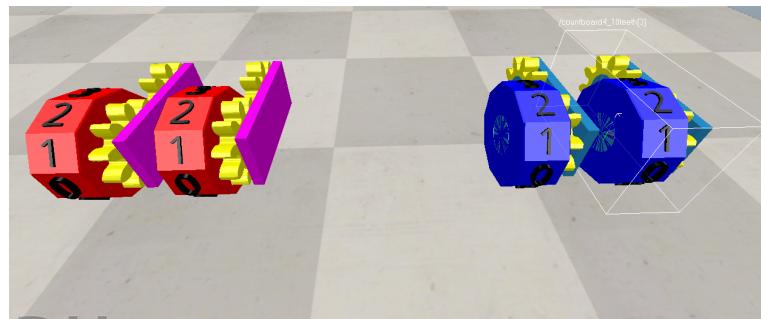


圖. 3.4: 匯入記分板

3.4 建立球場

我們使用 Onshape 繪製了球場底板及球門，如 (圖.1.5)，匯入 CopeliaSim 後接著建立感測器，如 (圖.1.6)。

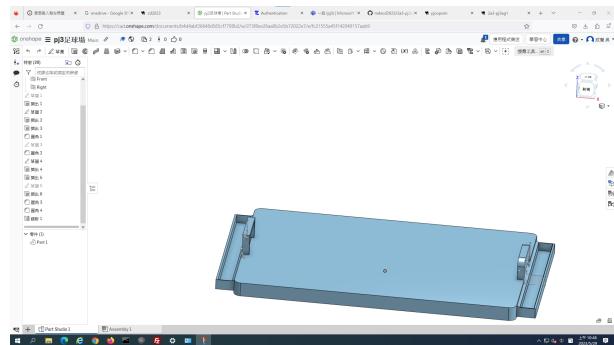


圖. 3.5: 球場繪製

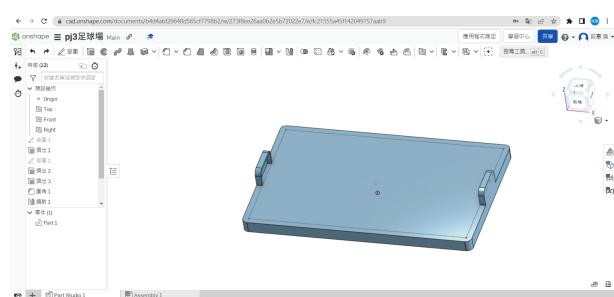


圖. 3.6: 建立球場

第四章 訓練環境

4.1 CoppeliaSimEdu 4.5.1

CoppeliaSim Edu 4.5.1 是一個虛擬機器人仿真軟體。它是由 Coppelia Robotics 開發的，旨在提供一個仿真環境，讓使用者可以設計、開發和測試各種機器人系統。具有豐富的功能和工具，可以模擬和仿真各種機器人的運動、感知和控制。它提供了一個圖形化的界面，讓使用者可以直觀地建立機器人模型、設計場景、添加感測器和行為，並進行仿真和測試。並且支援多種程式語言，包括 C++、Python、MATLAB 等，使用者可以選擇自己熟悉的語言來編寫控制和行為代碼。它還提供了各種感測器和執行器的模型，包括視覺感測器、力覺感測器、輪子和關節等，用戶可以根據自己的需求選擇和配置這些模型。它還支援 ROS (Robot Operating System)，這是一個廣泛使用的機器人開發框架。使用者可以將 CoppeliaSim Edu 和 ROS 結合使用，實現更複雜的機器人系統開發和測試。整體來說，CoppeliaSim Edu 4.5.1 是一個功能豐富的虛擬機器人仿真軟體，它提供了一個強大的環境，讓使用者可以進行機器人系統的設計、開發和測試，並支援多種程式語言和 ROS，使得機器人開發更加靈活和便利。



Coppell
from the creato

圖. 4.1: ATARI Pong

第五章 模擬環境

5.1 模擬模型

在模擬的模型上，延用了學長設計的冰球機，並進行了部分的設計變更，將原本的人機對打更改為機器對打，且因為搭配深度強化學習的訓練，所以將兩邊的擊球器都僅保留 X 軸向（左右）移動，而冰球則是使用原本設計。多虧了學長們所設計的冰球機模型，讓我們在運作上有問題時可以直接發問，設計變更的地方也可以快速完成。

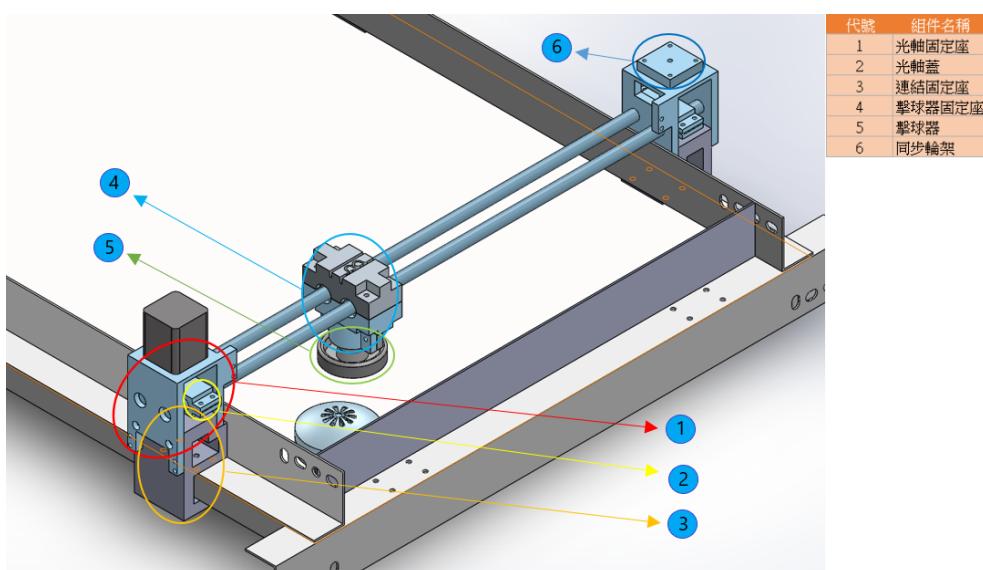


圖. 5.1: 組合圖

將原本 Y 軸移動機構移除，並將其改為固定在特定位置上，此固定座設計是取代原本鎖在光軸固定座上的（圖.?? 代號 1)Y 軸皮帶固定座（圖.??），並使光軸固定座可以通過連結固定做鎖固於桌面，如（圖.??）。

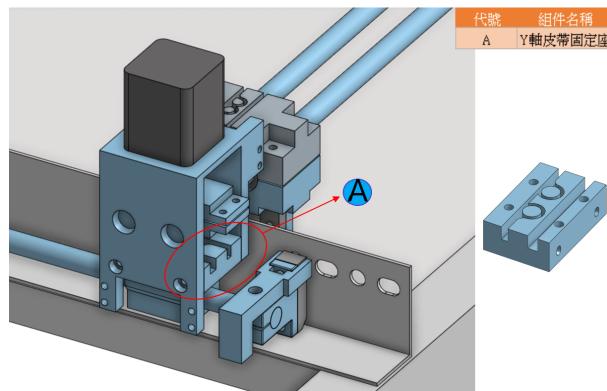


圖. 5.2: Y 軸皮帶固定座

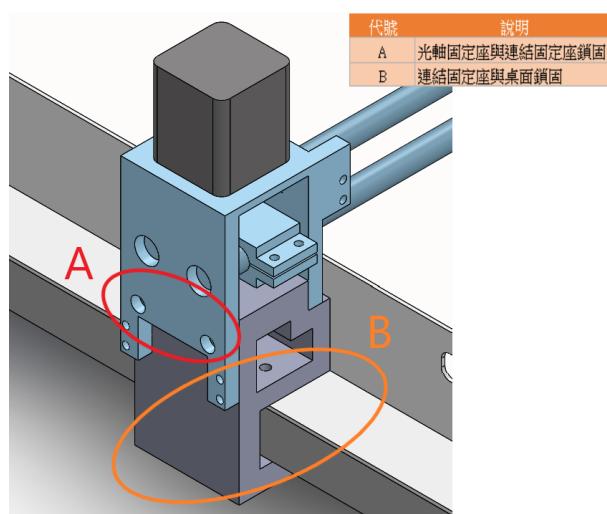


圖. 5.3: 連結固定座

分別在擊球器外側保留約冰球直徑 1.5 倍之區域作為得分判定區，如圖 4.4 中的紅色區域。

5.2 CoppeliaSim 模擬

CoppeliaSim 是具有集成開發環境的機器人模擬器，基於分佈式控制體系架構，可以通過嵌入式腳本，插件，ROS 或 BlueZero 節點，RemoteAPI 客戶端或自定義解決方案進行模型控制。



圖. 5.4: CoppeliaSim Logo

且 CoppeliaSim 中，控制器可以用 C / C ++、Python、Java、Lua、Matlab 或 Octave 編寫。

5.2.1 使用原因

本專題之最終目標是希望可以在虛擬環境中進行深度強化學習來訓練機器對打，通過虛擬環境中的模擬後，可以更直接地看到深度強化學習訓練的狀況，且因為在虛擬環境中不會有金費的支出，所以可以不斷的重複模擬直到模擬達到最佳的狀態，除此之外 CoppeliaSim 的虛擬環境更接近真實環境，基於以上原因，所以使用了 CoppeliaSim 開發。

5.2.2 RemoteAPI

RemoteAPI(Remote Application Programming Interface) 是 CoppeliaSim API 框架的一部分。它允許 CoppeliaSim 與外部應用程序之間的通訊，

是跨平台並支持服務調用和雙向數據流。有兩個不同的版本/框架分別為:Remote API 和 The B0-based remote API。

5.2.3 常用功能

1. 以下為簡易功能說明:



圖. 5.5: CoppeliaSim 工具列



圖. 5.6: CoppeliaSim 工具列 (續)

代號	功能說明	代號	功能說明
1	畫面平移	10	複製所有設定
2	畫面旋轉	11	回復/取消回復
3	畫面縮放	12	模擬設定
4	畫面視角	13	開始/暫停/停止模擬
5	畫面縮放至適當大小	14	即時模擬切換
6	選取物件	15	模擬速度控制
7	移動物件	16	線程渲染/視覺化
8	旋轉物件	17	場景/頁面選擇
9	加入/移出樹狀結構		

表. 5.1: 功能說明

5.3 影像處理

在影像處理中我們主要使用了 Python 套件中的 OpenCV(全稱:Open Source Computer Vision Library), 並搭配其他套件或模組進行了影像處理, 藉此來取得訓練神經網路訓練時所需的資訊。



圖. 5.7: OpenCV 及 Python logo

5.3.1 CoppeliaSim 中的 Vision sensor(視覺傳感器)

CoppeliaSim 的視覺傳感器輸出的影像是以每個像素中以 RGB 三個位元組所組成的，舉例來說: 在 CoppeliaSim 中視覺傳感器取出畫面像素為 $512*256$ ，則我們會接收到 $(512*256) \text{ 像素} * 3 = 393,216$ 個資料，是一筆相當大的資料，所以在影像處理上會消耗掉大量的資源。

5.3.2 影像辨識

透過 CoppeliaSim 中的 Vision sensor 接收場景影像並輸出後，便可以開始進行影像辨識的處理。

1. RGB 與 HSV 的轉換 [??]

RGB 即光的三原色 Red(紅)Green(綠)Blue(藍)，HSV 則是一種將 RGB 色彩模型中的點在圓柱坐標系中的表示法，HSV 分別表示 Hue(色相)、Saturation(飽和度)、Value(明度)，而會將 RGB 轉換為 HSV 是因為 HSV 相較於 RGB 可以更直接的判斷色彩、明暗和鮮豔度對於顏色過濾可以更方便定義出色彩範圍。

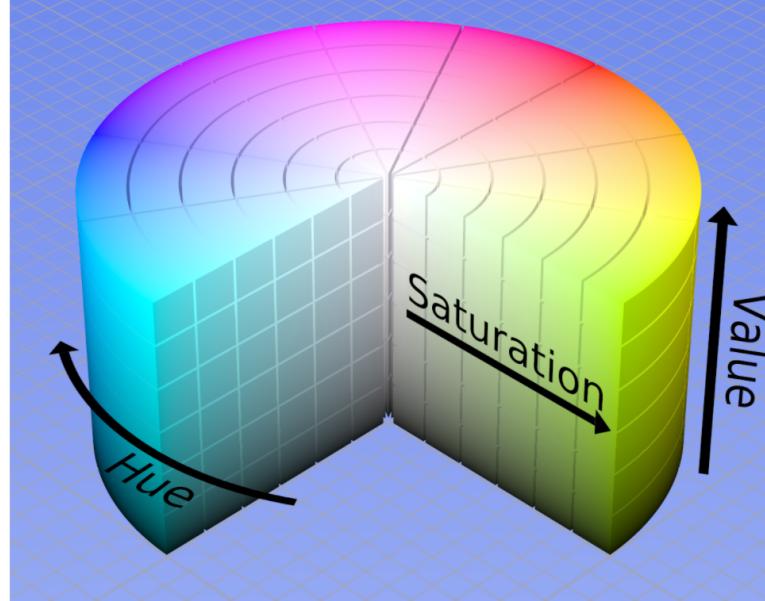


圖. 5.8: HSV 色彩空間

下列為 RGB 與 HSV 之間轉換的公式，首先是 RGB 轉為 HSV，其中 \max 及 \min 分別為 (r, g, b) 中的最大與最小值：

$$h = \begin{cases} 0^\circ, & \text{if } \max = \min \\ 60^\circ + \frac{g-b}{\max-\min} + 0^\circ, & \text{if } \max = r \text{ and } g \geq b \\ 60^\circ + \frac{g-b}{\max-\min} + 360^\circ, & \text{if } \max = r \text{ and } g < b \\ 60^\circ + \frac{g-b}{\max-\min} + 120^\circ, & \text{if } \max = g \\ 60^\circ + \frac{g-b}{\max-\min} + 240^\circ, & \text{if } \max = b \end{cases}$$

$$s = \begin{cases} 0, & \text{if } \max = 0 \\ \frac{\max-\min}{\max} = 1 - \frac{\min}{\max}, & \text{otherwise} \end{cases}$$

$$v = \max$$

接著是 HSV 轉為 RGB:

$$\text{when } 0 \leq H < 360, 0 \leq S \leq 1, 0 \leq V \leq 1$$

$$C = V \times S$$

$$X = C \times (1 - |(H/60^\circ)\bmod 2 - 1|)$$

$$m = V - C$$

$$(R', G', B') = \begin{cases} (C, X, 0) & , 0^\circ \leq H < 60^\circ \\ (X, C, 0) & , 60^\circ \leq H < 120^\circ \\ (0, C, X) & , 120^\circ \leq H < 180^\circ \\ (0, X, C) & , 180^\circ \leq H < 240^\circ \\ (X, 0, C) & , 240^\circ \leq H < 300^\circ \\ (C, 0, X) & , 300^\circ \leq H < 360^\circ \end{cases}$$

$$(R, G, B) = ((R' + m) \times 255, (G' + m) \times 255, (B' + m) \times 255)$$

2. 顏色過濾

進行顏色過濾時，需要先定義出過濾顏色的上下限，在開始過濾後僅會保留介於上下界線範圍的影像，而介於上下限範圍之外的影像則會被剔除，如圖.??所示以上限 (77, 255, 255) 及下限 (35, 43, 46) 為例。

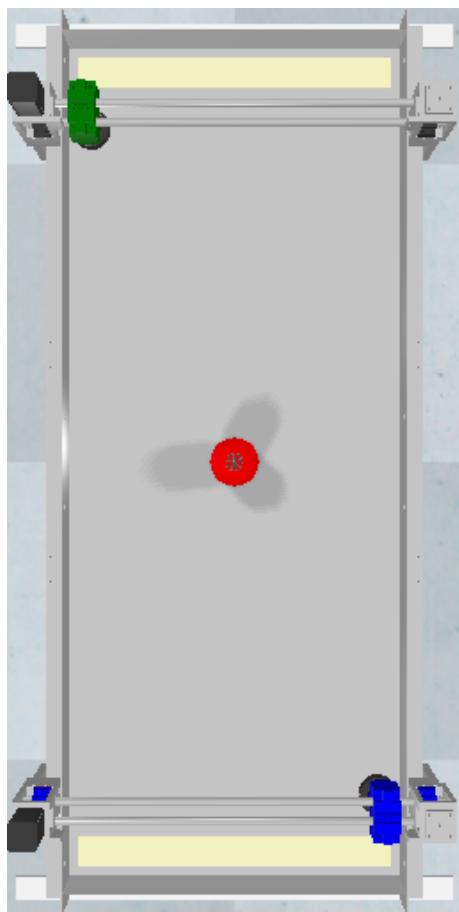


圖. 5.9: 場景原圖

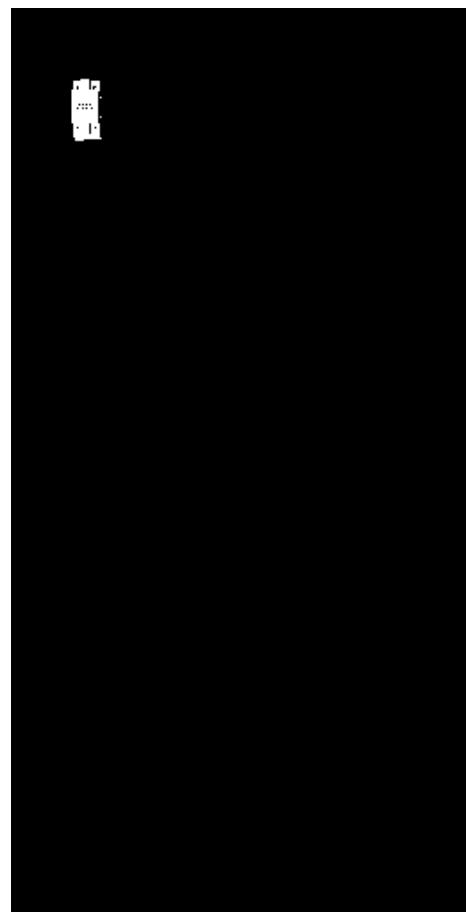


圖. 5.10: 顏色過濾後的場景