

Shootcar-Project

投球機期末專案

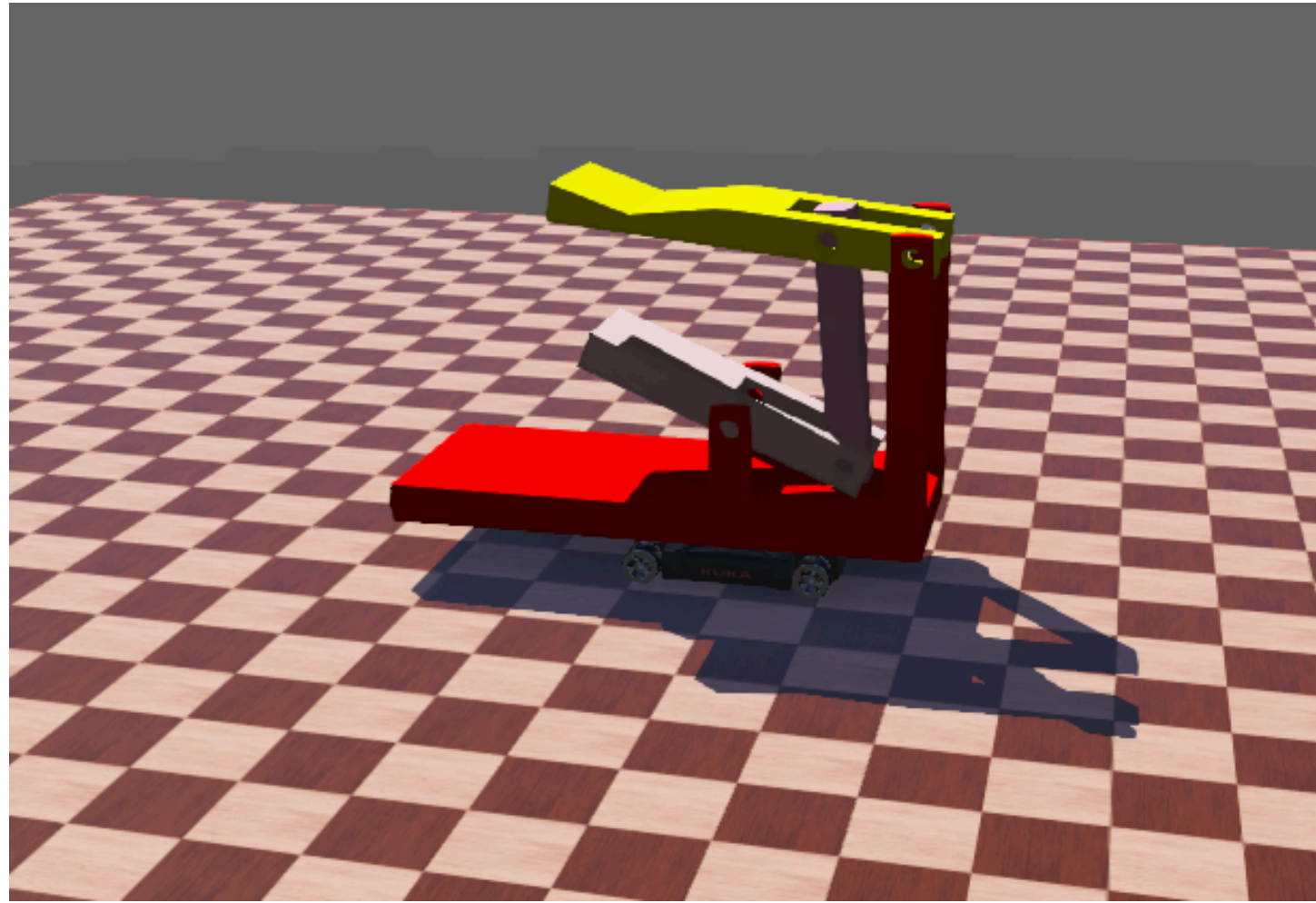
組長:41223215 吳承睿

組員: 41223206 陳顥亘 、41223226 林堃燁 、41223227 施宗廷、 41223228 洪英毓、41223235 陳脩升

本專案結合人工智慧、團隊協作及模擬技術，致力於開發一款能自動投籃的機器裝置。過程中運用繪圖軟體設計主要零件與整體結構，並透過 Webots 平台進行模擬與功能驗證。專案前期學習與實作涵蓋以下核心技術與模組：

- 1.四連桿機構之設計與運動模擬分析。**
- 2.投籃機主體零件的繪製、組裝與基本投籃動作的模擬。**
- 3.七段顯示器的設計與控制，並投球的動態顯示模擬以及加分。**
- 4. 搭載式底座車輛的建模與移動模擬。**

搭載式底座車輛的建模



實體模型都是使用.obj檔

組裝在不同的 Solid 節點中，並透過 HingeJoint + RotationalMotor 控制動作。

車子主體使用外層Robot+best製作

籃球基本體是用內層Robot+HingeJoint+ Mesh製作
所有結構都建立在 Robot 內部的 children 中，不同的部位靠著正確的 anchor、translation 和 HingeJoint 組合成一個整體，所有部位最後都繫在 base 上，所以整體能跟著車子移動。

```
1 Robot {  
2   children [  
3     Robot { // <-- 這裡嵌套了一個子機器人，裡面才是重點！  
4       ...  
5     }  
6   ]  
7 }
```

外層Robot是一個總控制單位

內層的Robot投籃機構、馬達等

```
1 Solid {  
2   name "base"  
3   geometry Mesh {  
4     url [ "models/base.obj" ]  
5   }
```

為車子的主體底座，零件都要接在此平台上

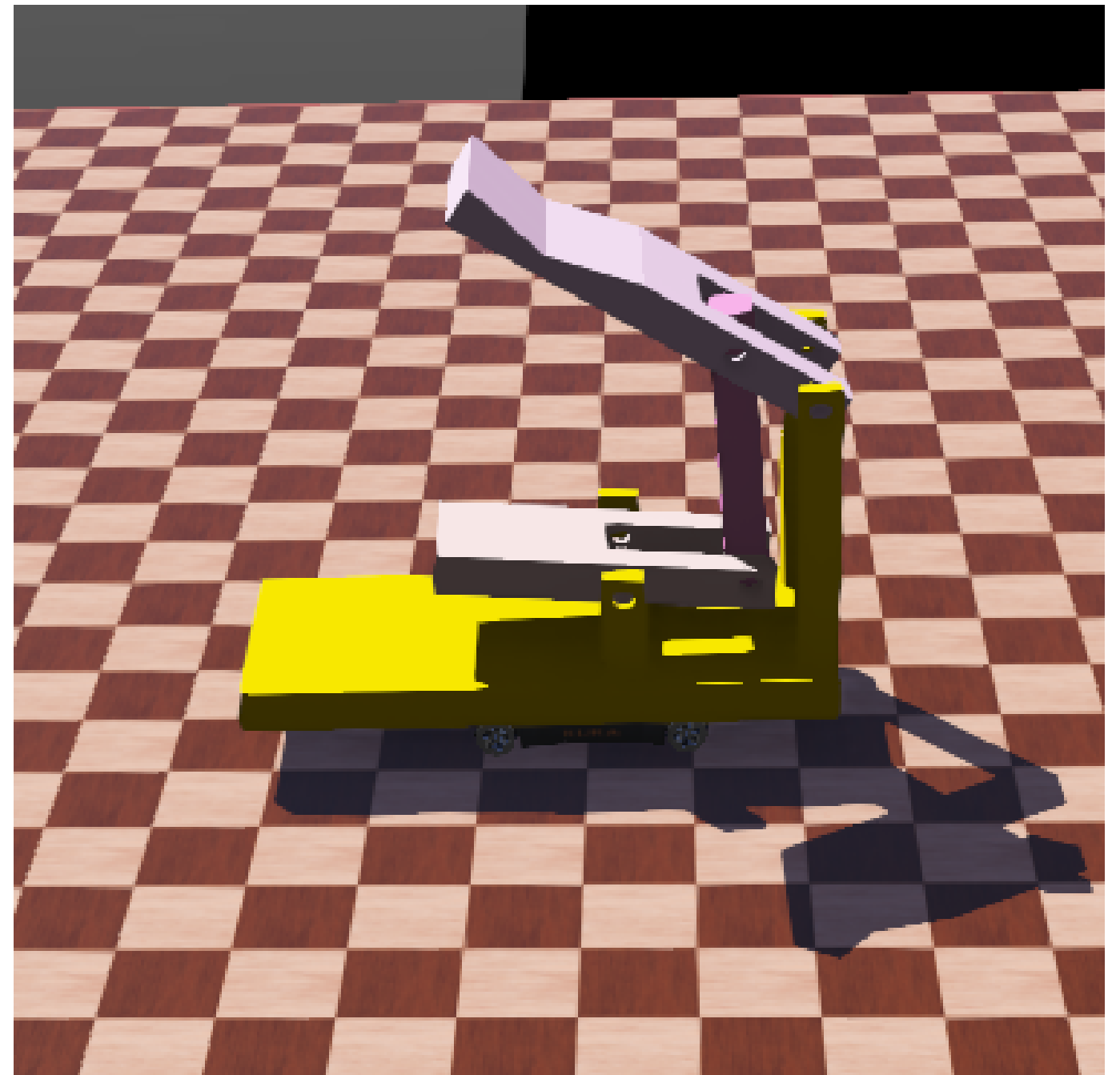
```
1 HingeJoint {  
2   jointParameters HingeJointParameters {  
3     position 0.0031144628241826593  
4     axis 0 0 1  
5     anchor 0.15 0.68 0  
6   }
```

中間使用許多的HingeJoint來進行結構接，使其能夠作動

投籃車作動機制

- 車子底盤 (part_1.obj)
 - └─ HingeJoint (motor1)
- └─ 第一段連桿 (part_2.obj)
 - └─ HingeJoint
- └─ 第二段連桿 (part_3.obj)
 - └─ HingeJoint
- └─ 第三段連桿 (part_4.obj) = shooter

自訂機器人主體的基底，車輪與其綁定，也就是 youbot 的主結構。被定義在 Solid 中，名稱為 "base"。

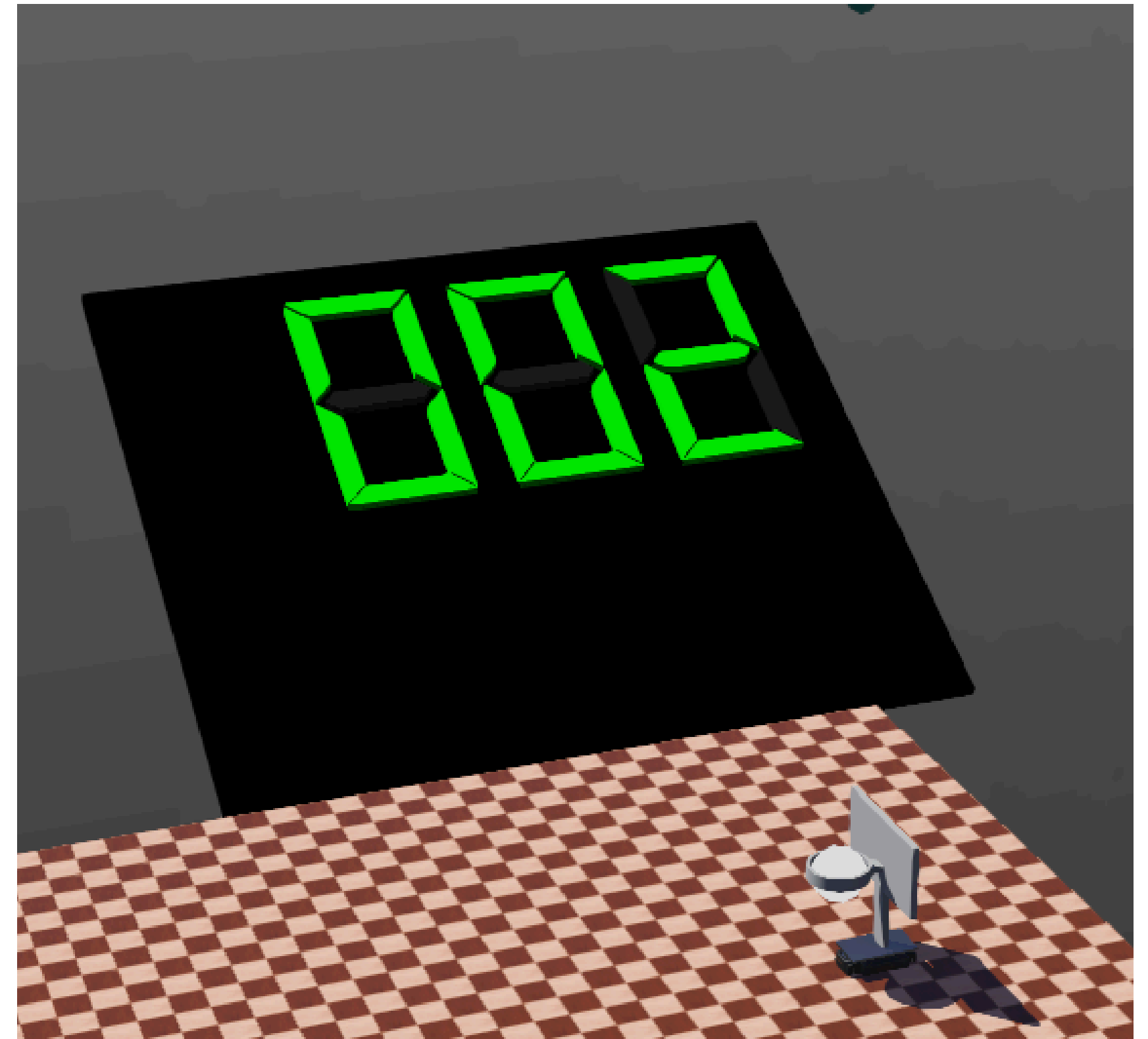
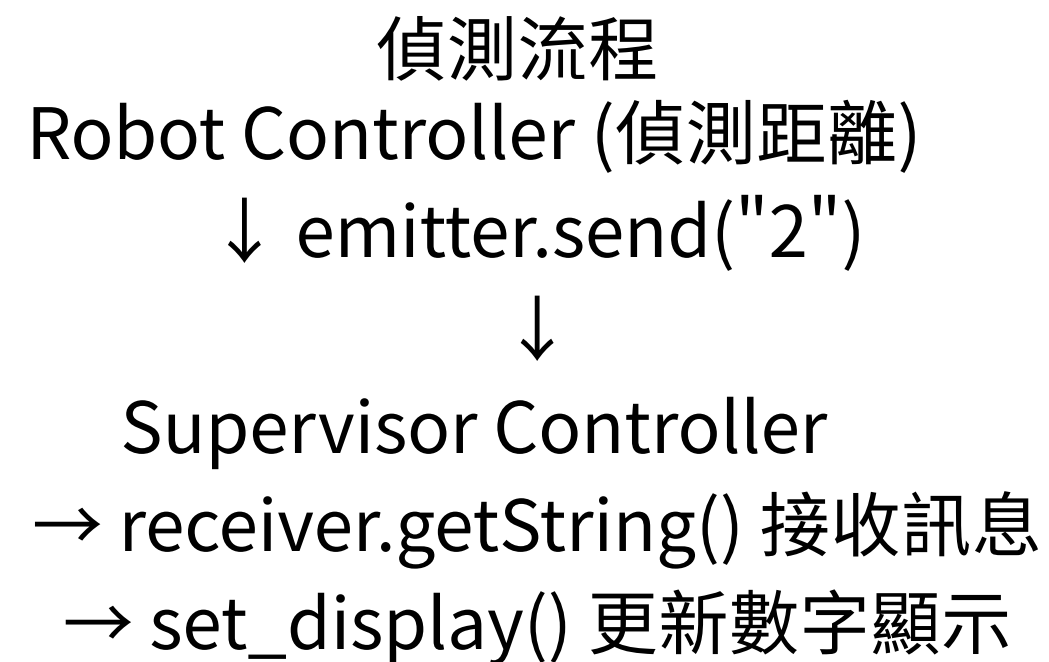


七段顯示器與籃筐偵測加分

```
if distance < 0.19 and (current_time - last_score_time) > cooldown:  
    score += 2  
    last_score_time = current_time  
    print("得分 +2")  
    print("Current Distance:", distance)  
    emitter.send(str(2)) # 送出得分
```

當距離感測器偵測到球進籃框時，會執行 emitter.send(...) 傳送分數（例如 "2"）。

要注意的是在感測器的channel 一定要相同，這是Receiver和Emitter可以通訊的主要原因



控制發球機構 餵球機制

```
51 def youbot_local_to_world(local_pos):
52     youbot_node = supervisor.getFromDef('youbot')
53     if youbot_node is None:
54         raise RuntimeError("找不到 DEF 為 youbot 的 Robot 物件")
55     youbot_translation = np.array(youbot_node.getField('translation').
56     youbot_rotation = youbot_node.getField('rotation').getSFRotation()
57     youbot_axis = youbot_rotation[:3]
58     youbot_angle = youbot_rotation[3]
59     youbot_rot_mat = axis_angle_to_rotation_matrix(youbot_axis, youbot
60     rotated = youbot_rot_mat @ np.array(local_pos)
61     world_pos = youbot_translation + rotated
62     return tuple(world_pos)

158 def create_static_sphere(supervisor, x, y, z):
159     global waiting_ball_def, waiting_ball_info
160     def_name = generate_valid_def_name()
161     waiting_ball_def = def_name
162     r, g, b = generate_random_color()
163     world_pos = youbot_local_to_world((x, y, z))
164     waiting_ball_info = (world_pos, r, g, b)
165     create_static_ball(def_name, world_pos, r, g, b)
```

create_static_sphere() 函式，於指定位置產生一顆靜止球，等待擊出。 靜態球生成（按下 A 鍵）

activate_dynamic_ball() 將靜止球轉為具有物理屬性的動態球，啟動模擬物理運動。 球體動態化（按下 M 鍵）

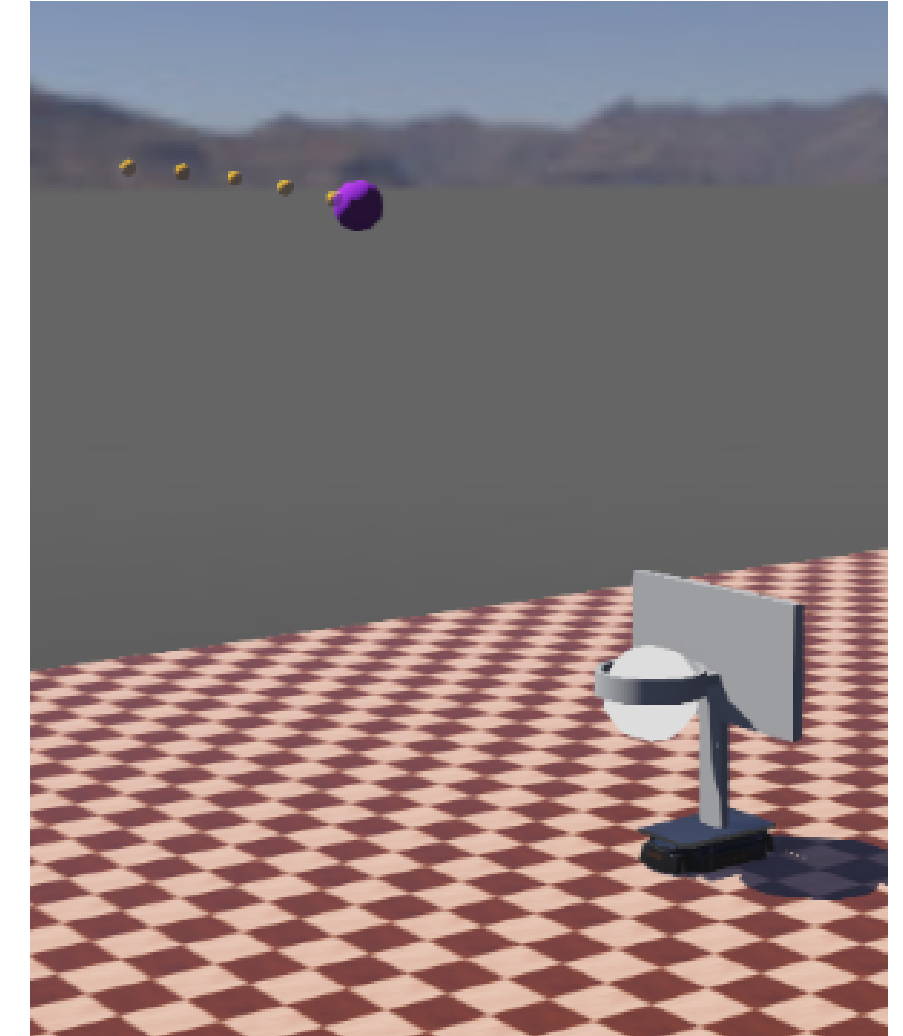
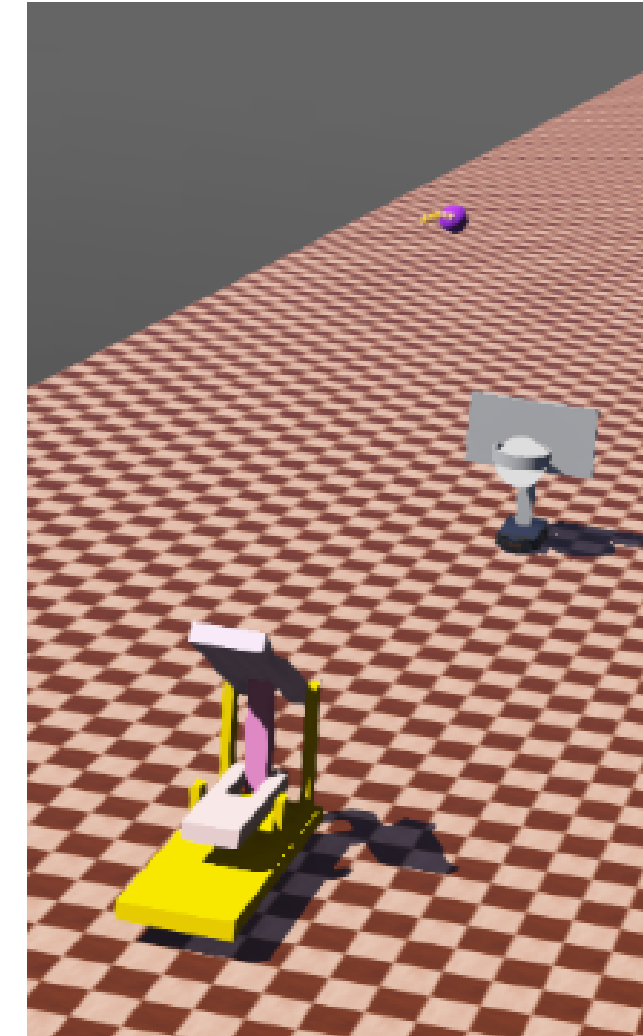
create_trajectory_point() 建立拋物線軌跡點，以透明橘色球標示投射路徑。 控制最多保留五個軌跡點（FIFO 機制），避免畫面過於擁擠。

當球接近地面 ($z < 0.13$) 自動視為落地，並清除所有軌跡點。 軌跡點追蹤顯示

youbot_local_to_world() 將投籃機相對位置轉換為世界座標，使球出現於正確位置。 世界座標轉換

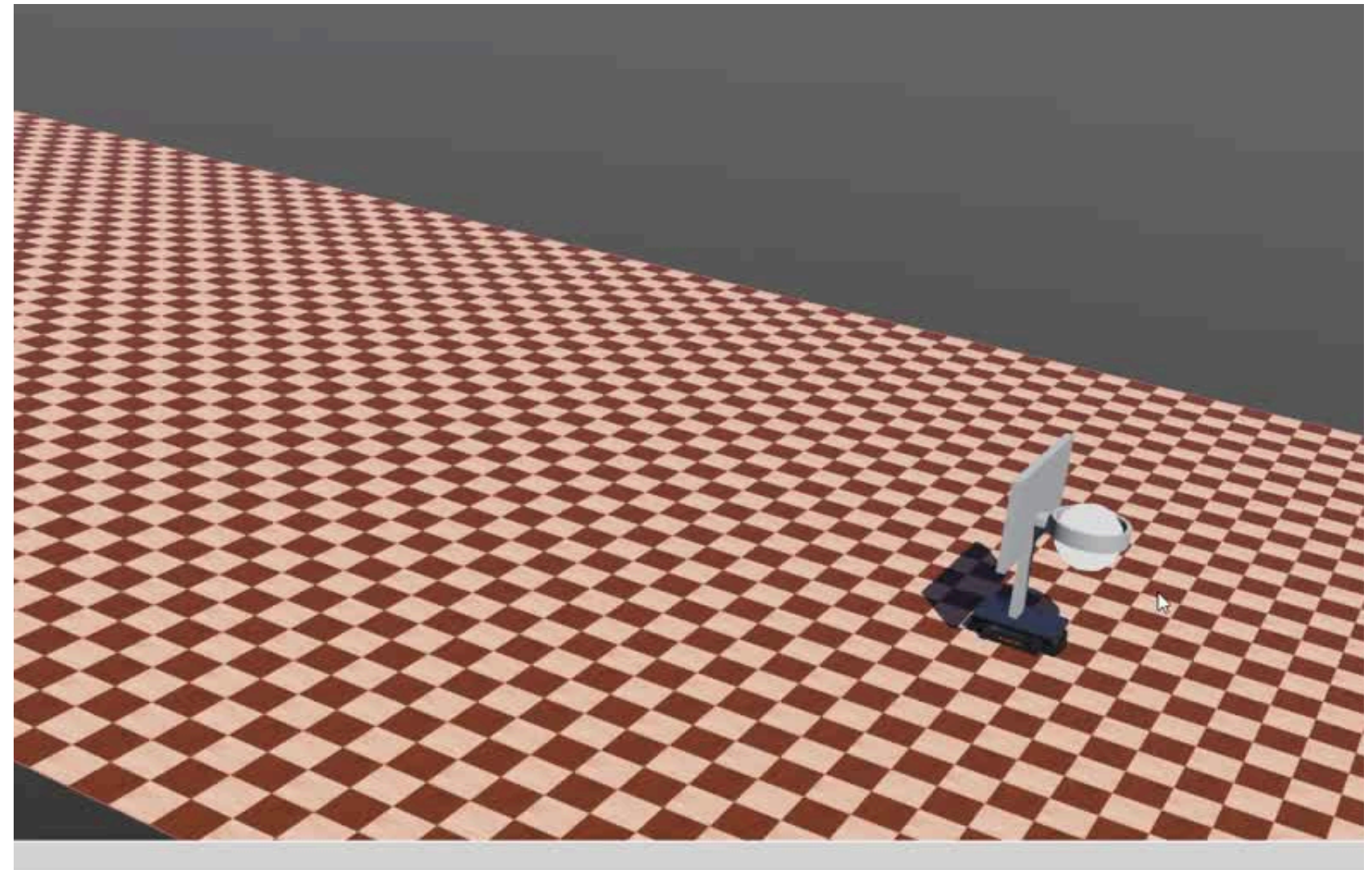
create_static_ball() 產生不含物理屬性的靜態球。

create_dynamic_ball() 則附加質量與密度，進行真實模擬。球體外觀與物理設定



籃框移動作動及偵測系統

```
1  from controller import Robot, Keyboard, Emitter
2
3  # Constants
4  WHEEL_RADIUS = 0.1 # Radius of the wheels in meters
5  L = 0.471 # Half of the robot's length
6  W = 0.376 # Half of the robot's width
7  MAX_VELOCITY = 10.0 # Maximum wheel velocity
8
9  # Initialize robot and keyboard
10 robot = Robot()
11 timestep = int(robot.getBasicTimeStep())
12 keyboard = Keyboard()
13 keyboard.enable(timestep)
14
15 # Get motors
26 wheel5 = robot.getDevice("wheel5") # Front-right
27 wheel6 = robot.getDevice("wheel6") # Front-left
28 wheel7 = robot.getDevice("wheel7") # Rear-right
29 wheel8 = robot.getDevice("wheel8") # Rear-left
30
31
78 if ord('M') in keys or ord('m') in keys:
79     print("Distance (M):", distance)
80 if ord('K') in keys or ord('k') in keys:
81     print("Distance (K):", distance)
82
83 if distance < 0.18 and (current_time - last_score_time) :
84     score += 2
85     last_score_time = current_time
86     print("得分 +2")
87     print("Current Distance:", distance)
88     emitter.send(str(2)) # 送出得分
89
90 # Movement control
91 if ord('U') in keys or ord('u') in keys:
92     set_wheel_velocity(MAX_VELOCITY, MAX_VELOCITY, MAX_VELOCITY, MAX_VELOCITY)
93 elif ord('J') in keys or ord('j') in keys:
94     set_wheel_velocity(-MAX_VELOCITY, -MAX_VELOCITY, -MAX_VELOCITY, -MAX_VELOCITY)
95 elif ord('K') in keys or ord('k') in keys:
96     set_wheel_velocity(-MAX_VELOCITY, MAX_VELOCITY, -MAX_VELOCITY, MAX_VELOCITY)
97 elif ord('H') in keys or ord('h') in keys:
98     set_wheel_velocity(MAX_VELOCITY, -MAX_VELOCITY, MAX_VELOCITY, -MAX_VELOCITY)
99 elif ord('Q') in keys or ord('q') in keys:
100     print("Exiting...")
101     break
```



- **WHEEL_RADIUS**、**L**、**W**、**MAX_VELOCITY** 等是設置機器人運動的參數，如輪子半徑、機器人長寬、最大速度等。
- **robot** 是初始化機器人的物件，**keyboard** 用來接收鍵盤輸入，**emitter** 用來發送得分。
- **sensor** 用來獲取距離傳感器的數據
- **ad_to_distance** 函數使用了一個查找表（lookup_table）將來自距離傳感器的模擬數值轉換為實際的距離。
- 得分會在控制台顯示，並且透過 **emitter.send(str(2))** 發送出去。
- 控制指令: **U** = 前進 **J** = 後退 **H** = 左轉 **K** = 右轉 **M** = 顯示距離 **Q** = 退出程式

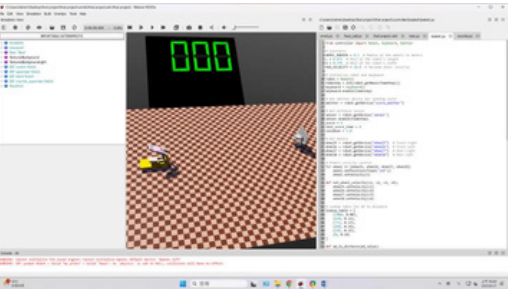
在本學期的期末報告中，41223226的學生以「期末提問」為主題，進行了一項關於創作與設計的報告。本報告將正面答覆，並深入探討期末的作品製作過程、挑戰以及最終的提問。

1. 本協同專案的目標與主要功能為何？



Final Report

2. 場景中包含哪幾個主要子系統或模組？



3. 如何在 Webots 中建立一個可移動的投籃輪車？



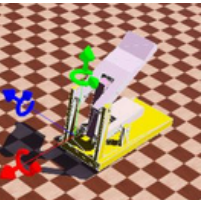
投籃機加到車上

4. 籃框架的移動模式是如何設計與實現的？



籃框加到車上

5. 投籃輪車的移動控制與發射功能分別由哪些裝置組成？



車 如何設定投籃輪車初始位置與限制其活動範圍？

6. 沒限制範圍初始(0,0,0)

完整答覆過程可以參考此以下鏈接：

https://drive.google.com/file/d/1vcA-8HGq5X5PwBRYYJKvsMTp9AxpSw21H/view?fbclid=PAQ0xDSwK9vc9leHRuA2FlbQIxMAABp-Dgv4gVe5zHz5lYnPDUDZ8B0YalgPtiS2vPpfftCq7T3SmlUHynqbE-cWuo_aem_vPo2TVGyv1ZdXMz9IlcnmQ