國 立 虎 尾 科 技 大 學

機 械 設 計 工 程 系

cd2023 2a-pj1ag2 分組報告

網際足球機器人

Web-based Football Scene Design

指導教授： 嚴 家 銘 老 師

班 級： 四 設 二 甲

學 生： 第 一 位 (41023146)

第 二 位 (41023148)

中華民國 112 年 3 月

國立虎尾科技大學　機械設計工程系
分組報告製作合格認可證明

分組報告製作修習學生：　四設二甲 41023146 第一位
四設二甲 41023148 第二位

分組報告題目　：網際手足球場景設計

經評量合格，特此證明

評 審 委 員：＿＿＿＿＿＿＿＿＿＿＿＿＿＿

＿＿＿＿＿＿＿＿＿＿＿＿＿＿

＿＿＿＿＿＿＿＿＿＿＿＿＿＿

指 導 老 師：＿＿＿＿＿＿＿＿＿＿＿＿＿＿

中 華 民 國 一 一 二 年 三 月 三 十 一 日

摘 要

本課程將在設計簡單的移動機器人 BubbleRob 的同時，介紹相當多的 CoppeliaSim 功能。本教程相關的 CoppeliaSim 場景文件位於 scenes/tutorials/BubbleRob。

此專題是運用足球機器人，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運用 AI 進行訓練，找到適合此系統的演算法後，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。

關鍵字: 類神經網路、強化學習、caht gpt、CoppeliaSim、OpenAI Gym

誌謝

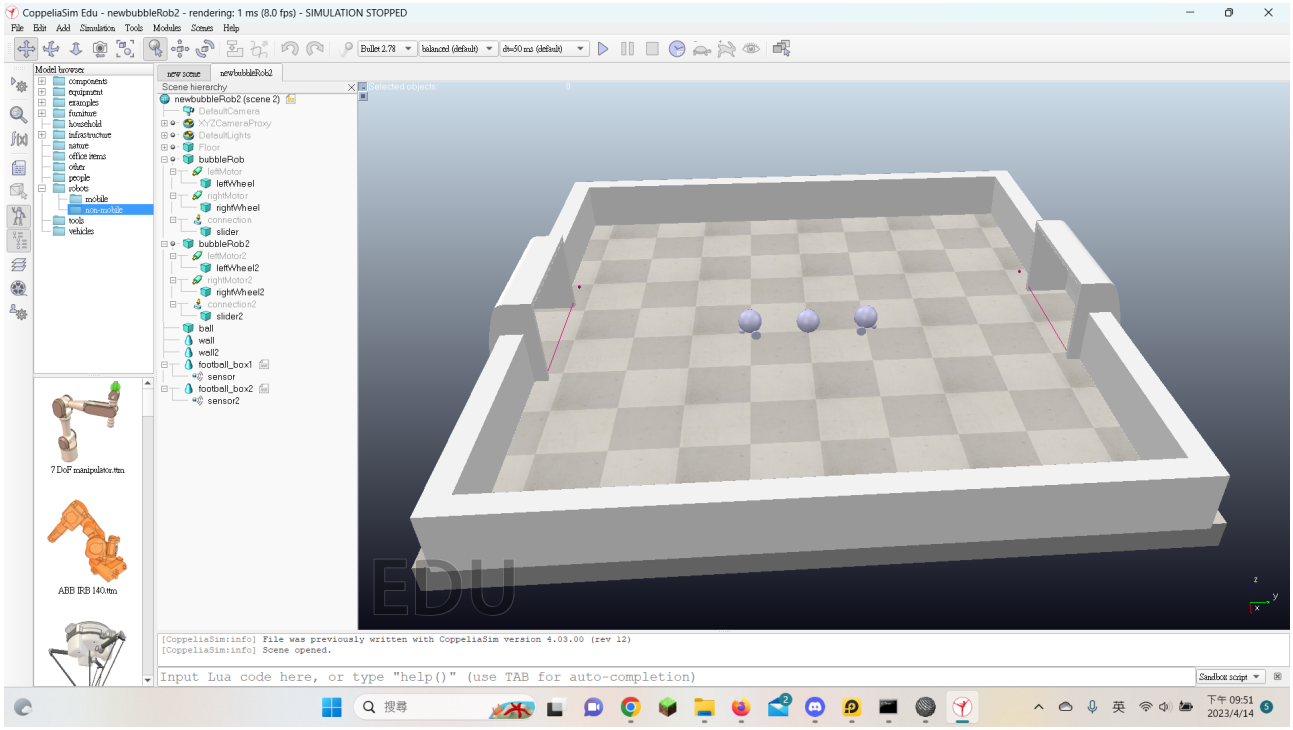
在此鄭重感謝製作以及協助本分組報告完成的所有人員，首先感謝學長範本，讓我們比較好改,也感謝老師平日的教導。

目　錄

第一章　前言

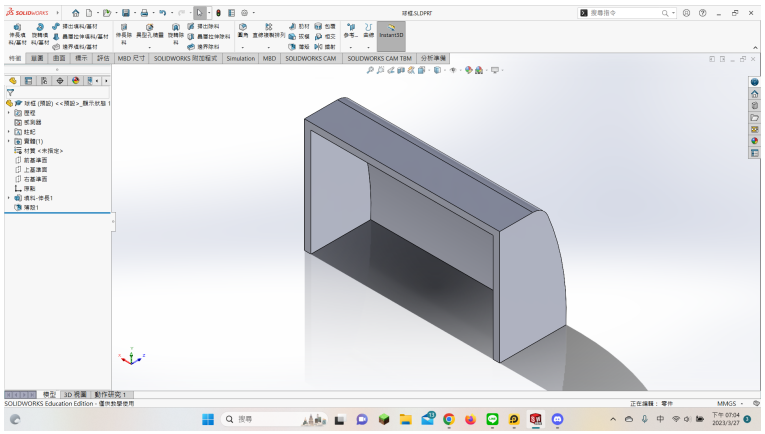
1.1 研究動機

機器學習與各領域結合的應用越來越廣泛，在機電系統採用強化學習是為了讓機電系統的控制達到最佳化。本專題以實體的足球機之機電系統作為訓練模型，將實體機器轉移到虛擬環境進行模擬，為了找到適合的訓練參數，因此將模型簡化後再進行測試各種參數的優劣，透過不斷的訓練來得到一個優化過的對打系統，以下是成品圖。



1.2 製作過程

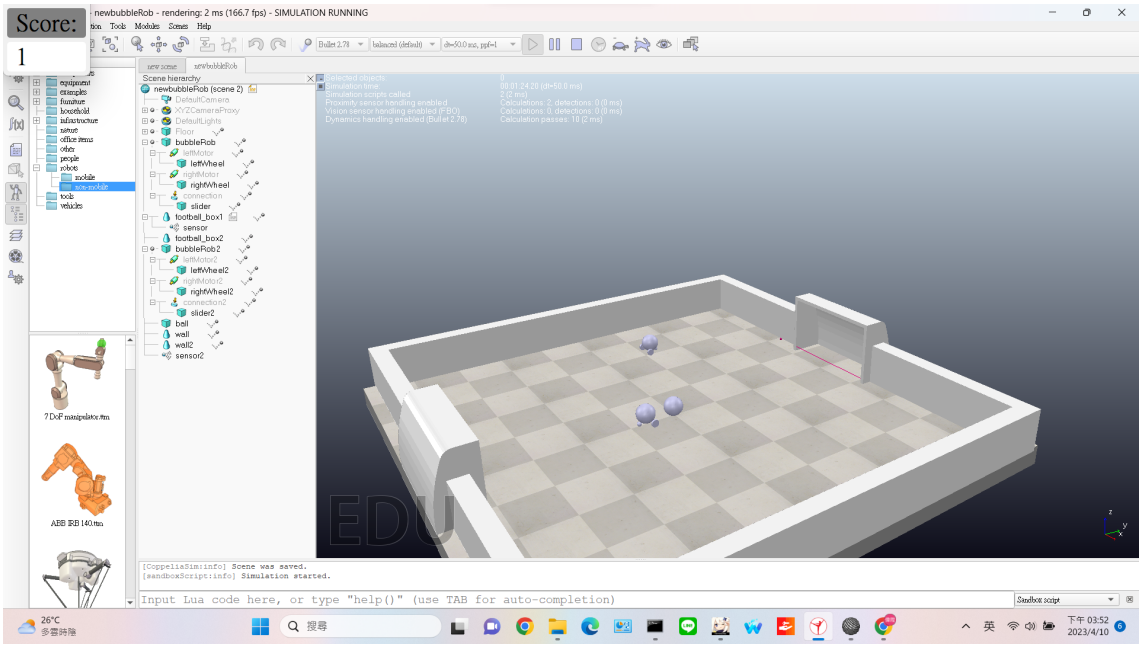
1. 首先繪製球框。



2. 這是 lua 腳本控制是 bubbleRob 前後左右 sim.getObjectHandle 這個函式在 coppeliasim4.3.0 版本被淘汰了但還是可以使用，但用 sim.getObject 會比較好在後面感測器腳本已做改善，這個程式是用上下左右控制 bubbleRob 空白鍵暫停開始。

```
function sysCall_init()
    right_wheel= sim.getObjectHandle('joint1')
    left_wheel= sim.getObjectHandle('joint2')
    right_velocity=0
    left_velocity=0
    speed=5
    sim.setJointTargetVelocity(right_wheel,0)
    sim.setJointTargetVelocity(left_wheel,0)

end

function sysCall_actuation()
    message,auxiliaryData=sim.getSimulatorMessage()
    while message~=-1 do
        if (message==sim.message_keypress) then

            if (auxiliaryData[1]==32) then

                right_velocity=0
                left_velocity=0
                sim.setJointMaxForce(right_wheel, 0)
                sim.setJointMaxForce(left_wheel, 0)
                break
            else
                sim.setJointMaxForce(right_wheel, 10)
                sim.setJointMaxForce(left_wheel, 10)


                if (auxiliaryData[1]==2007) then -- up key

                    sim.setJointTargetVelocity(right_wheel,speed)
                    sim.setJointTargetVelocity(left_wheel,speed)
                end
                if (auxiliaryData[1]==2008) then -- down key

                    sim.setJointTargetVelocity(right_wheel,-speed/2)
                    sim.setJointTargetVelocity(left_wheel,-speed/2)

                end
                if (auxiliaryData[1]==2009) then -- left key
                    sim.setJointTargetVelocity(right_wheel,speed)
                    sim.setJointTargetVelocity(left_wheel,speed/2)

                end
                if (auxiliaryData[1]==2010) then -- right key
                    sim.setJointTargetVelocity(right_wheel,speed/2)
                    sim.setJointTargetVelocity(left_wheel,speed)

                end

            end
        end
        message,auxiliaryData=sim.getSimulatorMessage()
    end
end
```

3. 加入球框感測器和記分板。



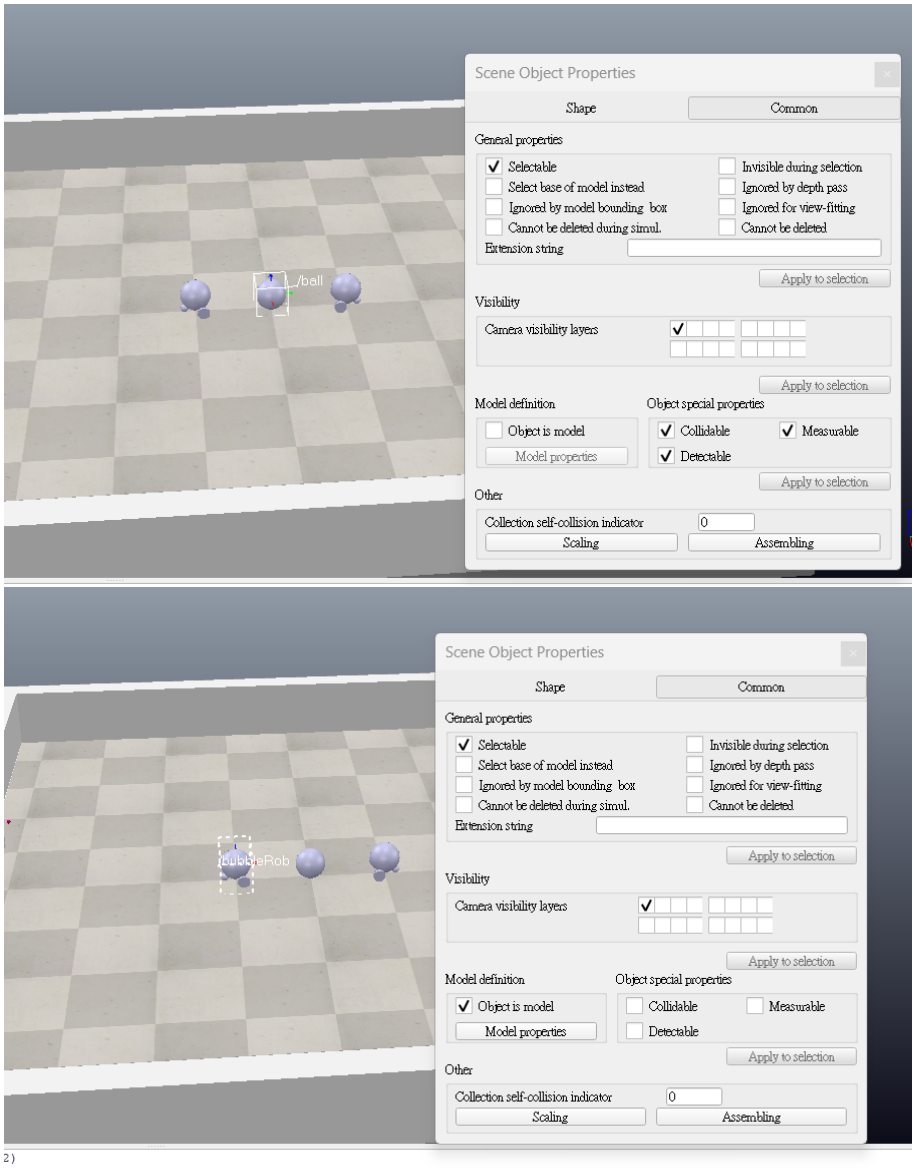
詳情可見

https://mdecd2023.github.io/football-apj1/content/ag2.html

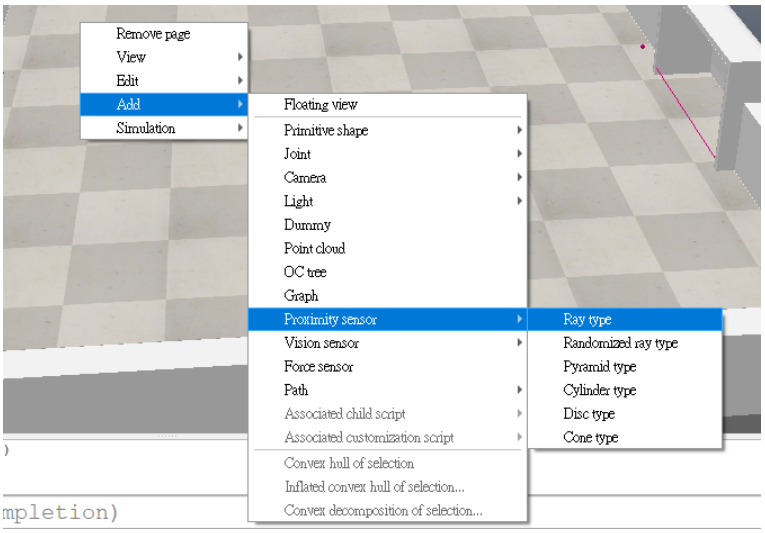
第二章　環境設定

2.1 碰撞檢測

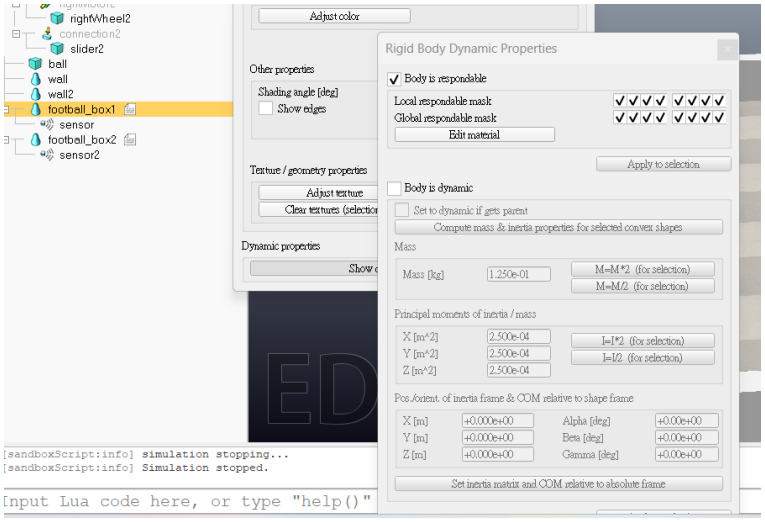
1. 可探測 (Detectable)，可讓感測器感測到物體。在本遊戲中，球應該將 Detectable 打開，這樣才可以感測到進球，而 bubbleRob 機器人則要把 Detectable 關掉，不然機器人碰到感測器也會得分。



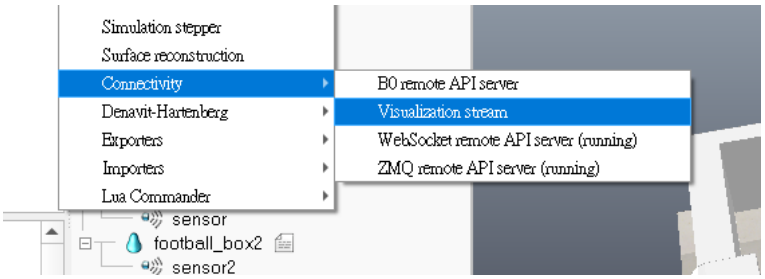
2. 在球框前加入射線感測器 (Ray type)，這樣球進框就一定會碰到感測器，射線感測器不能貼地不然感測不到。



3. 球框及圍牆的 Body is respondable 要打開不然球會穿過去，而 Body is dynamic 則要關上不然球框會亂動。



4. 開啟 Connectivity->Visualization，從 http://127.0.0.1:23020/ 中可看見。



第三章　程式

3.1 程式講解

1. 先用 xml 製作記分板，然後用 sim.getObjectPosition 將 bubbleRob、bubbleRob2 和球的初始位置紀錄下來，然後用 sim.getObjectOrientation 將初始方向記錄下來。

```
function sysCall_actuation()
    --simUI.setLabelText(ui, 30, tostring(sim.getFloatSignal("myVariable")))
    result=sim.readProximitySensor(sensor)
    if(score1<5)then
        if(result>0)then
            score2 = score1+1
            simUI.setLabelText(ui, 30, tostring(score2))

            sim.setObjectPosition(bubbleRob, -1, initialPosition)
            sim.setObjectOrientation(bubbleRob, -1, initialOrientation)
            sim.setObjectPosition(bubbleRob2, -1, initialPosition2)
            sim.setObjectOrientation(bubbleRob2, -1, initialOrientation2)
            sim.setObjectPosition(ball, -1, initialballPosition)
            sim.setObjectOrientation(ball, -1, initialballOrientation)
            score1=score2
        end
    else
        sim.pauseSimulation()
    end
end
```

2.sim.readProximitySensor 這個函數若傳感器未檢測到任何物體，返回值將是負數。如果傳感器與一個物體相交，返回值將是一個非負數的距離值，所以可以用 if 迴圈判斷函示返回值大於零去做加分跟回歸初始位置的動作 simUI.setLabelText(ui, 30, tostring(score2)) 這個函式則是將獲地的分數轉成 string 並輸入到 id 為 30 的 text。

```
function sysCall_init()
    score1 = 0

    sensor = sim.getObject('./sensor')
    xml = [[
        <ui title="Scoreboard" closeable="false" resizable="false" style="pl
        <label text="Score:" style="* {background-color: #808080; color: #00
        <label text="0" style="* {background-color: #FFF; color: #000000; fo

        </ui>
    ]]
    ui = simUI.create(xml)
    simUI.setPosition(ui, 0,0, true)
    bubbleRob = sim.getObject('/bubbleRob')
    ball = sim.getObject('/ball')
    bubbleRob2 = sim.getObject('/bubbleRob2')
    initialPosition = sim.getObjectPosition(bubbleRob, -1)
    initialOrientation = sim.getObjectOrientation(bubbleRob, -1)
    initialPosition2 = sim.getObjectPosition(bubbleRob2, -1)
    initialOrientation2 = sim.getObjectOrientation(bubbleRob2, -1)
    initialballPosition = sim.getObjectPosition(ball, -1)
    initialballOrientation = sim.getObjectOrientation(ball, -1)

end
```

3. 這個程式要執行要先下載兩個東西 pip install keyboard、pip install pyzmq cbor 將這兩行打進 cmd 執行。

```
# pip install pyzmq cbor
from zmqRemoteApi import RemoteAPIClient
import keyboard
```

4. 這個程式要連線遠端電腦要將:

client=RemoteAPIClient('localhost', 23000)

中的 localhost 改成遠端電腦的 ipv4 位址並將遠端電腦的防火牆關掉，後面只要這個軸 ('/leftMotor')('/rightMotor') 的名稱對應到 coppeliasim 就可以遠端控制。

```
# pip install pyzmq cbor
from zmqRemoteApi import RemoteAPIClient
import keyboard

client = RemoteAPIClient('localhost', 23000)

print('Program started')
sim = client.getObject('sim')
sim.startSimulation()
print('Simulation started')

def setBubbleRobVelocity(leftWheelVelocity, rightWheelVelocity):
    leftMotor = sim.getObject('/leftMotor')
    rightMotor = sim.getObject('/rightMotor')
    sim.setJointTargetVelocity(leftMotor, leftWheelVelocity)
    sim.setJointTargetVelocity(rightMotor, rightWheelVelocity)

'''
# Example usage 1:
setBubbleRobVelocity(1.0, 1.0)
time.sleep(2)
setBubbleRobVelocity(0.0, 0.0)
'''
# use keyborad to move BubbleRob

while True:
    if keyboard.is_pressed('up'):
        setBubbleRobVelocity(3.0, 3.0)
    elif keyboard.is_pressed('down'):
        setBubbleRobVelocity(-3.0, -3.0)
    elif keyboard.is_pressed('left'):
        setBubbleRobVelocity(-3.0, 3.0)
    elif keyboard.is_pressed('right'):
        setBubbleRobVelocity(3.0, -3.0)
    elif keyboard.is_pressed('q'):
        # stop simulation
        sim.stopSimulation()
    else:
        setBubbleRobVelocity(0.0, 0.0)
```

參考文獻

[1] https://www.coppeliarobotics.com/helpFiles/index.html

附錄

LaTeX

LaTex 為一種程式語言，支援標準庫 (Standard Libraries) 和外部程式庫 (External Libraries)，不過與一般程式語言不同的是，它可以直接表述 Tex 排版結構，類似於 PHP 之於 HTML 的概念。但是直接撰寫 LaTex 仍較複雜，因此可以藉由 Markdown 這種輕量的標註式語言先行完成文章，再交由 LaTex 排版。此專題報告採用編輯軟體為 LaTeX，綜合對比 Word 編輯方法，LaTeX 較為精準正確、更改、製作公式等，以便符合規範、製作。

表. 1: 文字編輯軟體比較表

	相容性	直觀性	文件排版	數學公式	微調細部
LaTeX	√		√	√	√
Word		√			√

- 特點:

1. 相容性：以 Word 為例會有版本差異，使用較高版本編輯的文件可能無法以較低的版本開啟，且不同作業系統也有些許差異；相比 LaTeX 可以利用不同編譯器進行編譯，且為免費軟體也可移植至可攜系統內，可以搭配 Github 協同編譯。
2. 文件排版：許多規範都會要求使用特定版型，使用文字編譯環境較能準確符合規定之版型，且能夠大範圍的自定義排定所需格式，並能不受之後更改而整體格式變形。
3. 數學公式呈現：LaTex 可以直接利用本身多元的模組套件加入、編輯數學公式，在數學推導過程能夠快速的輸入自己需要的內容即可。

4. 細部調整：在大型論文、報告中有多項文字、圖片、表格，需要調整細部時，要在好幾頁中找尋，而 LaTeX 可以分段章節進行編譯，再進行合併處理大章節。

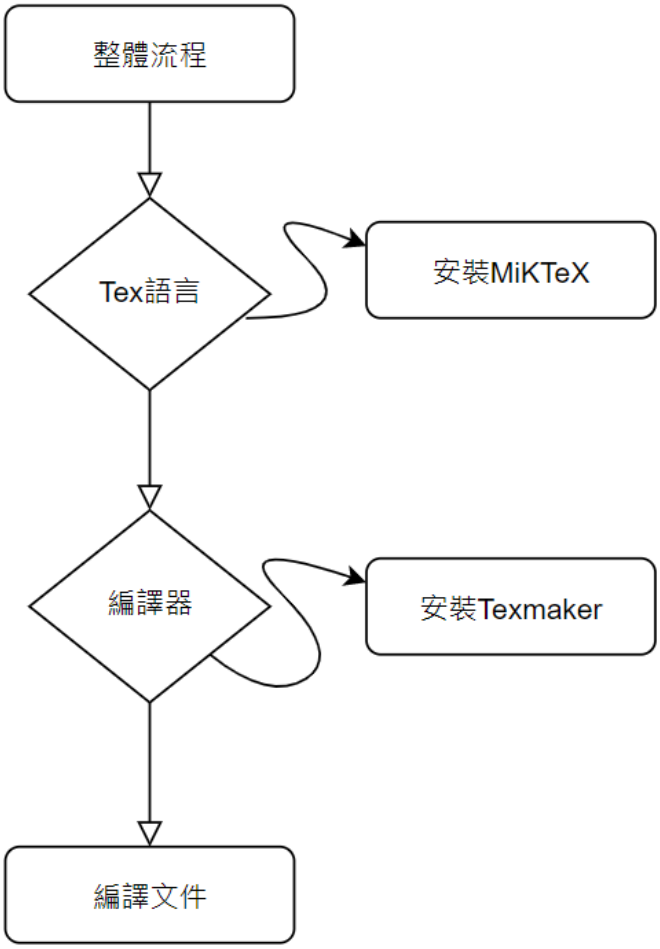


圖. 1: 編譯流程

FFmpeg

FFmpeg 是一個開放原始碼的自由軟體，可以對音訊和視訊進行多種格式的錄影、轉檔、串流功能。在專題訓練過程中透過 FFmpeg 的視訊錄製的功能記錄對打影像來了解實際訓練狀況。