3D互動數位內容製作

期末報告

Unity ML-Agent AI Car

1053320黃莉婷

起初因為一直試不出來課堂上的方法，為了重振信心，因此到Unity官方提供的教程一步一步跟著做，希望能先有個好兆頭。

首先是環境設定，完全參照此網站

<https://github.com/Unity-Technologies/ml-agents/blob/master/docs/Installation-Windows.md>

Unity和Anaconda之前已經裝好了，主要是要新創一個Pythob環境，安裝

* Python3.6
* Tensorflow1.7.1

並使用官方範例(ml-agent專案內的3D-Ball)，將之修改成我要的AI車子。

開始修改之前，要先看懂原本的內容

如何**放入已經訓練好的模型**和如何**自己訓練出一個模型**看這

<https://github.com/Unity-Technologies/ml-agents/blob/master/docs/Basic-Guide.md>

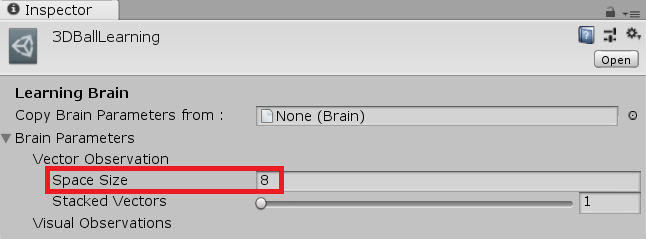
3D-Ball的**程式碼解說**看這

<https://github.com/Unity-Technologies/ml-agents/blob/master/docs/Getting-Started-with-Balance-Ball.md>

也有中文翻譯好的網站<https://blog.csdn.net/u010019717/article/details/80472011>

針對Ball3DAgent.cs程式碼解說，官網並沒有講得很詳細，以下會有些許我自己理解的補充，若有錯誤理解還請告知。

Ball3DAgent.cs內有三個function

* CollectObservations()
  + 負責收集Agent對環境的觀察數據，範例中觀察平台的旋轉角度、球的位置等共8個，所以會調用8次AddVectorObs(position內涵x,y,z三個變數，因此算三個)，會產生大小為8的空間給Brain計算。
  + 此數必須要與Brain(3DBallLearning)的向量空間觀察大小相同
* AgentAction(float[] vectorAction , string textAction)
  + 接收Brain選擇的步驟，此範例環境為連續的運動空間類型(亦有離散的)，多數範例會在這裡分配獎勵
* AgentReset()
  + Agent重置時會調用(範例中重置平台角度和球的位置等)，適當使用random(隨機調整數值)可以使測資多元，較容易訓練出不同情況下的反應

看懂原本的程式碼在寫甚麼之後，我們就來改改看。首先，整理一下要修改的部分

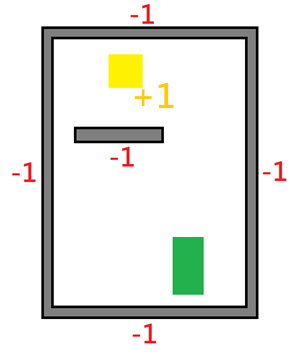
[原本範例] >> [更換成]

* 球 >> 方塊 (車子)
* 控制平台 >> 控制車子
* 控制X,Z軸旋轉方向 >> 控制車子位移、旋轉
* [獎懲] 沒掉+0.1，掉了-1 >> 底達目標+1、撞到障礙物-1
* 觀察xxxxxx >> 觀察車旋轉、和目標&障礙物的距離

新增

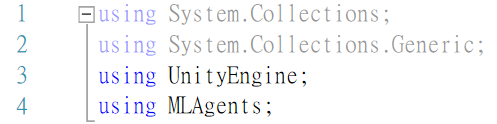
* 為車子加裝前後射線，感知障礙物和目標

概念設計圖 :

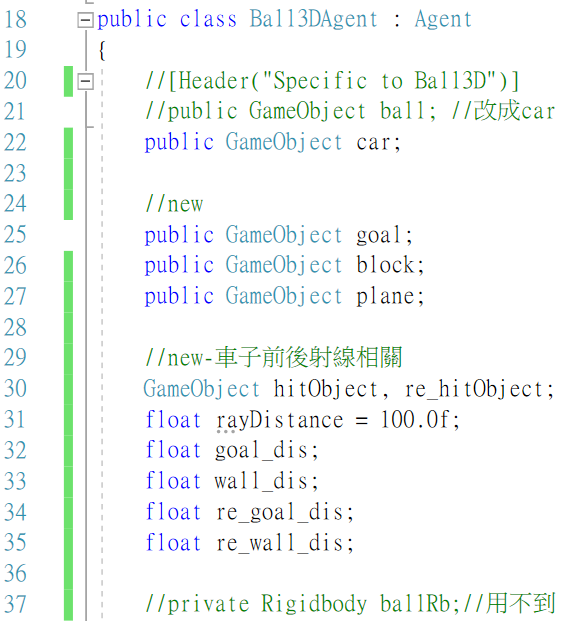


以下為實際修改過後的程式碼:

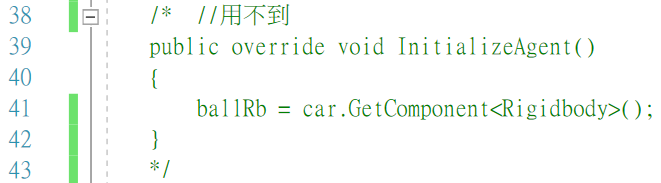
Using和原本的一樣，不變



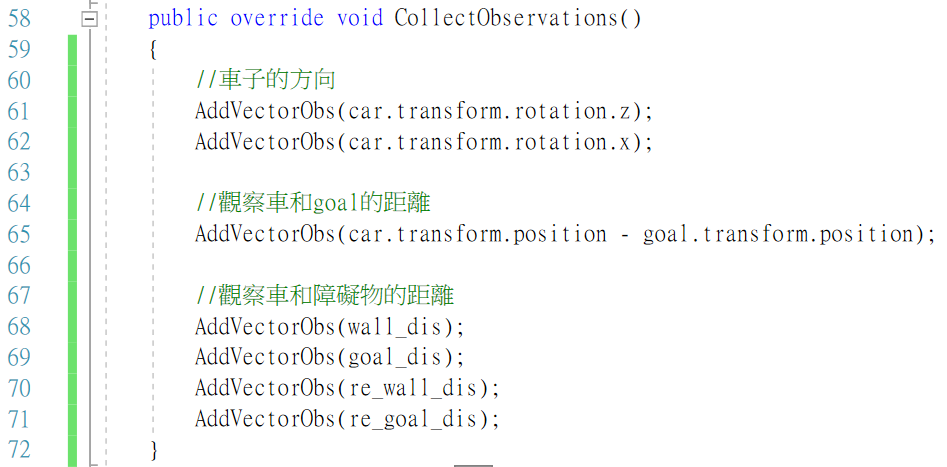
新增變數



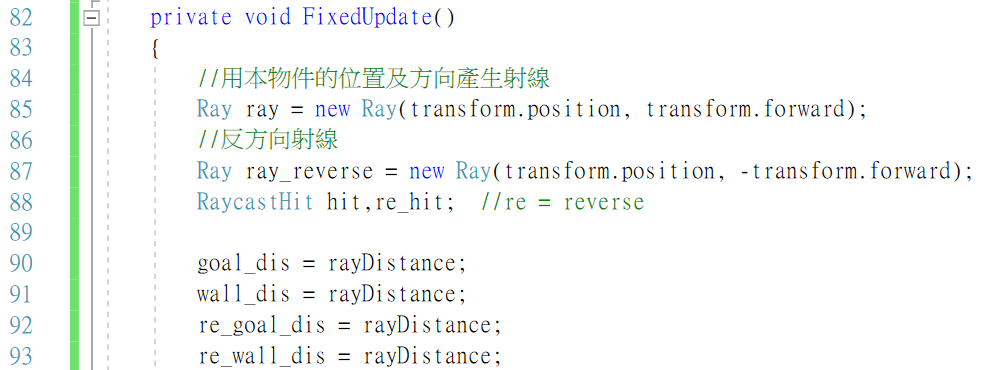
註解掉InitializeAgent()，用不到

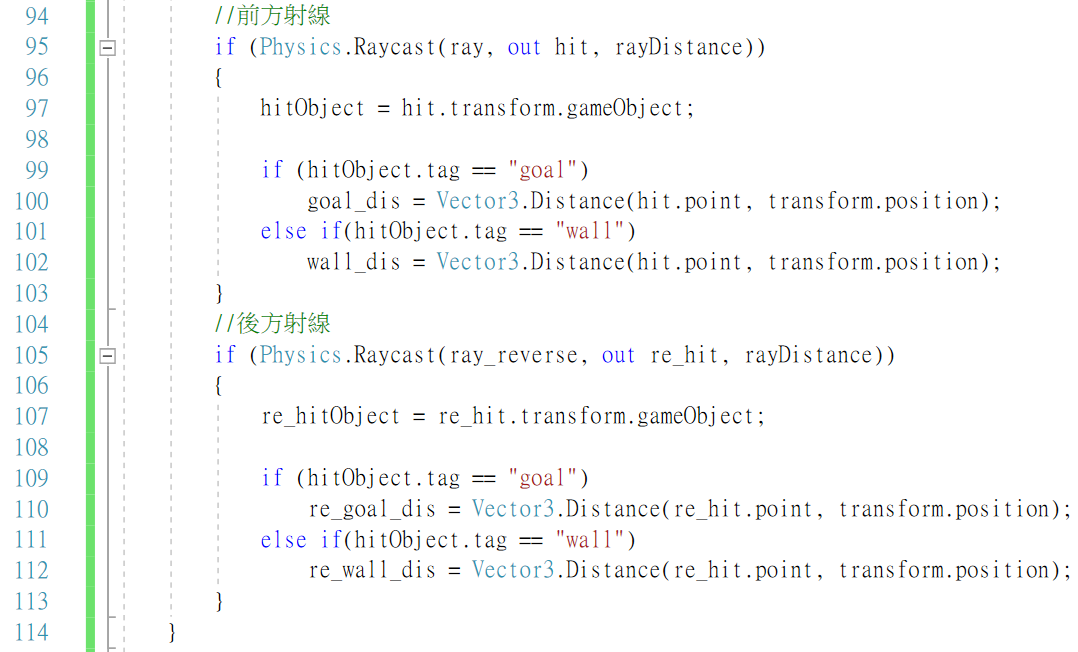


清空改寫CollectObserations()，此時觀察對象有9個

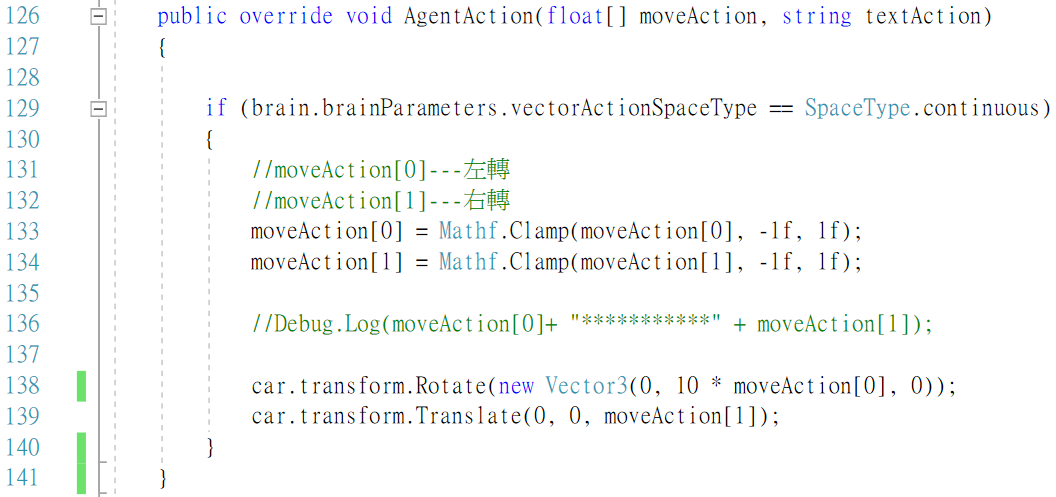


新增FixedUpdate()，時時刻刻更新前後物體距離數值

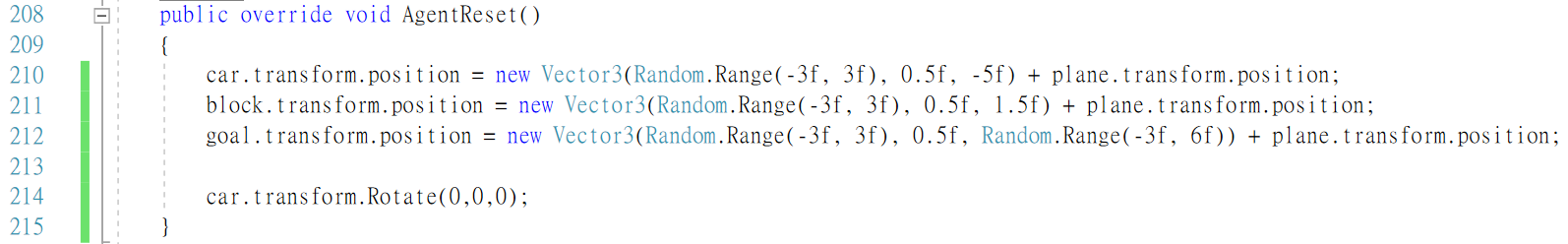




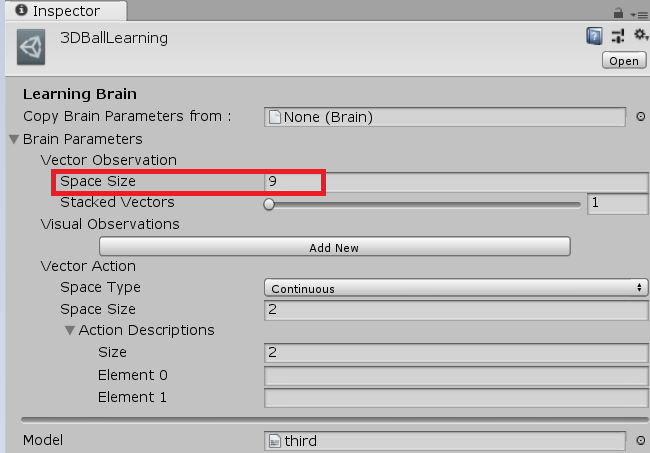
更改AgentAction()



清空改寫AgentReset()

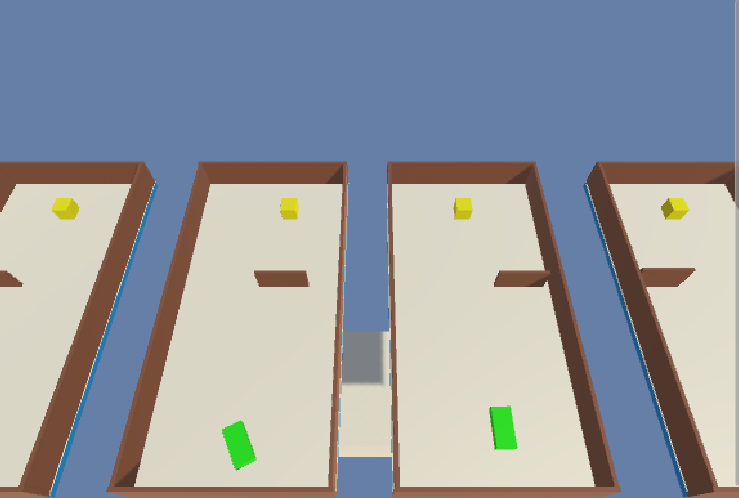
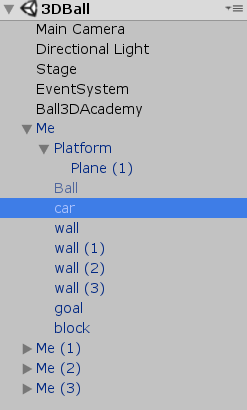


因為在CollectObserations()，觀察對象有9個，所以這裡數目也要改



以上是程式碼的部分。

完成後將此檔案(原本在plane物件下)改放到Car(自己新增的cube)物件下

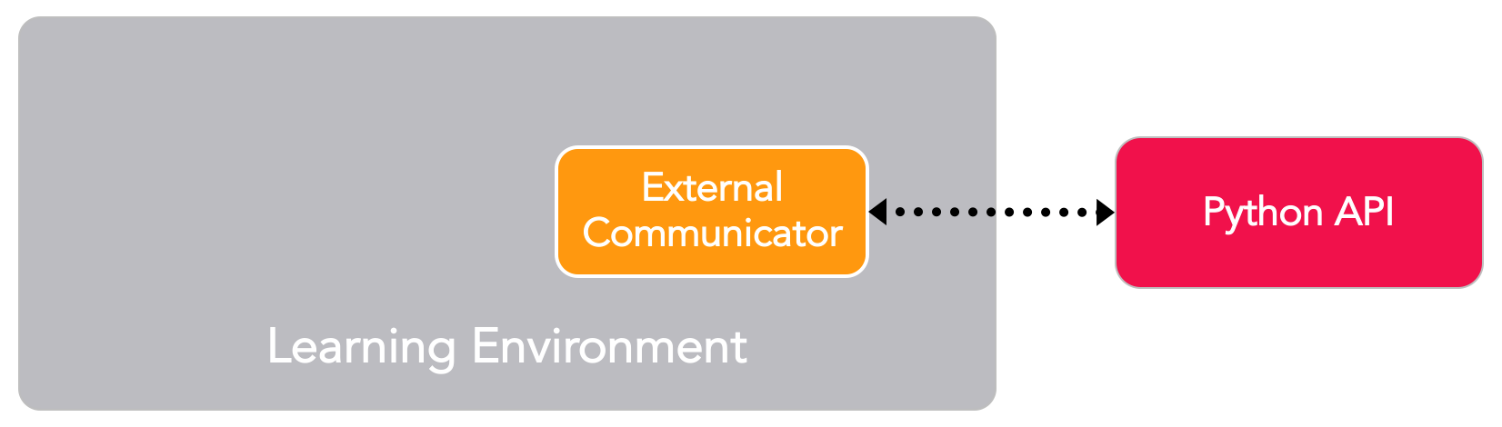


最後照著前面訓練模型的方式即可。

**ML-Agents概述**

ML-Agents是一個Unity插件，包含了三個高級組件

* 學習環境 – 包含Unity場景和所有遊戲腳色
* Python API – 包含用於訓練的所有機器學習算法，與學習環境不同的是，Python API不適Unity的一部分，而是位於外部通過External Communicator與Unity進行通訊
* External Communicator - 他將Unity環境與Python API串接起來。位於Unity環境中



**機器學習過程**

在3D-Ball範例中使用強化學習(Reinforcement Learning)訓練Brain。為了訓練agent對球進行平衡，使用一種稱為Proximal Policy Optimization(PPO)的強化學習算法。與其他許多RL算法相比，經證明這種算法是更有效、通用的方法。

為了在Ball Balance環境中訓練Agent，將使用Python包。官方提供了一個方便的命令mlagents-learn，用於接收訓練等參數。

總結來說，實際上的操作就是開啟cmd到ml-agents資料夾下，輸入

mlagents-learn config/trainer\_config.yaml --run-id=<run-identifier> --train

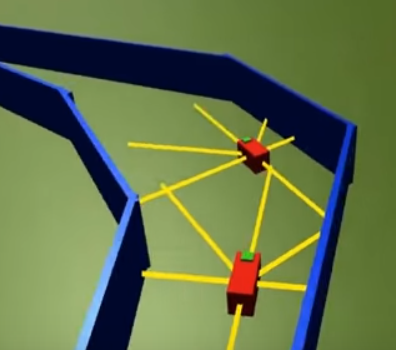
範例輸入

mlagents-learn config/trainer\_config.yaml --run-id=firstRun --train

其中YAML是用來表達資料序列的格式。YAML參考了其他多種語言，包括：C語言、Python…等

後記

最後車子雖然在訓練過程中有接近30%會抵達目標，但實際上拿訓練好的模型來跑的話，會經常在原地猶豫不決，因此後續考慮在車子上多裝幾個射線(如下圖)，協助判斷。



果然要在一天內看完所有ML-Agnet的觀念實在有點吃不消，因此我打算日後慢慢看完所有github 上ml-agnet這專案的文件，由於我英文也不太好，所以我已經將此專案fork一份到我的github上，打算每天看一點，並在看過後將之翻譯成繁體中文，讓英文不好的人也可以輕鬆學機器學習。

我的github: <https://github.com/tsumikihuang/ml-agents-/blob/master/README.md>

我完成的作品: https://drive.google.com/open?id=1xX9QtmwWfczfuY9\_ImD7bP-H32KwHIVb