深度學習作業報告

3D模擬環境中的導航訓練

系級：

學號：

姓名：

2020年6月16日

**介紹**

近期利用模擬環境來訓練神經網路非常盛行，本作業利用Habitat-API(註3)來獲得3D環境，並利用強化學習(Reinforcement Learning)方法，讓Agent(代理人)進行環境導航。

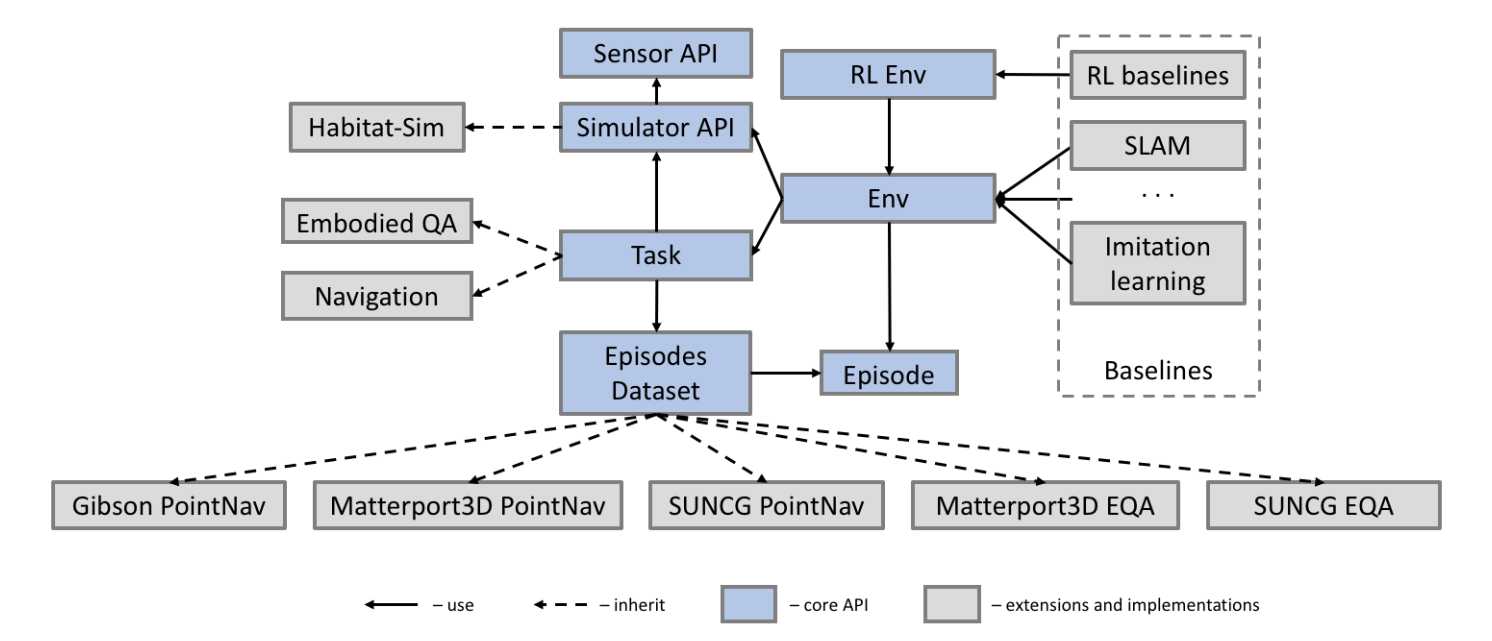
本作業實作了兩種RL的演算法PPO和Split+PPO，Split+PPO為參考論文(註7) 來實現，並比較了兩者之間的結果。

**方法**

本作業利用到Habitat-API、Python、CUDA、Pytorch、PPO、Split+PPO。以下將介紹Habitat-API、論文(註1)提供的Baseline、PPO、Split+PPO。

ㄧ、Habitat-API

模擬3D環境。  
Python藉由呼叫Habitat-API，獲得環境資訊(影像、深度、GPS、方位)，利用演算法(RL)訓練Agent達成任務。

  
Env

模擬3D環境，本作業使用Gibson資料集(註2)，88種場景。Agent可以在環境中移動。



(示意圖)

Task

在Env中可訓練、評估的任務，包含:導航(Navigation)、問題詢問(Embodied QA)、探索。

定義任務的目的、位置、成功條件、時間限制…。

Sensor API

可以藉由Sensor獲得Env中的數值，包含:RGB影像、深度影像(depth)、GPS、羅盤(目標方位)。

要使用哪些Sensor資料，取決於演算法要哪些資料，可以只使用RGB訓練，也可以只使用depth訓練…。

Agent

代理人。控制機器人在Env中走動，包含動作(step): 停(任務結束)、前進(0.25 公尺) 、左轉(10度)、右轉(10度)。

Episode

Agent執行一次Task的完整過程。Agent每一個frame都可以獲得Sensor資訊，藉由獲得的資訊，Agent做下一步的動作(step)。當超過500 frame的動作後，此Episode自動結束，任務若沒完成，即算失敗。

二、Baselines

Habitat(註1) 論文用導航來當作Baselines的目標任務。以各類演算法來執行導航任務，給予使用者一個評斷演算法的基準(圖2)。

導航 (Navigation)

在Env中，Agent在一開始可以藉由GPS，獲得目的地位置以及自身位置。

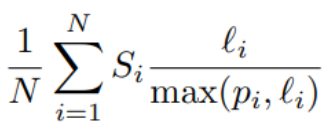
此任務目的:從自身位置走到目的地位置。  
以下影片範例 (影片1) ，為由上而下的俯視圖，用來觀測結果的好壞。



(影片1)

Success weighted by Path Length (SPI) (註4)

Habitat採用此評量準則(圖1)，以下為評量準則的公式。SPI越高代表演算法結果越好。



(圖1)

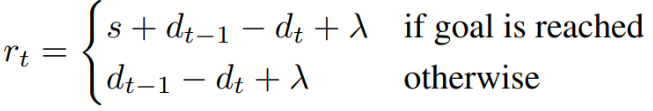
Si : 此次Episode的任務是否成功，成功為1，失敗為0

Pi : Agent在此次Episode共走了多少距離  
li : 最短路徑

i : Episode 第i個

Reward function (圖2)

Habitat-API會根據執行的任務，回饋Agent剛剛所執行動作的好壞。以下為導航任務的回饋數值。



(圖2)

s : 若成功的獎勵 (10)

λ : 時間逞罰 (-0.01)

dt : 當前離目的地的距離

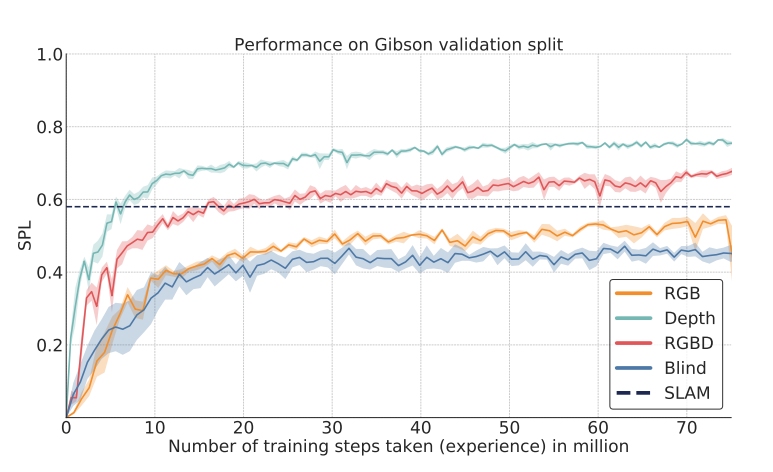
dt-1 : 上一幀離目的地的距離

演算法

比較五種算法結果(圖3)，分別訓練7千5百萬次step。

只有Depth的RL結果最好，論文推測有RGB的訓練結果會因為overfitting效能下降。

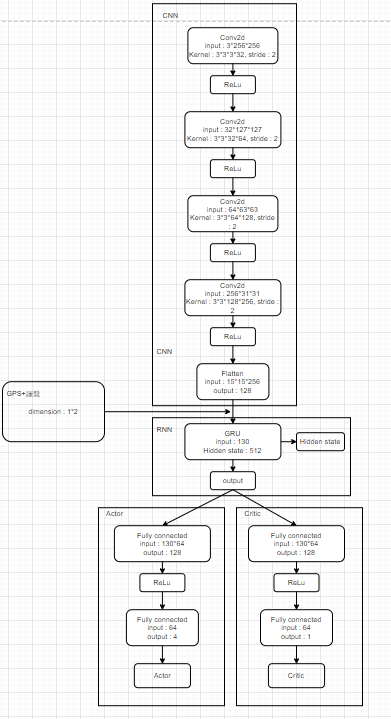
1. RL (PPO)
   1. RGB
      1. 使用RGB
      2. 訓練時間566小時
   2. Depth
      1. 使用Depth
      2. 訓練時間475小時
   3. RGBD
      1. 使用RGB和Depth
      2. 訓練時間906小時
   4. Blind
      1. 完全不使用影像
      2. 訓練時間320小時
2. SLAM(The simultaneous localization and mapping)(註5)
   1. SLAM不需要訓練



(圖3)

# 三、PPO (Proximal Policy Optimization Algorithms) (註6)

PPO演算法為強化學習的其中一種演算法。強化學習藉由獲得每一幀的depth image、GPS(當前位置)、羅盤(目的地方向)，決定Agent該執行哪個動作(step)。

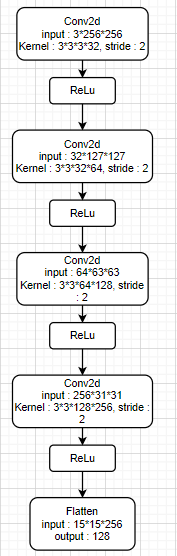
本作業粗略實作PPO演算法，以下為網路架構。  


Replay Buffer

紀錄前數次Episode的所有資料(每一步step)，包含Depth image、Reward、RNN的Hidden state、Actor輸出、Critic輸出。用來之後倒傳遞訓練網路。

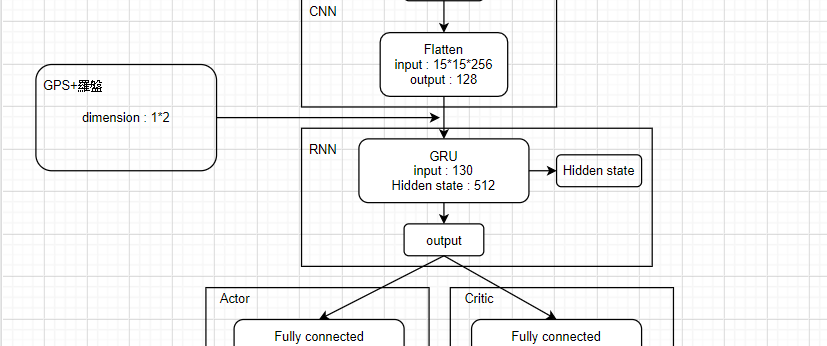
CNN Encoder

將depth image當作input，丟入連續的捲積層當中，獲得較低維特徵。



RNN

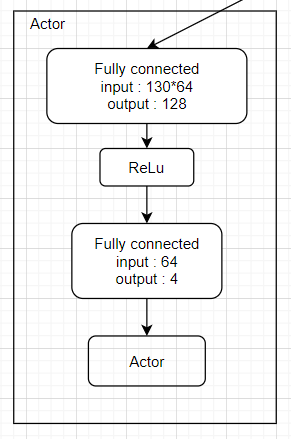
將CNN獲得的特徵+GPS+羅盤數值，丟入RNN當中，輸出給Actor和Critic使用。



Actor

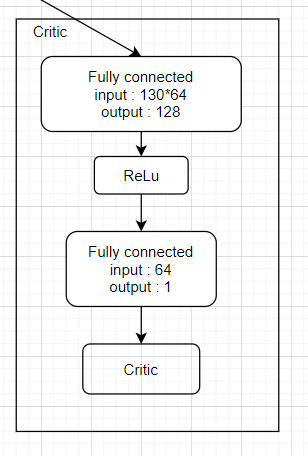
Agent根據輸出，選擇4個動作 (step)的其中之一。

包含動作 : 停(任務結束)、前進(0.25 公尺) 、左轉(10度)、右轉(10度)。

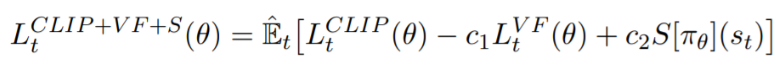


Critic

評論當前depth image能夠獲得的未來獎勵(reward)。



PPO演算法目標函數

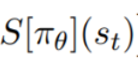


 : actions loss

透過累加每一幀動作(step)獲得的reward，來評估此神經網路的好壞。Reward總合越高，代表此網路越好，因此此項loss越高越好。

 : values loss

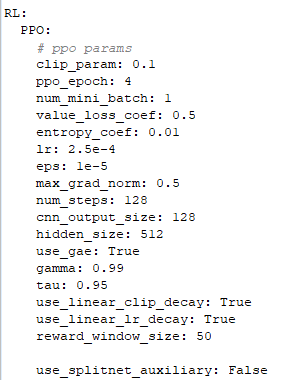
Critic網路輸出數值，預測reward，當reward預測越準確，loss越小。

 : distributions entropy

Actor 4種動作的機率越平均，entropy越高，每個動作就都有機會被採取，越有機會探索到最佳策略。

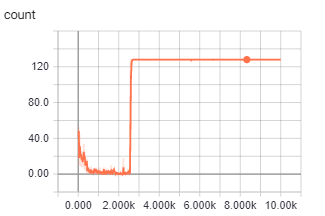
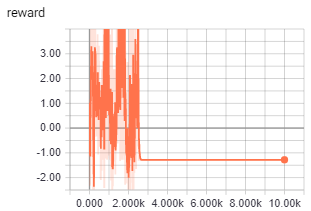
結果

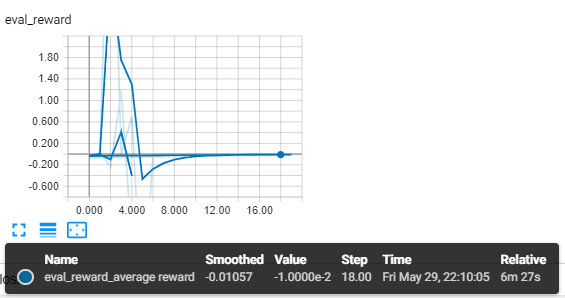
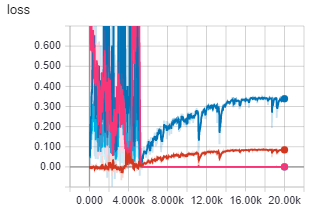
訓練10000次epochs，下圖 (圖4) 為PPO演算法的參數設置。

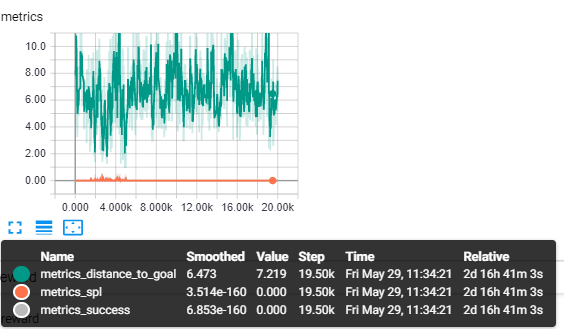


(圖4)

此演算法剛開始時，能夠逐漸收斂優化。但後來Agent僅在Task剛開始時，就直接採取”停”的策略，Agent無法學習導航技能。結果極差。SPI最高為35%，成功導航45%。



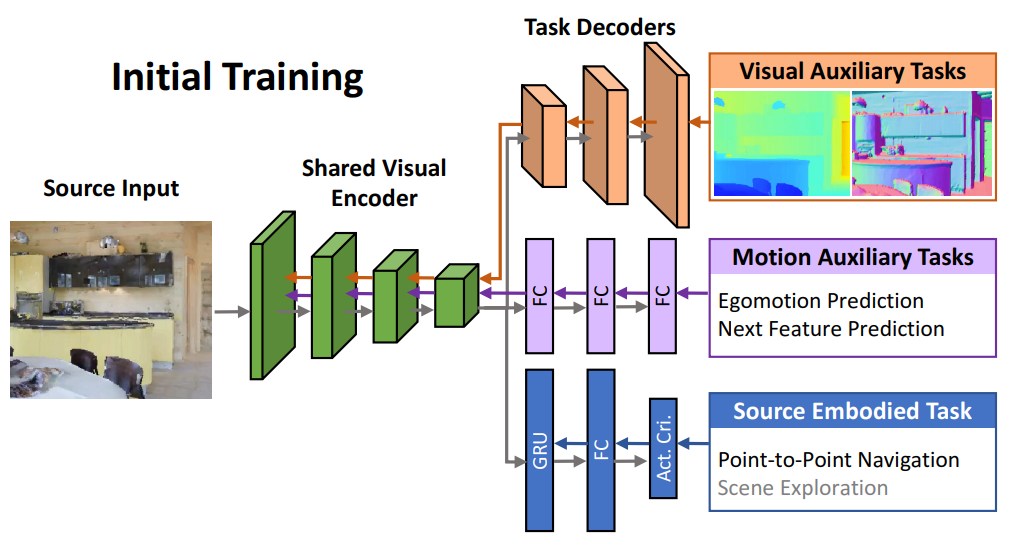


以下影片 (影片2)(影片3) 為Agent尚未收斂時的導航情況。

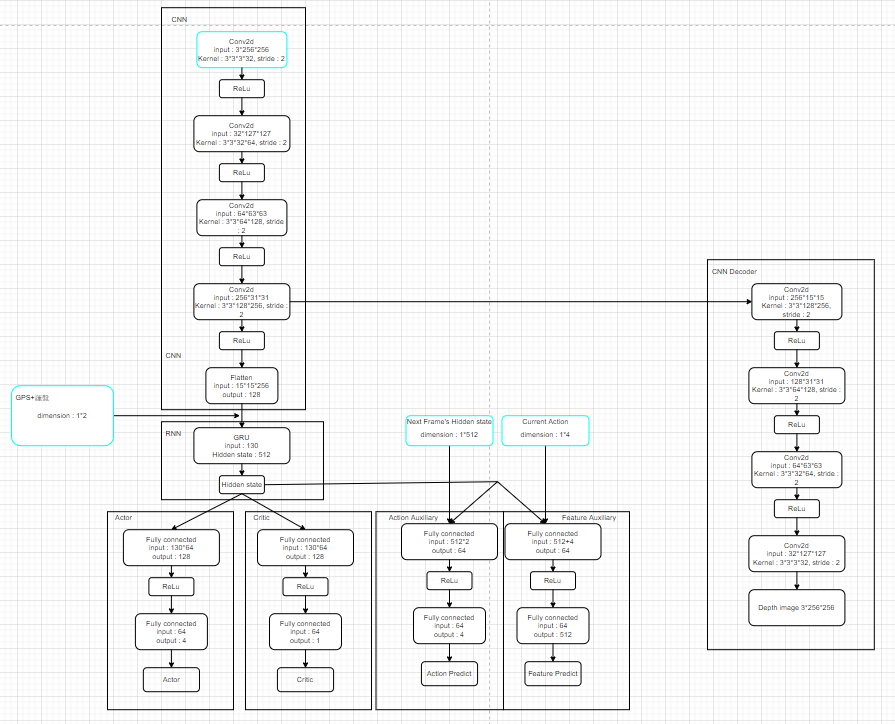
   
 (影片2) (影片3)

# 四、SplitNet: Sim2Sim and Task2Task Transfer for Embodied Visual Navigation (註7)

此論文實驗在導航任務中，除了原本利用PPO演算法(Source Embodied Task)、實驗不同任務的transfer能力，額外添加輔助網路(Motion Auxiliary Tasks)、影像深度網路(Visual Auxiliary Tasks)，主要是讓CNN編碼(Shared Visual Encoder)提取正確特徵，在實際導航過程中，不需要Auxiliary的輸出。

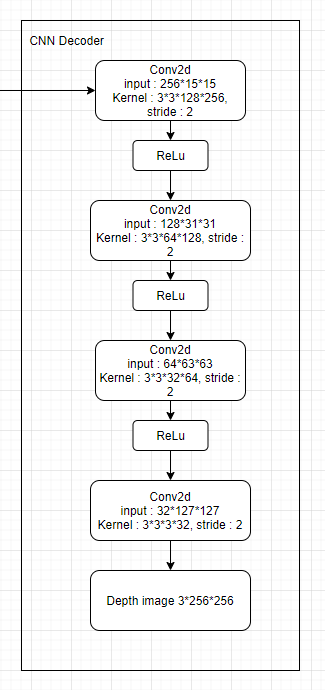
(註7)

本作業粗略實作SplitNet方法，結合 三、PPO 的網路架構，稍微改變輸入資料、網路架構。以下為網路架構。



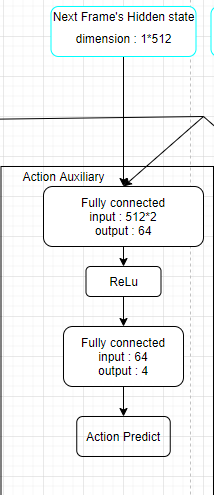
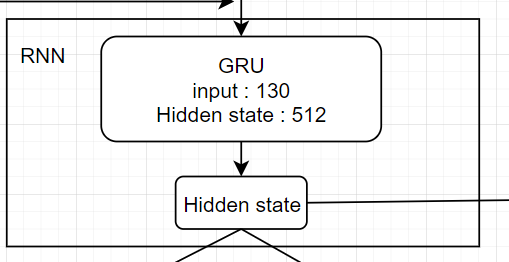
CNN Decoder

將CNN Encoder所獲得的低解析度特徵圖，進行上採樣，獲得3\*256\*256的影像，跟原本CNN Encoder 輸入影像，做均方誤差更新網路。



Action Auxiliary  
Input : 當前幀的RNN Hidden state + 下一幀的RNN Hidden state

Output : 動作(step)  
  
利用”當前幀的RNN Hidden state”和“下一幀的RNN Hidden state”，期望能夠預測出是哪一個動作(step)，導致從”當前幀的RNN Hidden state”改變成“下一幀的RNN Hidden state”。

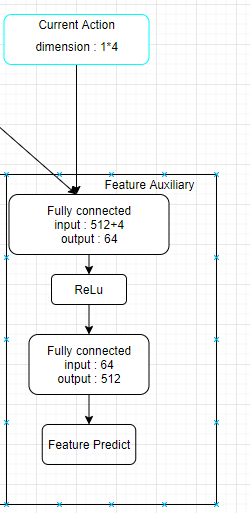
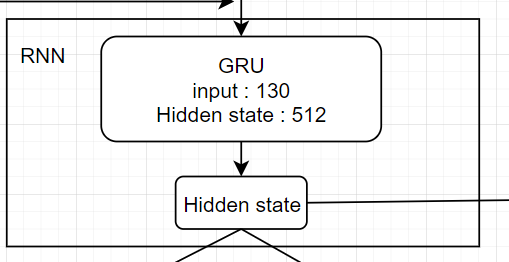
 

Feature Auxiliary

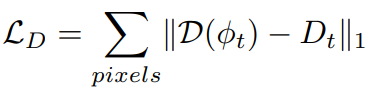
Input : 當前幀的RNN Hidden state + 當前Actor採取的動作(step)

Output : 下一幀的RNN Hidden state

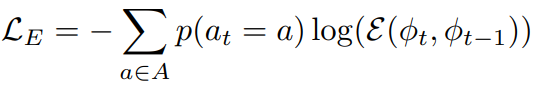
利用” 當前幀的RNN Hidden state”和 ”當前Actor採取的動作(step)”，預測” 下一幀的RNN Hidden state”。

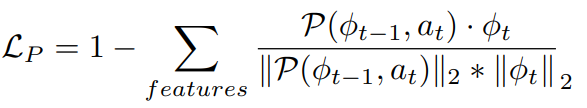
SplitNet Auxiliary目標函數

 : CNN Decoder loss

真實深度圖跟解碼出來的深度圖，彼此之間差距越小越好。

 : Action Auxiliary loss

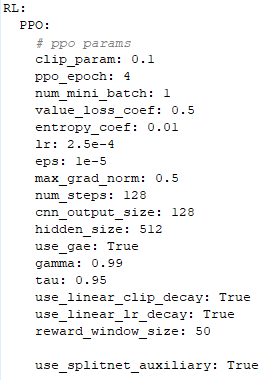
利用cross entropy 計算損失函數，比對Ground True Action和網路預測的Action差距，差距越小越好。

 : Feature Auxiliary loss

利用”預測的下一幀 Hidden state”以及”真實下一幀Hidden state”，使用cosine 比對兩者相似度，最後被1減，就是Feature Auxiliary的loss

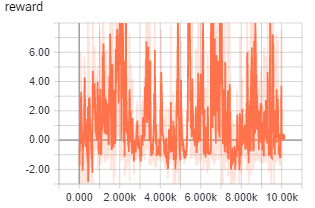
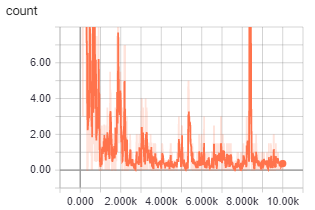
結果

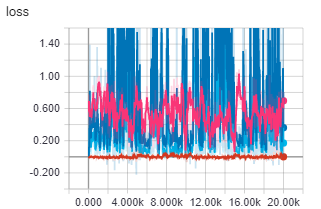
訓練10000次epochs，下圖 (圖5) 為Split+PPO演算法的參數設置。

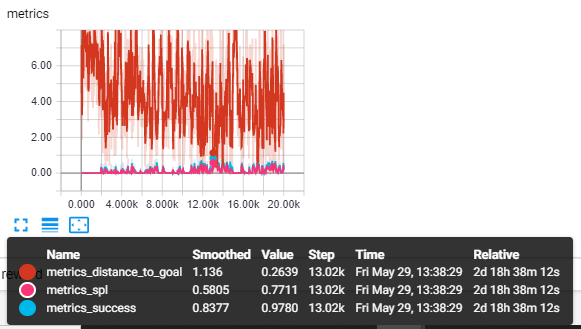


(圖5)

相較於之前的PPO結果，Split+PPO演算法能夠預防Agent最終直接採取”停”的策略。SPI可以到達最高0.58的結果，成功導航83%。比起PPO演算法更加穩定。







以下影片 (影片4)(影片5) 為Agent導航成功以及導航失敗的影片。

(影片4) (影片5)

**參考資料**

[1] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, Dhruv Batra. Habitat: A Platform for Embodied AI Research. arXiv:1904.01201, 2019

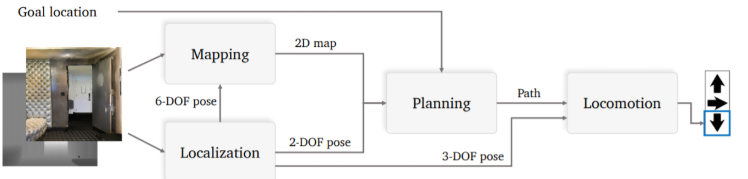
[2] Gibson (<http://gibsonenv.stanford.edu/database/>)

3D環境模型

[3] Habitat-API (<https://github.com/facebookresearch/habitat-api>)

[4] Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, Amir R. Zamir. On Evaluation of Embodied Navigation Agents. arXiv:1807.06757, 2018

[5] Dmytro Mishkin, Alexey Dosovitskiy, and Vladlen Koltun. Benchmarking classic and learned navigation in complex 3D environments. arXiv:1901.10915, 2019.



拆解成4個小模組分工

* mapping
  + outputs a two-dimensional obstacle map of the environment
  + 將深度影像投影到local map；之後合併到Global map；做路徑規劃。
* localization
  + estimates the agent’s pose
* planning
  + plan a path to the goal
* locomotion
  + selects an action to follow the path

[6] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov. Proximal Policy Optimization Algorithms. arXiv:1707.06347, 2017

[7] Daniel Gordon, Abhishek Kadian, Devi Parikh, Judy Hoffman, Dhruv Batra. SplitNet: Sim2Sim and Task2Task Transfer for Embodied Visual Navigation. arXiv:1905.07512, 2019