Agent-Based Reinforcement Learning Model of Burglary (ARLMB) – V.1.0.0

Author: Sedar Olmez
Date: 24/06/2022
Correspondence: solmez@turing.ac.uk

Abstract

This agent-based model was developed using the Unity game engine to incorporate multi-agent reinforcement learning algorithms from the ml-agents OpenAI package. The model simulates offender agents over a 2D landscape containing interventions, targets, and routine activity nodes. Offenders train using a multi-agent reinforcement learning algorithm Proximal-Policy Optimisation (PPO) to learn behaviours that demonstrate realistic patterns of burglary in agreement with the Rational Choice Perspective, Crime Pattern Theory and Routine Activity Theory. The novelty presented by this model is based on the ability for offender agents to learn behaviours naturally from the environment without any hard-coded pre-determined behavioural rules. Users can test Situational Crime Prevention Intervention (SCPI) policies where interventions can be placed in a specific location run-time, thus, increasing risk in the area and the reactions of offenders can be analysed. Overall, the experiment results show that offenders learn to offend at targets where rewards outweigh risks and effort, demonstrating a degree of intelligence, such as offending closer to home, frequently victimising high-rewarding targets, and learning to avoid areas of high risk.

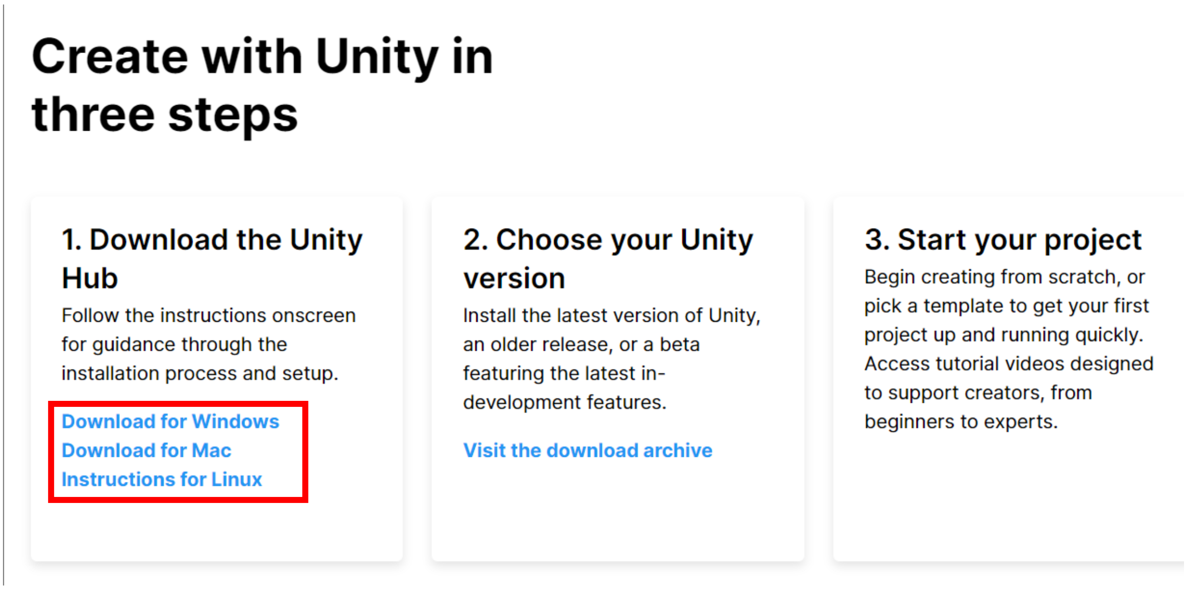
The work conducted in this research is part of a wider PhD research project funded by the Economic and Social Research Council UK, Grant Number: ES/P000401/1.

Installation

Pre-requisites

- Unity version 2020.3.11f
- (Optional) ml-agents 0.25.0, TensorFlow 2.1.0, PyTorch 1.7.1 (for training your own RL agents) *Note: this will be provided as an anaconda environment.*

1) Download the UnityHub at the following link: https://unity.com/download



Create with Unity in three steps

1. Download the Unity Hub

Follow the instructions onscreen for guidance through the installation process and setup.

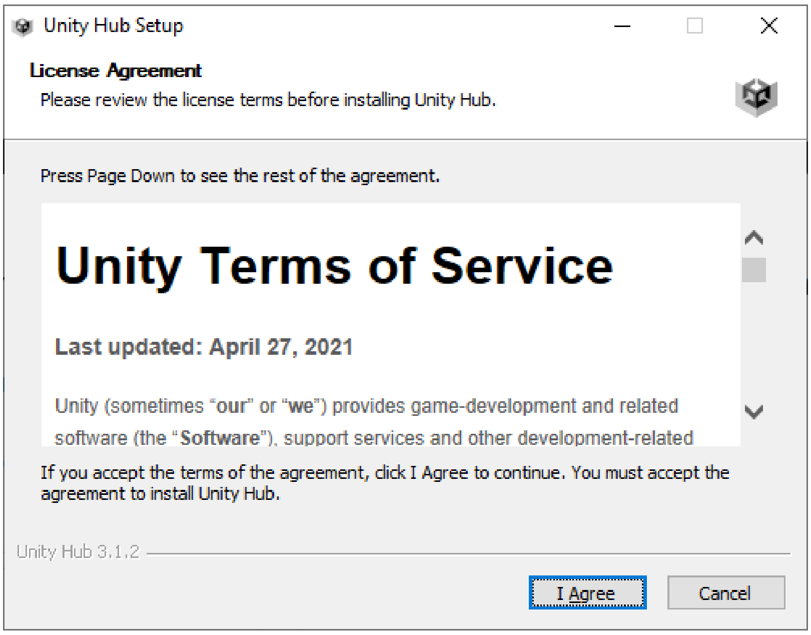
2. Choose your Unity version

Install the latest version of Unity, an older release, or a beta featuring the latest in-development features.

3. Start your project

Begin creating from scratch, or pick a template to get your first project up and running quickly. Access tutorial videos designed to support creators, from beginners to experts.

2) Scroll down the page and click "Download for *" where * is your operating system.
3) Once the UnityHub software has finished downloading, locate it (usually in your Downloads file) and start the installation process by double clicking the program.



Unity Hub Setup

License Agreement

Please review the license terms before installing Unity Hub.

Press Page Down to see the rest of the agreement.

Unity Terms of Service

Last updated: April 27, 2021

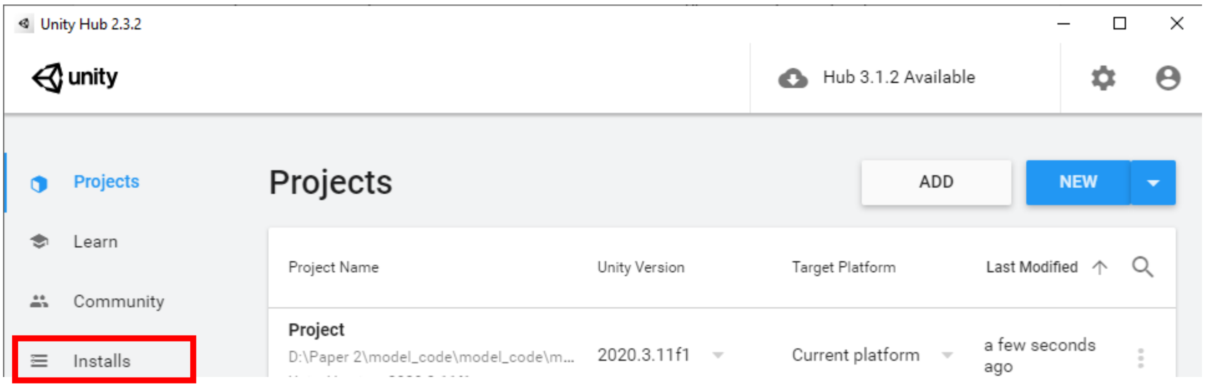
Unity (sometimes "our" or "we") provides game-development and related software (the "Software"), support services and other development-related

If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Unity Hub.

Unity Hub 3.1.2

4) The above window should appear, follow the instructions until the program has finished installing.

5) Once installed, open the UnityHub program and you should see the following window:

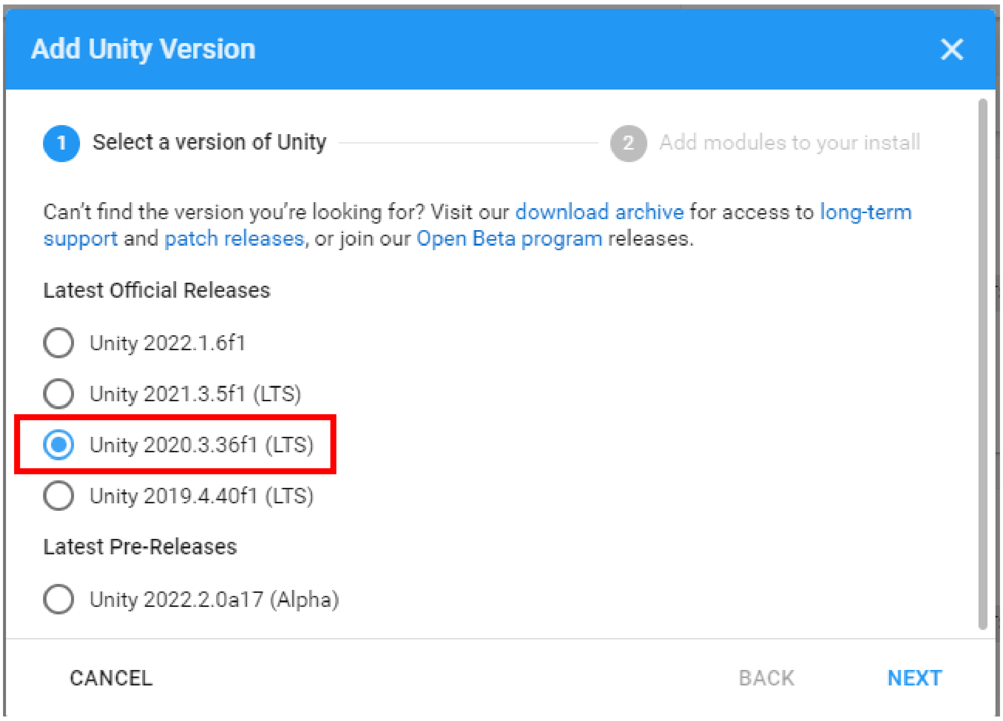


Projects

Project Name	Unity Version	Target Platform	Last Modified
Project D:\Paper 2\model_code\model_code\m...	2020.3.11f1	Current platform	a few seconds ago

6) Click on the **Installs** button on the left then click the **Add** button.

7) Once the following window appears, you should select the latest **2020** version of Unity to install:



Add Unity Version

1 Select a version of Unity — 2 Add modules to your install

Can't find the version you're looking for? Visit our download archive for access to long-term support and patch releases, or join our Open Beta program releases.

Latest Official Releases

○ Unity 2022.1.6f1

○ Unity 2021.3.5f1 (LTS)

● Unity 2020.3.36f1 (LTS)

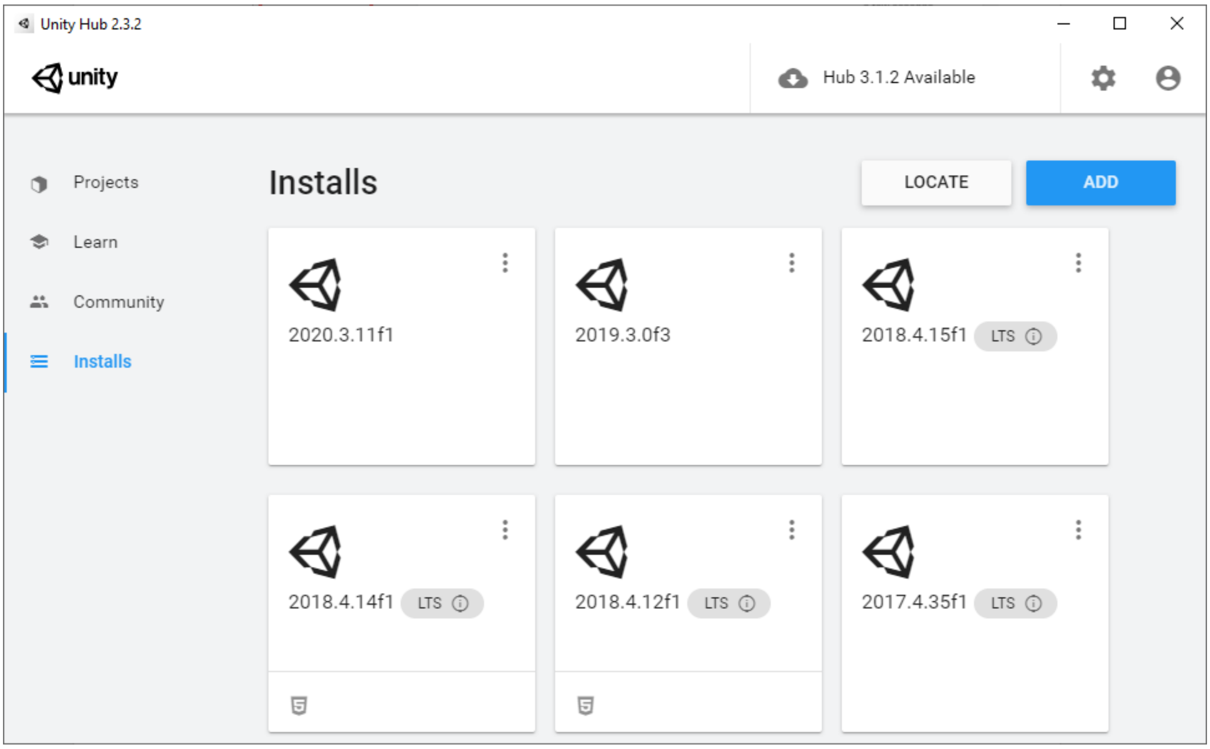
○ Unity 2019.4.40f1 (LTS)

Latest Pre-Releases

○ Unity 2022.2.0a17 (Alpha)

8) Click **Next** then **Done** and this should install the latest Unity 2020 version (which was the version the model was developed with).

9) You should have a window that looks something like this:



But with one install.

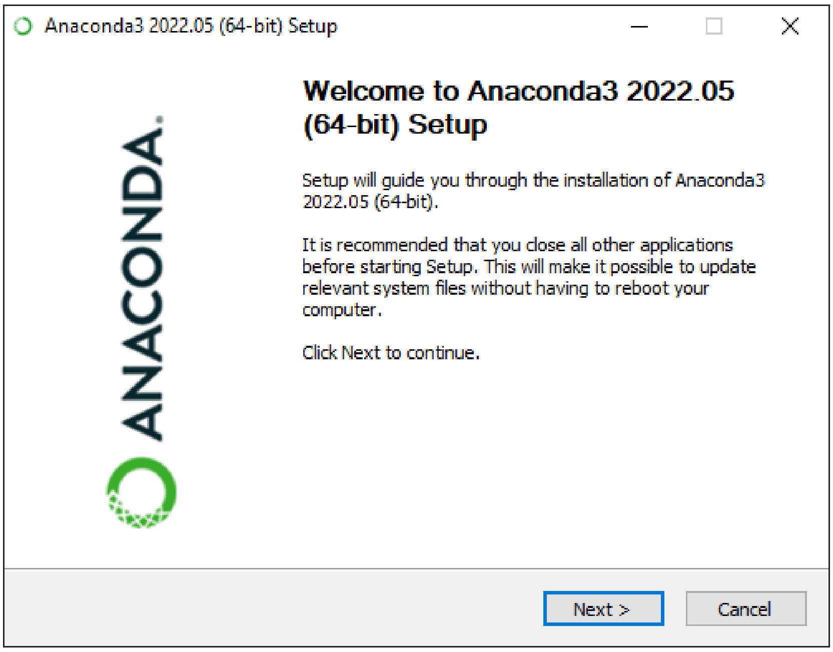
10) Congratulations, Unity has successfully installed. Now we can open and run the model.

If you wish to run the model with the pre-trained RL neural networks, continue to the **Tutorial** section. If you wish to create a conda-environment with the required packages to train your own scenarios, then follow the next steps.

Setup Conda Environment with ML-agents (Optional)

1) Download Anaconda at the following link: https://www.anaconda.com/products/distribution

2) Locate the downloaded file and start the installation procedure like so:



Welcome to Anaconda3 2022.05 (64-bit) Setup

Setup will guide you through the installation of Anaconda3 2022.05 (64-bit).

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

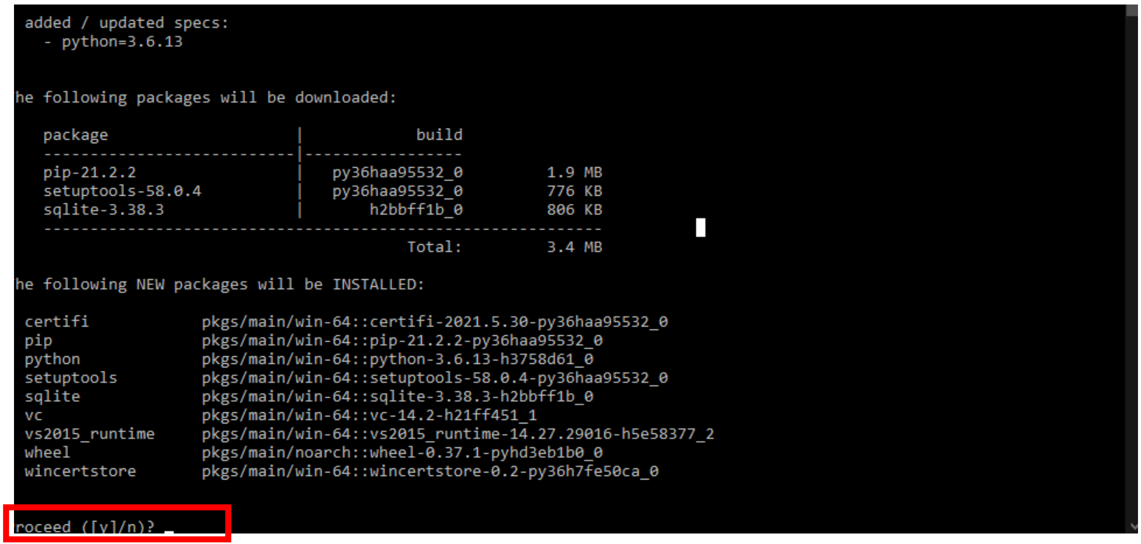
Click Next to continue.

3) Once Anaconda has installed, you can open your "terminal" if using OSX or "Command Prompt" if using Windows.

4) We want to now create a conda environment with the following python version 3.6.13, and the packages ml-agents version 0.25.0, TensorFlow version 2.1.0.

5) Once our command prompt (or terminal) is open, we type the following command and hit enter: **conda create -n MLEnv python=3.6.13**

6) Anaconda should start creating the environment. You should receive the following prompt:



```
 added / updated specs:
   - python=3.6.13


he following packages will be downloaded:

   package                    |            build
   ---------------------------|-----------------
   pip-21.2.2                 |   py36haa95532_0         1.9 MB
   setuptools-58.0.4          |   py36haa95532_0         776 KB
   sqlite-3.38.3              |       h2bbff1b_0         806 KB
   ------------------------------------------------------------
                                          Total:         3.4 MB

he following NEW packages will be INSTALLED:

 certifi            pkgs/main/win-64::certifi-2021.5.30-py36haa95532_0
 pip                pkgs/main/win-64::pip-21.2.2-py36haa95532_0
 python             pkgs/main/win-64::python-3.6.13-h3758d61_0
 setuptools         pkgs/main/win-64::setuptools-58.0.4-py36haa95532_0
 sqlite             pkgs/main/win-64::sqlite-3.38.3-h2bbff1b_0
 vc                 pkgs/main/win-64::vc-14.2-h21ff451_1
 vs2015_runtime     pkgs/main/win-64::vs2015_runtime-14.27.29016-h5e58377_2
 wheel              pkgs/main/noarch::wheel-0.37.1-pyhd3eb1b0_0
 wincertstore       pkgs/main/win-64::wincertstore-0.2-py36h7fe50ca_0


roceed ([y]/n)? _
```

7) Saying Proceed y/n, at this point type **y** and hit enter, this tells conda to install the required packages to create a conda environment successfully with our requested python version.

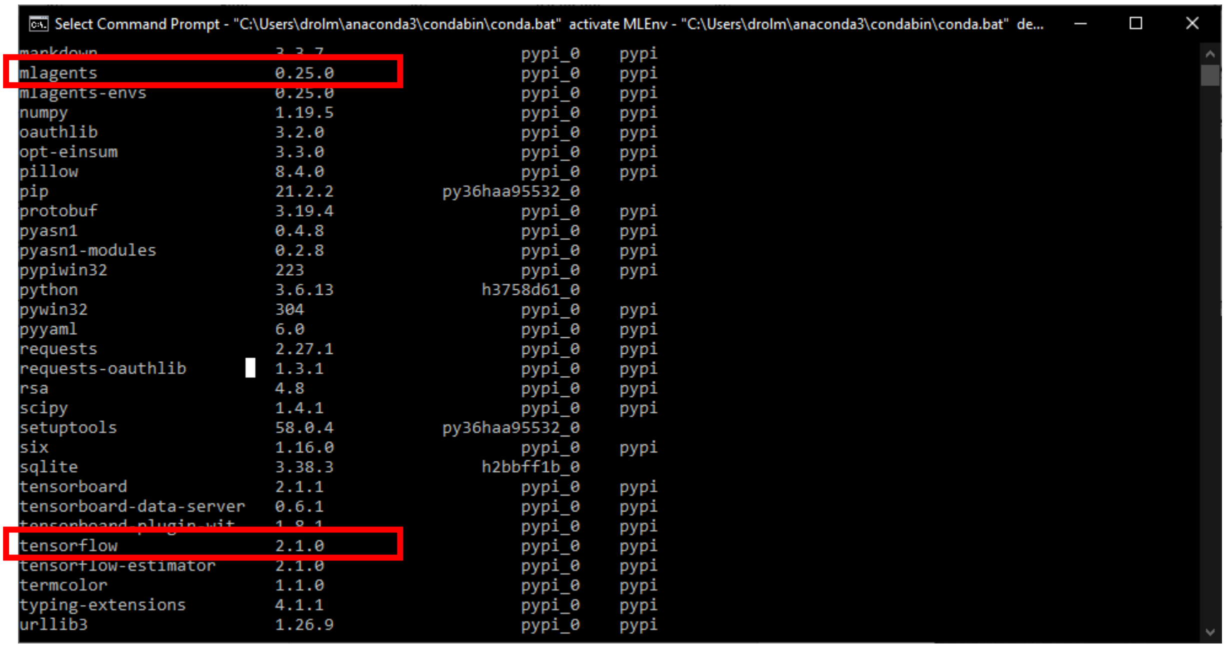
8) Now we can activate our newly created environment, type: **conda activate MLEnv2** and hit enter.

9) You should see (MLEnv2) appear in your command prompt window. This means our environment has successfully installed and activated. We can now install the necessary packages to train our offender agents.

10) Once the MLEnv2 environment has been activated, type the following to install our packages type the following to install TensorFlow and hit enter:
pip install tensorflow==2.1.0

11) Once TensorFlow has installed, we should now install ml-agents by typing: **pip install mlagents==0.25.0** and then install pytorch by typing **pip install torch==1.7.1**

12) Once the packages have installed, if we type: **conda list** and hit enter, we should be able to see our installed packages on the newly created MLEnv2 conda environment.


Select Command Prompt - "C:\Users\drolm\anaconda3\condabin\conda.bat" activate MLEnv - "C:\Users\drolm\anaconda3\condabin\conda.bat" de...

```
markdown                  3.3.7                    pypi_0    pypi
mlagents                  0.25.0                   pypi_0    pypi
mlagents-envs             0.25.0                   pypi_0    pypi
numpy                     1.19.5                   pypi_0    pypi
oauthlib                  3.2.0                    pypi_0    pypi
opt-einsum                3.3.0                    pypi_0    pypi
pillow                    8.4.0                    pypi_0    pypi
pip                       21.2.2           py36haa95532_0
protobuf                  3.19.4                   pypi_0    pypi
pyasn1                    0.4.8                    pypi_0    pypi
pyasn1-modules            0.2.8                    pypi_0    pypi
pypiwin32                 223                      pypi_0    pypi
python                    3.6.13               h3758d61_0
pywin32                   304                      pypi_0    pypi
pyyaml                    6.0                      pypi_0    pypi
requests                  2.27.1                   pypi_0    pypi
requests-oauthlib         1.3.1                    pypi_0    pypi
rsa                       4.8                      pypi_0    pypi
scipy                     1.4.1                    pypi_0    pypi
setuptools                58.0.4           py36haa95532_0
six                       1.16.0                   pypi_0    pypi
sqlite                    3.38.3               h2bbff1b_0
tensorboard               2.1.1                    pypi_0    pypi
tensorboard-data-server   0.6.1                    pypi_0    pypi
tensorboard-plugin-wit    1.8.1                    pypi_0    pypi
tensorflow                2.1.0                    pypi_0    pypi
tensorflow-estimator      2.1.0                    pypi_0    pypi
termcolor                 1.1.0                    pypi_0    pypi
typing-extensions         4.1.1                    pypi_0    pypi
urllib3                   1.26.9                   pypi_0    pypi
```

13) We should see the two main packages we need with the correct versions.

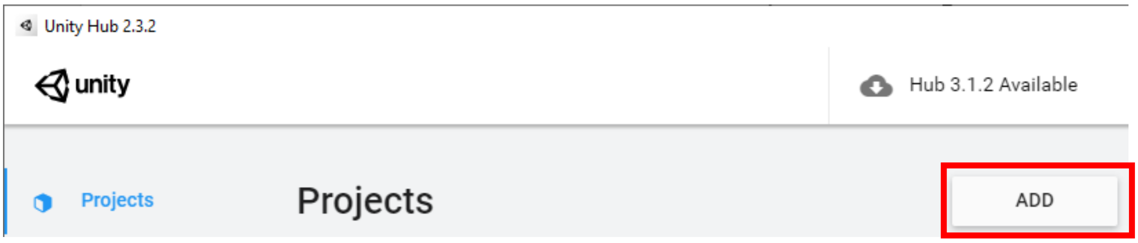
14) Now that we have installed our conda environment, we can train the model with our own environment configuration to test a hypothesis. Complete the next **Tutorial** section first to become familiar with the model then try out the **RL Training Tutorial** next.

15) You can close your conda environment by typing **conda deactivate** and hit enter. Whenever you want to re-use it, re-open your command prompt(terminal) and type **conda activate MLEnv2** then hit enter.

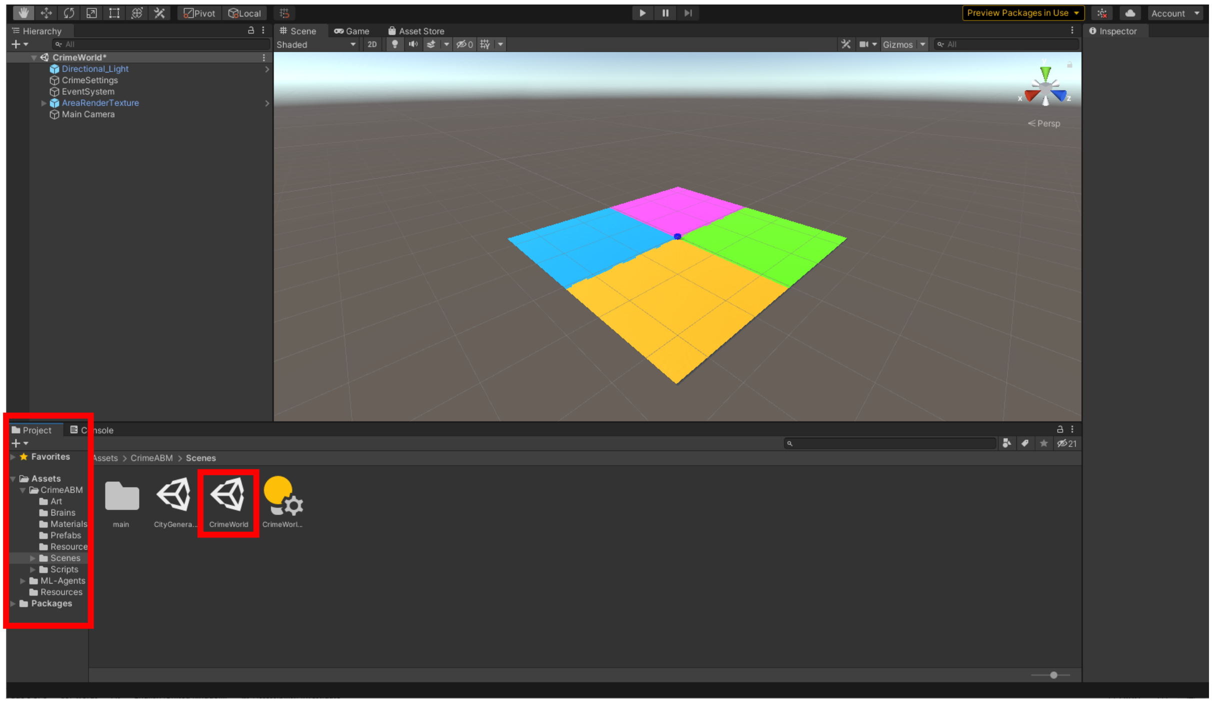
Tutorial

Now that we have installed UnityHub, and downloaded the model locally, we can open the model and run a simple scenario.

1) Open UnityHub and click **Add**:

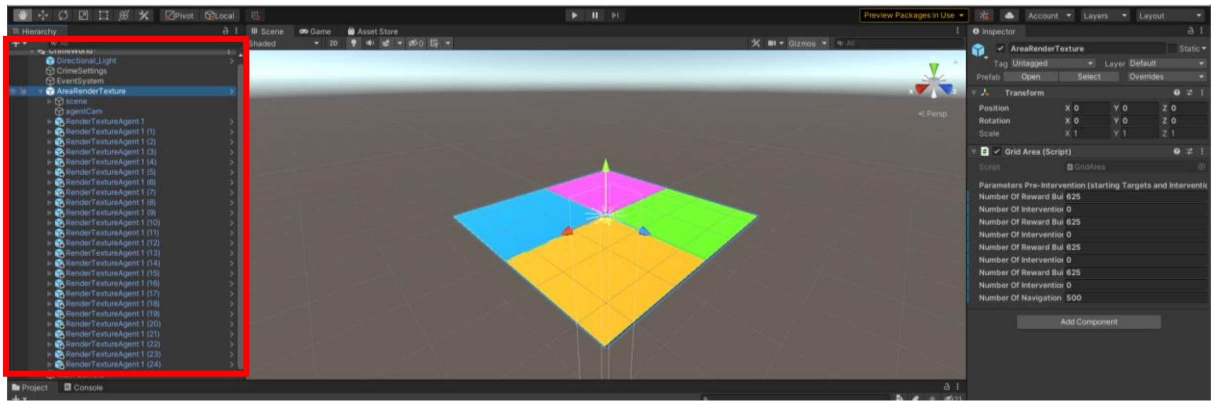


2) Then locate the **Project** folder within the downloaded model document, the file path should look something like this: **..\model_code\model_code\ml-agents\Project**
3) Open this Project file and wait till Unity opens the model. You should see a similar window to the one below:



4) If the window is not the same, click on **Scenes** in the project window (bottom left) and select **CrimeABM** then double click on the **CrimeWorld** scene to open the starting configuration.

5) Now that the model environment is open, let us take a look at our agents and set them up, click on AreaRenderTexture object in the Hierarchy window (top left)

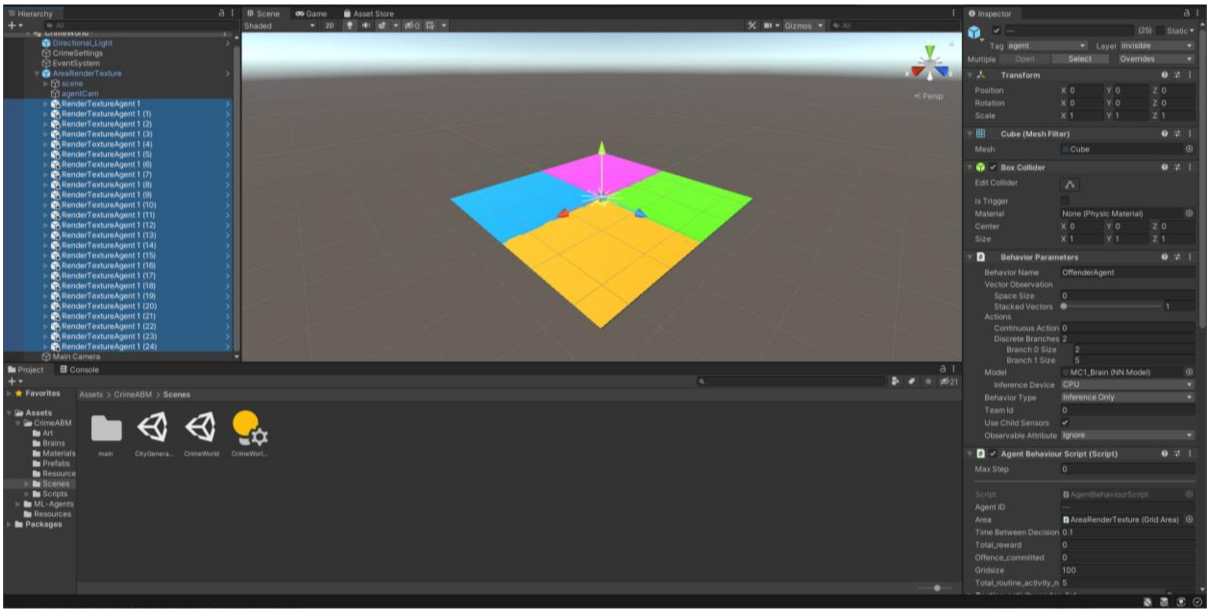


AreaRenderTexture
Tag Untagged Layer Default
Prefab Open Select Overrides

Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Grid Area (Script)
Script GridArea
Parameters Pre-Intervention (starting Targets and Interventic
Number Of Reward Bui 625
Number Of Interventio 0
Number Of Reward Bui 625
Number Of Interventio 0
Number Of Reward Bui 625
Number Of Interventio 0
Number Of Reward Bui 625
Number Of Interventio 0
Number Of Navigation 500

6) Our 25 offender objects can be observed, we can now click on them and set their neural networks up for the scenario we wish to run, let us select all the 25 agent objects select one then click shift then click the last one to select all:



(25) Static
Tag agent Layer invisible
Multiple Open Select Overrides

Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Cube (Mesh Filter)
Mesh Cube

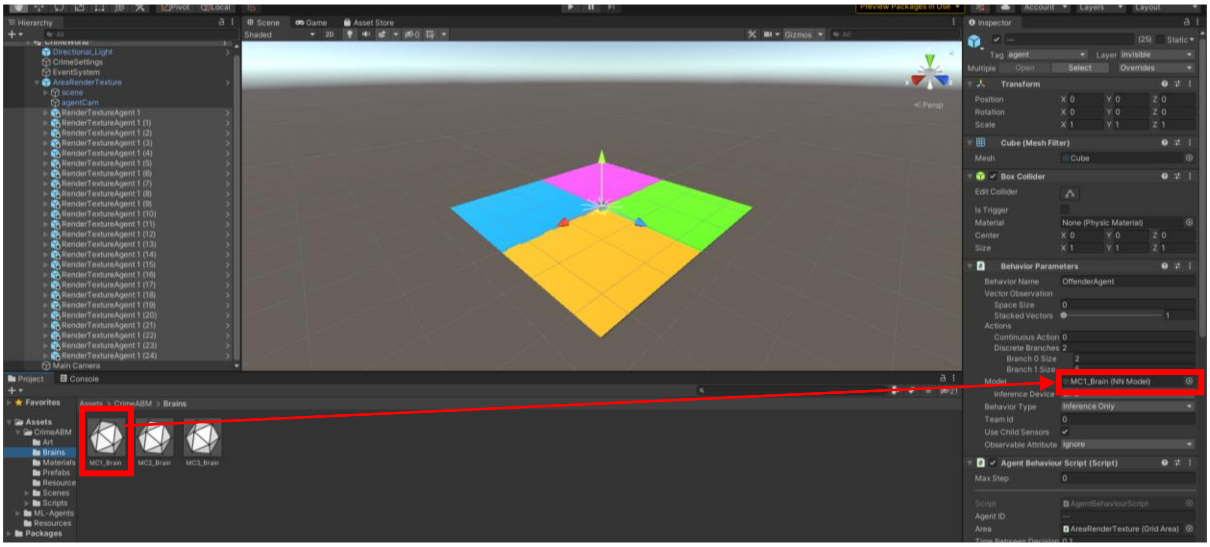
Box Collider
Edit Collider
Is Trigger
Material None (Physic Material)
Center X 0 Y 0 Z 0
Size X 1 Y 1 Z 1

Behavior Parameters
Behavior Name OffenderAgent
Vector Observation
Space Size 0
Stacked Vectors 1
Actions
Continuous Action 0
Discrete Branches 2
Branch 0 Size 2
Branch 1 Size 5
Model MC1_Brain (NN Model)
Inference Device CPU
Behavior Type Inference Only
Team Id 0
Use Child Sensors ✓
Observable Attribute Ignore

Agent Behaviour Script (Script)
Max Step 0
Script AgentBehaviourScript
Agent ID —
Area AreaRenderTexture (Grid Area)
Time Between Decision 0.1
Total_reward 0
Offence_committed 0
Gridsize 100
Total_routine_activity_n 5

7) On the right, we can see the Inspector window which has all the settings for each agent game object. We can now add the “Brain” for scenario 1 to these agents by

selecting the “Brains” folder under “CrimeABM” in the bottom left:



Behavior Parameters

Behavior Name OffenderAgent

Vector Observation

Space Size 0

Stacked Vectors 1

Actions

Continuous Action 0

Discrete Branches 2

Branch 0 Size 2

Model MC1_Brain (NN Model)

Behavior Type Inference Only

Team Id 0

Use Child Sensors ✓

Observable Attribute Ignore

Agent Behaviour Script (Script)

Max Step 0

Script AgentBehaviourScript

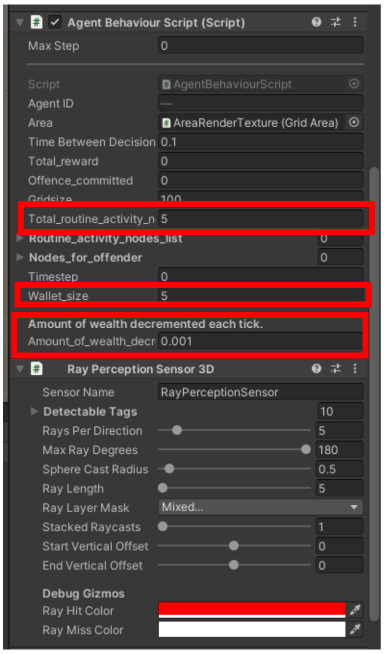
Area AreaRenderTexture (Grid Area)

8) We can then select **MC1_Brain** and drag-drop it into the Model field on the right. This sets our Offender agents with the trained neural network from experiment 1. In this experiment we trained offenders in an environment that contained targets with uniformly distributed rewards of 1. This means offenders found more rewarding targets and subsequently offended more often, however, once interventions were applied, offenders focussed their attentions to locations that did not have interventions. *These three neural networks will be described in the published article in detail with results.*

9) Behaviour type just under model should be set to **Inference** this means the agents use their trained neural nets to make decisions, when training the model, you must remove the **MC1_Brain** by clicking the little circle on the right of it and setting **BehaviourType** to **Default**.

10) If we scroll down in the **Inspector** window, we can see all the other parameters we can change for our offender agents, where we can set the number of routine activity nodes, the wallet size also known as the TargetCumulativeReward which we hope the agent to achieve as a way of expressing satisfaction and lastly, Amount of wealth decremented each tick(timestep) this is the amount of losses the agent has

conceptualising expenses such as money:



Agent Behaviour Script (Script)

Max Step	0
Script	AgentBehaviourScript
Agent ID	—
Area	AreaRenderTexture (Grid Area)
Time Between Decision	0.1
Total_reward	0
Offence_committed	0
Gridsize	100
Total_routine_activity_n	5
Routine_activity_nodes_list	0
Nodes_for_offender	0
Timestep	0
Wallet_size	5

Amount of wealth decremented each tick.

Amount_of_wealth_decr	0.001

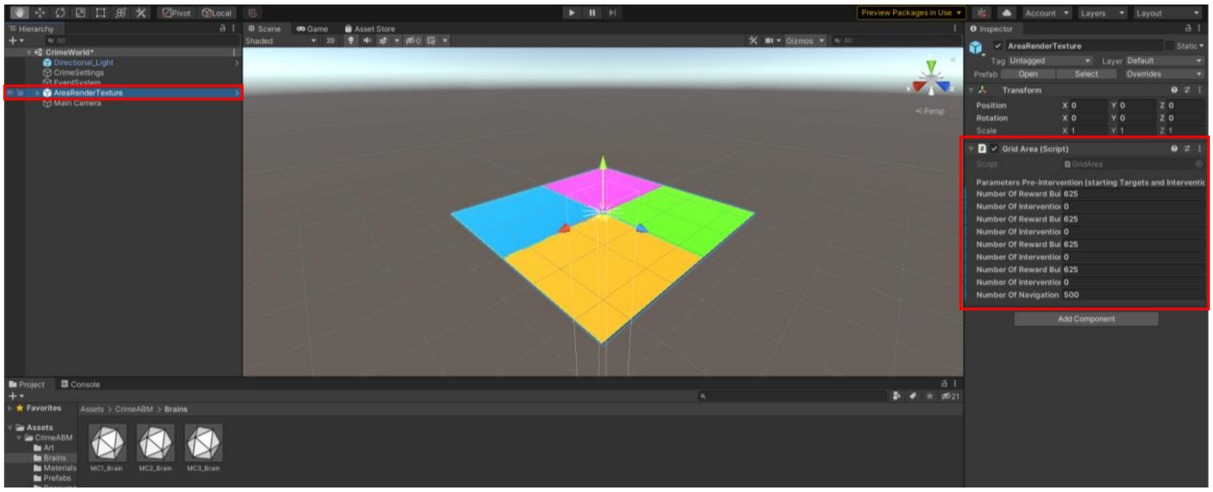
Ray Perception Sensor 3D

Sensor Name	RayPerceptionSensor
Detectable Tags	10
Rays Per Direction	5
Max Ray Degrees	180
Sphere Cast Radius	0.5
Ray Length	5
Ray Layer Mask	Mixed...
Stacked Raycasts	1
Start Vertical Offset	0
End Vertical Offset	0

Debug Gizmos

Ray Hit Color	🟥
Ray Miss Color	⬜

11) Now that we have set out agents up, to run the trained neural network from experiment 1, we can set up our environment parameters. If we click on **AreaRenderTexture** on the left in the Hierarchy window:



AreaRenderTexture
Tag Untagged Layer Default
Prefab Open Select Overrides

Transform

Position	X 0	Y 0	Z 0
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

Grid Area (Script)

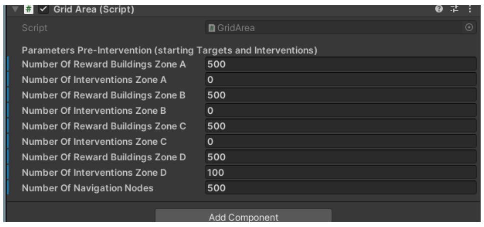
Script GridArea

Parameters Pre-Intervention (starting Targets and Interventic

Number Of Reward Bui	625
Number Of Interventio	0
Number Of Reward Bui	625
Number Of Interventio	0
Number Of Reward Bui	625
Number Of Interventio	0
Number Of Reward Bui	625
Number Of Interventio	0
Number Of Navigation	500

12) We should now see the agents disappear and the **Inspector** window show the **GridArea** component. Here we set the number of Targets known as **RewardingBuildings** in each of the four spatial areas (vulnerable buildings). For Zones A to D: where **Green** and **Orange** are **A** and **B** and **Pink** and **Blue** are **C** and **D**. I will set **500 Targets** in each of the **four Zones** and **100 Interventions** in **Zone D (Blue).** Lastly, we will set 500 navigational nodes.

13) We should have parameter values like the below:



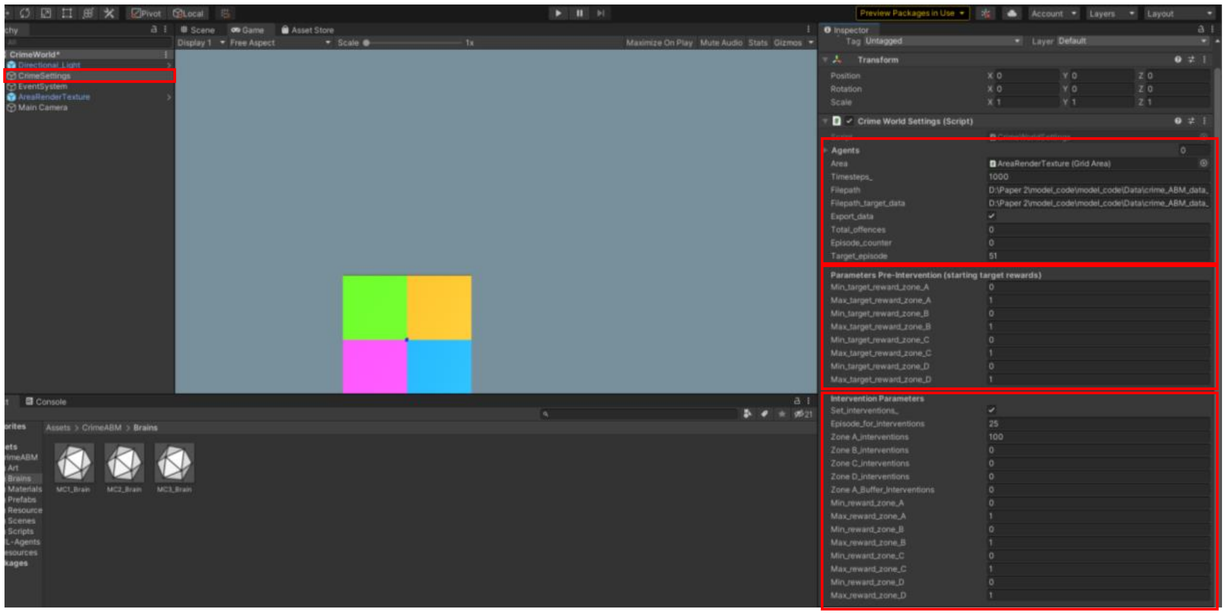
Grid Area (Script)

Script GridArea

Parameters Pre-Intervention (starting Targets and Interventions)

Number Of Reward Buildings Zone A	500
Number Of Interventions Zone A	0
Number Of Reward Buildings Zone B	500
Number Of Interventions Zone B	0
Number Of Reward Buildings Zone C	500
Number Of Interventions Zone C	0
Number Of Reward Buildings Zone D	500
Number Of Interventions Zone D	100
Number Of Navigation Nodes	500

14) Now our pre-interventions setting (initial setting) of the model is ready. We can set up the experiment. Click on **CrimeSettings** in the **Hierarchy** window (top left):



Position	X 0	Y 0	Z 0
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

Crime World Settings (Script)

Agents	0
Area	AreaRenderTexture (Grid Area)
Timesteps_	1000
Filepath	D:\Paper 2\model_code\model_code\Data\crime_ABM_data_
Filepath_target_data	D:\Paper 2\model_code\model_code\Data\crime_ABM_data_
Export_data	✓
Total_offences	0
Episode_counter	0
Target_episode	51
Parameters Pre-Intervention (starting target rewards)	
Min_target_reward_zone_A	0
Max_target_reward_zone_A	1
Min_target_reward_zone_B	0
Max_target_reward_zone_B	1
Min_target_reward_zone_C	0
Max_target_reward_zone_C	1
Min_target_reward_zone_D	0
Max_target_reward_zone_D	1
Intervention Parameters	
Set_interventions_	✓
Episode_for_interventions	25
Zone A_interventions	100
Zone B_interventions	0
Zone C_interventions	0
Zone D_interventions	0
Zone A_Buffer_Interventions	0
Min_reward_zone_A	0
Max_reward_zone_A	1
Min_reward_zone_B	0
Max_reward_zone_B	1
Min_reward_zone_C	0
Max_reward_zone_C	1
Min_reward_zone_D	0
Max_reward_zone_D	1

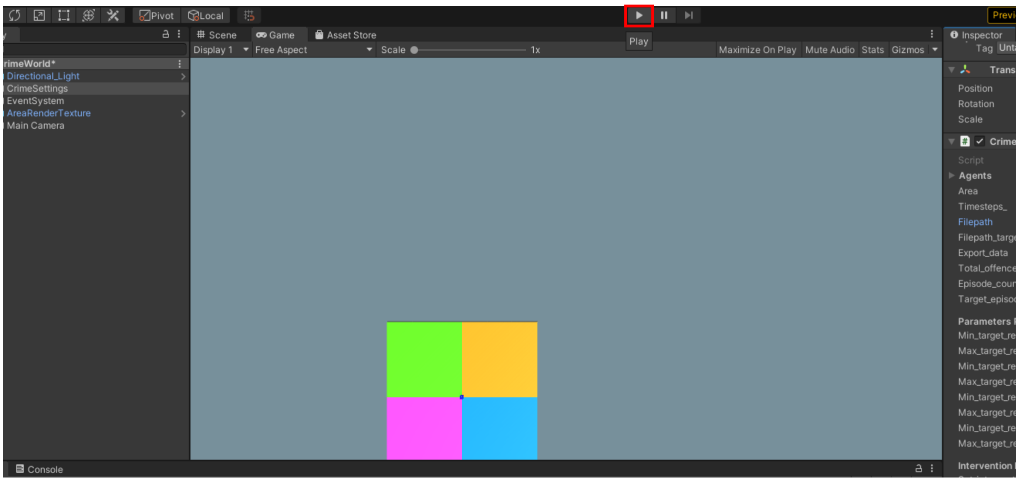
15) Our parameters for this component (Inspector window, right) are split into three sections. The first section sets the total number of timesteps(ticks) each simulation should run for, in our case we set it to **1000**, the complete file path of the location we want our output data to be saved to **including the file name and type i.e., .txt for example:** D:\model_code\model_code\Data\crime_ABM_data_analysis\Experiment_Data\MC2_Data\MC1_Run_1.txt the second path outputs data about each **Target** i.e., location, amount of reward and so on to use for spatial analysis. Make sure this file path is also like the above but with a different file name i.e., **MC1_Run_1_TargetData.txt** if we tick **Export_Data** then we want output data, lastly the **Target_episode** is the number of iterations we want to run our simulation, we set this as 50 + 1 which Unity requires. So however, many times you wish to run your simulation, make sure to add 1. **Total_Offences** and **Episode_counter** is for viewing purposes only; we see the total number of offences made and the current episode the model is running.

16) The second section allows you to set the reward distributions for each target pre-intervention. Here we set the minimum and maximum reward for each Zone. For our tutorial, we set the distribution to be random between [0, 1] inclusive at every Zone. We can set this to [1, 1] or [0, 0] which means a reward of 1 at every target or no reward at any target.

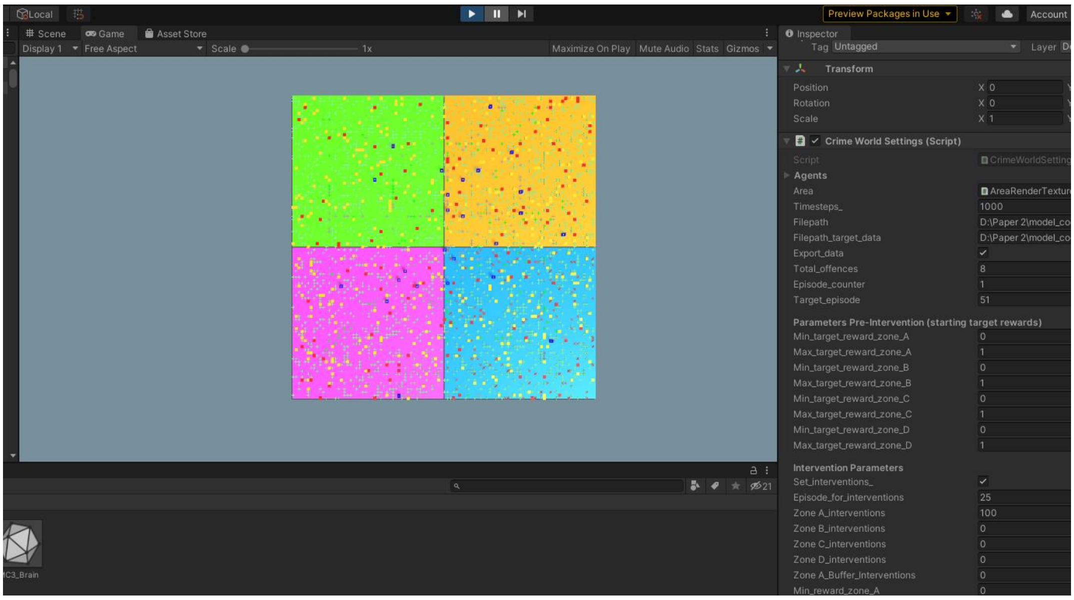
17) The intervention parameters allow us to set some interventions during the simulation run at a specific **Episode**. If **Set_interventions_** is ticked, then we must fill out the below parameter values else leave them. The **Episode_for_intervention** parameter defines the episode in which the interventions are applied, for our example we set this to be **25**. We can set the number of interventions introduced in a particular Zone, I set there to be 100 spatial interventions in Zone A (**Green).** Ignore **ZoneABufferIntervention** this was purely a testing parameter. Lastly, we can

change the reward distribution for a specific zone as well. For this experiment we leave them the way we set out rewards in the **pre-intervention** parameters.

18) Now we are ready to run our model experiments. We click on the **Play** button at the top:



19) We should start to see the model running, in the **Total_offences** field we see 8 offences have taken place at episode 1:



Position X 0
Rotation X 0
Scale X 1

Crime World Settings (Script)

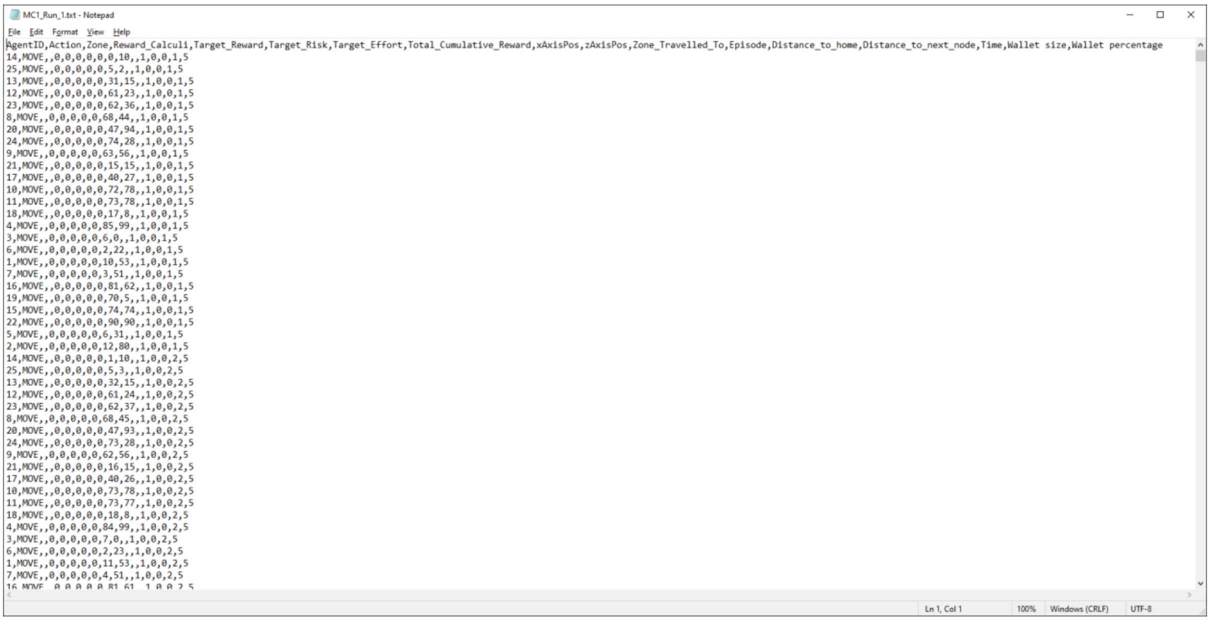
Script CrimeWorldSetting
Agents
Area AreaRenderTextur
Timesteps_ 1000
Filepath D:\Paper 2\model_co
Filepath_target_data D:\Paper 2\model_co
Export_data ✓
Total_offences 8
Episode_counter 1
Target_episode 51

Parameters Pre-Intervention (starting target rewards)
Min_target_reward_zone_A 0
Max_target_reward_zone_A 1
Min_target_reward_zone_B 0
Max_target_reward_zone_B 1
Min_target_reward_zone_C 0
Max_target_reward_zone_C 1
Min_target_reward_zone_D 0
Max_target_reward_zone_D 1

Intervention Parameters
Set_interventions_ ✓
Episode_for_interventions 25
Zone A_interventions 100
Zone B_interventions 0
Zone C_interventions 0
Zone D_interventions 0
Zone A_Buffer_Interventions 0
Min_reward_zone_A 0

20) Once our experiment is up, the model should automatically stop, if you look in the directory you saved your output data to, you should see two .txt files. If you open

the file, it should look something like this:



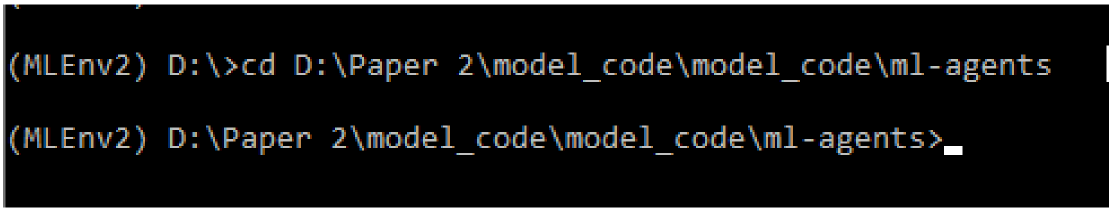
```
AgentID,Action,Zone,Reward_Calculi,Target_Reward,Target_Risk,Target_Effort,Total_Cumulative_Reward,xAxisPos,zAxisPos,Zone_Travelled_To,Episode,Distance_to_home,Distance_to_next_node,Time,Wallet size,Wallet percentage
14,MOVE,,0,0,0,0,0,10,,1,0,0,1,5
25,MOVE,,0,0,0,0,0,5,2,,1,0,0,1,5
13,MOVE,,0,0,0,0,0,31,15,,1,0,0,1,5
12,MOVE,,0,0,0,0,0,61,23,,1,0,0,1,5
23,MOVE,,0,0,0,0,0,62,36,,1,0,0,1,5
8,MOVE,,0,0,0,0,0,68,44,,1,0,0,1,5
20,MOVE,,0,0,0,0,0,47,94,,1,0,0,1,5
24,MOVE,,0,0,0,0,0,74,28,,1,0,0,1,5
9,MOVE,,0,0,0,0,0,63,56,,1,0,0,1,5
21,MOVE,,0,0,0,0,0,15,15,,1,0,0,1,5
17,MOVE,,0,0,0,0,0,40,27,,1,0,0,1,5
10,MOVE,,0,0,0,0,0,72,78,,1,0,0,1,5
11,MOVE,,0,0,0,0,0,73,78,,1,0,0,1,5
18,MOVE,,0,0,0,0,0,17,8,,1,0,0,1,5
4,MOVE,,0,0,0,0,0,85,99,,1,0,0,1,5
3,MOVE,,0,0,0,0,0,6,0,,1,0,0,1,5
6,MOVE,,0,0,0,0,0,2,22,,1,0,0,1,5
1,MOVE,,0,0,0,0,0,10,53,,1,0,0,1,5
7,MOVE,,0,0,0,0,0,3,51,,1,0,0,1,5
16,MOVE,,0,0,0,0,0,81,62,,1,0,0,1,5
19,MOVE,,0,0,0,0,0,70,5,,1,0,0,1,5
15,MOVE,,0,0,0,0,0,74,74,,1,0,0,1,5
22,MOVE,,0,0,0,0,0,90,90,,1,0,0,1,5
5,MOVE,,0,0,0,0,0,6,31,,1,0,0,1,5
2,MOVE,,0,0,0,0,0,12,80,,1,0,0,1,5
14,MOVE,,0,0,0,0,0,1,10,,1,0,0,2,5
25,MOVE,,0,0,0,0,0,5,3,,1,0,0,2,5
13,MOVE,,0,0,0,0,0,32,15,,1,0,0,2,5
12,MOVE,,0,0,0,0,0,61,24,,1,0,0,2,5
23,MOVE,,0,0,0,0,0,62,37,,1,0,0,2,5
8,MOVE,,0,0,0,0,0,68,45,,1,0,0,2,5
20,MOVE,,0,0,0,0,0,47,93,,1,0,0,2,5
24,MOVE,,0,0,0,0,0,73,28,,1,0,0,2,5
9,MOVE,,0,0,0,0,0,62,56,,1,0,0,2,5
21,MOVE,,0,0,0,0,0,16,15,,1,0,0,2,5
17,MOVE,,0,0,0,0,0,40,26,,1,0,0,2,5
10,MOVE,,0,0,0,0,0,73,78,,1,0,0,2,5
11,MOVE,,0,0,0,0,0,73,77,,1,0,0,2,5
18,MOVE,,0,0,0,0,0,18,8,,1,0,0,2,5
4,MOVE,,0,0,0,0,0,84,99,,1,0,0,2,5
3,MOVE,,0,0,0,0,0,7,0,,1,0,0,2,5
6,MOVE,,0,0,0,0,0,2,23,,1,0,0,2,5
1,MOVE,,0,0,0,0,0,11,53,,1,0,0,2,5
7,MOVE,,0,0,0,0,0,4,51,,1,0,0,2,5
16,MOVE,,0,0,0,0,0,81,61,,1,0,0,2,5
```

21) Congratulations, you successfully ran an experiment using the agent-based reinforcement learning model.

RL Training Tutorial

To train our agents using a specific environmental configuration we must first familiarise ourselves with the above tutorial and model. Open your command prompt(terminal) and activate the conda environment we created earlier **MLEnv2.**

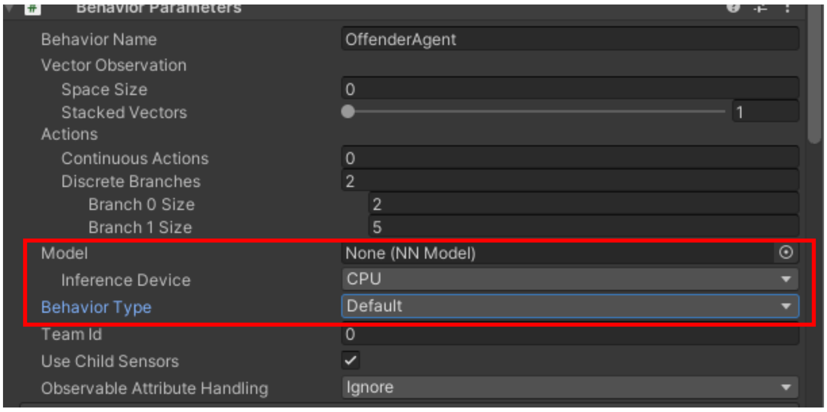
1) Once the environment is activated, navigate to the **..model_code\model_code\ml-agents** directory:



```
(MLEnv2) D:\>cd D:\Paper 2\model_code\model_code\ml-agents

(MLEnv2) D:\Paper 2\model_code\model_code\ml-agents>_
```

2) Make sure the **Model** parameter is empty and **Behaviour type** is set to **Default** for the offender agents:



Behavior Parameters

Behavior Name: OffenderAgent

Vector Observation
- Space Size: 0
- Stacked Vectors: 1

Actions
- Continuous Actions: 0
- Discrete Branches: 2
  - Branch 0 Size: 2
  - Branch 1 Size: 5

Model: None (NN Model)
- Inference Device: CPU

Behavior Type: Default

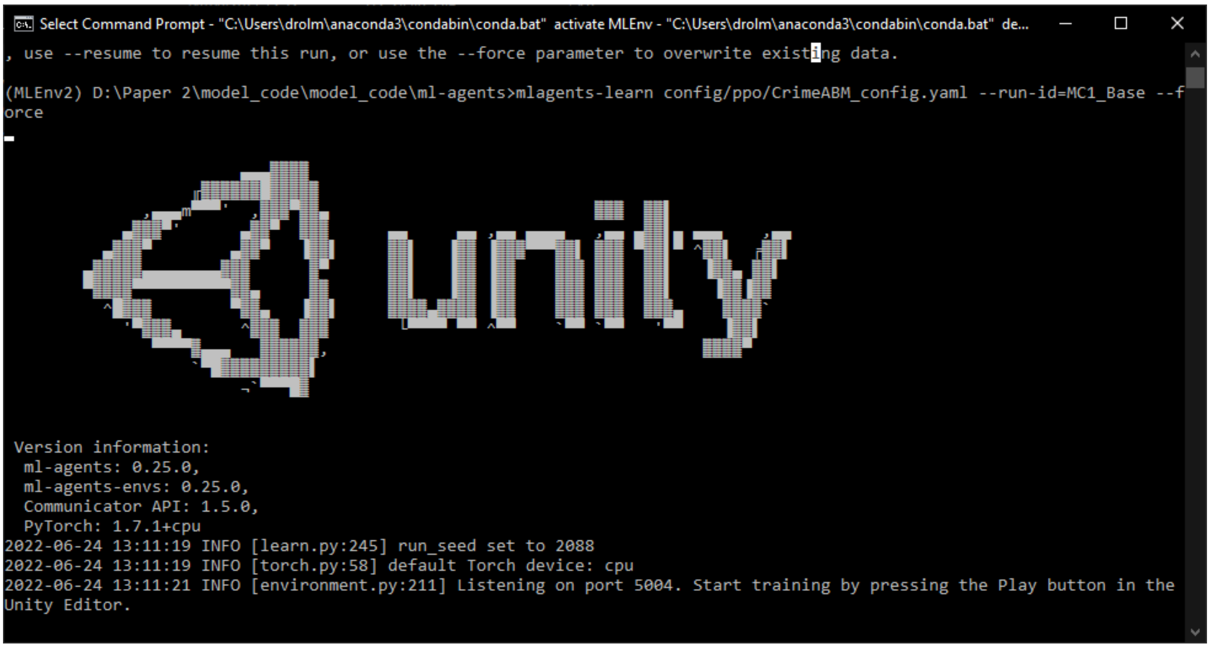
Team Id: 0

Use Child Sensors: ✓

Observable Attribute Handling: Ignore

3) If you have set the model up through tweaking the parameters in the **Tutorial** section for the training, we can now run our training command: **mlagents-learn config/ppo/CrimeABM_config.yaml --run-id=MC1_Base**

4) In the above training command, we are telling the ml-agents package to start learning, by using the hyperparameters we defined in the ../config/ppo/ directory and we call our “Brain” the **MC1_Base**



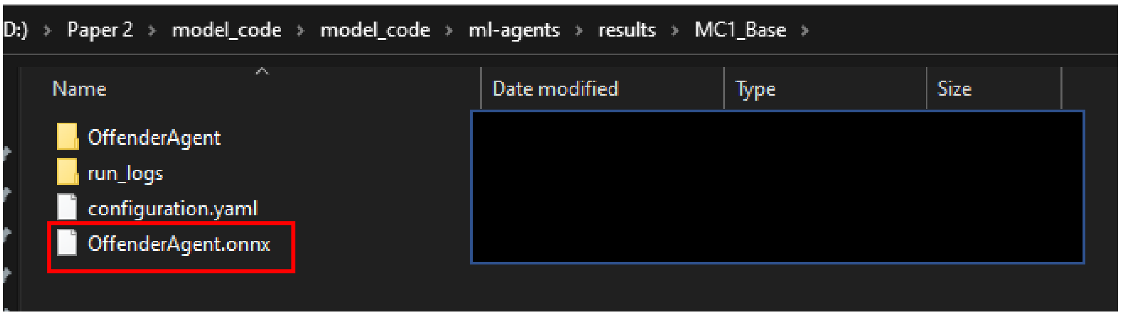
Select Command Prompt - "C:\Users\drolm\anaconda3\condabin\conda.bat" activate MLEnv - "C:\Users\drolm\anaconda3\condabin\conda.bat" de...

```
, use --resume to resume this run, or use the --force parameter to overwrite existing data.

(MLEnv2) D:\Paper 2\model_code\model_code\ml-agents>mlagents-learn config/ppo/CrimeABM_config.yaml --run-id=MC1_Base --f
orce

 Version information:
  ml-agents: 0.25.0,
  ml-agents-envs: 0.25.0,
  Communicator API: 1.5.0,
  PyTorch: 1.7.1+cpu
2022-06-24 13:11:19 INFO [learn.py:245] run_seed set to 2088
2022-06-24 13:11:19 INFO [torch.py:58] default Torch device: cpu
2022-06-24 13:11:21 INFO [environment.py:211] Listening on port 5004. Start training by pressing the Play button in the
Unity Editor.
```

5) Once we see the above window, ml-agents is now waiting for us to press **Play** and start the learning process. Once the training ends, we should find a **MC1_Base** folder appear in the ../results/ directory:



D:) › Paper 2 › model_code › model_code › ml-agents › results › MC1_Base ›

Name	Date modified	Type	Size
OffenderAgent			
run_logs			
configuration.yaml			
OffenderAgent.onnx			

6) The **OffenderAgent.onnx** file is the trained neural network. We can now place this into the model field during testing. To recap this process, look at the **Tutorial** section. Also remember to change the **Behaviour Type** to **Inference**.

END OF DOCUMENT