Robot Arm ML Agent Unity
https://github.com/rkandas/RobotArmMLAgentUnity
https://medium.com/xrpractices/how-to-train-your-robot-arm-fbf5dcd807e1

This project requires ML-Agent 1.0.3 and Python 3.7.

1) Create a conda environment with Python 3.7

```
conda create --name unityenv python=3.7
conda activate unityenv
```

2) Clone the github

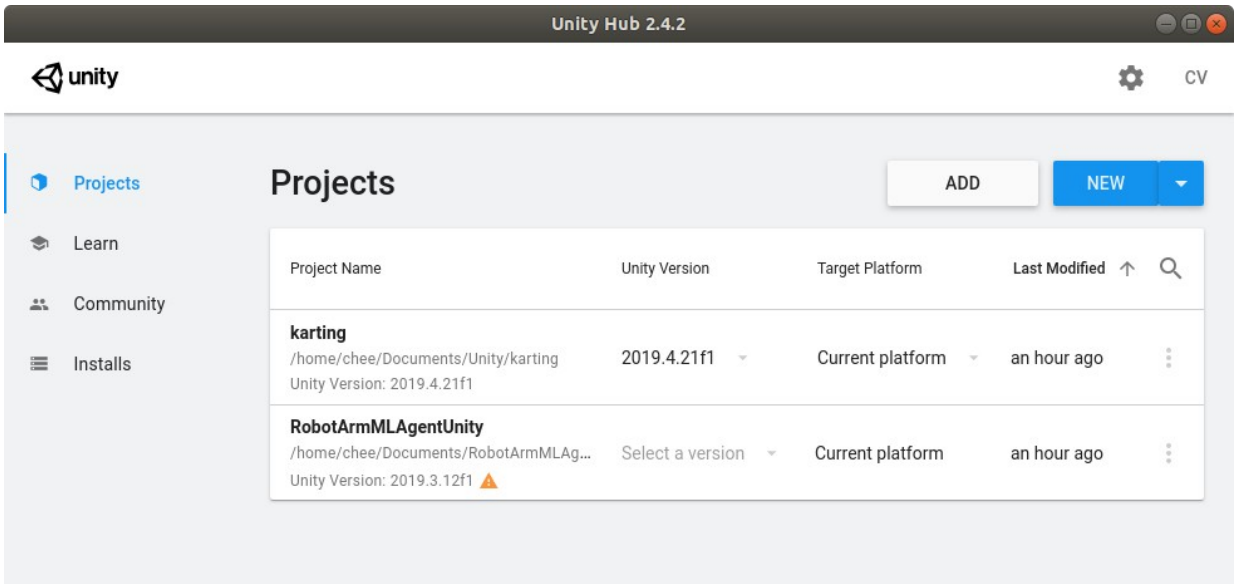
```
cd ~/<where you want to save the project>/
git clone https://github.com/rkandas/RobotArmMLAgentUnity.git
```

3) Install ML-Agents

```
pip install mlagents
```

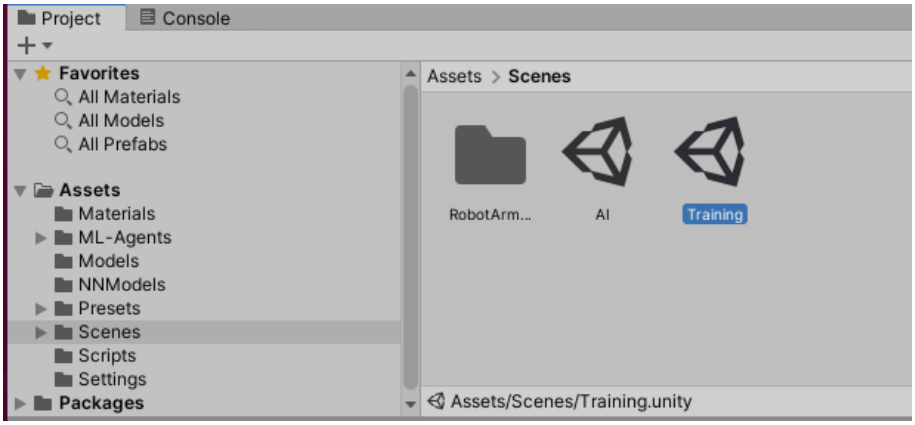
4) Open Unity Hub. **Add** a new project and select the folder **RobotArmMLAgentUnity** that was just downloaded.

5) There might be a warning symbol on the project about the Unity version. You can ignore the warning, or install Unity 2019.3.12f1.

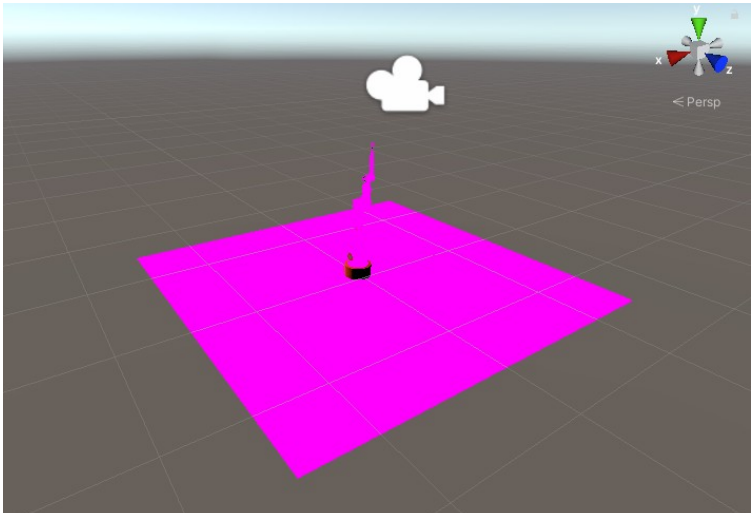


6) When the project is first opened, there will be a few errors. Just import ML Agents by going to **Window** >> P**ackage Manager** and on **ML Agents** click on **See all version** to import version **1.0.3**.

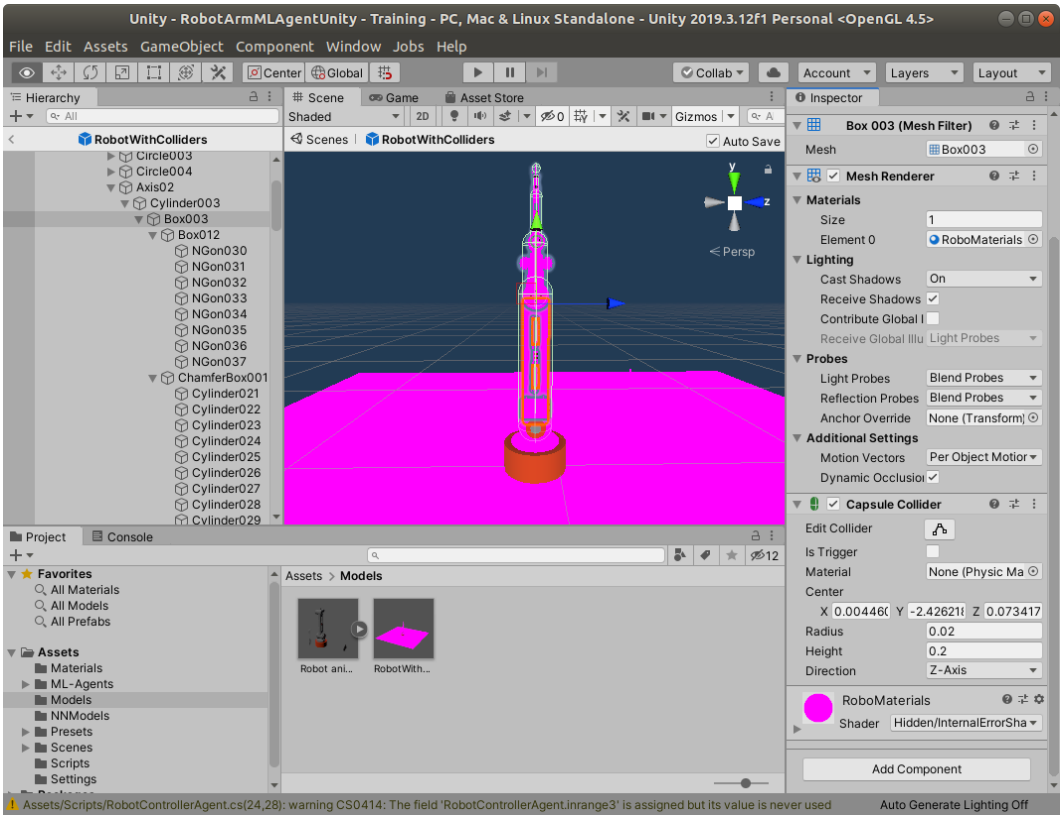
7) In the **Project** window, go to **Assets** >> **Scenes** and open the **Training** scene with 12 robot arms.



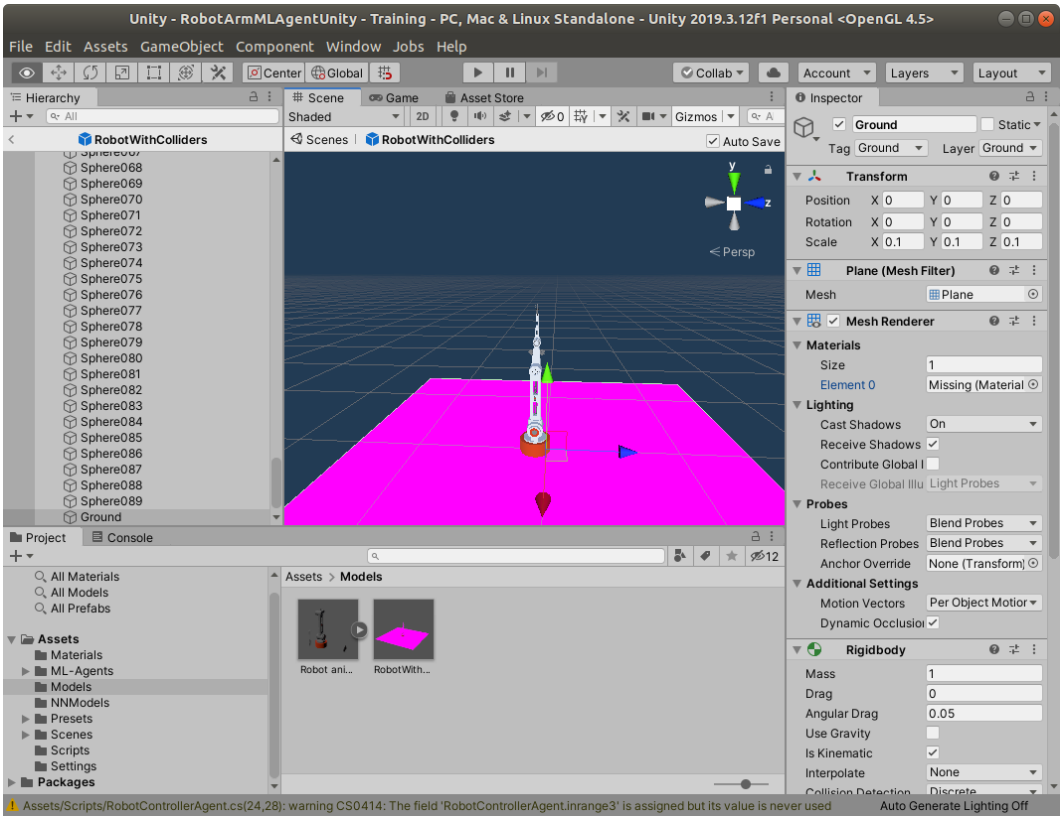
8) An issue will appear where the prefabs are pink like the figure below. If not, go to the step (9). Otherwise, continue.



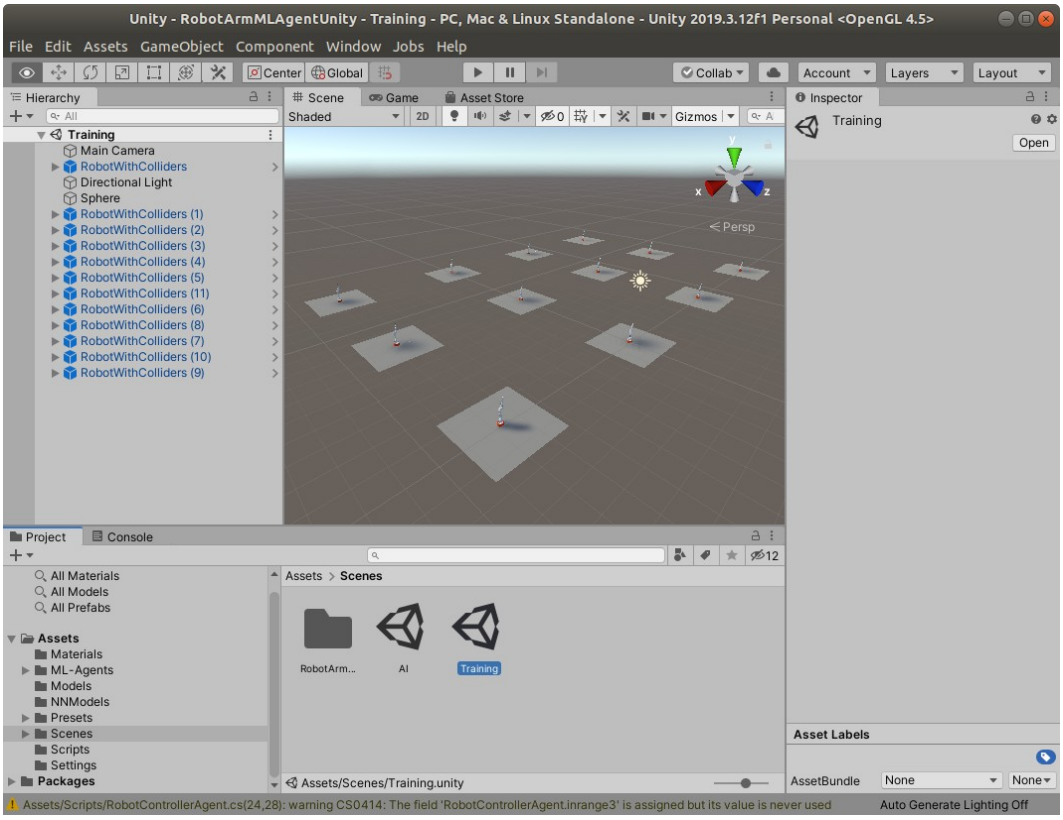
- In the **Project** window, go to **Assets** >> **Models**.
- Double click on **RobotWithColliders.prefab** (shown below)
- Click on the robot arm (it will be highlighted)
- In the **Inspector** >> **Robot Material**, change **Shader** to a shade you prefer. Do the same for other pink items: bottle and grabber.



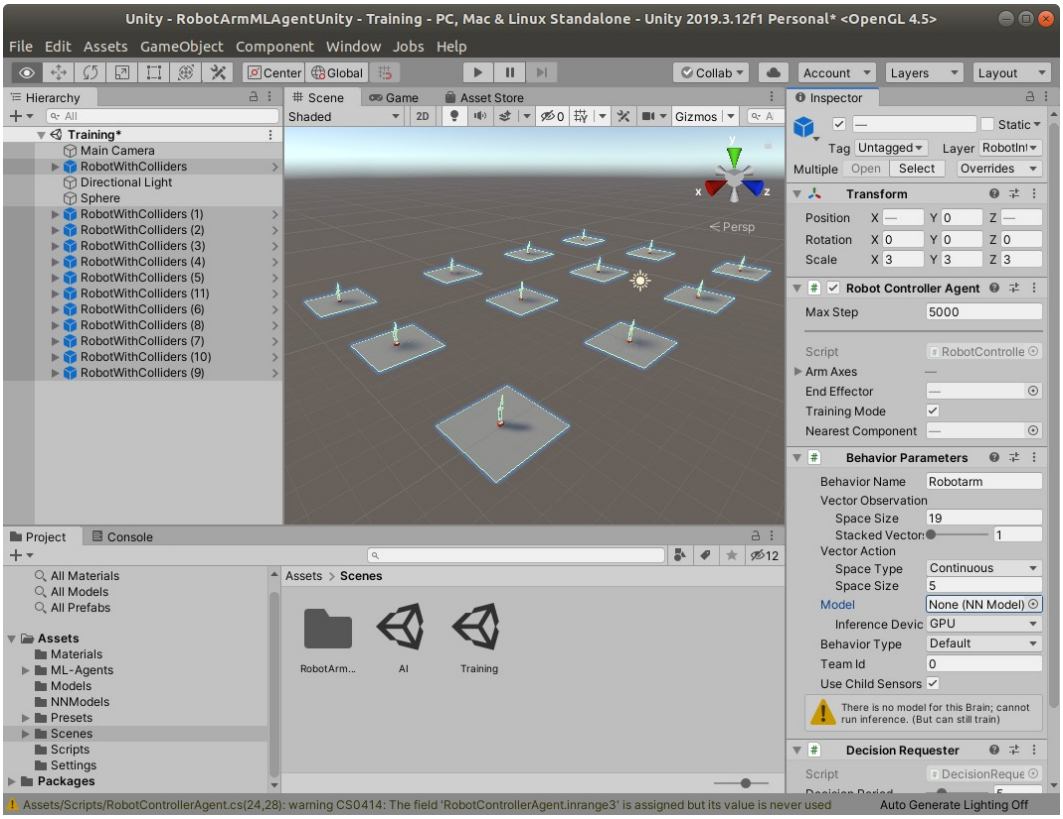
- To change the ground: select the ground and change the material in **Inspector** >> **Mesh Renderer** >> **Materials** >> **Element 0** to another material



After changing the materials/shader, this should also affect the prefabs in the scenes we will use. Go back to **Project** >> **Assets** >> **Scenes** >> **Training** and the robot arm should be the selected color.



9) Select all the **RobotWithColliders** in the **Hierarchy** and change the model to **None** in **Inspector** >> **Behavior** >> **Model**



10) The **trainer_config.yaml** file can be modified to change some of the configurations for **Robotarm**:

batch_size, **hidden_units** : number of nodes per layer in NN
num_layers : number of layers
max_steps : number of training steps (Mine was changed to 1.0e6)

11) On the terminal, start the training process with the command:

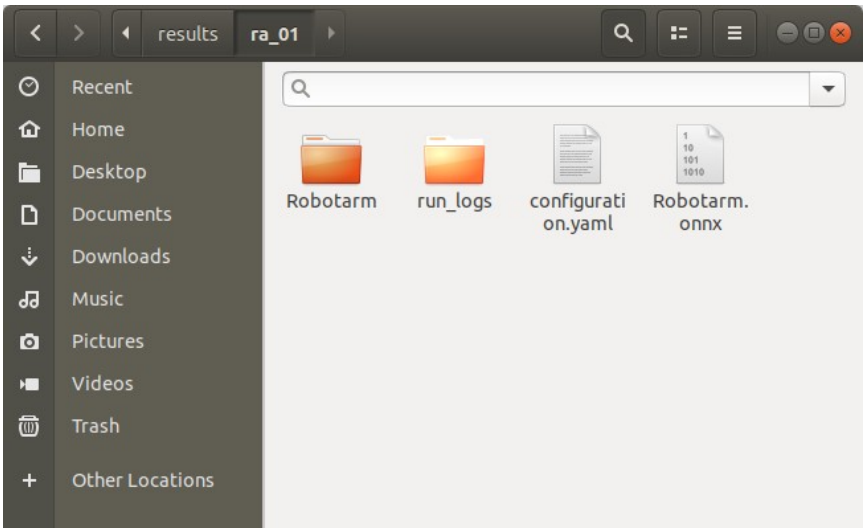
```
mlagents-learn ./trainer_config.yaml --run-id ra_01
```

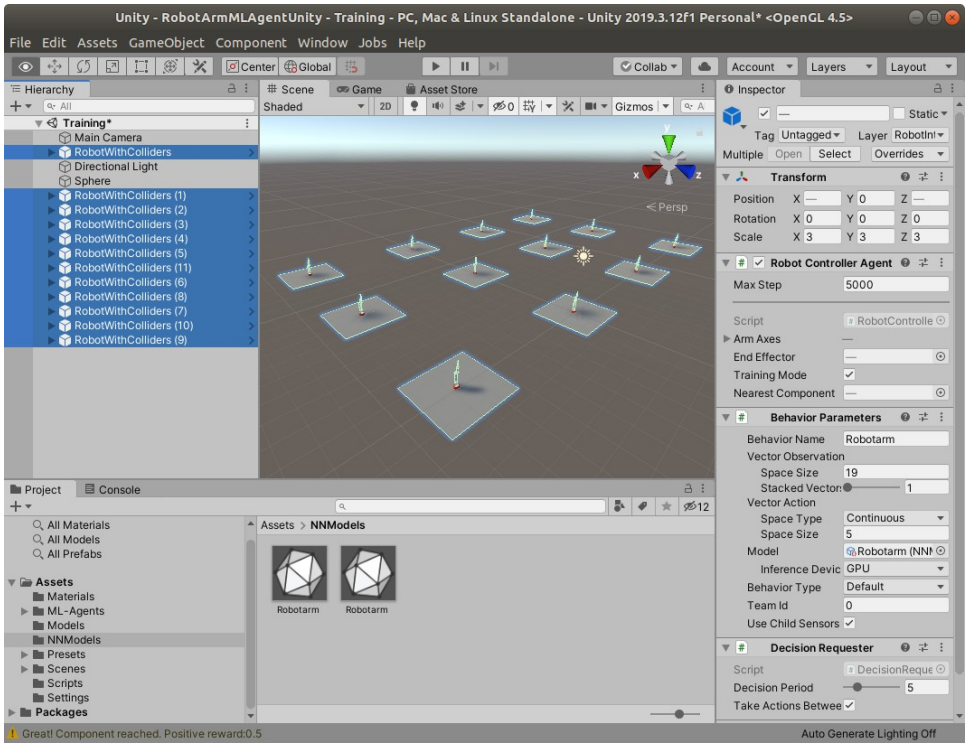
11) Go back to Unity and press the **Play** button to start the training. It will take several minutes.

12) Once it completes, the terminal should have the following:

```
2021-03-15 12:00:40 INFO [stats.py:176] Robotarm. Step: 900000. Time Elapsed: 2114.591 s. Mean Reward: 4.154. Std of Reward: 12.075. Training.
2021-03-15 12:02:29 INFO [stats.py:176] Robotarm. Step: 950000. Time Elapsed: 2223.325 s. Mean Reward: 4.339. Std of Reward: 11.698. Training.
2021-03-15 12:04:15 INFO [stats.py:176] Robotarm. Step: 1000000. Time Elapsed: 2329.232 s. Mean Reward: 4.567. Std of Reward: 12.675. Training.
2021-03-15 12:04:15 INFO [model_serialization.py:130] Converting to results/ra_02/Robotarm/Robotarm-999993.onnx
2021-03-15 12:04:15 INFO [model_serialization.py:142] Exported results/ra_02/Robotarm/Robotarm-999993.onnx
2021-03-15 12:04:15 INFO [model_serialization.py:130] Converting to results/ra_02/Robotarm/Robotarm-1000114.onnx
2021-03-15 12:04:15 INFO [model_serialization.py:142] Exported results/ra_02/Robotarm/Robotarm-1000114.onnx
2021-03-15 12:04:15 INFO [torch_model_saver.py:116] Copied results/ra_02/Robotarm/Robotarm-1000114.onnx to results/ra_02/Robotarm.onnx.
2021-03-15 12:04:15 INFO [trainer_controller.py:81] Saved Model
```

13) Drag and drop the trained model in **RobotArmMLAgentUnity/results/ra_01/Robotarm.onnx** to **NNModels**





14) Select all the **RobotWithColliders**. Drag and drop the new **Robotarm.onnx** into **Model** in **Inspector** >> **Behavior Parameters** >> **Model**

15) Press **Play** button to watch the inferencing.