RUTGERS

Rutgers - Computer Graphics Course - Assignment C0

Unity ML-Agents Toolkit: Navigation

This is an optional group assignment.

Total Assignment Points: **10**

INSTRUCTIONS

Install the Unity ML-Agents Toolkit by following the installation guide. After the completion of the Basic Guide, you will have been able to train a RL model to balance a 3D ball on a plane.

INSTALLATION

Required ML-Agents version: 0.6a

Required Unity version: Unity 4.18f for **Windows**

Main Website: https://github.com/Unity-Technologies/ml-agents

Installation on Windows:

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

Basic Guide:

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

Making a New Learning Environment:

https://github.com/Unity-Technologies/ml-agents/blob/master/docs/Learning-Environment-Create-New.md

Then, after completing the "Making a New Learning Environment" tutorial, please adapt the scene to train a cylindrical agent to navigate from a random starting point to a goal point on a flat, static plane. Please write your code in class `GoToAgent`, which will need to inherit from class **Agent**. You are only permitted to move the agent by applying forces to its rigid body in AgentAction(). Please change AgentReset() and CollectObservations() as needed.

Train your model until you are satisfied with the performance. Also, set `run-id` to **`GoToAgent`** and change the Learning Brain's filename to GoToAgentBrain. Please change your configuration file (`ml-agents-0.6.0a/config/config.yaml`) as needed.

SUBMISSION

Since this is an extra-credit assignment, points will be awarded at the graders' discretion based on the quality of the agent's navigation.

Your final submission (**ONE** per group) will include the following:

- Source code (GoToAgent.cs)
- The saved model which can be found in (`ml-agent-0.6.0a/models/GoToAgent/GoToAgentBrain.bytes`)
- A video demo showcasing the result of the trained model.
- A complete description of your observations, actions, and rewards; what particular aspect of agent locomotion you were aiming to improve; and to what degree you succeeded in doing so.

GUIDELINES

1. CHANGING THE ENVIRONMENT

- If you want to change the environment --e.g., change the size of the floor or add/remove agents/objects before/during the simulation, you can implement the appropriate methods in the Academy.
- When the Agent reaches its target, it marks itself done and its Agent reset function moves the target to a random location. In addition, if the Agent rolls off the platform, the reset function puts it back onto the floor.

2. INITIALIZATION & RESETTING, OBSERVING THE ENVIRONMENT, ACTION, REWARDS

ACTION

- The Index value corresponds to the index of the action array passed to AgentAction() function. Value is assigned to action[Index] when Key is pressed.

REWARDS

- When you mark an Agent as done, it stops its activity until it is reset. You can have the Agent reset immediately, by setting the Agent.ResetOnDone property to true in the inspector or you can wait for the Academy to reset the environment. This RollerBall environment relies on the ResetOnDone mechanism and doesn't set a Max Steps limit for the Academy (so it never resets the environment).

DEBUG

- Note that for more involved debugging, the ML-Agents SDK includes a convenient Monitor class that you can use to easily display Agent status information in the Game window.

TRAINING PARAMETERS

Since this example creates a very simple training environment with only a few inputs and outputs, using small batch and buffer sizes speeds up the training considerably. However, if you add more complexity to the environment or change the reward or observation functions, you might also find that training performs better with different hyperparameter values.

Note: In addition to setting these hyperparameter values, the Agent Decision Frequency parameter has a large effect on training time and success. A larger value reduces the number of decisions the training algorithm has to consider and, in this simple environment, speeds up training.

3. START TRAINING

```
mlagents-learn config/config.yaml --run-id=RollerBall-1
--train
```

4. CONTINUE TRAINING

```
mlagents-learn config/config.yaml --run-id=RollerBall-1
--train --load
```

5. PARAMETER SETUP

https://github.com/Unity-Technologies/ml-agents/blob/master/docs/Training-ML-Agents.md

6. TRAINING WITH PROXIMAL POLICY OPTIMIZATION (PPO)

https://github.com/Unity-Technologies/ml-agents/blob/master/docs/Training-PPO.md