Deep RL Arm Manipulation

Snigdha Dongre

Abstract - The Goal of this project was to use Deep Reinforcement learning technique (Q-Learning) to create a DQN agent and define reward functions to teach a robotic arm to carry out two primary objectives:

1 Have any part of the robot arm touch the object of interest, with at least a 90% accuracy.
2 Have only the gripper base of the robot arm touch the object, with at least an 80% accuracy.

F

1 TASK PERFORMED

Following tasks are performed in ArmPlugin.cpp file:

1. Subscribers are created in ArmPlugin::Load() for camera and collision topics as per the supplied specifications.
2. Created a DQN agent using the Create() function from the dqnAgent Class,in ArmPlugin::createAgent().Variable names defined at the top of the file are passed to this function call.
3. The ArmPlugin::updateAgent() function has 2 different mechanisms to control the arm joint movements namely velocity control and position control. As we have rewards defined in terms of distance to the object in this project we have used only position control to control arm movement.
4. We use Gazebo API for GetBoundingBox() function which is used to check if gripper is hitting the ground. Appropriate rewards are defined for the same.
5. An interim reward is also defined for distance between the arm and the object. A function called BoxDistance() is used to calculate the distance between 2 bounding boxes. This is used to calculate the distance between the arm and the object which is then used to calculate the interim reward. The reward is a smoothed moving average of the delta of the distance to the target. The formula to calculate the same is avgGoalDelta = (avgGoalDelta * alpha) + (distDelta * (1.0f - alpha));
6. In the function onCollisionMsg() a check is performed to see if a collision between the arm(COLISSION_ITEM) and the target object has taken place. Appropriate reward is assigned for the same.
7. Hyperparameters are tuned to get desired accuracy for the 1st objective.
8. The onCollisionMsg() function is modified to see if a collision between the gripper (COLLISION_POINT) and the target object has taken place. Appropriate reward is assigned for the same.
9. Hyperparameters are re-tuned to get desired accuracy for the 2nd objective.

2 REWARD FUNCTIONS FOR 1st OBJECTIVE

Following reward functions are defined to achieve objective 1 i.e robot arm touching the object of interest with at least 90% accuracy:

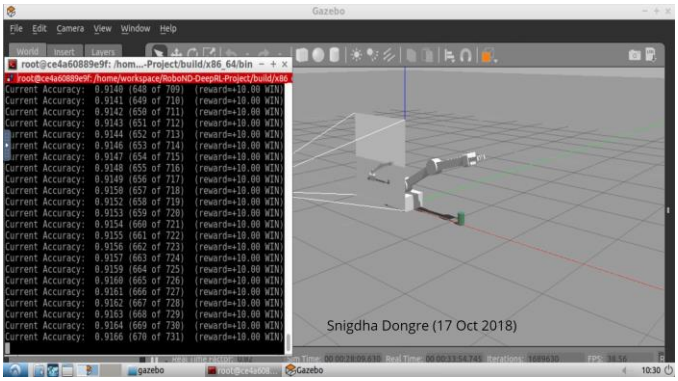
- rewardHistory – This is the value of the previous reward, it can be set to a positive or a negative value.
- REWARD WIN – This is the reward for positive reward. It is set to +10
- REWARD LOSS - This is the reward for negative reward. It is set to -10
- newReward - If a reward has been issued or not. This has been set to false
- endEpisode - If the episode is over or not. This has been set to false.

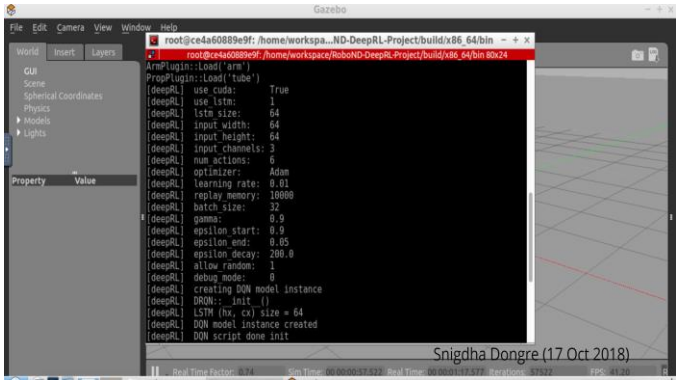
Hyperparameters for 1st Objective:

After trying different set of values for hyperparameters we have finally chosen following values to achieve 1st objective.

Parameters	Value
INPUT WIDTH	64
INPUT HEIGHT	64
OPTIMIZER	Adam(performed better than RMSprop)
LEARNING RATE	0.1
REPLAY MEMORY	10000
BATCH SIZE	32

USE LSTM	True
LSTM SIZE	64





REWARD FUNCTIONS FOR 2nd OBJECTIVE

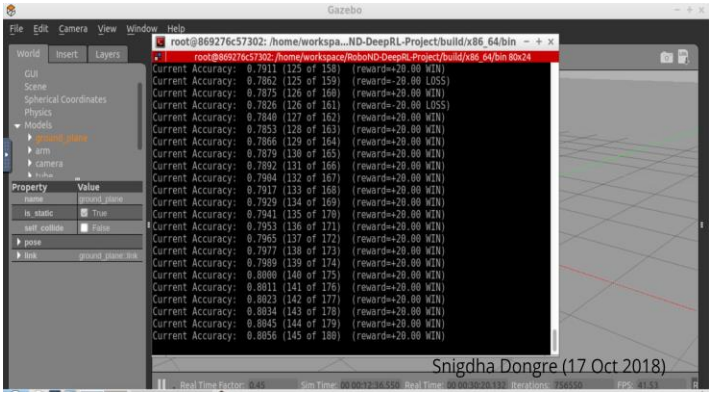
Following reward functions are defined to achieve objective 2 i.e robot gripper touching the object of interest with at least 80% accuracy:

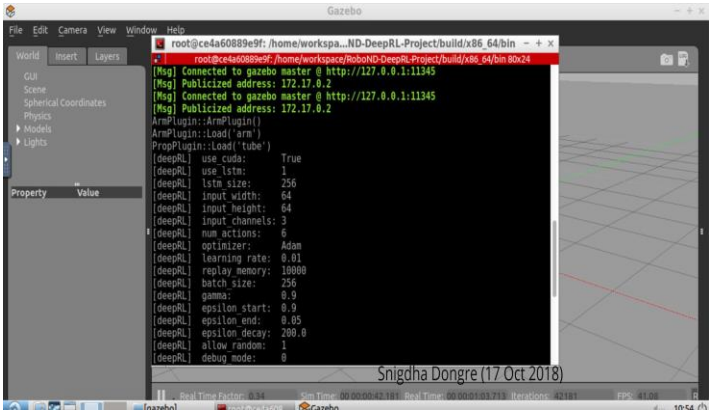
- rewardHistory – This is the value of the previous reward, it
- Can be set to a positive or a negative value.
- REWARD WIN – This is the reward for positive reward. It is set to +20
- REWARD LOSS - This is the reward for negative reward. It is set to -20
- newReward - If a reward has been issued or not. This has been set to false
- endEpisode - If the episode is over or not. This has been set to false.

Hyperparameters for 2nd Objective:

After trying different set of values for hyperparameters we have finally chosen following values to achieve 1st objective.

Parameters	Value
INPUT WIDTH	64
INPUT HEIGHT	64
OPTIMIZER	Adam(performed better than RMSprop)
LEARNING RATE	0.1
REPLAY MEMORY	10000
BATCH SIZE	256
USE LSTM	True
LSTM SIZE	256





3 RESULTS

A DQN agent is created. Rewards functions are defined and hyperparameters are tuned. Several iterations of training is performed. The arm was able to touch the object of interest with an accuracy of 91.66% with 731 iterations of training. Also the gripper was able to touch the object of interest with 80.55% with 180 iterations of training.

4 CONCLUSION / FUTURE WORK

The DQN agent was successful to learn with experience and through rewards to perform required tasks with adequate accuracy. However, with more training the arm can certainly perform better. Also, we used only position control to control the arm movement. If we create a custom complex movement mechanism then the arm can learn better and faster. We can also experiment with different moving averages for defining the interim reward. Also we can modify the project to increase arm length and to make it work for objects placed at different locations.

REFERENCES

[1] Udacity, Robotics software engineering nanodegree, Deep RL Arm Manipulation Project