Deep RL Arm Manipulation project

1 Project overview

The goal of this project is 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 for a minimum of 100 runs.

- Have only the gripper base of the robot arm touch the object, with at least a 80% accuracy for a minimum of 100 runs.

The upcoming sections describe how we set out to accomplish these two objectives in detail, in particular in explaining the reward functions, hyperparameters. Ater that, a dedicated section is presented to show our final results, and then reflects on how we might further improve them later on.

2 Reward functions

2.1 Explaining joint control

The robotic arm in this project consists of three revolute joints, meaning it has three degrees of freedom. We chose increasing or decreasing the joints position to manipulate the robot arm. For a given joint, an even value of action number indicates increasing joint position by `0.15` and an odd value increase the position by `-0.15` within a range of `[-0.75, 2.0]`. In realization, combining these two actions with the three joints, we adopted six actions, in numerical form `0, 1, 2, 3, 4, 5` to control the robot arm. For *Joint0*, action `0` increase its joint position and `1` decrease its joint position. Actions `2` and `3` manipulate *Joint1*, `4` and `5` manipulate *Joint2* in the same manner. However, because *Joint0* is fixed rigidly to the world, only actions for *Joint1* and *Joint2* are meaningful in this project.

2.2 Explaining Reward functions

We define *REWARD_WIN* as `1.0` and *REWARD_LOSS* as `-1.0` respectively and set either of them to the DeepRL agent accordingly at the end of every episode. While in the training episode, we implemented the reward functions as follows:

1. Whenever the gripper link hits ground, issue *REWARD_LOSS* and end current episode.

- If it reaches 100 update steps without accomplishing objective successfully, end the episode and issue *REWARD_LOSS*.
- For objective 1, whenever a collision is detected by the target tube (line 273 of ArmPlugin.cpp), issue *REWWARD_WIN* to the agent and end the episode.
- For objective 2, if the collision on target tube is caused by gripper link (line 265-266 of ArmPlugin.cpp), end episode and issue *REWARD_WIN* to the agent.
- Within every update step, we calculate the distance from gripper link to the target tube, namely `distGoal`. Then compare it with previous distance `lastGoalDistance` and get the difference `distDelta`. In theory, we'd like to use this `distDelta` as the interim reward for the RLAgent, but it's too coarse. So an averaging method is applied to smooth it out. The pseudo code of the formula is,

```
avgGoalDelta = (avgGoalDelta * 0.3) + (distDelta * 0.7)
```

3 hyperparameters

In this project, we set out to experiment with various optimizers and hyperparameters based on the performance in the first twenty episodes. To begin with objective 1, we chose `128 x 128` for the input image shape of a `64` batch size, `Adam` optimizer with default learning rate `0.001`, a `20000` replay memory, and enable `LSTM` with a size of `256`. After trial and error, we ended up with following hyperparameters settings for objective 1.

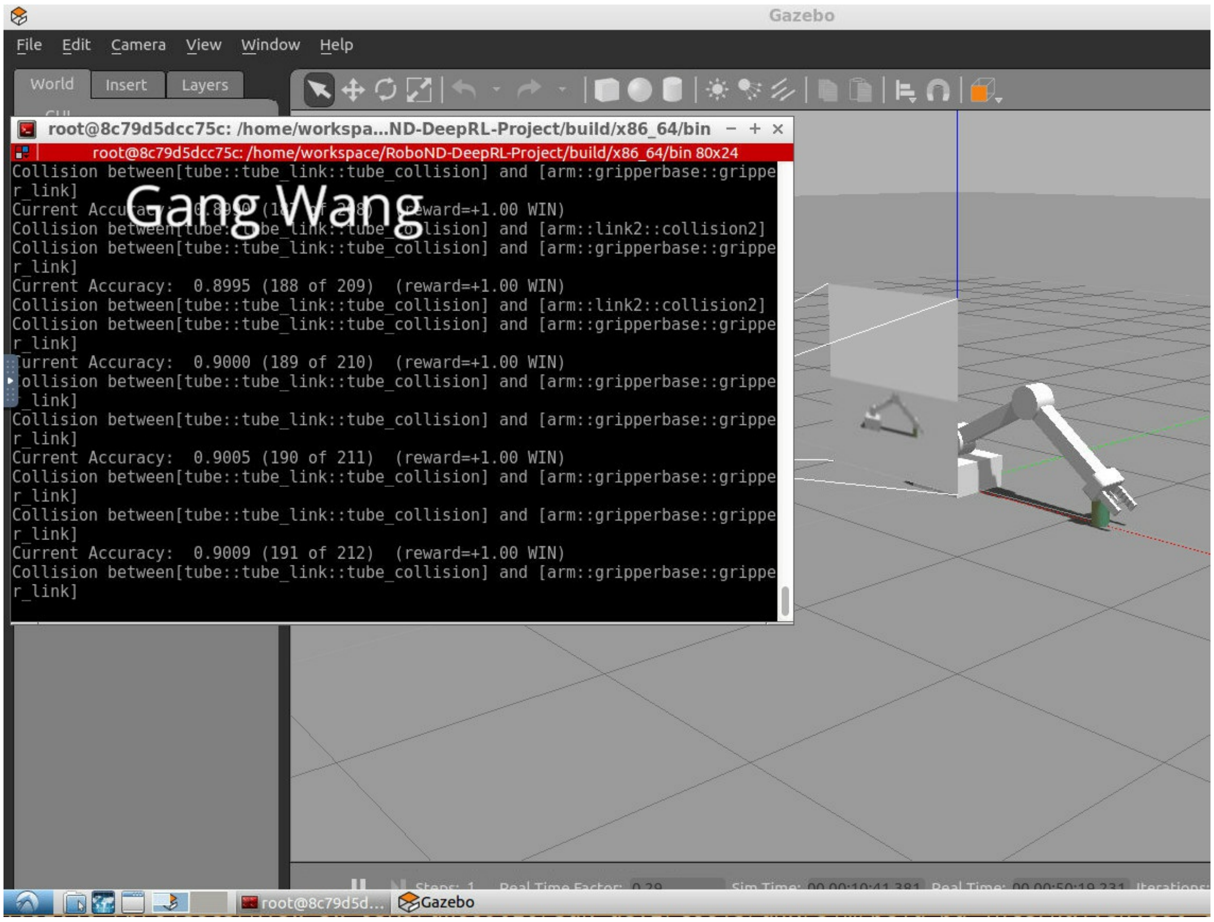
Name	Value
INPUT_WIDTH	128
INPUT_HEIGHT	128
OPTIMIZER	Adam
LEARNING_RATE	0.01
REPLAY_MEMORY	20000
BATCH_SIZE	512
USE_LSTM	true
LSTM_SIZE	512

Surprisingly, from the collision debug messages we noticed the RLAgent has a high probability in colliding gripper link with target tube for carrying out objective 1. So we just modified collision detection in pursuing objective 2 without changing any above hyperparameters.

4 Results

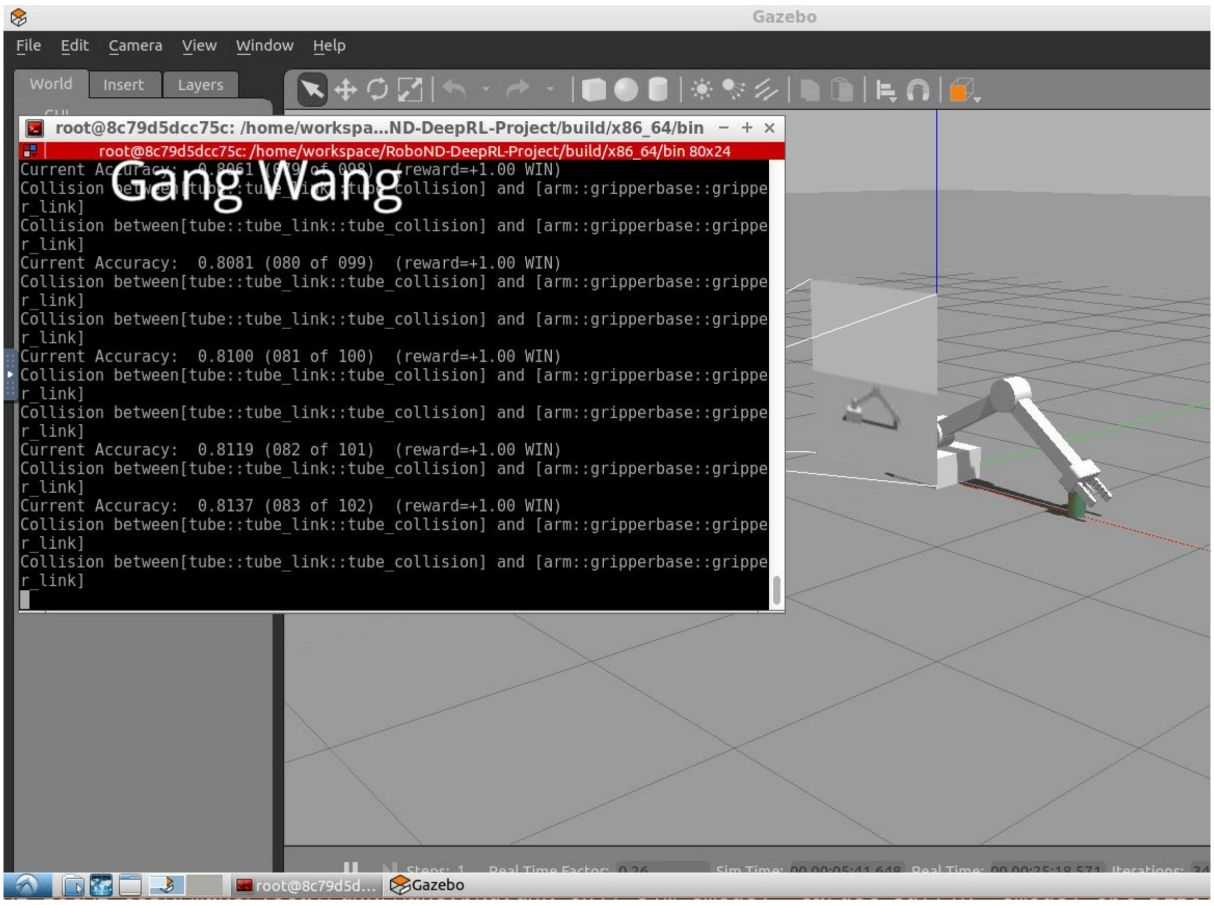
4.1 Snapshot of results

1. Any part of robot arm touches target tube.


Gazebo
File Edit Camera View Window Help
World Insert Layers
root@8c79d5dcc75c: /home/workspa...ND-DeepRL-Project/build/x86_64/bin
root@8c79d5dcc75c: /home/workspace/RoboND-DeepRL-Project/build/x86_64/bin 80x24
Gang Wang
root@8c79d5d...
Gazebo

```
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
Current Accuracy:  0.8990 (187 of 208)  (reward=+1.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::link2::collision2]
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
Current Accuracy:  0.8995 (188 of 209)  (reward=+1.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::link2::collision2]
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
Current Accuracy:  0.9000 (189 of 210)  (reward=+1.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
Current Accuracy:  0.9005 (190 of 211)  (reward=+1.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
Current Accuracy:  0.9009 (191 of 212)  (reward=+1.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
```

- Gripper link touches target tube.


Gazebo
File Edit Camera View Window Help
World Insert Layers
root@8c79d5dcc75c: /home/workspa...ND-DeepRL-Project/build/x86_64/bin
root@8c79d5dcc75c: /home/workspace/RoboND-DeepRL-Project/build/x86_64/bin 80x24
Gang Wang
root@8c79d5d...
Gazebo

```
Current Accuracy:  0.8061 (079 of 098)  (reward=+1.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
Current Accuracy:  0.8081 (080 of 099)  (reward=+1.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
Current Accuracy:  0.8100 (081 of 100)  (reward=+1.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
Current Accuracy:  0.8119 (082 of 101)  (reward=+1.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
Current Accuracy:  0.8137 (083 of 102)  (reward=+1.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::gripper_link]
```

4.2 Discussions

Because in most *REWARD_WIN* scenarios, it is actually gripper link collides w ith target tube. This implies to reach a relatively low er accuracy task as objective 2, it takes few er episodes in training the RLAgent.

Our RLAgent achieved the requirements decently, how ever, w e noticed even if it reached an accuracy not far from the target, say about 89% for objective 1 and 79% for objective 2, there is still some chance it failed to get a *REWARD_WIN* due to odd manipulations, w hich looked like the agent hadn't learned anything yet.

Another concern goes to the fine tuning hyperparameters. We noticed there's a discernable speed up if w e reduce the input image size to be `64x64` . It's tempting to think there might exist a more optimized hyperparameters combination yet to be explored. We w ould like leave this as a future w ork later on.

5 Future Work

An obvious future w ork w ould be to try to experiment more hyperparameters in order to reduce training episodes, as w ell as speeding up each episode as mentioned in previous section.

It is also tempting to think how to apply a DeepRL Arm Manipulator to a more complex robotic application such as the **Pick and Place Project** in Term1. Given the chance, w e'd like to make a DeepRL agent for a 6-DOF robotic arm to accomplish some pick & place tasks.