Deep RL Arm Manipulation

Max Podkolzin

1. Abstract

This report describes building a DQN agent and define reward functions to teach a robotic arm to the robot arm touch the object of interest

The task of this project was controlling robotic arm with three joints in the Gazebo simulator with help of the Deep Q Network. That has been implemented in C++ programming language by accessing DQN API with the help of dqnAgent class.

2. Reward functions

1. Reward function defined as negative reward that happens when the gripper hits the ground. There is negative reward of 100 when the gripper's z coordinate reaches bellow or equal to the height of 0.05. This action cancels the whole episode, as it sets the parameter "endEpisode" to true.
2. Interim reward based on the gripper's distance to the object. This reward function measures current and previous distances to the object and gives the positive reward if the distance is getting smaller and the negative reward if the distance is getting larger. As proposed in task 5 for the project and smoothed average of the mean of the distance of the goal is used to reward the arm when getting close to the object.
3. Collision reward function. Two positive rewards are the object's collision with the robot's gripper base reward (task #2) and the object's collision with the robot's tube(arm) reward (task #1).

3. Hyperparameters

1. The first two hyperparameters defined where the input width and the input height. Both values were reduced to 64 from original 512 as this is the image format sufficient to perform the training that doesn't require too much computational resources.
2. RMSprop has been set as optimizer as it is one of most common optimizers and adaptively adjust its learning rate using gradients and outperformed the Adam optimizer based on experimentation.
3. Learning rate of 0.15 with the batch size of 64 proved to perform best.
4. Replay memory has been set to 10000, so that there is enough information for the model to train on.
5. LSTM network has been used as well, with the LSTM size set to 128.

4. Results

There are two tasks in this project:

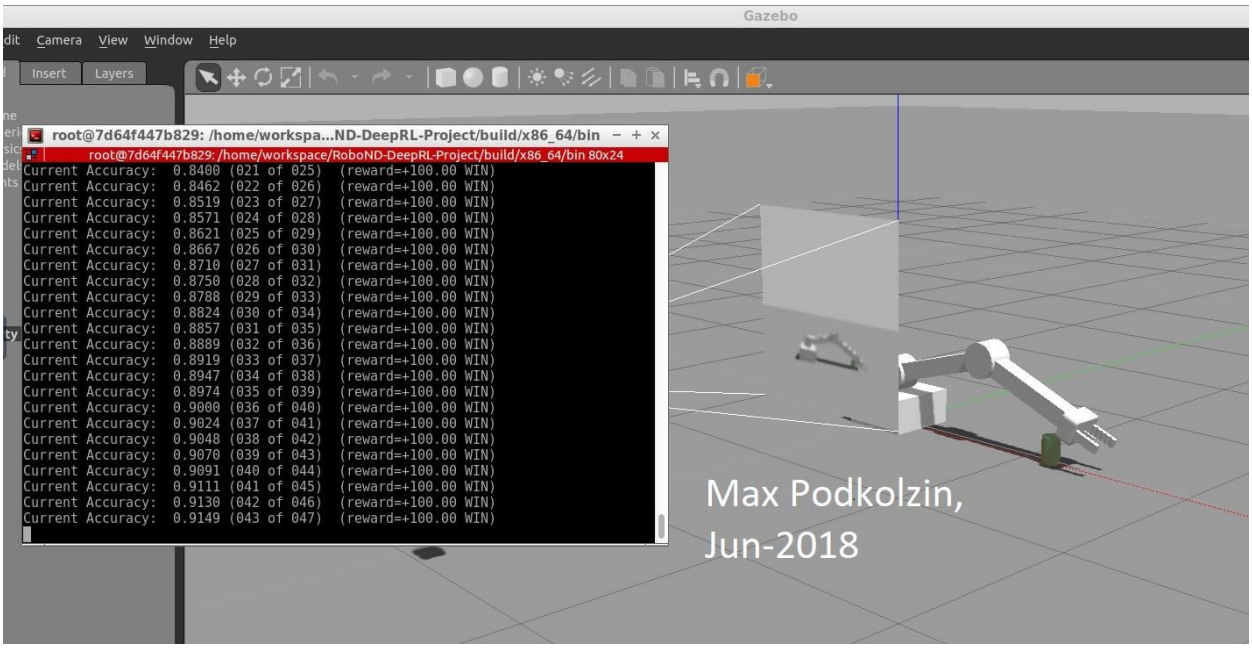
1) Any part of the robot arm should touch the object with at least an accuracy of 90%.

2) Only the gripper base of the robot arm should touch the object with at least an accuracy of 80%.

Both tasks are completed with settings described in previous sections. For both objectives specified accuracy has been achieved in the provided environment. Based on the multiple experiments it was established that sooner the robot reaches the goal for the first time, the faster the training goes

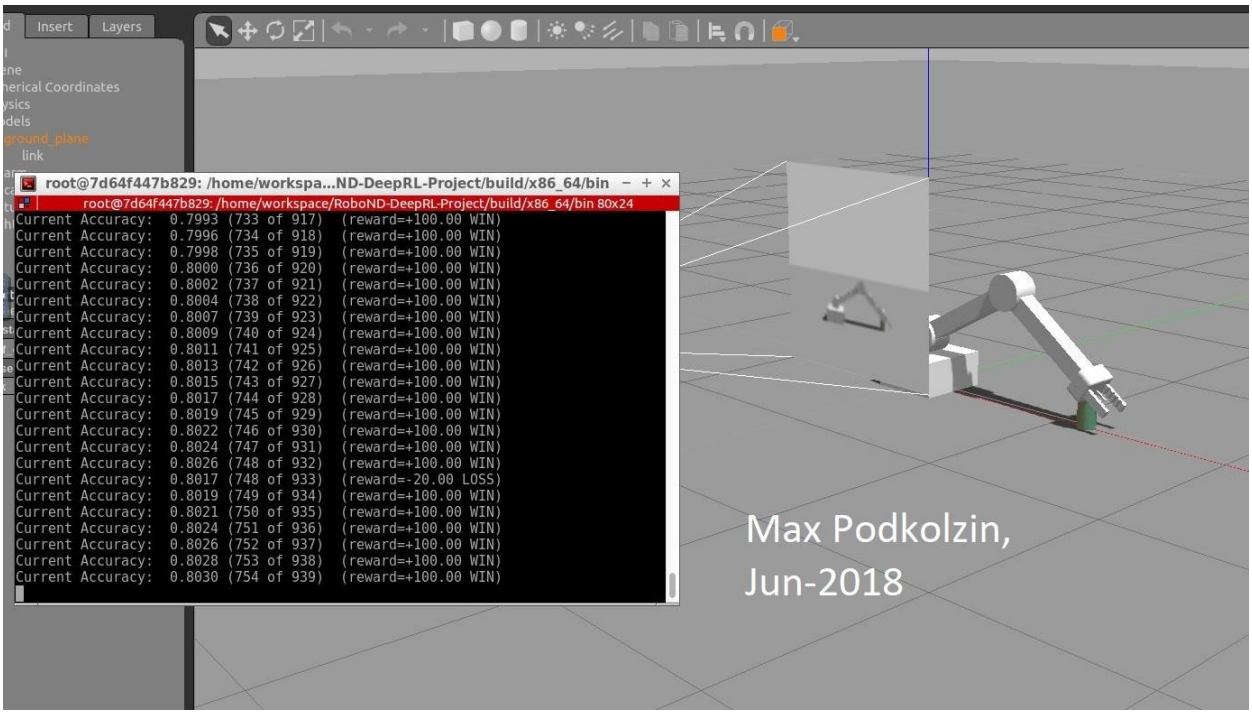
4.1 Task #1

The goal for Task #1 was achieved within 50 runs, however



4.2 Task #2

The goal for Task #2 was achieved in 700 runs. The number of runs required to achieve desired accuracy is far from ideal, this could be explained by the fact that the reward function used was quite simple and does not take into account variables such as time or joint angle values.



1. Future work

As a result of the experiments, it became obvious that training begins to go well only after the robot first touches the target. Therefore, further improvements should be concentrated precisely on this task. Most likely this can be achieved by optimizing the reward functions.

Adding RGBD camera to the gripper would give more input data that could be used to train dqnAgent.

Adding gripping functionality would be the next logical step as well.