

# Deep RL Arm Manipulation

Saminda Abeyruwan

**Abstract**—Deep reinforcement learning has enabled robotic systems to learn control or predictive policies directly from camera inputs and raw pixels. In this project, a simulated 3 degree-of-freedom robotic arm is trained to reach and touch an object of interest using a deep  $Q$ -network. We have simulated two tasks, and show that the robot is successfully learned the abilities to complete the scenarios.

**Index Terms**—Robot, IEEEtran, Udcity, L<sup>A</sup>T<sub>E</sub>X, Deep RL, LSTM.

## 1 INTRODUCTION

DEEP reinforcement learning algorithms have been successfully applied to wide range of robotic control tasks, e.g., locomotion, robotic arm manipulation, and autonomous vehicle control [1]. With the development of the deep  $Q$ -networks and its variants [2] [3] [4], the reinforcement learning paradigm has been scale to problems that were previously deemed intractable, e.g., learning to play video games from direct raw pixels.

In this project, we have used an LSTM based deep  $Q$ -network, similar to presented in [5], on 3 degree-of-freedom (DoF) robotic manipulator arm to reach and touch an object of interest in two scenarios:

- 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, and
- 2) 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 robot and the two tasks have been simulated in Gazebo [6], and we show that the robot is successfully learned the abilities to complete the tasks with the given project requirements.

## 2 REWARD FUNCTIONS

The first task requests to have any part of the robot arm touch the object of interest. The following reward structure has been used:

- 1) any part of the robot touches the object with reward +500.
- 2) any part of the robot, including gripper, collides with ground or itself, the reward has been set to -500.
- 3) an interim reward based on smoothed moving average. This function is composed of two parts:
  - a) For the time step  $t$  and  $t + 1$ , lets assume the robot gripper has incrementally moved  $\delta_t = d_t - d_{t+1}$  distance, where,  $d_t$  is the distance to the object from gripper. If  $\delta_t > 0$ , it is an indication of the gripper moved closer to the object. Lets assume  $\bar{\delta}_t$  is the moving average of  $\delta_t$  distances. Then, we use the smoothing

formula,  $\bar{\delta}_{t+1} = \bar{\delta}_t \times \alpha + \delta_t \times (1 - \alpha)$ , where  $\alpha \in [0, 1]$ .

- b) In order to encode the objective of completing the task as soon as possible, we have introduced a time step penalty, *time\_penalty*. This penalty has eliminate the gripper oscillating near the object of interest.

Hence, using the prior decomposition, each step we have provided the reward  $\bar{\delta}_t - \text{time\_penalty}$ .

For the second task, we have used the following reward structure:

- 1) Only gripper touches the object with +500.
- 2) Any other collision resulted in a reward of -50.
- 3) Similar to the first task, an interim reward based on smoothed moving average use been used.

## 3 HYPERPARAMETERS

We have used images of size  $64 \times 64 \times 3$ . The *RMSprop* optimizer has been used in both tasks, while setting the learning rate to 0.1. Neither tasks required changing the learning rate. We set the replay memory 10000, and a batch size of 32 has been used. We have enabled the LSTM mode, and the LSTM size has been set to 512. The exploration rate,  $\gamma$ , has been set to 0.9 and annihilated to 0.05 over 200 steps. We have used position control method to control the arm. Please refer to the definitions, INPUT\_WIDTH, INPUT\_HEIGHT, OPTIMIZER, LEARNING\_RATE, REPLAY\_MEMORY, BATCH\_SIZE, USE\_LSTM, LSTM\_SIZE, GAMMA, EPS\_START, EPS\_END, EPS\_DECAY, and VELOCITY\_CONTROL, in the associated code for the value binding.

## 4 RESULTS

The robot has learned successful control policies for both tasks within 200 episodes. We have let the first task, Fig. 1, to train over 600 episodes to reach 99% of local accuracy. An instance of the completed second tasks is shown in Fig. 2. Both tasks were simulated using the same set of parameters. Both tasks were completed using position based control.

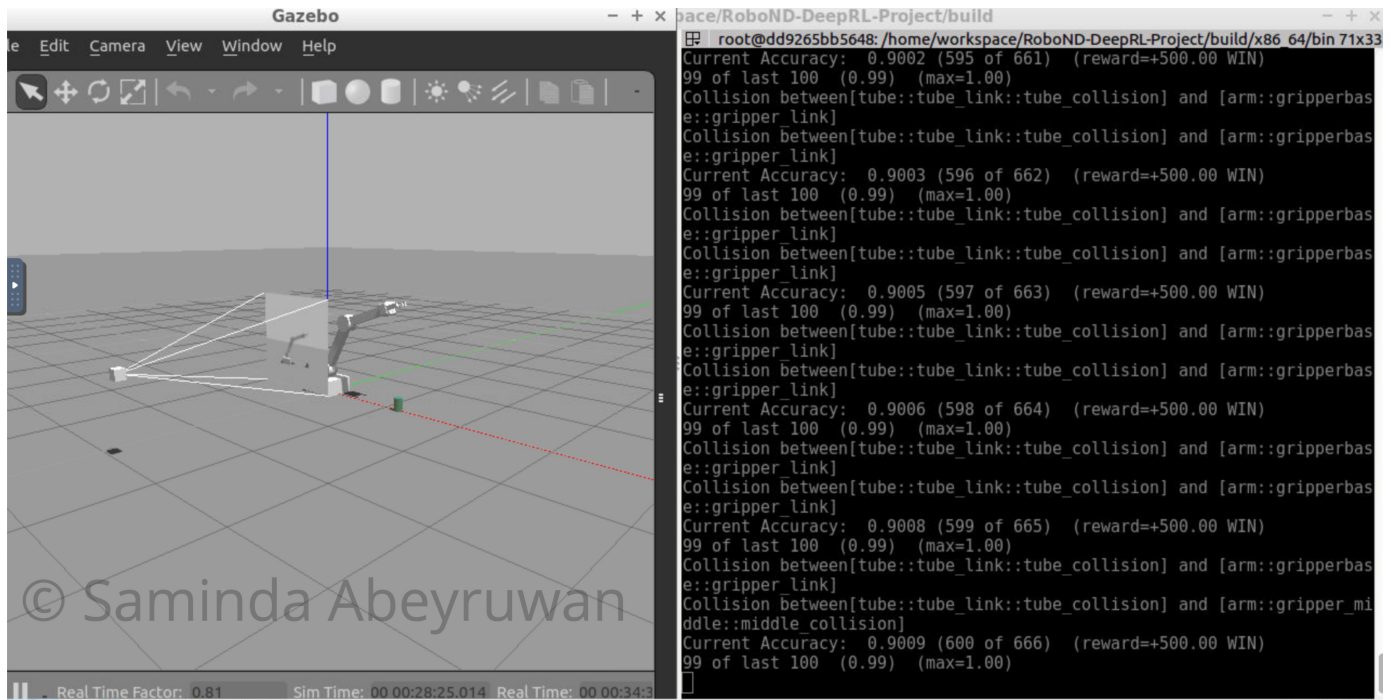


Fig. 1. Task 1: have any part of the robot arm touch the object of interest.

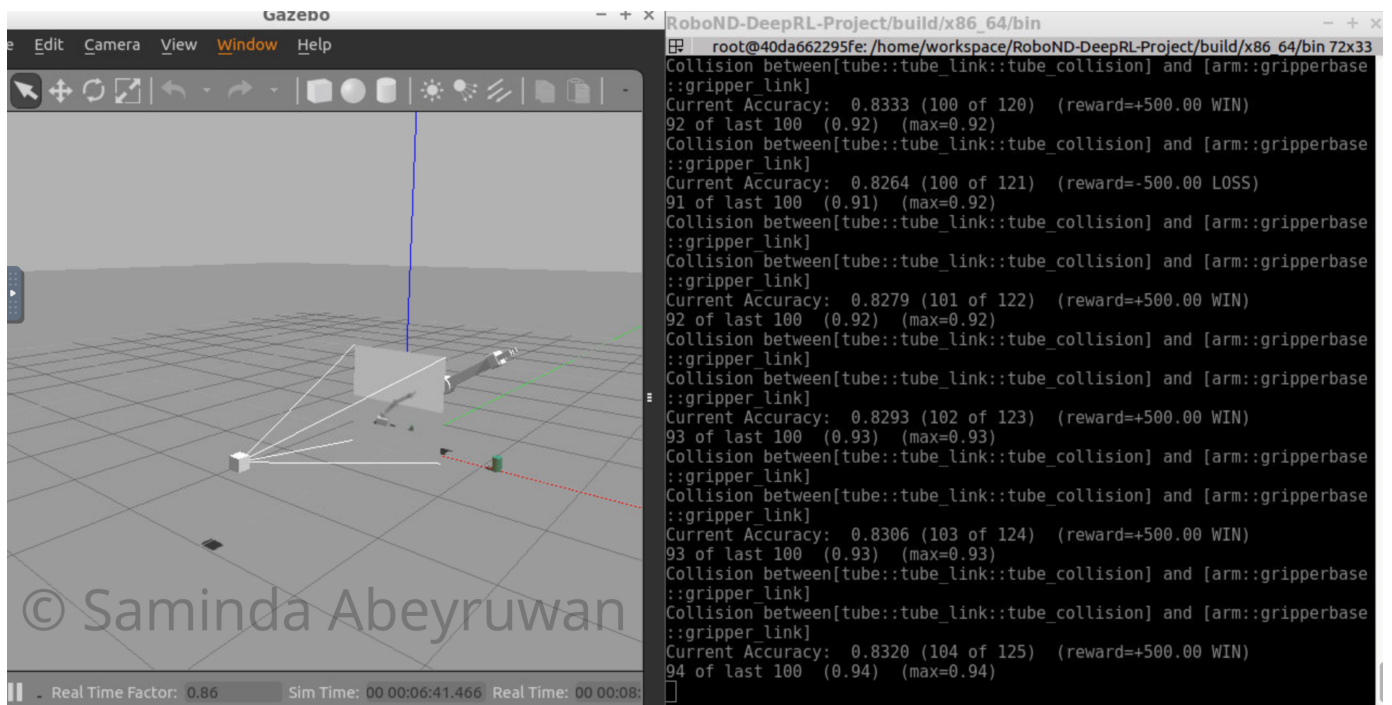


Fig. 2. Task 2: have only the gripper base of the robot arm touch the object.

## 5 CONCLUSION / FUTURE WORK

In this project, we have trained 3 DoF robotic arm to successfully complete two tasks in simulation using LSTM based deep-Q-networks. A generalized set up based on the current project would help to train other robotic problems. Even though we have used position control, a more realistic robotic control is based on velocity control, which requires continuous control for best performance. The algorithms such as Deep Deterministic Policy Gradient (DDPG), and Normalized Advantage Function algorithm (NAF) [1] have shown state-of-the-art results, and will be benefited to the tasks presented in this project to obtain more realistic results.

## REFERENCES

- [1] S. Gu\*, E. Holly\*, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *Proceedings 2017 IEEE International Conference on Robotics and Automation (ICRA)*, (Piscataway, NJ, USA), IEEE, May 2017. \*equal contribution.
- [2] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.
- [3] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Harley, T. P. Lillicrap, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pp. 1928–1937, JMLR.org, 2016.
- [4] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "A brief survey of deep reinforcement learning," *CoRR*, vol. abs/1708.05866, 2017.
- [5] M. J. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," in *AAAI Fall Symposia*, pp. 29–37, AAAI Press, 2015.
- [6] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.