Deep RL Arm Manipulation

Michael Hetherington

Abstract—This project is the fourth in the second semester of the Udacity Robotics Software Engineer Nanodegree. The purpose of this project is to provide the student with hands-on experience of training a Deep Reinforcement Learning Network to learn how to use a robotic arm to make contact with a small static object. The network was trained using a camera image of the environment. The trained network achieved greater than 90% accuracy in making the robotic arm contact the object (Objective 1) and greater than 80% accuracy in making the gripper base contact the object (Objective 2).

Index Terms—Robot, IEEEtran, Udacity, LaTeX, Reinforcement Learning, q-learning, Deep RL, Gazebo.

1 Introduction

This project is the fourth in the second semester of the Udacity Robotics Software Engineer Nanodegree. The purpose of this project is to provide the student with hands-on experience of training a Deep Reinforcement Learning Network to learn how to use a robotic arm to make contact with a small static object. The type of reinforcement network adopted was a Deep Q-Learning Network (DQN). The student was provided with sample code to spawn the robotic arm within a Gazebo environment but was required to complete a number of tasks to initiate and then tune the DQN to achieve the following objectives:

1) 90% accuracy of making contact with the object with any part of the arm; and
2) 80% accuracy of making contact with the object with the gripper base.

2 Background

A DeepRL learning algorithm enables a robot to learn a new task. In this project a DQN algorithm has been selected to provided the learning functionality. The DQN will enable the robot to execute actions that are available to it (rotating joints) to achieve a goal. As part of the DQN establishment the student provides a reward function which rewards or punishes the robotic arm when it either achieves a goal or part goal (i.e. makes contact with the object, or progressively moves closer to the object) or fails to achieve a goal (e.g. makes contact with the ground).

For this project, the input to the DQN will be output from a camera positioned orthogonally to the robotic arm and object. The camera will relay to the DQN the position of the arm and object and enable it to improve its accuracy in achieving the objectives.

All training and simulation was undertaken on an NVIDIA Jetson TX2 operated in a headless configuration (e.g. remotely accessed via another computer).

The Gazebo environment that was created for the project is shown in Figure 1. This image is taken from the Udacity course notes and shows:

- The robotic arm with gripper attached to it;
- A camera sensor, to capture images to feed in to the DQN; and
- A cylindrical object.

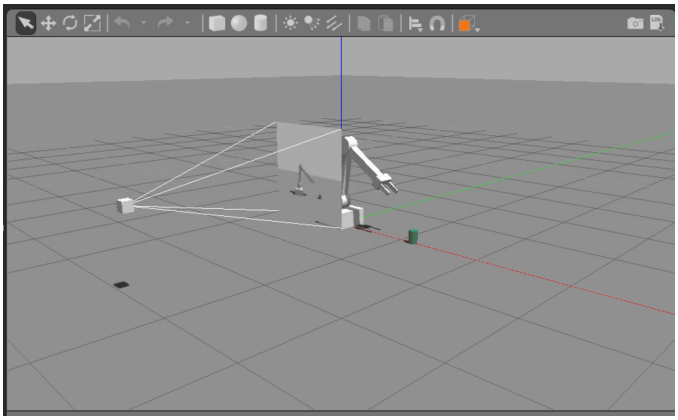

Fig. 1. Demonstration of the Gazebo world with robotic arm, camera for DQN training and cylindrical target object.

3 Reward Functions

Reward functions were built in to the DQN that rewarded or punished the DQN for certain behaviours.

At the end of each simulation episode the DQN was either rewarded 500s point for a successful episode or was punished -500 points for a failed episode. A successful episode was defined as the robotic arm (objective 1) or gripper (objective 2) making contact with the object. An unsuccessful episode was defined as any part of the robotic arm making contact with the ground or more than 100 simulation frames occurring without any other end of episode event occurring.

The DQN was also programmed with an 'interim reward' which would be provided at the end of each frame throughout a training episode. The interim reward was based on the smoothed moving average of the distance between the robotic arm/gripper and the target object calculated frame-to-frame. This interim reward didn't impact

the end of episode rewards (+500/-500) but would incrementally guide the robotic arm towards its goal within a training episode.

The smoothed moving average formula used as input the distance from the arm/gripper to the object in the previous frame and the distance difference in the current frame. An alpha of 0.4 was adopted such that the moving average update retained 40% of the previous frame's value and took on 60% of the new frame's value. The formula for the smoothed moving average used to issue interim rewards has the general form:

newAvg = (prevAvg * alpha) + (prevDistToGoal - newDistToGoal)*(1.0 - alpha)

This interim reward would reward the robotic arm when it moved closer to the object during a training episode.

3.1 Hyperparameters

A number of hyperparameters were available for tuning that would effect how the DQN learned to activate the robotic arm. These hyperparameters and their chosen values are detailed in Table 1.

In additional to the hyperparameters it was essential to select whether the robotic arm would be used with position control or velocity control. Through trial and error, and learning from the experiences of other students on the Udacity Slack channel, velocity control was adopted for objective 1 while position control was found to be more effective for objective 2.

TABLE 1
DQN hyperparameters

Hyperparameter	Value
Input_width	64
Input_height	64
Optimizer	RMSprop
Learning_rate	0.1
Replay_memory	10000
Batch_size	32
Use_LSTM	true
LSTM_size	128

The reasons for selecting these hyperparameters is explained as follows:

- The input width and height were reduced to 64x64 as this was the size of the image that the camera captured and provided to the DQN;
- The learning rate was set at 0.1 as this is a reasonable value to commence from as learned in the previous neural network lessons. Furthermore, it was seen that the robotic arm was able to achieve its goal with this initial learning rate. Varying the learning rate may allow for faster training/convergence to the objectives;
- Batch size was increased to 64 for objective 1 and then reduced to 32 for objective 2 - no noticeable difference in Gazebo FPS which indicates that higher batch sizes could be used for even faster training;
- Long-Short-Term Memory was turned on by setting the parameter to TRUE. This enabled the DQN to retain a memory of its recent and significant previous training episodes so that it need not relearn significant behaviour from scratch for each episode; and
- The LSTM size was increased from default of 32 to 256 for objective 1 and then reduced to 128 for objective 2 - no noticeable difference was observed between these two variable choices. Presumably, increasing the LSTM size will improve training times provided adequate computing power and memory storage are available.

3.2 Results

3.2.1 *Objective 1*

The DQN achieved a 90% accuracy for making contact between the robotic arm and the target object (Objective 1) as shown by the terminal output in Figure 2.

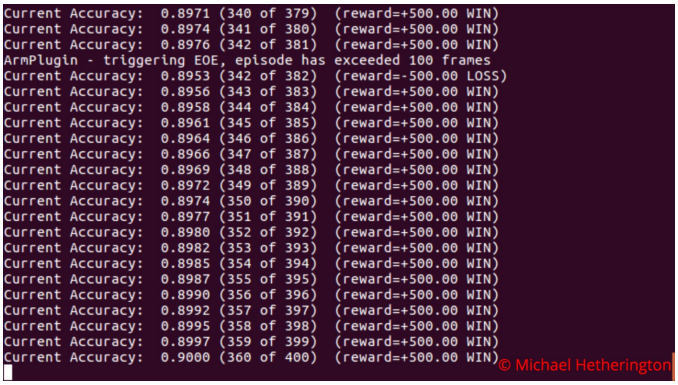

Fig. 2. Terminal output for Objective 1.

The terminal output shows that the objective was achieved after 400 episodes however it was achieved prior to this. The robot still performed random actions from time to time as part of the learning process which would appear to have knocked it off its stable 90% accuracy until it regained this level of accuracy again at 400 episodes.

3.2.2 *Objective 2*

The DQN achieved an 80% accuracy for making contact between the gripper and the target object (Objective 2) as shown by the terminal output in Figure 3.

Fig. 3. Terminal output for Objective 2.

Training time to achieve Objective 2 was significantly longer than for Objective 1. The Jetson TX2 that was used

for training was left to run overnight by the student. By morning the DQN had completed 10,750 training episodes and was achieving a stable accuracy of nearly 83%. On review, the student found that the DQN had been achieving this accuracy since around the 2,500th training episode. Terminal output prior to this was not available and it's possible that an accuracy of 80% was achieved well before 2,500 episodes.

4 Discussion and Future Work

In this project it was shown how a DQN could be used to train a robotic arm to make contact with a target object using only the input of a visual camera.

The DQN achieved both its objectives of making contact with the object - first with any part of the arm and then secondly with the gripper base. Future work could entail:

- Implementing the additional project challenges including: object randomisation; increasing the arm's reach; and combining the randomisation with the increase in reach. (the student plans to undertake these tasks while studying for the final project);
- Fine tuning the training parameters to achieve even higher accuracies. Particularly with a focus on how the LSTM is used to emphasise the significant interim results further;
- Trial different interim reward functions including exponential functions that the student observed have been adopted by others;
- While an ADAM optimiser was used by the student at one point briefly its effectiveness with different hyperparameters could be tested; or
- Tune the robotic arm to make contact with the object located between the two gripper fingers - furthermore, activate these gripper fingers to grab and then lift the object.