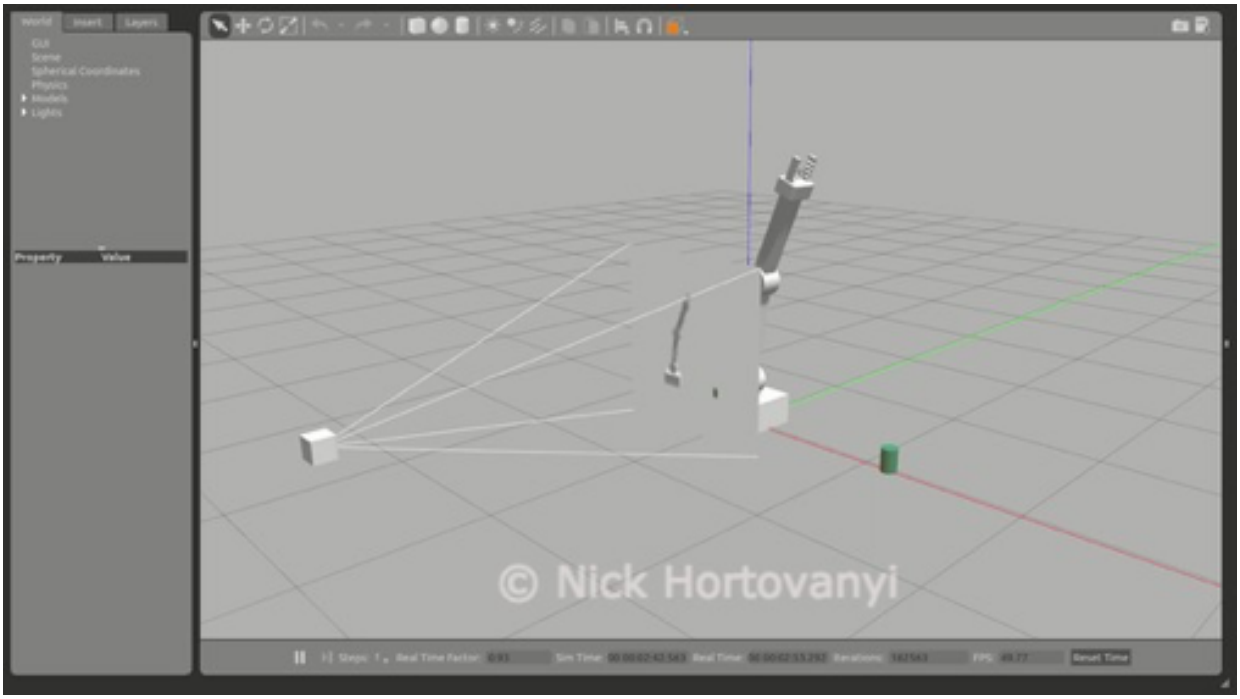
Deep RL Arm Manipulation

This Deep Reinforcement Learning Arm Manipulation project has two objectives to achieve using a template project. Whereby to achieve each objective we create a DQN agent and define reward functions to teach a robotic arm.

The template project is based on the Nvidia open source project "jetson-reinforcement" developed by Dustin Franklin.



Robot Arm in Gazebo simulator

The two primary project objectives are:

- 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.

Reward functions

The reward functions are defined in ArmPlugin.cpp. The arm joints were updated using position control (as that was the programs default setting). For each joint there are two actions (either increase or decrease joint position).

`REWARD_WIN` was set to `0.125` (`0.1` 2nd objective) with `REWARD_LOSS` to `-0.125` (`-0.1` 2nd objective).

If the robot gripper hit the ground a `REWARD_LOSS * 10` was given and the episode ended.

Interim rewards, within the episode, were issued if there was no ground contact or 100 frames had not beed exceeded nor the

The main interim reward was based on the distance goal delta between the gripper and the cycling prop. If a positive weighted average was derived then a `REWARD_WIN` was recorded otherwise `REWARD_LOSS * distance to goal` was issued. Thus the `REWARD_LOSS` was higher then further away from the goal the arm was.

For the gripper base (2nd) objective an additional `REWARD_LOSS` was added if the absolute average goal delta was < `0.001` to penalise no movement.

If the robot arm hit the prop, a `REWARD_WIN * 10` was used for the first objective otherwise a `REWARD_LOSS * 5` for the second objective if the collision was not with the `gripper_middle` .

However for the second objective a `REWARD_WIN * 20` was issued if the collision point was `gripper_middle` .

Any collision ends the episode.

Hyper Parameters

Image dimensions were set to the same size as the input. Training was performed on a GTX1070 and there was no need to restrict memory usage. `INPUT_WIDTH 64` `INPUT_HEIGHT 64`

`OPTIMIZER "Adam"` was chosen as it in general performs better then `RMSProp` whilst maintaining its advantages.

For objective 1 the `LEARNING_RATE` was `0.1` with `REPLAY_MEMORY` at `1000`. The value was chosen via trail and error.

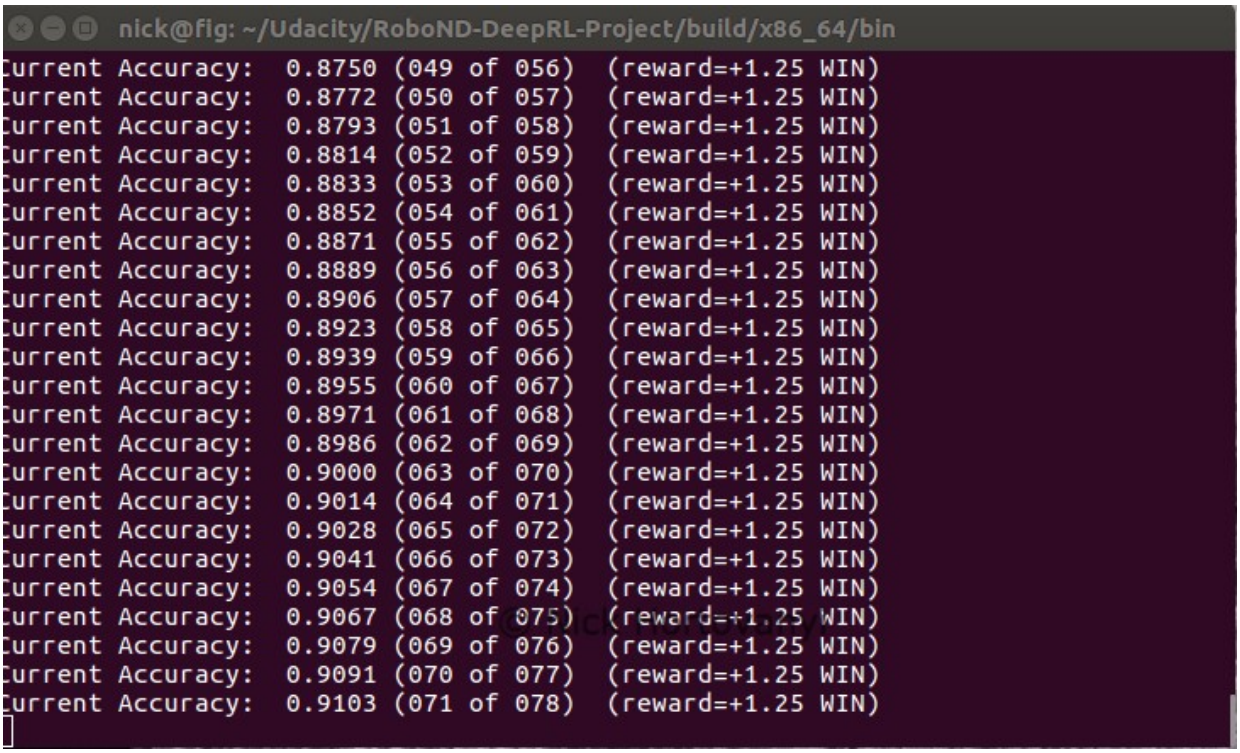
For objective 2 the `LEARNING_RATE` was decreased to `0.01` due to the higher `REPLAY_MEMORY` set at `20000`. The higher `REPLAY_MEMORY` was used so as to allow for more discrete learning, due to the smaller surface area required to achieve a collision to meet objectives.

For both `BATCH_SIZE` was set to 512 (again sufficient memory on the GTX 1070).

LSTM was used `USE_LSTM` `true` with `LSTM_SIZE` `256` which was set via trial and error.

Results

Objective 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.



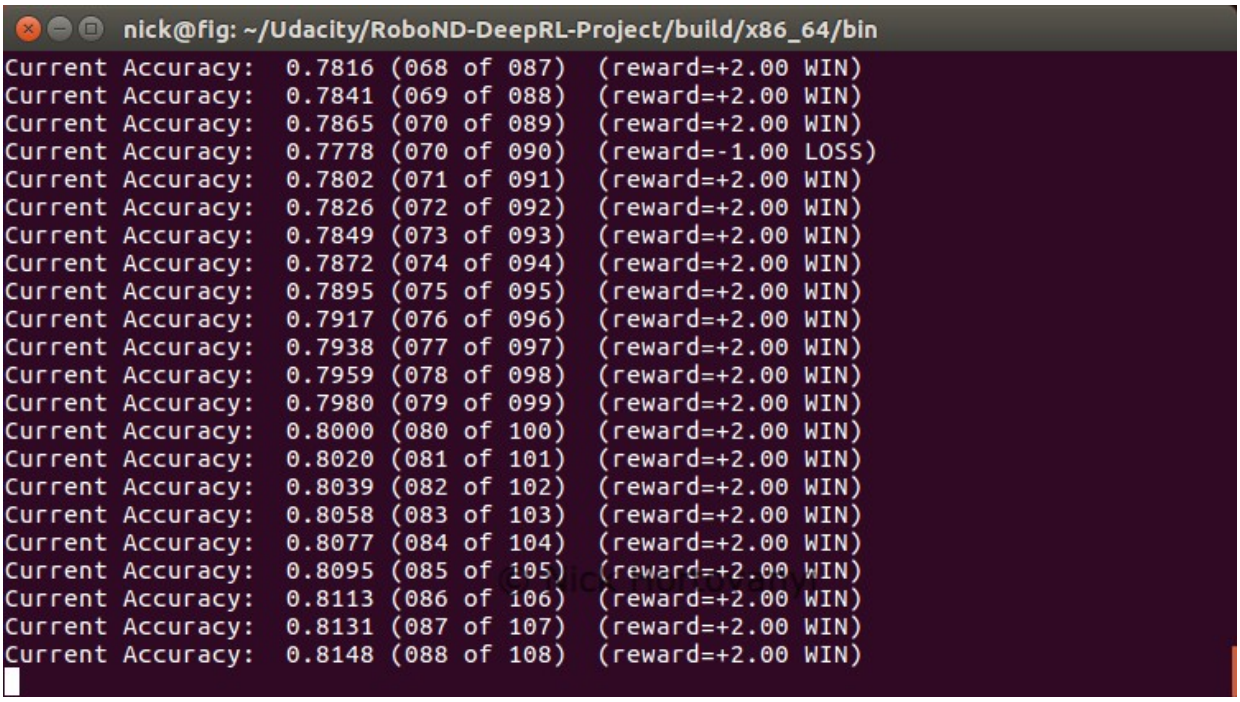
Results objective 1

The robotic arm quickly learnt how to hit the prop with a degree accuracy in a repeatable fashion. On occasion if the arm trained initially away from the prop, it would take longer to achieve a higher accuracy.

Once a winning path was learnt this configuration consistently had the robotic arm quickly hitting the prop objective.

As can be seen in the above summary output the objective was achieved well within the criteria specified.

Objective 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.



Results objective 2

With the finer control required, and alteration to the interim reward system, this configuration would often hesitate before making a move. Whilst it learnt quickly how to get very close to having the `gripper_middle` hit the prop, it would also often just miss either hitting the ground or the arm itself hitting the prop. There seemed to be a repeatable pattern, of just extending past and swinging down in an arch, that once learnt gave consistent winning results.

Occasionally the middle joint would collide with the ground and this would lead to the objective not being met.

This configuration was not always reproducible, however with the above screen shot it was able to meet the objectives.

Future work

There were clear arcs that once found achieved a win quickly. Such that it would be worthwhile investigating an interim reward system based on not just the distance from the goal but also distance from an ideal arc trajectory as the arm approached.

Further using centre points to calculate distance from goals becomes less accurate the closer to the goal the arm is. Such that other points like the end of the gripper_middle and top of prop cylinder, would be worthwhile experimenting with.