Vision Based Robot Manipulation Testbed for Reinforcement Learning

Department for Interdisciplinary Research
College of Engineering Trivandrum

July 7, 2020

Guided by,
Linu Shine
Electronics and Comm. Engg.

Presented by,
Sreejith Krishnan R
Robotics & Automation

Motivation

Why robot manipulation?

- In order to assist in general tasks, autonomous robots should be able to interact with dynamic objects in unstructured environments
- Designing machines that can grasp and manipulate objects with anything approaching human levels of dexterity is first on the to-do list for robotics [1]

A reinforcement learning testbed provides,

- A standardized benchmarking environment for comparing performance of different RL algorithms
- An entry point for quickly testing RL algorithms for robot manipulation tasks enabling quick development
- A high performance framework for efficient training of RL algorithms

Literature survey I

Title	SURREAL: Open-Source Reinforcement Learning Framework and Robot Manipulation Benchmark [2] - Conference on Robot Learning - 2018
Methodology	Provides an open-source framework for benchmarking reinforcement learning algorithms for different robot manipulation tasks. Decomposed architecture enables scaling reinforcement learning speed with computational power
Merits	• Allows scaling RL speed with computation power • Prebuilt standardized environments for common robot manipulation tasks
Demerits	• No integration with RayLib - A common framework for scalable reinforcement learning • No prebuilt support for experiment logging and tracking

Literature survey II

Title	Comparing Task Simplifications to Learn Closed-Loop Object Picking Using Deep Reinforcement Learning [3] - IEEE Robotics and Automation Letters - 2019
Methodology	Uses autoencoder to reduce dimensionality of camera data which is given to 3 layer CNN to get a low dimensional encoding. This encoding is used by a 2 layer feed-forward neural network to predict the optimum action. Uses RL to train the mentioned networks
Merits	• No hand labeled data required
Demerits	• Low success rate (78%) for manipulation of objects in clutter by real robot • Non modular. Difficult to reuse model for similar task

Literature survey III

Title	Regularized Hierarchical Policies for Compositional Transfer in Robotics [4] - DeepMind - 2019
Methodology	Use hierarchical modular policies for continuous control.
Merits	• Best sample efficiency on both simulated and real robot • Uses MPO optimization algorithm which reduces the number of hyperparameters
Demerits	• High level tasks are not automatically decomposed to sub tasks • Low level policy is shared across all low level tasks making interpretability complicated • Transferring specific skills from sub-tasks policy in a predictable manner is difficult • Experiment results are obtained using model whose inputs include pose of objects in workspace

Research Gap and Objectives

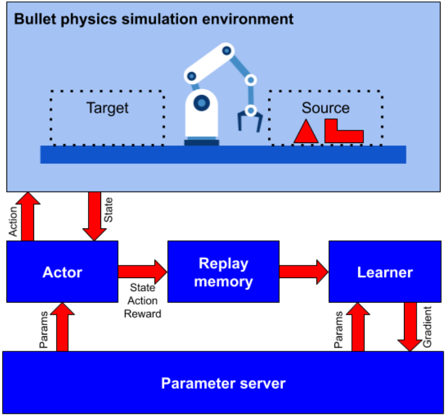
Research Gap

- Slow training data collection speed
- Non standard/readily available frameworks used

Objective

- Improve RL model training speed by running multiple simulations in parallel
- Use standard frameworks that support training distributed RL algorithms
- Prebuilt experiment logging and tracking
- Flexibility for adding different types of robot manipulation tasks like grasping, moving etc.

- Simulation environment will be created on Bullet Physics Simulator - An open source physics engine commonly used for Reinforcement learning
- For fast experiment feedback loop, experiments will be run in distributed fashion. Simulation environments will be tuned for maximum throughput.



- For scaling reinforcement learning this testbed will be integrated with RayLib

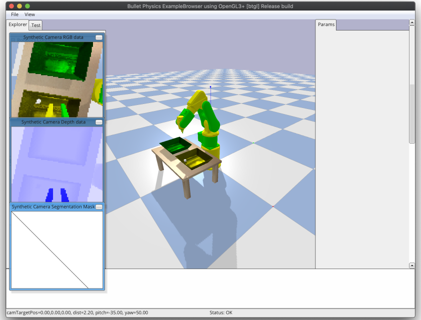
Methodology II

- RayLib provides framework for scaling RL algorithm training speed with computation power by running different instances of actor in parallel. This framework also supports training RL models in multi-GPU environments.
- For experiment logging, tracking and visualization, the testbed will be integrated with comet.ml platform. Different metrics and model parameters will be send to comet.ml platform at regular epoch intervals. These metrics can be visualized in realtime by using the web dashboard provided by the platform
- Testbed will be flexible to allow training RL algorithms on robot manipulation tasks which is not prebuilt to the platform.

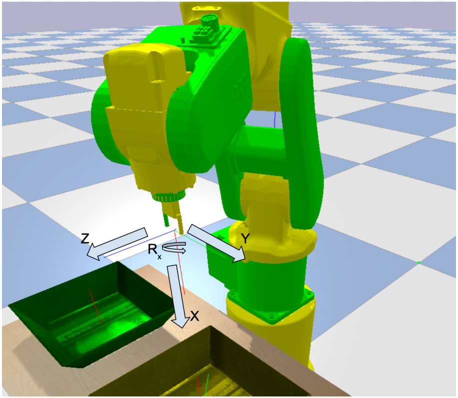
Results

ABB IRB 120 simulator

- PD control tuned for 1mm positioning accuracy which is same as ABB IRB 120 robot positioning accuracy
- Rendering can be configured to be disabled for faster training data collection
- Multiple simulator instances can be run in parallel to scale up agent sampling throughput
- Grasp detection and collision detection algorithms



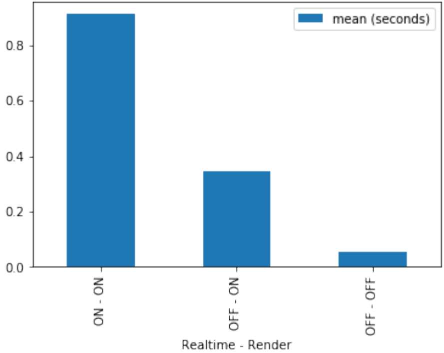
- Task is to move the object from yellow tray to green tray
- Only input to RL model is the depth image from RGB-D camera mounted on end effector. Observation space is of shape 84x84x4
- RL model can move and rotate the end effector by providing input relative to coordinate frame attached to end effector. Binary variable can be modified to open / close gripper. Action space is $[\delta x, \delta y, \delta z, \delta r_x, open]$



Results

Simulator performance

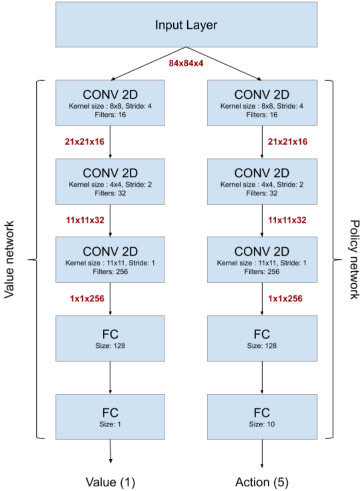
- Mean action time of 0.053 seconds without rendering and 0.345 seconds with rendering
- 2.4 times faster than data collection from real robot when rendering is enabled
- 17.2 times faster than data collection from real robot when rendering is disabled



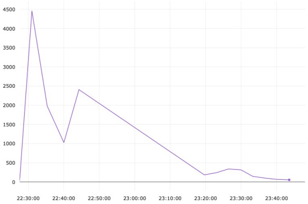
Render — Realtime	Mean	Std	25%	50%	75%
ON — ON	0.912	2.783	0.358	0.374	0.39
ON — OFF	0.345	0.583	0.198	0.206	0.214
OFF — OFF	0.053	0.023	0.046	0.047	0.048

- Convolution layers are used for feature extraction
- Input in RGB-D 84x84x4 matrix
- Value network predicts the how good it is to be at a particular state
- Policy network directly predicts the mean and standard deviation of a Gaussian PDF from which actions are sampled for a particular state
- Train batch size is 10240 and SGD minibatch size is 512. Number of iterations per train batch is 30
- Data from an episode is added to train batch only after the it is complete

Architecture

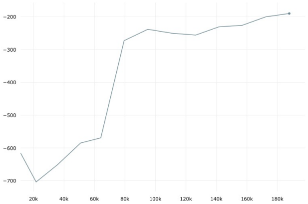


Results
Baseline PPO

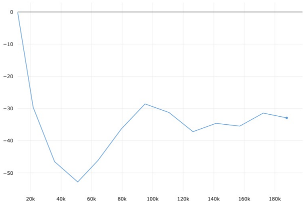


Value network loss

Policy network loss



Collision penalty

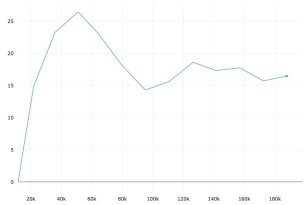


Drop penalty

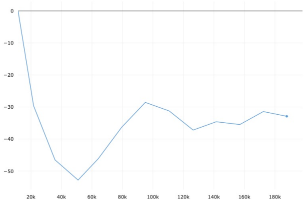
Results

Baseline PPO

- Reward shaping is critical. Small changes in reward function can change the learning process.
- Providing +ve reward when end effector moves towards target and -ve reward when end effector moves away will not work
- Initializing each episode at a random state like grasped and not grasped can improve training speed



Grasp Reward



Drop Penalty

Future scope

- Including DDPG and RHPO baseline models
- Evaluating baseline models on real robot
- Including more common robot manipulation tasks to testbed
- Including baseline support for multi agent robot manipulation tasks

Challenges I

- Initially objective of this project was to evaluate Regularized Hierarchical Policy Optimization reinforcement learning algorithm for robot manipulation tasks
- Reinforcement learning (RL) models have low sample efficiency and require lot of data for training. This data is generated by simulator
- For fast data collection, multiple instances of simulator is run in parallel which requires multiple CPU cores (32 cores recommended)
- Neural networks of RL models are build using Tensorflow or PyTorch framework. For fast training of the deep neural networks used, GPU is required (24GB GPU recommended)
- In CET, computer cluster with more than 32 CPU cores is available. But it does not have GPU

Challenges II

- Efforts were made to try and use compute cluster from CET for running simulations and use GPU from google colab / cloud for training neural networks. But slow networking speed between CET computer cluster and cloud made the training process extremely slow
- Also efforts were made to use both GPU and CPU from google cloud by using 300 USD trial provided by google cloud. To limit the compute costs within 300 USD trial limit, a particular class of virtual machines known as preemptible instances needs to be used. These instances will run for a maximum of 24 hours, but may be terminated earlier if demand is high for same VM configuration. Recently due to unknown reasons, preemptible instances with GPUs are terminated after for around 15 mins. This prevented us from google cloud for training.

Challenges III

- Hence the objective of the project was changed to development of testbed which require lower compute power since only simulator development is required.

Demo

References I

Richard Hodson.
How robots are grasping the art of gripping.
https://www.nature.com/articles/d41586-018-05093-1.
Accessed: 2019-09-24.

Linxi Fan, Yuke Zhu, Jiren Zhu, Zihua Liu, Orien Zeng, Anchit Gupta, Joan Creus-Costa, Silvio Savarese, and Li Fei-Fei.
Surreal: Open-source reinforcement learning framework and robot manipulation benchmark.
In *Conference on Robot Learning*, 2018.

M. Breyer, F. Furrer, T. Novkovic, R. Siegwart, and J. Nieto.
Comparing task simplifications to learn closed-loop object picking using deep reinforcement learning.
IEEE Robotics and Automation Letters, 4(2):1549–1556, April 2019.

References II

Markus Wulfmeier, Abbas Abdolmaleki, Roland Hafner, Jost Tobias Springenberg, Michael Neunert, Tim Hertweck, Thomas Lampe, Noah Siegel, Nicolas Heess, and Martin A. Riedmiller.
Regularized hierarchical policies for compositional transfer in robotics.
CoRR, abs/1906.11228, 2019.