Grasping Manipulation for Throwing a Ball under Constraints

Rohith Peddi
Abhiramon Rajasekharan
Dat Ngo

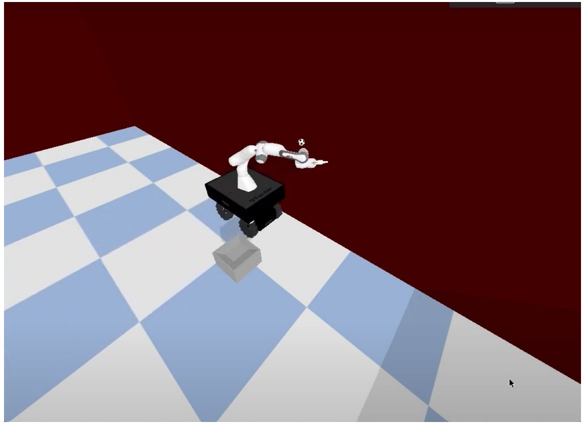
Objective

Motivation:

- Throwing can increase physical reachability and picking speed of an arm
- Past work : Tossing Bots

Proposed Objective:

- Investigate learning of a robot arm to throw under constraints
- Constraint:
  - Throw a ball into the box after bouncing off a wall.



https://tossingbot.cs.princeton.edu/paper.pdf

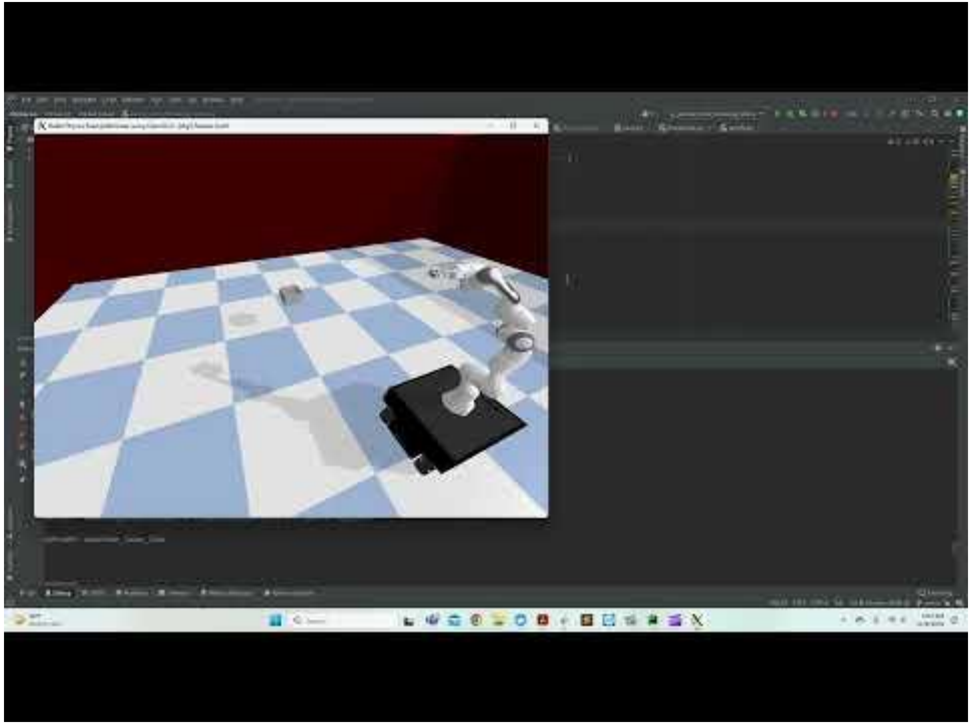
Environment

- PyBullet
- Stable Baselines : Train model using RL (Deep Deterministic Policy Gradient)
- Ruckig : A motion planning library for robotic applications.
- Open AI-Gym

Model-based + Model-Free Results

Throwing ball into boxes initialized at random positions

This involves both
1. Robot Navigation
2. Goal driven robot throwing motion

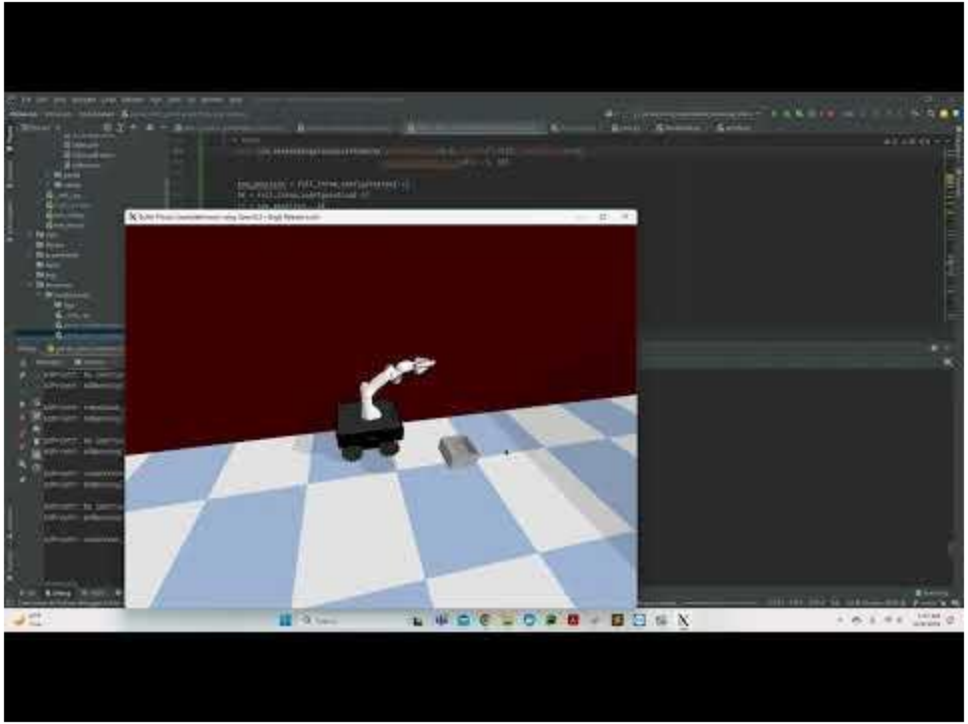


Model-based + Model-Free Results

Throwing ball into baskets initialized at random positions under a constraint of hitting the wall.

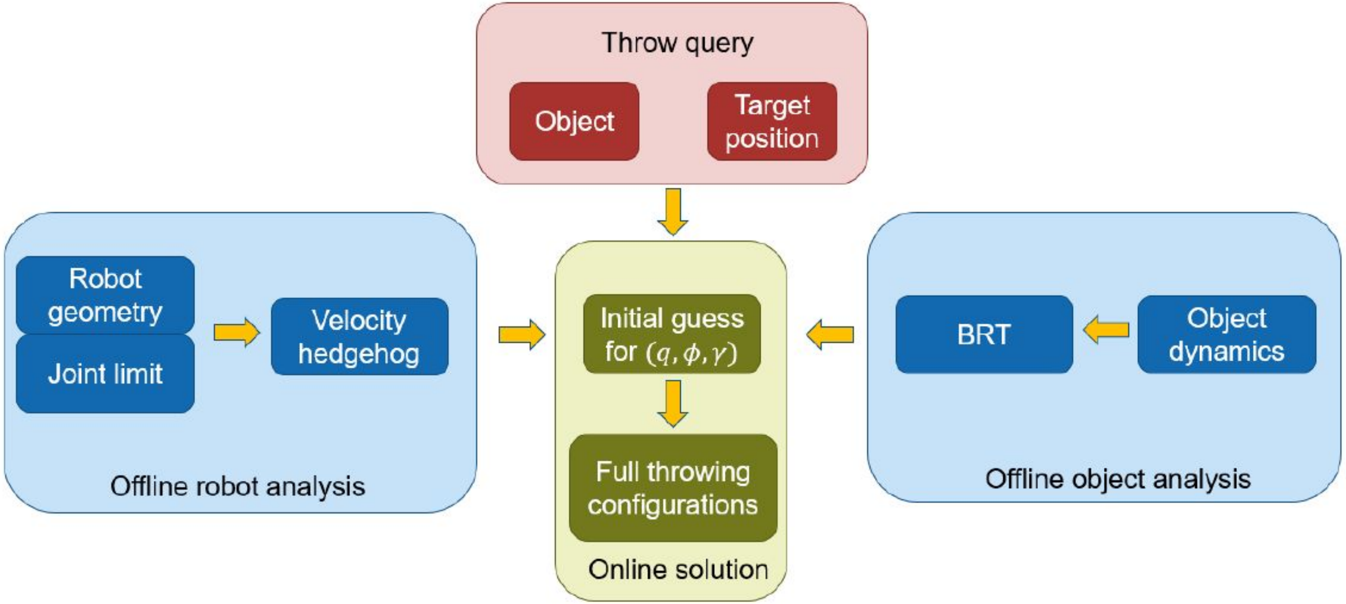
This involves both

1. Robot Navigation
2. Goal driven robot throwing motion under constraint



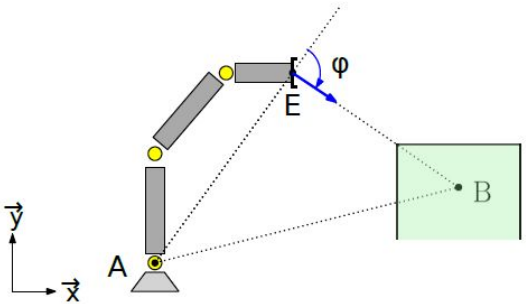
Model-based + Model-Free

Questions:
1. Appropriate representation for throwing?
2. How to separate mundane computation offline while ensuring reliable and efficient online solution generation?

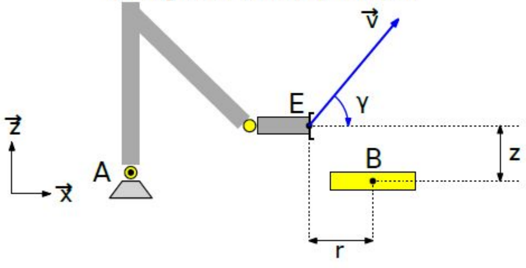


ɔs/2207.10629

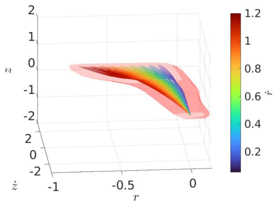
Model-based + Model-Free



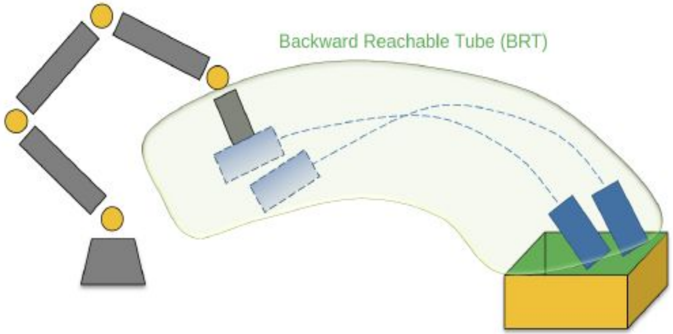
(a) Top view (throwing triangle)



(b) Side view (throwing plane)



Model based approach
Throwing as a feasibility problem



Model Free approach
Model non-linear object dynamics

Throwing task

1. Object flying dynamics
2. Target landing position
3. Landing velocity range

Set of valid throwing configurations as BRT

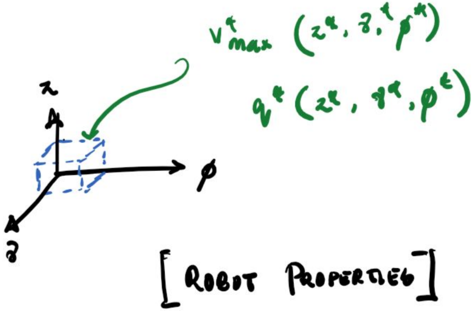
Bring the object towards the BRT and release the object once the robot end-effector enters the BRT

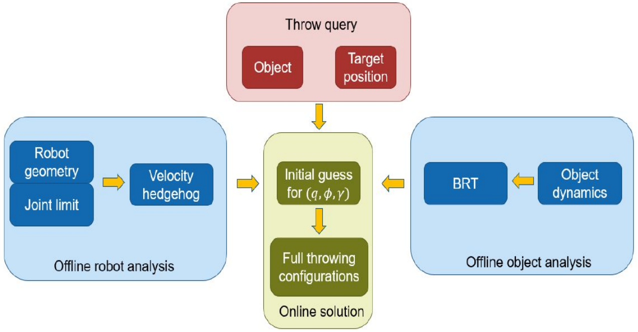
Model-based + Model-Free

$$\text{Find } \{\overrightarrow{AB}, q, \dot{r}, \dot{z}\}$$

$$\text{such that: } \begin{cases} q_{\min} \leq q \leq q_{\max} \\ \dot{q}_{\min} \leq J^{\dagger}(q)\vec{v}(\overrightarrow{AB}, q, \dot{r}, \dot{z}) \leq \dot{q}_{\max} \\ f_{BRT}(r(\overrightarrow{AB}, q), z(\overrightarrow{AB}, q), \dot{r}, \dot{z}) \leq 0 \end{cases}$$

$$\dot{x} = f(x) = \frac{d}{dt}\begin{bmatrix} r \\ z \\ \dot{r} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \dot{r} \\ \dot{z} \\ 0 \\ -g \end{bmatrix}$$





$$\mathcal{X}_l = \left\{ \begin{array}{c} r = 0 \\ z = 0 \\ 0.2 \leq \dot{r} \leq 2.0 \\ -5.0 \leq \dot{z} \leq -2.0 \end{array} \right\}.$$

Model-Free

We pursued different ways to approach this problem

APPROACH - 1

Action Space:

1. Joint angles
2. End effector velocity
3. Throwing yaw angle

$$(7 + 1 + 2)$$

$$[q_0, q_1, q_2 \ldots, q_6, \phi, \dot{r}, \dot{z}]$$

Reward:

Weighted function of

1. Final distance of the ball from box
2. Distance of the end effector from the box

Model: Deep Deterministic Policy Gradient (DDPG)

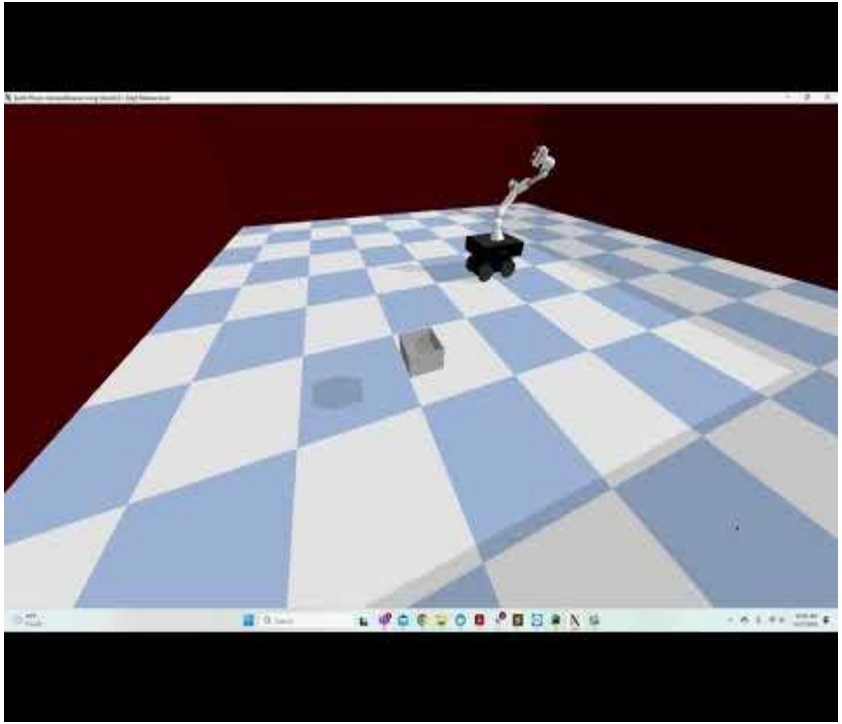
Model-Free

<u>APPROACH - 2</u>

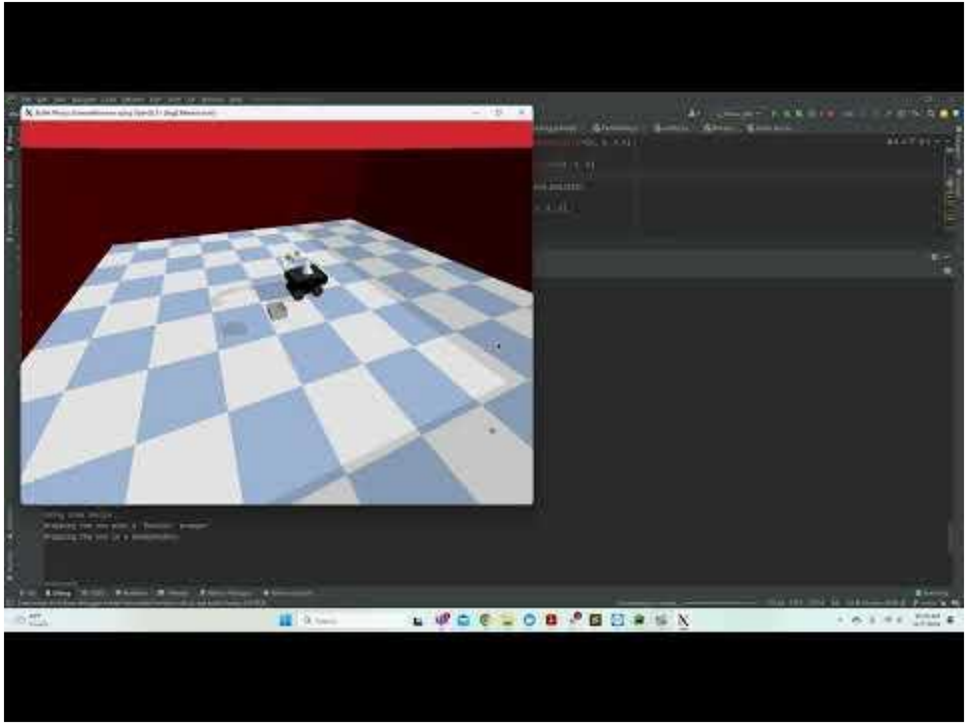
Action Space:

1. Joint angles
2. End effector velocity
3. Throwing yaw angle
4. Use the results from model based approach as a prior

Model-Free Results



Approach -1



Approach -2

Future Work

- Reproduce Tossing Bot (code and models are not public) for performance comparison
- Model Free Training
  - Training RL Algorithms is an art.
    - We hope to crack it
  - Add different objects with different flying dynamics
- Model Based Training
  - Extend it to different objects with varying flying dynamics
  - Using knowledge from model based approaches to aid training/evaluation of end-to-end learning.

Questions?