Course Project - Core Reinforcement Learning

Release: 2/16/2023; Report Due: Friday 2/24/2023, 11:59 PM

- You are **allowed** to consult any external resources, but you must cite them. You are also **allowed** to discuss with each other, but you need to acknowledge them. Members of the same team can submit the same copy; however, please mark down how each one contributed.
- This is a mandatory project which helps you to gain experience in tuning RL algorithms. A report (a project milestone report) is required as a single PDF file containing a description of your approach and results. Please see the detailed instructions below.
- This project is worth 20/100 of your final grade and will be graded based on the demonstration of your effort.
- It is highly recommended that you begin working on this assignment early.

1 Warm-up

- Please go over the tutorials at `https://maniskill2.github.io/#resources`.
- Please read the papers of ManiSkill (`https://arxiv.org/pdf/2107.14483.pdf`) and ManiSkill2 (`https://arxiv.org/pdf/2302.04659.pdf`).

2 Project Description

You are asked to train state-based RL agents (meaning `obs_mode=state`) to solve 5 object manipulation tasks (see Section 3). To start with, refer to the state-based RL section in the Google Colab RL tutorial of ManiSkill2. You can use whatever RL algorithms and libraries to train the agents (remember to cite the source). While you do not need to solve all tasks (meaning a relatively high success rate), it might prepare you better for the open-ended final projects which will be released soon. We expect you to show decent effort in tuning RL agents of **at least** two tasks by presenting a description of your approach, the training curves and the task success rates in your PDF report (*around two pages length*). You should also include a link to Google Drive that contains your source code and videos of 3 trajectories per task (not necessarily successful trajectories).

Note that this is NOT a competition, and we expect that you understand the code and mechanisms through practice. We will grade your report not only by checking the performance, but more importantly, by your description of phenomenons.

3 Environments

We will use the following environments, a subset of ManiSkill2, for this core RL project. The tasks range from different levels of difficulty (not ordered) with LiftCube as the easiest.

- LiftCube (`LiftCube-v1`)
- StackCube (`StackCube-v1`)
- PegInsertion (`PegInsertionSide-v1`)

- TurnFaucet (`TurnFaucet-v1`, which contains 10 different faucet models, a simpler subset of all faucet models; `TurnFaucet-v2`, which contains all 60 faucet models)
- PushChair, (`PushChair-v1`, which contains 5 different chair models, a simpler subset of all chair models; `PushChair-v2`, which contains all 26 chair models)

If you are interested in the details of each environment, you can read the appendix of the original paper.

4 Detailed Instructions

Below are some instructions you might find helpful for this project.

- Please uninstall the previous version of Maniskill2 and install a new version (the `cse291-projects` branch), e.g., by running

  ```
  pip install git+https://github.com/haosulab/ManiSkill2.git@cse291-projects
  ```

- We highly recommend you use `python3.8`.
- For PushChair, you will be using a mobile robot and we recommend you to use

  ```
  control_mode=base_pd_joint_vel_arm_pd_joint_vel
  ```

  For all the other tasks, we recommend `pd_ee_pos(e)`. You are welcome to explore other control modes as they might be harder or easier for RL training.
- Starter code for evaluating the RL agents is provided in `ManiSkill2/examples/eval_starter.py` at the `cse291-projects` branch.