RL for object grasping by robot (Franka Emika)

Overview

Prerequisites

Note: Tensorflow currently (01.2020) is not working with python in version 3.8! This was tested with python3.6.9.

System

You'll need system packages: **CMake**, **OpenMPI** and **zlib**. Those can be installed as follows

Ubuntu

```
$ sudo apt-get update && sudo apt-get install cmake libopenmpi-dev python3-dev zlib1g-dev
```

Mac OS X

Installation of system packages on Mac requires Homebrew. With Homebrew installed, run the following:

```
$ brew install cmake openmpi
```

Windows 10

To install stable-baselines on Windows, please look at the documentation.

Python

This project **requires *python3 (>=3.5 and <=3.7)*** with the development headers.

Before installing the required dependencies, you may want to create a virtual environment (using pyenv) with specified python version and choose it:

```
$ pyenv virtualenv 3.6.9 python3.6.9
$ pyenv shell python3.6.9
```

(After work) To return to system python:

```
$ pyenv shell system
```

If it does not work you may need to add those lines in *.bash_profile* or .bashrc file depending on terminal for pyenv to work:

```
$ eval "$(pyenv init -)"
$ eval "$(pyenv virtualenv-init -)"
```

Installation

Clone the repository:

```
$ git clone https://github.com/Sladzio/rl-for-object-grasping.git
$ cd rl-for-object-grasping
```

Install using requirements

Install all the necessary dependencies:

```
$ pip3 install -r requirements.txt
```

Note: Installing the requirements will install also Stable Baselines, Pybullet, Tensorflow and Gym.

OR

Install using pip

Install the Stable Baselines package:

```
$ pip3 install stable-baselines[mpi]
```

This includes an optional dependency on MPI, enabling algorithms DDPG, GAIL, PPO1 and TRPO. If you do not need these algorithms, you can install without MPI:

```
$ pip3 install stable-baselines
```

Install the Pybullet package:

```
$ pip3 install pybullet
```

Install the ruamel.yaml package:

```
$ pip3 install ruamel.yaml
```

Install the Tensorflow 1.x (1.8 <= x <= 1.15) package, required by Stable Baselines:

```
$ pip install tensorflow==1.15
```

Documentation

Stable Baselines online documentation for more details or in pdf file.

Pybullet documentation for more details.

Tensorboard

TensorBoard is a tool for providing the measurements and visualizations needed during the machine learning workflow. It enables tracking experiment metrics like loss and accuracy, visualizing the model graph, projecting embeddings to a lower dimensional space, and much more.

1. Run tensorboard by specifying the log directory:

```
$ tensorboard --logdir ../tensorboard
  TensorBoard 1.13.1 at <url>:6006 (Press CTRL+C to quit)
```

2. Enter the \<url>:6006 into the web browser and track the mean reward per episode.

Train and Test

Training

To train model with DQN algorithm and tuned version of hyperparameters with enabled rotation of grasped object and save checkpoint models every 25000 steps. Final model is concluded from evaluation every 50000 steps on 25 esisodes. Both final model and checkpoint models it this case will be saved in './DQN/DQN_ENABLED_ROT_TUNED/models' directory. Run train.py file, with parameters:

```
$ python train.py --algo DQN --tag TUNED --lockRot False --saveFreq 25000 --evalFreq 50000 --evalNum 25
```

Another example to train with locked rotation of grasped object and DDPG algorithm with noise imposed on actions and save checkpoint models every 20000 steps. Models *.zip will be saved in './DDPG/DDPG_LOCKED_ROT_TUNED/models' directory.

```
$ python train.py --algo DDPG --tag TUNED --lockRot True --saveFreq 20000
```

For more options type:

```
$ python train.py --help
```

Test and evaluate policy

For testing model trained by DQN algorithm you need to specify directory path with trained models and concrete version of model and specify for obserbvation if object rotation should be locked:

```
$ python evaluate.py --algo DQN --dir DQN/DQN_ENABLED_ROT_TUNED/models/ --model rl_model_1150000_steps.zip --lockRot False
```

You can additionally turn of visualization for faster evaluation and evaluate on 5000 episodes instead of 100:

```
$ python evaluate.py --algo TD3 --dir TD3/TD3_LOCKED_ROT_TUNED/models/ --model best_model.zip --lockRot True --render False
                               --evalNum 5000
```

If you don't want to perform any evaluation and just simply watch object being grasped you can turn it off by setting eval flag to False:

```
$ python evaluate.py --algo DDPG --dir DDPG/DDPG_LOCKED_ROT_TUNED/models/ --model rl_model_200000_steps.zip --lockRot False
                               --eval False
```

For more options run:

```
$ python evaluate.py --help
```

Already trained models

You can also try best models already trained for both experiments located at './BestModels/' folder\ For experiment with random position and locked rotation of grasped object with DQN algorithm:

```
$ python evaluate.py --algo DQN --model DQN_locked_rot.zip
```

For experiment with random pose of grasped object with TD3 algorithm:

```
$ python evaluate.py --algo TD3 --model TD3_rotating.zip -l False
```