Day 1. Settings

NPEX Reinforcement Learning

July 26, 2021

Jaeuk Shin, Minkyu Park

CORE
Control + Optimization Research Lab

Contents

CORE
Control + Optimization Research Lab

Anaconda

www.anaconda.com/

Handy management of separate environments

Convenient installation of scientific computing libraries →
numpy, scipy, pandas, matplotlib, tensorflow, PyTorch, … etc.

Anaconda

Create your environment:

```
sju5379@sju5379-System-Product-Name:~$ conda create -n npex python=3.6
Collecting package metadata (current_repodata.json): done
Solving environment: done
```

Check whether the environment is successfully created:

```
sju5379@sju5379-System-Product-Name:~$ conda env list
# conda environments:
#
base                  *  /home/sju5379/anaconda3
baselines                /home/sju5379/anaconda3/envs/baselines
drl                      /home/sju5379/anaconda3/envs/drl
npex                     /home/sju5379/anaconda3/envs/npex
torchbeast               /home/sju5379/anaconda3/envs/torchbeast
tsearch                  /home/sju5379/anaconda3/envs/tsearch
```

Anaconda

Activate the created environment:

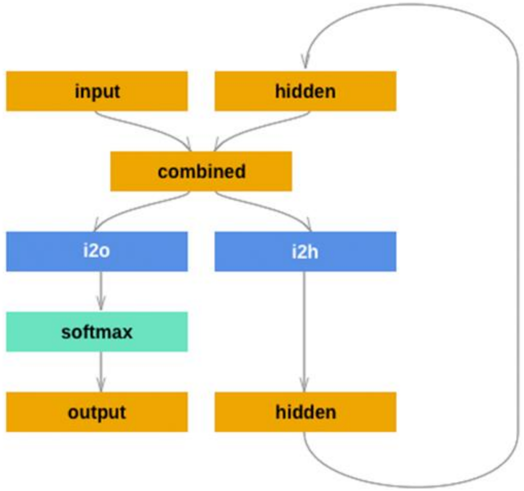
```
sju5379@sju5379-System-Product-Name:~$ conda activate npex
(npex) sju5379@sju5379-System-Product-Name:~$ conda install
```

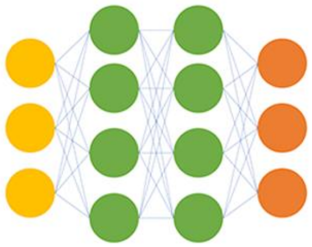
```
(npex) sju5379@sju5379-System-Product-Name:~$ conda list
# packages in environment at /home/sju5379/anaconda3/envs/npex:
#
# Name                    Version                   Build  Channel
_libgcc_mutex             0.1                        main
blas                      1.0                         mkl
ca-certificates           2020.6.24                     0
certifi                   2020.6.20                py36_0
cloudpickle               1.3.0                    pypi_0    pypi
```

For more info:

https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html

PyTorch





torch.nn

High-level Deep Learning Framework

Efficient Building/Training of large-scale models (ex. OpenAI GPT-3)

No such DL frameworks? → multiprocessing, CUDA, etc.

PyTorch - Installation

In your Conda env:

conda install pytorch torchvision cpuonly –c pytorch

```
(npex) sju5379@sju5379-System-Product-Name:~$ conda install pytorch torchvision
cpuonly -c pytorch
```

For the class, we will use cpu-only version(if you have already installed gpu version, it doesn’t matter)

pytorch.org/

Anaconda

Install git by running the following command:

conda install –c anaconda git

and run

git clone https://github.com/npex2020rl/rl.git

CORE
Control + Optimization Research Lab

PyTorch - Examples

Open torch_test.py

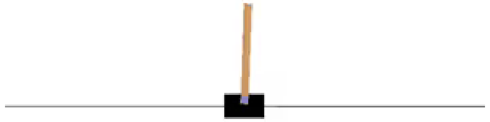
When computing the forwards pass, autograd simultaneously performs the requested computations and builds up a graph representing the function that computes the gradient.

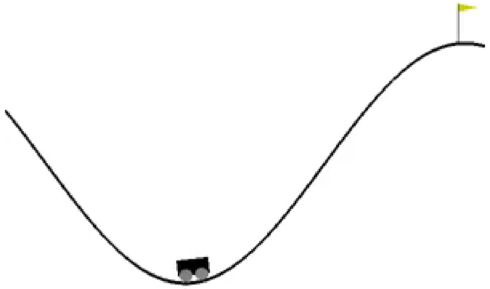
Note : graph is recreated from scratch at **every iteration**!

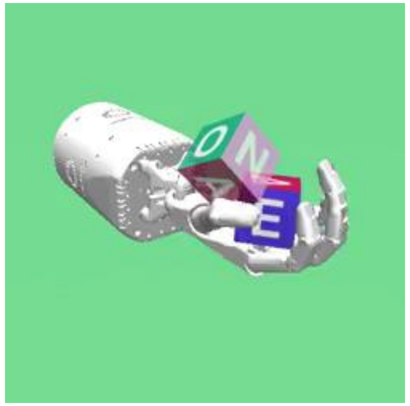
[Reference] pytorch.org/docs/stable/notes/autograd.html

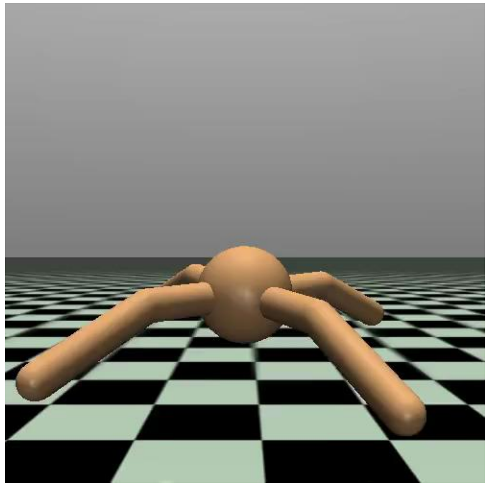
OpenAI Gym

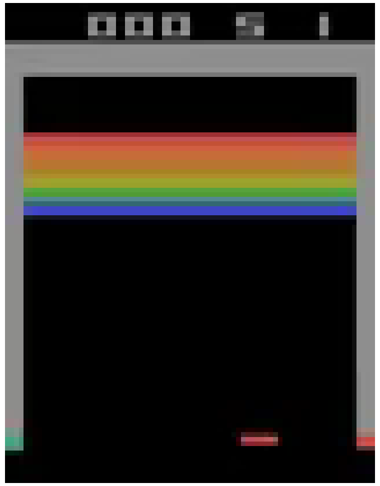
provides various types of RL benchmark problems











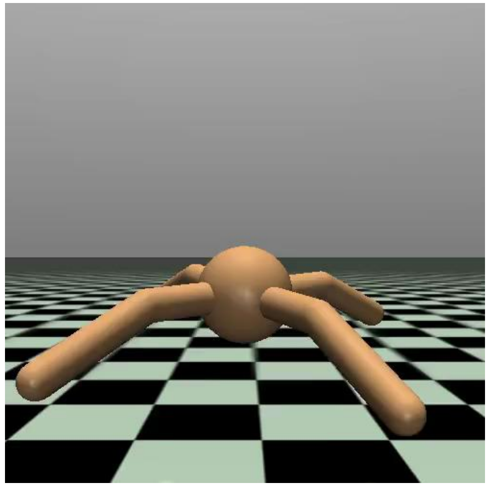
Brockman, Greg, et al. "Openai gym." *arXiv preprint arXiv:1606.01540* (2016).
https://gym.openai.com/

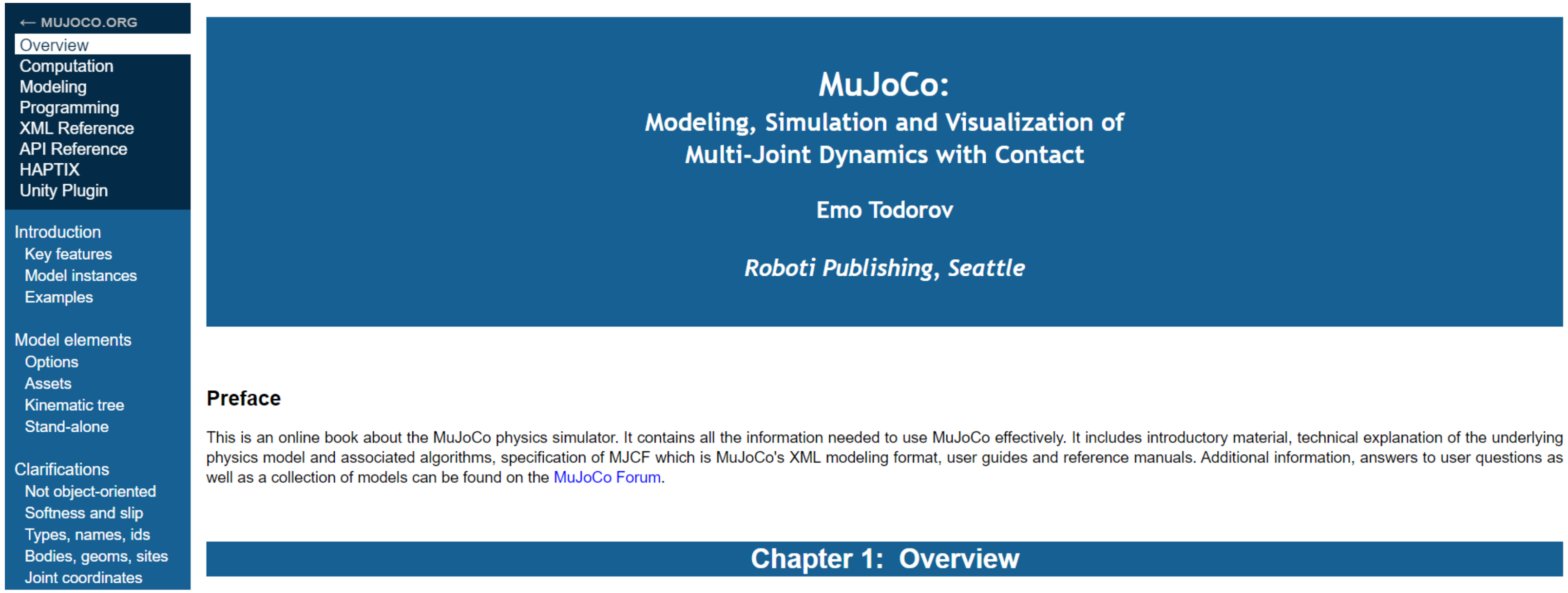
OpenAI Gym

high-level API for agent-environment interaction

Intuitive abstraction, easy interface

Imagine MuJoCo without Gym




← MUJOCO.ORG
Overview
Computation
Modeling
Programming
XML Reference
API Reference
HAPTIX
Unity Plugin

Introduction
Key features
Model instances
Examples

Model elements
Options
Assets
Kinematic tree
Stand-alone

Clarifications
Not object-oriented
Softness and slip
Types, names, ids
Bodies, geoms, sites
Joint coordinates

MuJoCo:
Modeling, Simulation and Visualization of Multi-Joint Dynamics with Contact

Emo Todorov

Roboti Publishing, Seattle

Preface

This is an online book about the MuJoCo physics simulator. It contains all the information needed to use MuJoCo effectively. It includes introductory material, technical explanation of the underlying physics model and associated algorithms, specification of MJCF which is MuJoCo's XML modeling format, user guides and reference manuals. Additional information, answers to user questions as well as a collection of models can be found on the MuJoCo Forum.

Chapter 1: Overview

OpenAI Gym - Installation

In your Conda env:

pip install gym

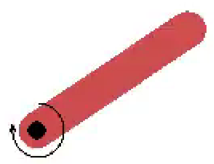
```
(npex) sju5379@sju5379-System-Product-Name:~$ pip install gym
Processing ./.cache/pip/wheels/be/a1/84/6b4caa6c1cea703acbfea8a24cc3c1729bd359cd
4a65755d8b/gym-0.17.2-py3-none-any.whl
Collecting scipy
  Downloading scipy-1.5.2-cp36-cp36m-manylinux1_x86_64.whl (25.9 MB)
     |████████████████████████████████| 25.9 MB 1.3 MB/s
```

gym.openai.com
github.com/openai/gym

OpenAI Gym - Examples

Pendulum Swing Up

Goal : keep a frictionless pendulum standing up



OpenAI Gym - Examples

Open gym_test.py

```python
import gym


env = gym.make('Pendulum-v0')
state = env.reset()

for _ in range(200):
    state, reward, done, info = env.step(env.action_space.sample())
    env.render()

env.close()
```

Class Env(object):

action_space =
observation_space =

def step(self, action):

def reset(self):

def render(self, mode='human):

def close(self):

OpenAI Gym - Examples

See pendulum.py

```
class PendulumEnv(gym.Env):
    metadata = {
        'render.modes': ['human', 'rgb_array'],
        'video.frames_per_second': 30
    }

    def __init__(self, g=10.0):
        self.max_speed = 8
        self.max_torque = 2.
        self.dt = .05
        self.g = g
        self.m = 1.
        self.l = 1.
        self.viewer = None

        high = np.array([1., 1., self.max_speed], dtype=np.float32)
        self.action_space = spaces.Box(
            low=-self.max_torque,
            high=self.max_torque, shape=(1,),
            dtype=np.float32
        )
        self.observation_space = spaces.Box(
            low=-high,
            high=high,
            dtype=np.float32
        )

        self.seed()
```

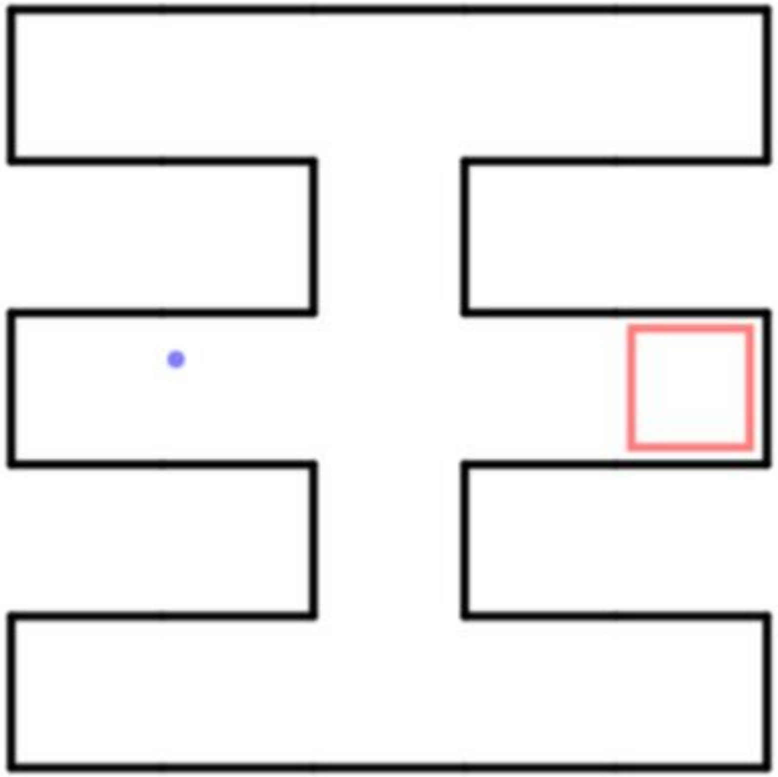
Why observation space instead of state space?
Notion of **Partially-Observable MDP(POMDP)**

determine the **state space** $\mathcal{S}$
and the **action space** $\mathcal{A}$

CORE
Control + Optimization Research Lab

OpenAI Gym - Examples

(a) Sample observation



(b) Layout of the 5×5 maze in (a)

state space?

observation space?

OpenAI Gym - Examples

```python
def reset(self):
    high = np.array([np.pi, 1])
    self.state = self.np_random.uniform(low=-high, high=high)
    self.last_u = None
    return self._get_obs()
```

sample an initial state s_0

```python
def _get_obs(self):
    theta, thetadot = self.state
    return np.array([np.cos(theta), np.sin(theta), thetadot])
```

$$\text{state}: \ s = (\theta, \dot{\theta})$$

$$\text{observation}: \ o = (\cos\theta, \sin\theta, \dot{\theta}))$$

Jupyter Notebook

We will use Jupyter Notebook to write code

In your Conda env:

conda install –c conda-forge notebook

Then, run notebook with:

jupyter notebook

Google Colab

If you don’t have GPU or you are only available with windows, you can use Google Colab instead of installing Jupyter

https://colab.research.google.com/notebooks/intro.ipynb?utm_source=scs-index/

Free GPU session with basic libraries pre-installed

Upload the tutorial session directory to your google drive

Thank you!