Day 1. Settings

SAMSUNG AI

Reinforcement Learning

June 17, 2022

Jaeuk Shin, Mingyu Park

CORE
Control + Optimization Research Lab

For RL course, we need…

- Google Colab
- PyTorch
- OpenAI Gym

Google Colab

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

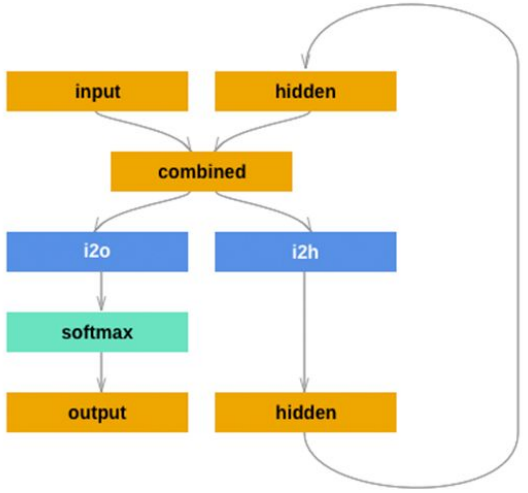
https://colab.research.google.com/notebooks/intro.ipynb?utm_source=scs-index/

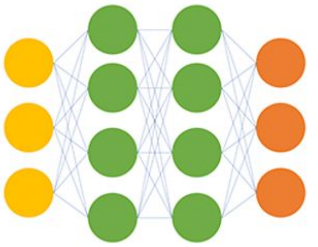
Free GPU session with basic libraries pre-installed

Upload the tutorial session directory to your google drive

PyTorch

PyTorch





torch.nn

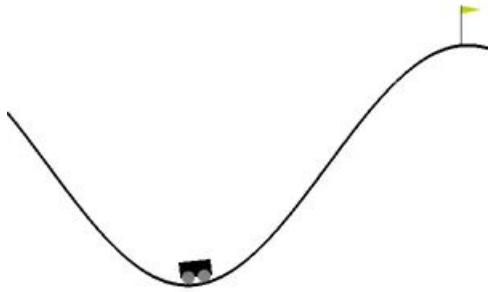
High-level Deep Learning Framework

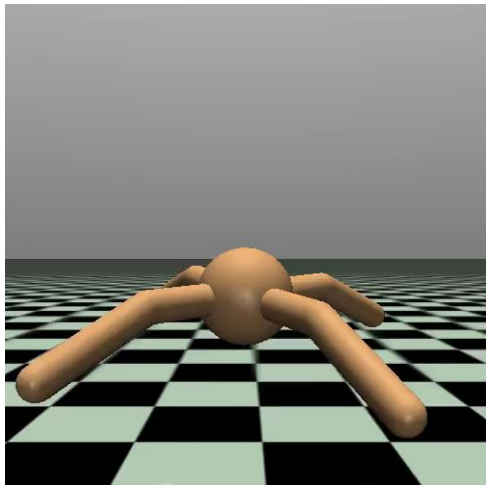
Efficient Building/Training of large-scale models (ex. OpenAI GPT-3)

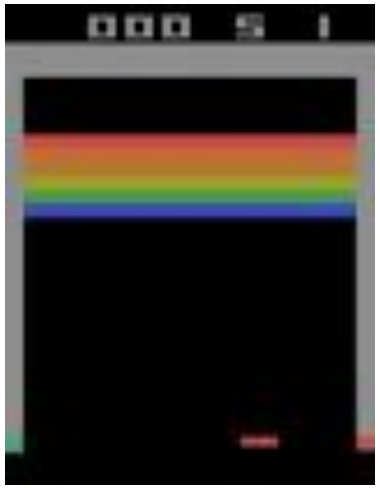
No such DL frameworks? → multiprocessing, CUDA, etc.

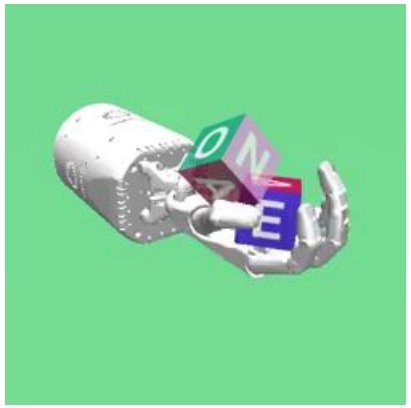
OpenAI Gym

provides various types of RL benchmark problems









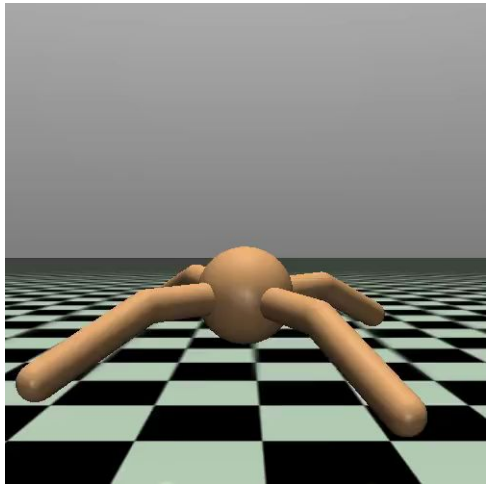
Brockman, Greg, et al. "Openai gym." *arXiv preprint arXiv:1606.01540* (2016).
https://gym.openai.com/

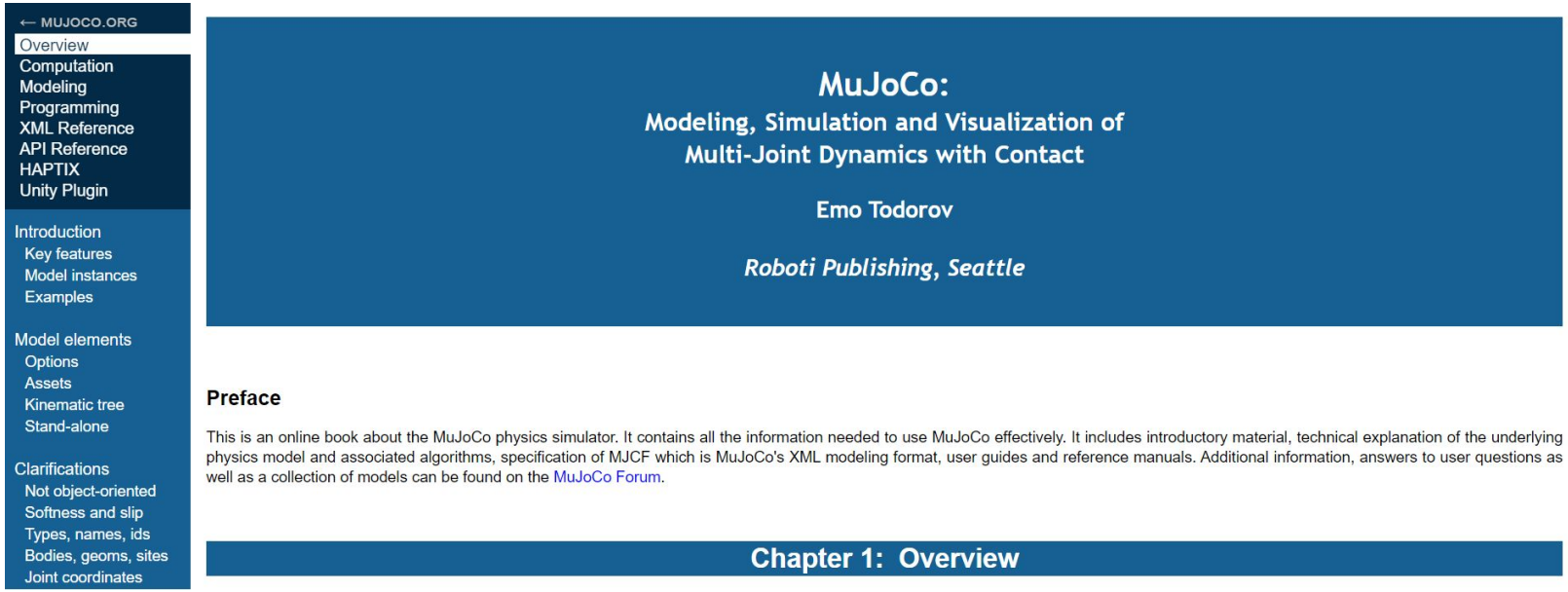
OpenAI Gym

high-level API for agent-environment interaction

Intuitive abstraction, easy interface

Imagine MuJoCo without Gym





CORE
Control + Optimization Research Lab

OpenAI Gym - Examples

Open gym_test.py

```
1   import gym
2
3
4   env = gym.make('Pendulum-v0')
5   state = env.reset()
6
7   for _ in range(200):
8       state, reward, done, info = env.step(env.action_space.sample())
9       env.render()
10
11  env.close()
```

Class Env(object):

action_space =
observation_space =

def step(self, action):

def reset(self):

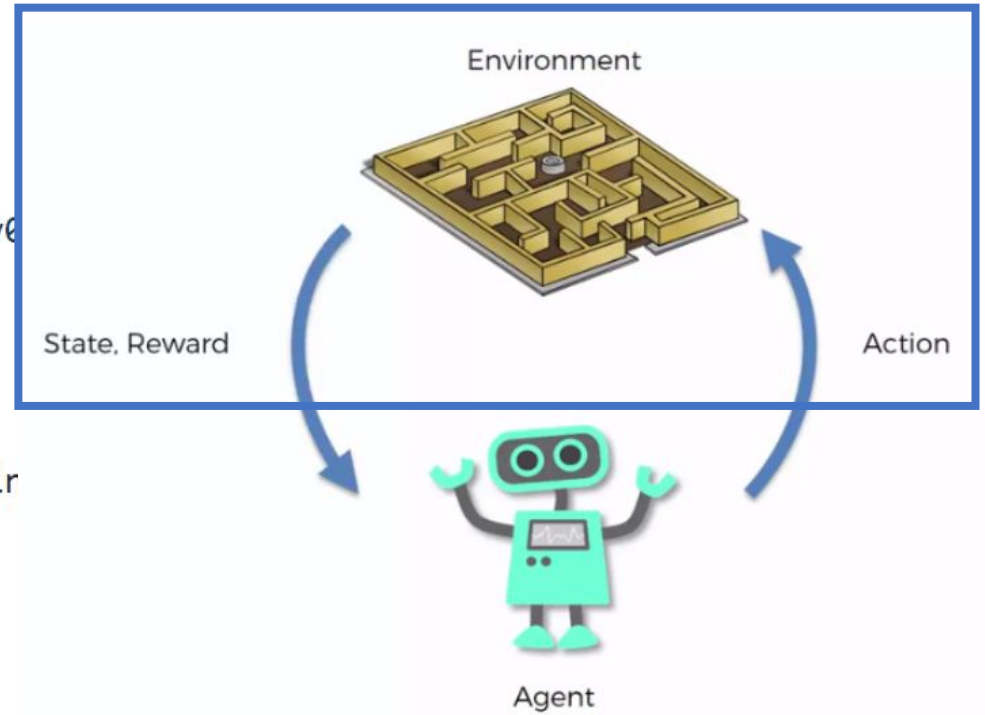
def render(self, mode='human):

def close(self):

OpenAI Gym - Examples

Open gym_test.py

```
1   import gym
2
3
4   env = gym.make('Pendulum-v0
5   state = env.reset()
6
7   for _ in range(200):
8       state, reward, done, ir
9       env.render()
10
11  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

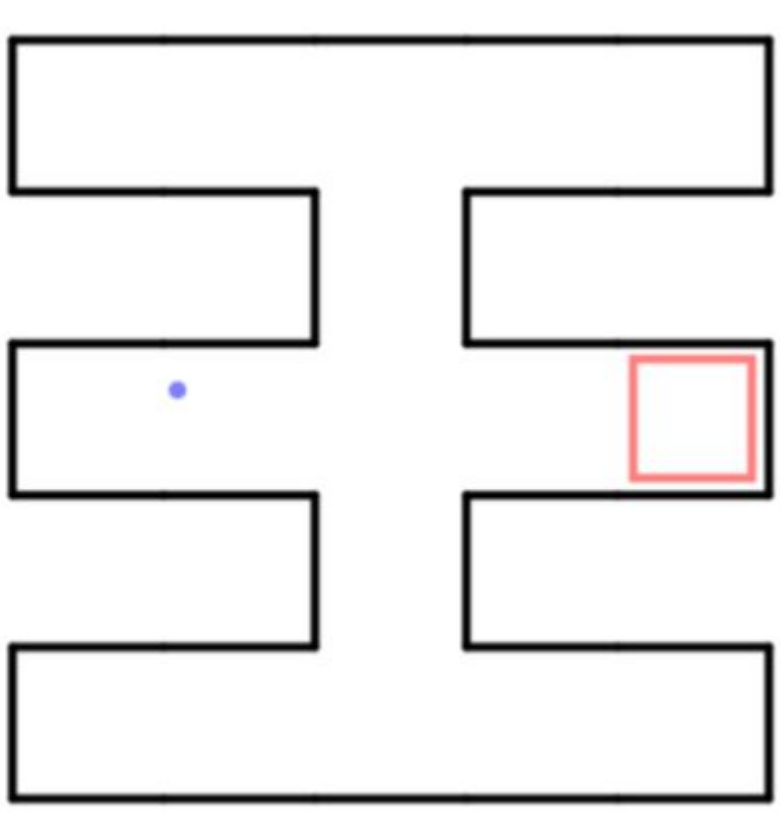
```
class PendulumEnv(gym.Env):
    metadata = {
        'render.modes': ['human', 'rgb_array'],
        'video.frames_per_second': 30
    }

    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** $\mathcal{A}$

OpenAI Gym - Examples

(a) Sample observation



(b) Layout of the 5×5 maze in (a)

state space?

observation space?

Thank you!