gym-softrobot

Release 0.1.4

Seung Hyun Kim, Chia-Hsien Shih

May 08, 2022

ENVIRONMENT

CHAPTER

ONE

ENVIRONMENT SETUP

Our environment contains set of controllable slender-bodies to achieve set of tasks. The theory and details of the physics simulation is documented in CosseratRods.org.

1.1 Interface

1.1.1 State

The state information may or may-not reflect the entirety of the system: partially observable. The state space is composed with the following parameters:

- **Position** (relative) and **director** at each element (spatial discretization in simulation)
  - Could be supplemented with **velocity/acceleration** and **angular velocity/acceleration**
- **6-modes of strains** (normal/binormal shear, normal/binormal curvature, stretch, twist)
- **Absolute position**
- **Target location** and its velocity (if applicable)
- Index of the agent (multi-agent case)
- Previous action

1.1.2 Action

- **Internal curvature** resembling tendon-driven actuation or muscle actuation.
  - The number of DoFs (degrees of freedoms) along the arm depends on the length of the arm, and may vary depending on the environment and its version. Actuation functions equals to the interpolation of provided DoFs.
- Internal torque/force for direct activation

1.1.3 Reward

The reward function is not yet finalized. Different version may contain different reward function.

The reward is defined as the composition of the following quantities:

- **Forward Reward**: typically used in locomotion case
  - Velocity of the body
  - Position difference between control-steps
- **Survive Reward**: given for stability purpose and to prevant wild policy
  - Nan panelty: unstable or unexpected behavior
  - Cross-over panelty: panelty for multiple arm crossing eachother (in 2D)
- **Control Panelty**: minimum-actuation solution
  - Square-average of the action
- **Energy**: minimum-energy solution
  - Total bending energy
  - Total shear energy
- **Time Limit**:
  - Small constant panelty at each steps
  - Large constant panelty for not achieving goal.
- **Miscellaneous**:
  - Target reaching/grabbing reward
  - Remaining distance to the target

CHAPTER

TWO

OCTOPUS

2.1 Single Arm Control

This environment is the testcase for simplest one-arm control.

- `OctoArmSingle-v0` [Alpha]

2.2 Multi Arm Control

- `OctoFlat-v0` [Alpha]

2.3 Advanced

The goal of this environment is to control 8 arms of the octopus attached to the body, and move towards the targeted location.

- `OctoReach-v0` [Working in Process]
- `OctoSwim-v0` [Working in Process]
- `OctoHunt-v0` [Working in Process]

CHAPTER

THREE

SOFT ARM

CHAPTER

FOUR

SOFT ARM WITH MUSCLE CONTROL

- SoftArmTrackingEnv-v0

CHAPTER

FIVE

MISCELLANEOUS ENV

5.1 Soft Pendulum [Transfer learning, Soft Arm Control]

- `SoftPendulum-v0`

5.2 Snake

- `ContinuumSnake-v0` [Alpha]

This environment is inspired from the Continuum Snake case in PyElastica. The goal is to control the snake to achieve fastest velocity. Unlike the original example, where the control is defined by the amplitude and phase-shift of the sinusoidal activation, our environment challenges the player to give an action every `dt` time-steps.

CHAPTER

SIX

AVAILABLE WRAPPERS

Here is the list of available wrappers that we provide. The purpose of the wrapper is to convert the environment to make it compatible with other external packages. These wrappers should be compatible with typical *OpenAI-gym* wrappers, such as *SubprocVecEnv* or *VecFrameStack*, although we highly recommend using our wrapper on the outer-most layer to be safe.

Note: Because of the nature of the wrapper design, we cannot guarantee 100% compatibility with all other available tools. Please make an GitHub issue if you find any bug in this feature.

6.1 Converter

6.1.1 Description

ConvertToPyMarlEnv	Convert environment to PyMarl multi-agent environments.

6.1.2 Built-in Wrappers

class gym_softrobot.wrapper.**ConvertToPyMarlEnv**(*ma_env*, **args*, ***kwargs*)

Convert environment to PyMarl multi-agent environments. Template from: oxwhirl/pymar/src/envs/multiagentenv.py

Available benchmark algorithms:

- oxwhirl/PyMarl
- uoe-agents/EPyMarl

The wrapper to convert gym_softrobot environment (that is MA-compatible) to 'multiagentenv' that is compatible to PyMARL. The purpose is to run benchmark study with standard CTDE algorithms, such as QMIX, COMA, etc. Here is the example snippet:

```
env = ConvertToPyMarlEnv(ma_env=gym.make("OctoCrawl-v0"))
```

__init__(*ma_env*, **args*, ***kwargs*)

Parameters

ma_env [gym.Env] Multi-agent compatible environment In gym-softrobot, environment with 'multiagent' tag in meta data must be true.

`step`(*actions*)

Returns reward, terminated, info

`get_obs`()

Returns all agent observations in a list

`get_obs_agent`(*agent_id*)

Returns observation for agent_id.

`get_obs_size`()

Returns the shape of the observation

The observation in regular gym_softrobot is given as (n * state_space)

`get_state`()

return global state.

Notes

Ideally, this function should not be used during decentralized execution.

`get_state_size`()

Returns the shape of the state

`get_avail_actions`()

Returns the available actions of all agents in a list.

Return type `List`

`get_avail_agent_actions`(*agent_id*)

Returns the available actions for agent_id

`get_total_actions`()

Returns the total number of actions an agent could ever take

`reset`(*, *seed=None*, *return_info=False*, *options=None*)

Returns initial observations and states

CHAPTER

SEVEN

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX

Symbols

C

G

M

R

S