PPO×Family

第二讲：解构复杂动作空间

主办 中国计算机学会 CCF

承办 上海人工智能实验室 Shanghai Artificial Intelligence Laboratory

协办 GAIAA 北京大学人工智能研究院 INSTITUTE FOR ARTIFICIAL INTELLIGENCE, PEKING UNIVERSITY 浙江大学上海高等研究院 SHANGHAI INSTITUTE FOR ADVANCED STUDY ZHEJIANG UNIVERSITY 商汤 sensetime

支持 知乎 机器之心 Synced 智海 新一代人工智能科教平台 paperweekly

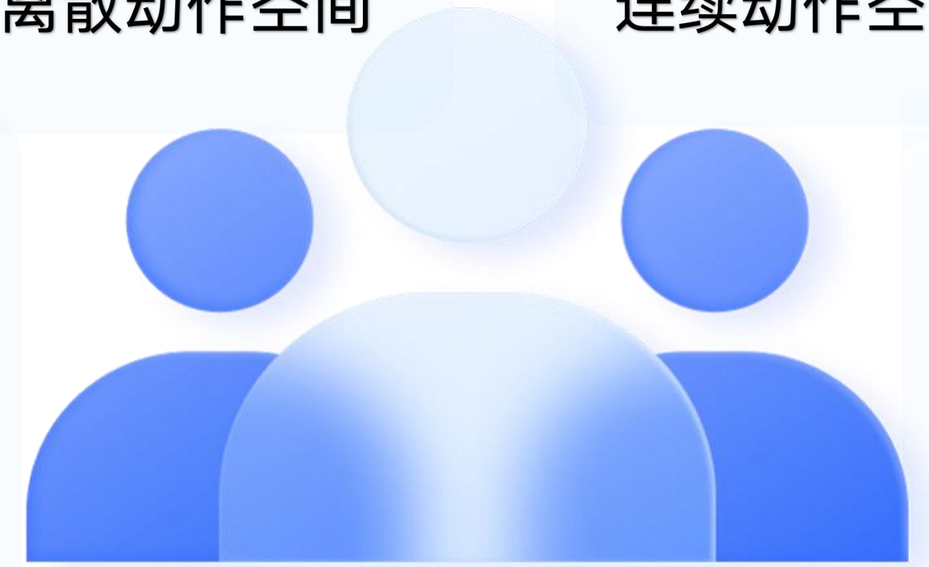
动作空间四重奏

01

离散动作空间

02

连续动作空间



03

多维离散动作空间

04

混合动作空间

Action Space Overview

动作空间概述

https://en.wikipedia.org/wiki/Keyboard_layout

动作空间的分类

特点：

- 不同的动作空间常需要不同的算法
- 标准的 RL 算法常需要对特定动作进行适配和定制化
- 同一个环境可以用不同的动作空间

Action Shaping

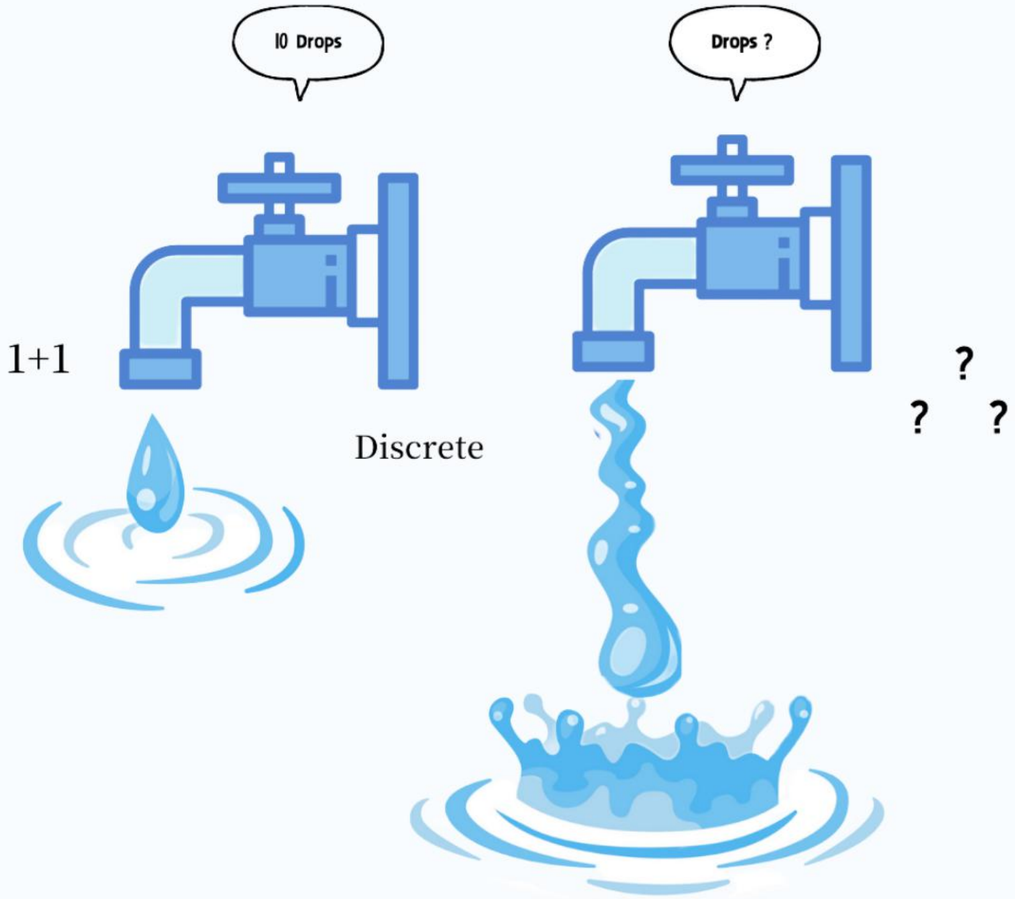
动作塑形

Environment Name	Original action space	Transform	Transformed action space	Performance
MineRL	Multi-discrete(2, 2, 2, 2, 2, 2, 2, 2, 7, 8, 5, 8, 3), Continuous(2)	DC RA CMD	Discrete(36)	
		DC RA CMD	Discrete(10)	1st place
		DC RA CMD	Discrete(216)	2nd place
		DC RA	Multi-discrete(2, 2, 3, 3, 7, 8, 5, 8, 3, 40)	3rd place
		DC RA	Multi-discrete(2, 2, 2, 5, 8, 3, 8, 7, 3, 3)	5th place
Unity Obstacle Tower Challenge	Multi-discrete(3, 3, 2, 3)	RA CMD	Discrete(12)	1st place
		RA	Discrete(6)	2nd place
VizDoom (Doom)	38 binary buttons, 5 continuous	RA CMD	Discrete(256)	1st place (Track 2)
		RA	Discrete(6)	1st place (Track 1)
Atari	Discrete(18)	RA	Discrete(4 - 18)	
StarCraft II	Multi-discrete	DC RA	Multi-discrete	
Dota 2	Multi-discrete	DC RA	Multi-discrete	
GTA V (car driving only)	Multi-discrete	RA CMD	Discrete(3)	
Torcs	Multi-discrete	RA CMD	Discrete(3)	
DMLab (Quake 3)	Multi-discrete(3, 3, 2, 2, 2), Continuous(2)	RA CMD	Discrete(9)	
Honor of Kings (MOBA)	Multi-Discrete, Continuous	RA CMD	Multi-discrete, Continous	
Little Fighter 2 (lf2gym)	Multi-Discrete(2,2,2,2,2,2,2)	CMD	Discrete(8)	

Action Space Shaping in Deep Reinforcement Learning: https://ieeexplore.ieee.org/document/9231687

Discrete

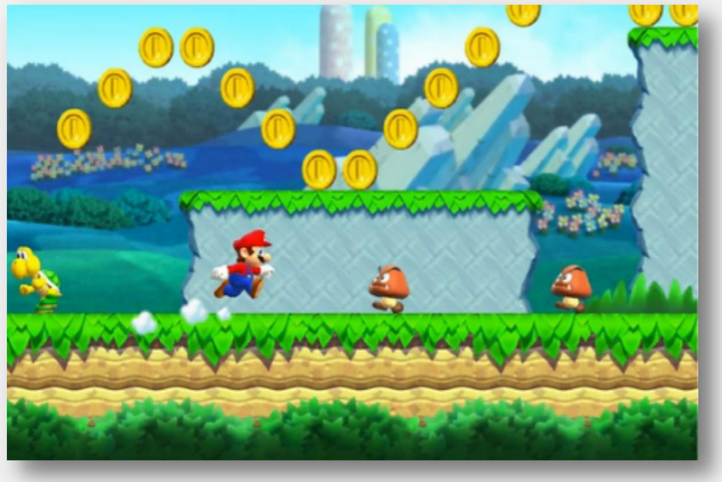
离散动作空间



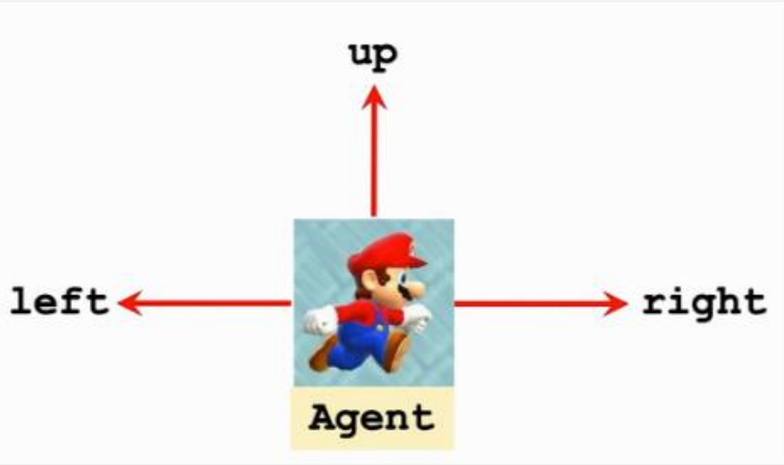
https://www.cuemath.com/data/discrete-data/

Discrete

理论：PPO+离散动作 定义



图片观察状态s



离散动作a

示例：随机性策略

$\pi(a_t|s_t)$

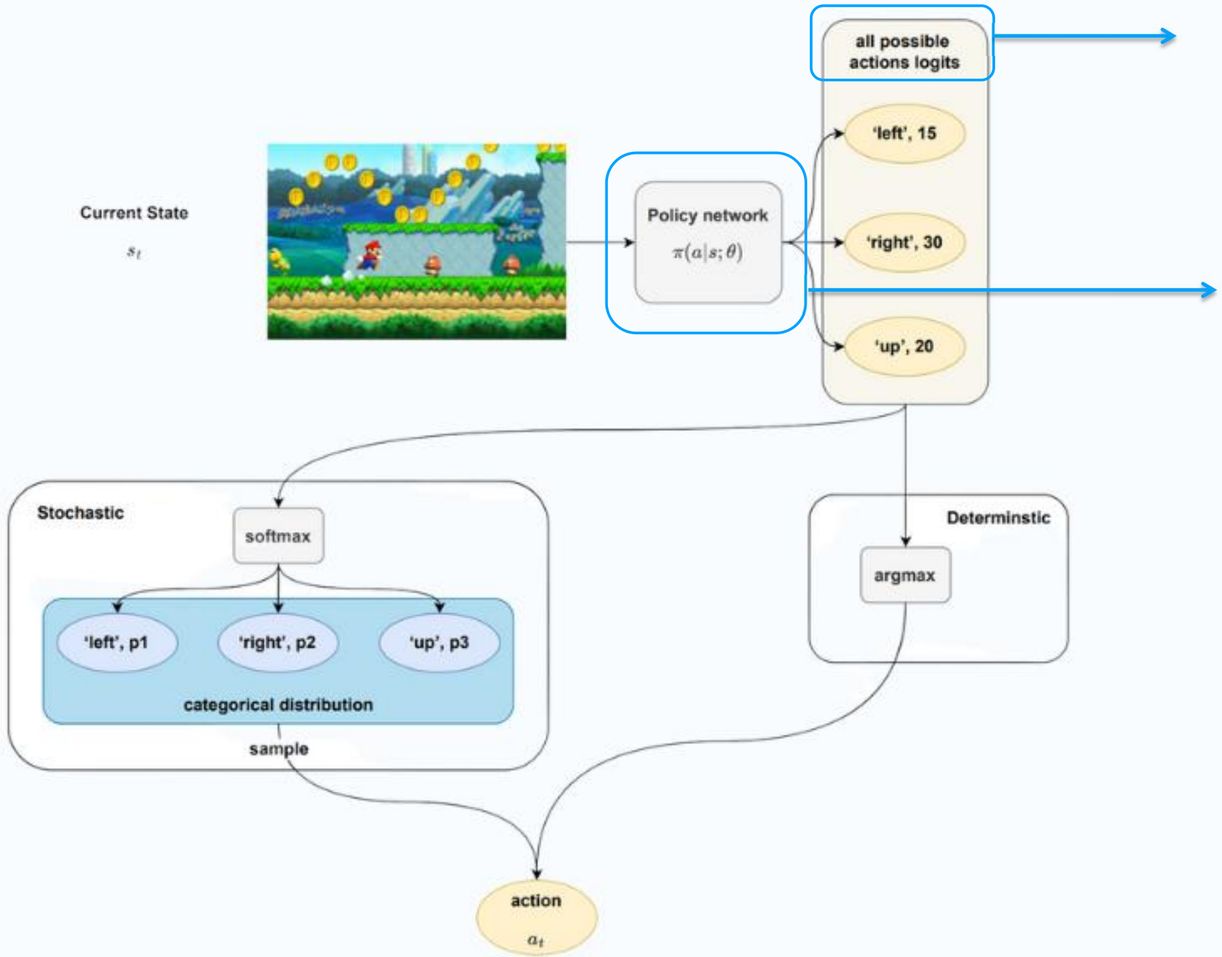
$\pi(\text{left}|s_t) = 0.2$

$\pi(\text{right}|s_t) = 0.7$

$\pi(\text{up}|s_t) = 0.1$

Discrete

理论：PPO+离散动作 计算图

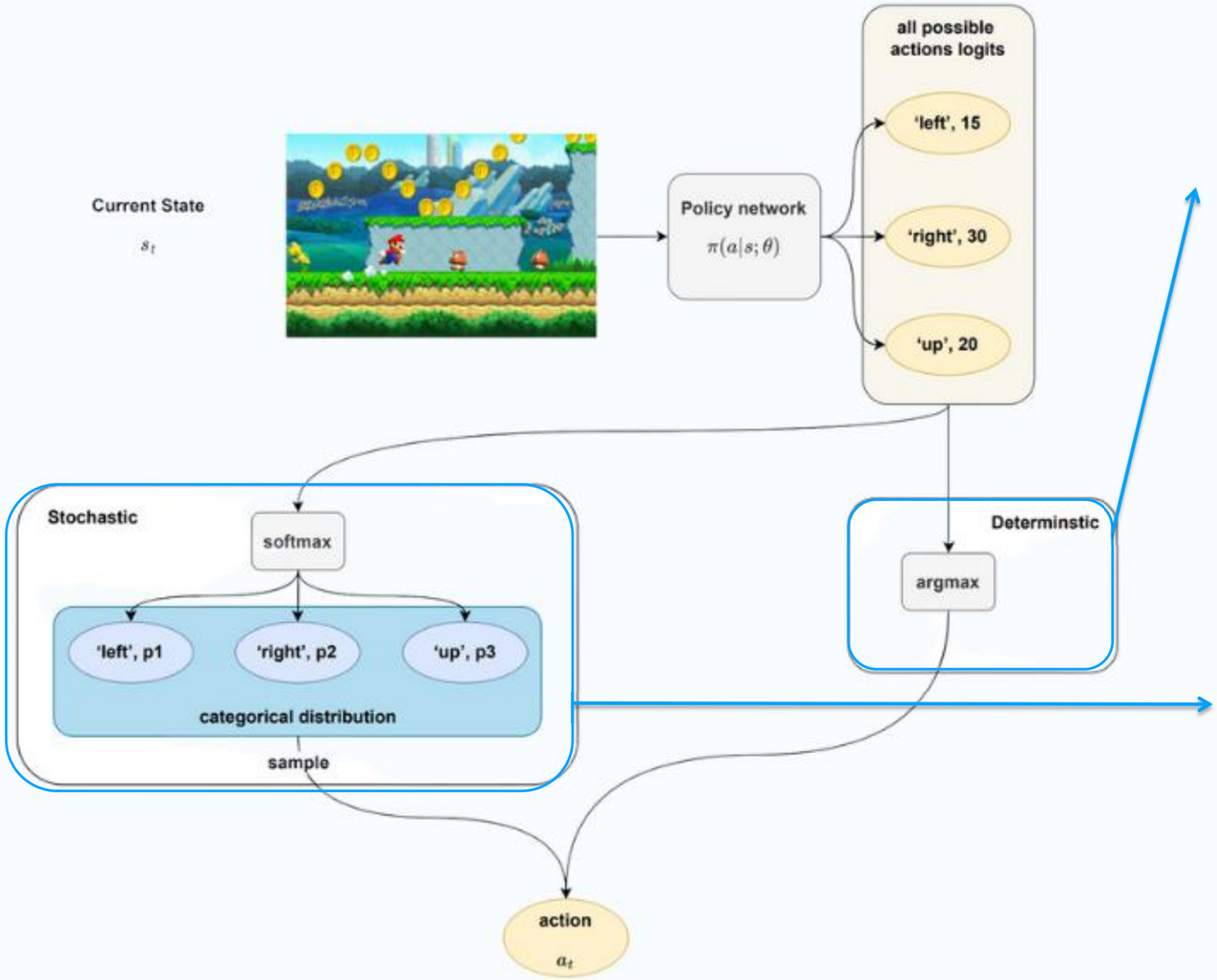


logit：策略神经网络的原始输出，比如最后一层全连接层的输出，不加激活函数

用神经网络来实现参数化的随机性策略，策略网络输出 logit，构建选择动作的 Categorical Distribution

Discrete

理论：PPO+离散动作采样



确定性决策：直接贪心选择 logit 最大的那一维动作（确定性评测时使用）

随机性决策：根据概率构建玻尔兹曼分布进行采样（随机性评测+收集数据使用）

Discrete

代码：PPO如何实现离散动作

PyTorch demo of PPO algorithm in discrete action space.

Stars 566 | bilibili video course | Follow @opendilab 39

View code on GitHub

Overview
The computation graph of discrete action policy network used in PPO.

```python
def forward(self, x: torch.Tensor) -> torch.Tensor:
```

Transform original state into embedding vector (i.e. (B, *) -> (B, N)).

```python
        x = self.encoder(x)
```

Calculate logit for each possible discrete action dimension (i.e. (B, action_shape)).

$$\pi_\theta(a|s)$$

```python
        logit = self.head(x)
        return logit


```

Overview
The function of sampling discrete action, input shape = (B, action_shape), output shape = (B,).
In this example, batch_shape = (B,), event_shape = (), sample_shape = ().

```python
def sample_action(logit: torch.Tensor) -> torch.Tensor:
```

Transform logit (raw output of policy network, e.g. last fully connected layer) into probability.

```python
    prob = torch.softmax(logit, dim=-1)
```

Construct categorical distribution.
Related Link

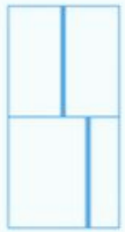
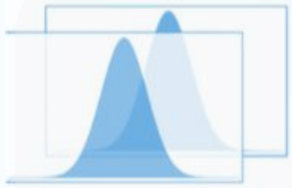
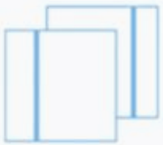
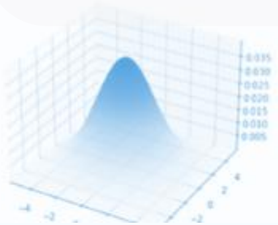
```python
    dist = torch.distributions.Categorical(probs=prob)
```

Sample one discrete action per sample and return it.

```python
    return dist.sample()

```

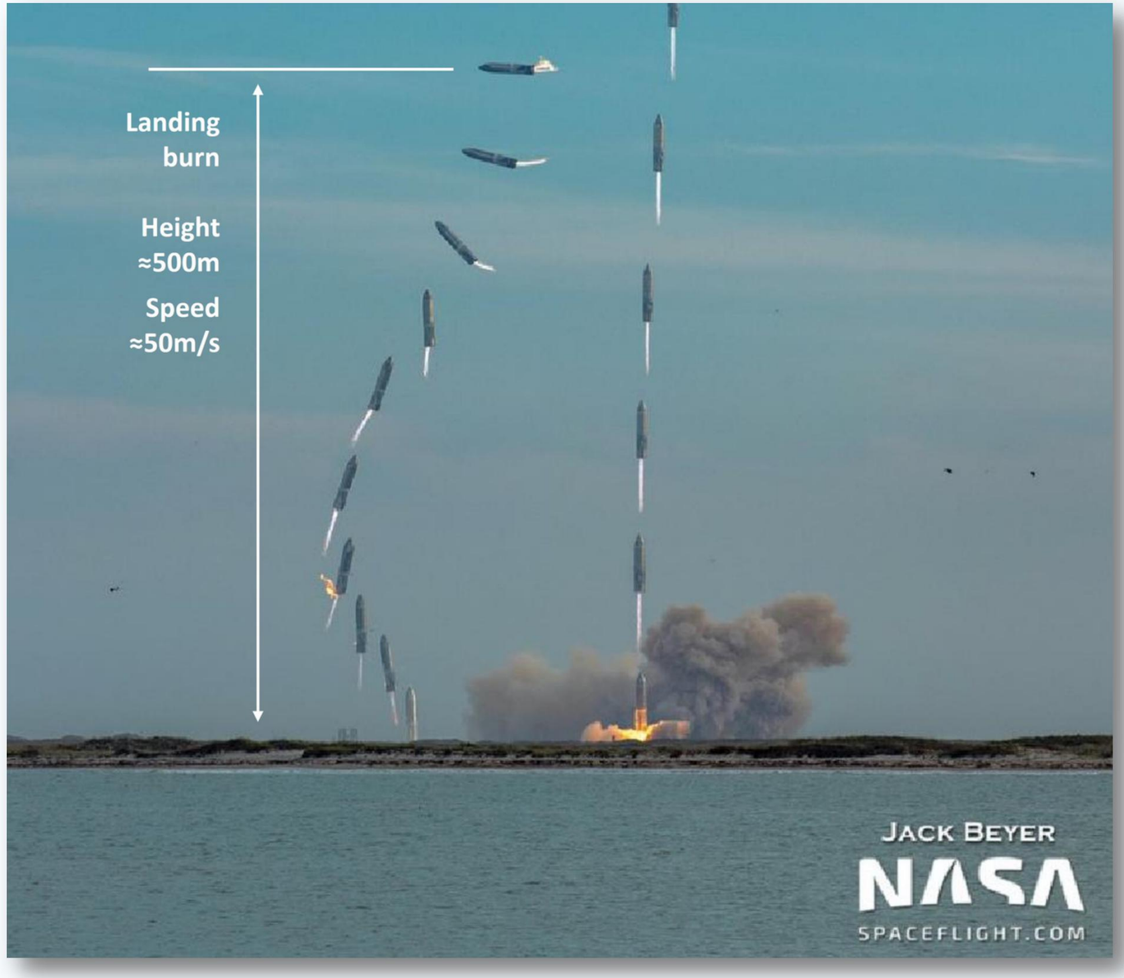
完整示例：https://opendilab.github.io/PPOxFamily/

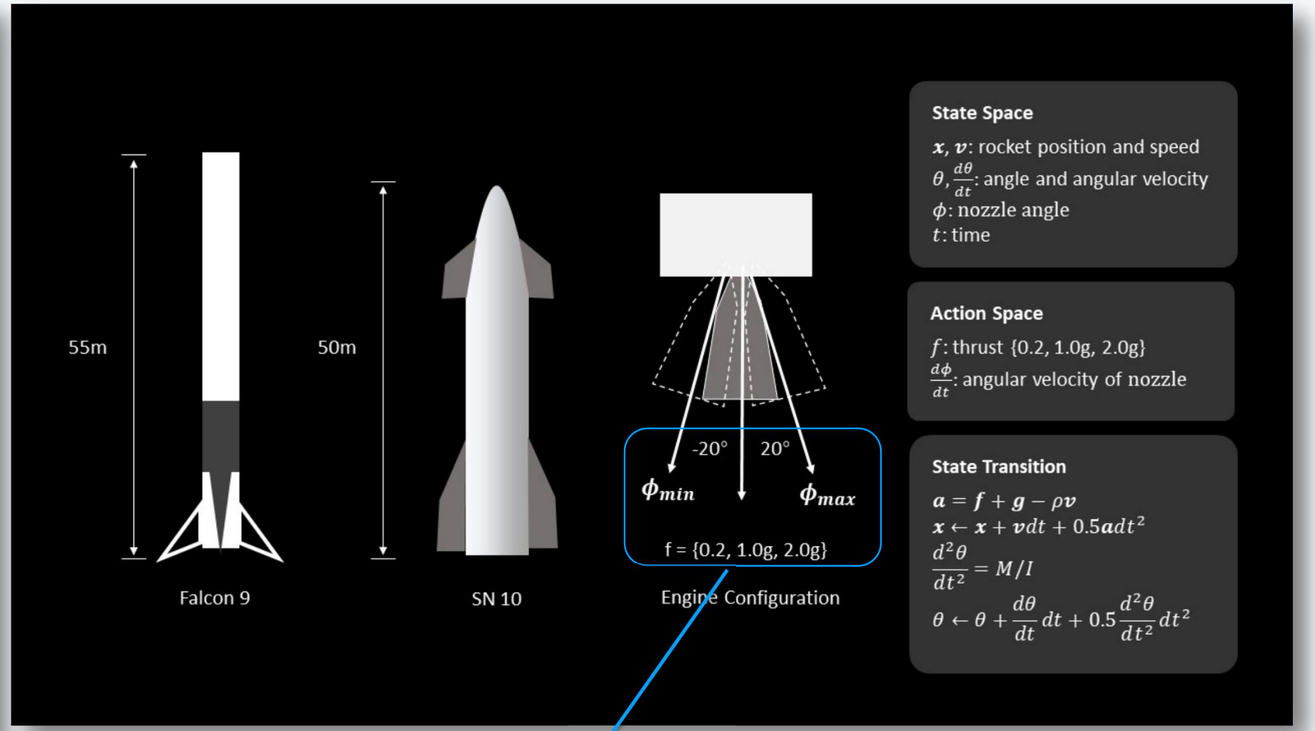
代码：PPO如何实现动作采样

分布	抽样	Event Shape	Batch Shape	Sample Shape	解释	
		()	()	(2,)	对1个1元正态分布进行2次抽样	独立同分布地多次采样
		()	(2,)	()	对2个1元正态分布进行1次抽样	从多个独立分布中采样
		(2,)	()	()	对1个2元正态分布进行1次抽样	对一个多维事件采样

TensorFlow Distributions: https://arxiv.org/abs/1711.10604

实践：PPO+火箭回收

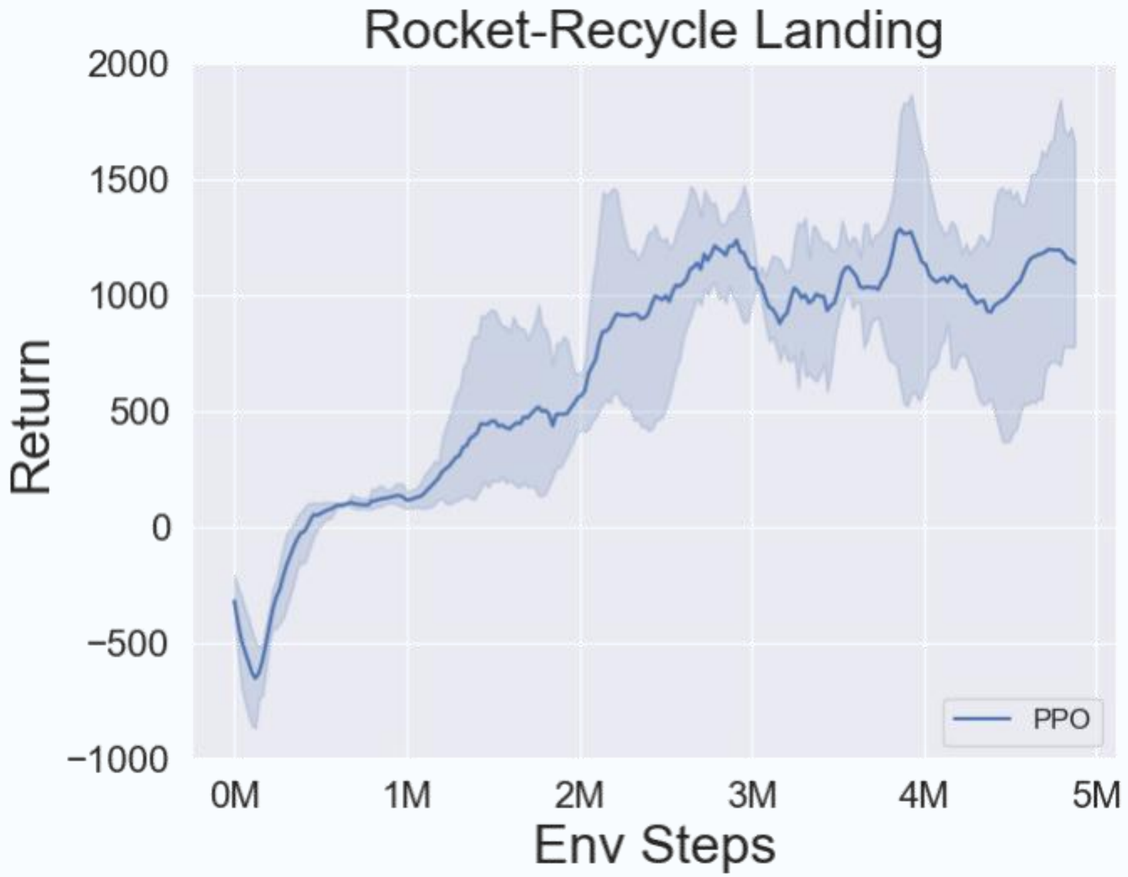




离散化为3个角度 x 3种推力共9维离散动作

Rocket Recycle Landing: https://github.com/jiupinjia/rocket-recycling

实践：PPO+火箭回收



奖励空间：与着陆目标的距离 + 姿态控制

1. 乱喷气直接坠落（0–100k）
2. 学会半空悬停（200k–500k）
3. 学习控制落点大趋势（600k–1M）
4. 学会落地减速（1M–2M）
5. 探索较短的轨迹，反复横跳（2M–3M）
6. 学习落地保持稳定（3M–5M）

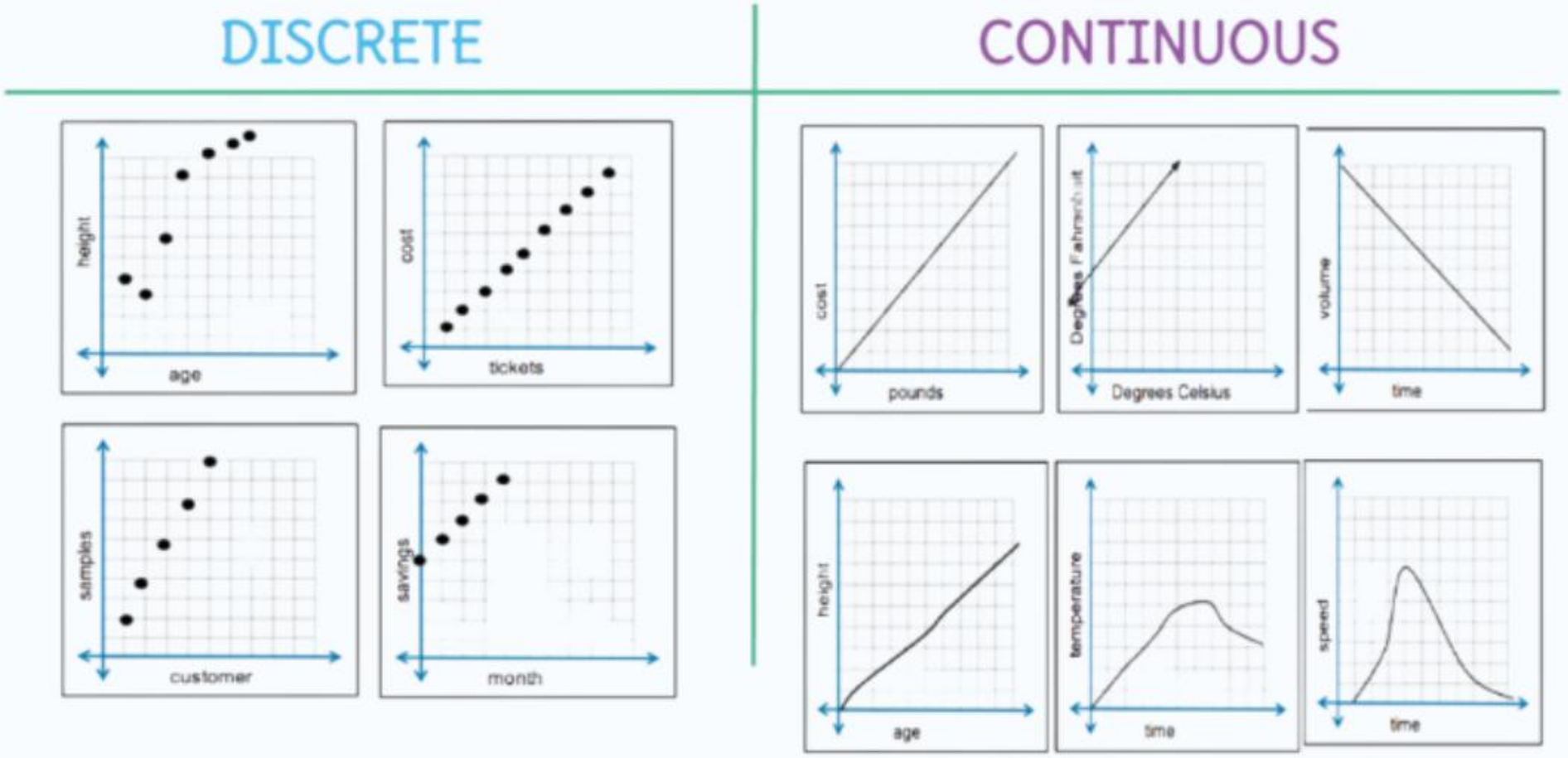
实验细节：https://github.com/opendilab/PPOxFamily/issues/4

实践：PPO+火箭回收

视频样例：https://github.com/opendilab/PPOxFamily/issues/4

Continuous

连续动作空间



https://learnzillion.com

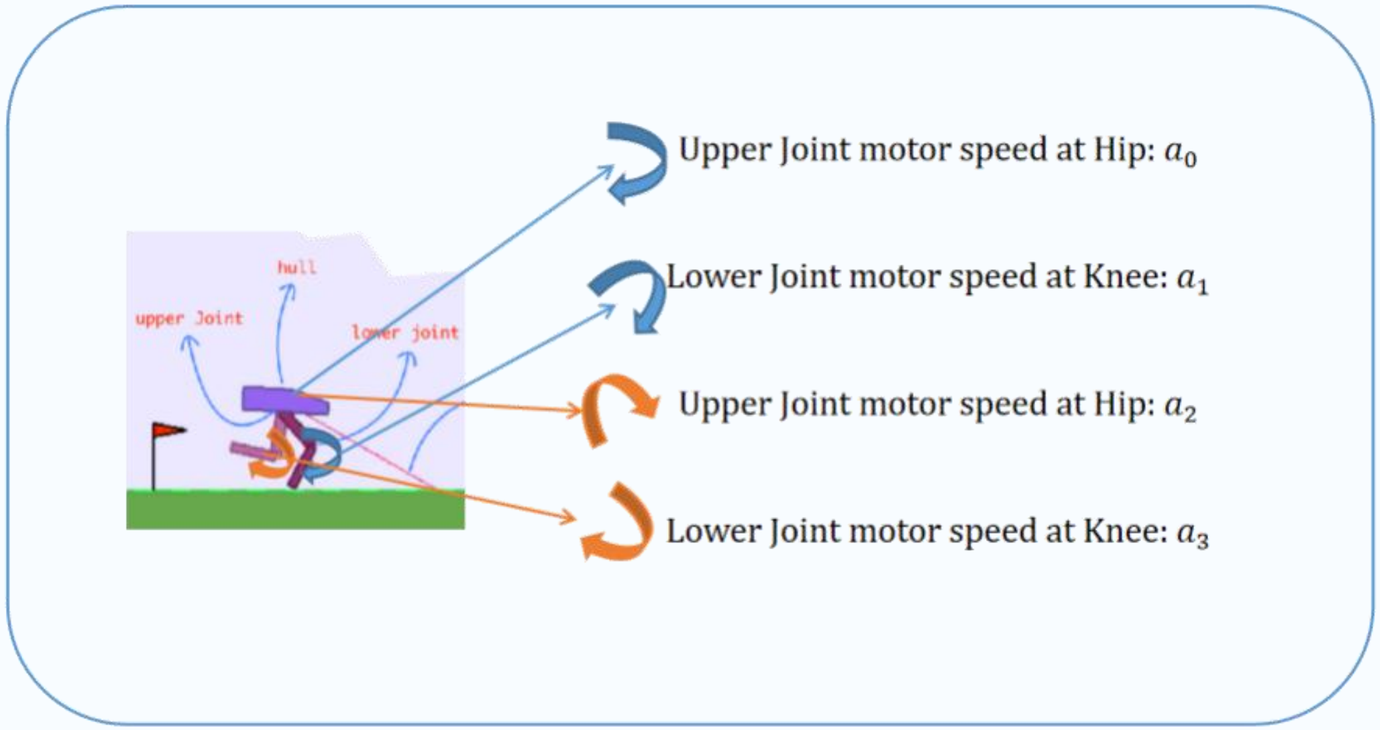
离散

Discrete

连续

Continuous

理论：PPO+连续动作 定义

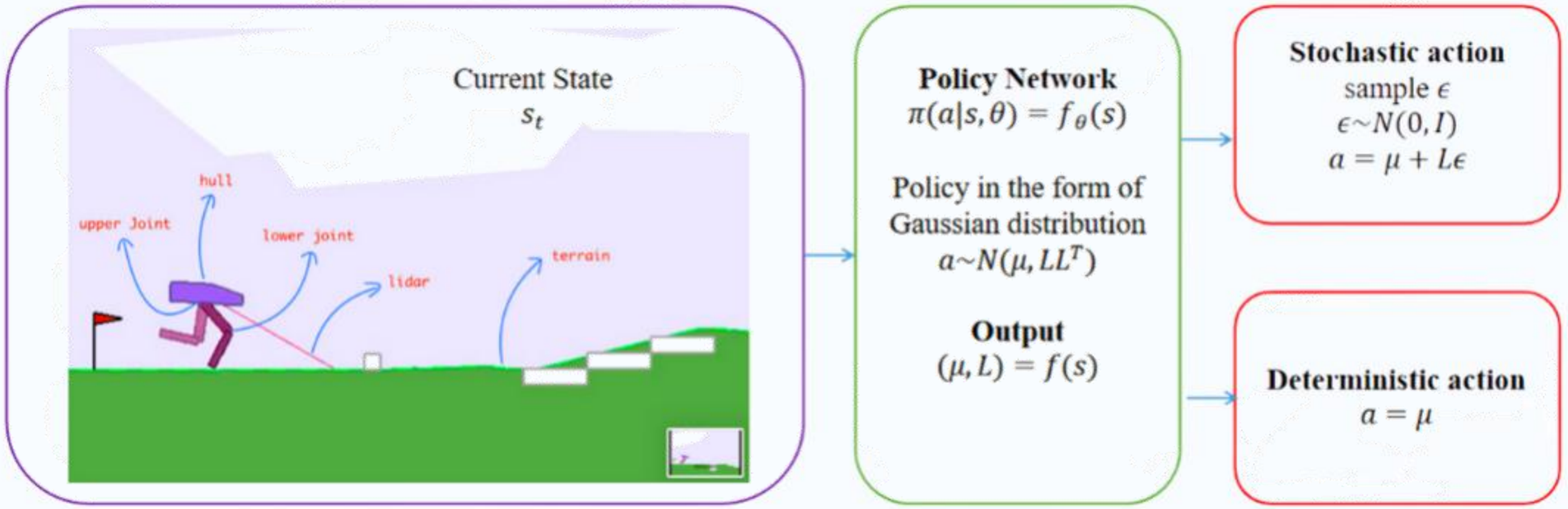


特点：

- 连续的，稠密的，不可数个动作元素组成的连续区间（类似回归问题）
- 可以根据具体取值来衡量数值大小和远近关系
- 可以直接回归，也可重参数化

Bipedal Walker: https://di-engine-docs.readthedocs.io/zh_CN/latest/13_envs/bipedalwalker_zh.html

理论：PPO+连续动作 计算图

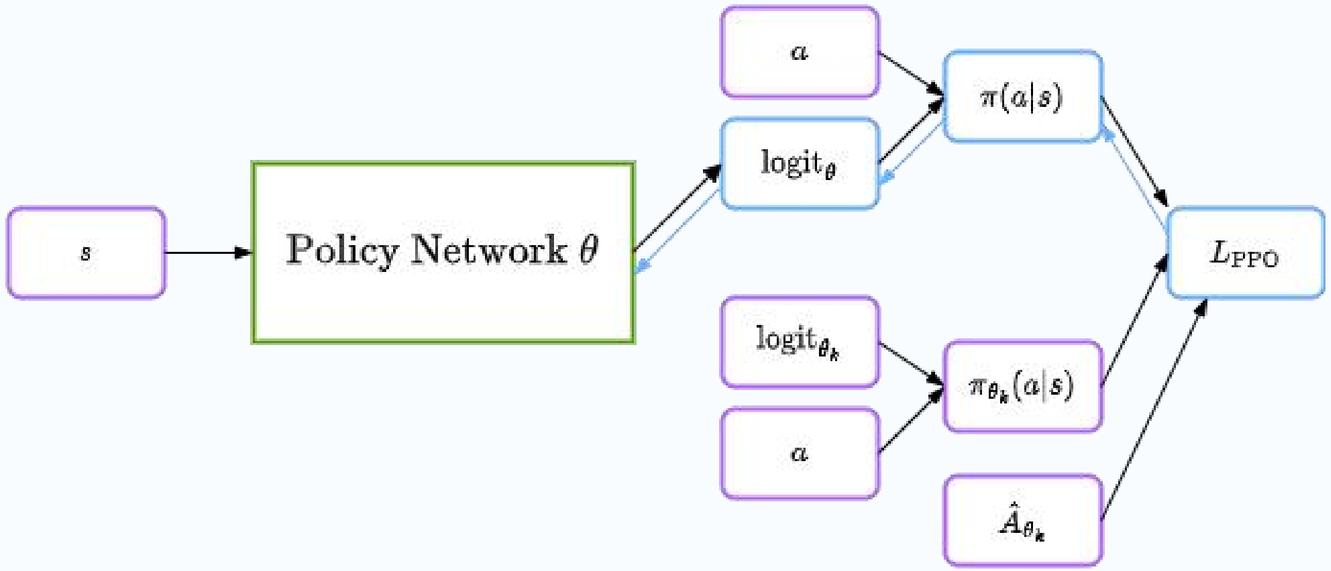


运动状态（24维） -> 策略神经网络 -> 分布参数 -> 连续动作（4维）

理论：PPO+连续动作 梯度流

Data: $< s, a, \hat{A}_{\theta_k}, \text{logit}_{\theta_k}, \cdots >$

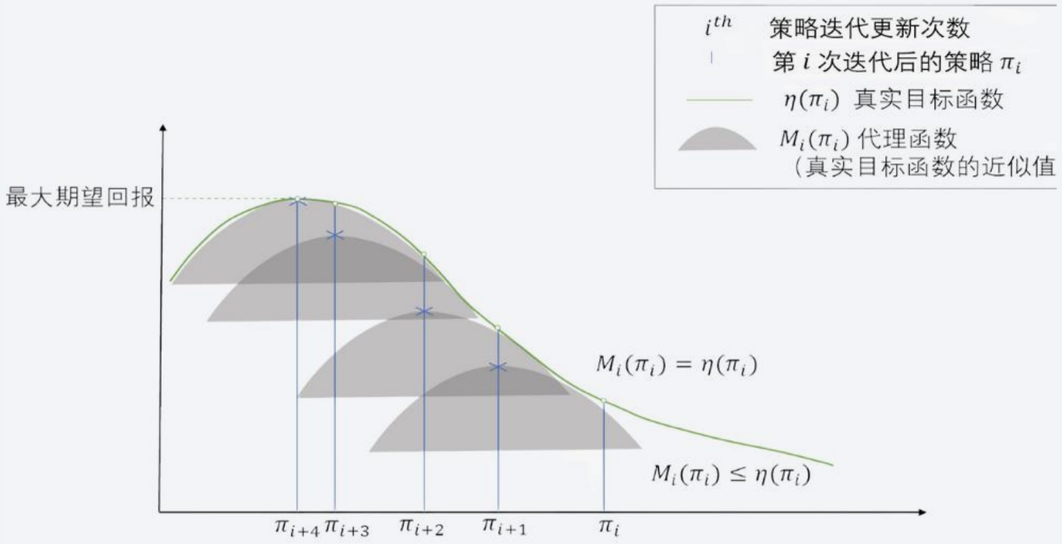
Loss: $\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)} \hat{A}_{\theta_k}$



只有 IS 项的分子回传梯度，IS 项的分母和优势函数都不参与 BP

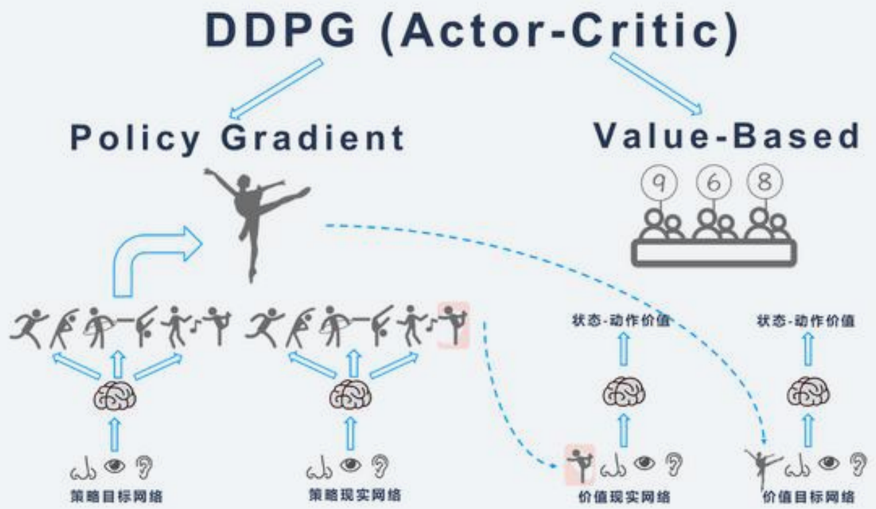
理论：PPO+连续动作 设计理念

PPO



PPO 围绕策略提升定理展开，而其价值函数主要起辅助评价作用

DDPG



DDPG 围绕优化动作价值函数展开，策略函数由对价值函数求极值的方式引导生成

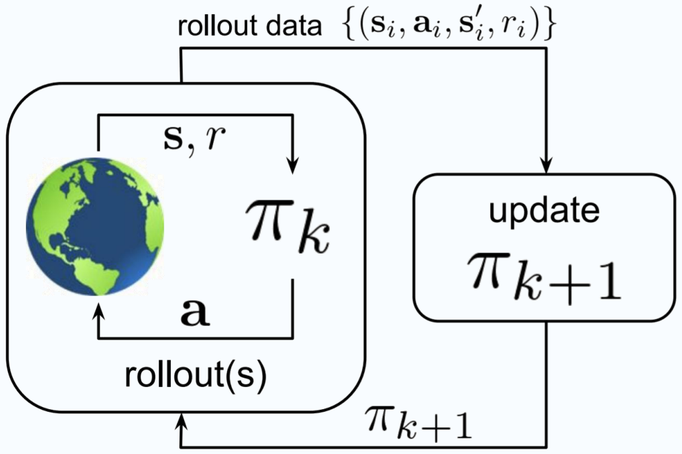
DDPG: https://arxiv.org/pdf/1509.02971.pdf

理论：PPO+连续动作 数据属性

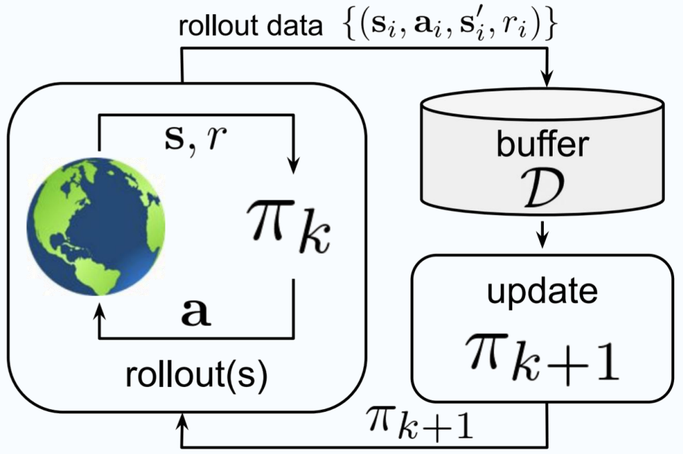
PPO

DDPG

(a) online reinforcement learning



(b) off-policy reinforcement learning



- PPO 为使其策略提升稳定有效，需要保持两个策略之间足够接近
- DDPG 取决于动作值函数的建模质量，更需要提高数据多样性

Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems: https://arxiv.org/pdf/2005.01643.pdf
关于 IS 的对比分析：https://github.com/opendilab/PPOxFamily/blob/main/chapter2_action/chapter2_supp_ppovsddpg.pdf

代码：PPO如何实现连续动作

PyTorch demo of PPO algorithm in continuous action space.

Stars 565 | bilibili video course | Follow @opendilab 39

View code on GitHub

Overview

The definition of continuous action policy network used in PPO, which is mainly composed of three parts: encoder, mu and log_sigma.

```python
from typing import Dict
import torch
import torch.nn as nn
from torch.distributions import Normal, Independent


class ContinuousPolicyNetwork(nn.Module):
    def __init__(self, obs_shape: int, action_shape: int) -> None:
```

PyTorch necessary requirements for extending `nn.Module` .

```python
        super(ContinuousPolicyNetwork, self).__init__()
```

Define encoder module, which maps raw state into embedding vector.
It could be different for various state, such as Convolution Neural Network for image state.
Here we use two-layer MLP for vector state.

```python
        self.encoder = nn.Sequential(
            nn.Linear(obs_shape, 16),
            nn.ReLU(),
            nn.Linear(16, 32),
            nn.ReLU(),
        )
```

Define mu module, which is a FC and outputs the argument mu for gaussian distribution.

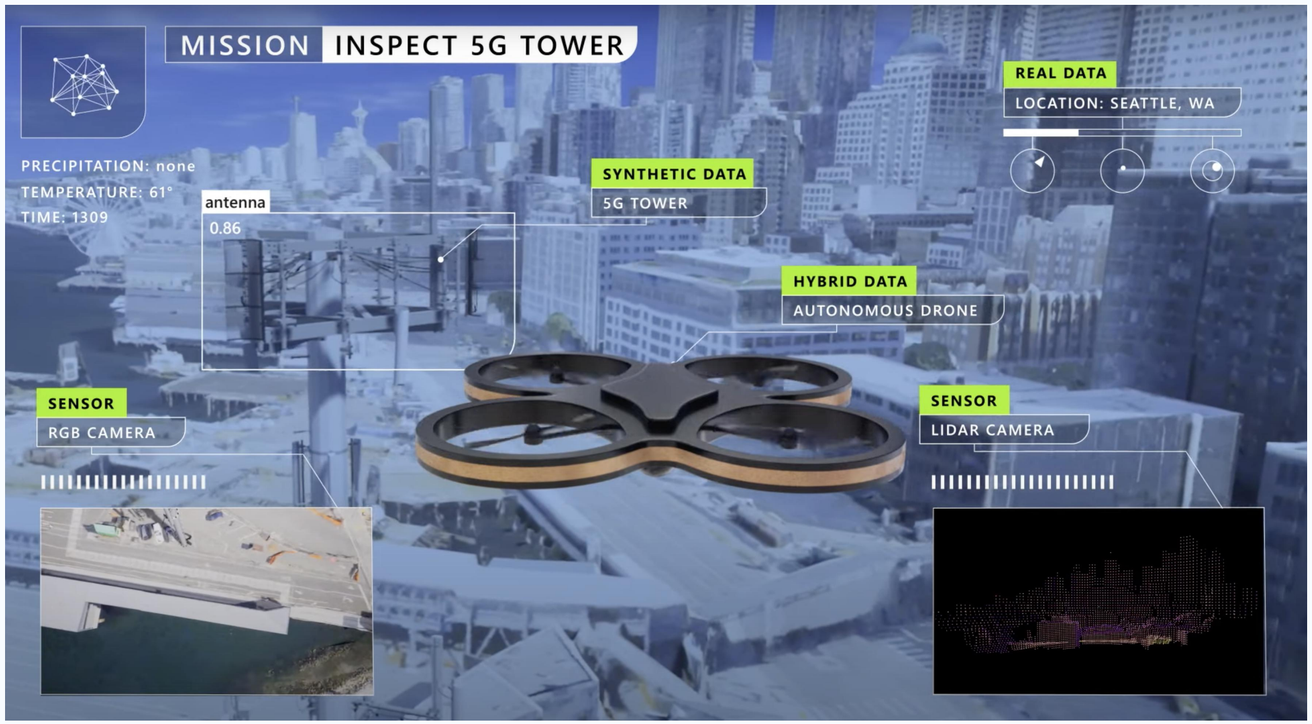
```python
        self.mu = nn.Linear(32, action_shape)
```

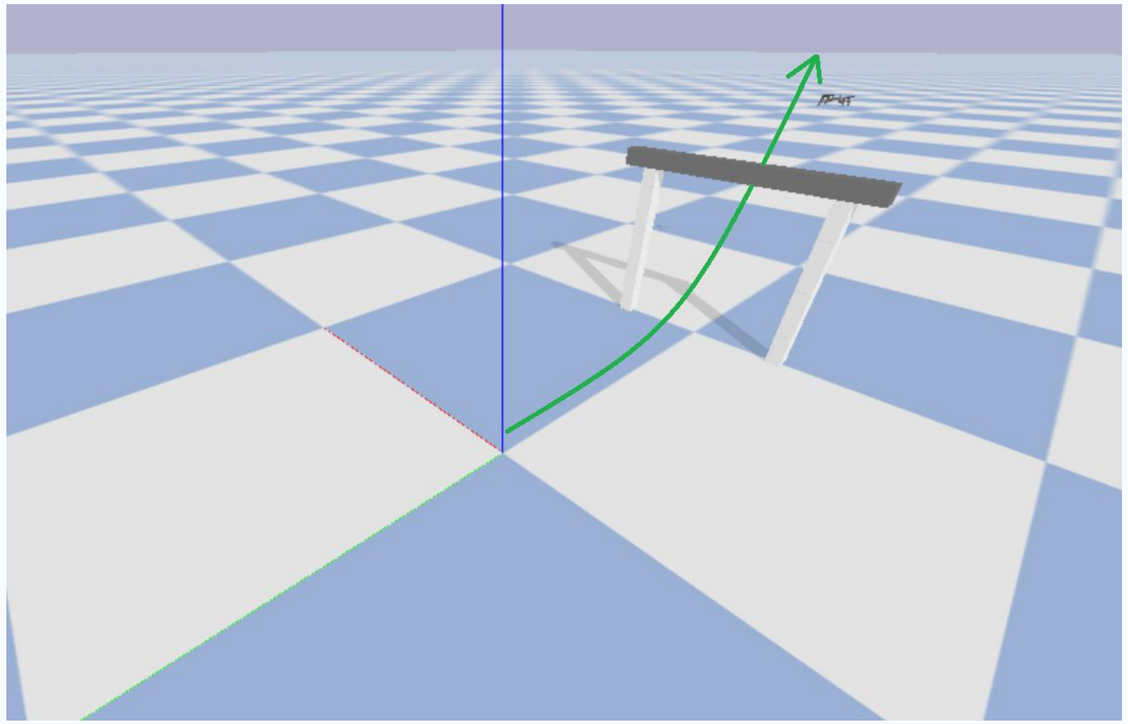
Define log_sigma module, which is a learnable parameter but independent to state.

```python
        self.log_sigma = nn.Parameter(torch.zeros(1, action_shape))

```

完整示例：https://opendilab.github.io/PPOxFamily/

实践：PPO+无人机姿态控制

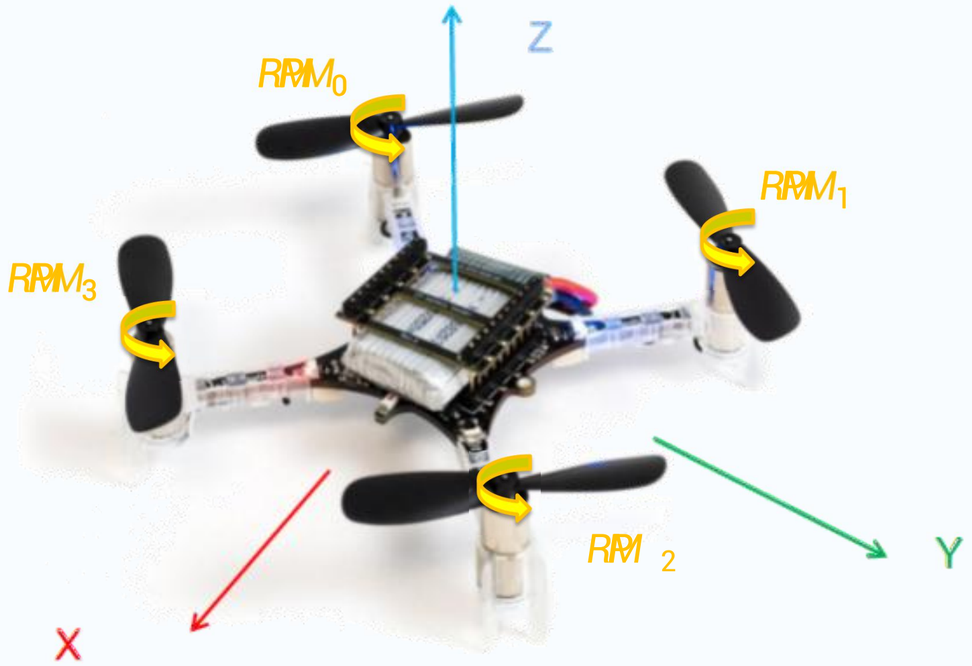




任务：四旋翼无人机从原点起飞，并持续飞行成功穿过一座门框

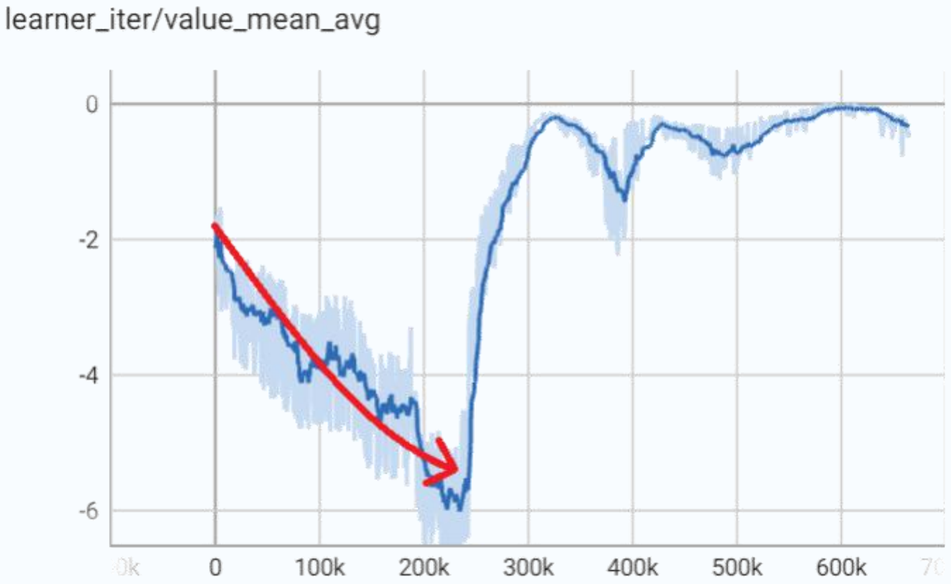
AirSim: https://www.youtube.com/watch?v=mhEw-fecjQY
gym-bullet-drones: https://github.com/utiasDSL/gym-pybullet-drones

实践：PPO+无人机姿态控制

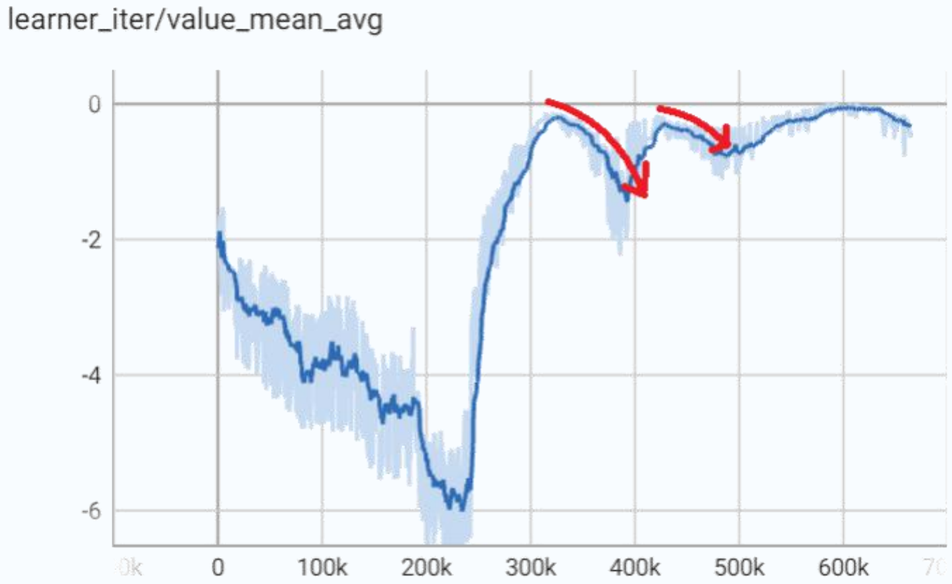


- **原始动作空间：** 4维连续动作，直接控制四个旋翼的转速
- **PID 辅助模式动作空间：** 3维连续动作，期望运动位置坐标
- **奖励空间：** 与期望轨迹点之间的距离

实践：PPO+无人机姿态控制



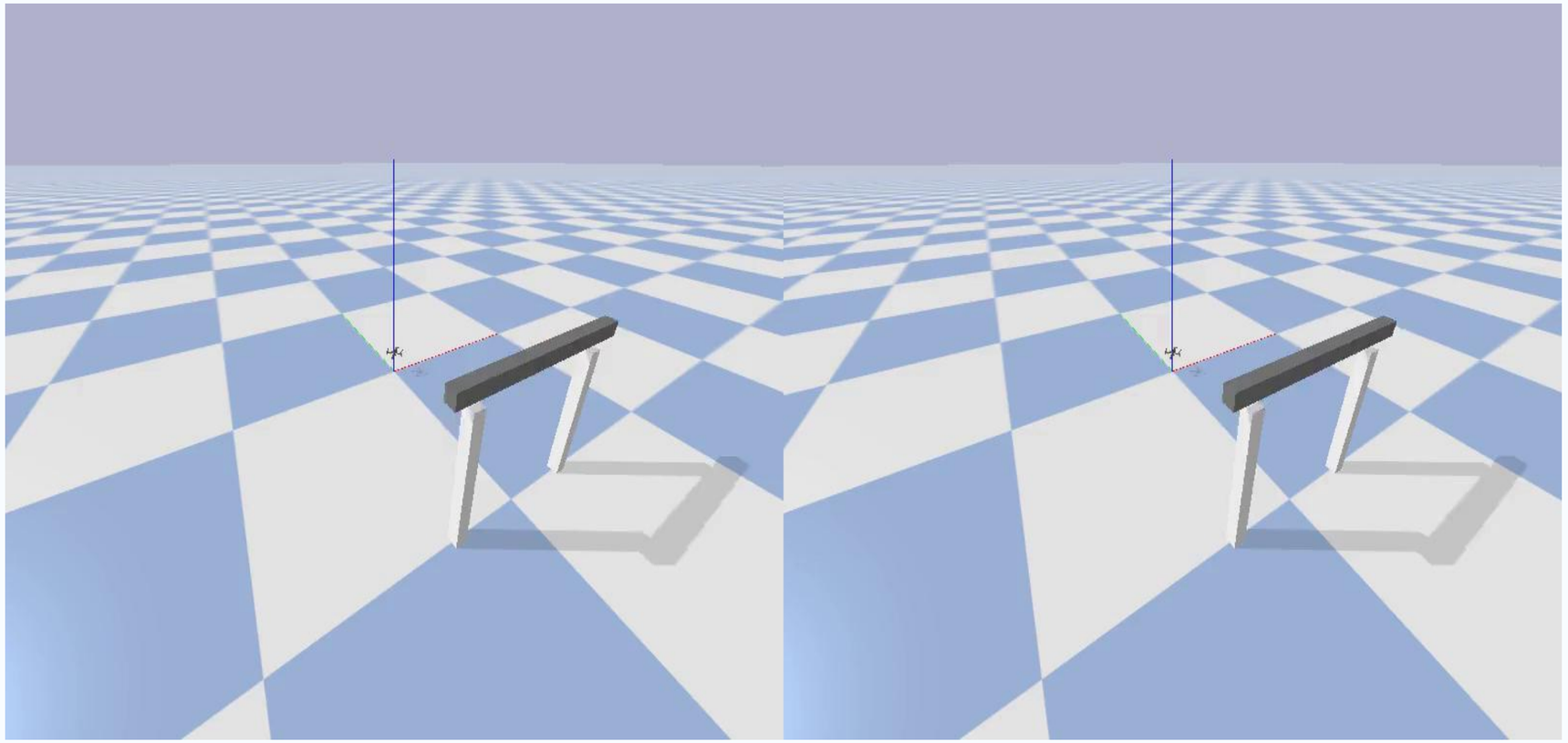
价值函数拟合阶段



策略提升阶段

实验细节：https://github.com/opendilab/PPOxFamily/issues/4

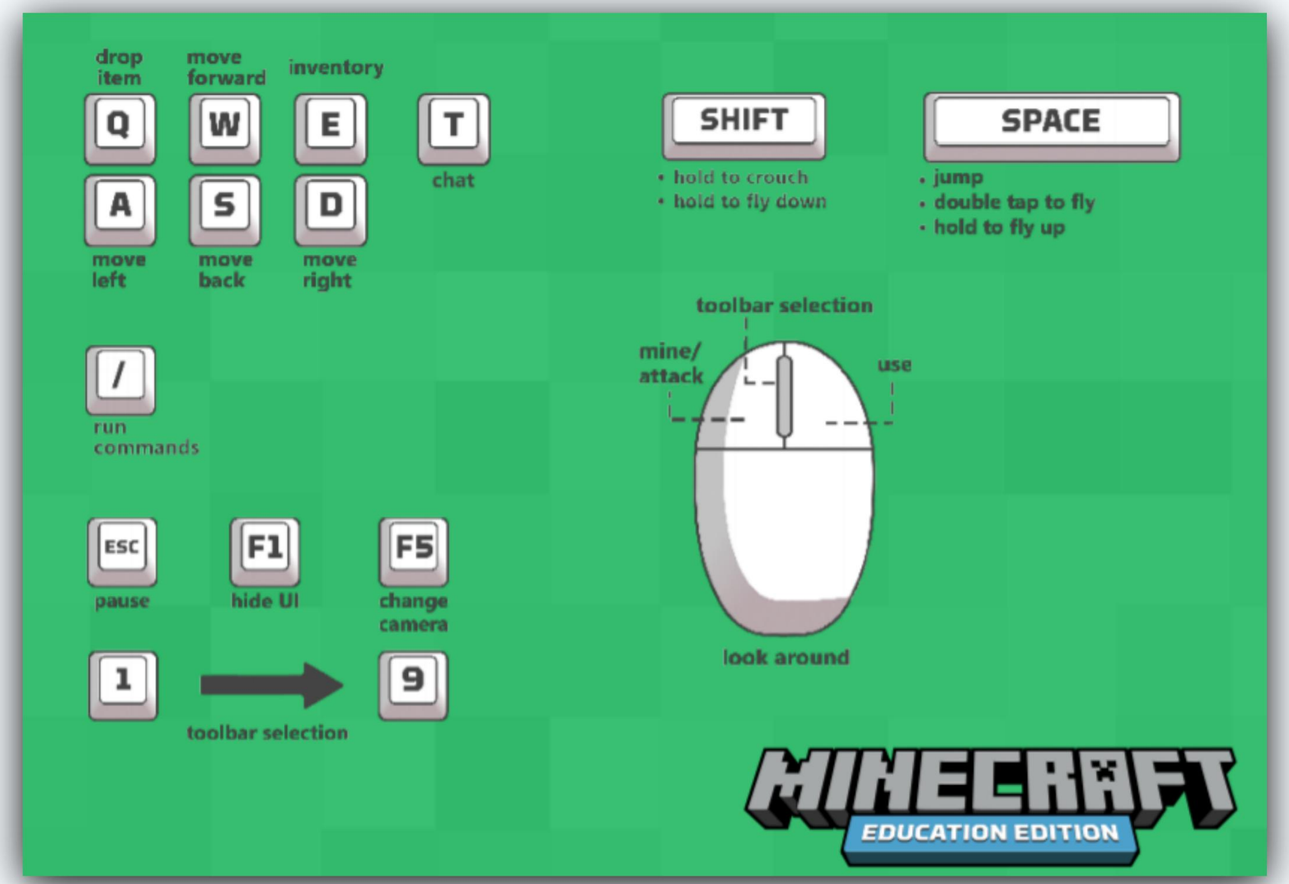
实践：PPO+无人机姿态控制



视频样例：https://github.com/opendilab/PPOxFamily/issues/4

Multi-Discrete

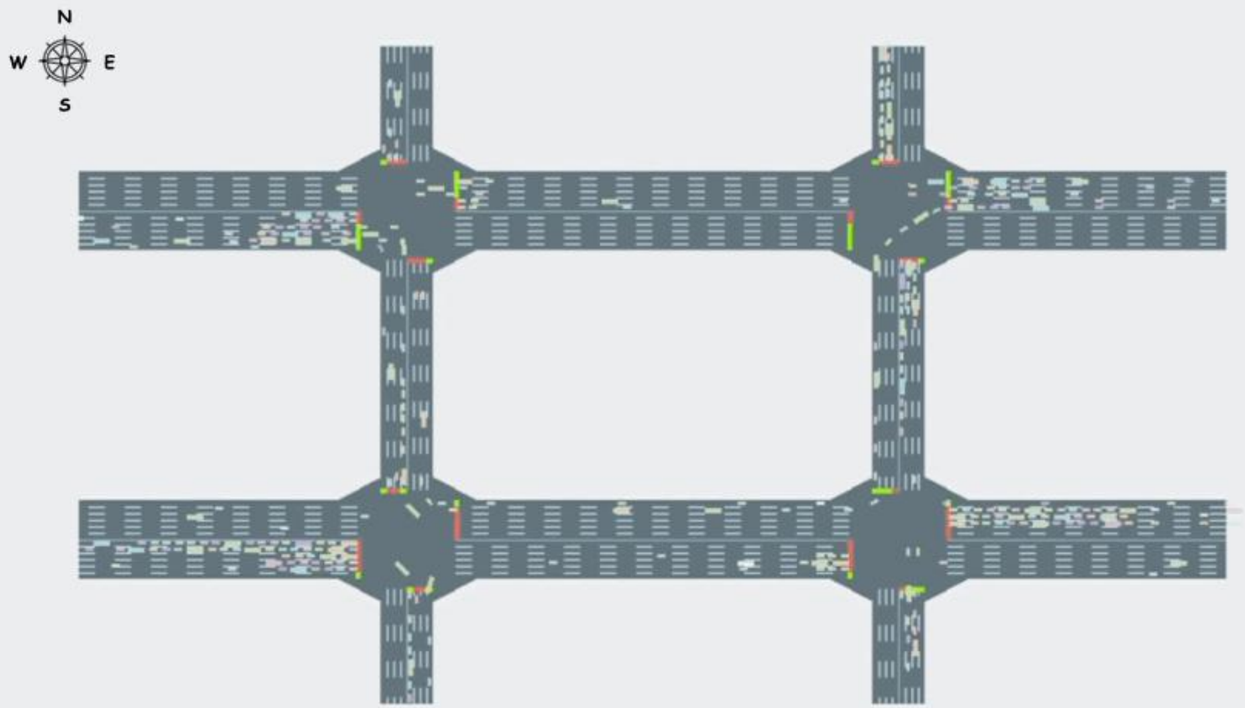
多维离散动作空间



https://learn71.ca/minecraft-challenge-accessibility/

Multi-Discrete

理论：PPO+多维离散定义

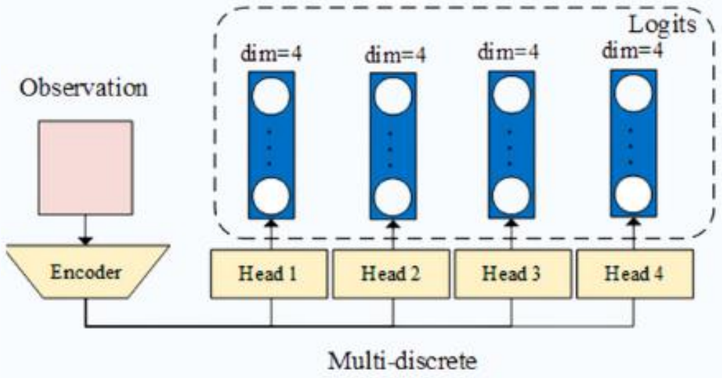


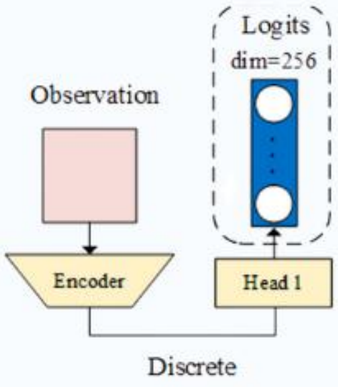
特点：

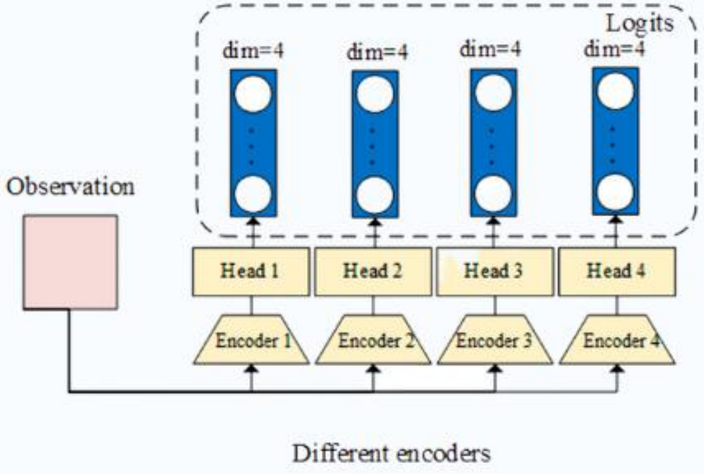
- 多个不同离散空间的组合（比如键盘，多路口信控）
- 离散动作的个数可能非常大
- 各个离散动作之间存在比较弱的联系

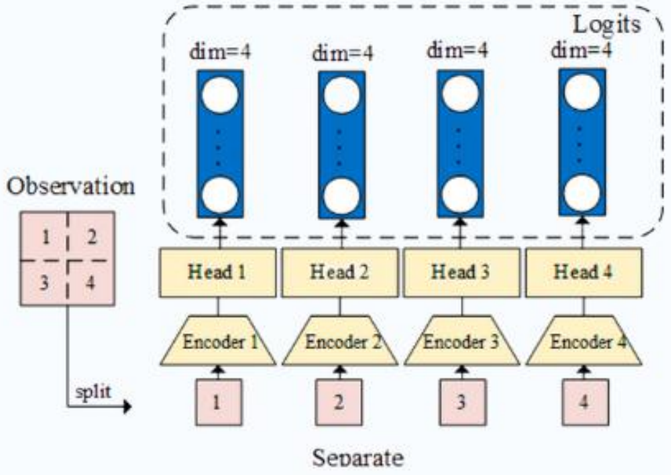
CityFlow: https://github.com/cityflow-project/CityFlow

理论：PPO+多维离散架构







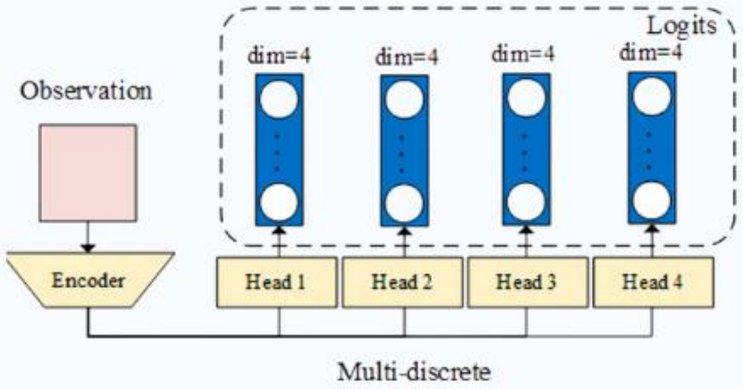


四种建模方法：

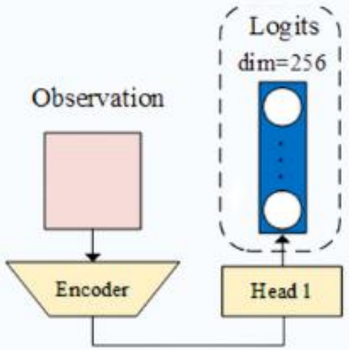
- Multi-Discrete
- Discrete
- Different encoders
- Separate

多个head的PPO优化目标各自计算，加起来一起BP

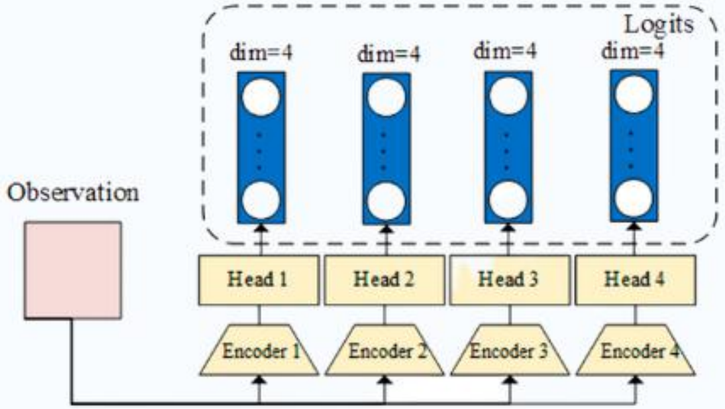
理论：PPO+多维离散架构



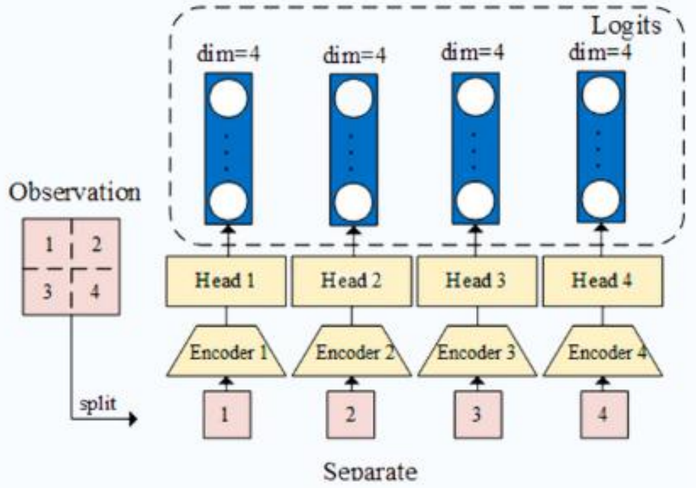
Multi-discrete



Discrete



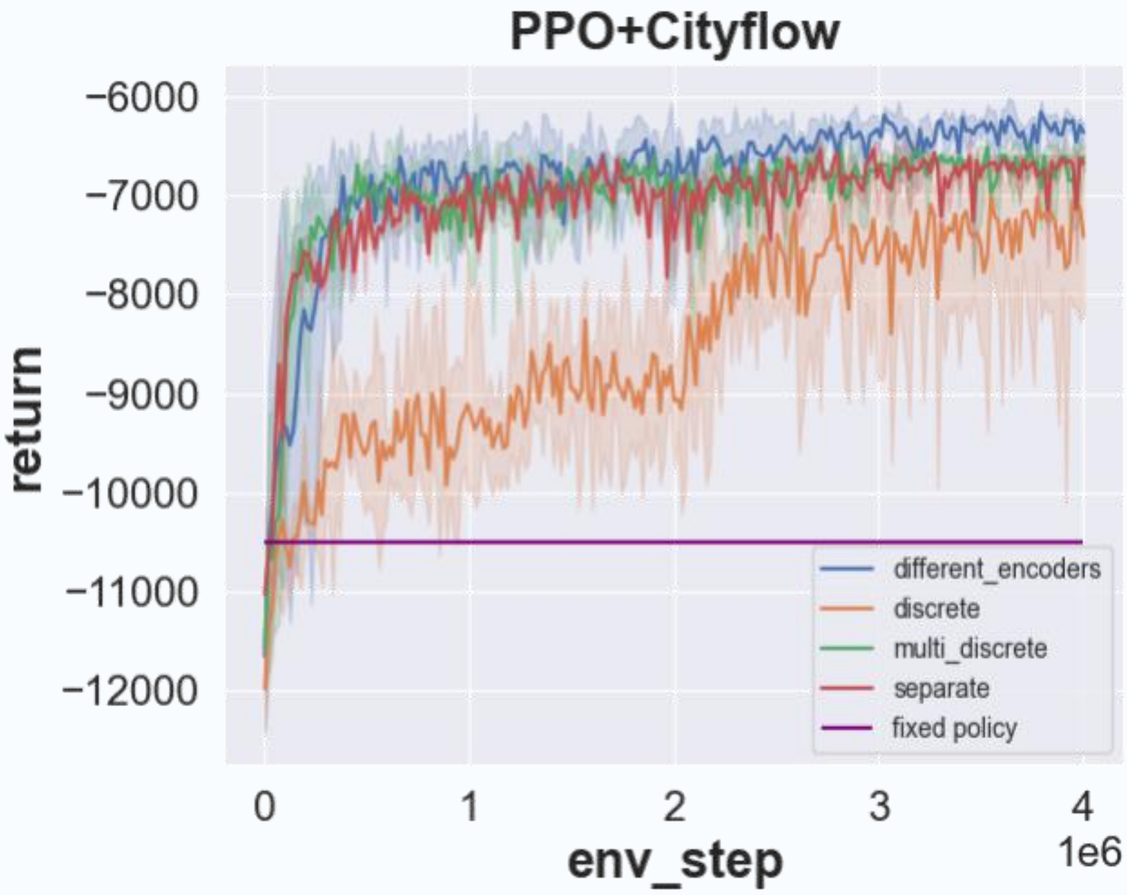
Different encoders

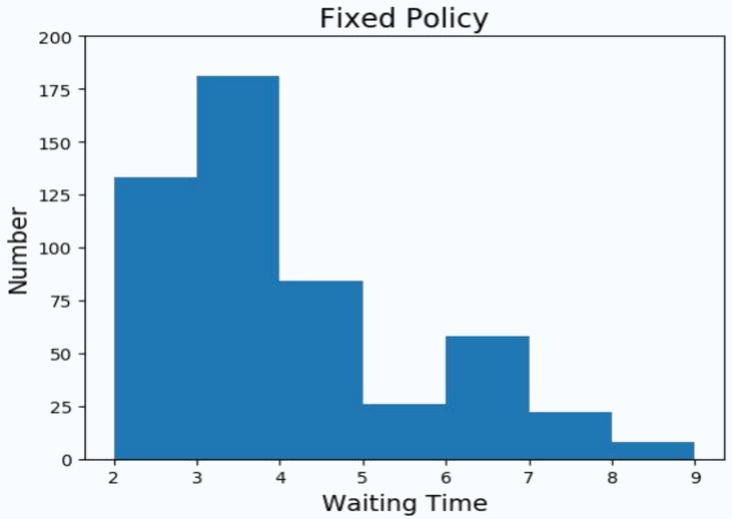


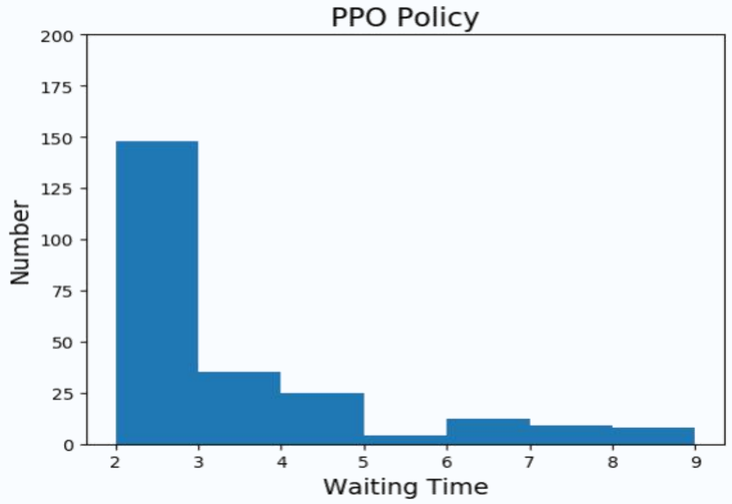
Separate

实验名	encoder 输入维度	head 输出维度	encoder 数量	head 数量
Multi-discrete	174	4	1	4
Discrete	174	256	1	1
Different encoders	174	4	4	4
Separate	44	4	4	4

实践：PPO+交通信号控制

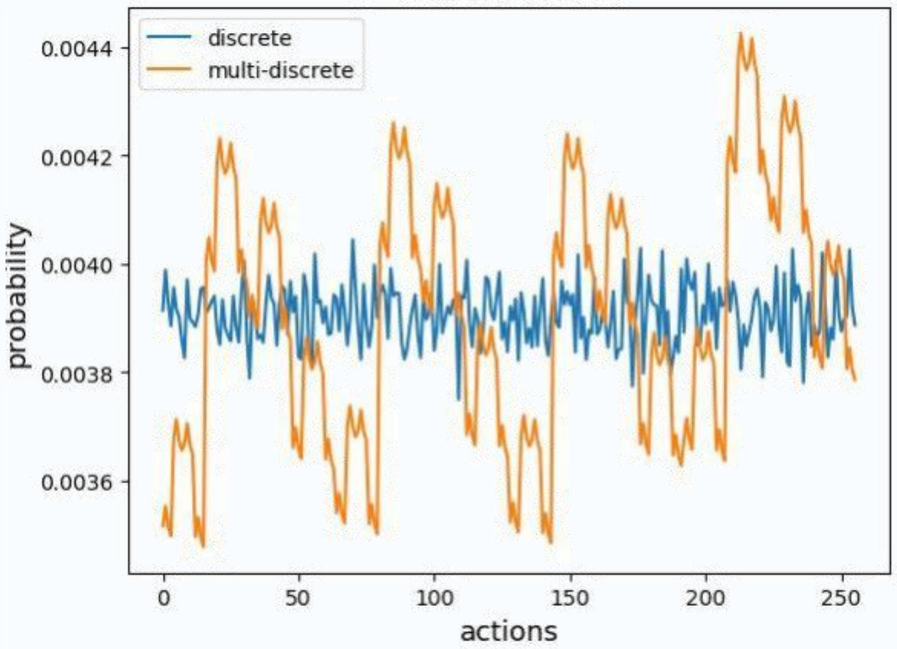


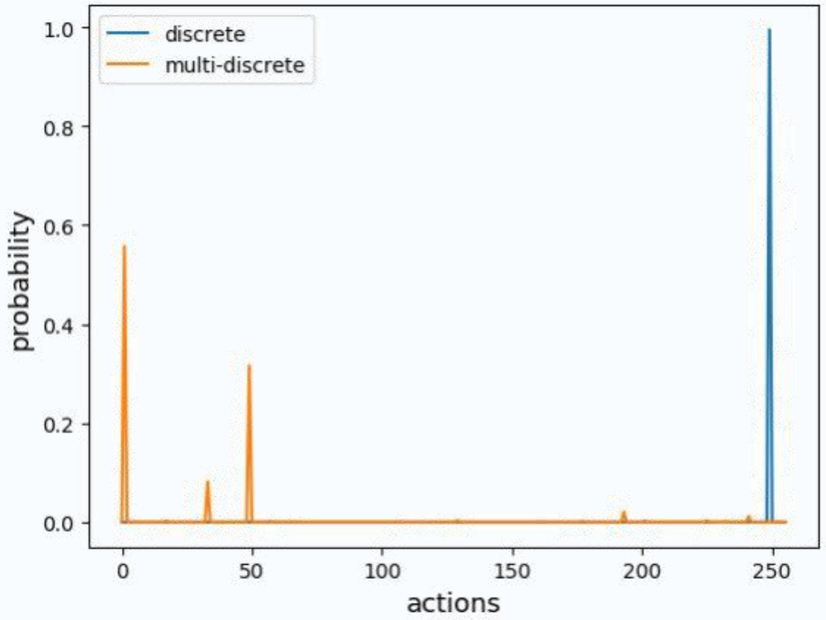




实践：PPO+交通信号控制

为什么 multi-discrete 在这个场景里优于 discrete（探索角度）





不同训练迭代下，策略选择各个动作的概率（左1k，右1w）

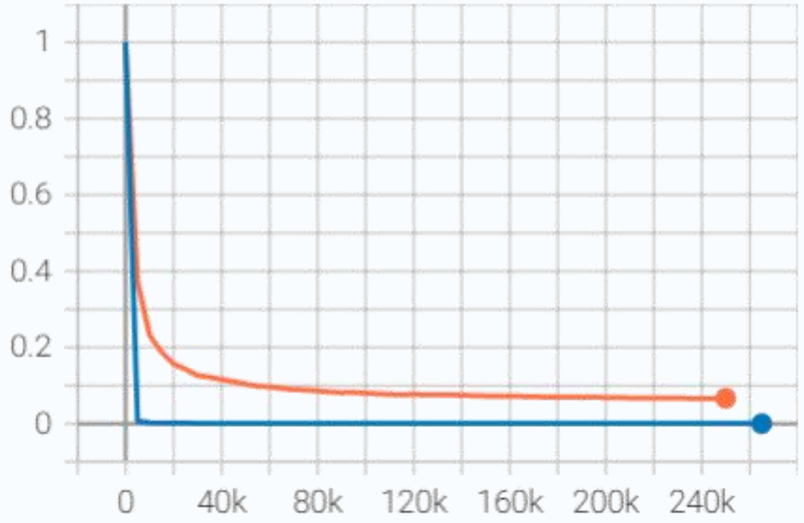
实验细节：https://github.com/opendilab/PPOxFamily/issues/4

实践：PPO+交通信号控制

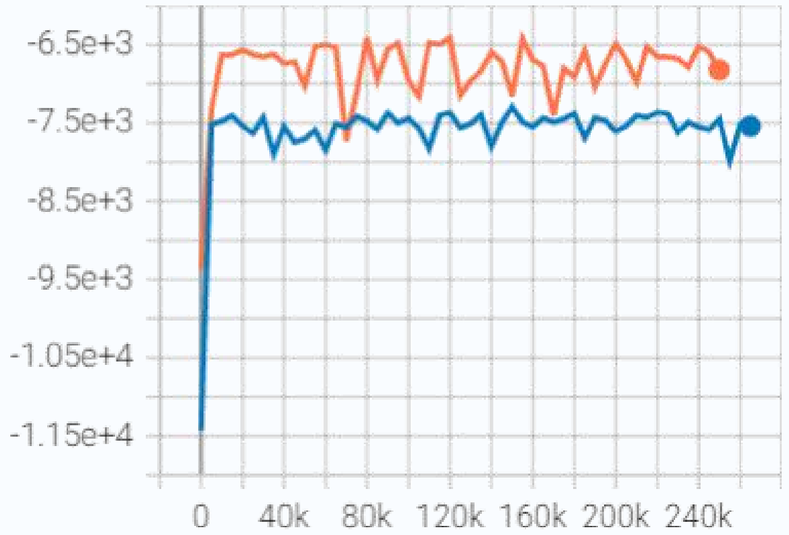
为什么 multi-discrete 在这个场景里优于 discrete（利用角度）

——专家数据集上的 Behavior Cloning 实验

Training Loss



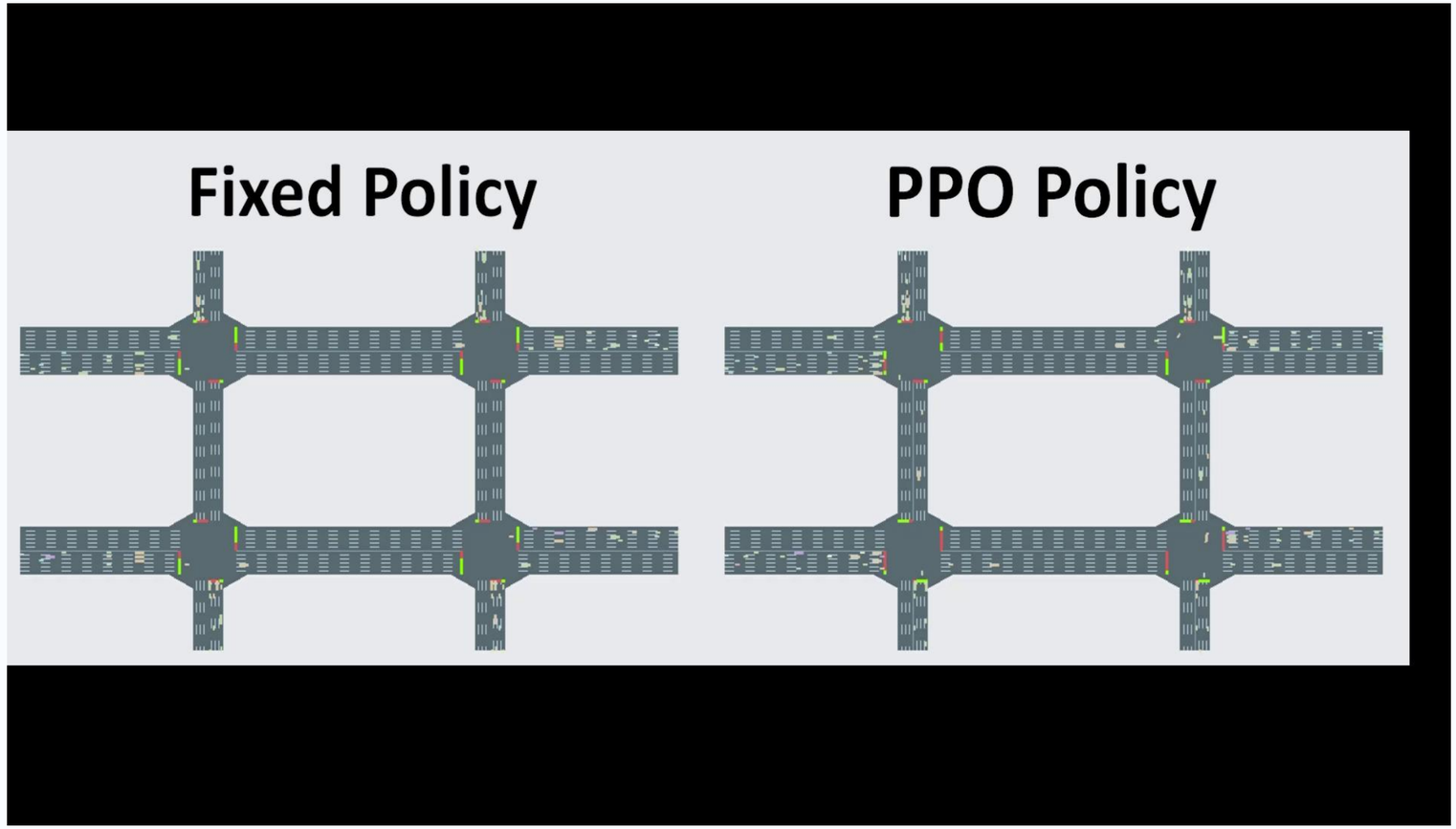
Evaluation Episode Return



蓝线是 discrete，橙线是 multi-discrete，横轴是训练迭代数

实验细节：https://github.com/opendilab/PPOxFamily/issues/4

实践：PPO+交通信号控制



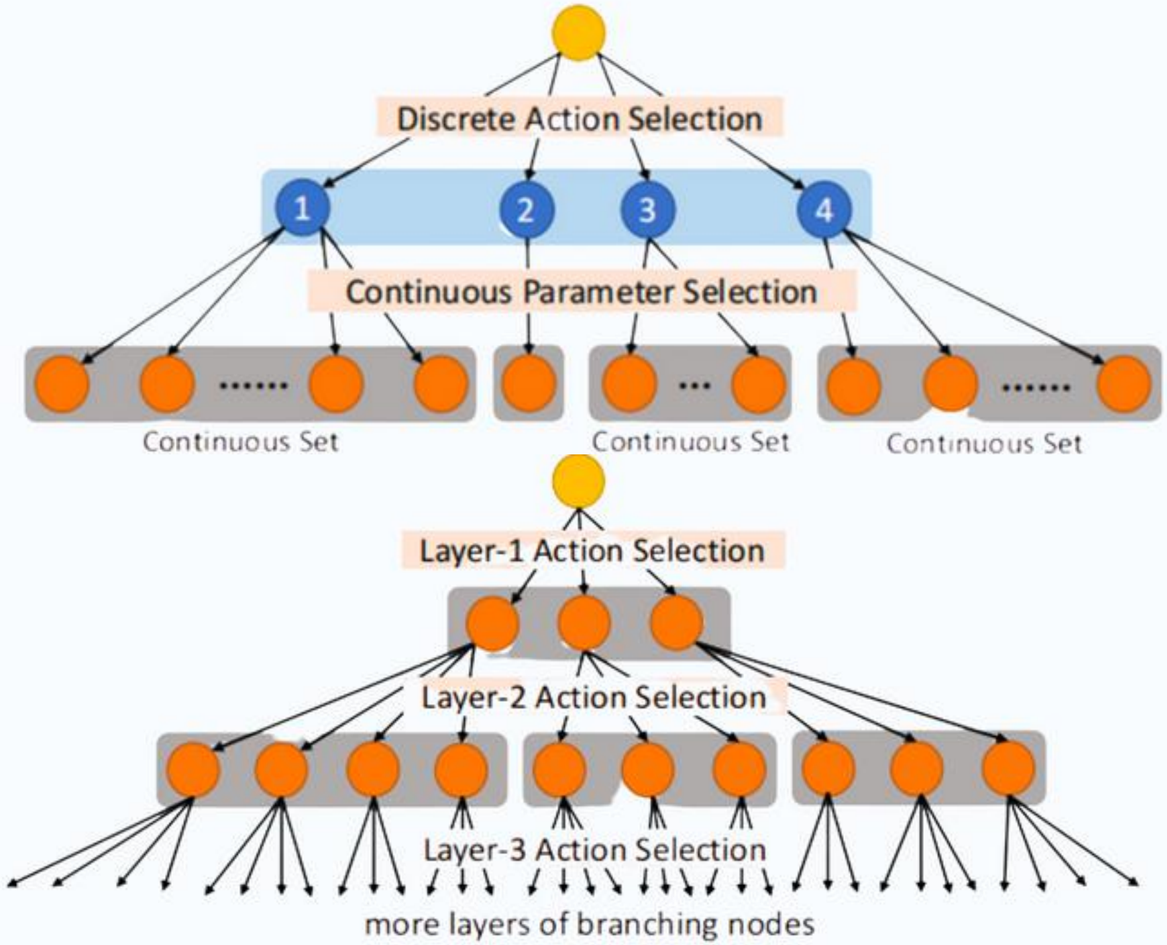
视频样例：https://github.com/opendilab/PPOxFamily/issues/4

Hybrid

混合动作空间

AlphaStar: https://www.deepmind.com/blog/alphastar-grandmaster-level-in-starcraft-ii-using-multi-agent-reinforcement-learning

理论：PPO+混合动作 定义



参数化动作空间：

- 离散动作类型和连续动作参数
- 动作类型和参数之间有强依赖关系
- 不同动作类型对应不同参数

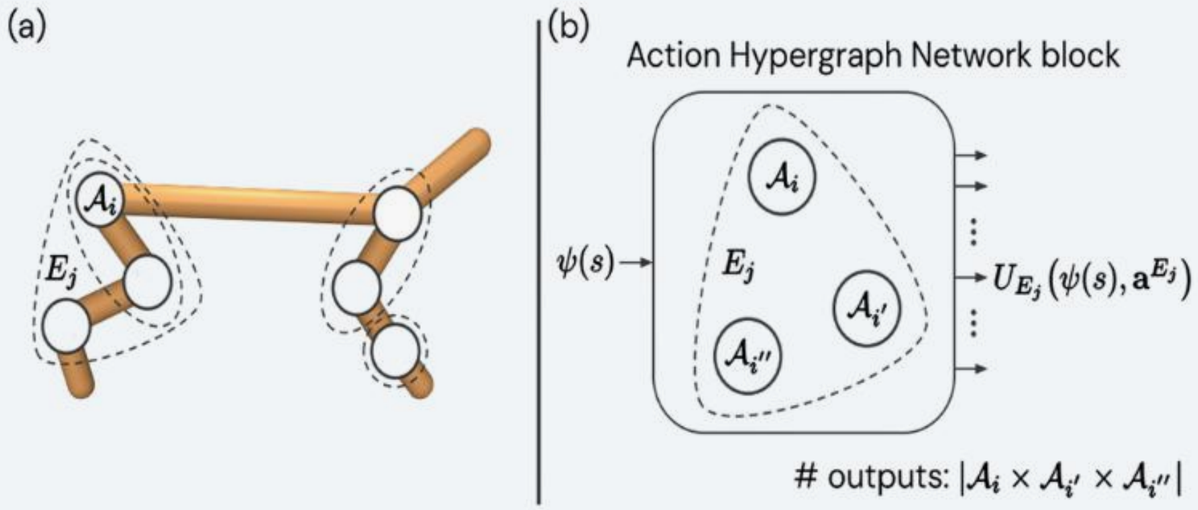
层次结构化动作空间：

- 树形结构、中间节点是离散动作，叶节点是离散或连续参数
- 各个子树之间的动作性质相差很大

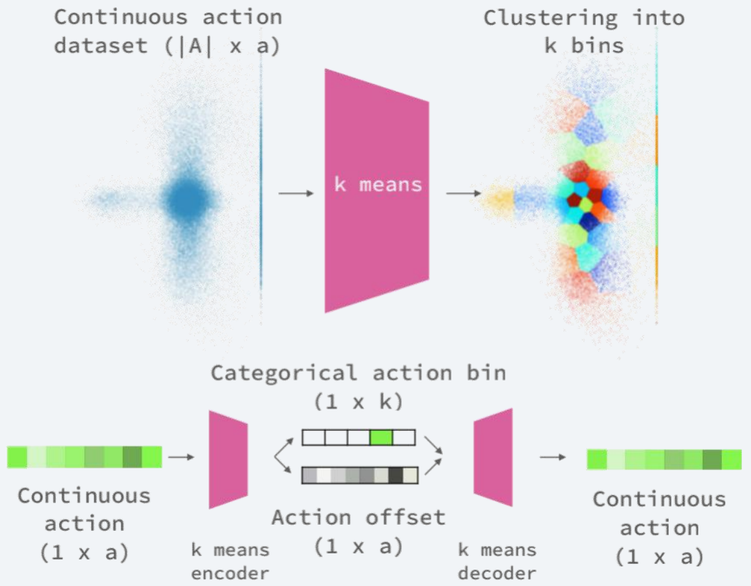
H-PPO: https://arxiv.org/pdf/1903.01344.pdf

理论：PPO+混合动作 离散化

根据动作空间的先验知识离散化

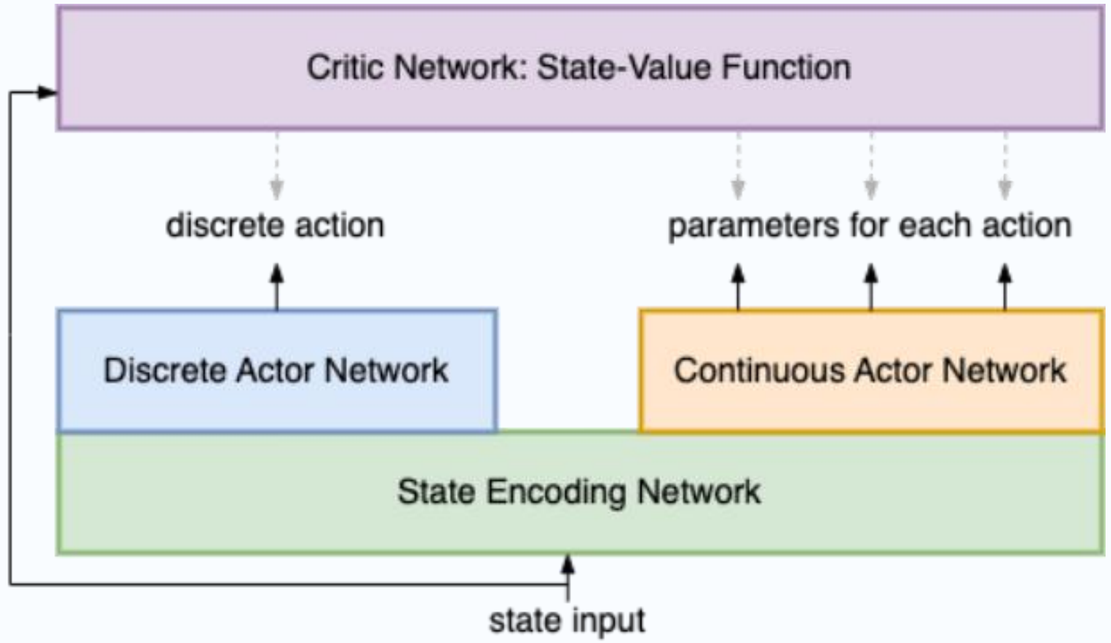


根据数据特性自动挖掘离散化



HGQN: https://openreview.net/pdf?id=Xv_s64FiXTv
Behavior Transformers: https://arxiv.org/pdf/2206.11251.pdf

理论：PPO+混合动作 HPPO



核心思路

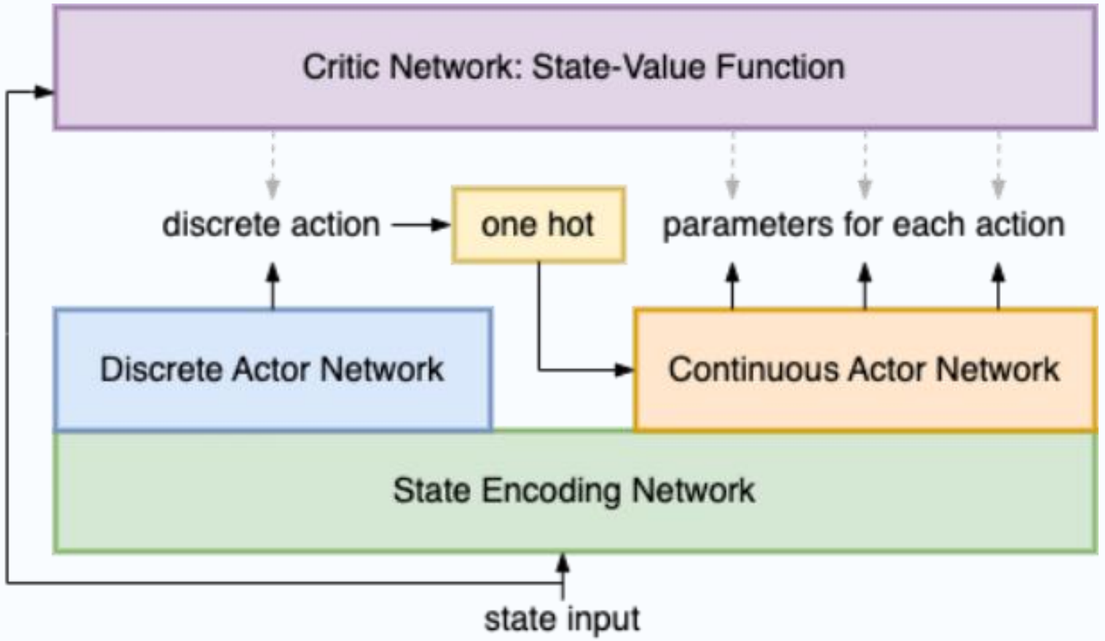
- 离散和连续部分各自使用相应的 PPO，但是共享特征编码部分
- 整体共用一个 Value 指导优化

优势： 实现简洁，可扩展性高

劣势： 并没有显式去建模动作间的依赖

H-PPO: https://arxiv.org/pdf/1903.01344.pdf

理论：HPPO优化方案 自回归

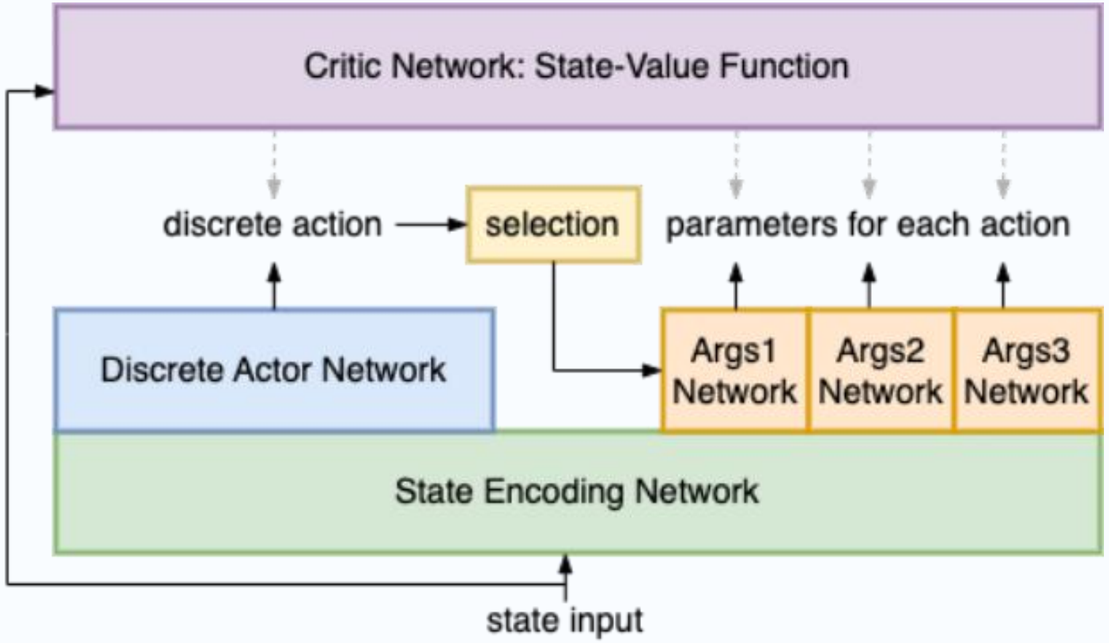


核心思路：

- 对于动作之间的强依赖关系，可以采用自回归（autogressive）的方式建模依赖关系

例如：离散的动作类型决定相应的连续参数（转向和转向角），可以将选出的离散动作进行one-hot编码，再和提取到的状态特征拼到一起，去预测连续参数

理论：HPPO优化方案 分离head



核心思路：

- 对于动作之间的强依赖关系，可以采用分离head的方式建模依赖关系

具体来说，对于每个离散动作的连续参数，单独设置一个连续策略网络，根据选出的离散动作，选择其中对应的一个连续网络输出（仅限于离散动作数量较少，否则开销过大）

更多混合动作空间的解决思路（动作空间表征学习）：https://github.com/opendilab/PPOxFamily/blob/main/chapter2_action/chapter2_supp_hyar.pdf

代码：使用mask和TreeTensor

PyTorch demo of PPO algorithm in hybrid action space.

Stars 797 | bilibili video course | Follow @opendilab 41

View code on GitHub

Overview

The definition of hybrid action policy network used in PPO, which is mainly composed of three parts: encoder, action_type head (discrete) and action_args head (continuous).

```python
from typing import Dict
import torch
import torch.nn as nn
from torch.distributions import Normal, Independent


class HybridPolicyNetwork(nn.Module):
    def __init__(self, obs_shape: int, action_shape: int) -> None:
```

PyTorch necessary requirements for extending `nn.Module`.

```python
        super(HybridPolicyNetwork, self).__init__()
```

Define encoder module, which maps raw state into embedding vector.
It could be different for various state, such as Convolution Neural Network for image state.
Here we use two-layer MLP for vector state.

```python
        self.encoder = nn.Sequential(
            nn.Linear(obs_shape, 16),
            nn.ReLU(),
            nn.Linear(16, 32),
            nn.ReLU(),
        )
```

Define action_type head module, which outputs discrete logit.

```python
        self.action_type_shape = action_shape['action_type_shape']
        self.action_type_head = nn.Linear(32, self.action_type_shape)
```

Define action_args head module, which outputs corresponding continuous action arguments.

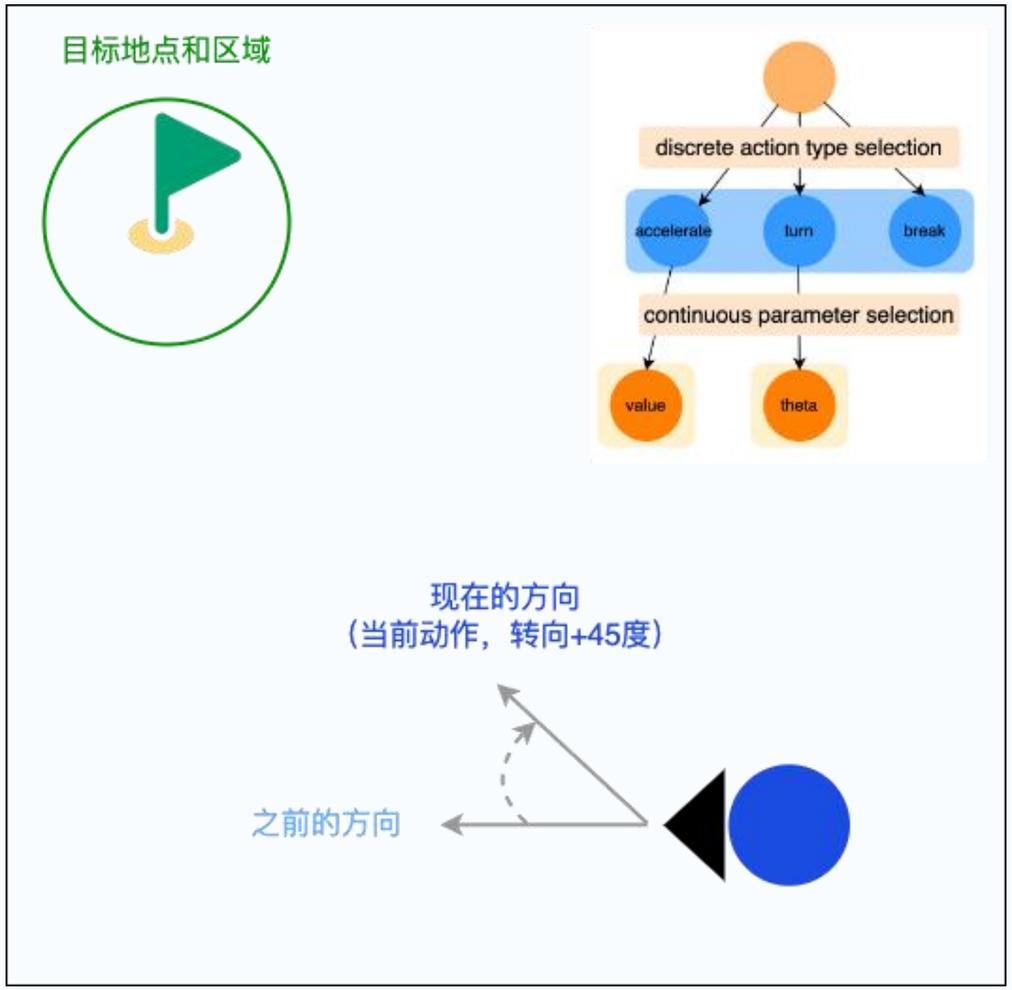
```python
        self.action_args_shape = action_shape['action_args_shape']
        self.action_args_mu = nn.Linear(32, self.action_args_shape)
        self.action_args_sigma = nn.Parameter(torch.zeros(1, self.action_args_shape))

```

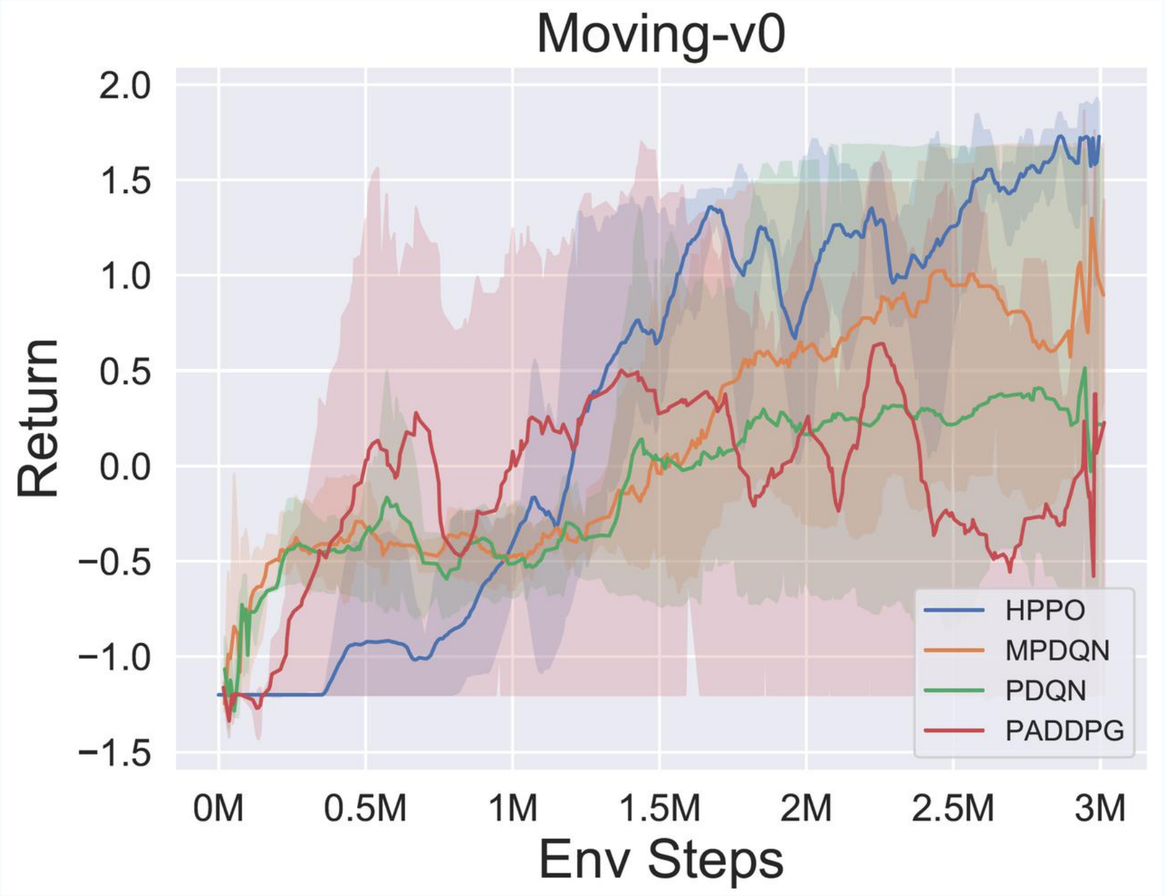
TensorTree on Nested Data

```python
import tensortree.torch as torch

# create a nested data tensor
t = torch.randn({
    'a': (6, 2, 3),
    'b': {'x': (6, 3), 'y': (6, 1, 4)},
})

# structural operations
print(torch.stack([t, t]))
print(torch.split(t, (1, 2, 3)))
# math calculations
print(t ** 2)
print(torch.sin(t).cos())
# access like attribute
print(t.b.y)
```

完整代码示例：https://opendilab.github.io/PPOxFamily/
TreeTensor：https://github.com/opendilab/DI-treetensor

实践：PPO+导航控制





环境文档：https://di-engine-docs.readthedocs.io/zh_CN/latest/13_envs/gym_hybrid_zh.html
相关对比算法参考博客：https://zhuanlan.zhihu.com/p/478728586

实践：PPO+导航控制

Random Agent

Trained Agent

实验环境：https://github.com/opendilab/DI-engine/tree/main/dizoo/gym_hybrid/envs
视频样例：https://github.com/opendilab/PPOxFamily/issues/4

下节预告

（三）降维之术：表征多模态观察空间

- 向量观察空间的千层套路
- 图片观察空间的最佳实践
- 结构化观察空间的奇技淫巧