

TD-MPC

Presented by:

Erisa KOHANSAL
Sama SATARIYAN
Kim SAÏDI
Yanis DJERMOUNI

2023-2024
PANDROIDE

CONTENT

Introduction of the Project

Summary of the Paper

Delving Deeper Into the Model

Presenting the BBRL Library

Adjustment Needed for the Integration

Integration of Model into the Library

INTRODUCTION OF THE PROJECT

Integration of "Temporal Difference Learning for Model Predictive Control" algorithm by Nicklas Hansen, Xiaolong Wang and Hao Su into BBRL library.

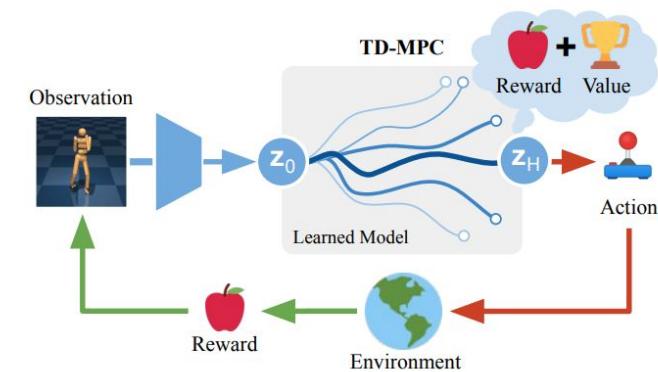
SUMMARY OF THE PAPER

Temporal Difference Learning for Model Predictive Control

Nicklas Hansen¹ Xiaolong Wang^{* 1} Hao Su^{* 1}

Abstract

Data-driven model predictive control has two key advantages over model-free methods: a potential for improved sample efficiency through model learning, and better performance as computational budget for planning increases. However, it is both costly to plan over long horizons and challenging to obtain an accurate model of the environment. In this work, we combine the strengths of



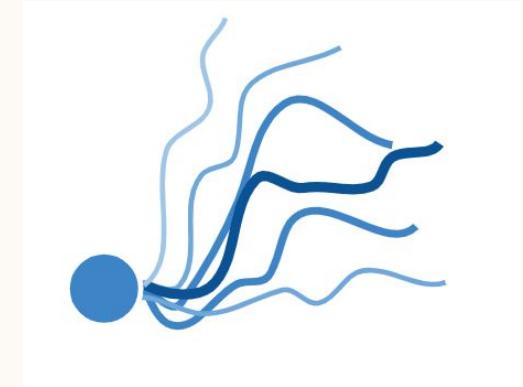
- TD-MPC is a novel framework in RL that integrates the strengths of **model-free** and **model-based** approaches for data-driven model predictive control.
- It leverages a learned task-oriented latent dynamics (TOLD) model for short-term trajectory optimization and a learned terminal value function for long-term return estimation, both optimized *jointly* through temporal difference learning.

MODEL PREDICTIVE CONTROL

- Model-based algorithm
- Planning is done using a learned model of the environment
- It models everything in the environment



Downside?



MODEL PREDICTIVE CONTROL

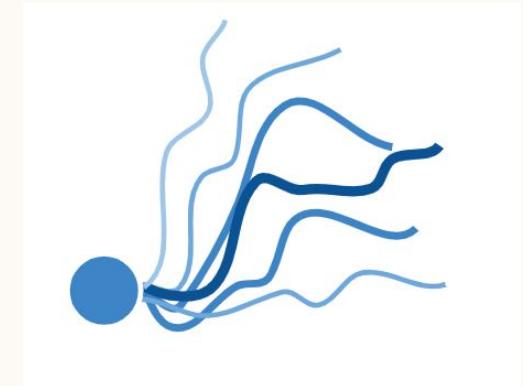
- Model-based algorithm
- Planning is done using a learned model of the environment
- It models everything in the environment



Downside?



How to resolve ?



Total return of a sampled trajectory :

$$\phi_{\Gamma} \triangleq \mathbb{E}_{\Gamma} \left[\gamma^H Q_{\theta}(\mathbf{z}_H, \mathbf{a}_H) + \sum_{t=0}^{H-1} \gamma^t R_{\theta}(\mathbf{z}_t, \mathbf{a}_t) \right]$$

TD LEARNING

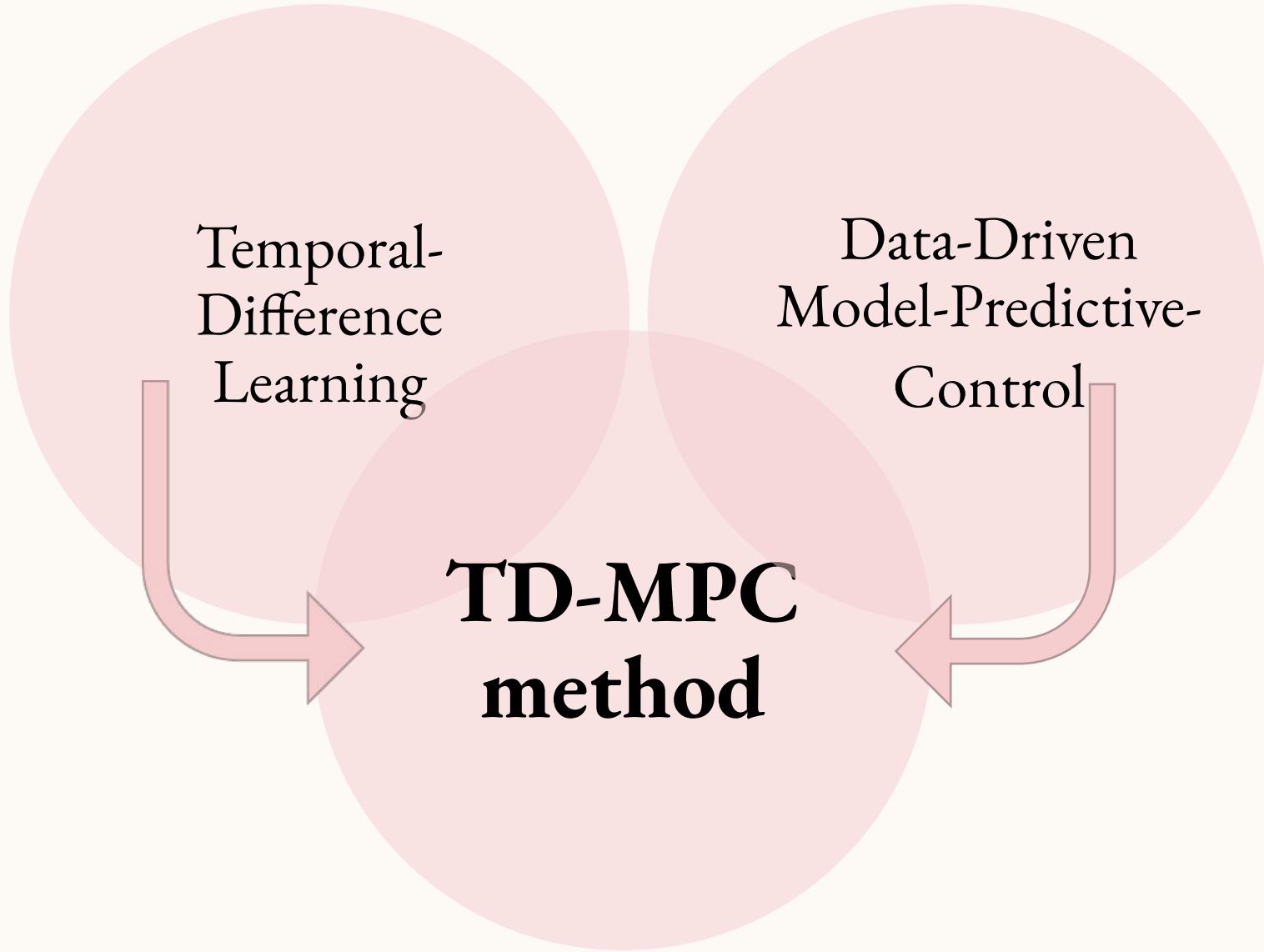
- Model-free algorithm
- Learning by bootstrapping from the current estimate of value function through samples from the environment

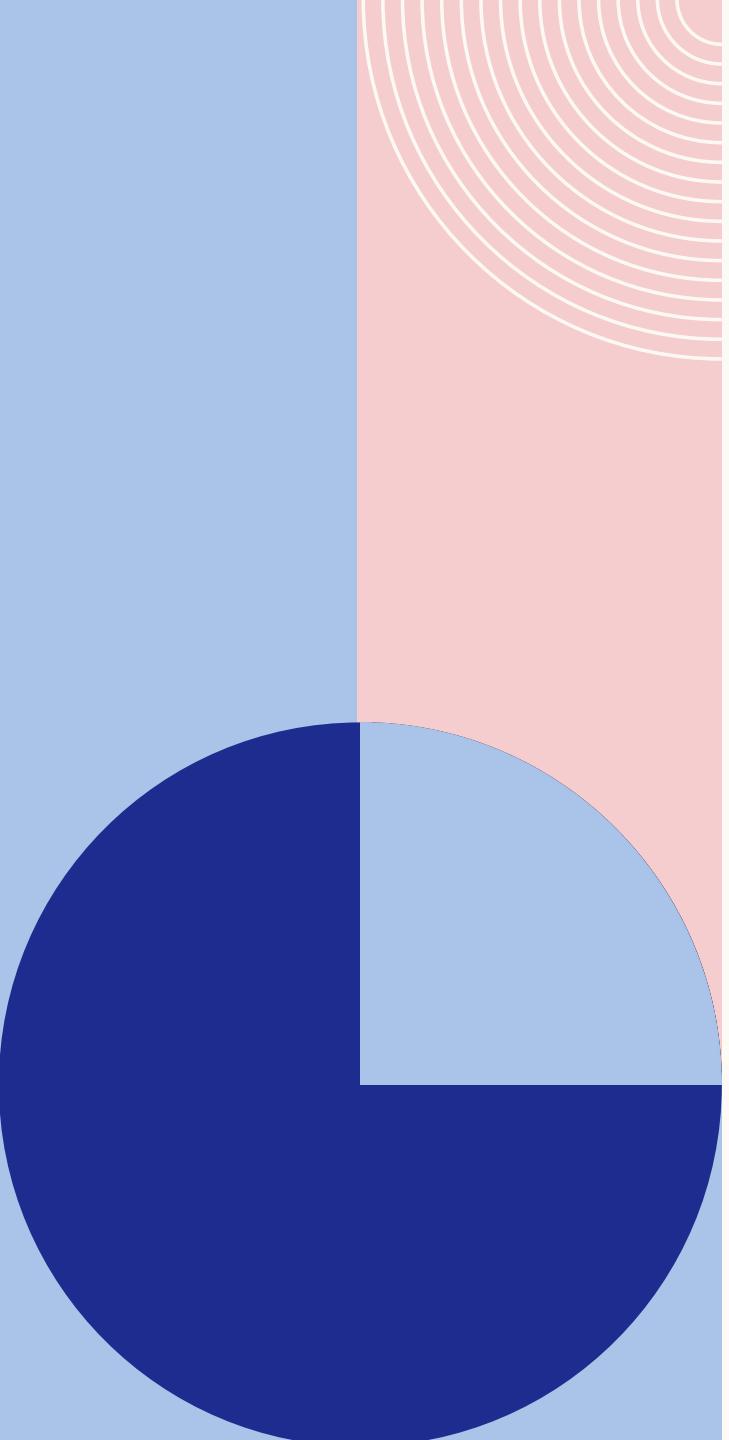


Downside?



Can TD-Learning mitigate challenges of MPC?

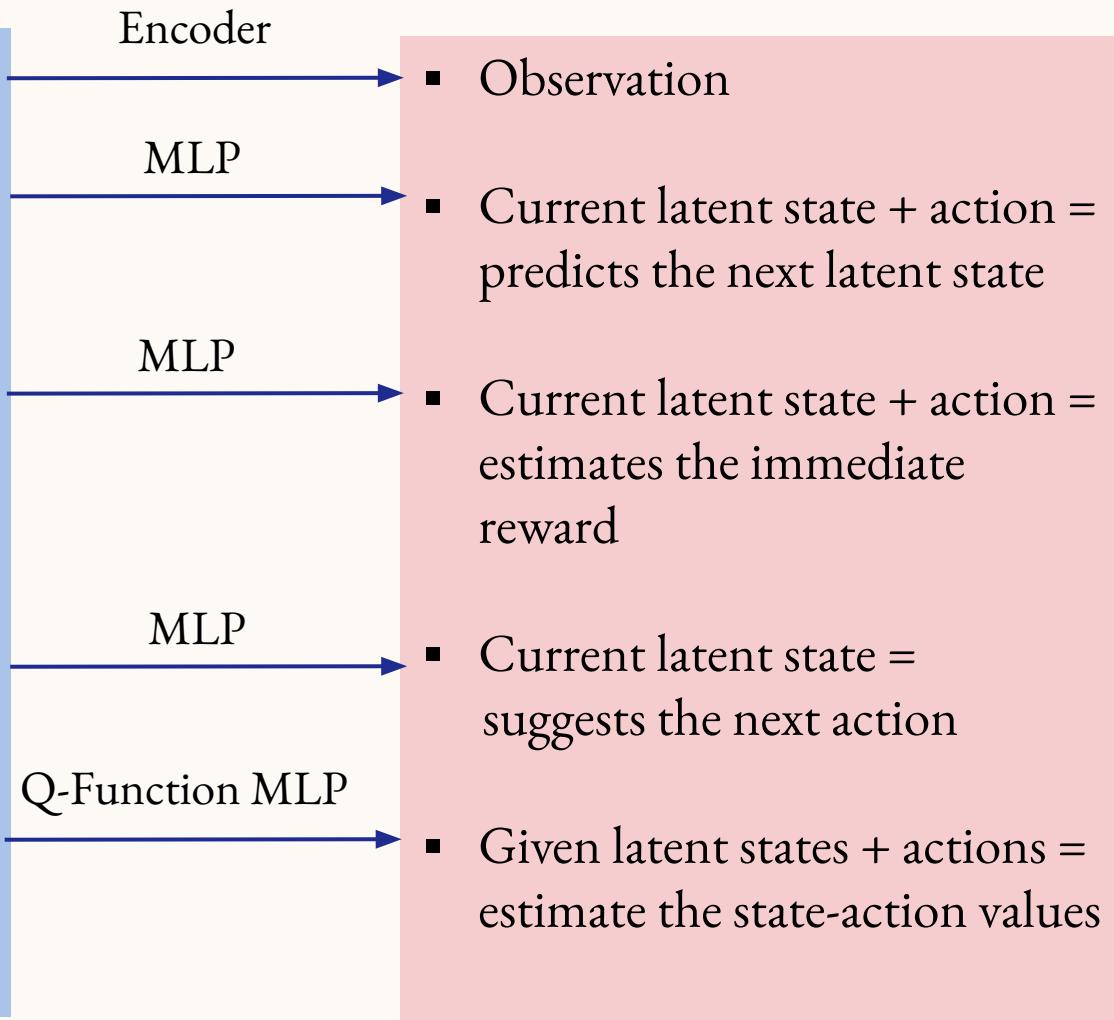




DELVING DEEP INTO THE MODEL/CODES OF THE PAPER

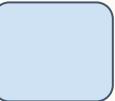
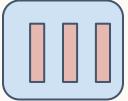
Task-Oriented Latent Dynamics Model

- Latent Representation
- Latent Dynamics Model
- Reward Model
- Policy Model
- Double Q-learning

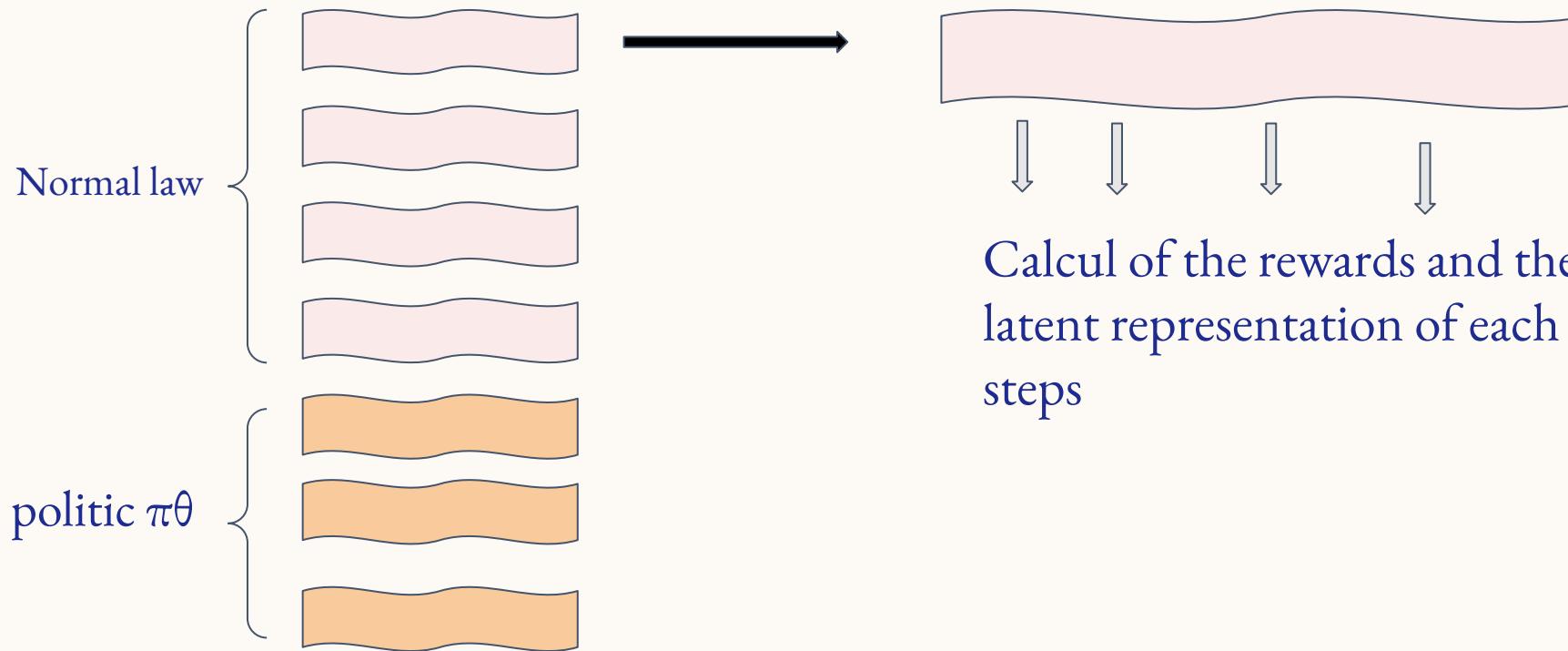


- $z_t = h_\theta(st)$
- $z_{t+1} = d_\theta(z_t, a_t)$
- $\hat{r}_t = R_\theta(z_t, a_t)$
- $\hat{a}_t \sim \pi_\theta(z_t)$
- $\hat{q}_t = Q_\theta(z_t, a_t)$

Overview

1. Initialization of a replay buffer 
2. Initialization of an episode 
3. **Collect trajectory** 
4. Add the full episode to the replay buffer 
5. **Update the model** with the buffer
6. Go to step 2

Planning

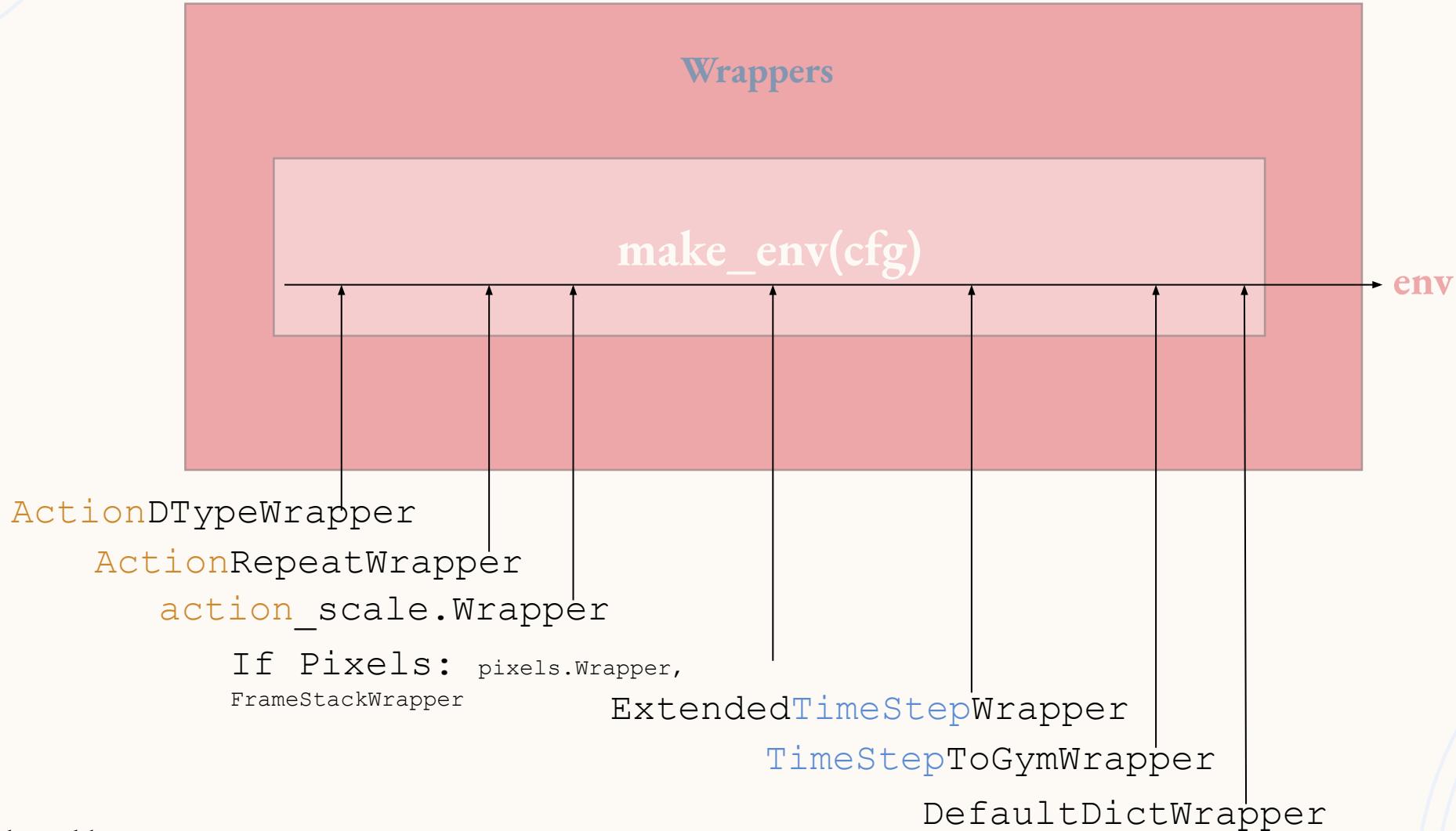


Calcul of the terminal value function

Update of the model

Use of TD error to make the difference between the predict rewards and the actual rewards

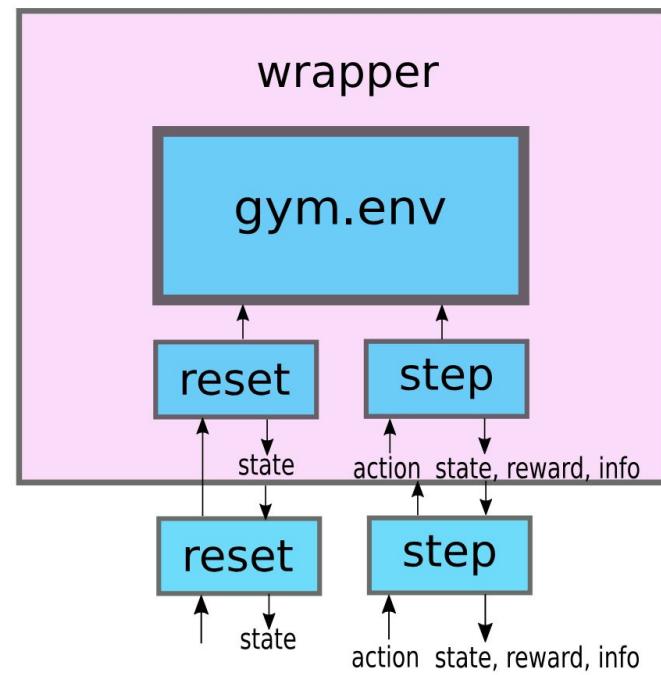
Environment Architecture (env.py)



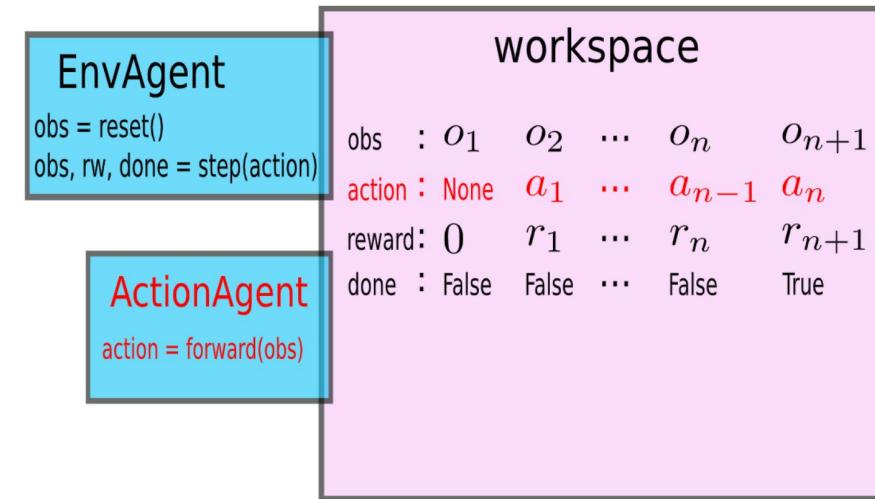
PRESENTING BBRL LIBRARY

- **BBRL** (BlackBoard RL) Library is a derivation from **SaLinA** Library but limited to RL algorithms.

WRAPPERS



BBRL Structure



- **BBRL** consists of Workspace and Agents
- Workspace is a blackboard where all agents read and write temporal data, everything else is agent
- Agents are **PyTorch nn.Modules**

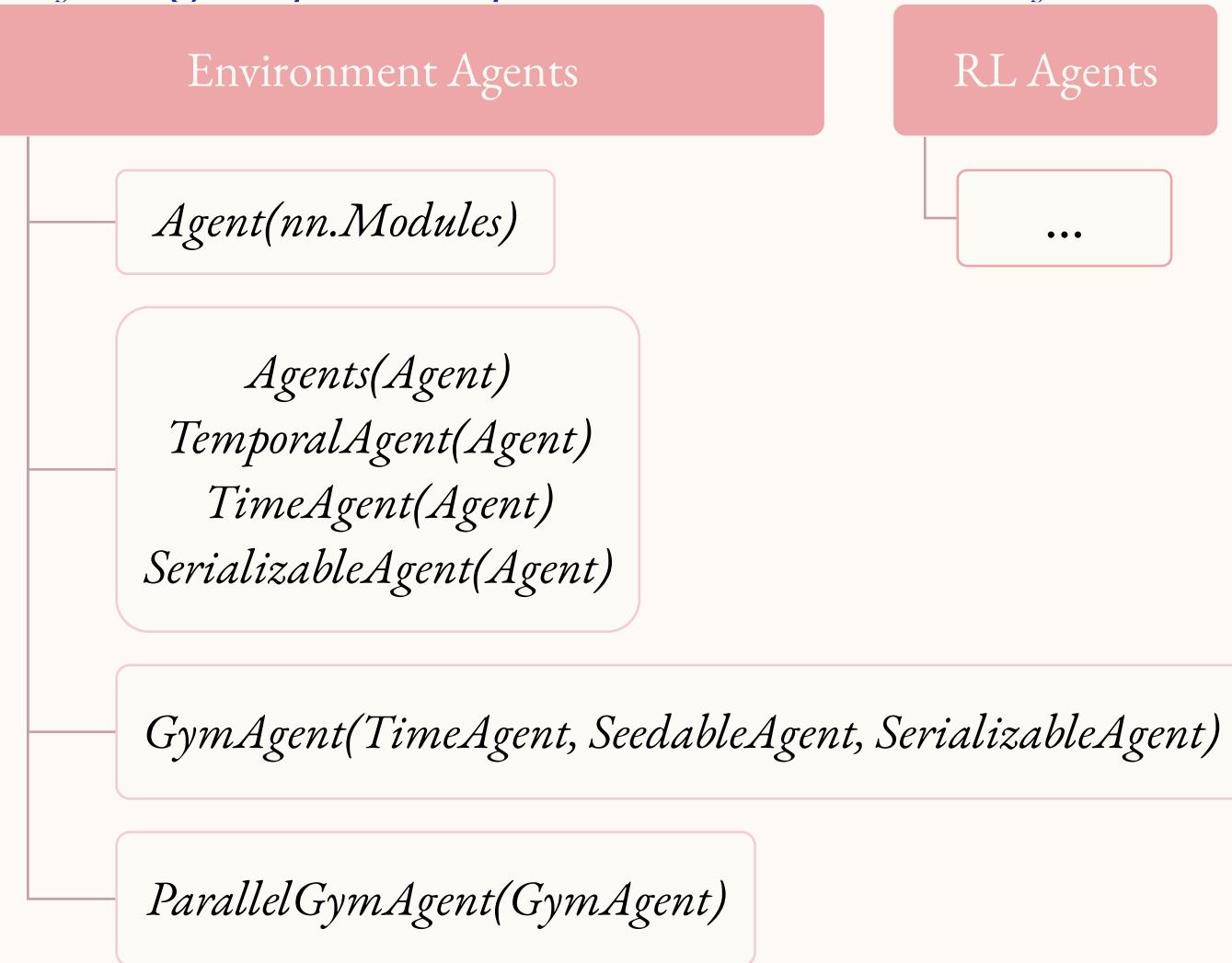
WORKSPACE AND REPLAYBUFFER IN THE BBRL LIBRARY

- Workspace is a collection of tensors (*SlicedTemporalTensor*, *CompactTemporalTensor* or *CompactSharedTensor*) organized in a dictionary-like structure. It is designed to manage the state and data during learning processes.
- ReplayBuffer utilizes the Workspace class to manage the storage and retrieval of experiences efficiently during learning

				Workspace	
state :	s_0	s_1	...	s_n	s_{n+1}
action :	a_0	a_1	...	a_n	a_{n+1}
reward:	0	r_1	...	r_n	r_{n+1}
done :	False	False	...	False	True

AGENTS IN THE BBRL LIBRARY

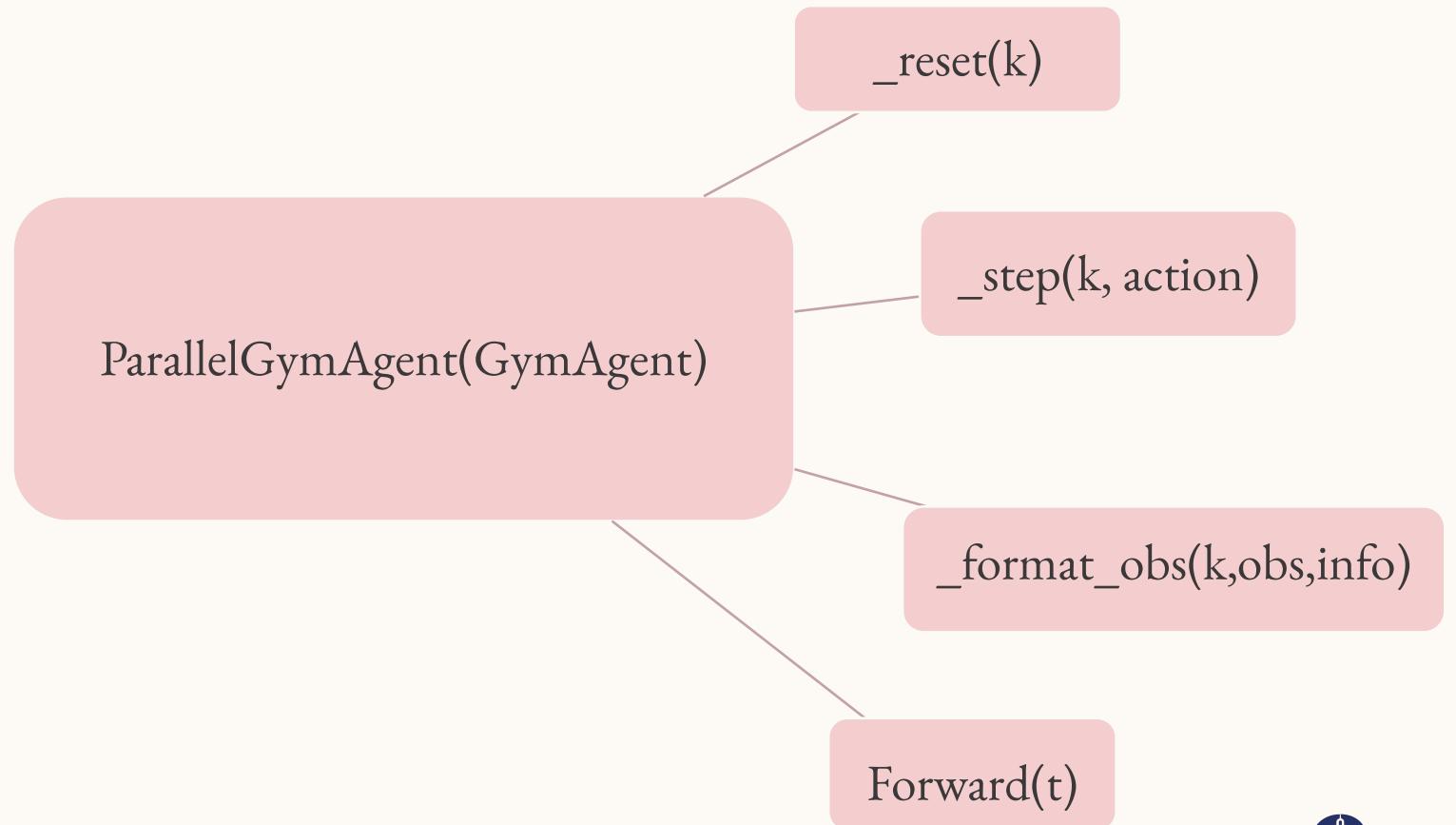
- *Everything except Workspace in the BBRL Library is structured into agents.*



workspace	
EnvAgent	obs : $o_1 \ o_2 \ \dots \ o_n \ o_{n+1}$
ActionAgent	action : None $a_1 \ \dots \ a_{n-1} \ a_n$ reward: 0 $r_1 \ \dots \ r_n \ r_{n+1}$ done : False False ... False True action = forward(obs)

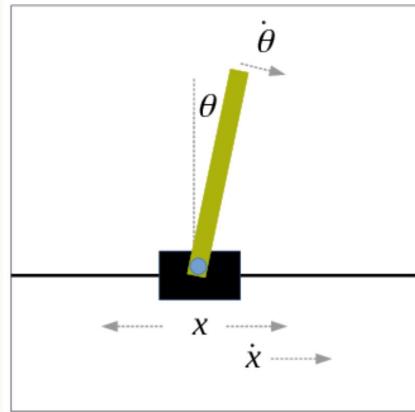
PARALLELGYMAgent(GYMAGENT)

- For creating an Agent from a gymnasium environment



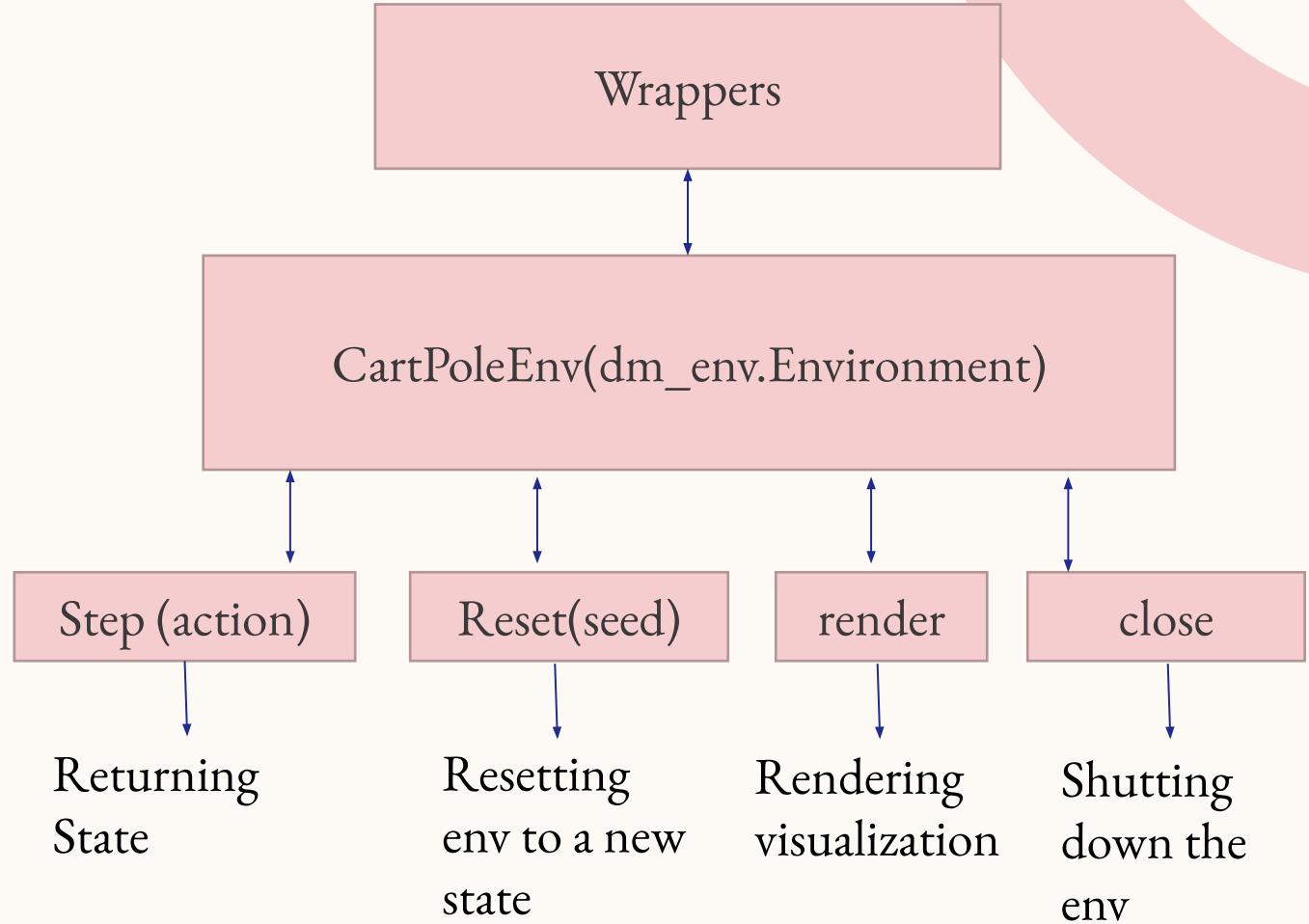
INTEGRATION OF TDMPC INTO BBRL LIBRARY

PREPROCESSING OF TD-MPC (CARTPOLE EXAMPLE TASK)

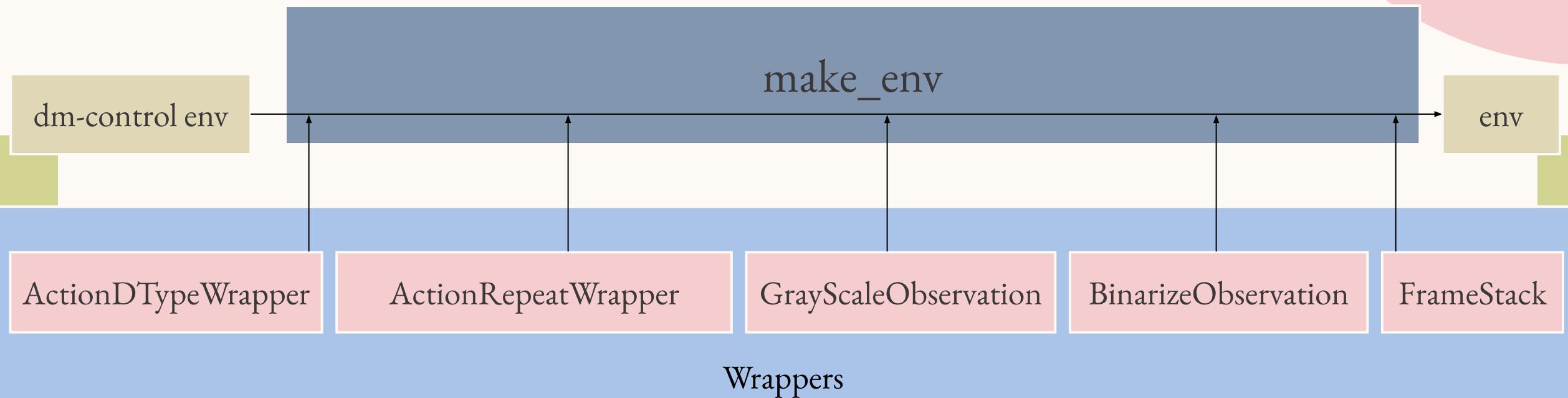


States: Position, Velocity, Angle, Angular Velocity

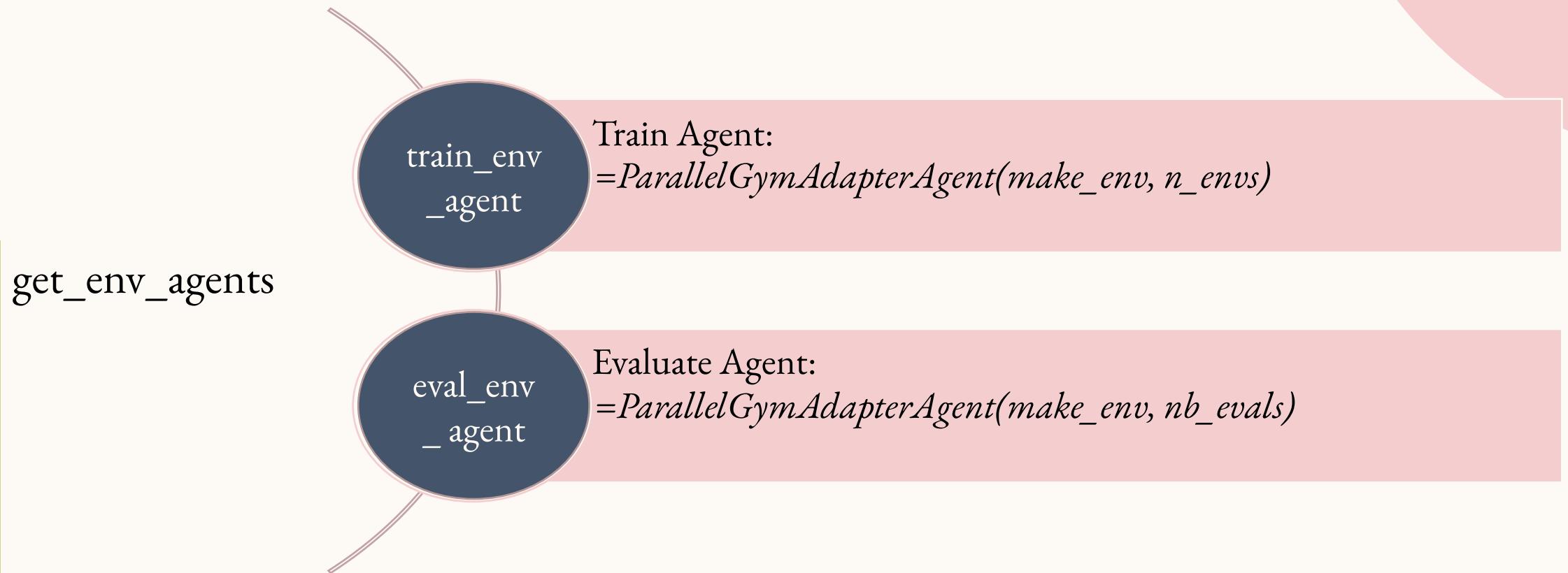
Actions: Left, Right



ENVIRONMENT OF TD-MPC

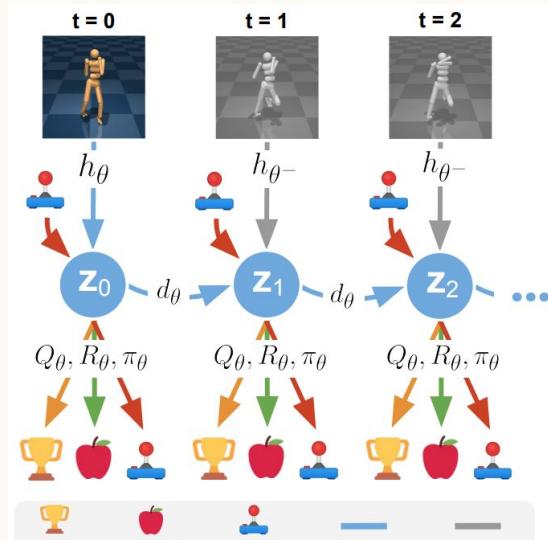


ENVIRONMENT AGENTS OF TD-MPC

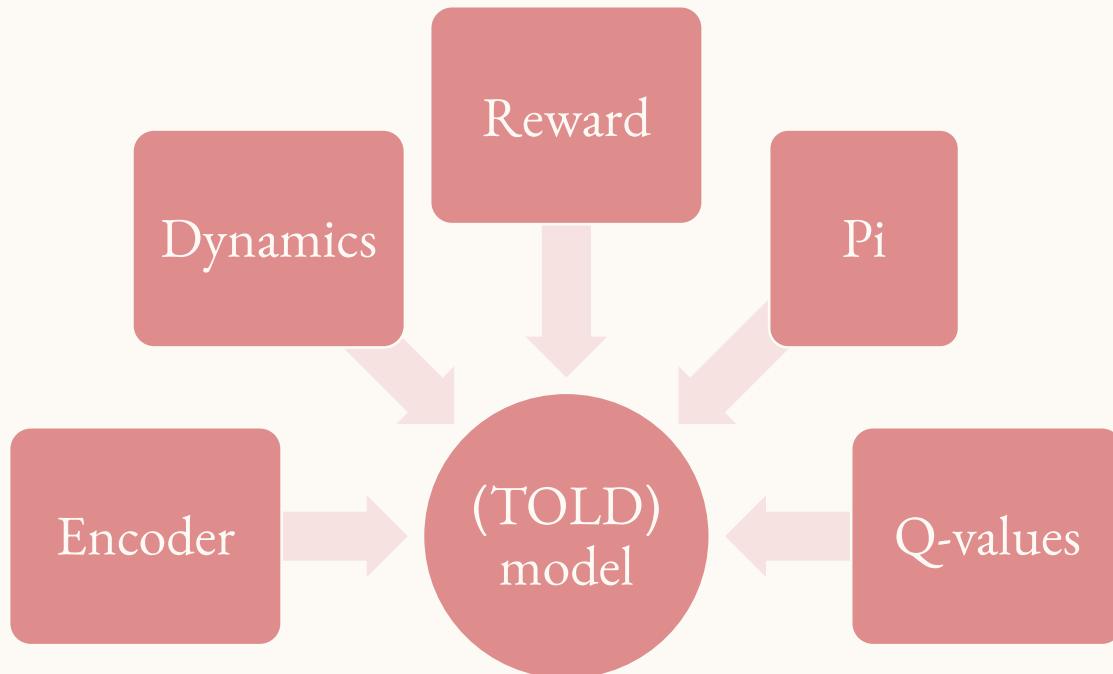


TDMPC AGENTS: TOLD MODEL

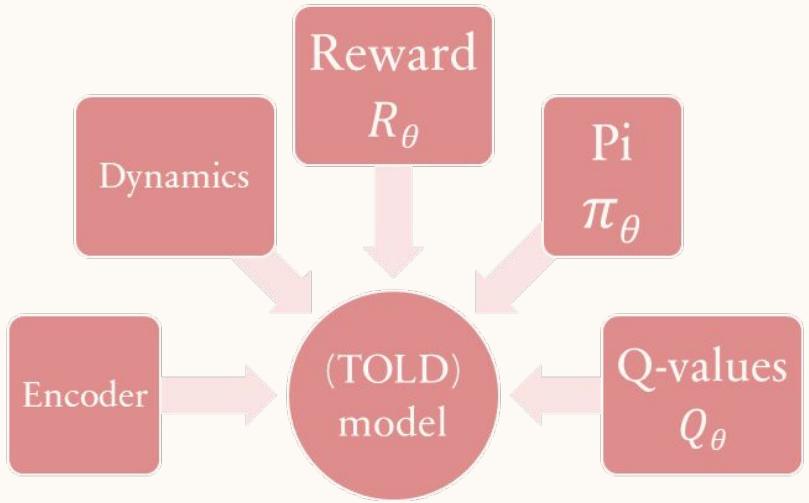
- Referring to “TOLD Structure” explained in the slide number 12



Elements of TOLD model



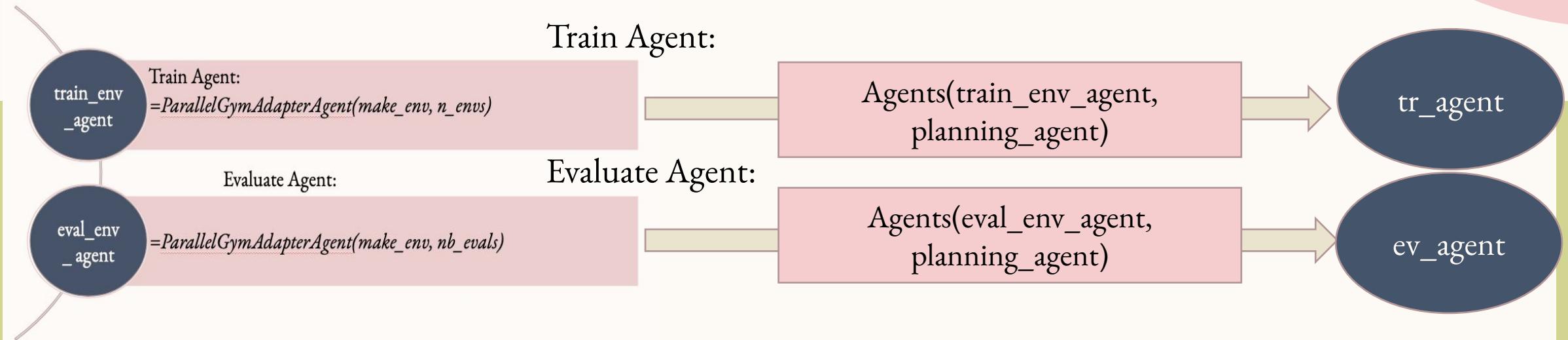
TDMPC AGENTS: TOLD AGENTS



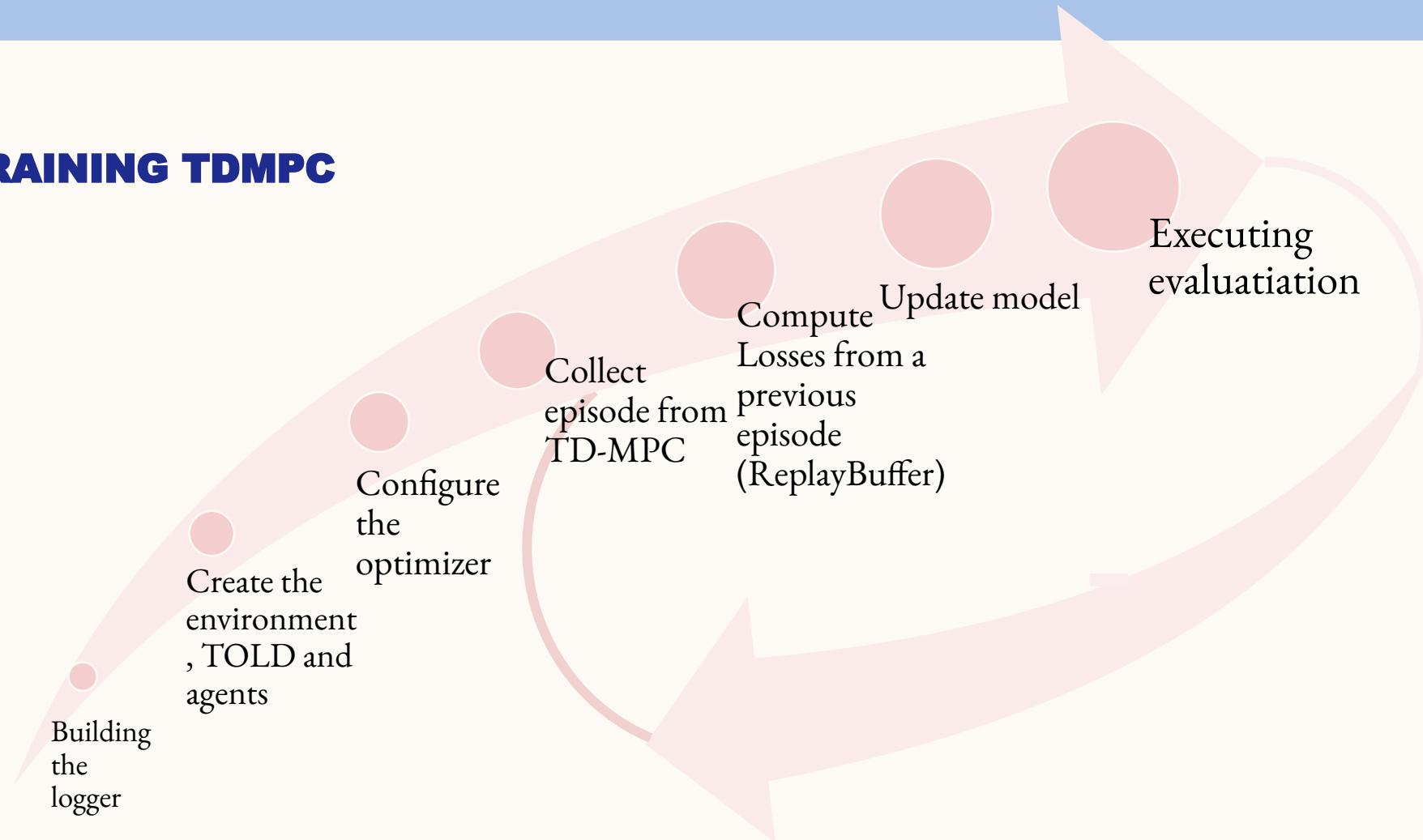
encoder	$=RepresentationAgent(Agent)$
dynamics	$=LatentDynamicAgent(Agent)$
reward	$=RewardAgent(Agent)$
Pi	$=PolicyAgent(Agent)$
Qvalues	$=ValueAgent(Agent)$

ENVIRONMENT AGENTS OF TD-MPC

- train and evaluate agents are then combined with encoder, pi, reward and dynamics agents

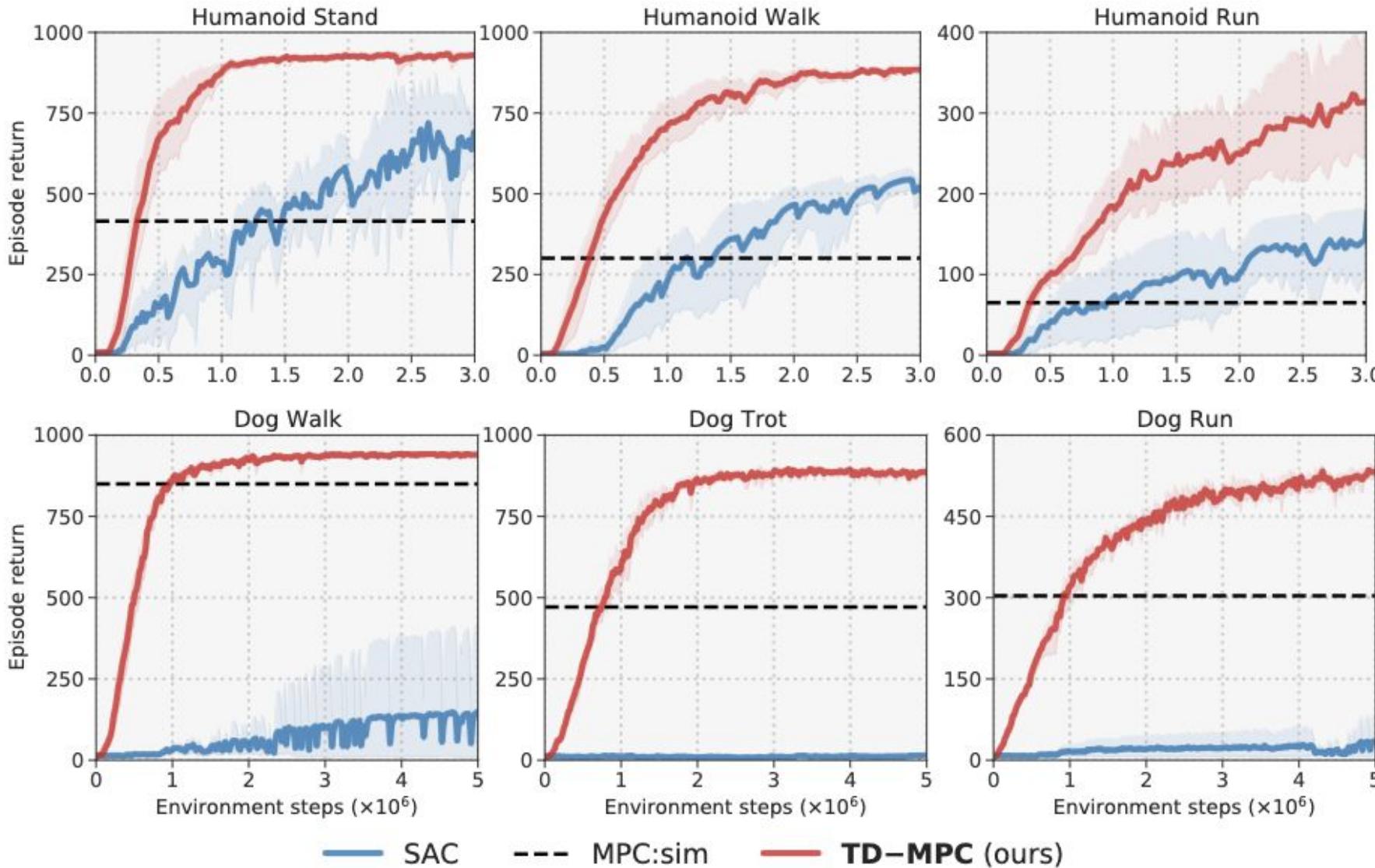


TRAINING TD-MPC



Results

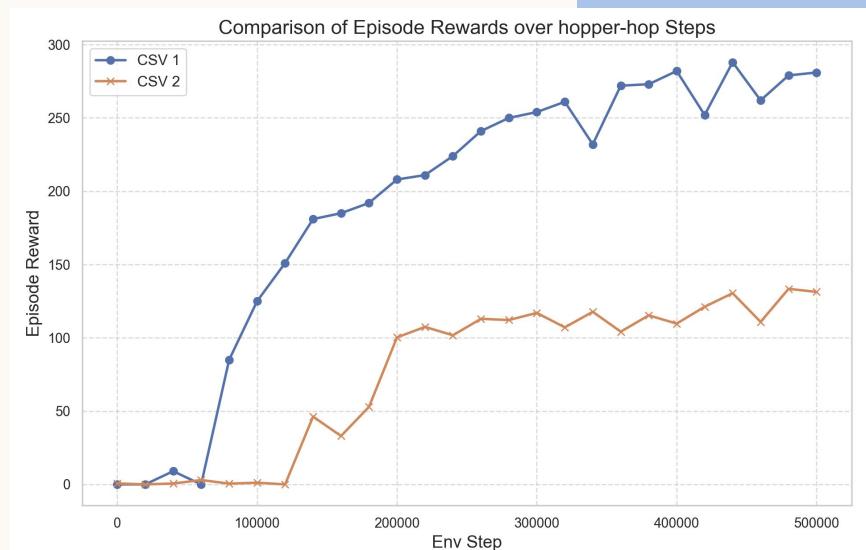
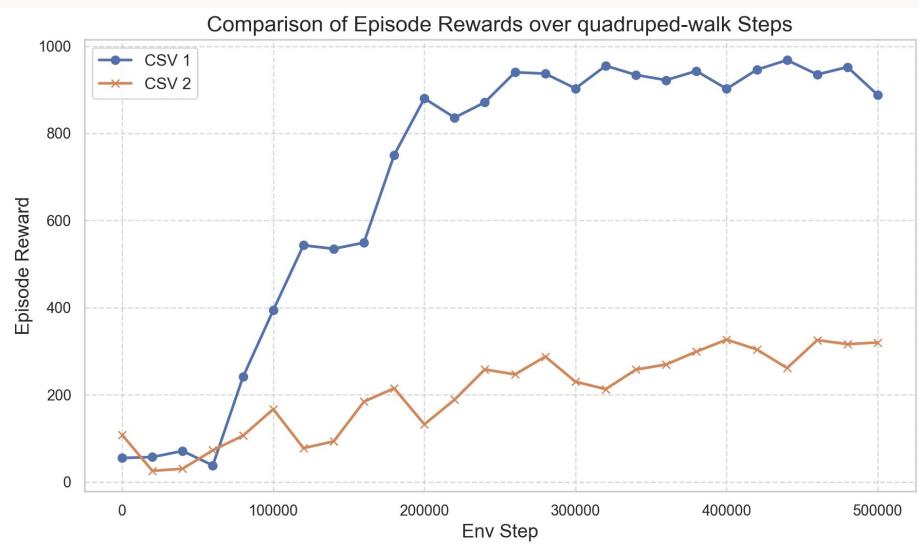
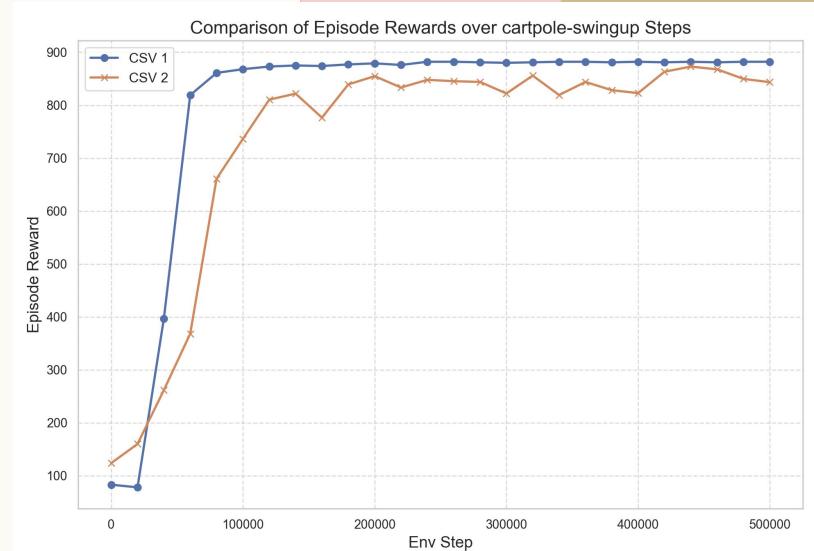
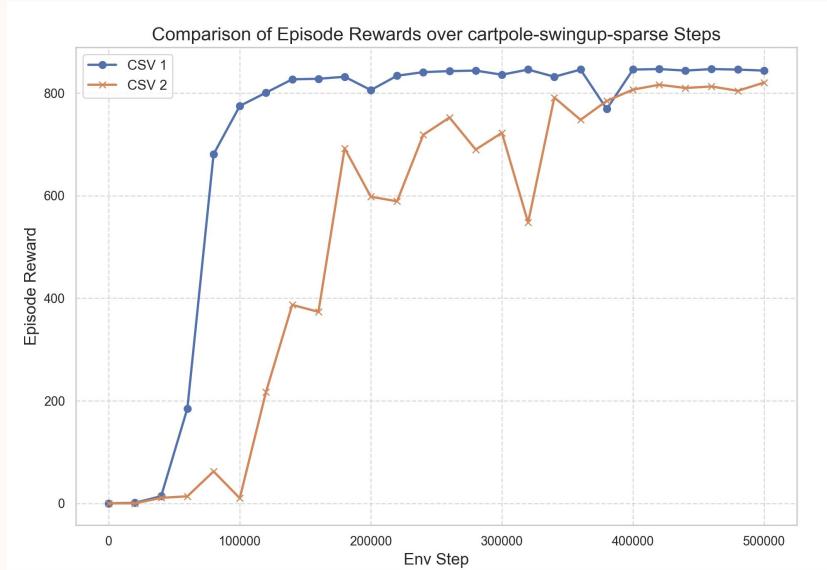
30



Results

31

Blue for the
Article's code,
Orange for our
implementation



**THANK
YOU FOR YOUR
ATTENTION**