The Thirty-Ninth Annual Conference on
Neural Information Processing Systems

Real-DRL: Teach and Learn in Reality

Yanbing Mao[1,*], Yihao Cai[1,*], Lui Sha[2]

[1]Wayne State University

[2]University of Illinois Urbana-Champaign

*Indicates Equal Contribution

Presenter: Yihao Cai

Preliminary

Motivation

Runtime Safety for Deep Reinforcement Learning (DRL)

Autonomous Vehicles[1]

Unmanned Aircraft[2]

Quadruped Robots[3]

Humanoid Robots[4]

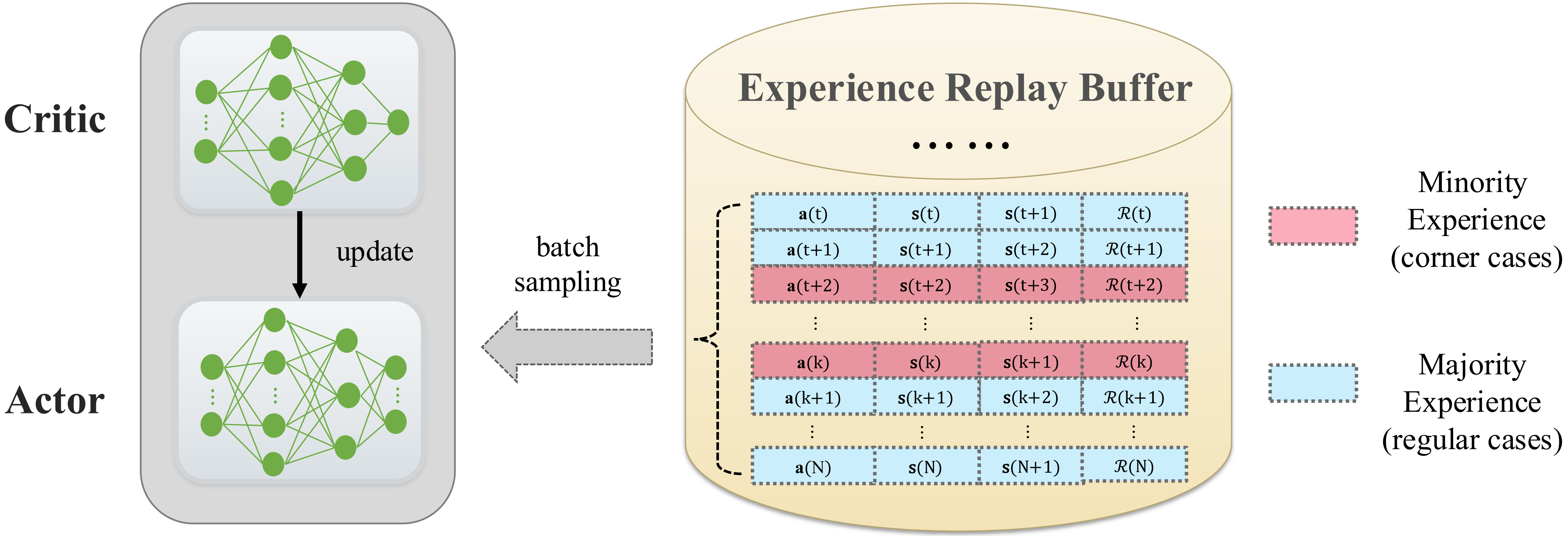
How do we ensure runtime safety in safety-critical autonomous systems while DRL agents perform online learning?

Reference:
[1] https://www.wired.com/story/dashcam-footage-shows-driverless-cars-cruise-waymo-clogging-san-Francisco/
[2] https://flyfrompti.com/unmanned-aircraft-systems-uas-drones/
[3] https://droneblocks.io/product/go2-edu-quadruped-robot/?srsltid=AfmBOoqbUHBaaWUpBTC0kkCZOT4tc_DKzTiHbY6uM4-DF36bHmMejDqA
[4] https://manlybattery.com/guide-to-leading-humanoid-robots/?srsltid=AfmBOoo1P5Dza-0L1jEdroApnsv2Um_yD2Wxozw_w1V-tYzqF2XObhkJ

Motivation

NEURAL INFORMATION PROCESSING SYSTEMS

Data Imbalance Issue from Sampling



How can the challenges of data imbalance be tackled to achieve more robust and generalizable DRL policies?

Challenges

NEURAL INFORMATION PROCESSING SYSTEMS

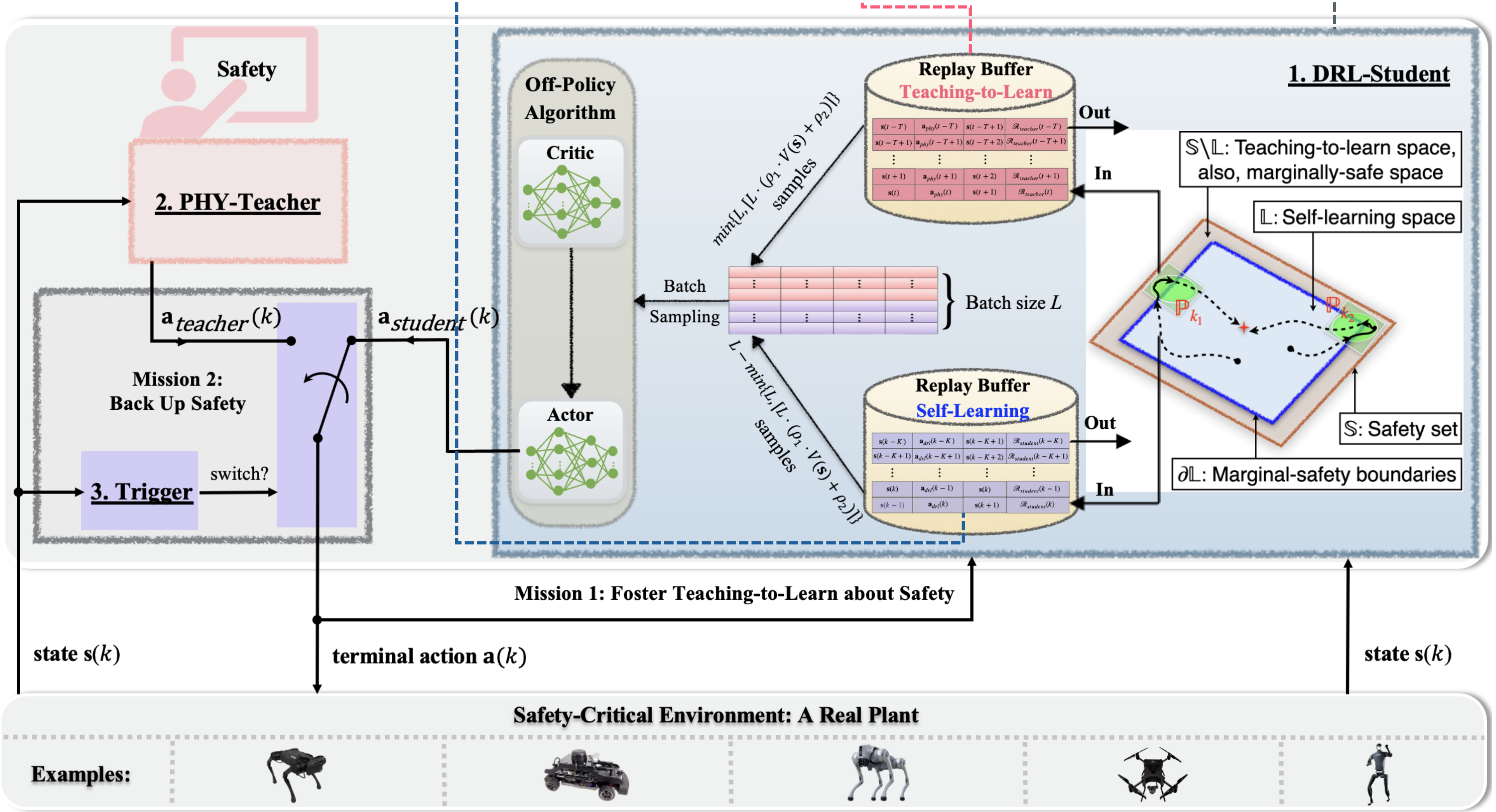
- **Runtime Learning Safety**
  - The risky nature of trial-and-error exploration in DRL
  - Learning in hard-to-predict and hard-to-simulate environments requires timely and adaptive responses
- **Safety-related Data Imbalance Issue**
  - Underrepresentation of rare but crucial data → poor safety at critical moments
  - Leading to training bias and limited generalization capability
- **Sampling Efficiency**
  - High-quality data fosters efficient and safe learning
  - Inefficient sampling prolongs training, and increases runtime safety risks

Proposed Solution

Proposed Framework

= Real-world Data + Safety-critical Data + Novel RL Architecture

NEURAL INFORMATION PROCESSING SYSTEMS



Component 1: DRL-Student

1. Dual buffer for self-learning and teaching-to-learn paradigm
2. Safety-informed batch sampling

Component 2: PHY-Teacher

1. Fostering the teaching-to-learn mechanism regarding safety
2. Safety backup for the real plants

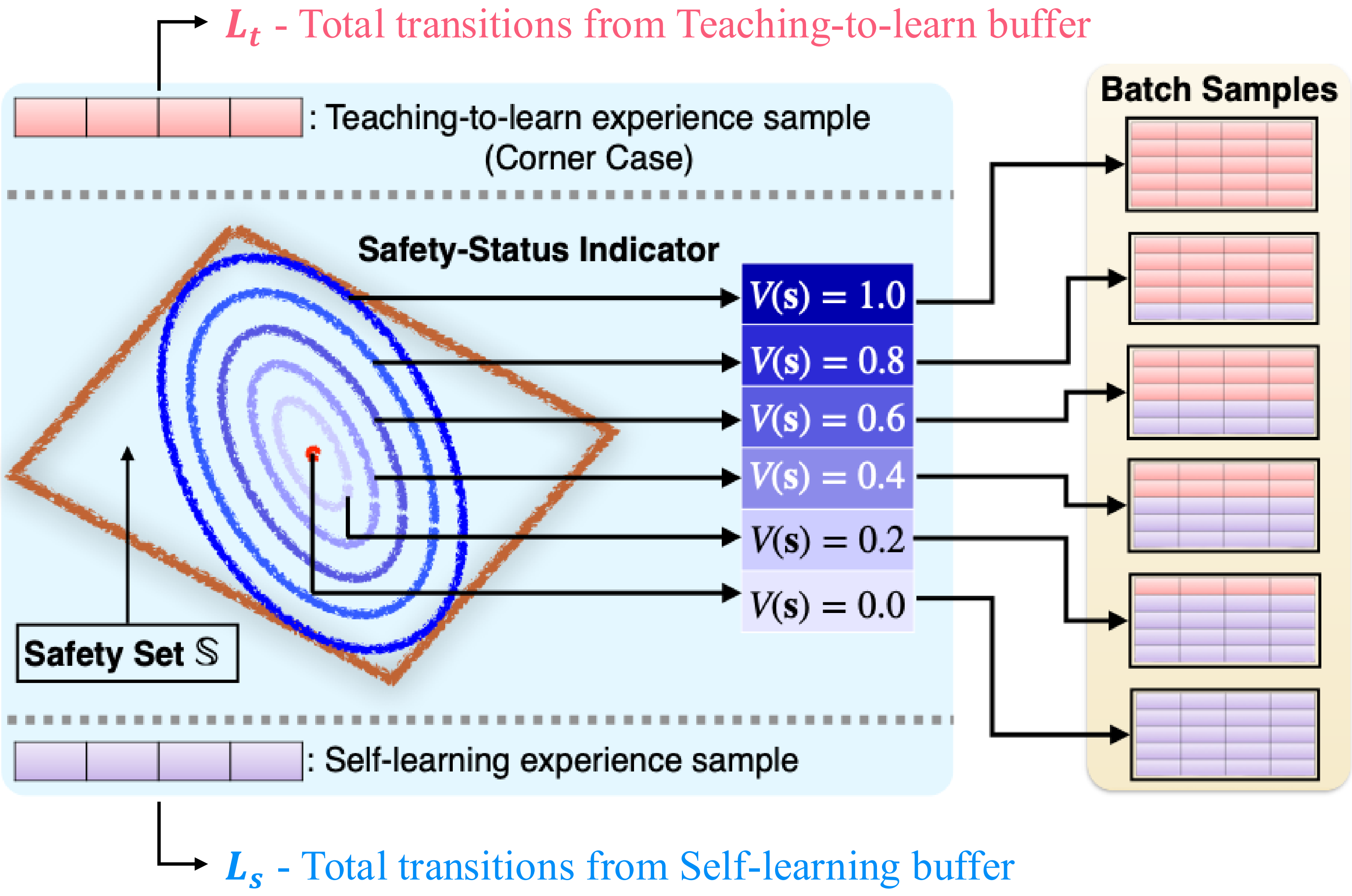
Component 3: Trigger

Monitoring the real-time safety status of the physical plant, and also deciding the terminal action to the plant

Component-I: DRL-Student

NEURAL INFORMATION PROCESSING SYSTEMS

Safety-informed Batch Sampling



Safety-status indicator

$$V(\mathbf{s}) \triangleq \mathbf{s}^T \cdot \mathbf{P} \cdot \mathbf{s}$$

$\mathbf{s}$ – real time state $\mathbf{s}(\mathrm{t})$

Total Sampled Batch Size

$$L = L_t + L_s$$

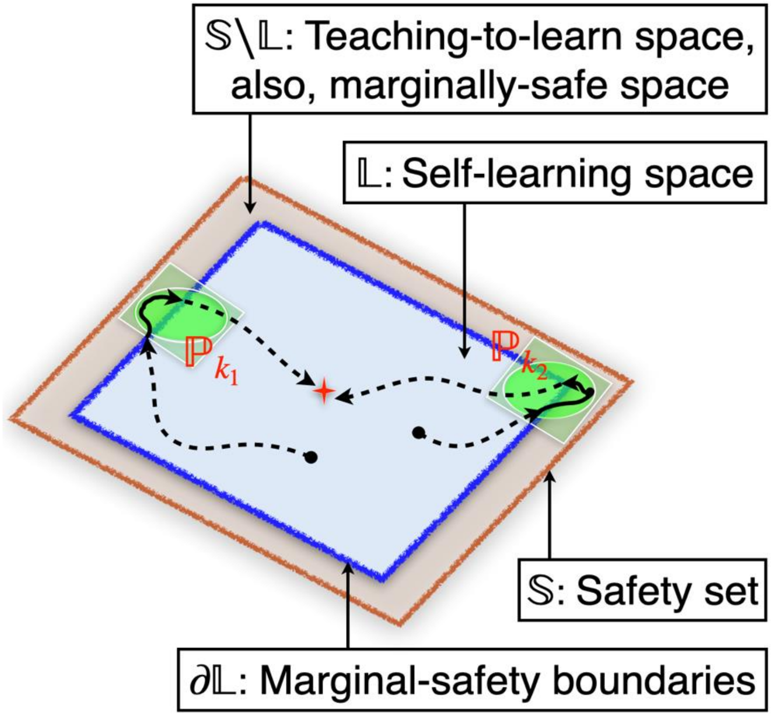
$$L_t = \min\{L, \lceil L \cdot (\rho_1 \cdot V(\mathbf{s}(\mathrm{t})) + \rho_2 \rceil\}$$

$$L_s = L - \min\{L, \lceil L \cdot (\rho_1 \cdot V(\mathbf{s}(\mathrm{t})) + \rho_2 \rceil\}$$

ρ_1, ρ_2 – hyperparameters

Component-II: PHY-Teacher

Real-time Patch Design



System Dynamics: $\boldsymbol{s}(t+1) = \mathrm{f}\,(\boldsymbol{s}(t), \mathbf{a}(t)), t \in \mathbb{N}$

Safety Set: $\mathbb{S} \triangleq \{\boldsymbol{s} \in \mathbb{R}^n | -\mathbf{c} < \mathbf{C} \cdot \boldsymbol{s} < \mathbf{c}\}$

Self-Learning Space: $\mathbb{L} \triangleq \{\boldsymbol{s} \in \mathbb{R}^n | -\eta \cdot \mathbf{c} < \mathbf{C} \cdot \boldsymbol{s} < \eta \cdot \mathbf{c}\}, 0 < \eta < 1$

Real-time Patch Design $\mathbb{P}_k$

① **Control Goal** : $\boldsymbol{s}_k^* \triangleq \chi \cdot \boldsymbol{s}(k)$, $\boldsymbol{s}(k\text{-}1) \in \mathbb{L}$ and $\boldsymbol{s}(k) \in \partial\mathbb{L}$

② **LMI Feasibility** : Construct LMIs and optimize for $\boldsymbol{F}_k$

③ **Action Policy** : $\mathbf{a}_{teacher}(t) = \boldsymbol{F}_k \cdot (\boldsymbol{s}(\mathrm{t}) - \boldsymbol{s}_k^*)$, $t \in \mathbb{T}_k$

Component-III: Trigger

Triggering Condition $\mathcal{T}$: $\boldsymbol{s}(k\text{-}1) \in \mathbb{L}$ and $\mathbf{s}(k) \in \partial\mathbb{L}$

PHY-Teacher Action Step: $\mathbb{T}_k \triangleq \{\mathrm{k}+1, \mathrm{k}+2, \ldots, \mathrm{k}+\tau_k\}$

Switching Law

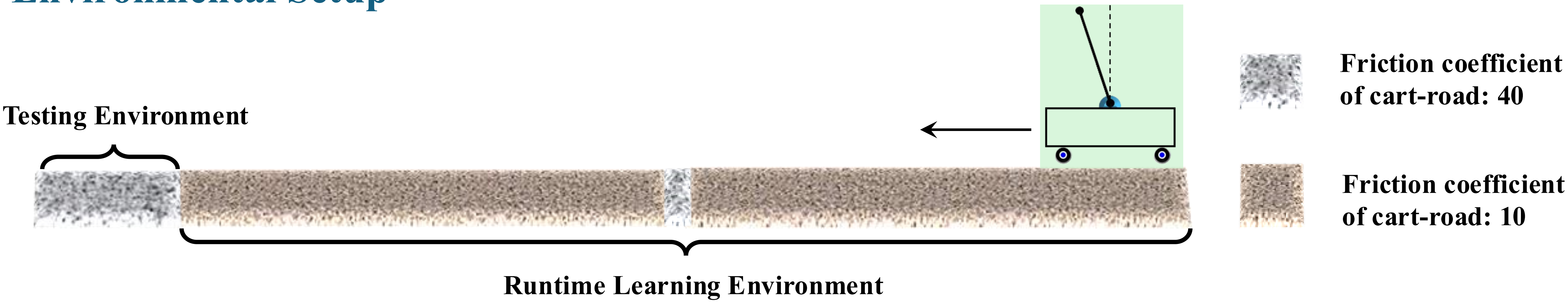
$$\mathbf{a}(t) = \begin{cases} \mathbf{a}_{teacher}(t), & \text{if } \mathcal{T} \text{ holds at } k, \text{ and } t \in \mathbb{T}_k \\ \mathbf{a}_{student}(t), & \text{if } \boldsymbol{s}(k) \in \mathbb{L} \end{cases}$$

Experiment

Experiment-I: Cartpole

NEURAL INFORMATION PROCESSING SYSTEMS

Environmental Setup



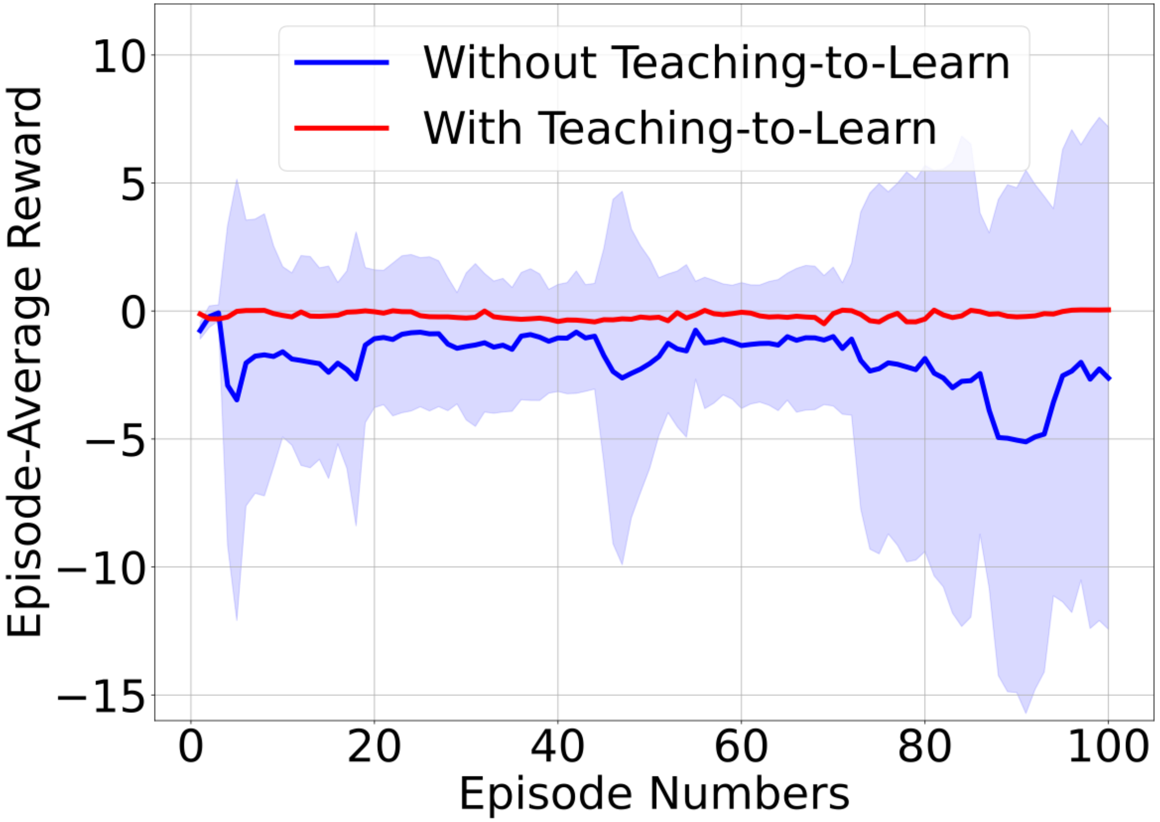
Ablation Study – Demonstrating Three Key Features of Real-DRL

Three Key Features

- **Feature I:** Teaching-to-Learn Mechanism
- **Feature II:** Safety-informed Batch Sampling
- **Feature III:** Automatic Hierarchical Learning

Experiment-I: Cartpole

Feature I: Teaching-to-Learn mechanism



Episode-Average Reward

Episode-Average Reward:

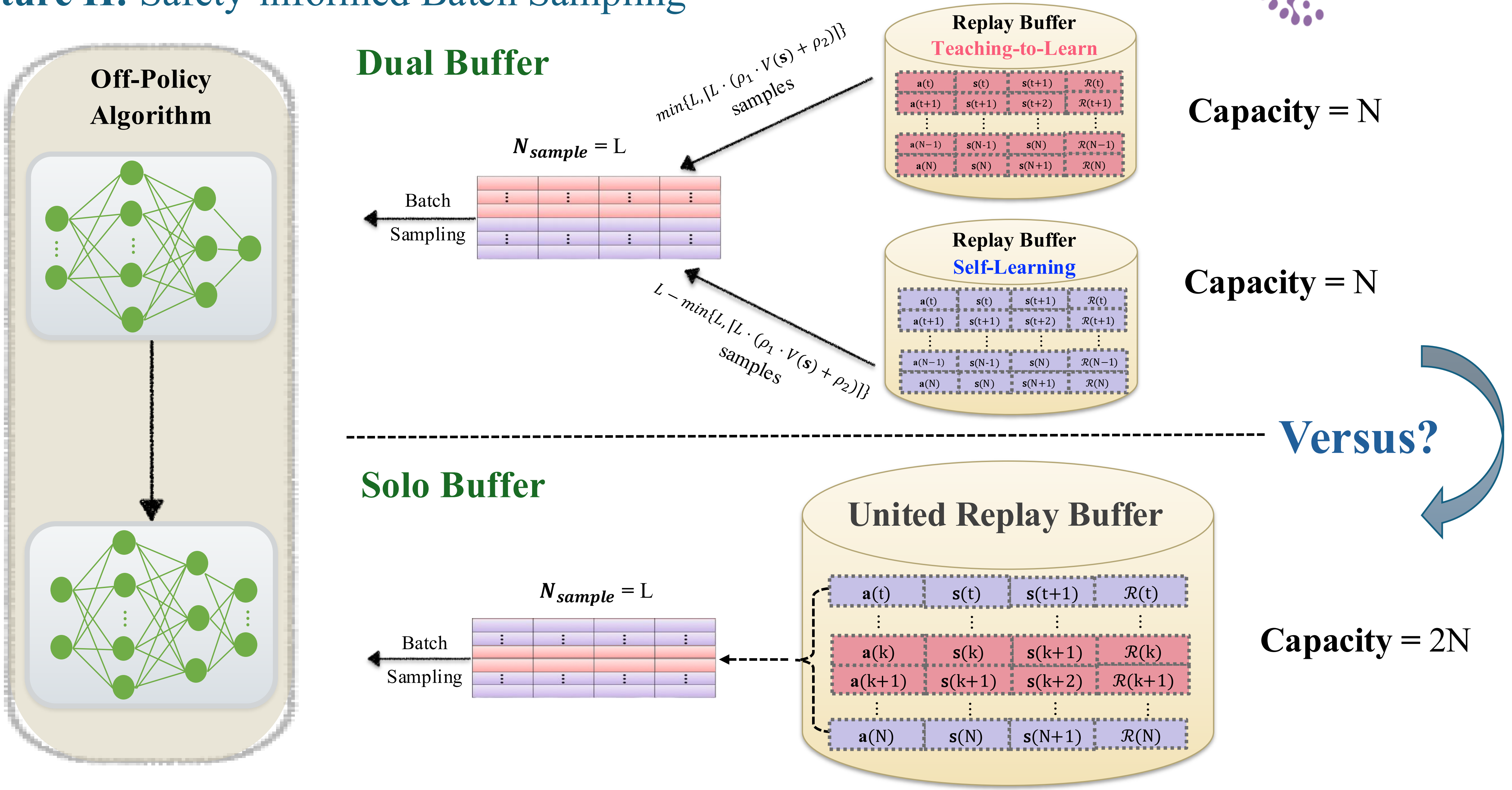
$$\frac{\text{Return (i.e., cumulative reward) in one episode}}{\text{DRL-Student's total activation time in one episode}}$$

Adopting Teaching-to-Learn paradigm leads to overall improved episode-average reward and stable learning

Experiment-I: Cartpole

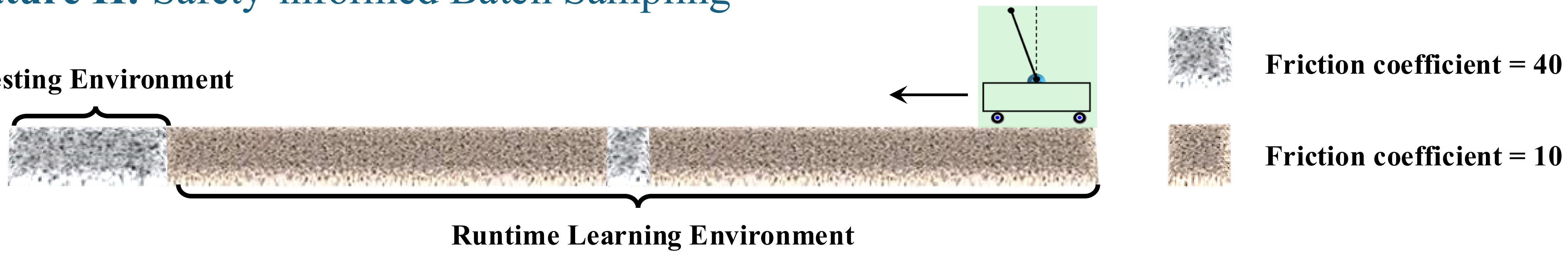
NEURAL INFORMATION PROCESSING SYSTEMS

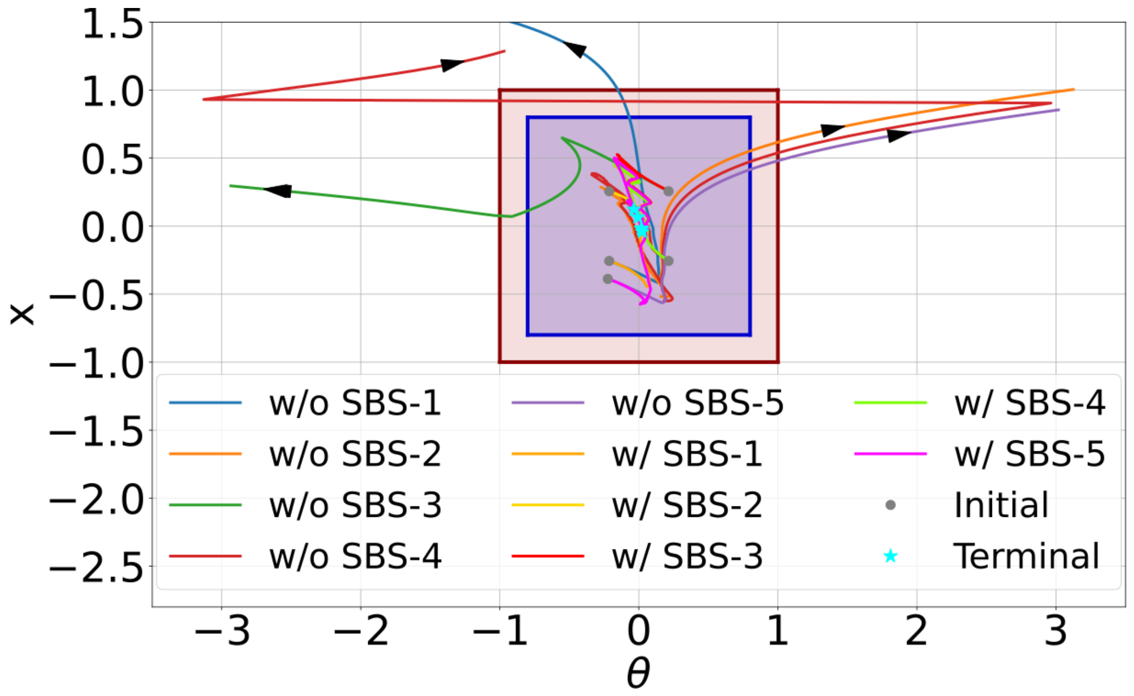
Feature II: Safety-informed Batch Sampling



Experiment-I: Cartpole

Feature II: Safety-informed Batch Sampling





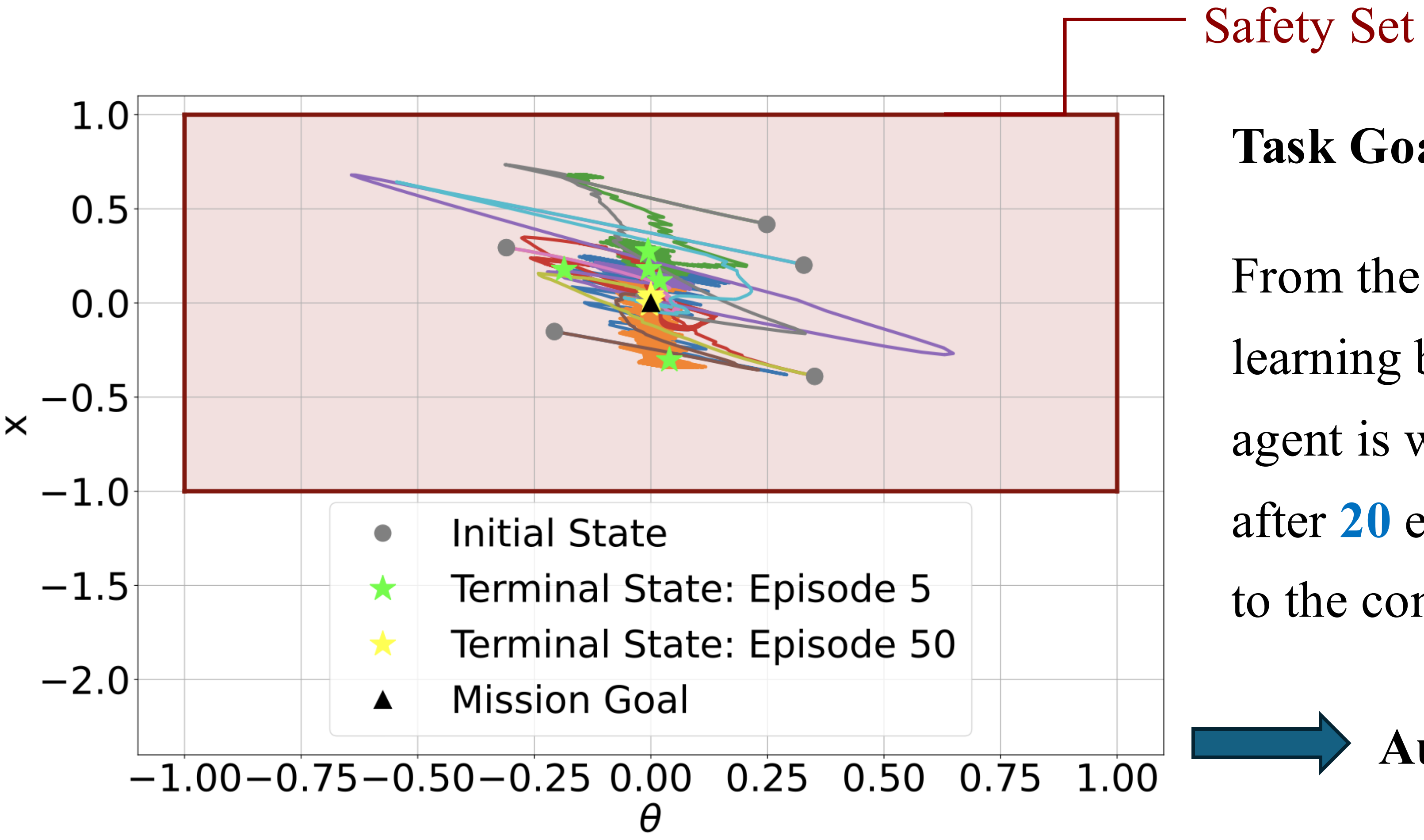
Phase Plot (with vs. without safety-informed sampling)

1. Agent built on Real-DRL maintains safety on **both** cases after runtime learning

2. Agent sampling from a united replay buffer maintains safety in the **majority** cases but failed on **corner** cases

Experiment-I: Cartpole

Feature III: Automatic Hierarchical Learning



Agent Trajectory from Different Episodes (Inference)

Task Goal: $(\bar{x}^*, \bar{\theta}^*) = (0, 0)$

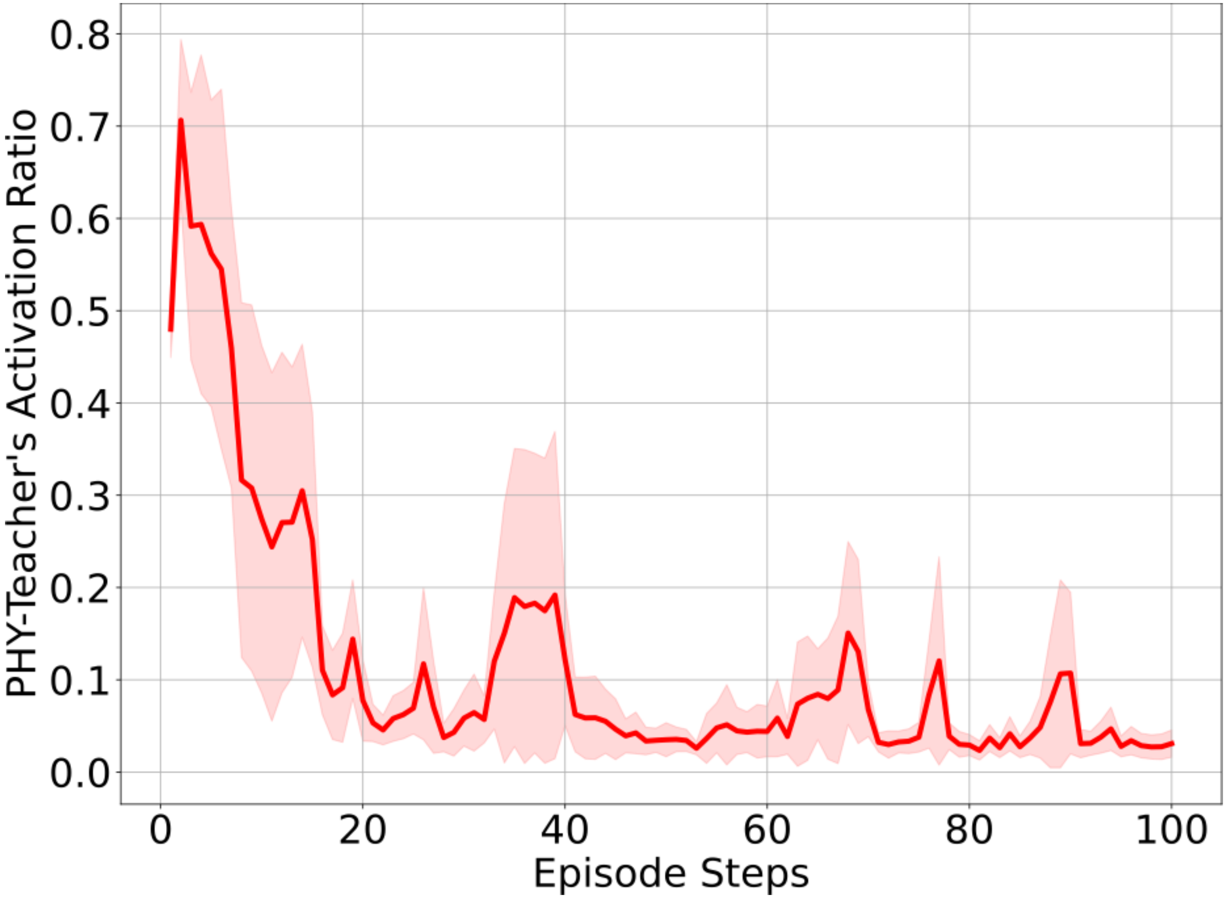
From the same initial state, after **5** episodes learning by Real-DRL, the trajectory of the agent is within the safety set (**safety-first**); after **20** episodes, the trajectory gets closer to the control goal (**high-performance**)

Automatic Hierarchical Learning:

Safety-first → **High-Performance**

Experiment-I: Cartpole

Feature III: Automatic Hierarchical Learning



PHY-Teacher Activation Ratio

PHY-Teacher Activation Ratio:

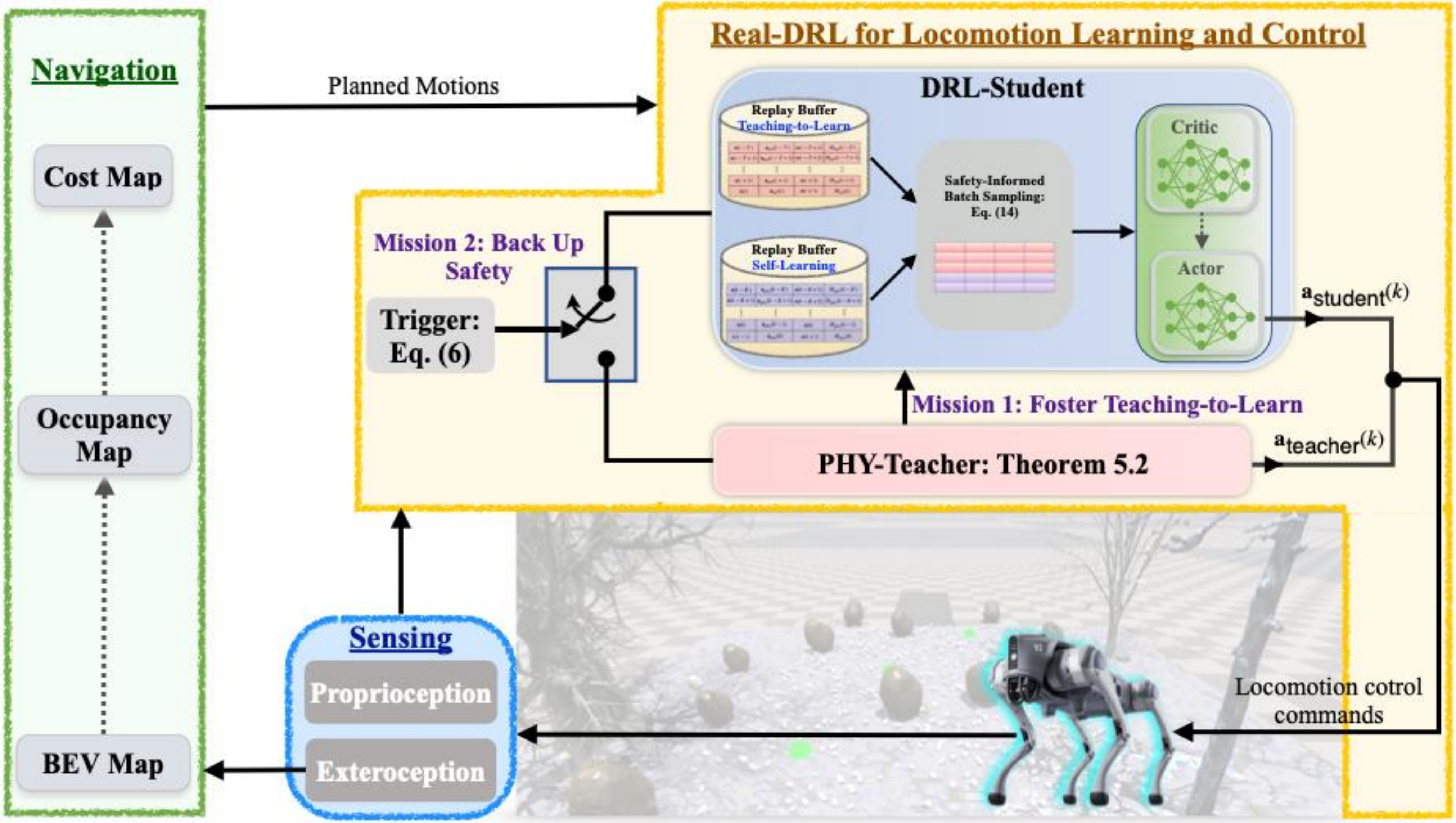
$$\frac{\text{PHY-Teacher's total activation times in one episode}}{\text{one episode length}}$$

The activation ratio of PHY-Teacher within an episode decreases over time

DRL-Student becomes independent of PHY-Teacher as learning evolves

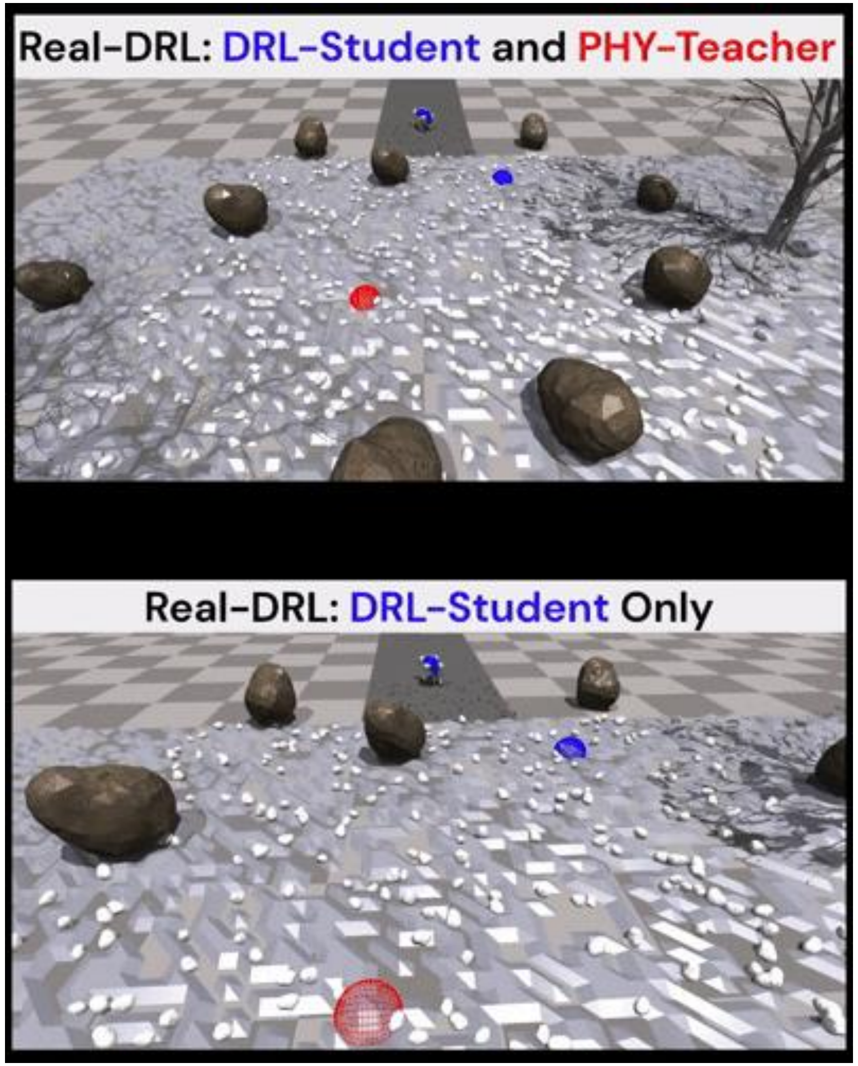
Experiment-II: Go2 in IsaacGym

Architecture built on Real-DRL

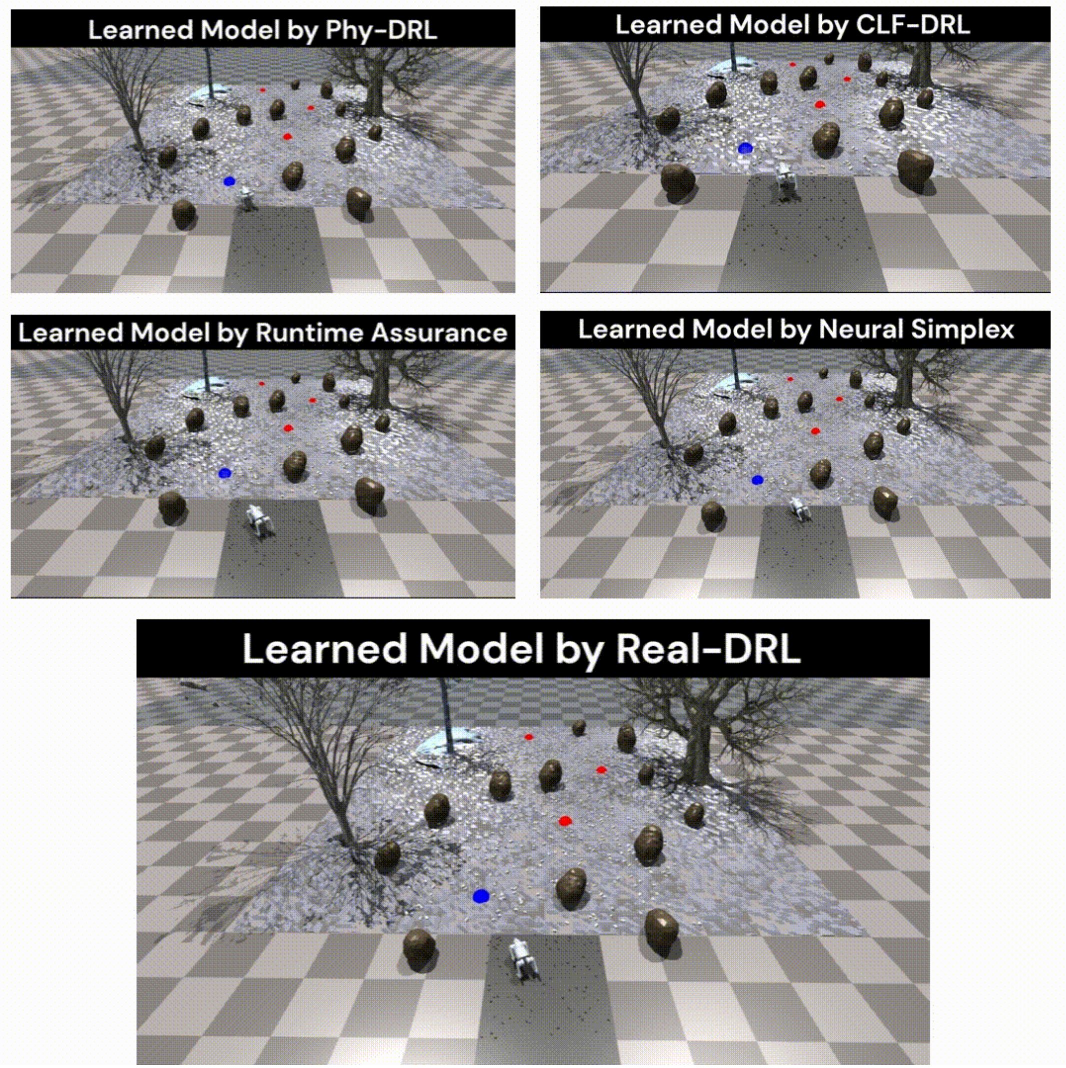


Experiment-II: Go2 in IsaacGym

Evaluation Result



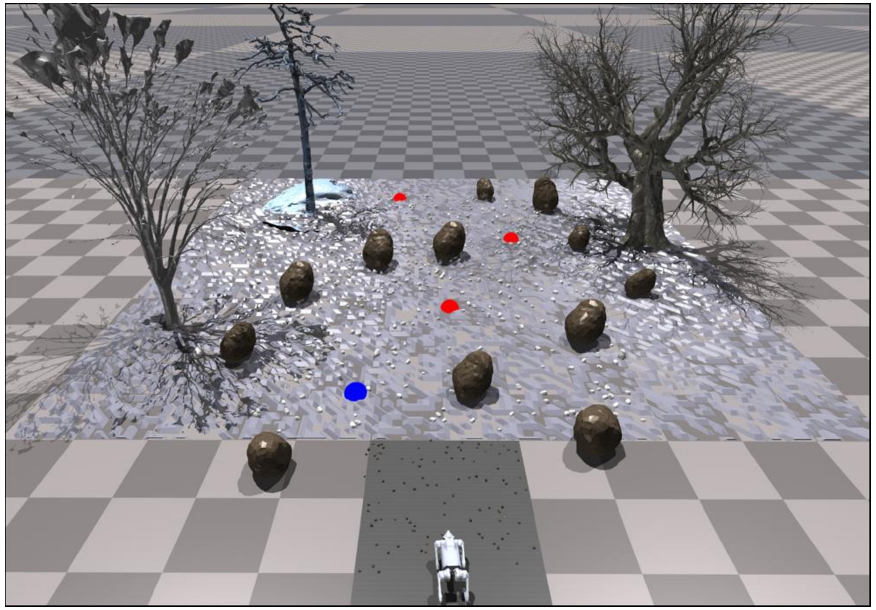
Real-DRL in safety guarantee

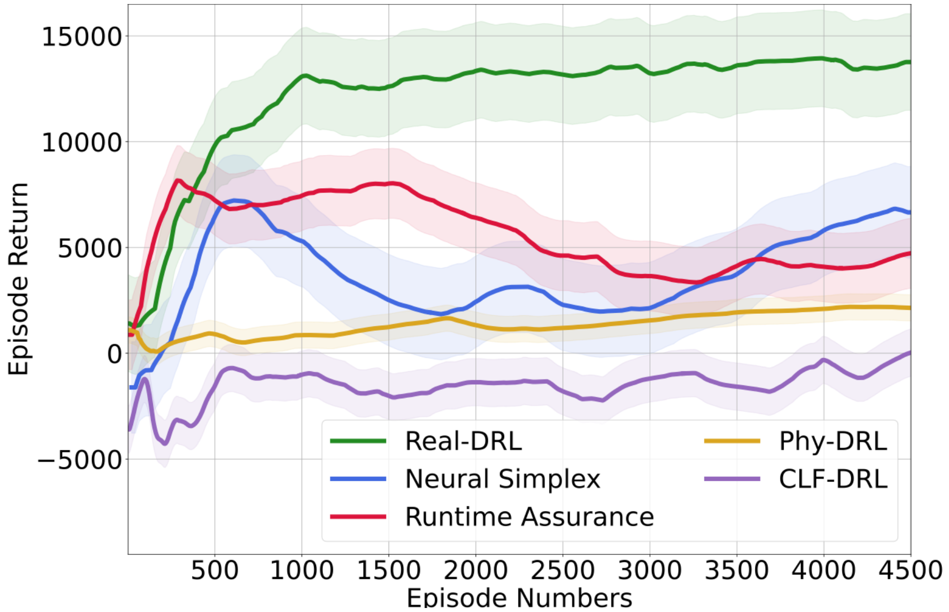


Real-DRL in learning high-performance policy

Experiment-II: Go2 in IsaacGym

Comparison with SOTA

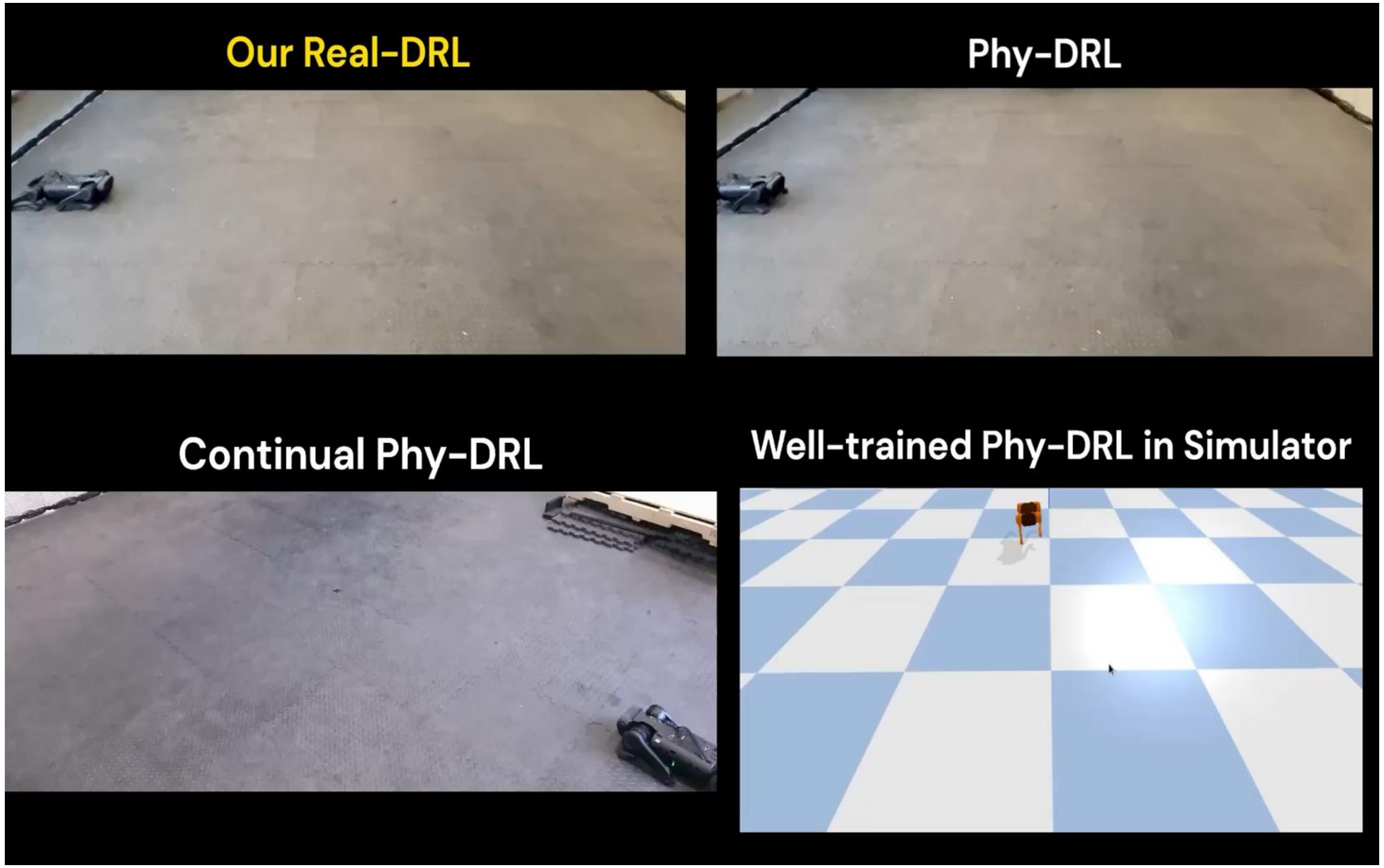




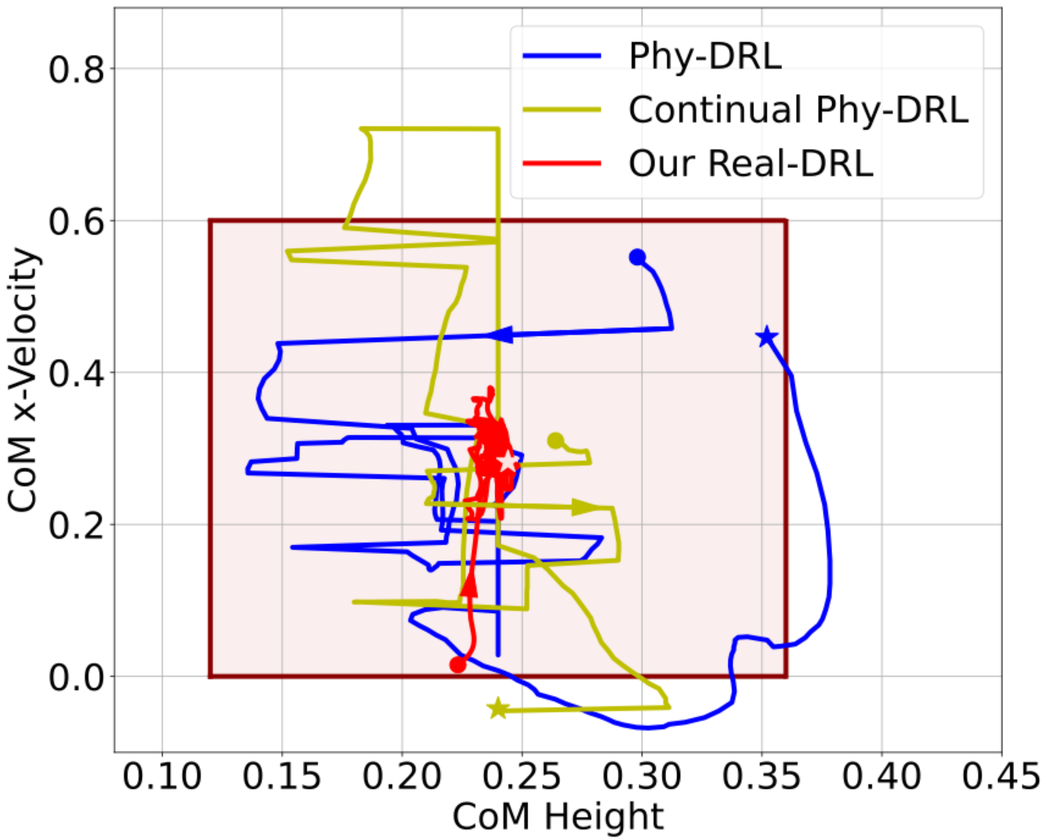
Model ID	Navigation Performance					Energy Efficiency	
	Success	Is Fall	Collision	Num (wp)	Travel Time (s)	Avg Power (W)	Total Energy (J)
CLF-DRL	No	Yes	No	0	N/A	N/A	N/A
Phy-DRL	No	No	Yes	1	∞	507.9441	∞
Runtime Assurance	No	Yes	No	2	N/A	N/A	N/A
Neural Simplex	No	No	Yes	2	∞	487.9316	∞
PHY-Teacher	**Yes**	No	No	**4**	55.5327	482.8468	26817.68
Our Real-DRL	**Yes**	No	No	**4**	**45.3383**	**479.4638**	**21742.42**

Experiment-III: A1 in Real World

Sim2Real using Real-DRL



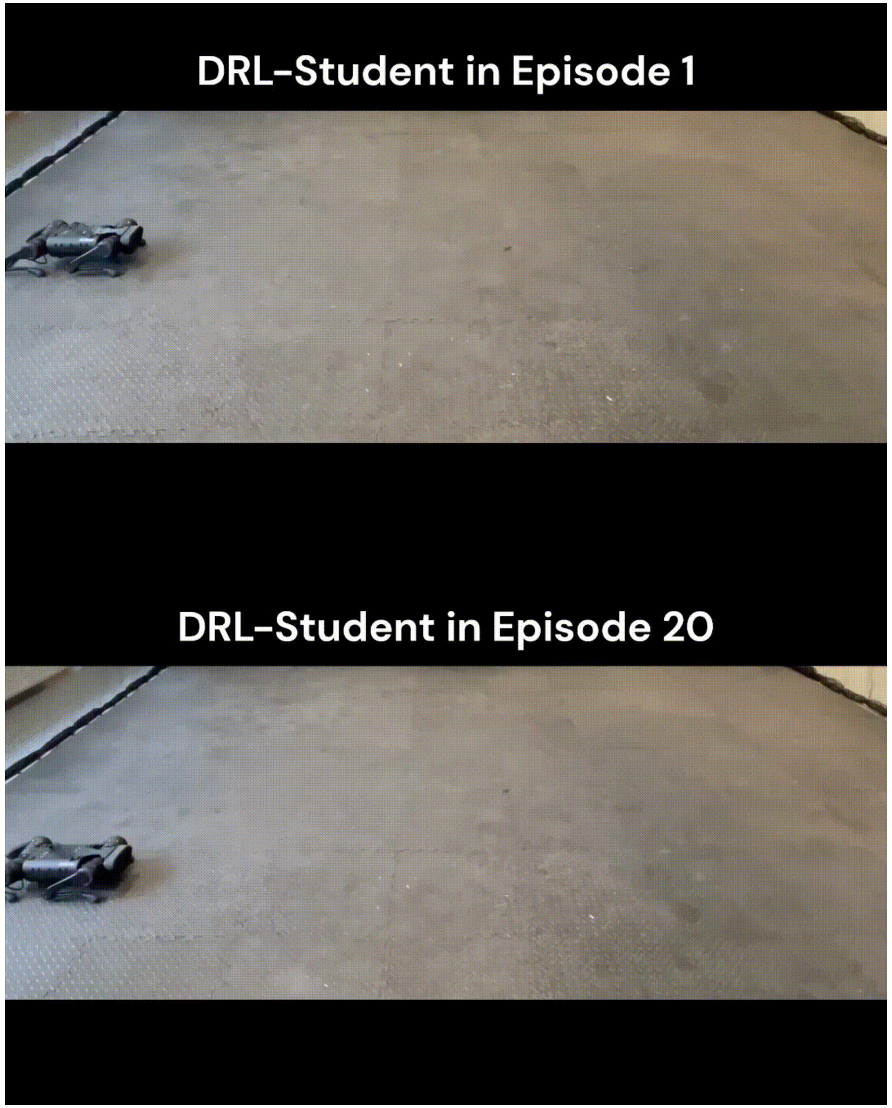
Effectiveness of Real-DRL in Sim2Real



Robot Dog Phase Plot

Experiment-III: A1 in Real World

Real-DRL Fosters Safety-first Learning



Learning in the first Episode (early stage)

Safe learning with Real-DRL after 20 episodes

Summary

Conclusion

NEURAL INFORMATION PROCESSING SYSTEMS

- ***Core Contribution***
  - Real data collection from the hard-to-predict environment
  - Good data (regarding safety) generation from a verifiable PHY-Teacher
  - An innovative RL architecture that supports modular design

- ***Three Notable Features***
  - Teaching-to-learn Mechanism (e.g., foster safe learning and fast convergence)
  - Automatic Hierarchy Learning (e.g., learn safety first and high-performance policy)
  - Safety-informed batch sampling (e.g., resolve data imbalance caused by corner cases)

- ***Soundness and Generality***
  - The framework is evaluated across a variety of autonomous systems
  - The experiments incorporate both simulation and real-world evaluations
  - The design of PHY-Teacher provides a theoretical proof of soundness

Miscellaneous

NEURAL INFORMATION PROCESSING SYSTEMS

ECVXCONE – A Toolbox Towards Real-DRL on Edge Devices

Cross-Platform and **Runtime-Efficient** Conic Optimization Toolbox for LMIs

Hardware Platforms	CPU Configurations			Runtime Memory Usage		LMIs Solve Time	
	Arch	Core	Frequency	CVXPY	ECVXCONE	CVXPY	ECVXCONE
Dell XPS 8960 Desktop	x86/64	32	5.4 GHz	485 MB	9.87 MB	49.15 ms	13.81 ms
Intel GEEKOM XT 13 Pro Mini	x86/64	20	4.7 GHz	443 MB	7.32 MB	61.76 ms	33.26 ms
NVIDIA Jetson AGX Orin	ARM64	12	2.2 GHz	423 MB	8.16 MB	137.54 ms	35.73 ms
Raspberry Pi 4 Model B	ARM64	4	1.5 GHz	436 MB	8.21 MB	509.41 ms	149.87 ms

Python **CVXPY** vs **C ECVXCONE** (Computational Overhead)

NEURAL INFORMATION PROCESSING SYSTEMS

The Thirty-Ninth Annual Conference on Neural Information Processing Systems

Thank You for the Attention

Scan To Know More

WAYNE STATE UNIVERSITY

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

NVIDIA

NSF