Mission-driven Exploration for Accelerated Deep Reinforcement Learning with Temporal Logic Task Specifications

QR Code for Code Videos

spec-driven-rl.github.io

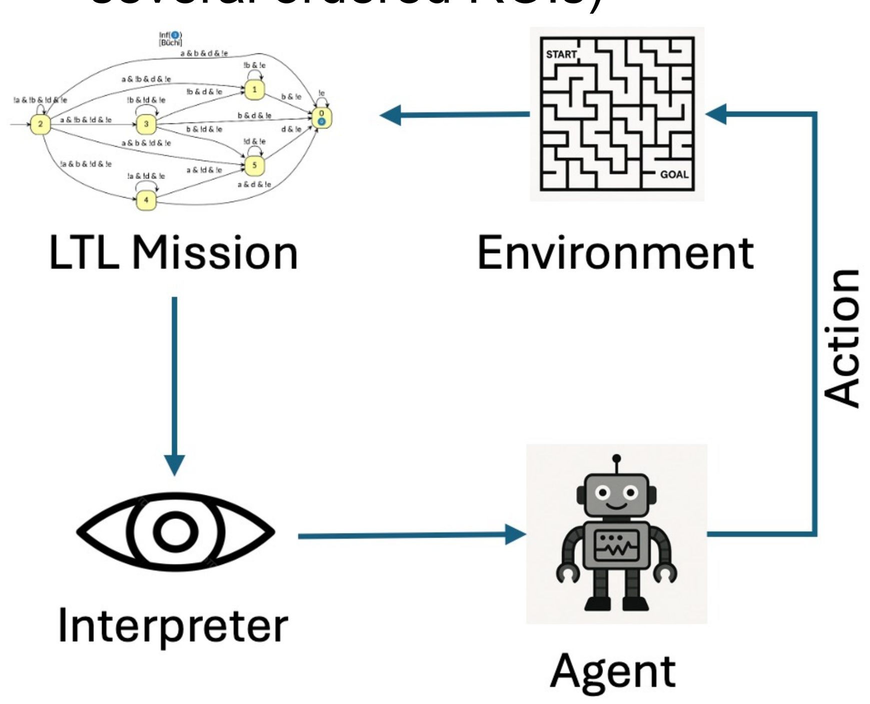
Jun Wang[1], Hosein Hasanbeig[2], Kaiyuan Tan[3], Zihe Sun[1], and Yiannis Kantaros[1]
Preston M. Green Department of Electrical & Systems Engineering, Washington University in St Louis[1]
Microsoft Research[2]
Department of Computer Science, Vanderbilt University[3]

L4DC 2025

Keywords: Reinforcement Learning, Temporal Logic Planning, Sample Efficiency

Motivation

- Linear Temporal Logic (LTL) has been used to encode complex tasks (e.g., navigation task with several ordered ROIs)



- **Model-free** DRL methods require a product state space that grows exponentially, result in *slow learning process*
- **Model-based** RL methods rely on learned MDPs but are limited to *discrete state spaces*

Goal

Design a ***sample-efficient*** DRL algorithm to learn control policies for agents with LTL-encoded tasks

Contribution

- Propose a new *DQN* algorithm for agents with *unknown MDPs* , *continuous state spaces* and LTL-encoded tasks
- Our policy is *complementary* with existing deep temporal-difference methods for LTL tasks to enhance sample efficiency
- Present comparative numerical and hardware experiments that demonstrate the *sample efficiency* of our method

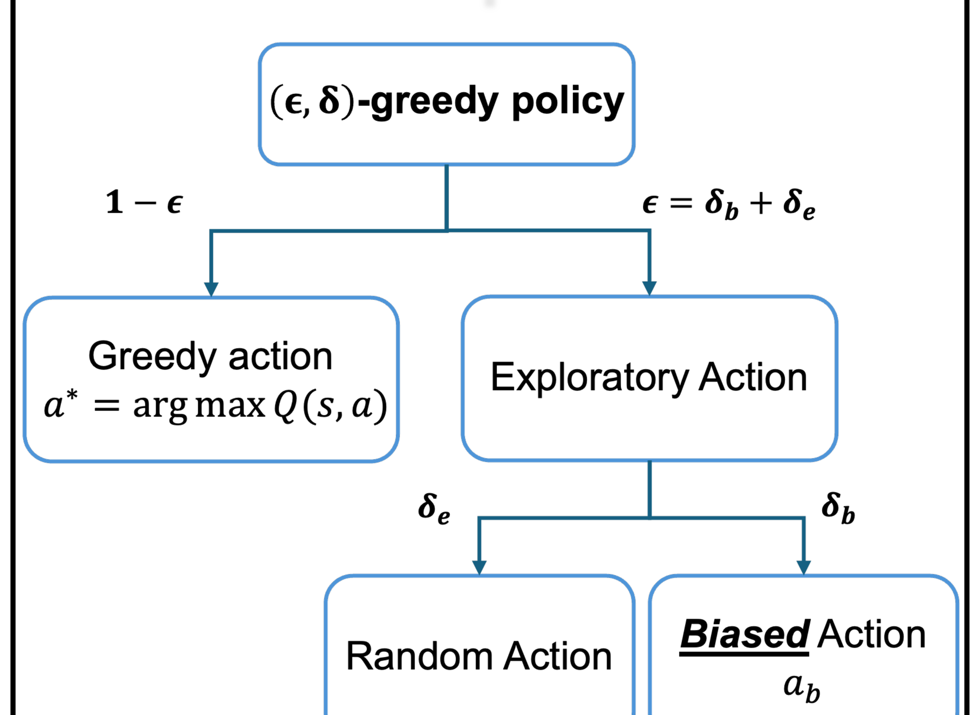
Contact Email: junw@wustl.edu

Problem Formulation

- Environment $\mathcal{W} \subseteq R^d, d \in \{2,3\}$
- LTL over set of $\mathcal{AP}$: $\phi = true \mid \pi \mid \phi_1 \wedge \phi_2 \mid \neg\phi \mid \circ\, \phi \mid \phi_1 \mathcal{U}\, \phi_2$
- Fully observable MDP: $\mathfrak{M} = (\mathcal{X}, \mathcal{A}, P, \mathcal{AP})$ with continuous state space $x \in \mathcal{X}$ and a finite set of actions $a \in \mathcal{A}$ (*transition probabilities unknown*)
- ***Problem 1***: Given a known LTL-encoded task specification ϕ, develop a ***sample-efficient*** DRL method that can synthesize a finite memory control policy ξ^* for the unknown MDP that ***maximizes satisfaction probability*** of ϕ

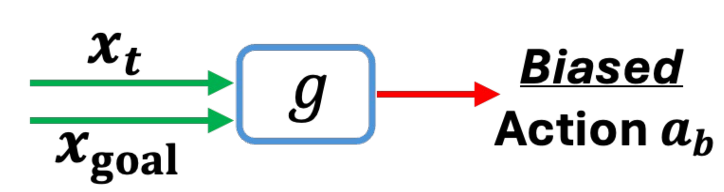
Methods

- Translate ϕ into DRA $\mathfrak{D} = (\mathcal{Q}_\mathfrak{D}, q_D^0, \Sigma, \delta_D, \mathcal{F})$ with the set of accepting states $\mathcal{F}$, and compute product MDP $\mathfrak{P} = \mathfrak{M} \times \mathfrak{D}$
- For generalization, we leverage features related to agent state (e.g., *distances to obstacles*)
- Our DQN algorithm (**(ϵ, δ)-greedy**) produces a policy μ^* for the PMDP $\mathfrak{P}$; Projection of μ^* onto MDP $\mathfrak{M}$ yields ξ^* (Problem 1)



Methods (Biased Action)

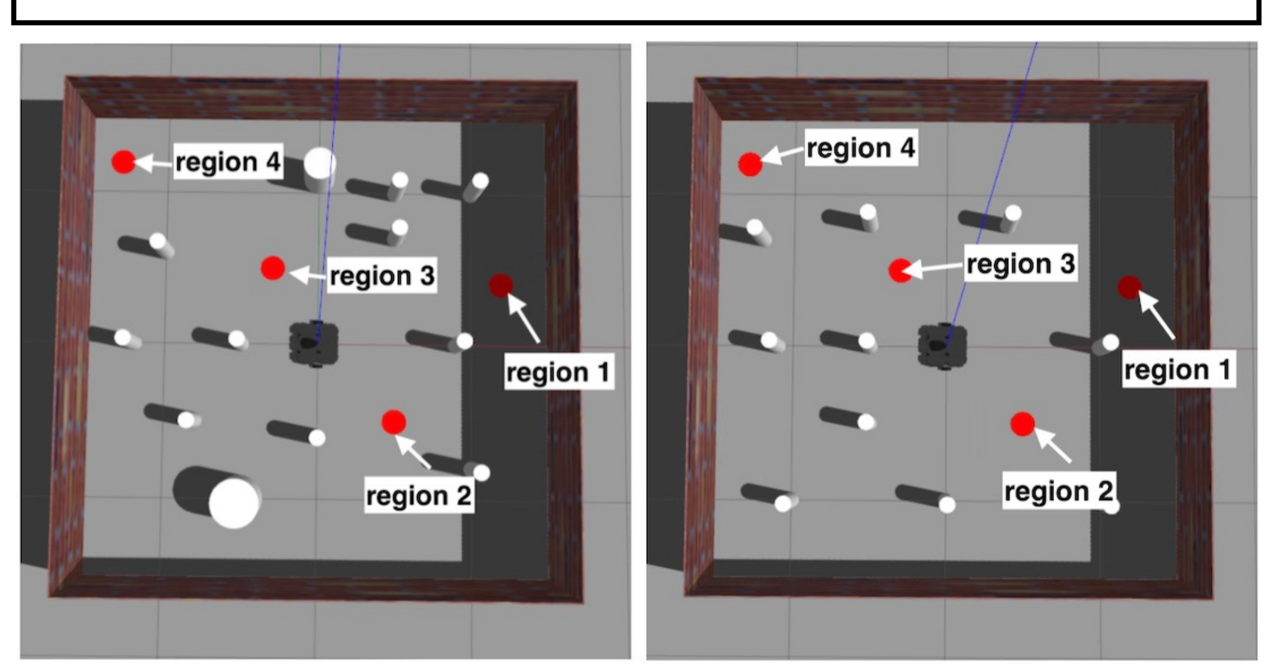
- Compute set $\mathcal{Q}_{\text{goal}}(q_D^t)$ that is one-hop reachable from q_D^t and closer to accepting DRA states $\mathcal{F}$, and collect a set $\mathcal{X}_{\text{goal}}(\mathcal{Q}_D^t)$ of 'goal' MDP states and randomly select one as x_{goal}
- A ***biased*** action can drive **closer** to goal state
- **Train a NN model g to compute the *biased* action** *prior to* RL training



- How to Collect Training Dataset:
- (i) Environment discretization (into grids) with goal states placed at center of each cell
- (ii) Randomly sample a set $\mathcal{X}_{\text{start}}$, assign each state to the nearest discrete cell i
- (iii) Compute weighted Dijkstra distances to all goal cells, simulate Z times each for each action and compute the optimal action a_b to reach goal
- (iv) Collect a dataset of $\left(x_{\text{start}}, x_{\text{goal}}, a_b\right)$

Experiments

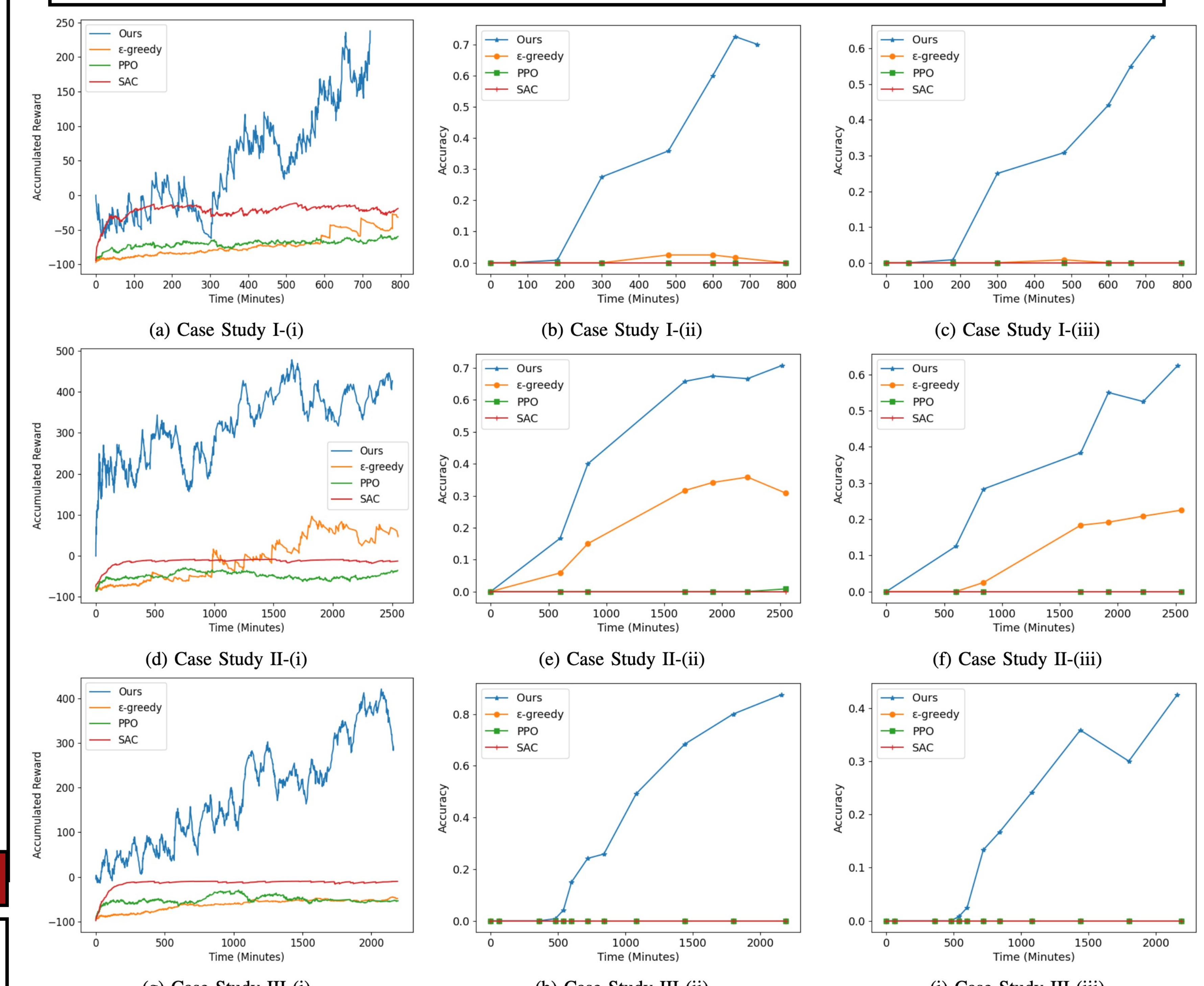
- Unknown robot dynamics with states $x_t = [p_t^1, p_t^2, \theta_t]$ with $|\mathcal{A}| = 23$ actions of (u, ω), and additive Gaussian actuation noise
- **Baselines**: ϵ-greedy DQN | PPO | SAC



Training (Left) and Test (Right) Gazebo Envs

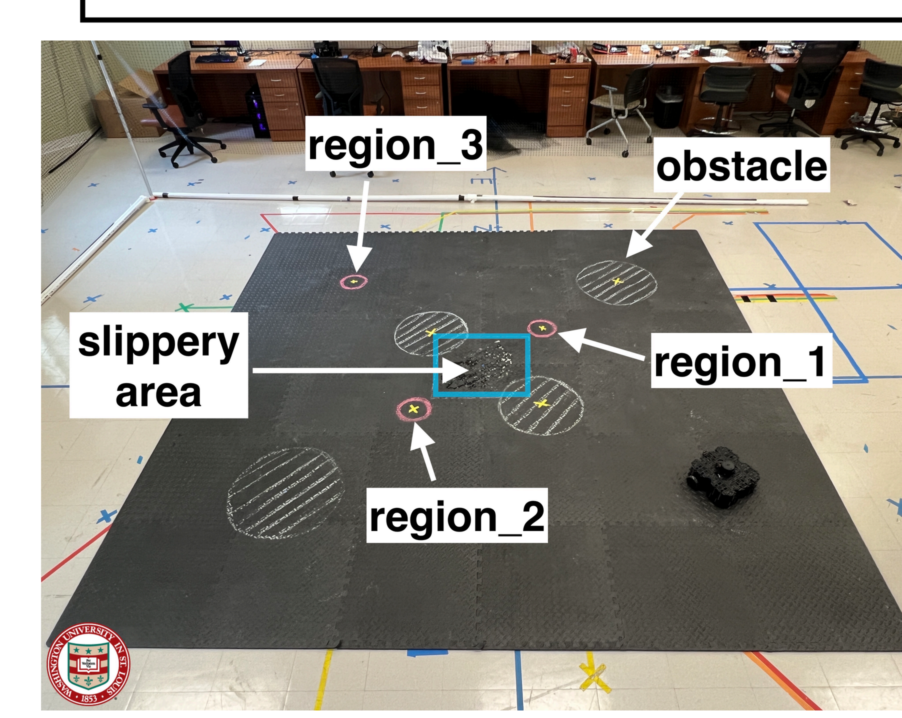
Experiments and Results

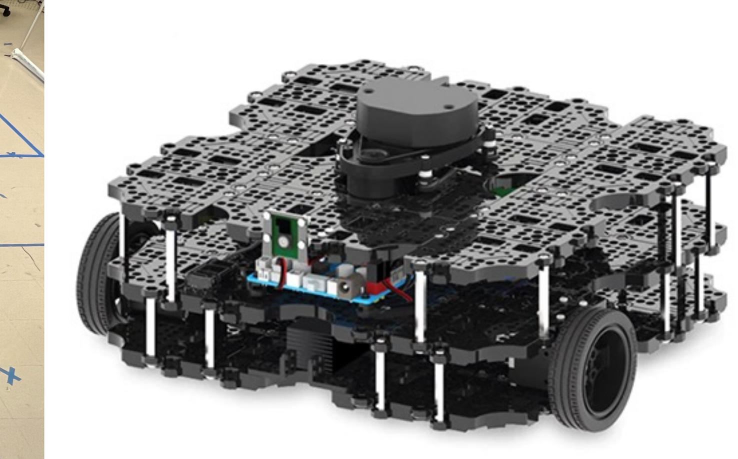
- Eval Metric: (i) Average return per episode; (ii) Accuracy on **training** environments; (iii) Accuracy on **test** environments
- We outperformed baselines in sample efficiency, especially when task or environment complexity increases.



(a) Case Study I-(i) (b) Case Study I-(ii) (c) Case Study I-(iii)
(d) Case Study II-(i) (e) Case Study II-(ii) (f) Case Study II-(iii)
(g) Case Study III-(i) (h) Case Study III-(ii) (i) Case Study III-(iii)

Ours (Blue), DQN (Orange), PPO(Green), SAC(Red)
Columns 1 ~ 3 plot metrics (i), (ii), and (iii), respectively.





Hardware Environment (Left) and Turtlebot Waffle Pi(Right)