Improving Safety in Deep RL Using Unsupervised Action Planning

Hao-Lun Hsu[1], Qiuhua Huang[2], and Sehoon Ha[1]

[1] Georgia Tech, [2] Pacific Northwest National Laboratory, USA

Paper **Video**

Georgia Institute of Technology

This work is a novel approach integrating on-policy RL algorithms with unsupervised action planning (*e.g.* k-means clustering). It enables RL agents to reduce number of failures during both training and testing, evaluated with both discrete and continuous control problems.

Motivation: *How to improve safety during learning*

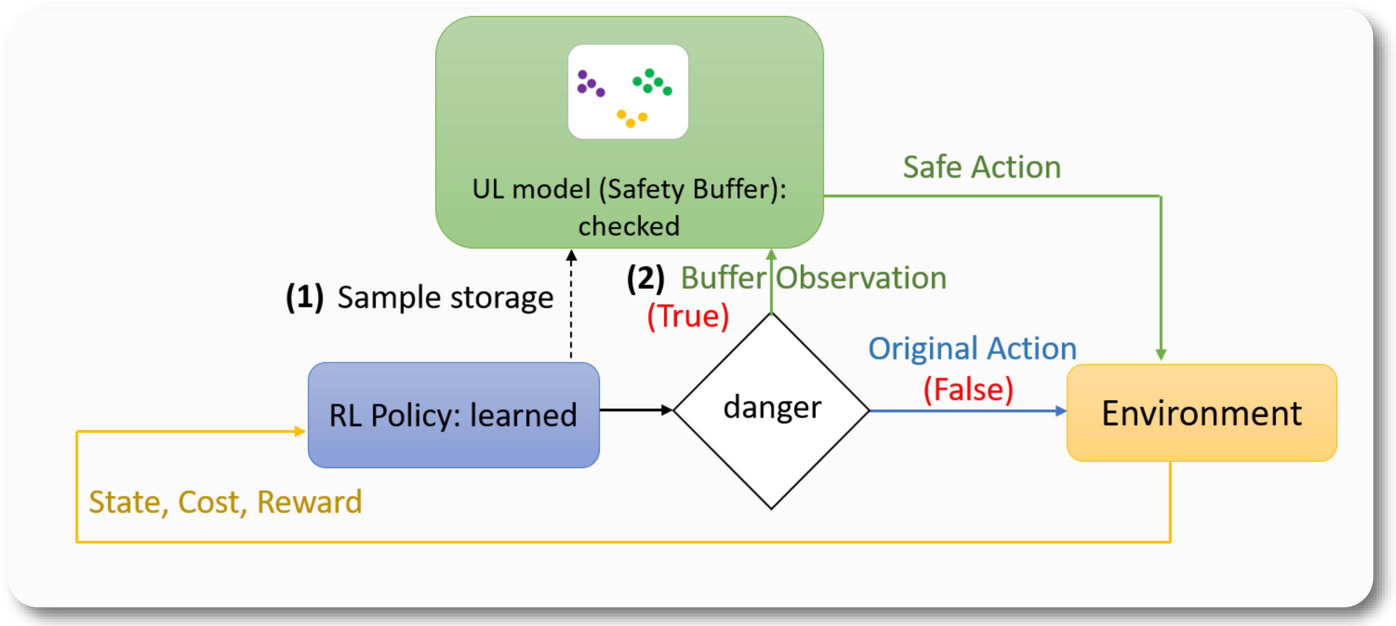
- Constrained RL algorithms with Lagrangian method can only ensure safety asymptotically [1] [2].

$$\max_{\theta}\min_{\lambda\geq 0} L(\theta,\lambda) = f(\theta) - \lambda g(\theta)$$

Original Objective Constraint

- Inspired by humans' mechanism in avoiding danger
  - ⇨ Recall the past experience of overcoming similar situations
  - ⇨ Avoid the risk by repeating similar recovery actions

Method: *Managing safety buffer using unsupervised learning*



- Safety buffer stores "recovery" actions that bring the agent from dangerous states to safe zones.
- "Recovery" action is executed depending on whether the situation is danger.

☺ Execute original action from trained RL policy

☹ Activate safety buffer for acquiring a candidate action set

References

[1] Altman, 1998, "Constrained Markov decision process with total cost criteria: Lagrangian approach and dual linear program"

[2] Chow et al., 2019, "Lyapunov-based safe policy optimization for continuous control"

Algorithm: *Safe Reinforcement Learning*

- Hypothesis: *Enough* number of clusters instead of the *optimal* number of clusters can lead to good performance

Algorithm 1 Safe RL using Unsupervised Action Planning

Initialize policy π_ϕ and safety buffer D
Pre-train the policy π for a small number of epochs
for $epoch = 1, 2, \dots$ **do**
  $[s_0, c_0] \sim P(s_0, c_0)$ ▷ Initialize state s and cost c
  for $t = 0, 1, \dots, T$ **do**
    $a_t \sim \pi_\phi(a_t|s_t)$
    $b_t = b(s_t)$ ▷ Extract the state features
    if $c_t \geq \hat{c}$ **then** ▷ If dangerous
      a_t = queryRecoveryAction(a_t, b_t, D) ▷ Activate safety protection mechanism
    end if
    $[s_{t+1}, c_{t+1}, b_{t+1}, r_t] \sim P(s_{t+1}, c_{t+1}, b_{t+1}, r_t | s_t, a_t)$
    if $c_t \geq \hat{c}$ and $c_{t+1} < \hat{c}$ **then** ▷ If recovers from danger
      $D \leftarrow D \cup (b_t, a_t, r_t)$
    end if
    $s_t \leftarrow s_{t+1}$, $c_t \leftarrow c_{t+1}$, $b_t \leftarrow b_{t+1}$
    if end of the episode **then**
      Rebuild clusters in the safety buffer D ▷ Regularly updates clusters
    end if
  end for
  Update π_ϕ ▷ Standard RL steps
end for

Algorithm 2 queryRecoveryAction

Input: action a_t, state feature b_t, and the safety buffer D
Acquire an action set A containing actions in the same cluster with b_t
if $a_t \in A$ **then**
  return a_t
else
  return the action $\tilde{a}_t \in A$ with the maximum reward
end if

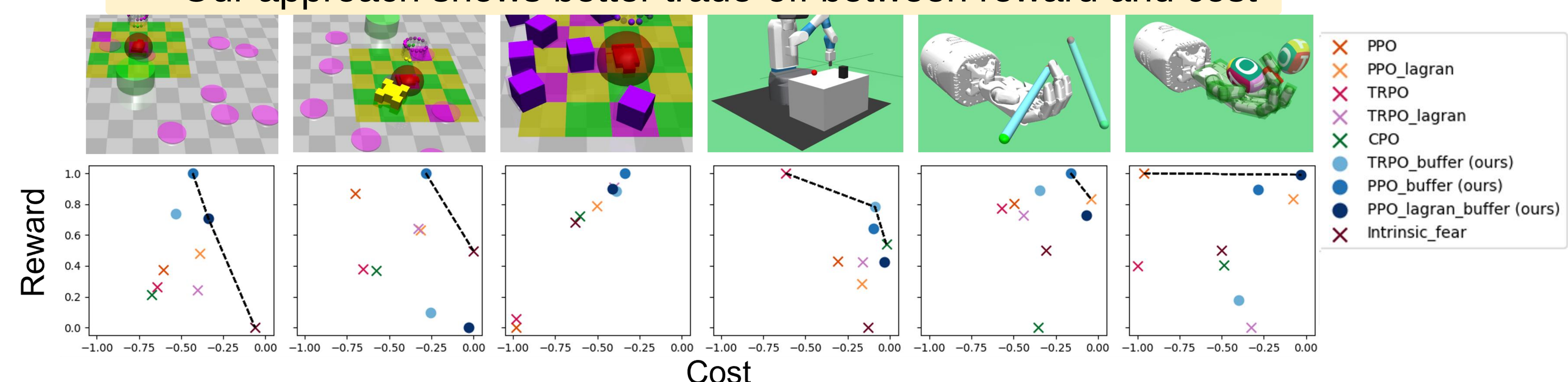
- Algo. 1: augmented learning process with conservative exploration mechanism via safe action planning
- Algo. 2: action planning process

Experiment: *Benefit of integrating RL with action planning*

- Relative cumulative failures during learning
  - ⇨ Our approach have lower number of failures compared with Lagrangian methods
  - ⇨ Intrinsic fear model can only reduce failures in simple tasks

Task/ Algorithm	PPO	TRPO	PPO+Lag	TRPO+Lag	CPO	TRPO+Buffer (Ours)	PPO+Lag (Ours)	PPO+Lag+Buff (Ours)	Intrinsic Fear
Goal Navigation	1.00	0.90	0.70	0.39	0.58	0.37	0.43	0.29	**0.14**
Push Navigation	0.99	1.00	0.51	0.46	0.74	0.22	0.32	0.15	**0.10**
Survival Navigation	1.00	0.98	0.15	0.13	0.17	0.06	**0.05**	0.12	0.70
Fetch Push w/o Toppling	0.72	1.00	0.29	0.32	0.06	0.16	0.13	**0.04**	0.19
Pen Manipulation w/o Falling	0.98	1.00	0.41	0.76	**0.36**	0.56	0.45	0.41	0.50
Egg Manipulation w/o Crush	1.00	0.90	0.15	0.39	0.27	0.21	0.13	**0.12**	0.33

- Pareto optimal solutions during testing
  - ⇨ Our approach shows better trade-off between reward and cost



Experiment: *Ablation studies*

- Number of clusters (N) with corresponding reward (R) and failure (F)

Task/ Number of Clusters	Brute Force R	Brute Force F	$N^{1/10}$ R	$N^{1/10}$ F	$N^{1/3}$ R	$N^{1/3}$ F	$N^{1/2}$ R	$N^{1/2}$ F	$N^{8/10}$ R	$N^{8/10}$ F
Goal Navigation	**28.52**	0.47	20.66	0.56	25.37	**0.43**	28.44	0.45	25.32	0.44
Push Navigation	2.69	0.30	2.41	0.38	**3.12**	0.35	2.93	0.31	2.72	**0.29**
Survival Navigation	1.73	0.33	1.06	0.37	1.95	0.32	**2.03**	**0.31**	1.55	0.36
Fetch Push w/o Toppling	-0.39	0.08	-0.41	**0.05**	-0.40	0.08	**-0.32**	0.10	-0.37	0.11
Pen Manipulation w/o Falling	-1.00	0.24	-1.20	0.37	-1.13	0.24	-0.87	**0.21**	**-0.84**	0.26
Egg Manipulation w/o Crush	-1.84	0.33	-1.91	0.35	**-1.77**	0.29	-1.82	**0.25**	-1.80	0.28

- Danger threshold for 6 robotic control tasks