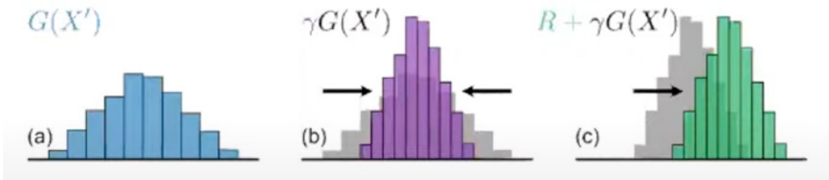
Distributional Reinforcement Learning

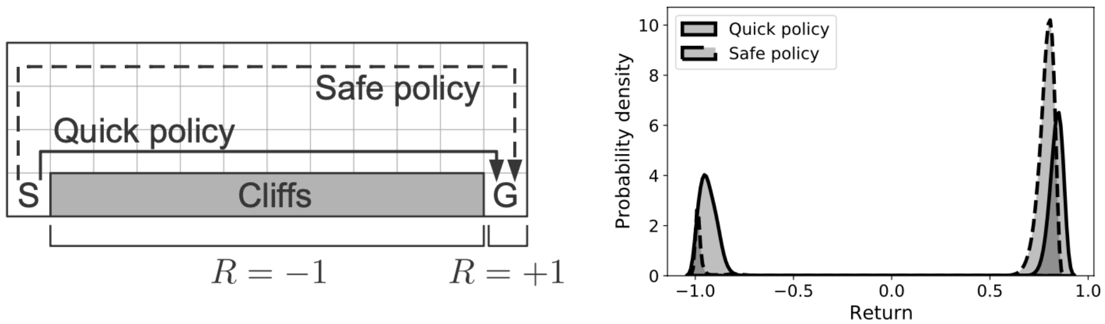


香港科技大学(广州)
THE HONG KONG
UNIVERSITY OF SCIENCE AND
TECHNOLOGY (GUANGZHOU)

Why Distributional Reinforcement Learning?

A toy example

cliff walking with noise – wind could blow the agent to a random direction



Left: The Cliffs environment, along with the path preferred by the quick and safe policies. **Right**: The return distribution for these policies, estimated by sampling 100,000 trajectories from the environment. The figure reports the distribution as a probability density function computed using kernel density estimation (with bandwidth 0.02).

Conclusion: A quick policy that walks along the cliff's edge and a safe policy that walks two cells away from the edge. The return distribution of the faster policy, in particular, is sharply peaked around -1 and 1: the goal may be reached quickly, but the agent is more likely to fall.

Advantage: The expected return alone fails to differentiate these policies. For risk-sensitive decision-making, the safe policy is superior as its distribution avoids the significant negative tail.

[1] Distributional Reinforcement Learning, Chapter 2, p22

Standard & Distributional RL and Convergence

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

Standard RL algorithms estimate the expected value of return Z^π:

$$V^\pi(x) := \mathbb{E}\left[Z^\pi(x)\right] = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(x_t, a_t) \mid x_0 = x\right]$$

the action is fixed to a particular choice

$$Q^\pi(x, a) := \mathbb{E}\left[Z^\pi(x, a)\right] = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(x_t, a_t)\right]$$

$$x_t \sim P(\cdot|x_{t-1}, a_{t-1}), a_t \sim \pi(\cdot|x_t), x_0 = x, a_0 = a.$$

Standard Bellman Operator:

$$\mathcal{T}^\pi Q(x, a) = \overbrace{\mathbb{E}\left[R(x, a)\right]}^{\text{Expectation of current reward}} + \overbrace{\gamma}^{\text{Future-to-current discount}} \underbrace{\mathbb{E}_{P,\pi}\left[Q(x', a')\right]}_{\text{expectation of future rewards}}.$$

Distributional Bellman Operator:

$$\mathcal{T}^\pi Z(x, a) :\overset{D}{=} \overbrace{R(x, a)}^{\text{Distribution of current reward}} + \gamma \overbrace{Z(x', a')}^{\text{Distribution of future rewards}},$$

$$x' \sim P(\cdot|x, a), a' \sim \pi(\cdot|x').$$

Bellman optimality operator:

$$Q(x, a) = \mathcal{T} Q(x, a) := \mathbb{E}\left[R(x, a)\right] + \gamma \mathbb{E}_P \max_{a'} Q(x', a').$$

Distributional Bellman optimality operator:

$$\mathcal{T} Z(x, a) :\overset{D}{=} R(x, a) + \gamma Z(X', \arg\max_{a' \in \mathcal{A}} \mathbb{E}\, Z(X', a'))$$

Standard TD update:

$$V(x) \leftarrow V(x) + \alpha(r + \gamma V(x') - V(x)),$$
$$a \sim \pi(\cdot|x), r \sim R(x, a), x' \sim P(\cdot|x, a).$$

What is distributional TD update?

Return Distribution Representation

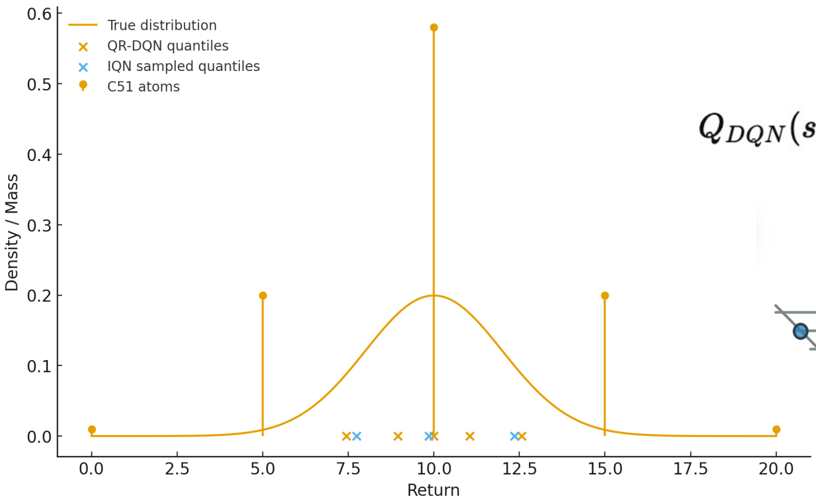
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

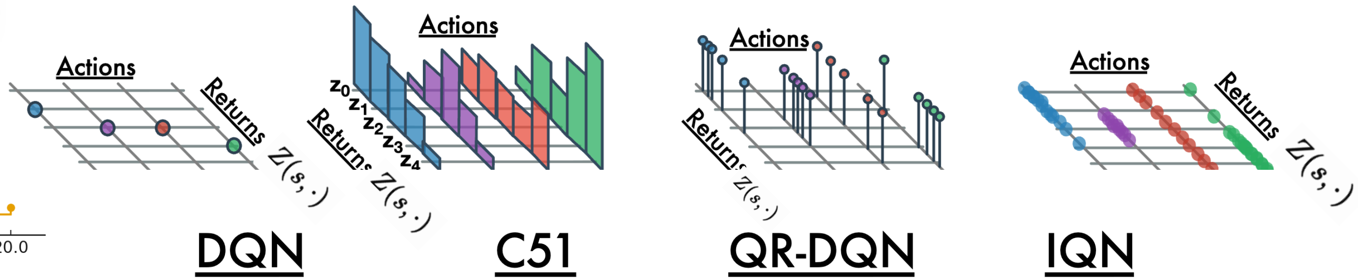
Q: How to represent the return distribution and how to do the dynamic programming?

A: Network architectures for DQN and recent distributional RL algorithms.

Return Representation: given a distribution by taking a fixed action $Z(s,a) \sim \mathcal{N}(\mu = 10,\ \sigma^2 = 4)$

Representation Way	Expectation	Predefined uniform return atoms	Predefined uniform return quantiles	Sampled quantiles
		$z = [0,\ 5,\ 10,\ 15,\ 20]$	$\tau = [0.1,\ 0.3,\ 0.5,\ 0.7,\ 0.9]$	$\tau = 0.13, 0.47, 0.88, \ldots$
	$Q_{DQN}(s,a) = 10$	$Z_{C51}(s,a) = \{(z_i, p_i)\}_{i=1}^{N}$	$\theta_i(s,a) \approx F_Z^{-1}(\tau_i)$	$f_\theta(s,a,\tau) \approx F_Z^{-1}(\tau)$



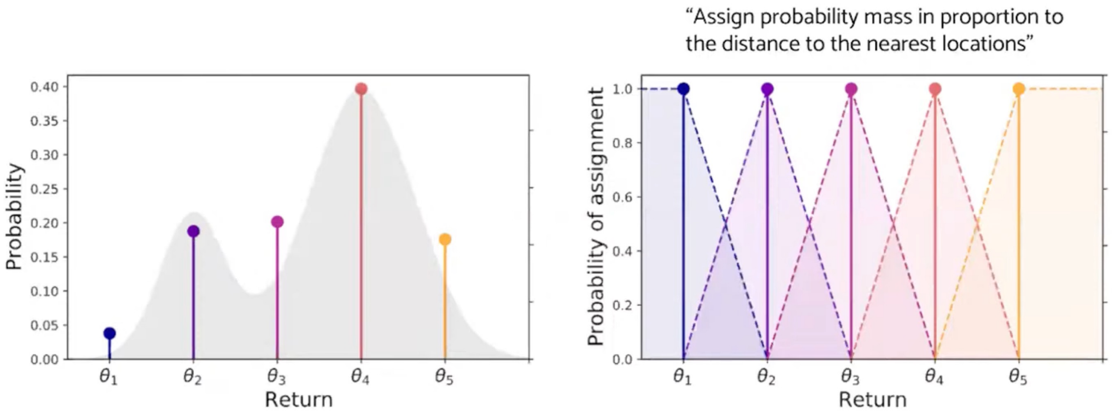


[2] Implicit Quantile Networks for Distributional Reinforcement Learning, ICML. 2018

Categorical and Quantile-based Representation

Q: How to represent the return distribution and how to do the dynamic programming?

A1: Categorical (Predefined uniform return atoms**):**



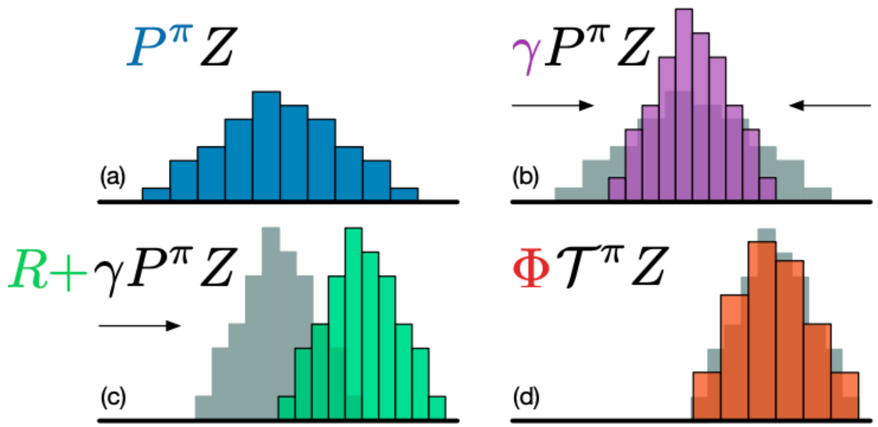


Figure 1. A distributional Bellman operator with a deterministic reward function: (a) Next state distribution under policy π, (b) Discounting shrinks the distribution towards 0, (c) The reward shifts it, and (d) Projection step (Section 4).

(a) $P^\pi Z = \{(0, 0.01), (5, 0.20), (10, 0.58), (15, 0.20), (20, 0.01)\}$

(b) $\gamma P^\pi Z \rightarrow \{(0, 0.01), (4.5, 0.20), (9, 0.58), (13.5, 0.20), (18, 0.01)\}$

(c) $R + \gamma P^\pi Z \rightarrow \{(2, 0.01), (6.5, 0.20), (11, 0.58), (15.5, 0.20), (20, 0.01)\}$

(d) $\Phi \mathcal{T}^\pi Z \rightarrow \{(0, 0.006), (5, 0.144), (10, 0.524), (15, 0.296), (20, 0.030)\}$

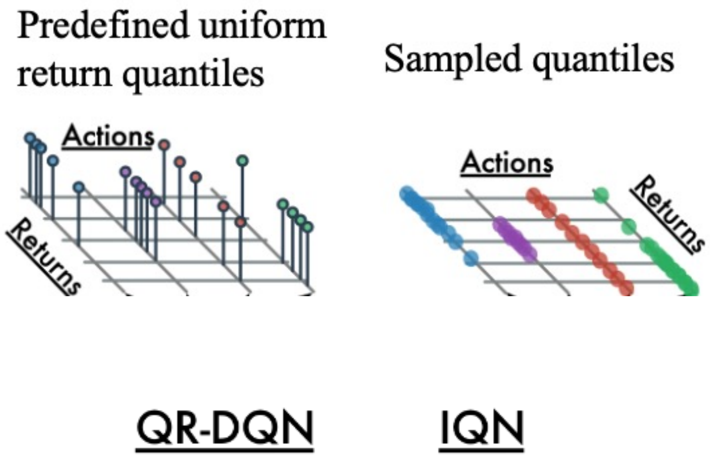
[2] A Distributional Perspective on Reinforcement Learning, ICML. 2017

Categorical and Quantile-based Representation

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

Q: How to represent the return distribution and how to do the dynamic programming?

A2: Quantile-based methods:



QRDQN (with predefined quantile positions) :

$$
\begin{aligned}
&Q(x', a') := \textstyle\sum_j q_j \theta_j(x', a') \\
&a^* \leftarrow \arg\max_{a'} Q(x, a') \\
&\mathcal{T}\theta_j \leftarrow r + \gamma\theta_j(x', a^*), \quad \forall j
\end{aligned}
$$

$\theta_i(x, a)$ represents the estimated value of the return distribution $Z_\theta(x, a)$ at the i-th quantile.

Objective function with quantile Huber loss

$$\textstyle\sum_{i=1}^{N} \mathbb{E}_j \left[\rho^{\kappa}_{\hat{\tau}_i}(\mathcal{T}\theta_j - \theta_i(x, a))\right]$$

$$\rho^{\kappa}_{\tau}(u) = |\tau - \delta_{\{u<0\}}|\mathcal{L}_\kappa(u). \quad \mathcal{L}_\kappa(u) = \begin{cases} \frac{1}{2}u^2, & \text{if } |u| \le \kappa \\ \kappa(|u| - \frac{1}{2}\kappa), & \text{otherwise} \end{cases}.$$

If we underestimate the target (target > predicted; u>0), we multiply by the weight τ . If we overestimate, we multiply by 1- τ .
It means that if the quantile τ is big, we penalize underestimation more to balance the data distribution over quantile.

[2] Implicit Quantile Networks for Distributional Reinforcement Learning, ICML. 2018

Quadrupedal Locomotion -- ICRA24

Robust Quadrupedal Locomotion via Risk-Averse Policy Learning

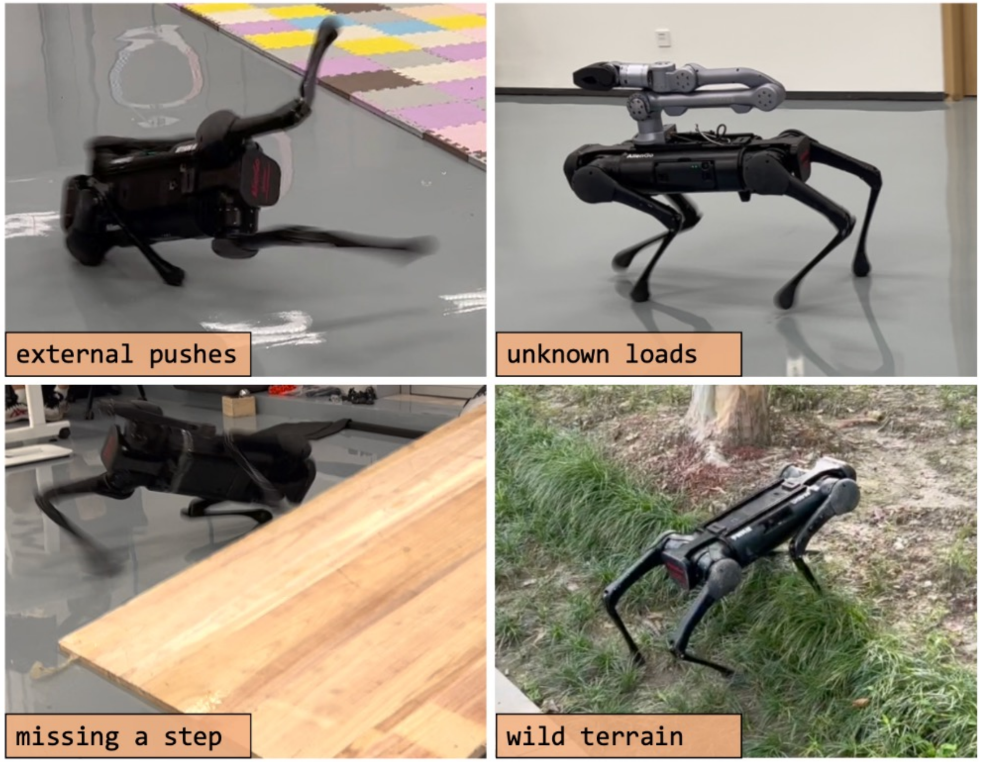
Jiyuan Shi[1,2*], Chenjia Bai[2†], Haoran He[2,3], Lei Han[4], Dong Wang[2], Bin Zhao[2,5], Mingguo Zhao[1], Xiu Li[1], Xuelong Li[2,5]

Aleatoric Uncertainty:

The inherent, irreducible randomness in the environment

Robust locomotion controller:

Encountering risks in the environment, such as sudden pushes and missing a step.



[5] Robust Quadrupedal Locomotion via Risk-Averse Policy Learning, ICRA, 2024

Problem Definition

MDP: $\mathbf{o}_t = [\mathbf{v}_t, \boldsymbol{\omega}_t, \mathbf{g}_t, \mathbf{c}_t, \mathbf{q}_t, \dot{\mathbf{q}}_t, \mathbf{a}_{t-1}] \in \mathbb{R}^{48}$

Action: target joint positions

Reward: $$\boldsymbol{r} = \boldsymbol{r}_{\text{task}} - \sum_{i=1}^{M} w_i \mathbb{I}_{|s_i| > \bar{s}_i} \cdot \mathcal{B}_p$$

TABLE I: Definition of reward functions. Here τ refers to joint torque.

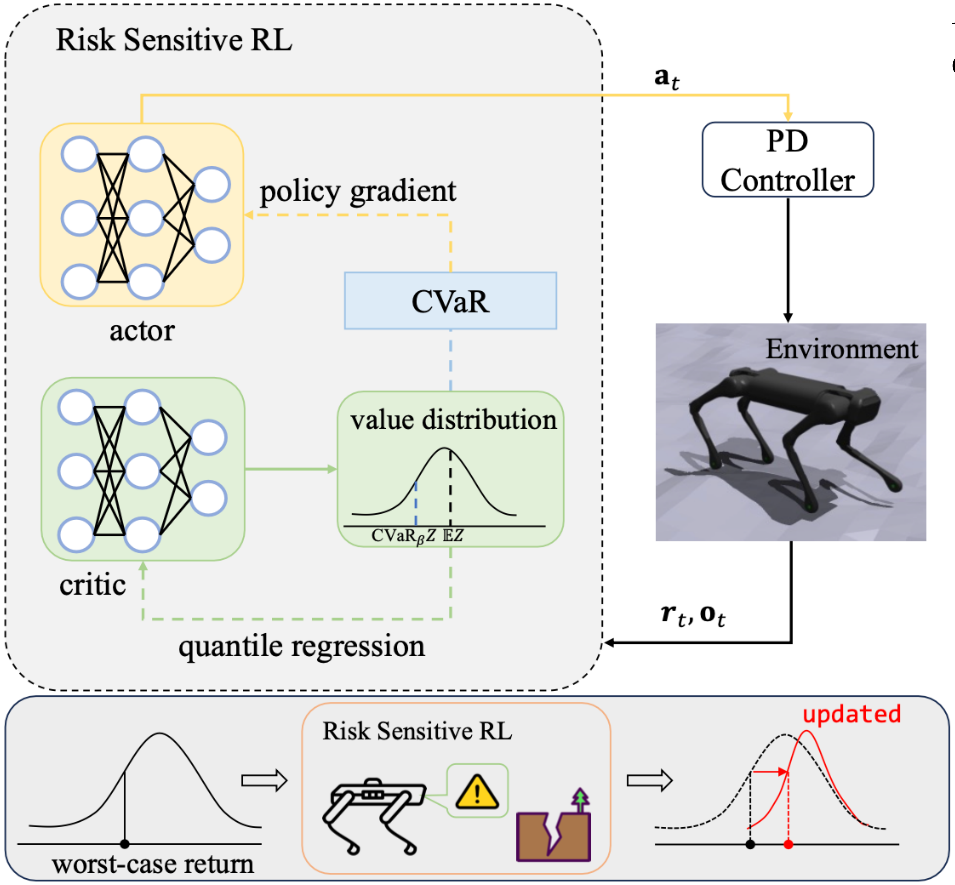
Reward Term	Definition	Weight
linear velocity tracking	$e^{-(\mathbf{v}_{xy}^{\text{cmd}} - \mathbf{v}_{xy})^2/\sigma}$	5
angular velocity tracking	$e^{-(\boldsymbol{\omega}_{\text{yaw}}^{\text{cmd}} - \boldsymbol{\omega}_{\text{yaw}})^2/\sigma}$	0.5
linear velocity penalty	v_z^2	-1.0
angular velocity penalty	$\boldsymbol{\omega}_{\text{roll,pitch}}^2$	-0.05
joint acceleration	$\ddot{\mathbf{q}}^2$	-2.5e-7
torques	τ^2	-2e-5
action magnitude	$\mathbf{a}^2$	-0.01
collision	$n_{\text{collision}}$	-1e-3
action rate	$(\mathbf{a}_t - \mathbf{a}_{t-1})^2$	-0.01
torque smooth	$(\tau_t - \tau_{t-1})^2$	-3e-4
feet air time	$\Sigma_{f=0}^{4}(\mathbf{t}_{air,f} - 0.5)$	2

TABLE II: Definition of risks. The risk terms will be added to the reward function defined in (2).

Risk Term	Threshold	Weight
base pitch	0.5 rad	20
base roll	1 rad	100
joint velocity	10 rad·s^{-1}	100
joint acceleration	1000 rad · s^{-2}	100
joint torque	40 N·m	150

Method Framework

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)



Risk-sensitive policy learning: Critic is based on IQN. The goal of RALL is to find a risk-sensitive policy by maximizing the distorted expectation of Z. （V没用到，beta，At；intuiation）

$$\nabla_\phi J(\phi) = \mathbb{E}\left[\sum_{t=0}^{T} A_t \nabla_\phi \log\left(\pi_\phi(a_t \mid s_t)\right)\right]$$

$$V_\beta(s) := \mathbb{E}_{\tau \sim U([0,1])}\left[Z_{\beta(\tau)}(s)\right]$$

$$\mathbf{CVaR}_\eta(Z(s)) = \mathbb{E}\left[Z_{\tau \sim U([0,\eta])}(s)\right]$$

$$\nabla_\phi J(\phi) = \mathbb{E}\left[\sum_{t=0}^{T} A_t^{\mathrm{CVaR}} \nabla_\phi \log(\pi_\phi(a_t \mid s_t))\right]$$

$$A_t^{\mathrm{CVaR}} = r_t + \gamma \mathbf{CVaR}_\eta(Z(s_{t+1})) - \mathbf{CVaR}_\eta(Z(s_t))$$

Aleatoric uncertainty of return:

$$IQR = Q_3 - Q_1, \quad Q_3 = F_Z^{-1}(0.75), Q_1 = F_Z^{-1}(0.25)$$

When IQR is higher than a threshold, choose

the policy trained by a lower CVaR value (0.5)