2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

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], *Fellow, IEEE*

***Abstract*— The robustness of legged locomotion is crucial for quadrupedal robots in challenging terrains. Recently, Reinforcement Learning (RL) has shown promising results in legged locomotion and various methods try to integrate privileged distillation, scene modeling, and external sensors to improve the generalization and robustness of locomotion policies. However, these methods are hard to handle uncertain scenarios such as abrupt terrain changes or unexpected external forces. In this paper, we consider a novel risk-sensitive perspective to enhance the robustness of legged locomotion. Specifically, we employ a distributional value function learned by quantile regression to model the aleatoric uncertainty of environments, and perform risk-averse policy learning by optimizing the worst-case scenarios via a risk distortion measure. Extensive experiments in both simulation environments and a real Aliengo robot demonstrate that our method is efficient in handling various external disturbances, and the resulting policy exhibits improved robustness in harsh and uncertain situations in legged locomotion.**

I. Introduction

Quadrupedal robots are widely recognized for their exceptional agility and remarkable capability to traverse complex terrains, which is crucial for scenarios such as industrial inspections and firefighting. Previous methods adopt Model Predictive Control (MPC) for quadrupedal robots, while it typically requires precise dynamics modeling with domain-specific knowledge [1], [2] and there is a trade-off between the model accuracy and computational complexity. Recently, model-free Reinforcement Learning (RL) demonstrated impressive performance in legged locomotion without dynamics modeling [3]. The RL policy can be trained by interacting with simulated environments, especially the parallel simulator like Isaac Gym [4], allowing robots to traverse various complex terrains such as rocks, stairs, snow, and beaches [5].

Existing RL-based approaches try to enhance the robustness and generalization ability of locomotion policies via privileged distillation, scene modeling, and external sensors. Specifically, privileged distillation methods adopt a teacher-student architecture to help the student policy infer the privileged state of the true environment [6]–[9]; scene modeling methods explicitly learn the scene geometry [10], terrain traversability [11], or via volumetric models; other methods equipped robots with cameras or LiDAR to enhance their terrain traversal capabilities [12], [13]. Although these methods show extraordinary performance in legged locomotion, they are still hard to handle abrupt events in locomotion such as unexpected terrain changes or external forces. The reason is they only infer the information of the current state from historical or privileged information, without considering the possible risk events in the future. Meanwhile, external sensors are often unreliable and need additional models for understanding and reasoning in the environment. In addition, RL-based methods often adopt domain randomization [14], [15] to enhance the robustness in sim-to-real transfer, while excessive randomization may lead to an overly conservative policy.

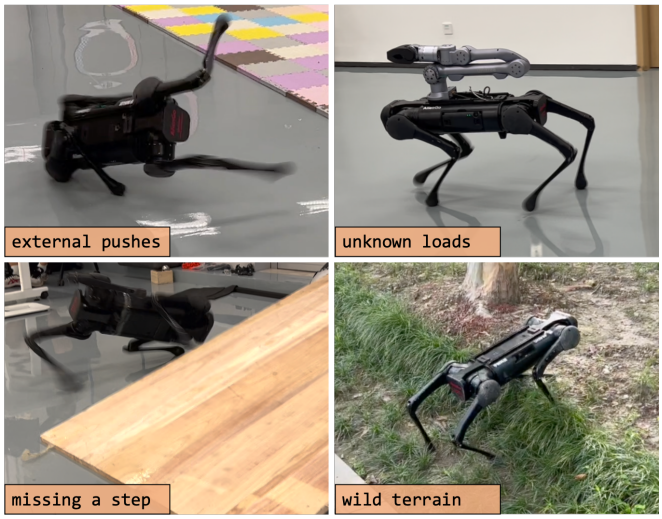

Fig. 1: We trained a robust locomotion controller through risk-sensitive RL. The robot demonstrated robustness when encountering risks in the environment, such as sudden pushes and missing a step.

Different from the above methods, we explicitly consider modeling the risks in legged locomotion and take a *risk-sensitive* perspective in policy learning to enhance the robustness of locomotion policy. Specifically, optimizing the expected return like previous methods cannot avoid risky events (e.g., falling) since avoiding risks may slow down the robot's speed and reduces the expected return [16]. However, we prioritize preventing the robot from falling which may cause hardware damage rather than precisely tracking the speed and directional commands. Such a risk-averse perspective can enhance the robot's ability to withstand uncertain disturbances [17].

In this work, we propose Risk-Averse Legged Locomotion (RALL), which performs risk-averse policy learning via a distributional value function to estimate the aleatoric uncertainty (i.e., risks) [18] of the environment. The value

[1]Tsinghua University, China. [2]Shanghai Artificial Intelligence Laboratory, China. [3]Shanghai Jiao Tong University, China. [4]Tencent Robotics X, China.[5]Northwestern Polytechnical University, China.
*The work was conducted during the internship of Jiyuan Shi at Shanghai Artificial Intelligence Laboratory. †Corresponding Author.

distribution is learned by quantile regression in an actor-critic framework, then the agent can obtain risk-sensitive policies by considering various risk preferences. To obtain a robust policy in legged locomotion, we optimize the bottom percentile of value distribution to learn a policy that performs well in the worst case via Conditional Value-at-Risk (CVaR), resulting in a *risk-averse* policy. Further, considering environments which contain multiple types of terrains, the robot may need to switch between policies with different risk preferences according to the specific circumstance. To this end, we propose IQR (Interquartile Range) of the quantile function as a risk-level measurement of the environment. The agent can choose to use a risk-averse policy or an ordinary policy based on IQR. For implementation, we design several proprioception-based risks in legged locomotion where each kind of risk occurs following a Bernoulli distribution. The resulting controller is able to generalize well across diverse perturbations, transcending the limitations imposed by the training environment's diversity.

To the best of our knowledge, RALL provides the first risk-sensitive policy learner in locomotion control of quadruped robots. Through a comprehensive evaluation in both simulation and real-world experiments, we demonstrate that RALL significantly enhances the robustness of locomotion. We show that RALL equips a real Unitree Aliengo [19] robot with the capability of traversing challenging terrains, withstanding dynamic loads, and resisting substantial external disturbances, without relying on the external sensors and extensive randomization. The main contributions of this paper are as follows:

- We present a novel perspective on achieving a robust locomotion controller via distributional value function and risk-sensitive policy learning. The resulted controller enabled the robot to resist heavy impact and traverse challenging terrains.
- We propose IQR as a risk measure in legged locomotion to enable the robot to choose policy with different risk preferences according to the current environment. IQR can be combined with other locomotion policies to enhance their robustness by switching to a risk-averse policy for environments with a higher IQR.
- The simulated and real-world experiments on Aliengo robot show that the RALL agent performs robustness and can traverse challenging terrains, endure dynamic loads, and recover from significant external pushes.

II. Related Work

A. RL-based Quadruped Locomotion

RL-based methods for locomotion control in quadrupedal robots demonstrate the capability to traverse complex terrains [9], [13], [20]. However, the robustness of quadruped robot is still an open problem [21], [22]. A prevalent approach to bridge the reality gap and enhance robustness is privileged distillation, which involves a teacher policy to encode privileged information (e.g., elevation map of the surrounding terrain and randomized parameters) into a latent vector [6], [7], [9], [10], [12], [13], [23]. The student is trained via supervised learning using accessible states on real robots. However, this framework is inefficient in training independent policies [22] and cannot predict risky events in the future. DreamWaQ [8] leveraged an asymmetric actor-critic architecture and a context-aided estimator network to infer the terrain properties, while it still needs privileged information in training. In contrast, we only use proprioceptive feedback in policy training without privileged knowledge or teacher-student architectures. Other methods also incorporate external sensors like cameras or LiDAR in the RL framework for scene understanding [24], while it relies on vision models and additional computing. In RALL, we find the risk-averse policy can perform robustly without external sensors, and our method can be easily combined with vision-based policies [13]. Several works propose learning a recovery controller to enable robots to recover from a fall [25], [26]. Nevertheless, our work has a different objective to prevent the agent from falling since it can easily result in hardware damage.

B. Distributional RL and Risk-Sensitive Learning

The distributional perspective in RL has a rich history [27]–[29]. In deep RL, C51 [30] first applies distributional Bellman equation to learn the value distribution, whose support is a set of atoms. In distributional RL, the value function estimates the whole distribution of return rather than its mean. An improvement over C51 is QR-DQN [31], which parameterizes the values of fixed quantiles and minimizes the Wasserstein distance to a target distribution. IQN [32] further extends the discrete quantile fractions to a continuous function by using a distortion function. IQN can approximate different quantiles of return distribution and learn policies under different risk measures, such as Wang [33], CPT [34], and Conditional Value at Risk (CVaR) [35], [36], leading to risk-averse or risk-seeking policies [37]–[39]. The value distribution can reflect the aleatoric uncertainty of the environment [39], which is used for risk-sensitive learning in online exploration [40]–[43] to avoid risk events or in an offline setting to learn risk-averse policy [44], [45].

While there have been numerous studies, the application of return distribution in real quadruped robots is limited. Haarnoja et al. use a C51-style agent [30] for controlling a biped robot playing soccer [46], without considering risk-sensitive learning. Contemporary work [47] proposed a Distributional PPO algorithm based on QR-DQN and PPO, enabling robots to learn policies with different risk preferences. Although it considers the issue of risk preferences in quadruped locomotion control, it does not further improve the robustness of locomotion policy through risk-sensitive reinforcement learning. Another major distinction between our work and [47] is the proposal of using the interquartile range of the distributional value function as a measure of environmental risk level, which allows robot to assess environmental risk levels in real-time.

III. METHOD

A. Problem Definition

We model the problem of robust locomotion control as a Markov Decision Process (MDP) as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P}(\cdot|s,a)$ is the transition probability, $R : \mathcal{S} \times \mathcal{A} \to \mathcal{R}$ is the stochastic reward function, and $\gamma \in [0,1)$ is the discount factor. Since we do not use the privileged information in training, the observation $\mathbf{o}_t$ of our framework is equivalent to the state $\mathbf{s}_t$, then we have $\mathbf{s}_t = \mathbf{o}_t$ and

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

where $\mathbf{v}_t$ is the base linear velocity, $\boldsymbol{\omega}_t$ is the base angular velocity, $\mathbf{g}_t$ is the gravity vector in the body frame, $\mathbf{c}_t \in \mathbb{R}^3$ is the velocity command, $\mathbf{q}_t$ is the joint angle, $\dot{\mathbf{q}}_t$ is the joint velocity, and $\mathbf{a}_{t-1}$ is the last action. We use the target joint position as action, and the low-level torque command is calculated via a PD controller.

Although it's hard to anticipate risks in the environment, the proprioception of the robot could provide clues to indicate whether the robot is facing potential risks. For instance, when the robot's roll angle exceeds a certain threshold, the robot is prone to roll over. In RALL, we introduce risks by adding penalization terms to the reward function. To be specific, when certain state components of the robot exceed predefined thresholds, we apply a relatively large penalization to the reward with a probability of p, so the final reward function can be presented as

$$\boldsymbol{r} = \boldsymbol{r}_{\text{task}} - \sum_{i=1}^{M} w_i \mathbb{I}_{|s_i| > \bar{s}_i} \cdot \mathcal{B}_p, \tag{2}$$

where $\boldsymbol{r}_{\text{task}}$ is the reward function to accomplish the locomotion task. The details of $\boldsymbol{r}_{\text{task}}$ are given in Table I. The penalty term in (2) indicates risks, where M is the number of risks, w_i is the weight of each kind of risk, $\mathbb{I}$ is an indicator function, s_i and $\bar{s}_i$ are the state and its risk threshold, and $\mathcal{B}_p$ is a Bernoulli variable with a probability of p. Table II gives the detailed setup of risks.

In (2), we adopt a very small probability (i.e., $p = 10^{-4}$) for risk events. Thus, the risk penalty can barely affect the expected return but will have a significant impact on the value distribution, especially in the worst case.

B. Distributional Value Function

To estimate the return distribution for risk-sensitive learning, we propose to train a distributional critic based on the risk-injected reward function in (2). The learned value distribution will be further used to 1) obtain a risk-averse policy and 2) estimate the current risk level.

Normally, the objective of RL is to maximize the expected cumulative return $\mathbb{E}[Z^\pi(s,a)]$, where $Z^\pi(s,a)$ is the return distribution. We have $Z^\pi(s,a) = \Sigma_{t=0}^{\infty} \gamma^t R(s_t, a_t)$ is a random variable representing the sum of discounted rewards for the agent following policy π. Many standard RL methods estimate the action-value function $Q^\pi(s,a) = \mathbb{E}[Z^\pi(s,a)]$, which could be characterized by the Bellman equation $Q^\pi(s,a) = \mathbb{E}[R(s,a)] + \gamma \mathbb{E}_{\mathcal{P},\pi}[Q^\pi(s',a')]$. In distributional RL, the action-value distribution can be learned using distributional Bellman operator [30], as

$$\mathcal{T}^\pi Z(s,a) :\overset{D}{=} R(s,a) + \gamma Z(S', A'), \tag{3}$$

where $S' \sim \mathcal{P}(\cdot|s,a), A' \sim \pi(\cdot|s')$, and $Y :\overset{D}{=} U$ denotes that two random variables have equal probability laws. In the following, we denote $F_Z(z) = Pr(Z \leq z)$ as the cumulative density function (CDF) of the distribution Z, and $F_Z^{-1}(\tau)$ as the quantile function (i.e., inverse CDF). For $\tau \in [0,1]$, $F_Z^{-1}(\tau) := \inf\{y \in \mathbb{R} : \tau \leq F_Y(y)\}$. Theoretical work [30] shows the distributional Bellman operator is a contraction in the p-Wasserstein metric, which measures the optimal transport between distributions, as

$$W_p(Z, \mathcal{T}^\pi Z) = \left(\int_0^1 \left| F_Z^{-1}(\omega) - F_{\mathcal{T}^\pi Z}^{-1}(\omega) \right|^p d\omega \right)^{1/p}. \tag{4}$$

In order to take risk into account, we follow Implicit Quantile Network (IQN) [32] to estimate the quantile function by using a continuous quantile function $Z_\tau(s,a;\theta) := F_{Z(s,a)}^{-1}(\tau)$ parameterized by θ. For two samples $\tau, \tau' \sim U([0,1])$, the temporal difference (TD) error is

$$\delta_{\tau,\tau'} = r + \gamma Z_{\tau'}(s', a'; \theta) - Z_\tau(s, a; \theta), \tag{5}$$

The overall critic loss is given by

$$\mathcal{L}_{\text{critic}}(\theta) = \frac{1}{N'} \sum_{i=1}^{N} \sum_{j=1}^{N'} \rho_\tau^\kappa(\delta_{\tau_i,\tau_j}), \tag{6}$$

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

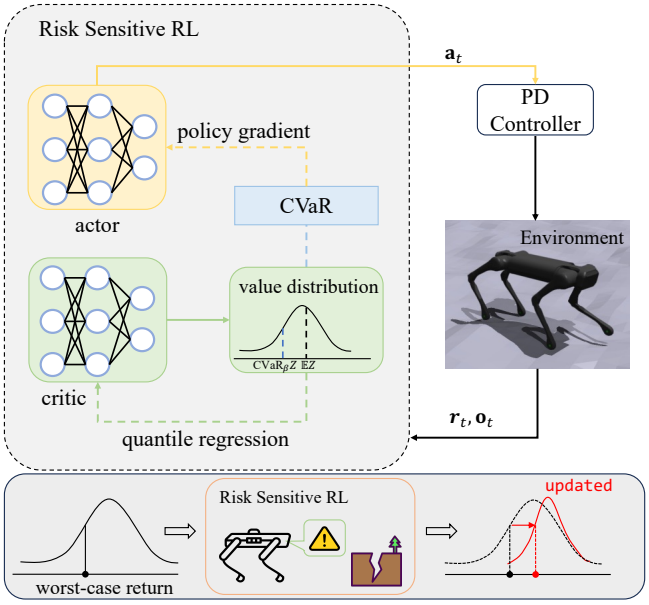

Fig. 2: Overall framework of our method. The critic network estimates the value distribution, and the risk-averse policy is obtained by optimizing the CVaR objective. The policy is supposed to perform well under worst-case scenarios.

where N and N' are the number of samples , and $\rho_{\tau_i}^{\kappa}$ is the quantile Huber loss [31] defined by

$$\rho_{\tau}^{\kappa}(\delta_{\tau_i,\tau_j}) = |\tau - \mathbb{I}_{\{\delta_{\tau_i,\tau_j}<0\}}|\mathcal{L}_{\kappa}(\delta_{\tau_i,\tau_j}), \text{ where}$$
$$\mathcal{L}_{\kappa}(\delta_{\tau_i,\tau_j}) = \begin{cases} \frac{1}{2}\delta_{\tau_i,\tau_j}^2, & \text{if } |\delta_{\tau_i,\tau_j}| \le \kappa \\ \kappa\left(|\delta_{\tau_i,\tau_j}| - \frac{1}{2}\kappa\right), & \text{otherwise} \end{cases}, \tag{7}$$

In continuous action settings, the next state-action value distribution $Z(S', A')$ in (3) is a mixture distribution of all possible state-action value distributions, which is infeasible to compute. Thus, we follow [48] to avoid this issue by directly approximating the next state value distribution $Z(S')$ instead of $Z(S', A')$, where $Z(S')$ can be integrated into policy gradient algorithms.

C. Risk-Sensitive Policy Learning

Based on the distributional value distribution, we adopt the policy gradient method to learn the policy, as

$$\nabla_{\phi} J(\phi) = \mathbb{E}\left[\sum_{t=0}^{T} A_t \nabla_{\phi} \log\left(\pi_{\phi}(a_t \mid s_t)\right)\right], \tag{8}$$

where A_t is the advantage function of the value distribution, and ϕ is the parameter of the actor-network. According to discussions in Section III-B, A_t is given by $A_t = r_t + \gamma V(s_{t+1}) - V(s_t)$, where $V(s) := \mathbb{E}[Z_{\tau}(s)]$.

In RALL for legged locomotion, we aim to learn policies with specific *risk preferences* by applying a distortion function [37] to τ. Let $\beta : [0,1] \to [0,1]$ be a distortion risk measure, then the distorted expectation of Z under β is

$$V_{\beta}(s) := \mathbb{E}_{\tau \sim U([0,1])}\left[Z_{\beta(\tau)}(s)\right]. \tag{9}$$

The goal of RALL is to find a risk-sensitive policy by maximizing the distorted expectation of Z. A commonly used risk-aversion measure is conditional value-at-risk (CVaR) as $\text{CVaR}(\eta, \tau) = \eta\tau$, which simply changes the sampling distribution from $\tau \sim U([0,1])$ to $\tau \sim U([0,\eta])$. Then the CVaR objective of the policy is

$$\text{CVaR}_{\eta}(Z(s)) = \mathbb{E}\left[Z_{\tau \sim U([0,\eta])}(s)\right], \tag{10}$$

and the policy gradient under the CVaR measure is given as,

$$\nabla_{\phi} J(\phi) = \mathbb{E}\left[\sum_{t=0}^{T} A_t^{\text{CVaR}} \nabla_{\phi} \log(\pi_{\phi}(a_t \mid s_t))\right], \text{where}$$
$$A_t^{\text{CVaR}} = r_t + \gamma \text{CVaR}_{\eta}(Z(s_{t+1})) - \text{CVaR}_{\eta}(Z(s_t)). \tag{11}$$

Intuitively, since CVaR_{η} only considers the bottom quantiles are less than η (with $\eta < 1$), the policy is learned to be risk-averse and performs well in worst-case scenarios [37].

In quadrupedal locomotion, we consider risk events such as sudden changes in terrain, external disturbance, and uncertain load. Such scenarios may bring large penalties in reward function, which can be learned by bottom quantile functions. By performing policy gradient in (11), we obtain a robust policy to exhibit favorable performance in worst-case scenarios, thereby enhancing the robot's resistance to disturbances. We set η to be 0.5 in practice. The schematic diagram of our approach is illustrated in Fig. 2. We also use advanced update tricks in policy gradients such as GAE [49] and PPO-Clip [50] to improve the performance. For real-world applications, we conduct domain randomization in simulation to facilitate sim-to-real transfer.

D. Risk-Aware Meta Controller

In RALL, one consideration is that the CVaR objective only optimizes the worst-case scenario, which may lead to suboptimal performance in normal situations. An intuitive solution would be switching control policies according to specific circumstances. Specifically, for relatively simple environments, we can employ a risk-neutral policy by setting $\eta = 1$ in the above objectives; while for scenarios with increased risk events, the controller should switch to a conservative policy obtained via the optimization of $\text{CVaR}_{0.5}$.

Since we focus on the robust locomotion control of quadrupedal robots without external sensors, devices such as camera or LiDAR is infeasible in estimating the surrounding information. Fortunately, given the value distribution, we can use the return distribution as a metric for assessing the risk level of the current environment, which has been verified in previous distributional RL works [41], [42]. In RALL, we propose using the variance of the quantile distribution as the risk measure. Instead of directly calculating the variance, we employ the *interquartile range (IQR)* to estimate the aleatoric uncertainty of returns, which is given by

$$IQR = Q_3 - Q_1, \quad Q_3 = F_Z^{-1}(0.75), Q_1 = F_Z^{-1}(0.25). \tag{12}$$

Compared to variance, the advantage of using IQR is that it is less affected by outliers, which makes it suitable for locomotion tasks in quadruped robots.

IV. Experiments

A. Simulation

We train our policy in Isaac Gym simulation [4] based on the open-source framework in [5]. The actor and critic networks have the same hidden dimensions of $[512, 256, 128]$. The critic network outputs estimated values of 64 quantiles, which are sampled from $U([0, 1])$, and optimization objective of the policy is distorted by $\text{CVaR}_{0.5}$. For the actor-critic algorithm, we set the clipping range, generalized advantage estimation factor λ, and discount factor γ to 0.2, 0.95, and 0.99, respectively, with a learning rate of 1e-3. In order to facilitate the sim-to-real transfer, we incorporate domain randomization throughout the training process, as detailed in Table III. We introduce Gaussian noise to state to make the robot robust against observation errors. The robot is trained on various terrains in simulation, including smooth slopes, rough slopes, and discrete obstacles. We employ the terrain curriculum introduced by [5] to enable the robot to traverse challenging terrains progressively.

We train 4096 agents parallelly on a PC with a 32-core Intel i9-13900K CPU @ 5.5GHz, 128GB RAM, and an NVIDIA RTX 4090 GPU. We train the policy for 6000 iterations, which take approximately 7 hours.

B. Hardware Setup

We use Unitree Aliengo [19] robot for real-world experiments. The robot has 12 degrees of freedom and weighs about 21kg. The computations are performed on an onboard NVIDIA Jetson TX2. The policy runs at 50Hz and the target joint angles were tracked by a PD controller at a frequency of 200 Hz. The PD gains are $K_p = 50$ and $K_d = 0.8$, respectively. During the evaluation, we send linear and angular velocity commands to the robot from a remote host. The command was updated at a frequency of 50Hz.

C. Compared Methods

To quantitatively analyze the improvement in robustness, we conducted experiments with the following methods:

- **Baseline**: The policy is trained using PPO with curriculum terrain setting following [5].
- **Expanded Domain Randomization(DR)**: The policy is trained under a broader range of domain randomization compared to *Baseline* to withstand larger disturbances.
- **RMA**: The policy is trained using RMA [7], a method that leverages the teacher-student framework to estimate a latent vector of the environment.

The compared methods and RALL are implemented with the same parameter configuration, including PPO parameters, domain randomization ranges (except DR) and observation noise. And the compared methods share the same network configurations with RALL except the value distribution head.

TABLE III: Domain randomization terms and their ranges. We simulate pushing on the robot by randomly introducing velocity perturbations to its base. The push interval is 5s.

Randomized Term	Range	Unit
friction coefficient	[0.5,1.5]	-
base mass	[-0.5, 1.0]	kg
push	[-1,1]	m/s

TABLE IV: Robustness evaluation for robot carrying challenging payloads. The masses of the ball, container, and robot arm are about 2kg, 0.1kg, and 4.5kg, respectively. For experiments with external pushes, we randomly applied a force of 100N in the x/y direction on the robot's body every 300 steps, and this force was sustained for 50 steps.

Experiment setup		Time to Fall (TTF)			
Load type	Terrain	Flat		Random	
	Push	False	True	False	True
Ball	Baseline	0.539	0.212	0.292	0.189
	DR	0.233	0.165	0.205	0.154
	RMA	0.338	0.224	0.305	0.189
	RALL	**0.963**	**0.947**	**0.366**	**0.301**
Frozen arm	Baseline	0.187	0.143	0.156	0.131
	DR	0.228	0.197	0.163	0.149
	RMA	0.318	0.178	0.191	0.148
	RALL	**0.987**	**0.986**	**0.384**	**0.317**
Moving arm	Baseline	0.299	0.246	0.259	0.206
	DR	0.249	0.247	0.190	0.183
	RMA	0.244	0.237	0.281	0.219
	RALL	**0.517**	**0.437**	**0.309**	**0.285**

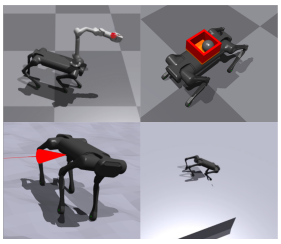

Fig. 3: We consider challenging scenarios including dynamic payload, external push and missing a step.

TABLE V: Success rates of robot walking down a platform.

Height	Policy	Success %
0.4m	baseline	14.4
	DR	20.6
	RMA	43.5
	RALL	**92.7**
0.45m	baseline	13.9
	DR	14.3
	RMA	21.4
	RALL	**30.5**

D. Results

To comprehensively assess the robustness of the controller, we consider various factors that could potentially give rise to risks, including loads, external forces, and terrain, as shown in Fig. 3. Learning methods have been proved effective in enabling robots to walk while carrying loads [51]. However, a more challenging scenario arises when the load carried by the robot is dynamic. This will introduce continuous changes to the inertia of the robot, increasing the risk of instability. We conducted three sets of experiments on carrying payloads. The first is carrying a ball that could roll within a rectangular container. The second is attaching a fixed Unitree Z1 robot arm [52] to Aliengo, and in the third setting we let the arm track random end effector pose when the quadrupedal robot is walking. For each experiment, we introduced two sets of variables: the terrain, which could either be flat or

Fig. 4: Snapshots of the robot's recovery when it missed a step or got pushed. The robust policy learned through risk-sensitive RL makes the robot retain balance rapidly when encountering intense disturbance.

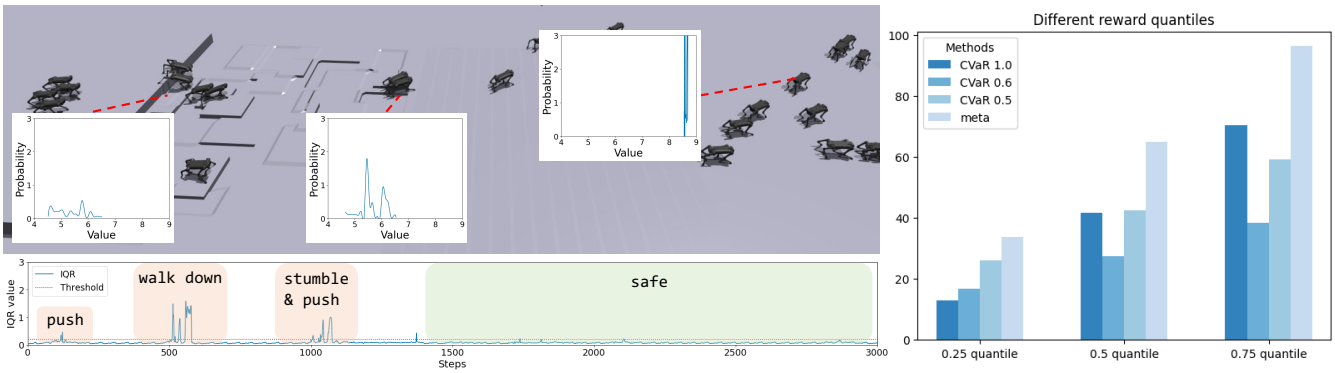

Fig. 5: We designed a race track to show the relationship between the IQR value and the current risk level. Robots that are encountering risks exhibit a more dispersed value distribution and a higher IQR value (*left*). We conducted experiments with 50 agents parallelly over 10 episodes. The result shows the meta controller receives highest return on all quantiles (*right*).

randomly generated rough terrain, and whether to introduce external forces to the robot base. To mitigate randomness, we conducted three trials with different random seeds, and repeated 10 times for each seed. We recorded the *Time to Fall* (TTF) for each experiment, which is the average episode length across all robot instances divided by the total episode length. The results, as shown in Table IV, indicate that our method outperforms others by making the robot survive for the longest duration under each setting. We also conducted experiments to assess the robot's robustness in handling hazardous terrains in the simulation. We let the robot to walk down a high platform and recorded the success rate of landing. Table V shows that our method has a significantly higher success rate compared to others.

We conducted numerous experiments on a real Aliengo robot. The results show the effectiveness of our approach in enhancing the robot's robustness. Snapshots of two experiments highlight the robot's ability to regain stability when approaching risks as shown in Fig. 4. For more real-world experiments, please refer to the supplemental video.

Finally, we conducted experiments on the meta controller proposed in Section III-D. We designed a race track that combined both challenging and normal terrains, and randomly applied external forces to the robot throughout the entire episode. Fig. 5 showcases the simulation environment and the IQR of the value distribution during the process. It could be seen that when the robot stepped down the platform and traversed the obstacles, the IQR of the value distribution was relatively higher than that of safe scenarios. To validate the effectiveness of the proposed meta controller quantitatively, we conducted experiments in this track with 50 agents parallelly over 10 episodes. The compared policies were obtained by optimizing different CVaR objectives, and the meta controller switched between a CVaR_1 and a $\text{CVaR}_{0.5}$ policy according to the IQR value. We recorded all the returns and calculated the 0.25, 0.5 and 0.75 quantile of the return distribution, which represented the worst-case, middle-case and best-case return, respectively. The result shows that the meta controller achieves the highest return in all worst, middle and best cases, which is consistent with our expectation.

V. CONCLUSION

In this work, we present a novel approach to enhancing the robustness of quadrupedal robot locomotion through risk-sensitive reinforcement learning. Experimental results show that our method enables the robot to resist significant disturbances. Moreover, the value distribution given by the critic could serve as an assessment of the current risk level.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (Grant No.62306242), the National Key R&D Program of China (Grant No.2022ZD0160102), STI 2030-Major Projects 2021ZD0201402 and Shanghai Artificial Intelligence Laboratory.

REFERENCES

[1] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli, "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1560–1567, 2018.

[2] D. Kim, J. Di Carlo, B. Katz, G. Bledt, and S. Kim, "Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control," *arXiv preprint arXiv:1909.06586*, 2019.

[3] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.

[4] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," in *Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

[5] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Conference on Robot Learning (CoRL)*, 2022.

[6] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, p. eabc5986, 2020.

[7] G. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," in *Robotics: Science and Systems*, 2022.

[8] I. M. A. Nahrendra, B. Yu, and H. Myung, "Dreamwaq: Learning robust quadrupedal locomotion with implicit terrain imagination via deep reinforcement learning," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 5078–5084.

[9] G. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," in *Robotics: Science and Systems*, 2022.

[10] R. Yang, G. Yang, and X. Wang, "Neural volumetric memory for visual locomotion control," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 1430–1440.

[11] J. Frey, D. Hoeller, S. Khattak, and M. Hutter, "Locomotion Policy Guided Traversability Learning using Volumetric Representations of Complex Environments," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2022, pp. 5722–5729.

[12] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science Robotics*, vol. 7, no. 62, p. eabk2822, 2022.

[13] A. Agarwal, A. Kumar, J. Malik, and D. Pathak, "Legged locomotion in challenging terrains using egocentric vision," in *6th Annual Conference on Robot Learning (CoRL)*, 2022.

[14] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, and S. Levine, "Learning to walk via deep reinforcement learning," in *Robotics: Science and Systems*, 2019.

[15] C. S. Imai, M. Zhang, Y. Zhang, M. Kierebiński, R. Yang, Y. Qin, and X. Wang, "Vision-guided quadrupedal locomotion in the wild with multi-modal delay randomization," in *2022 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2022, pp. 5556–5563.

[16] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Conference on Robot Learning (CoRL)*. PMLR, 2022, pp. 91–100.

[17] Y. Shen, M. J. Tobia, T. Sommer, and K. Obermayer, "Risk-sensitive reinforcement learning," *Neural computation*, vol. 26, no. 7, pp. 1298–1328, 2014.

[18] S. Depeweg, J.-M. Hernandez-Lobato, F. Doshi-Velez, and S. Udluft, "Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning," in *International Conference on Machine Learning*. PMLR, 2018, pp. 1184–1193.

[19] "Aliengo - Multifunctional, Industrial Level Application - Unitree," https://www.unitree.com/en/aliengo/.

[20] S. Choi, G. Ji, J. Park, H. Kim, J. Mun, J. H. Lee, and J. Hwangbo, "Learning quadrupedal locomotion on deformable terrain," *Science Robotics*, vol. 8, no. 74, p. eade2256, Jan. 2023.

[21] L. Wellhausen and M. Hutter, "ArtPlanner: Robust Legged Robot Navigation in the Field," *Field Robotics*, vol. 3, no. 1, pp. 413–434, Jan. 2023.

[22] G. Ji, J. Mun, H. Kim, and J. Hwangbo, "Concurrent Training of a Control Policy and a State Estimator for Dynamic and Robust Legged Locomotion," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4630–4637, Apr. 2022.

[23] H. He, C. Bai, H. Lai, L. Wang, and W. Zhang, "Privileged knowledge distillation for sim-to-real policy generalization," 2023.

[24] R. Yang, M. Zhang, N. Hansen, H. Xu, and X. Wang, "Learning vision-guided quadrupedal locomotion end-to-end with cross-modal transformers," in *International Conference on Learning Representations (ICLR)*, 2022.

[25] J. Lee, J. Hwangbo, and M. Hutter, "Robust recovery controller for a quadrupedal robot using deep reinforcement learning," *arXiv preprint arXiv:1901.07517*, 2019.

[26] Y. Ma, F. Farshidian, and M. Hutter, "Learning arm-assisted fall damage reduction and recovery for legged mobile manipulators," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 12 149–12 155.

[27] M. J. Sobel, "The variance of discounted markov decision processes," *Journal of Applied Probability*, vol. 19, no. 4, pp. 794–802, 1982.

[28] S. C. Jaquette, "Markov decision processes with a new optimality criterion: Discrete time," *The Annals of Statistics*, vol. 1, no. 3, pp. 496–505, 1973.

[29] D. J. White, "Mean, variance, and probabilistic criteria in finite markov decision processes: A review," *Journal of Optimization Theory and Applications*, vol. 56, pp. 1–29, 1988.

[30] M. G. Bellemare, W. Dabney, and R. Munos, "A distributional perspective on reinforcement learning," in *International conference on Machine Learning*. PMLR, 2017, pp. 449–458.

[31] W. Dabney, M. Rowland, M. Bellemare, and R. Munos, "Distributional reinforcement learning with quantile regression," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[32] W. Dabney, G. Ostrovski, D. Silver, and R. Munos, "Implicit quantile networks for distributional reinforcement learning," in *International conference on Machine Learning*. PMLR, 2018, pp. 1096–1105.

[33] S. Wang, "Premium calculation by transforming the layer premium density," *ASTIN Bulletin: The Journal of the IAA*, vol. 26, no. 1, pp. 71–92, 1996.

[34] L. Prashanth, C. Jie, M. Fu, S. Marcus, and C. Szepesvári, "Cumulative prospect theory meets reinforcement learning: Prediction and control," in *International Conference on Machine Learning*. PMLR, 2016, pp. 1406–1415.

[35] R. T. Rockafellar, S. Uryasev, *et al.*, "Optimization of conditional value-at-risk," *Journal of risk*, vol. 2, pp. 21–42, 2000.

[36] Y. Chow and M. Ghavamzadeh, "Algorithms for cvar optimization in mdps," *Advances in neural information processing systems*, vol. 27, 2014.

[37] N. A. Urpí, S. Curi, and A. Krause, "Risk-averse offline reinforcement learning," in *International Conference on Learning Representations (ICLR)*, 2021.

[38] Y. Jiang, Q. Liu, X. Ma, C. Li, Y. Yang, J. Yang, B. Liang, and Q. Zhao, "Learning diverse risk preferences in population-based self-play," *arXiv preprint arXiv:2305.11476*, 2023.

[39] C. Bai, T. Xiao, Z. Zhu, L. Wang, F. Zhou, A. Garg, B. He, P. Liu, and Z. Wang, "Monotonic Quantile Network for Worst-Case Offline Reinforcement Learning," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2022.

[40] H. Bharadhwaj, A. Kumar, N. Rhinehart, S. Levine, F. Shkurti, and A. Garg, "Conservative safety critics for exploration," *arXiv preprint arXiv:2010.14497*, 2020.

[41] B. Mavrin, H. Yao, L. Kong, K. Wu, and Y. Yu, "Distributional reinforcement learning for efficient exploration," in *International conference on Machine Learning*. PMLR, 2019, pp. 4424–4434.

[42] A. Mavor-Parker, K. Young, C. Barry, and L. Griffin, "How to stay curious while avoiding noisy tvs using aleatoric uncertainty estimation," in *International Conference on Machine Learning*. PMLR, 2022, pp. 15 220–15 240.

[43] Y. C. Tang, J. Zhang, and R. Salakhutdinov, "Worst cases policy gradients," *arXiv preprint arXiv:1911.03618*, 2019.

[44] N. A. Urpí, S. Curi, and A. Krause, "Risk-averse offline reinforcement learning," in *International Conference on Learning Representations (ICLR)*, 2021.

[45] Y. Ma, D. Jayaraman, and O. Bastani, "Conservative offline distributional reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 19 235–19 247, 2021.

[46] T. Haarnoja, B. Moran, G. Lever, S. H. Huang, D. Tirumala, M. Wulfmeier, J. Humplik, S. Tunyasuvunakool, N. Y. Siegel, R. Hafner, *et al.*, "Learning agile soccer skills for a bipedal robot

with deep reinforcement learning," *arXiv preprint arXiv:2304.13653*, 2023.

[47] L. Schneider, J. Frey, T. Miki, and M. Hutter, "Learning Risk-Aware Quadrupedal Locomotion using Distributional Reinforcement Learning," Sept. 2023. [Online]. Available: http://arxiv.org/abs/2309.14246

[48] D. W. Nam, Y. Kim, and C. Y. Park, "Gmac: A distributional perspective on actor-critic framework," in *International Conference on Machine Learning*. PMLR, 2021, pp. 7927–7936.

[49] J. Schulman, P. Moritz, S. Levine, M. I. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in *4th International Conference on Learning Representations (ICLR)*, 2016.

[50] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[51] G. Bellegarda, Y. Chen, Z. Liu, and Q. Nguyen, "Robust high-speed running for quadruped robots via deep reinforcement learning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 10 364–10 370.

[52] "Z1 - Dexterous Robotic Arm, Perfect Coordination - Unitree," https://www.unitree.com/en/arm/.