2023 IEEE Symposium Series on Computational Intelligence (SSCI)
Mexico City, Mexico. December 5-8, 2023

Uncertainty Quantification for Efficient and Risk-Sensitive Reinforcement Learning

1st Mohamed-Harith IBRAHIM
Mines Saint-Etienne, Univ. Clermont Auvergne, CNRS, UMR 6158 LIMOS, Institut Henri Fayol, F-42023, Saint-Etienne, France.
mharith.ibrahim@emse.fr

2nd Stéphane LECOEUCHE
IMT Nord Europe, Institut Mines-Télécom, Univ. Lille, Centre for Digital Systems, F-59000 Lille, France.
stephane.lecoeuche@imt-nord-europe.fr

3rd Jacques BOONAERT
IMT Nord Europe, Institut Mines-Télécom, Univ. Lille, Centre for Digital Systems, F-59000 Lille, France.
jacques.boonaert@imt-nord-europe.fr

4th Mireille BATTON-HUBERT
Mines Saint-Etienne, Univ. Clermont Auvergne, CNRS, UMR 6158 LIMOS, Institut Henri Fayol, F-42023, Saint-Etienne, France.
batton@emse.fr

***Abstract*—In complex real-world decision problems, ensuring safety and addressing uncertainties are crucial aspects. In this work, we present an uncertainty-aware Reinforcement Learning agent designed for risk-sensitive applications in continuous action spaces. Our method quantifies and leverages both epistemic and aleatoric uncertainties to enhance agent's learning and to incorporate risk assessment into decision-making processes. We conduct numerical experiments to evaluate our work on a modified version of Lunar Lander with variable and risky landing conditions. We show that our method outperforms both Deep Deterministic Policy Gradient (DDPG) and TD3 algorithms by reducing collisions and having significant faster training. In addition, it enables the trained agent to learn a risk-sensitive policy that balances performance and risk based on a specific level of sensitivity to risk required for the task.**

***Index Terms*—Reinforcement Learning, Uncertainty quantification, Risk-sensitive control**

I. Introduction

Reinforcement Learning (RL) has been successfully applied to solve a wide range of sequential decision-making problems in video games, board games, robotics, energy management and various other domains. However, applying RL approaches to real-world problems may face major challenges. First of all, one of the main challenges lies in developing agents that can make robust and safe decisions in complex environments. Traditional RL approaches typically focus on maximizing the expected value of return which is not always suitable to real-world problems with high variability in return and risky tasks. On the other hand, RL often requires a significant number of interactions between the agent and its environment to learn an optimal policy, leading to high sample-complexity. To address these challenges, researchers have developed various RL methods to reduce sample-complexity and to consider risk in decision-making. According to [1], the notion of risk in RL is actually related to the stochastic nature of the environment and the fact that even an optimal policy may perform poorly in some cases. This is because maximizing the expected return does not prevent rare occurrences of large negative outcomes.

In this work, we suggest to tackle these challenges by quantifying uncertainties and incorporating them into the learning process to explore efficiently and to learn a risk-sensitive policy. In fact, agents in RL encounter two types of uncertainties: aleatoric uncertainty and epistemic uncertainty. Aleatoric uncertainty arises due to the inherent stochastic nature of the environment, stemming from factors such as stochastic rewards, transition dynamics, or policies. On the other hand, epistemic uncertainty, also known as model uncertainty, expresses the agent's lack of knowledge about the problem. The first can be used to quantify variability and to consider risk in decision-making, while the second can be used to evaluate agent's knowledge to guide the learning process.

In this paper, we focus on continuous control tasks within environments characterized by variable and risky conditions. We use an actor-critic architecture based on DDPG [2] and introduce several modifications. These adaptations primarily focus on enhancing the random exploration process and refining the risk-neutral criterion commonly employed in methods like DDPG and TD3 [3]. To accomplish this, we use Distributional Reinforcement Learning (DRL) methods and ensemble bootstrapping to quantify both types of uncertainty. Instead of modeling the expected value of return, DRL allows to learn the complete distribution of return which can be more informative and can model intrinsic randomness of return. Furthermore, the distribution can be leveraged to incorporate a risk-sensitive criterion, enabling the consideration of risk in the decision-making process.

The reminder of the paper is structured as follows. Section II provides an overview of related works focusing on uncertainty quantification in RL. In Section III we present background information on Markov Decision Processes (MDP) and DRL. This section aims to provide readers a fundamental understanding of the key concepts and techniques that form the basis of our proposed method. Section IV details our method, outlining the key components and steps involved

978-0-7381-4408-5/23/$31.00 ©2023 IEEE

in our framework. Finally, in Section V, we examine the effectiveness and performance of our method by presenting experimental results and discussions that highlight the advantages of the proposed approach.

II. Related Work

Uncertainty quantification in RL is used to learn risk-sensitive policies, to train uncertainty-aware agents or to improve sample-complexity. Uncertainties are modeled using different methods. Authors in [4] provide an overview of existing techniques for uncertainty-aware RL.

Learning a risk-sensitive policy in RL can be achieved by incorporating a risk-sensitive criterion within the DRL framework. Different methods have been proposed to learn the return distribution such as C51 [5] that parameterizes the return distribution as a categorical distribution, QR-DQN [6] and IQN [7] that use quantile functions. Empirically, DRL methods have shown improvements in final performance and sample complexity over non-distributional methods. In addition, they have been used to improve the critic of a policy gradient algorithm called D4PG [8], achieving state-of-the-art performance in continuous control tasks.

Previous studies have explored using the return distribution to compute risk-sensitive measures. Authors in [9] suggest to approximate the return distribution as a Gaussian distribution and to use the Bellman equation for the mean and the variance to learn its moments. By doing so, the Gaussian distribution is used to calculate a risk-sensitive measure, the Conditional Value at Risk (CVaR), in a closed-form. Additionally, authors in [10] and in [11] suggest to learn the full return distribution using DRL methods in order to compute the CVaR or other risk-sensitive measures as in [12].

Estimating epistemic uncertainty is also important to train an uncertainty-aware agent. It is used as a measure to evaluate an agent's knowledge about its environment. Multiple works use epistemic uncertainty to improve sample-complexity by guiding the agent to explore unknown situations with limited knowledge [13] [14] [15]. In addition, epistemic uncertainty has been used in RL to train agents safely [16] [17].

To train an uncertainty-aware agent, ensemble RL methods are employed, which have gained popularity for quantifying uncertainties. In RL, ensemble methods serve various purposes, including efficient exploration [18] [13] [19] [20] and to reduce the overestimation bias of Q-values. Authors in [14] propose a unified framework combining mentioned benefices of ensemble RL.

Most prior work quantifies one of the two types of uncertainty to improve learning or to learn a risk-sensitive policy. However, some papers are interested in both for applications with discrete action spaces [21] [22] or to learn a risk-sensitive evaluation metric [23]. In this paper, we consider both uncertainties to design an uncertainty-aware framework for risk-sensitive applications with continuous action spaces. To the best of our knowledge, few papers deal with both uncertainties to solve problems with continuous action spaces. Perhaps most closely related is the work of authors in [24] who also train multiple distributional critics for risk-sensitive applications and for efficient exploration. As a main difference, in this paper, we build our framework on well known exploration methods that make use of ensemble learning [18] and epistemic uncertainty [13] using UCB (Upper-Confidence Bound) exploration technique. Furthermore, we propose to use noisy networks [25] to reduce the number of trained neural networks in our framework.

III. Preliminaries

We consider standard RL setting where the interaction of an agent with an environment is modeled as an MDP defined by a tuple $(\mathcal{S}, \mathcal{A}, R, P, \gamma)$. Here, $\mathcal{S}$, $\mathcal{A}$ denote the continuous state and continuous action spaces respectively, $R(s,a,s') : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the reward function, $P(s',a|s) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$ specifies the probability of transitioning to a state s' from state s and executing action a and finally $\gamma \in]0,1]$ is the discount factor.

We use DRL methods to model return distribution. For a policy π, the return, $Z^\pi(s,a) = \sum_{t=0}^{T} \gamma^t R_t(s,a,s')$, is considered as a random variable representing the sum of discounted rewards observed after taking action a in state s while following policy π. To learn the return distribution, we exploit the distributional Bellman equation of returns for policy evaluation:

$$\mathcal{T}Z^\pi(s,a) = R(s,a,s') + \gamma Z^\pi(s',a'), \tag{1}$$

where a' is the optimal action from state s' and the equality is in the sense of probability laws.

We follow [6] and represent the return distribution through its quantile function. Let F_Z^{-1} denote the quantile function of a random variable Z. By definition, $F_Z^{-1}(\tau) = inf\{z \in \mathbb{R} : \tau \leq F_Z(z)\}$ where τ is the probability that Z takes on a value less or equal to z and F_Z is the Cumulative Distribution Function (CDF) of Z. In the rest, we denote $F_Z^{-1}(\tau) := z_\tau$. Given a state s and an action a, the action value distribution is approximated by N quantile values $z_{\hat{\tau}_i}$ assigned to $\hat{\tau}_i = \frac{\tau_{i-1}+\tau_i}{2}$, for $1 \leq i \leq N$ where $\tau_i = \frac{i}{N}$. These quantile values are estimated by training a neural network to minimize:

$$\sum_{i=1}^{N} \frac{1}{N} \sum_{j=1}^{N} \rho_{\hat{\tau}_i}^{\kappa}(R(s,a,s') + \gamma z_{\hat{\tau}_j}(s',a') - z_{\hat{\tau}_i}(s,a)), \tag{2}$$

where ρ_τ^κ is the quantile Huber loss with threshold κ,

$$\rho_\tau^\kappa(u) = |\tau - \mathbb{1}_{\{u<0\}}| \mathcal{L}_\kappa(u), \tag{3}$$

with Huber loss,

$$\mathcal{L}_\kappa(u) = \begin{cases} \frac{1}{2}u^2 & \text{if } |u| \leq \kappa, \\ \kappa(|u| - \frac{1}{2}\kappa) & \text{otherwise.} \end{cases} \tag{4}$$

We leverage the quantile representation to compute risk measures which map a real-valued distribution to a real number and quantify the probability of occurrence of an event away from the expectation [26]. Some well-known risk measures used in risk-sensitive RL are variance, Value at Risk (VaR) and CVaR [27]. In this work, we use the CVaR which

represents the expected return we should experience in the worst $\alpha\%$ of cases defined as:

$$CVaR_\alpha = \mathbb{E}[Z(s,a)|Z(s,a) \leq z_\alpha], \tag{5}$$

where $\alpha \in [0,1]$ and z_α denotes the α-quantile of the distribution $Z(s,a)$.

IV. Method

In this section, we introduce UA2-MDC (**U**ncertainty-**A**ware **A**ctor with **M**ultiple **D**istributional **C**ritics) a method designed for RL problems with continuous action spaces. Our approach effectively deals with both aleatoric and epistemic uncertainties, enabling the learning of a risk-sensitive policy and enhancing exploration efficiency. We use an actor-critic architecture based on DDPG with a number of modifications: i) using a noisy neural network for a unique actor, ii) training M distributional critics to quantify both types of uncertainties, iii) optimizing a risk-sensitive criterion to learn a policy that prioritizes risk mitigation, and iv) employing UCB exploration for efficient learning.

Each critic is denoted as $\{Z_{\phi_k}\}_{k=1}^M$ where ϕ_k is the set of parameters of the k^{th} distributional critic while the actor is denoted as $\mu_{\tilde{\theta}}$ where $\tilde{\theta}$ is the set of noisy parameters.

A. Noisy Actor

Instead of training multiple actors, as commonly done in ensemble RL methods, we suggest to train one actor that uses a noisy network [25], denoted as $\mu_{\tilde{\theta}}$, with learnable parameters $\tilde{\theta}$. Noisy neural networks allow to model the distribution of parameters by assuming they follow a normal distribution, rather than learning single values. This approach involves adding noise directly to the parameters (weights and biases) of the actor, instead of injecting noise into the greedy action recommended by the actor. As a result, for the same input s_t, the output $\mu_{\tilde{\theta}}(s_t)$ of the actor can vary at each iteration, particularly during initial stages of training.

We suggest to use of a noisy neural network instead of training M neural networks for several reasons. Firstly, employing a noisy neural network helps to reduce the number of parameters that need to be trained. This reduction in parameters can lead to more efficient learning. Secondly, the addition of parameter noise helps to enhance overall performance and facilitate state-dependent exploration. By introducing noise directly to the actor's parameters, we can encourage exploration in a more adaptive and effective manner. Finally, in our method, as we handle continuous action spaces, the noisy actor allows to generate a diverse set of candidate actions, facilitating the application of UCB in action selection (refer to Subsection IV-D).

B. Distributional Critics

To capture both aleatoric and epistemic uncertainties, our approach relies on distributional critics. As mentioned previously, the distribution provides more informative insights than the expected value alone and effectively models the inherent randomness of returns. Additionally, ensemble methods are used to quantify the lack of knowledge about the problem. In this work, we represent the return distribution through its quantile function, following the approach proposed in [6] as introduced in Section III. To learn quantile values, we parameterize the quantile function through a neural network Z_ϕ with learnable parameters ϕ. For each state-action pair, the neural network outputs N quantile values. We also use a target network with parameters ϕ' to compute quantile target values.

To train an ensemble of critics, we adopt the approach outlined in [18]. Each critic's model is initialized independently and randomly to introduce diversity among the models. Furthermore, different samples are used to train each critic separately. Specifically, for each critic k, after an experience is played at time step t, a binary mask $m_{t,k}$ is generated from a Bernoulli distribution of parameter $p \in]0,1]$. Each bootstrap mask is associated to an experience to determine whether parameters of critic k should be updated using the played experience. Consequently, parameters $\{\phi_k\}_{k=1,\dots,M}$ are updated by taking gradient descent of the critic loss function:

$$\mathcal{L}_{critic}(\phi_k) = \mathbb{E}_{(s,a,r,s',m)\sim\mathcal{D}_K} m \times \sum_{i=1}^{N} \frac{1}{N} \sum_{j=1}^{N} \rho_{\hat{\tau}_i}^{\kappa}(u_{ij}), \tag{6}$$

with

$$u_{ij} = R(s,a,s') + \gamma z_j(s', \mu_{\tilde{\theta}'}(s')|\phi'_k) - z_i(s,a|\phi_k), \tag{7}$$

where ρ_τ^κ is the quantile Huber loss and $z_j(s', \mu_{\tilde{\theta}'}(s')|\phi'_k)$ represents the j^{th} quantile target value computed using the k^{th} critic's target network and the actor's target network. Each critic is trained on a batch $\mathcal{D}_K$ of K experiences (s,a,r,s',m) drawn from an experience replay memory $\mathcal{D}$.

C. Risk-Sensitive Policy

To train an agent to learn a risk-sensitive policy, we consider the use of a different optimization criterion. We denote the risk measure as ζ which represents a mapping from return distributions to a real number. In this work, we use the CVaR of returns. Unlike other risk measures, the CVaR is a coherent risk measure that can capture the tail risk which is crucial for capturing extreme events. In addition, using the CVaR allows to control the desired risk level through the choice of the CVaR level, enabling to adopt a customized risk management strategy. We denote ζ_α the CVaR of returns where α is a hyperparameter referring to the CVaR level. The actor is trained to maximize the risk-sensitive measure computed using all critics. The gradient of the loss function of the actor network is computed as:

$$\nabla_{\tilde{\theta}} J(\tilde{\theta}) = \mathbb{E}_{s\sim\mathcal{D}_K} \nabla_{\tilde{\theta}} \left(\frac{1}{M} \times \sum_{k=1}^{M} \zeta_\alpha(s, \mu_{\tilde{\theta}}(s)|\phi_k)\right), \tag{8}$$

where $\zeta_\alpha(s, \mu_{\tilde{\theta}}(s)|\phi_k)$ is the CVaR of level α computed using quantile values estimated by the k^{th} critic.

D. Uncertainty-Aware Agent and Action Selection

In addition to improving agent's performances and reducing learning time, training multiple critics can quantify the agent's uncertainty. An uncertainty-aware agent can estimate its lack of knowledge regarding a specific situation. The ensemble of distributional critics can be leveraged for efficient exploration based on epistemic uncertainty estimates. Instead of exploring randomly, this uncertainty can guide the agent to explore unknown situations with limited knowledge.

To achieve this, we adopt the concept of UCB, as demonstrated in [13]. UCB is constructed by combining the empirical standard deviation with the empirical mean of action values computed using all critics. In our approach, the agent exploits the action with the highest CVaR value, representing the lowest risk, while also actively exploring actions where critics produce high uncertainties:

$$a_t = arg\max_a \zeta_\alpha^{mean}(s_t, a) + \lambda \times \zeta_\alpha^{std}(s_t, a), \qquad (9)$$

where $\lambda \in \mathbb{R}_+$ is a hyperparameter that controls the degree of exploration. This encourages the exploration of state-action pairs with high epistemic uncertainties by adding an exploration bonus. UCB was initially proposed for efficient exploration in discrete action spaces. However, in [14], authors adapted UCB for continuous action spaces by generating multiple action candidates using separate actors and critics trained with different agents. In our approach, we leverage the capabilities of our actor, which employs a noisy neural network capable of producing different values for the same input, to generate action candidates. This allows us to effectively apply UCB in the context of UA2-MDC. The summarized algorithm for UA2-MDC can be found in Algorithm 1.

Algorithm 1: UA2-MDC

Set M, T, α, λ, p, K, γ, N
Initialize noisy actor network $\mu_{\tilde{\theta}}$, M distributional critic networks $\{Z_{\phi_k}\}_{k=1}^M$, and target networks $\mu_{\tilde{\theta}'}$, $\{Z_{\phi'_k}\}_{k=1}^M$
Initialize experience replay memory $\mathcal{D}$ of size T
for *episode* $= 1, 2, \ldots$ **do**
  episode starts at s_1
  for $t = 1, 2, \ldots$ **do**
    Create a set of M action candidates:
    $\mathcal{A} = \{a_{t,k} \sim \mu_{\tilde{\theta}}(s_t), \text{ with } k \in \{1, \ldots, M\}\}$
    Select the action that maximizes the UCB:
    $a_t = \arg max_{a \in \mathcal{A}}\ \zeta_\alpha^{mean}(s_t, a) + \lambda \zeta_\alpha^{std}(s_t, a)$
    Execute action a_t and get next state s_{t+1} and reward r_{t+1}
    Sample bootstrap mask $m_t \sim Ber(p)$
    Store experience $(s_t, a_t, r_{t+1}, s_{t+1}, m_t)$ in $\mathcal{D}$
    Sample a random batch $\mathcal{D}_K$ of size K from $\mathcal{D}$
    Update each critic according to (6)
    Apply the gradient in (8) to update the actor
    Soft update target networks

TABLE I
LANDING CONDITIONS DURING TRAINING.

Training scenarios				
Wind power values	$[0, 5]$	$[5, 10]$	$[10, 15]$	$[15, 20]$
Turbulence power values	$[0, 0.5]$	$[0.5, 1]$	$[1, 1.5]$	$[1.5, 2]$
Proportion of learning conditions	50%	25%	12.5%	12.5%

V. EXPERIMENTAL RESULTS

To evaluate our approach, we compare multiple agents using different methods: the first agent using DDPG, the second agent using TD3 and the third agent using UA2-MDC. Our method is designed to enable efficient learning and risk-aware decision-making in continuous action spaces. To demonstrate this, we train all agents in a risky environment with variable conditions. We choose the Lunar Lander environment available in gymnasium with wind effects applied to the lander [28]. In order to simulate an environment with diverse conditions and stochastic transition dynamics, we introduce variable wind power levels across episodes during the training process. Our main objective is to train an agent to safely solve the Lunar Lander environment with a variety of landing conditions that can occur over time. For more details about wind effects, please refer to gymnasium documentation. In Table I, we give more information on wind power during training. For instance, 50% of training episodes featured wind power levels ranging from 0 to 5, coupled with turbulence power levels ranging from 0 to 0.5. It is important to mention that all agents are trained on the exact same conditions including wind power time appearance during training. All hyperparameters used for UA2-MDC in experiments are also listed in Table II.

We compare our method with DDPG and TD3 that are usually used for continuous control tasks. In Fig. 1, we show the progression of both collision and success rates throughout

TABLE II
HYPERPARAMETERS OF UA2-MDC.

Hyperparameter	Value
Memory size (T)	1000000
Number of episodes	1000
Discount factor (γ)	0.99
Number of hidden layers (actor and critics)	2
Number of nodes (actor and critics)	256
Number of quantiles (N)	16
Activation function for hidden layers	ReLU
Activation function for the output (actor)	Tanh
Batch size (K)	128
Learning rate of the actor (β_1)	0.0005
Learning rate of the critic (β_2)	0.0005
Number of critics (M)	5
Degree of exploration (λ)	1
Binary mask probability (p)	0.75
CVaR level (α)	0.25

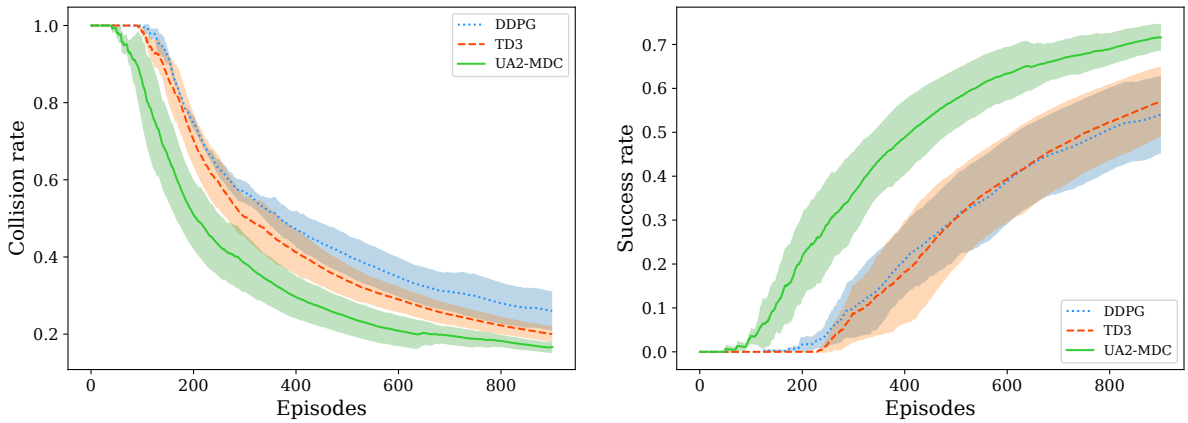

Fig. 1. Learning curves for the Lunar Lander environment. Each learning curve is averaged over 5 different random seeds and shaded by the standard deviation.

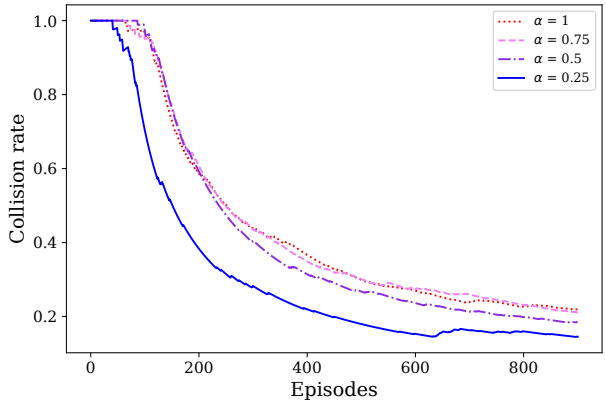

Fig. 2. Collision rate using UA2-MDC for the Lunar Lander environment with different α values.

the training process. Collision rate is calculated by dividing the number of episodes in which the agent crashes by the number of played episodes, while the success rate indicates the number of episodes in which the agent safely solves the task divided by the number of played episodes. We use the collision rate as a metric to assess performance under risk, while the success rate serves as an indicator of learning efficiency. We conducted simulations with 5 random seeds to ensure meaningful results.

Results in Fig. 1 show that UA2-MDC (with $\alpha = 0.25$) outperforms both DDPG and TD3 to solve the Lunar Lander environment under variable conditions. One notable advantage of our method is its lower collision rate during training, indicating a reduced number of failures encountered by the agent in comparison to other methods when attempting to solve the task. This advantage can be attributed to the use of a risk-sensitive optimization criterion. Additionally, UA2-MDC achieves a higher success rate and learns faster than DDPG and TD3. By leveraging the benefits of state-dependent exploration, as well as the advantages of DRL and ensemble methods, UA2-MDC significantly improves learning.

It is worth noting that the level of risk sensitivity can be controlled through the hyperparameter α. In our UA2-MDC approach, we trained several agents using different CVaR values α. Fig. 2 illustrates the impact of varying this hyperparameter on the collision rate during training. When $\alpha = 1$, the collision rate is similar to that achieved by TD3 (see Fig. 1), as it corresponds to maximizing the expectation of returns.

The trained agents are then evaluated by attempting to land in 100 episodes with variable landing conditions as in training and 100 episodes with extreme landing conditions characterized by maximum wind. Table III and Table IV present performance metrics for each method, including the average sum of rewards, the average number of steps to solve the task, the number of crashes, the number of successful landings across all episodes and the number of trained parameters.

Overall, UA2-MDC needs fewer iterations for each landing attempt and manage to keep the vehicle safe in all landing conditions compared to other existing methods. Results presented in Table III demonstrate that our method successfully finds a policy that offers a compromise between performance and risk. It prioritizes vehicle safety in all episodes, ensuring robustness and mitigating potential risks. Moreover, in the

TABLE III
PERFORMANCES WITH VARIABLE LANDING CONDITIONS.

Methods	UA2-MDC $\alpha = 0.25$	Other methods DDPG	TD3
Rewards	232	206	248
Steps	318	518	349
Crashes	0	3	8
Success	83	74	90
Parameters	413 449	137 475	206 340

TABLE IV
PERFORMANCES WITH EXTREME LANDING CONDITIONS.

Methods	UA2-MDC $\alpha = 0.25$	Other methods DDPG	TD3
Rewards	214	171	190
Steps	307	616	437
Crashes	0	5	8
Success	78	57	73

most extreme conditions (as shown in Table IV), UA2-MDC outperforms other existing approaches, showcasing its ability to handle variability in the environment effectively. Our method's ability to anticipate variability in the environment contributes to its significant advantage, especially in safety-critical systems. It is worth noting that all methods encountered these extreme conditions in less than 12.5% of their training episodes, highlighting the robustness of the learned policy by the agent.

VI. CONCLUSION

This work introduces UA2-MDC an uncertainty-aware agent designed for risk-sensitive applications in continuous action spaces. Our method is an off-policy actor-critic algorithm that effectively utilizes both epistemic and aleatoric uncertainties to learn a risk-sensitive policy efficiently. In comparison to DDPG and TD3, our approach demonstrates superior performance in a modified version of the Lunar Lander environment, particularly under extreme landing conditions. Our method not only outperformed other approaches in terms of learning speed and effectiveness but also integrates risk considerations into the decision-making process, establishing its relevance for real-world applications.

However, it is important to note that quantifying both types of uncertainty in our proposed method requires more computational resources and a larger number of parameters to train compared to DDPG and TD3. Furthermore, given the limited availability of risky environments with diverse conditions in continuous action spaces, our approach was primarily evaluated in a single environment. To validate the effectiveness of our approach, further evaluation in future environments with similar conditions would be beneficial.

REFERENCES

[1] J. Garcıa and F. Fernández, "A comprehensive survey on safe reinforcement learning," *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 1437–1480, 2015.

[2] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[3] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*. PMLR, 2018, pp. 1587–1596.

[4] O. Lockwood and M. Si, "A review of uncertainty for deep reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 18, no. 1, 2022, pp. 155–162.

[5] 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.

[6] 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, 2018.

[7] 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.

[8] G. Barth-Maron, M. W. Hoffman, D. Budden, W. Dabney, D. Horgan, D. Tb, A. Muldal, N. Heess, and T. Lillicrap, "Distributed distributional deterministic policy gradients," *arXiv preprint arXiv:1804.08617*, 2018.

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

[10] R. Singh, Q. Zhang, and Y. Chen, "Improving robustness via risk averse distributional reinforcement learning," in *Learning for Dynamics and Control*. PMLR, 2020, pp. 958–968.

[11] N. A. Urpí, S. Curi, and A. Krause, "Risk-averse offline reinforcement learning," *arXiv preprint arXiv:2102.05371*, 2021.

[12] X. Ma, L. Xia, Z. Zhou, J. Yang, and Q. Zhao, "Dsac: Distributional soft actor critic for risk-sensitive reinforcement learning," *arXiv preprint arXiv:2004.14547*, 2020.

[13] R. Y. Chen, S. Sidor, P. Abbeel, and J. Schulman, "Ucb exploration via q-ensembles," *arXiv preprint arXiv:1706.01502*, 2017.

[14] K. Lee, M. Laskin, A. Srinivas, and P. Abbeel, "Sunrise: A simple unified framework for ensemble learning in deep reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2021, pp. 6131–6141.

[15] F. L. Da Silva, P. Hernandez-Leal, B. Kartal, and M. E. Taylor, "Uncertainty-aware action advising for deep reinforcement learning agents," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, 2020, pp. 5792–5799.

[16] B. Lütjens, M. Everett, and J. P. How, "Safe reinforcement learning with model uncertainty estimates," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8662–8668.

[17] G. Kahn, A. Villaflor, V. Pong, P. Abbeel, and S. Levine, "Uncertainty-aware reinforcement learning for collision avoidance," *arXiv preprint arXiv:1702.01182*, 2017.

[18] I. Osband, C. Blundell, A. Pritzel, and B. Van Roy, "Deep exploration via bootstrapped dqn," *Advances in neural information processing systems*, vol. 29, 2016.

[19] X. Chen, C. Wang, Z. Zhou, and K. Ross, "Randomized ensembled double q-learning: Learning fast without a model," *arXiv preprint arXiv:2101.05982*, 2021.

[20] T. Hiraoka, T. Imagawa, T. Hashimoto, T. Onishi, and Y. Tsuruoka, "Dropout q-functions for doubly efficient reinforcement learning," *arXiv preprint arXiv:2110.02034*, 2021.

[21] W. R. Clements, B. Van Delft, B.-M. Robaglia, R. B. Slaoui, and S. Toth, "Estimating risk and uncertainty in deep reinforcement learning," *arXiv preprint arXiv:1905.09638*, 2019.

[22] H. Eriksson, D. Basu, M. Alibeigi, and C. Dimitrakakis, "Sentinel: taming uncertainty with ensemble based distributional reinforcement learning," in *Uncertainty in Artificial Intelligence*. PMLR, 2022, pp. 631–640.

[23] G. Liu, Y. Luo, O. Schulte, and P. Poupart, "Uncertainty-aware reinforcement learning for risk-sensitive player evaluation in sports game," *Advances in Neural Information Processing Systems*, vol. 35, pp. 20 218–20 231, 2022.

[24] T. Kanazawa, H. Wang, and C. Gupta, "Distributional actor-critic ensemble for uncertainty-aware continuous control," in *2022 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2022, pp. 1–10.

[25] M. Fortunato, M. G. Azar, B. Piot, J. Menick, I. Osband, A. Graves, V. Mnih, R. Munos, D. Hassabis, O. Pietquin *et al.*, "Noisy networks for exploration," *arXiv preprint arXiv:1706.10295*, 2017.

[26] G. Szegö, "Measures of risk," *Journal of Banking & finance*, vol. 26, no. 7, pp. 1253–1272, 2002.

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

[28] M. Towers, J. K. Terry, A. Kwiatkowski, J. U. Balis, G. d. Cola, T. Deleu, M. Goulão, A. Kallinteris, A. KG, M. Krimmel, R. Perez-Vicente, A. Pierré, S. Schulhoff, J. J. Tai, A. T. J. Shen, and O. G. Younis, "Gymnasium," Mar. 2023. [Online]. Available: https://zenodo.org/record/8127025