Uncertainty Aware Distributional Ensemble Reinforcement Learning for Flight Control

Micah Bryant, Joseph Gleason, Anastacia MacAllister

General Atomics ASI

Poway, CA

micah.bryant@ga-asi.com, joseph.gleason@ga-asi.com, anastacia.macallister@ga-asi.com

ABSTRACT

In recent years, artificial intelligence (AI) and machine learning (ML) have become important research foci for Department of Defense. A critical aspect of this heightened interest in AI is the development of fully autonomous, unmanned aerial system(s) (UAS). However, the inherent challenges posed by the complexity of operational environments in which these UAS operate present significant hurdles for traditional control algorithms. Reinforcement Learning (RL) offers a solution that will enable warfighters to approximate optimal actionable control strategies to achieve mission success in complex operations. Despite its potential, however, typical RL has limited ability to understand risk, which has raised concerns regarding its stability and safety in high-risk environments.

In this paper, we examine how a combination of distribution and ensemble methods can help improve RL agents understanding of risk and uncertainty. By understanding the distribution of environment rewards, we can utilize risk-aware metrics to improve safety and stability. With ensemble methods we can isolate the uncertainty due to limitation in exploration and knowledge, giving agents an estimate of their situational awareness. We can improve an agent's awareness by pushing it to explore areas it is uncertain about more and we can prevent, by simple throttling, agents from taking actions when their uncertainty is too high. Our paper will compare the combined methods against a baseline of soft-actor critic algorithm in a sample UAS dynamical environment built with a commercial off the shelf game engine.

ABOUT THE AUTHORS

Micah Bryant is a Machine Learning Engineer for General Atomics Aeronautical Systems. His work focuses on architecting, implementing, and validating reinforcement learning agents for dynamic air-to-air combat engagements. He holds a B.S. in Bioengineering from UCLA and his M.S. in Mechanical Engineering from UCSD.

Joseph D. Gleason, Ph.D., is a Machine Learning Engineer with General Atomics Aeronautic Systems. His work focuses on applied controls for dynamical systems and reinforcement learning algorithms for safety-critical systems. He has a B.S. in Electrical Engineering from Arizona State University and a Ph.D. in Electrical Engineering from the University of New Mexico.

Anastacia MacAllister, Ph.D., is Technical Director of Autonomy and Artificial Intelligence for General Atomics Aeronautical Systems (GA-ASI). Her work focuses on prototyping and developing novel machine learning algorithms using sparse, heterogenous, or imbalanced data sets, and exploratory data analytics. She has provided key contributions in prognostic health management, human performance augmentation, advanced sensing, and artificial intelligence for future warfare strategies. Dr. MacAllister has published over two dozen peer reviewed technical papers, conference proceedings, and journal. Dr. MacAllister holds a B.S from Iowa State University (ISU) in Mechanical Engineering, and an M.S. and Ph.D. from ISU in Mechanical Engineering and Human-Computer Interaction.

Uncertainty Aware Distributional Ensemble Reinforcement Learning for Flight Control

Micah Bryant, Joseph D. Gleason, Anastacia MacAllister
General Atomics
Poway, CA
micah.bryant@ga-asi.com, joseph.gleason@ga-asi.com, anastacia.macallister@ga-asi.com

INTRODUCTION

Artificial Intelligence (AI) and Machine Learning (ML) have become key research areas for the Department of Defense (DoD), as evidenced by the President's Fiscal Year 2024 request for over $1.8 billion, marking a 50% increase from the previous year's allocation. A critical aspect of this heightened investment in AI is the development of fully autonomous, unmanned systems. In this paper, we focus on the application of AI, in particular Reinforcement Learning (RL), for the development of unmanned aerial system(s) (UAS). There are many methods for designing control strategies UASs. In an attempt to categorize, we often separate control strategies into two categories: direct solutions and sampling-based methods.

Direct solution methods have been a staple of control theory and mathematics and rely on the ability to solve an optimal control problem explicitly or to some numerical approximation. There are myriad methods for solving problems of varying complexity. However, direct solutions often require restrictive assumptions, for example convexity and/or linearity, to the problem that have limited their applications for the warfighter in more complex scenarios. For example, the common suite of linear quadratic solution methods are popular regulation tools that are restricted to linear systems with quadratic cost functions. While this can be helpful to position landing gear on a plane to a given set point, it cannot be used to solve even basic constant-altitude flight controls because of the restriction to linear systems. On the other hand, sampling-based methods—e.g. genetic algorithms, particle swarm optimization—remove restrictive assumptions but often take too long to compute solutions to be effective in many modern operational environments, which can require frequent updating.

In the past two decades, improvements in the computational methods of neural-network-based artificial intelligence (now commonly just called AI) have demonstrated potential for AI to be a powerful tool for generating parametrized function approximations that can solve a diverse set of problems from categorization, regression, prediction, and, in the interest of this paper, control. To develop control policies to assist the warfighter, we rely on the recent advanced in reinforcement learning (RL). RL is also a sample-based method. However, RL has some unique advantages over other sample-based optimizations: 1) RL is (comparatively) computationally efficient at tuning models with a very large number of parameters. This enables RL to create models capable of representing highly complex behavior. 2) RL is capable of being trained in simulated scenarios and moved to, and potentially quickly retrained on, real-world systems. 3) As a parameterized model, RL can be used on real-time systems. These capabilities make RL uniquely qualified to assist the warfighter in complex mission operations. It has been notably applied in the area of games (Silver, et al., 2017) (Schrittwieser, et al., 2020) (Berner, et al., 2019), automated driving (Kiran, et al., 2022) (Cao, et al., 2023) (Wang, et al., 2023) (Jebessa, Olana, Getachew, Isteefanos, & Mohd, 2022), robotics (Ibar, et al., 2021) (Toyota Research Institute Unveils Breakthrough in Teaching Robots New Behaviors, 2023) (Toner, Saez, Tilbury, & Barton, 2023), and is an active area of research interest in academia, government, and industry.

Despite the advances, RL has seen limited real-world application, especially in the realm of safety-critical systems. This is because traditional RL methods have very limited capability of understanding risk and uncertainty. This can lead systems to risky operational behaviors or systems that will behave erratically because they encounter a state which they did not experience in training, commonly called an out-of-distribution problem. In closed physical environments this will typically only cause harm to the system on which it is applied. But in less regulated environments this has led to harm and/or death to operators and bystanders of the system (Hawkins, 2024). These concerns have

understandably limited their utility in aerial environments, even more so in operational environments for the warfighter where safety is paramount.

In this paper, we apply advances in RL algorithms that seek to address the UAS' ability to understand risk and uncertainty. These advances utilize ensemble (Song, et al., 2023) and distributional (Bellemare, Dabney, & Munos, 2017) RL extensions to the typical soft actor-critic (SAC) RL algorithm (Haarnoja, Zhou, Abbeel, & Levine, 2018). We demonstrate that by using these advanced methods, we can start to create systems that have an improved capacity to understand risk and uncertainty, all without requiring precise tuning of reward functions by developers of the control algorithms. This can lead to UAS control policies that are better designed to assist the warfighter in operational environments.

In the next section, we will provide some background to RL and how it compares to ensemble RL (ERL) and distributional RL (DRL). We will then describe our methods and experimental setup, followed by results, discussion, and impact on the warfighter.

BACKGROUND

In this section we detail traditional reinforcement learning and the updates—ensemble and distributional RL—used in this work. Common to all these methods, RL is a sample-based unconstrained optimization procedure that seeks to find an optimal control policy that maximizes some function of the reward (or minimizes the cost). In each we consider the dynamical system to be a Markov Decision Process (MDP), which is described by the tuple (S, A, P, r, γ), where S is the state space, A is the action space, $P: S \times A \to S$ is the state transition function, i.e. given a state and action it provides the next state according to some probability distribution, $r: S \times A \to \mathbb{R}$ is the reward function (or negative cost function), and $\gamma \in [0, 1)$ is the discount factor. Typically, MDPs are considered to represent discrete-time systems. While continuous MDP variants exist (Puterman, 2005), we will also assume a discrete system. We denote the state and action at time t as s_t and a_t, respectively.

In general, we would like to determine an optimal feedback control strategy $\pi: S \to A$ over some policy space Π. We denote the optimal policy as π^*. In practice, search over the space of all feedback policies is intractable, so, as with current RL methods, we search over a parametrized policy space Π_θ, where $\theta \in \Theta$ is a parameter vector and Π_θ is derived by the structure of the neural network we use. In the remainder of this paper, we will often use π to refer to π_θ for notational simplicity unless it is important to distinguish the difference.

Traditional reinforcement learning seeks to optimize the expected return over some (typically infinite) time-horizon:

$$\text{maximize}_{\theta \in \Theta} \quad \mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^i r(s_i, a) \,|a \sim \pi(s_i)\right] \tag{1}$$

This has several computational and theoretical advantages. Under some conditions, RL is guaranteed to converge to the optimal policy π^* (Sutton & Bartow, 2018). However, because it maximized the expected return, it can be susceptible to taking risky behavior if the reward for the behavior is sufficiently large.

Related Work

Previous work has examined combining ensemble and distributional RL methods (Hoel, Wolff, & Laine, 2023) (Williams, Gee, MacDonaland, & Liarokapis, 2024) (Eriksson, Basu, Alibeigi, & Dimitrakakis, 2022), however to the best of the authors' knowledge, there have been no works focusing on combined distributional and ensemble RL on UAS. In (Seres, Liu, & Kampen, 2023), the authors examine risk-sensitive distributional RL for flight control systems but do not examine the combination of ensemble and distributional RL. Both (Eriksson, Basu, Alibeigi, & Dimitrakakis, 2022) and (Williams, Gee, MacDonaland, & Liarokapis, 2024), the authors examine combined distributional and ensemble RL but from the lens of theoretical extension and focus less on application. The authors of (Hoel, Wolff, & Laine, 2023) examine applied ensemble and distributional RL for autonomous driving which, while similar, faces different challenges than the warfighter experiences in flight operations.

Distributional RL

For simulated systems, e.g. games, the maximization of the expected reward done by traditional RL may be acceptable. However, in real-world systems, e.g. UAS, this average risk tolerance may be unacceptable. From (1) we define the Q-function (Sutton & Bartow, 2018)

$$\begin{aligned} Q(s,a) &= \mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^i r(s_i,a) \,|a{\sim}\pi(s_i)\right] \\ &= \mathbb{E}[r(s,a)|\, a{\sim}\pi(s)] + Q(s',a') \end{aligned}$$

where $s'{\sim}P(s,a)$ and $a'{\sim}\pi(s')$. This is called the Bellman equation under policy π. We can also see that traditional RL seeks to maximize Q over the set of parameters θ.

For distributional RL, we instead examine the distributional Bellman equation:

$$Z(s,a) \overset{D}{=} r(s,a) + Z(s',a')$$

The above is an equality in distribution where s' and a' are defined as before.

The challenge with optimizing Q, or optimizing for the expectation, is that it can cause the RL system to strike an, often, unintentional balance between reward and risk. Consider the following example: A user can choose to press either button A or B. If A is pressed, the user receives a reward of 1. If B is pressed, then with probability $p = 0.5$, the user receives a reward of 4, and receives a reward of -1 otherwise. On average, i.e. from the perspective of Q, the best choice is to press B, whose average reward is 1.5. However, if we are concerned with determining optimal conditions that, for example, maximize the minimum possible return, then the optimal choice becomes A. While reward optimization with simple numbers is of little consequence, the warfighter, and systems that assist the warfighter, must be more considerate of the possibility for negative outcomes.

It is possible to reshape reward functions to manipulate how an RL agent will consider risk in the system but that usually requires time-consuming tuning of reward from subject matter experts. Distributional RL has the ability to remove some of these requirements by enabling the system to consider different types of risk sensitive metrics, e.g. conditional value at risk (CVAR). We will discuss in more detail the risk-sensitive metrics we use when we describe the methods employed and testing.

There are several ways to approximate a distribution over the return, and the one that is the most prevalent is a quantile representation. In this representation, the distribution function made of quantiles can be represented by $Z_\tau^\pi(s,a) \coloneqq \inf\{z \in \mathbb{R}: \tau \leq F_Z(z)\}$ where τ is the quantile fraction. The quantile fractions map to $\tau \in [0,1]$ and can be generated in several ways such as creating a set of equidistant quantiles as can be found in QR-DQN (Dabney, Rowland, Bellemare, & Munos, 2017) or randomly sampled as in IQN (Dabney, Ostrovski, Silver, & Muno, 2018).

The primary challenge with distributional RL is the increased complexity in the representing the distribution over the expectation. The expected reward metric Q resolves to a real value while Z is generally numerically approximated. There is a tradeoff between the quality of the approximation of Z and how much computational effort is required during the optimization process.

Ensemble RL

There are many uses for ensemble methods (Song, et al., 2023) but the focus of this work is on using ensemble methods to create an approximate metric for the uncertainty of the RL network, specifically the epistemic uncertainty. In uncertainty quantification there are two primary categories of uncertainty, aleatoric and epistemic. Aleatoric uncertainty deals with intrinsic uncertainty in the dynamics of the system while epistemic uncertainty describes

uncertainty in knowledge. In general, both are extremely hard, if possible, to exactly quantify (Hüllermeier & Waegeman, 2021). However, ensemble learning methods have demonstrated promise in creating estimates of uncertainty that can be practically used in complex system (Hoel, Wolff, & Laine, 2023) (Song, et al., 2023) (Abdar, et al., 2021), for example UAS.

Ensemble RL is an extension on traditional RL (or distributional RL) where instead of optimizing over a single Q-function, we optimize over several of them, Q_m for $m \in \{0, 1, 2, \dots, n\}$. We may also have an ensemble of policies π_k for $k \in \{0, 1, 2, \dots, l\}$. While exact quantification of epistemic uncertainty is nontrivial, with multiple critic networks, we can create an estimate of the epistemic uncertainty by examining different statistics in the variation of the predictions from the various critics. For example, we can assess the uncertainty given a state and action by looking at the variance in the critic predictions:

$$U(s, a) = \text{var}\big(Q_m(s, a)\big)$$

Higher variance indicates more variation in the critic predictions which typically indicates an insufficient number of training samples to create alignment in the ensemble. Variance in the critic estimates does not decouple aleatoric and epistemic uncertainty, however this is not possible for general systems (Hüllermeier & Waegeman, 2021).

METHODS

In this section we describe, in more detail, our distributional and ensemble soft actor critic setup, then we will describe the example problem that we use to compare traditional soft actor critic, with our extension. While our experiment is conceptually simple, we expect, because of the explicit uncertainty in the problem, that the distributed methods will result in safer trajectories. In the example, safety is related to the agent's ability to avoid detection.

Distributional Ensemble Soft Actor Critic (DESAC)

The distributional soft actor critic method (DSAC) as outlined by (Ma, Xia, Zhou, Yang, & Zhao, 2020) was modified to add in multiple critics which are distributional Z-function approximators. There are a variable number of critic networks rather than a static amount of two. When calculating the expected return from the bellman equation the outputs are averaged:

$$\bar{Z}(s, a) \overset{D}{=} r(s, a) + \frac{\gamma}{N} \sum_{i=1}^{N} Z_i(s', a')$$

Where N is the number of critics, Z_i is the output from the output critic number i, and $\bar{Z}$ is the expected quantile distribution of the return. Target networks are used at this step with soft updates. The epistemic and aleatoric uncertainties are also calculated at this step and used to apply intrinsic rewards to the bellman equation.

We can approximate epistemic uncertainty as the variance in the quantile value outputs. Since all critics share the same target expectation, the variance across their predictions for the same input can provide an estimate of how certain we are in the prediction. If the variance is high, the networks have not converged which thereby signifies a low confidence in that prediction. This is calculated as shown below:

$$u_e = \frac{1}{N_q} \sum_{i=1}^{N_q} Var_N\big(Z(s, a)\big) \quad (2)$$

Where N_q is the total number of quantiles output from a single critic network. For epistemic uncertainty then, it is approximated as the variance across the networks that is averaged over the quantiles.

For aleatoric uncertainty, it is approximated by how spread out the quantile predictions are. As training progresses, if the quantiles all converge around a very small set of values then the certainty in the return is very high. However, if

the quantiles are extremely spread out then the inherent uncertainty in those states is high. This can be approximated as the variance among the quantiles that is averaged over the critics:

$$u_a = \frac{1}{N}\sum_{i=1}^{N} Var_{N_q}\big(Z(s,a)\big) \quad (3)$$

Both the epistemic uncertainty (u_e) and aleatoric uncertainty (u_a) are used in the expected return as intrinsic reward values. This results in the following equation:

$$\bar{Z}(s,a) \overset{D}{=} r(s,a) + w_e u_e(s,a) + w_a u_a(s,a) + \frac{\gamma}{N}\sum_{i=1}^{N} \tilde{Z}_i(s',a')$$

Typically, the weight for epistemic uncertainty (w_e) is positive to promote exploring states with higher epistemic uncertainty to allow confidence in those states to go up. The weight for aleatoric uncertainty (w_a) is typically negative to promote avoiding states with higher aleatoric uncertainty as that uncertainty is not able to be reduced by increased exploration. The loss for the critic networks is calculated using the quantile pinball loss as given by:

$$L_c = \begin{cases} \tau(z-\bar{z}), & z \geq \bar{z} \\ (\bar{z}-z)\,(1-\tau), & \bar{z} > z \end{cases}$$

Where L_c is the loss for this critic, $\bar{z}$ is the specific quantile predicted by the expected return at quantile τ. At this point new actions $\hat{a}$ are polled from the policy using the initial state:

$$\hat{a} = \pi(s)$$

These are used to calculate the predicted quantile values for the new actions. The quantiles then undergo a risk distortion and then is averaged across all critics.

$$Q(s,\hat{a}) = \frac{1}{N}\sum_{i=1}^{N} \psi\left(\hat{Z}(s,\hat{a})\right)$$

Where ψ is the risk distortion function. The distortion CVAR which is the portion β of the lowest value returns (Choi, Dance, Kim, Hwang, & Park, 2021). This can be expressed as:

$$\psi^{CVaR} = (\tau;\beta) \coloneqq \beta\tau$$

At this point, the Q value can be passed into the actor loss for the traditional soft actor critic loss function for the actor. Where the actor loss is:

$$L_a = \alpha \log\big(\pi(\hat{a}|s)\big) - Q(s,\hat{a})$$

Where α is the autotuned temperature parameter that weights the entropy term in the actor loss.

Experimental Environment

For our simulations, we utilize the Unity game engine with a discretized dynamics that are meant to represent an aircraft in motion. Similar to a Dubins vehicle, the agent has the ability to control the rate at which the aircraft turns (alters its heading). The agent chooses actions between $[-1,1]$, and these values are mapped to the minimum and maximum turn angles. We hold altitude constant.

As shown in Figure 1, the agent starts 45 nautical miles (nmi) east from 0 latitude and 0 longitude with a random heading between [-180, 180] from true north. Its objective is to pass the 0 (degree or nmi) longitude. The target can be reached at 0 longitude at any altitude.

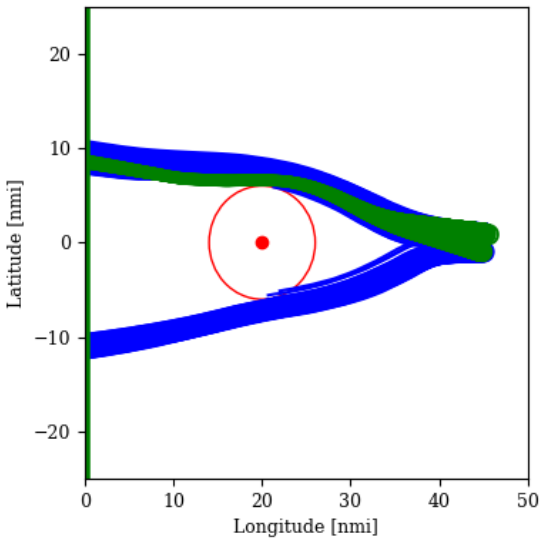

Figure 1: Comparison of SAC trajectories (orange) with DESAC trajectories (blue). The red dot is the center of the detector and the radius of the red circle is 6 nmi.

However, at $(0, 20)$ nmi there exists a radar that can detect the agent with increasing probability as the agent gets closer to the radar. The probability of detection uses the following Gaussian decay.

$$p_d = \exp\left(-\frac{\|(x,y) - (20,0)\|_2^2}{9}\right)$$

Where (x, y) is the distance of the agent in nmi from 0 latitude and 0 longitude. This detector could represent adversarial capabilities that pose a threat to the agent or counter measures that could put the agent / UAS / pilot at risk. If the agent is detected, the game ends. Thus, we want to develop and agent that intrinsically avoids this threat.

To incentivize the agent we give it a reward for closing distance to the target:

$$r_d = 0.05 * \exp\left(\frac{x^2}{100}\right)$$

If the agent reaches the target, it gets a reward of $r_t = 100$. This high reward is to incentivize the agent to reach the target instead of getting close but never fully reaching the target. We do not provide any explicit (i.e. in the reward function) influence on the agents to avoid the detector. The strictly positive reward creates a potential implicit bias to avoid detection as the agent can continue to collect rewards.

TESTING RESULTS & DISCUSSION

We trained two different agents on the experimental setup: The first is a standard SAC agent, and the second is the DESAC agent described in the methods section. The goal is to demonstrate two important features of the DESAC agent: first, that it can, through use of risk-aware metrics, provide safer control policies. Here, safer means reduced incidents of detection as the aircraft moves to the target. Second, that the agent can, by using its ensemble of critics provide an estimate of its epistemic uncertainty, enabling an autonomous agent to provide a numerical value of its confidence in its actions.

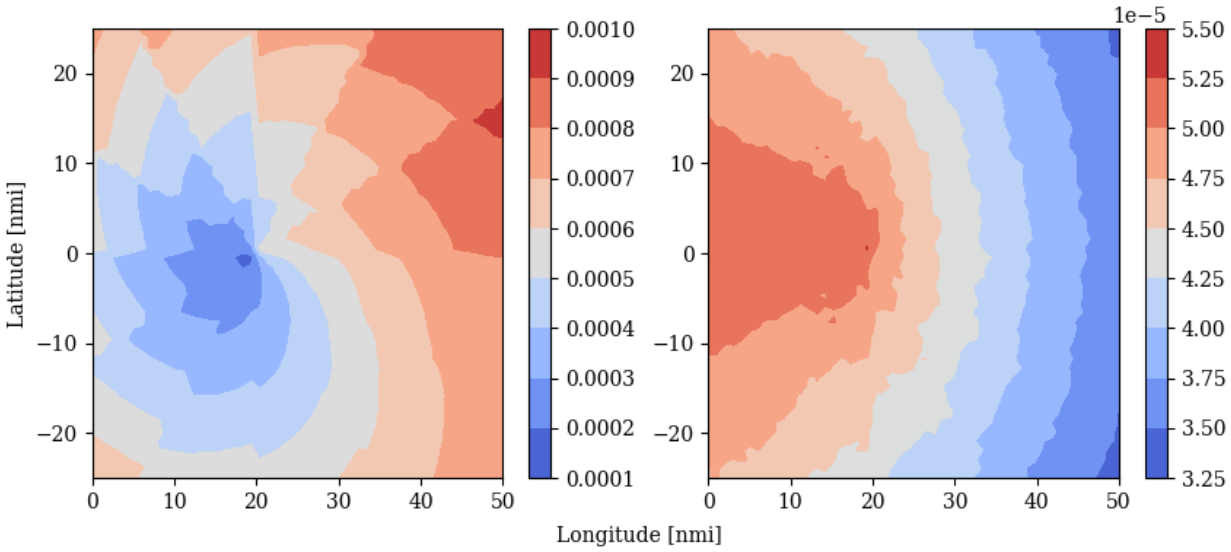

Figure 2: Epistemic (left) and aleatoric (right) uncertainty heat maps. Red represents areas of high uncertainty, and blue areas of low uncertainty.

While solutions to this problem can seem quite intuitive to us, due to detection happening probabilistically neither algorithm was able to solve the problem without the addition of a curriculum. Our curriculum consisted of three stages: starting with the denominator of the detection exponential at 1, increased to 4 after 200k steps, and finally to 9 after 400k steps. The agents were trained for approximately 1 million train steps.

Figure 1 shows example trajectories of the SAC agent vs the DESAC agent. Initially, DESAC and SAC take similar trajectories with DESAC closer to the detector. However, as both agents approach the detector, DESAC takes a wider arc. In the graph the red dot is the center of the detector, and the radius of the red circle is set to 6 nmi. This radius was chosen heuristically as many of the final detections happen around that distance. The following table shows the min, max, mean, variance in each agent's reward, and how likely each agent was to be detected from evaluation runs.

	SAC	DESAC
Min Reward	**2.665e-5**	1.455e-5
Mean Reward	76.286	**88.082**
Max Reward	**101.312**	101.311
Variance	1907.600	**1163.115**
Detection Probability	0.247	**0.130**

Table 1: Evaluation results comparison between SAC and DESAC.

These trajectory and table results align with our hypothesis that DESAC will, from its ability to use risk aware metrics, prefer safer trajectories. At a high level, DESAC is de-emphasizing trajectories with higher reward but also higher variance. Since the shortest paths would also cross through the detector, they are higher risk and high reward. SAC, on the other hand, seeks to optimize only the expectation. Thus, as SAC optimizes its trajectory it seeks a shorter path that has the potential for a higher payoff (as seen in the marginally higher maximum reward). Counter to our intuition, DESAC has a higher mean reward than SAC even though SAC seeks to maximize this metric specifically, see Table 1. We expect this may be a result of SAC having trouble converging on the optimum because of the balance between shorter paths and increased likelihood of detection.

In Figure 2 we show how the ensemble of critics could assess its own confidence in its control action. This confidence is determined by the estimate of epistemic and/or uncertainty from equations (2) and (3). In the figure, it is assumed that the heading of the agent is due west. From Figure 2, we see high epistemic uncertainty as we move away from the detector. This is because, although during initial training the agent explores more broadly, as the DESAC agent improves on its solution it spends much of its time exploring areas around the trajectories seen in Figure 1. In contrast,

the aleatoric uncertainty is higher near the detector. It remains high to the west of the detector likely due to the agent not exploring the region directly behind the detector as there is no incentive to do so.

When designing AI systems to aid the warfighter, we can use this uncertainty estimate to ensure that the agent only provides control actions in areas that it has sufficient confidence. For example, we could threshold our aircraft to only perform action when its epistemic uncertainty is below 0.0007. This has the potential to increase trust between autonomy and the warfighter. However, at the current state-of-the-art, determining an adequate threshold requires explicit tuning from subject matter experts.

For increased repeatability, we provide additional training hyperparameters used in Table 2.

Parameter	SAC	DESAC
Replay Buffer Size	150 000	150 000
β	N/A	0.7
γ	0.995	0.995
τ	0.005	0.005
α learning rate	0.00001	0.00001
Actor learning rate	0.00002	0.00002
Critic learning rate	0.0001	0.0001
Embedding size	N/A	64
Number Quantiles	N/A	32
Number Critics	N/A	2

Table 2: Training hyperparameters

CONCLUSIONS AND FUTURE WORK

In this work, we presented a distributional and ensemble extension to the SAC algorithm that seeks to imbue unmanned automated aerial systems with additional capacity to assess risk and uncertainty during their training. This gives these RL agents additional capabilities that are not available to a general RL agent without extensive tuning of reward function by subject matter experts. (And the ability to control risk through tuning of a smaller number of hyperparameters.) We believe that in complex scenarios, these additional capabilities will become necessary to ensure the safety of UAS more adequately in operational scenarios. Future work is necessary to determine how to better describe and optimize distributional models [e.g. Gaussian mixture models for distributional estimation (Choi, Lee, & Oh, 2019)], how to better quantify uncertainty so that it can be used to determine when a UAS may be eligible for control of a system, and either empirical or theoretical determination of a sufficient number of critics in an ensemble to provide adequate safety.

REFERENCES

Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., . . . Nahavandi, S. (2021). A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 243-297.

Bellemare, M. G., Dabney, W., & Munos, R. (2017). A Distributional Perspective on Reinforcement Learning. *34th International Conference on Machine Learning*, (pp. 449-458).

Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., . . . Zhang, S. (2019). Dota 2 with Large Scale Deep Reinforcement Learning. *arXiv*.

Cao, Z., Jiang, K., Zhou, W., Xu, S., Peng, H., & Yang, D. (2023). Continuous improvement of self-driving cars using dynamic confidence-aware reinforcement learning. *Nature*, 145-158.

Choi, J., Dance, C., Kim, J.-e., Hwang, S., & Park, K.-s. (2021). Risk-Conditioned Distributional Soft Actor-Critic for Risk-Sensitive Navigation. *IEEE*, 8337-8344.

Choi, Y., Lee, K., & Oh, S. (2019). Distributional Deep Reinforcement Learning with a Mixture of Gaussians. *2019 International Conference on Robotics and Automation (ICRA).* Montreal: IEEE.

Dabney, W., Ostrovski, G., Silver, D., & Muno, R. (2018). Implicit Quantile Networks for Distributional Reinforcement Learning. *Proceedings of Machine Learning Research* (pp. 1096-1105). PMLR.

Dabney, W., Rowland, M., Bellemare, M. G., & Munos, R. (2017). Dabney, Will, Mark Rowland, Marc G. Bellemare and Rémi Munos. "Distributional Reinforcement Learning with Quantile Regression." AAAI Conference on Artificial Intelligence (2017). *Conference on Artificial Intelligence.* AAAI.

Eriksson, H., Basu, D., Alibeigi, M., & Dimitrakakis, C. (2022). SENTINEL: taming uncertainty with ensemble based distributional reinforcement learning. *38th Conference on Uncertainty in Artificial Intelligence*, (pp. 631-640).

Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. *arXiv*.

Hawkins, A. J. (2024, April 26). *Tesla's Autopilot and Full Self-Driving linked to hundreds of crashes, dozens of deaths.* Retrieved from The Verge: https://www.theverge.com/2024/4/26/24141361/tesla-autopilot-fsd-nhtsa-investigation-report-crash-death

Hoel, C.-J., Wolff, K., & Laine, L. (2023). Ensemble Quantile Networks: Uncertainty-Aware Reinforcement Learning With Applications in Autonomous Driving. *IEEE Transactions on Intelligent Transportation Systems*.

Hüllermeier, E., & Waegeman, W. (2021). Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. *Machine Learning*, 457-506.

Ibar, J., Tan, J., Finn, C., Kalakrishnan, M., Pastor, P., & Levine, S. (2021). How to Train Your Robot with Deep Reinforcement Learning; Lessons We've Learned. *arXiv*.

Jebessa, E., Olana, K., Getachew, K., Isteefanos, S., & Mohd, T. K. (2022). Analysis of Reinforcement Learning in Autonomous Vehicles. *2022 IEEE 12th Annual Computing and Communication Workshop and Conference.* IEEE.

Kanazawa, T., Wang, H., & Gupta, C. (2022). Distributional Actor-Critic Ensemble for Uncertainty-Aware Continuous Control. *arXiv*.

Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Sallab, A. A., Yogamani, S., & Pérez, P. (2022). Deep Reinforcement Learning for Autonomous Driving: A Survey. *IEEE Transactions on Intelligent Transport Systems*, 4909-4926.

Ma, X., Xia, L., Zhou, Z., Yang, J., & Zhao, Q. (2020). DSAC: Distributional Soft Actor Critic for Risk-Sensitive Reinforcement Learning. *arXiv*.

Puterman, M. L. (2005). *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* Wiley-Interscience.

Schrittwieser, J., Antonoglou, I., Hubert, T., Sifre, L., Simonyan, K., Schmitt, S., . . . Silver, D. (2020). Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 604-609.

Seres, P., Liu, C., & Kampen, E.-J. v. (2023). Risk-sensitive Distributional Reinforcement Learning for Flight Control. *IFAC-PapersOnLine*, 2013-2018.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., . . . Hassabis, D. (2017). Mastering the game of Go without human knowledge. *Nature*, 354-359.

Song, Y., Suganthan, P. N., Pedrycz, W., Ou, J., He, Y., Chen, Y., & Wu, Y. (2023). Ensemble reinforcement learning: A survey. *Applied Soft Computing*.

Sutton, R. S., & Bartow, A. G. (2018). *Reinforcement Learning: An Introduction.* MIT Press.

Toner, T., Saez, M., Tilbury, D. M., & Barton, K. (2023). Opportunities and challenges in applying reinforcement learning to robotic manipulation: An industrial case study. *Manufacturing Letters*, 1019-1030.

Toyota Research Institute Unveils Breakthrough in Teaching Robots New Behaviors. (2023, September 19). Retrieved from Toyota Newsroom: https://pressroom.toyota.com/toyota-research-institute-unveils-breakthrough-in-teaching-robots-new-behaviors/

Wang, L., Liu, J., Shao, H., Wang, W., Chen, R., Liu, Y., & Waslander, S. L. (2023). Efficient Reinforcement Learning for Autonomous Driving with Parameterized Skills and Priors. *arXiv*.

Williams, D. V., Gee, T., MacDonaland, B. A., & Liarokapis, M. (2024). CTD4 - A Deep Continuous Distributional Actor-Critic Agent. *arXiv*.