Uncertainty-Driven Distributional Reinforcement Learning for Flight Control

Homola, M.; Li, Y.; van Kampen, E.

DOI
10.2514/6.2025-2793

Publication date
2025

Document Version
Final published version

Published in
Proceedings of the AIAA SCITECH 2025 Forum

AIAA SciTech Forum
6-10 January 2025, Orlando, FL
AIAA SCITECH 2025 Forum

10.2514/6.2025-2793

Uncertainty-Driven Distributional Reinforcement Learning for Flight Control

Marek Homola*, Yifei Li† and Erik-Jan van Kampen ‡
Massachusetts Institute of Technology, Cambridge, Massachusetts, 02139, USA
Delft University of Technology, P.O. Box 5058, 2600GB Delft, The Netherlands

In the rapidly evolving aviation sector, the quest for safer and more efficient flight operations has historically relied on traditional Automatic Flight Control Systems (AFCS) based on high-fidelity models. However, such models not only incur high development costs but also struggle to adapt to new, complex aircraft designs and unexpected operational conditions. As an alternative, deep Reinforcement Learning (RL) has emerged as a promising solution for model-free, adaptive flight control. Yet, RL-based approaches pose significant challenges in terms of sample efficiency and safety assurance. Addressing these gaps, this paper introduces Returns Uncertainty-Navigated Distributional Soft Actor-Critic (RUN-DSAC). Designed to enhance the learning efficiency, adaptability, and safety of flight control systems, RUN-DSAC leverages the rich uncertainty information inherent in the returns distribution to refine the decision-making process. When applied to the attitude tracking task on a high-fidelity, non-linear fixed-wing aircraft model, RUN-DSAC demonstrates superior performance in learning efficiency, adaptability to varied and unforeseen flight scenarios, and robustness in fault tolerance that outperforms the current state-of-the-art SAC and DSAC algorithms.

Nomenclature

$\mathcal{S}$	=	State-space
$\mathcal{A}$	=	Action-space
$\mathcal{R}$	=	Reward mapping
$R(s,a)$	=	Reward function as a function of state s and action a
$\mathcal{P}$	=	Stochastic state transition dynamics
γ	=	Discount factor
Q^π	=	Action-value function under policy π
π^*	=	Optimal policy
$\mathcal{H}$	=	Entropy of the policy
$\mathcal{L}_\pi$	=	Policy loss function
Z	=	Random variable representing the return distribution
τ	=	Quantile level
Ψ	=	Risk measure function
μ, σ_Q	=	Parameters in RUN-DSAC for handling uncertainty
Q_θ^V	=	Modified action-value function incorporating uncertainty
V	=	True airspeed [m/s]
α	=	Angle of attack [deg]
β	=	Angle of sideslip [deg]
ϕ, θ, ψ	=	Roll, pitch, and yaw angles, respectively [deg]
p, q, r	=	Roll, pitch, and yaw rates, respectively [deg/s]
X_e, Y_e, h	=	Translational positions in the local tangent plane (horizontal coordinates and altitude) [m]
$\delta_e, \delta_a, \delta_r$	=	Elevator, aileron, and rudder deflections, respectively [deg]

*Graduate Student, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology
†Graduate Student, Faculty of Aerospace Engineering, Delft University of Technology
‡Associate Professor, Faculty of Aerospace Engineering, Delft University of Technology

I. Introduction

In the dynamically advancing world of aviation, ensuring the safety and efficiency of flight operations remains paramount. Traditional automatic flight control systems (AFCS) have been the cornerstone of aviation for decades, leveraging established control algorithms and techniques. Yet, these systems are increasingly limited by their dependence on intricate, high-fidelity models of system dynamics [1]. Developing, validating, and verifying these models is resource-intensive, involving careful system identification through computational simulations, scaled wind tunnel measurements, and flight tests [2]. Moreover, this challenge escalates with growing complexity in the design and control of novel aerospace configurations, such as V-shaped flying wings [3, 4], vertical take-off and landing (VTOL) systems [5, 6], aircraft with morphing wings [7, 8], ornithopters [9, 10], and designs with high-aspect-ratio wings introducing significant dynamics non-linearity due to increased structural flexibility [11, 12]. Hence, there is an intensifying urgency for model-free control techniques that streamline the development cycle and minimize the reliance on extensive model identification and validation. Furthermore, traditional AFCS, which are constrained by predetermined design assumptions, are susceptible to failure in unforeseen situations like adverse weather or system malfunctions [13, 14]. Incidents, such as the 2009 Air France Flight 447 crash [15], emphasize the critical need for more intelligent and adaptable autonomous aerospace systems that can ensure safe operation under uncertain circumstances.

Extensive research has focused on reducing model fidelity requirements to enable more autonomous systems. Robust control techniques like H_∞-synthesis [16, 17] ensure closed-loop stability even in the presence of model uncertainties and disturbances, but at the expense of very conservative control performance [18, 19]. Nonlinear Dynamic Inversion (NDI) offers a popular alternative by inverting system dynamics to formulate control laws [20]. However, its limitations lie in sensitivity to modeling errors, uncertainties, and the accuracy of the system's state estimates. Sensor-based approaches, such as Incremental NDI (INDI) [21] and Incremental Backstepping (IBS) [22], mitigate the dependence on global models and instead rely on incremental control models derived from sensor measurements. This approach improves robustness to model imperfections, increasing adaptability and fault tolerance [23–25]. Nevertheless, INDI and IBS methods face challenges related to sensor synchronization and filtering. This research proposes an alternative solution based on bio-inspired Artificial Intelligence — *Reinforcement Learning (RL)*.

Deep RL, leveraging deep neural networks (DNN) to approximate complex functions [26], offers a promising avenue for designing autonomous flight control systems capable of learning without explicit a-priori knowledge of system dynamics. These Deep RL agents adaptively refine their actions through continuous interaction with the environment in order to achieve objectives such as trajectory tracking or fuel economy [27]. Yet, implementing RL in flight control presents notable challenges. First, RL algorithms necessitate extensive data samples to converge, which becomes even more challenging due to the complexity and high dimensionality of flight control tasks [28]. Secondly, the safety-critical nature of flight control raises concerns regarding the reliability and safety of the converged RL control policies [29].

Several methods have been proposed to address these challenges. Online incremental Approximate Dynamic Programming (ADP) methods like Incremental Dual-Heuristic Programming (IDHP) demonstrate sample-efficient online learning and adaptive control but face generalization and dimensionality limits [30, 31]. Hierarchical RL flight controllers offer a solution to dimensionality but at the cost of complicating policy learning and added hyperparameter complexity [28, 32]. In the domain of safety, Shielded RL introduces a specialized 'shield' controller to regulate flight path angles, yet its reliance on predefined safety rules restricts adaptability [33]. Alternatively, the Soft Actor-Critic (SAC) algorithm has shown potential for robust flight control and adaptability to unforeseen failures but suffers from training inconsistencies and low convergence success rates [34]. Building upon SAC, Distributional SAC (DSAC) was introduced, learning the full returns distribution rather than just the expected value. DSAC's improved sample efficiency and robustness in flight control [35, 36] serve as the impetus for this research.

In conventional DSAC algorithms, action selection is governed by the mean of the learned return distribution [37–39]. However, this underutilizes the valuable information embedded in the distribution. To fill this void, previous studies have explored using these distributions to distort the expectations and synthesize risk-sensitive policies, improving the safety of RL-based flight controllers [35, 40]. Building on these foundations, we introduce *Returns Uncertainty-Navigated DSAC* (RUN-DSAC), a novel variant designed to exploit the returns uncertainty for more informed decision-making. Our methodology offers an alternative to other uncertainty-aware approaches, such as Bayesian methods, which suffer from scalability issues [41] and rely on an accurate selection of a prior distribution [42–44].

This research advances intelligent flight control systems through four key contributions: First, it extends previous research [35] by validating DSAC's superior learning efficiency and stability over SAC in flight control, while exploring its robustness in previously untested generalization scenarios and fault conditions. More importantly, we introduce the novel uncertainty-driven RUN-DSAC algorithm, enabling the synthesis of sample-efficient and safe flight control policies. Thirdly, the study reveals that prioritizing low-uncertainty state-action pairs during learning markedly improves

both learning performance and exploration safety, resulting in superior tracking performance with improved fault tolerance and robustness to unseen flight conditions. Lastly, we show that while favoring high-uncertainty may hinder learning performance, it leads to policies with heightened resilience to perturbations in unexpected flight conditions.

II. Background

First, the core principles of RL need to be defined, including the notions of Maximum Entropy RL and Distributional RL. Additionally, the flight control task is formalized within the context of RL.

A. Fundamentals of Deep Reinforcement Learning

RL represents a bio-inspired machine learning methodology that relies on an iterative trial-and-error mechanism for deriving optimal control policies. In this framework, an autonomous agent incrementally refines its decision-making proficiency in a specific task domain through continuous interactions with the environment [27]. This sequential decision-making process is conceptualized as a Markov Decision Process (MDP), defined by the set $\mathcal{M} \sim \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma \rangle$. Here, $S \subset \mathbb{R}^n$ defines the state-space, $\mathcal{A} \subset \mathbb{R}^m$ the action-space, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ the reward mapping, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ the stochastic state transition dynamics, and $\gamma \in (0, 1)$ the discount factor. At each discrete time-step t, the agent selects an action $a_t \in \mathcal{A}$ according to the policy $a \sim \pi(\cdot|s)$ depending on the current state s_t. It then observes the state-transition tuple $T_t = \langle s, a, r, s' \rangle$, with the instantaneous reward $r = R(s, a)$ and the subsequent state s'.

The agent's objective is to identify the optimal policy π^* that maximizes the expected *return*, defined as the expected sum of discounted rewards over a sequence of decisions. This objective is facilitated by employing $Q^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, which estimates the expected return from the given state s upon selecting action a and thereafter following policy $\pi \in \Pi$, as given by Equation 1.

$$Q^\pi(s, a) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right] \mid a_t \sim \pi(\cdot|s_t),\ s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t),\ s_0 = s,\ a_0 = a \tag{1}$$

To enable systematic convergence through iterative algorithms, Q^π can be formulated using the contractive Bellman equation [45], as given by Equation 2.

$$\mathcal{T}^\pi Q(s, a) = \mathbb{E}[R(s, a)] + \gamma \mathbb{E}_{\mathcal{P},\pi} \left[Q(s', a') \right] \tag{2}$$

Deep RL integrates DNNs for function approximation in both value-based and policy-based RL frameworks, as well as in hybrid actor-critic architectures. Value-based methods like DQN use DNNs to estimate the action-value function $Q_\theta(s, a) \approx Q(s, a)$, thus implicitly defining the policy, predominantly in discrete action spaces [46, 47]. Conversely, policy-based approaches directly approximate the optimal policy $\pi_w(s) \approx \pi^*(s)$ and are suited for continuous action spaces, albeit at the cost of higher variance and slower convergence [48]. Nonetheless, recent advancements such as TRPO and PPO have partly mitigated these limitations [49, 50]. Actor-critic methods synergize the strengths of both approaches - the critic evaluates the action-value function, providing an estimate of the expected return that aids in reducing the variance of the policy optimized by the actor [51]. This enables efficient handling of complex, high-dimensional, continuous control tasks. State-of-the-art actor-critics, such as SAC, have demonstrated robust performance and fast convergence in intricate control environments [52].

B. Soft Actor-Critic (SAC)

In contrast to traditional RL that solely focuses on maximizing cumulative reward, *Maximum Entropy RL* methods like SAC aim to optimize a modified objective function that balances reward accumulation with policy entropy maximization, as given by Equation 3. Maximizing the entropy term $\mathcal{H}(\pi_w(\cdot|s_t))$ promotes more effective exploration of the state space and increases policy robustness.

$$J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t \left[R(s_t, a_t) + \alpha_T \mathcal{H}(\pi_w(\cdot|s_t)) \right] \right] \quad with \quad \mathcal{H}(\pi_w(\cdot|s)) = \mathbb{E}_{a \sim \pi_w} \left[-\log(\pi_w(a|s)) \right] \tag{3}$$

This leads to the formulation of a *soft Bellman equation*, as given by Equation 4. Here, α_T is a temperature parameter that controls the trade-off between maximizing the expected return and maximizing the entropy, thereby influencing the degree of stochasticity and exploration in the policy π.

$$\mathcal{T}_S^\pi Q(s, a) = \mathbb{E}\left[R(s, a) \right] + \gamma \mathbb{E}_{\mathcal{P},\pi} \left[Q(s', a') - \alpha_T \log \pi(a'|s') \right] \tag{4}$$

Haarnoja et al. [53] introduce dynamic adaptation of α_T to attain a specified target entropy $\bar{\mathcal{H}}$, thereby improving exploration efficiency and reducing hyperparameter sensitivity. The target entropy $\bar{\mathcal{H}}$ is typically set in correspondence with the action space dimension, $\bar{\mathcal{H}} = -m$. The loss function for optimizing this adaptive α_T is detailed in Equation 5. In this formulation, $\mathcal{B}$ denotes a mini-batch of transitions $\mathcal{T}$ drawn from the experience replay buffer $\mathcal{D} = \{\mathcal{T}_0, \mathcal{T}_1, ...\}$, which is characteristic for off-policy RL methods.

$$\mathcal{L}^{\mathcal{B}}(\alpha_T) = \mathbb{E}_{\mathcal{B}}\left[\alpha_T \bar{\mathcal{H}} - \alpha_T \log \pi_w(a|s)\right] \tag{5}$$

To mitigate action-value function overestimation, SAC employs a double-critic architecture [54]. This entails training two parallel Q-functions, $Q_{\theta_{1,2}}(s,a)$, and computing the temporal-difference (TD) error δ based on the minimum of the two Q-values (Equation 6). Alongside these *behavioral* Q-networks, SAC also maintains *target* Q-networks, parameterized by $\bar{\theta}$. These networks undergo slower updates, achieved through incremental adjustments via Polyak averaging with a step size ζ, thereby enhancing learning stability.

$$\mathcal{L}_Q^{\mathcal{B}}(\theta_i) = \mathbb{E}_{\mathcal{B}}[\delta_i^2] \quad with \quad \delta_i = r + \gamma(1-d)\left(\min_{i=1,2} Q_{\bar{\theta}_i}(s',a') - \alpha \log \pi_w(a'|s')\right) - Q_{\theta_i}(s,a), \quad a' \sim \pi_w(\cdot|s') \tag{6}$$

In SAC, the actor employs a stochastic policy represented by an m-dimensional multivariate Gaussian distribution with diagonal covariance, where the mean vector $\mu_w \in \mathbb{R}^m$ and covariance diagonal $\sigma_w \in \mathbb{R}^m$ are modeled by a DNN with parameters w. Furthermore, the *reparameterization* trick is employed to facilitate backpropagation through the stochastic policy [53], where the action a is reparametrized as a deterministic function of the policy parameters and an independent Gaussian noise variable ϵ. To bound action domain, a *tanh* squashing function is applied post-sampling, as in Equation 7. While the actor is stochastic during training, it opts for deterministic actions in the evaluation phase by selecting the mean vector μ_w.

$$a_w(s) = \tanh\left(\tilde{a}_w(s)\right) \quad with \quad \tilde{a}_w(s) \sim \mathcal{N}(\mu_w(s), \sigma_w(s)) \tag{7}$$

Finally, the policy loss function used to update the actor's parameters is detailed in Equation 8.

$$\mathcal{L}_\pi^{\mathcal{B}}(w) = \mathbb{E}_{\mathcal{B}}\left[\alpha \log \pi_w(a_w(s)|s) - \min_{i=1,2} Q_{\theta_i}(s, a_w(s))\right] \tag{8}$$

C. Distributional Reinforcement Learning

Unlike classical RL's focus on optimizing the expected sum of rewards, *Distributional RL* aims to learn the full probability distribution over returns. This is formalized as $Z : \mathcal{S} \times \mathcal{A} \to \mathcal{Z}$, where $\mathcal{Z}$ denotes the space of action-value distributions with finite moments for each state-action pair, as defined by Equation 9 [55].

$$\mathcal{Z} \coloneqq \left\{Z : \mathcal{S} \times \mathcal{A} \to \mathscr{P}(\mathbb{R}) \mid \mathbb{E}\left[\|Z(s,a)\|_p\right] < \infty \quad \forall (s,a),\ p \geq 1\right\} \tag{9}$$

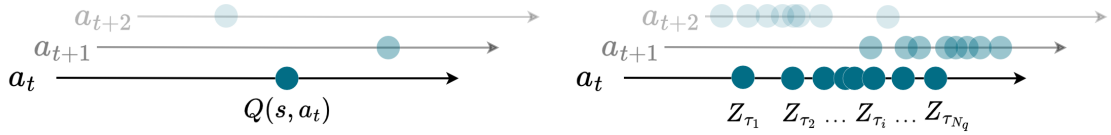
This leads to the formulation of the *distributional Bellman operator* $\mathcal{T}_D^\pi : \mathcal{Z} \to \mathcal{Z}$ [37], as articulated in Equation 10. Here, the notation $X \overset{\mathcal{D}}{=} Y$ signifies that random variables X and Y follow identical distributions.

$$\mathcal{T}_D^\pi Z(s,a) \overset{\mathcal{D}}{=} R(s,a) + \gamma Z(s',a') \tag{10}$$

The $\mathcal{T}_D^\pi$ operator is established to be contractive under a p-Wasserstein metric [37], given in Equation 11. This metric measures the optimal cost of transforming one probability distribution into another within a specified metric space [56]. Here, the quantile functions $F_X^{-1}(\tau)$ and $F_Y^{-1}(\tau)$ are formulated as $F_Z^{-1}(\tau) = \inf\{z \in \mathbb{R} : \tau \leq F_Z(z)\}$, where $F_Z(z)$ represents the cumulative distribution function and τ the quantile fraction.

$$W_p(F_X, F_Y) = \left(\int_0^1 \left\|F_X^{-1}(\tau) - F_Y^{-1}(\tau)\right\|_p d\tau\right)^{1/p} \tag{11}$$

Various techniques for parameterizing return distributions have been proposed, such as categorical distributions [37] and quantile regression [38]. The present study employs Implicit Quantile Networks (IQN), selected for their capacity to flexibly model complex, multi-modal return distributions while maintaining parametric and computational efficiency [39]. IQN implicitly approximate the continuous quantile function $F_Z^{-1}(\tau)$ by passing quantile fractions, stochastically sampled from a uniform distribution ($\tau \sim U([0,1])$), through a DNN to derive corresponding quantile values. The comparison between the traditional return expectation estimation and the IQN's approximation of $F_Z^{-1}(\tau)$ for randomly sampled τ is depicted in Figure 1.



Traditional Returns Expectation Estimation IQN's Returns Quantile Function Approximation

Fig. 1 Comparison of traditional (left) and distributional (right) approaches in representing the returns for a particular state s and selected action a at discrete time-step t, inspired by Dabney et al. [39].

D. Distributional Soft Actor-Critic (DSAC)

DSAC extends SAC through the incorporation of an IQN critic $Z_\tau(s, a; \theta)$ to approximate the return distribution function [57, 58]. This fusion of maximum entropy learning with distributional critics has been demonstrated to be effective in learning high-dimensional continuous control tasks [57]. Modeling the returns distribution also facilitates the generation of risk-sensitive policies, which were shown to enhance the safety of RL-based flight control in a quadcopter [40] and a business jet [35].

In accordance with the $\mathcal{T}_D^\pi$ operator (Equation 10), the TD-error δ_{ij}^k for a given pair of quantile fractions $\hat{\tau}_i$ and $\hat{\tau}_j$ is specified by Equation 12. Here, $k \in \{1, 2\}$ represents the index of each Z_τ-network within the double-critic framework, with $\hat{\tau}_{i,j} = (\tau_{i,j} + \tau_{i+1,j+1})/2$, $\bar{\theta}_k$ and $\bar{w}$ denoting parameter vectors for the target quantile and policy networks, respectively.

$$\delta_{ij}^k = r + \gamma \left[\min_{k=1,2} Z_{\hat{\tau}_i}(s', a'; \bar{\theta}_k) - \alpha \log \pi(a'|s'; \bar{w}) \right] - Z_{\hat{\tau}_j}(s, a; \theta_k) \quad with \quad a' \sim \pi_w(\cdot|s'),\ \tau_i, \tau_j \sim U([0, 1]) \tag{12}$$

Parameter vector θ is optimized via quantile regression, employing the weighted pairwise Quantile Huber Loss as given by Equation 13 [59], where $\mathbb{1}$ represents the indicator function. This loss function transitions from a quadratic to a linear form at a specified threshold κ, conferring robustness to outliers compared to conventional mean squared error.

$$\rho_\tau^\kappa(\delta) = \left| \tau - \mathbb{1}_{\{\delta<0\}} \right| \mathcal{L}_\kappa(\delta) \quad with \quad \mathcal{L}_\kappa(\delta) \begin{cases} \frac{1}{2}\delta^2 & if |\delta| \le \kappa \\ \kappa(|\delta| - \frac{1}{2}\kappa) & otherwise \end{cases} \quad and \quad \mathbb{1}_{\{\delta<0\}} \begin{cases} 1 & if \delta < 0 \\ 0 & otherwise \end{cases} \tag{13}$$

Consequently, for a set of N independent quantile fractions, the critic loss function is formalized as in Equation 14.

$$\mathcal{L}_Z^\mathcal{B}(\theta) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (\tau_{i+1} - \tau_i) \rho_{\hat{\tau}_j}^\kappa(\delta_{ij}^k) \tag{14}$$

The policy π_w in DSAC adheres to the parametrization structure inherent to SAC. However, DSAC's critics output a distribution of returns, hence a transformation function $\Psi : \mathcal{Z} \to \mathbb{R}$ is employed to convert this distribution into a scalar action-value function Q. While this *risk measure function* Ψ can be implemented as a risk-neutral expectation $\mathbb{E}[\cdot]$, it may also be realized through alternative functions such as a Wang transform [60], enabling the modulation of risk sensitivity by skewing the critic's estimated returns distribution towards either more *risk-averse* or *risk-seeking* behaviors [39]. The resultant distorted action-value Q^r is incorporated into the distributional policy loss function, as specified by Equation 15.

$$\mathcal{L}_\pi(w) = \mathbb{E}_\mathcal{B} \left[\alpha \log \pi_w(a_w(s)|s) - Q_\theta^r(s, a_w(s)) \right] \quad with \quad Q_\theta^r(s, a_w(s)) = \Psi \left[\min_{i=1,2} Z_{\theta_i}(s, a_w(s)) \right] \tag{15}$$

E. Reinforcement Learning for Flight Control

The fixed-wing aircraft flight control poses a highly complex problem with non-linear and highly-coupled transition dynamics with 6 degrees of freedom (DOF). In general terms, the flight control problem can be defined by Equation 16.

$$\dot{x} = f(x, u, t) + w \approx f(x, u) + w \tag{16}$$

Here, f symbolizes the non-linear transition dynamics governed by the aircraft's equations of motion, with the state vector $x \in \mathbb{R}^n$ and control input vector $u \in \mathbb{R}^m$ as its arguments. To concentrate on the system's inherent dynamics,

quasi-stationarity is assumed, thus excluding explicit time dependence from f. The additive term $w \in \mathbb{R}^n$ accounts for stochastic disturbances, modeling the system's inherent noise and aerodynamic perturbations. For analytical tractability, the dynamics is often decoupled into longitudinal and lateral components, each with its own subset of DOF.

To formulate a *flight trajectory tracking* control task as an MDP, the dynamic state vector x must be augmented with tracking errors with respect to the desired flight trajectories τ_{ref}. Furthermore, some flight states may be unobservable, uncertain, or distorted by measurement noise, leading to a Partially Observable MDP (POMDP). Hence, it is imperative to distinguish between the RL state vector s, which reflects the observations of the agent, and the true dynamic state vector x. Actions a may either directly correspond to control inputs u or represent incremental control commands to refine input smoothness, with the actual control inputs u incorporated into the state vector [34]. The reward function is usually designed to penalize deviations from the reference trajectory τ_{ref} proportionally to the absolute $R(s,a) \propto |\tau_{\text{ref}} - x|$ or the squared tracking error $R(s,a) \propto (\tau_{\text{ref}} - x)^2$.

The application of RL to partially observable flight control systems introduces significant challenges. The Markov property assumption, which states that the future state transition $\mathcal{P}(s'|a,s)$ is fully predictable from the observation vector s, is invalidated by incomplete state observability, which impedes the formulation of predictive and reliable control policies. Additionally, the high dimensionality of the state-action space demands highly sample-efficient algorithms capable of generalizing across diverse flight conditions. Flight control's safety-critical nature further constrains the use of exploratory methods, as suboptimal actions during online in-flight learning or when encountering unobserved states after training could lead to catastrophic outcomes. All these factors culminate in a pronounced *simulation-to-reality gap*, where RL agents trained in simulations may falter in real-world flight scenarios, emphasizing the need for safety-prioritizing agents that are both sample efficient and robust to environmental uncertainties. The proposed RUN-DSAC algorithm addresses these challenges by integrating uncertainty quantification into the decision-making framework of the RL agent, enhancing sample efficiency and safety.

III. Methodology

This section details the RUN-DSAC framework and the implementation of the aircraft control task addressed.

A. Returns Uncertainty-Navigated Distributional Soft Actor-Critic (RUN-DSAC)

RUN-DSAC extends DSAC by embedding uncertainty quantification into policy decisions, enhancing control strategies in complex and uncertain environments.

1. The Principle of RUN-DSAC

Building upon DSAC, the RUN-DSAC algorithm presents an innovative approach to adaptive control in uncertain environments. While DSAC distorts the value distribution to encode risk preferences implicitly, RUN-DSAC refines this method by explicitly adjusting policy decisions based on quantified uncertainty. This approach draws inspiration from Liu et al. [40], who estimated right truncated variance (RTV) using DSAC's quantile functions to modulate the distortion function for risk-adaptive flight control of a quadcopter. In contrast, RUN-DSAC computes variance from the full distribution of returns for a comprehensive understanding of returns variability, while it explicitly channels this information into the actor's policy derivation.

Research in *risk-sensitive* RL recognizes variance $\mathbb{V}[\cdot]$ as an intuitive and comprehensive metric for uncertainty, fluctuation, and decision robustness [61–63]. Additionally, Bellemare et al. [37] show that the distributional Bellman operator $\mathcal{T}_D^\pi$ contracts in the second moment of the discounted returns, asserting that convergence in value distribution space concomitantly yields accurate variance estimations. To incorporate variance into the learning objective, RUN-DSAC shifts the focus from solely optimizing the expected return, $\Psi(Z) = \mathbb{E}[Z]$, to balancing the mean against the standard deviation, $\Psi(Z) = \mathbb{E}[Z] + \mu\sqrt{\mathbb{V}[Z]}$. The Q-function is thus modified as in Equation 17, where Q_θ represents the expected return and σ_Q is the standard deviation, scaled by the *uncertainty modulation factor* μ, which balances the mean and standard deviation in the objective. This modified value then guides the policy update.

$$Q_\theta^V(s,a) = Q_\theta(s,a) + \mu\sigma_Q \tag{17}$$

The standard deviation σ_Q, representing the variability in returns, is approximated with Equation 18.

$$\sigma_Q = \sqrt{\sum_{i=0}^{N-1} (\tau_{i+1} - \tau_i) \left[Z_{\hat{\tau}_i,\theta}(s,a) - Q(s,a) \right]^2} \tag{18}$$

The coefficient μ determines the agent's risk preference. A positive μ inclines the policy towards actions with high variance in returns, promoting risk-affine exploration, which will be referred to as *Risky*. Conversely, a negative μ induces risk-averse tendencies by favoring lower-variance actions, thereby enhancing the predictability and safety of the policy, and will be referred to as *Conservative*. As the agent's familiarity with the environment evolves, the initial high variance due to exploratory actions naturally diminishes. To reflect this maturation, μ is programmed to linearly decay to zero across N_{RUN} learning episodes.

The architecture of the RUN-DSAC algorithm is depicted in Figure 2, with the corresponding pseudocode detailed in subsection V.B.

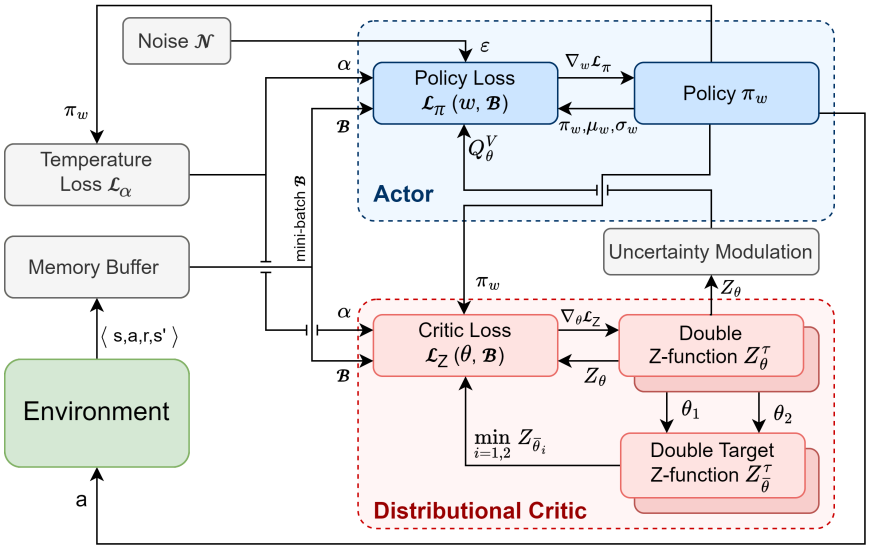


Fig. 2 Returns Uncertainty-Navigated Distributional Soft Actor-Critic (RUN-DSAC) architecture.

2. Actor and Critic Design

The actors used in this work are modeled as DNNs with a sequence of hidden layers that approximate a stochastic multivariate tanh-Gaussian policy with state inputs s, as depicted in Figure 3. The final hidden layer's output splits into two streams, one estimating the mean μ and the other computing the logarithm of the standard deviation $\log \sigma$, which is then exponentiated to derive σ. The actions a are produced by sampling from $\mathcal{N}(\mu, \sigma)$ and applying a tanh function to ensure boundedness within the action space. However, during deterministic policy evaluation, such as in post-training assessments, the network defaults to $\tanh(\mu)$ for consistent, non-stochastic output. Finally, the network applies reparametrization to facilitate gradient-based optimization during back-propagation.

The DSAC critics employ IQNs to estimate the return distribution Z_θ, as depicted in Figure 4. Here, states s and actions a are embedded in the Ψ layer. Concurrently, N randomly sampled quantile fractions $\tau \sim U([0, 1])$ undergo cosine embedding Φ, as given by Equation 19, with parameters w_{ij}, b_j, and *Sigmoid* activation σ. These embeddings are elementwise-multiplied through Hadamard product [39], followed by layer normalization to ensure consistent output scaling and numerical stability [64], before progressing through hidden layers H to the output layer F. Parameters are optimized iteratively with the *Adam* stochastic gradient descent (SGD) [65] using the quantile Huber loss (Equation 13).

$$\phi_j(\tau) = \sigma\left(\sum_{i=0}^{N-1} \cos(\pi i \tau) w_{ij} + b_j\right) \tag{19}$$

3. Policy Regularization

Deep RL controllers often learn policies with insufficient action smoothness, leading to high-frequency control signal oscillations. In aviation control systems, such oscillatory actions compromise tracking performance while

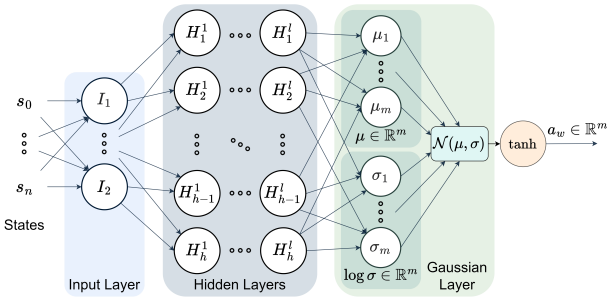


Fig. 3 Multivariate tanh**-Gaussian policy.**

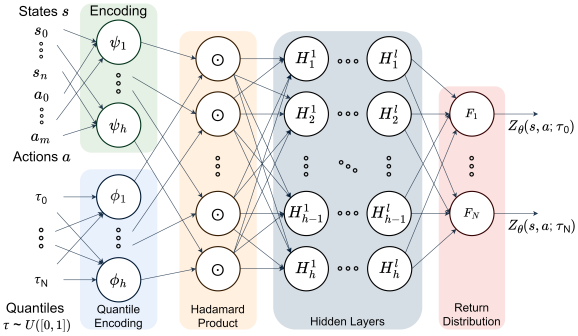


Fig. 4 Implicit quantile network (IQN) architecture, inspired by Dabney et al. [39].

precipitating overheating, increased power consumption, and actuator failure. This issue can be addressed using incremental control strategies, where the agent's actions are cumulative adjustments [34]. Such an approach, however, escalates dimensionality, negatively impacting sample efficiency. Instead, the present paper employs the Conditioning for Action Policy Smoothness (CAPS) regularization technique to achieve smoother control policies without increasing problem complexity [66].

The CAPS methodology uses regularization to minimize the policy function's Lipschitz constants both temporally and spatially, as reflected in the temporal smoothness term $\mathcal{L}_T$ (Equation 20) and the spatial smoothness term $\mathcal{L}_S$ (Equation 21). Here, $||.||_2$ represents the Euclidean norm, with $\mathcal{L}_T$ enforcing action consistency across successive time steps and $\mathcal{L}_S$ term promoting action equivalence for analogous states.

$$\mathcal{L}_T = ||\pi(s_t) - \pi(s_{t+1})||_2 \qquad (20) \qquad \mathcal{L}_S = \|\pi(s) - \pi(\bar{s})\|_2 \quad with \quad \bar{s} \sim N(s, \sigma_{\text{CAPS}}) \qquad (21)$$

The hyperparameters λ_S and λ_T balance action smoothness and policy improvement, as given in Equation 22, leading to a smoother control policy when integrated with the loss function, as in Equation 23.

$$\mathcal{L}_\pi^{CAPS} = \lambda_T \mathcal{L}_T + \lambda_S \mathcal{L}_S \qquad (22) \qquad \mathcal{L}_\pi(w) = \mathbb{E}_\mathcal{B}\left[\alpha \log \pi_w(a_w(s)|s) - Q_\theta^V(s, a_w(s)) + \mathcal{L}_\pi^{CAPS}\right] \qquad (23)$$

B. Flight Attitude Control

This subsection delves into the attitude control of a Cessna Citation II, employing the DASMAT simulation model.

1. Simulation Environment

This study examines the attitude control of a Cessna Citation II using DASMAT, a validated high-fidelity aircraft simulation model featuring fully-coupled, non-linear dynamics [67]. DASMAT models 12 dynamic states $x \in \mathbb{R}^{12}$, as defined in Equation 24. These include three airspeed components (true airspeed V, angle of attack α, and angle of sideslip β), three angular velocities (roll p, pitch q, and yaw r rates), and three Euler angles for angular orientation (roll ϕ, pitch θ, and yaw ψ). Also, translational positions in the local tangent plane are quantified through horizontal coordinates X_e and Y_e, and altitude h.

$$x = [p, q, r, V, \alpha, \beta, \phi, \theta, \psi, h, X_e, Y_e]^T \in \mathbb{R}^{12} \qquad (24)$$

The agent's control input $u \in \mathbb{R}^3$, defined in Equation 25, exclusively controls aerodynamic surface deflections: elevator δ_e, aileron δ_a, and rudder δ_r. The corresponding actuator deflection limits are given by Equation 26 [68], which define the control policy's action space. Thrust is regulated by a separate inner loop for velocity regulation, thereby reducing the overall control problem complexity. These dynamic states and control inputs are illustrated in Figure 5.

$$u = [\delta_e, \delta_a, \delta_r]^T \in \mathbb{R}^3 \qquad (25) \qquad \mathcal{A} = \underbrace{[-17°, 15°]}_{\delta_e} \times \underbrace{[-19°, 15°]}_{\delta_a} \times \underbrace{[-22°, 22°]}_{\delta_r} \in \mathbb{R}^3 \qquad (26)$$

The simulation operates at a refresh rate of $f_s = 100$ Hz. It assumes ideal sensors and models actuators through low-pass filter dynamics, coupled with predetermined deflection saturation limits.

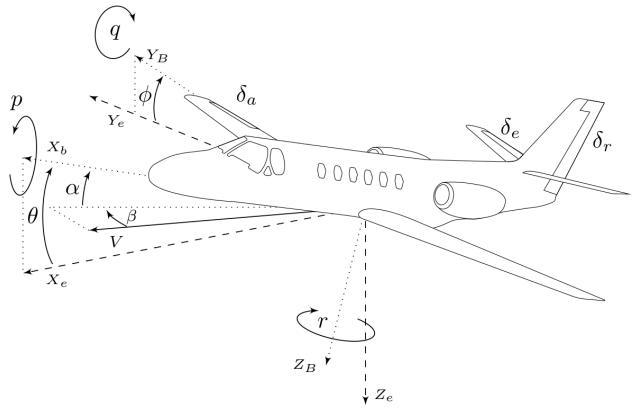


Fig. 5 Cessna Citation II's dynamic states and flight control surfaces, showing the body frame (X_b, Y_b, Z_b) at an attitude defined relative to the inertial frame (X_e, Y_e, Z_e). Yaw (ψ) is set to zero for clarity. (Adopted from Seres and van Kampen [35].)

2. Controller Architecture

The control diagram in Figure 6 shows the feedback loop between the non-linear RL agent and the controlled plant. The agent directly controls actuator deflections based on the observation vector s defined in Equation 27, which is a subset of the state vector x augmented by the tracking error e, as specified in Equation 28.

$$s = [\theta_e, \phi_e, \beta_e, p, q, r, \alpha]^T \in \mathbb{R}^7 \qquad (27) \qquad e = [\theta_e, \phi_e, \beta_e] = [\theta_{\text{ref}} - \theta, \phi_{\text{ref}} - \phi, \beta_{\text{ref}} - \beta] \in \mathbb{R}^3 \qquad (28)$$

The reward function r, detailed in Equation 29 and inspired by Dally and van Kampen [34], penalizes the $L1$ norm of attitude tracking error, where a scaling parameter c_r compensates for the lower magnitude of sideslip error by proportionally weighting its influence.

$$r(s,a) = -\frac{1}{3} \left\| \text{clip}\left[c_r \odot e, -1, 1\right] \right\|_1 \quad with \quad c_r = \frac{6}{\pi}[1,1,4]^T \in \mathbb{R}^3 \qquad (29)$$

Additionally, three soft constraints were implemented to enforce the flight envelope, specified as $h \geq 100$ m, $|\theta| \leq 60°$

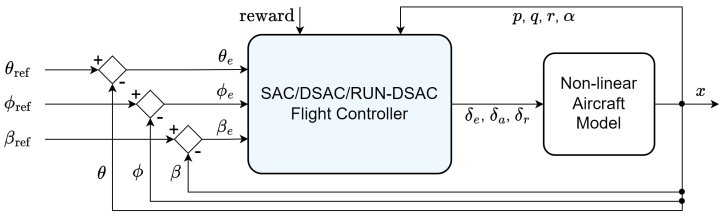


Fig. 6 RL-based flight attitude controller architecture.

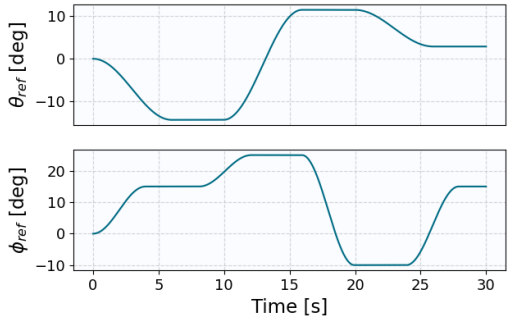


Fig. 7 Random cosine-smoothed step sequences, applied as training reference signals for pitch θ_{ref} and roll ϕ_{ref} during a single episode.

and $|\phi| \leq 75°$. Constraint violations trigger premature episode termination and incur a substantial sparse negative reward r_p, proportional to the remaining time frames as outlined in Equation 30. A positive constant c_p scales the penalty for early termination, with an empirically optimized value of $c_p = 2$. These constraints were observed to guide the agent towards desired behaviors, effectively improving the sample efficiency.

$$r_p = -c_p f_s (t_{\max} - t) \qquad (30)$$

C. Experiment Design

This paper compares the uncertainty-driven RUN-DSAC with the original DSAC in flight attitude control tasks. Additionally, these algorithms are benchmarked against SAC, a state-of-the-art RL algorithm that previously demonstrated fault-tolerant flight control capabilities, but exhibited inconsistencies in offline training convergence and heightened sensitivity to stochastic processes and hyperparameters [34]. Essentially, three approaches to uncertainty management are contrasted: SAC, which operates without uncertainty information; DSAC, which models uncertainty; and RUN-DSAC, which actively leverages modeled uncertainty for decision-making.

Table 1 Optimized hyperparameters for SAC, DSAC and RUN-DSAC agents.

Shared			**DSAC (Additional)**		
Hidden layers structure	$\bar{h}$	64x64	Num. of quantiles	N_q	32
Actor learning rate	-	$1{\cdot}10^{-3}$	Num. of cosine neurons	$\lvert\phi\rvert$	64
Critic learning rate	-	$4.4{\cdot}10^{-4}$	τ embedding activation	-	Sigmoid
Discount factor	γ	0.98	Huber regression threshold	κ	1.0
Entropy target	$\bar{\mathcal{H}}$	-3	**RUN-DSAC (Additional)**		
Batch size	$\lvert\mathcal{B}\rvert$	256	Initial uncertainty modulation factor	μ_{init}	1 or -1
Memory buffer size	$\lvert\mathcal{D}\rvert$	10^6	Uncertainty modulation decay horizon	N_μ	125
Non-linear activation	-	ReLU			
Polyak step size	ζ	0.995			
Policy regularization	$\mathcal{L}_T$, $\mathcal{L}_S$,	400			

Table 2 Evaluation scenarios assessing generalization capabilities and fault-tolerance.

Identifier	Scenario	Description
N0	Nominal	Trim condition consistent with training (h = 2000 m, V = 90 m/s).
G1	Wind Gust	15 ft/s vertical wind gust sustained for 3 seconds.
G2	Noisy Signal	Gaussian noise sensor models applied to the observations (Table 3).
G3	High Dynamic Pressure	Trim condition changed to h = 2000 m, V = 150 m/s.
G4	Low Dynamic Pressure	Trim condition changed to h = 10000 m, V = 90 m/s.
G5	Near-stall	The pitch angle reference θ_{ref} set to 40° at t = 35 s.
F1	Ice Accretion on Wings	Lift curve decreased by 30%, drag coefficient incremented by 0.06.
F2	Center of Gravity Shifted Aft	Center of gravity shifted aft by 0.25 m.
F3	Center of Gravity Shifted Forward	Center of gravity shifted forward by 0.25 m.
F4	Saturated Aileron	Aileron deflection constrained to ±1°.
F5	Saturated Elevator	Elevator deflection constrained to ±2.5°.
F6	Damaged Elevator	Elevator effectiveness coefficient decreased by 70%.
F7	Jammed Rudder	Rudder immobilized at a 15° deflection.

The agents are trained in 30-second episodes using randomly generated pitch θ_{ref} and roll ϕ_{ref} reference signals, starting from a trimmed flight at h = 2,000 m and V = 90 m/s. These reference signals are formulated as cosine-smoothed step functions, with their amplitudes uniformly sampled from 15 discrete levels within [-20°, 20°] for θ_{ref} and [-35°, 35°] for ϕ_{ref}. Figure 7 illustrates an instance of such a signal set, while the sideslip reference β_{ref} is set to zero. After each episode, *average returns* are evaluated by averaging the rewards accumulated over 10 independent trajectories $\mathcal{T}$ sampled under the current policy.

Table 3 Gaussian noise parameters for sensor models derived from the PH-LAB aircraft in-flight data [69].

Signal	Unit	Noise (σ^2)	Bias
p, q, r	rad/s	4.0×10^{-7}	3.0×10^{-5}
θ, ϕ	rad	1.0×10^{-9}	4.0×10^{-3}
α	rad	4.0×10^{-10}	-

The hyperparameters configured for the agents are listed in Table 1. The first column lists hyperparameters refined through rigorous tuning, with $\bar{\mathcal{H}}$, ζ, $\mathcal{L}_T$, and $\mathcal{L}_S$ values adopted from literature [34, 35, 70]. These proved optimal for all three agents. DSAC's additional parameters are inspired by Ma et al. [57], who also observed that employing *Sigmoid* over *ReLU* activation improves the smoothness of quantile fractions embedding. For RUN-DSAC, μ_{init} was selected such that uncertainty is effectively captured during learning, while N_μ was determined through iterative experimentation.

After training, the agents' *learning performance* is contrasted in terms of sample efficiency, return convergence, and consistency. Quantifying sample efficiency necessitates caution. Relying on a single metric, such as samples required to reach a certain return threshold, can lead to misleading interpretations, as different threshold choices may yield varied outcomes. To mitigate bias and ambiguity, a diverse set of metrics is employed, encompassing: average returns at the final episode (M1), average returns post-100 episodes (M2), sample count required to achieve average returns of -2000 on filtered learning curves (M3), area between learning curves and the x-axis (M4), and the sample count required for the gradients of filtered learning curves to remain within ±30 (M5).

Additionally, the robustness of each agent's policy is assessed across various unseen flight scenarios, as detailed in Table 2. First, agents' *tracking performance* is evaluated under the baseline scenario $N0$, which replicates training flight conditions to establish a performance benchmark. Subsequent scenarios, inspired by literature [34], were selected for their relevance to flight attitude control, complexity, and compatibility with the DASMAT framework. *Generalization capability* scenarios (G) evaluate the agents' adaptability to untrained flight regimes, such as variations in dynamic pressure. From these, $G1$ simulates a wind gust with velocity derived from MIL-F-8785C standards [71], while $G2$ evaluates robustness to sensor noise simulated by realistic sensor models as outlined in Table 3 [69]. *Fault tolerance* scenarios (F) probe fault-robustness by introducing variations in the controlled plant to simulate in-flight failures.

Each agent is evaluated on identical tasks, using extended episodes of $t_{\max}$ = 80 s with predefined cosine-smoothed step sequences in pitch and roll reference maneuvers, mirroring training signals. For a fair comparison of tracking proficiency across all controllers, the normalized mean absolute error (nMAE) metric is employed. This involves normalizing θ_e and ϕ_e tracking errors against the peak amplitudes of the evaluation reference signals, while β_e is normalized to the range of [-5°, 5°].

IV. Results and Discussion

This section presents findings on learning efficiency, fault tolerance, and robustness under various flight conditions, evaluated for SAC and three variants of distributional agents: DSAC, *Conservative* RUN-DSAC, and *Risky* RUN-DSAC. Unless stated otherwise, the mean and standard deviation of the results are calculated from ten independent evaluations. Line graphs display the average across policies as a bold line, with shaded areas representing standard deviations. In general, the SAC and RUN-DSAC agents are benchmarked against the baseline DSAC agent, with statistical significance in performance differences assessed using a t-test under the assumption of normally distributed samples.

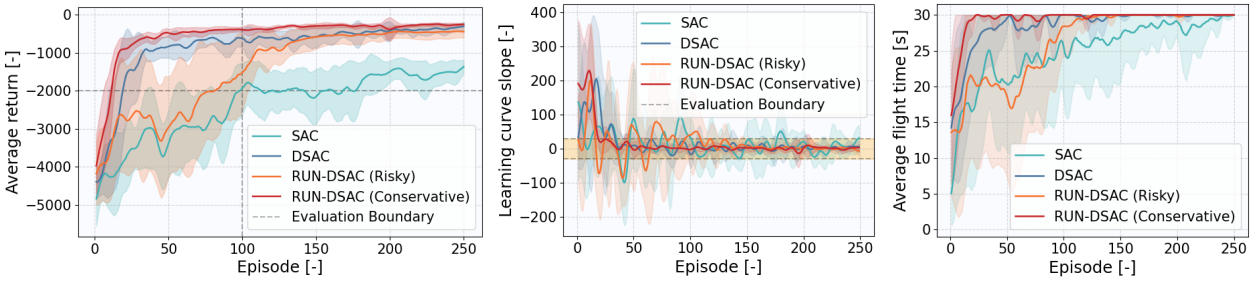
A. Learning Performance

The learning curves for each policy are illustrated in Figure 8a. Distributional agents consistently converge to higher return levels in line with those documented in prior research [34–36]. Conversely, SAC underperforms, yielding returns over threefold lower than DSAC and demonstrating greater variance. These deficiencies are attributable to SAC's lower convergence success rate and pronounced sensitivity to stochastic processes like initialization, as noted by Dally and van Kampen [34]. Nevertheless, contrary to their selective reporting of outcomes from only converged SAC policies, this paper presents unfiltered results.

The learning curve of *Conservative* RUN-DSAC demonstrates the effectiveness of integrating Q-value uncertainty into decision-making for enhanced sample efficiency as evidenced by its rapid ascent. Furthermore, prioritizing

predictable policies results in reduced variance in learning curves and the highest average converged returns, as indicated by *M1* in Table 4. This suggests that *Conservative* RUN-DSAC learned to track the training attitude reference signals more accurately and consistently compared to other agents. Comparative analysis in Table 4 also highlights *Conservative* RUN-DSAC outperforming DSAC across all five learning performance metrics, with four displaying statistically significant differences. The largest improvement of 54.5% is observed in the *M5* metric, which is visualized in Figure 8b. While the *Conservative* RUN-DSAC's learning curve slope is the fastest to converge to the metric boundary, it also shows the least oscillations, which suggest a consistent and predictable learning performance that is highly desirable in the safety-focused flight control domain.

Conversely, *Risky* RUN-DSAC's slow convergence and initially high variance suggest that promoting uncertainty impedes learning performance. This variance arises from the algorithm's exploratory nature and the reward function design, which terminates the training flight prematurely and imposes high penalties when the flight envelope is violated (Equation 30). Figure 8c compares the mean flight duration per episode, revealing that *Risky* RUN-DSAC initially experiences more frequent and earlier violations, but achieves "crash-free" flight after approximately 125 episodes (coinciding with the decay of μ to 0). In contrast, *Conservative* RUN-DSAC achieves this stability in about 25 episodes, emphasizing its safety advantage. SAC trails significantly, requiring the entirety of 250 episodes to achieve crash-free flight. Finally, although this study focuses on offline training, the characteristics of *Conservative* RUN-DSAC render it potentially suitable for online flight control, particularly for in-flight fault management, where sample-efficient, predictable and safe learning is paramount.



(a) Learning curves. **(b) Learning curve slopes.** **(c) Episodic flight time.**

Fig. 8 Comparison of learning performance, with curves smoothed using a Gaussian kernel with a window length of 10. Dashed lines and yellow shading indicate performance metric thresholds.

Table 4 Assessment of agent learning efficiency across five metrics includes mean scores with standard deviations, percent difference from DSAC, and p-values for significance (bold for $p < 0.05$). Green indicates an improvement, red a decline.

		M1 (Final)	M2 (Returns)	M3 (# Episodes)	M4 (Area)	M5 (Gradient)
DSAC	Value	-308.1	-626.4 ± 254.4	16.8 ± 6.0	(20.8 ± 5.3)·10^4	183.2 ± 56.7
SAC	Value	**-1369.5±187.9**	**-1926.0 ± 676.4**	**69.1 ± 47.1**	**(59.0 ± 7.3)·10^4**	**230.9 ± 27.0**
	Difference	**-344.5%**	**-207.5%**	**+311.3%**	**+183.7%**	**+26.0%**
	p-value	**1·10^{-10}**	**2·10^{-4}**	**9·10^{-3}**	**6·10^{-10}**	**4·10^{-2}**
RUN-DSAC (Risky)	Value	-447.7±167.0	**-1528.2 ± 998.8**	37.8 ± 32.8	**(36.0 ± 12.4)·10^4**	162.1 ± 48.0
	Difference	-45.3%	**-144.0%**	+125.0%	**+73.1%**	-11.5%
	p-value	5·10^{-2}	**2·10^{-2}**	7·10^{-2}	**3·10^{-3}**	4·10^{-1}
RUN-DSAC (Conserv.)	Value	-253.1±28.0	**-398.5 ± 88.6**	**9.1 ± 4.0**	**(12.9 ± 1.6)·10^4**	**83.3 ± 55.1**
	Difference	+17.9%	**+36.4%**	**-45.8%**	**-37.9%**	**-54.5%**
	p-value	2·10^{-1}	**3·10^{-2}**	**5·10^{-3}**	**4·10^{-4}**	**1·10^{-3}**

B. Tracking Performance

Learning curves provide an initial assessment of training effectiveness but do not fully represent control capabilities. Hence, the trained policies are evaluated against reference signals that mirror training conditions (Figures 9-12). The observed attitude tracking performance of the agents aligns with the learning curves presented in Figure 8a. SAC shows limited tracking accuracy with high variance in policies, reflected in an nMAE of 23.53±4.40%. This, along with large oscillations in its response, renders it aerodynamically and structurally unsuitable for flight control. In comparison, DSAC substantially outperforms SAC with smoother actions and an nMAE nearly four times smaller at 5.95±1.15%, marking a significant improvement ($p = 2 \cdot 10^{-6} < 0.05$).

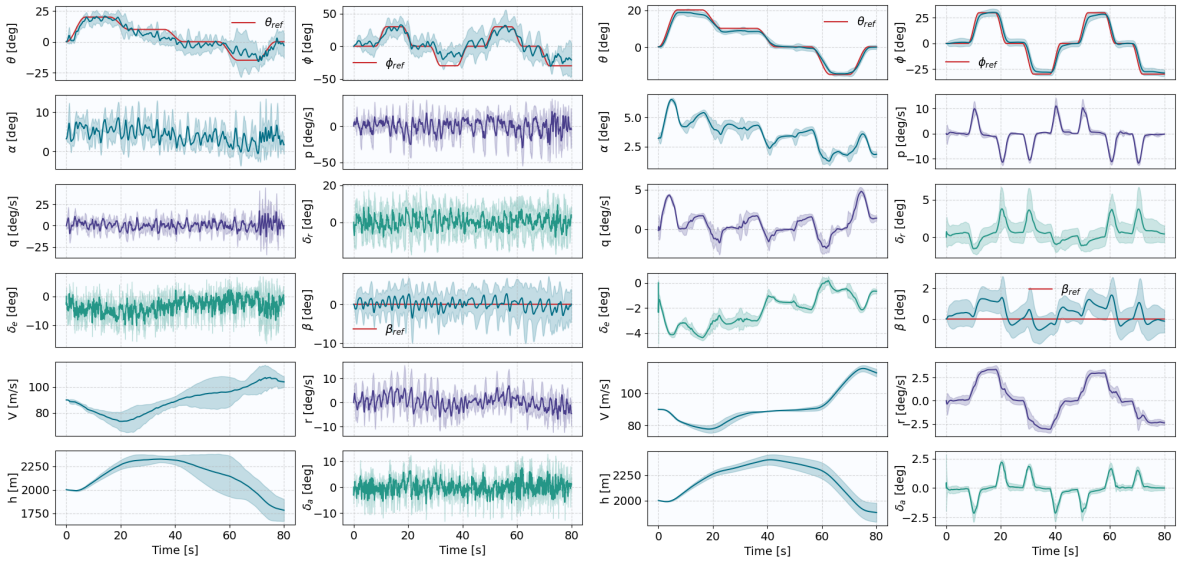


Fig. 9 State time-traces of the SAC policy in the nominal scenario.

Fig. 10 State time-traces of the DSAC policy in the nominal scenario.

Most notably, *Conservative* RUN-DSAC achieves the highest signal tracking accuracy and the least variability, evidenced by a reduction in nMAE to 3.95±0.42%, surpassing DSAC by 33% ($p = 4 \cdot 10^{-4} < 0.05$). This improved tracking accuracy and response predictability affirm the benefits of integrating uncertainty information into decision-making to enhance the efficiency and safety of intelligent flight controllers. Conversely, *Risky* RUN-DSAC shows higher tracking error and state variability, with a 57% higher nMAE of 9.36±2.48% compared to DSAC ($p = 3 \cdot 10^{-3} < 0.05$). These distinct results of the RUN-DSAC variants validate theoretical expectations: the *Conservative* RUN-DSAC's focus on predictability yields lower policy variability, while the exploration-driven *Risky* variant shows increased variability.

The distributional agents' state time-traces also reveal dynamic couplings between the principal axes, particularly where ϕ maneuvers influence θ and β tracking. The agents have adapted to exploit this coupling for enhanced control, reflected in the coordinated control surface deflections. However, this learned coupling can lead to undesired outcomes when actuator inputs are unintentionally correlated, which is especially notable in the *Conservative* RUN-DSAC agent. Here, the agent's initial sharp elevator deflection (for trim adjustment) inadvertently triggers corresponding aileron and rudder deflections, despite zero roll and yaw references. This causes transient roll and yaw rates that the agent swiftly neutralizes, yet a residual 2° roll offset and approximately 3° under-correction in the 30° ϕ_{ref} steps persist. In contrast, DSAC and *Risky* RUN-DSAC, which are less impacted by unintentional coupling, reach full roll reference but with compromised pitch precision.

The under-correction observed in *Conservative* RUN-DSAC is attributed to its inclination towards familiar, albeit more conservative, actions, thus favoring strategies with proven past efficacy even when they are suboptimal for the current situation. Additionally, this agent might implicitly prioritize precise pitch control over roll due to the greater risks associated with pitch deviations, such as their impact on aerodynamic forces and stall potential. This explains its superior performance in pitch tracking compared to other distributional agents at the cost of under-correcting roll.

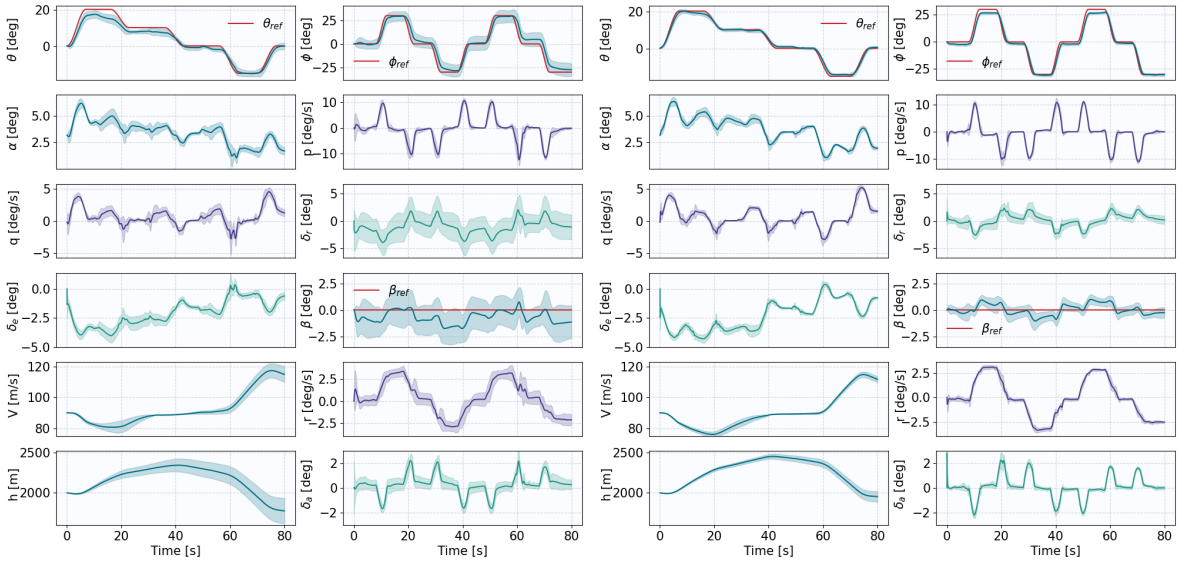


Fig. 11 State time-traces of the *Risky* RUN-DSAC policy in the nominal scenario.

Fig. 12 State time-traces of the *Conservative* RUN-DSAC policy in the nominal scenario.

C. Generalization Capability

The preceding section highlights the *Conservative* RUN-DSAC's reliance on the confidence in pre-learned strategies. However, this potentially impairs its adaptability to novel scenarios, which motivates an evaluation of the agents' robustness in untrained flight conditions. While prior research demonstrated enhanced safety in a near-stall scenario through distorting DSAC returns [35], this study investigates the impact of embedding return uncertainty into policy training on stall robustness, and expands the scope with four additional case studies simulating unforeseen common flight conditions.

Figure 13 presents the agents' generalization performance, with a separate ordinate for *G5* due to its higher nMAE scores and a distinct evaluation reference signal. In all scenarios, *Conservative* RUN-DSAC achieves the highest attitude tracking accuracy and the lowest result variance among agents, with statistical significance ($p < 0.05$). Conversely, the *Risky* variant underperforms compared to DSAC in all scenarios and exhibits higher variance (except in *G3*), with statistically significant differences in *G1* and *G2*. SAC, consistent with its learning curve shown in Figure 8a, demonstrates significantly the lowest performance and highest variance across policies.

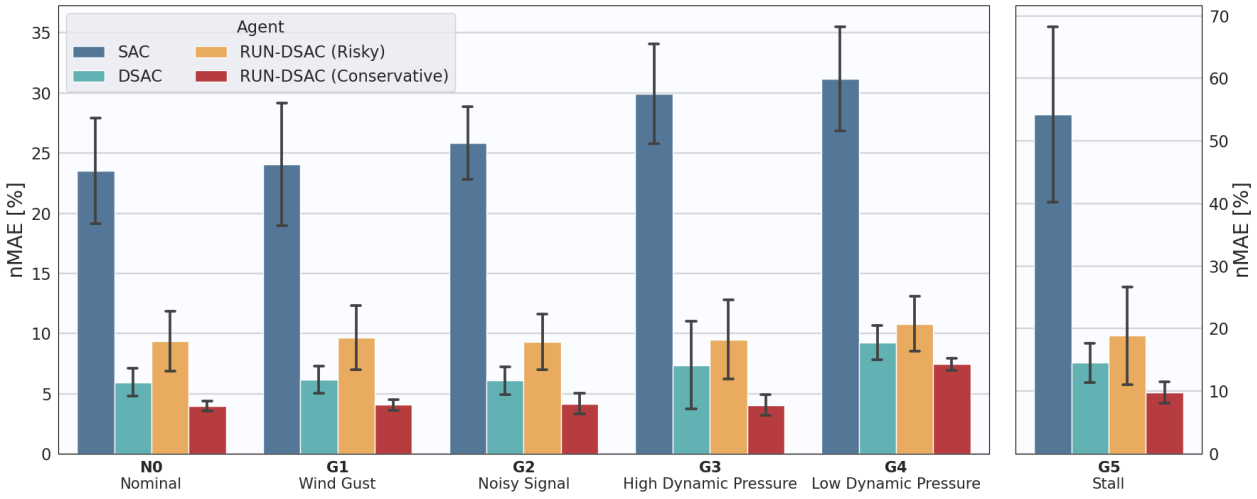


Fig. 13 Attitude tracking nMAE and its standard deviation across various generalization scenarios.

The *Stall* (*G5*) scenario is selected to showcase the generalization capability of distributional agents in flight control. This scenario is particularly relevant due to the substantial uncertainty inherent in near-stall conditions, which stems from the agents' lack of exposure to dynamics altered by unobserved airspeed and altitude states, or abrupt loss of lift. This uncertainty is directly correlated with heightened flight safety risks, potentially leading to loss-of-control (LOC) incidents. The *G5* task entails a sustained high pitch-up maneuver to induce near-stall conditions, with zero roll and sideslip references. Figures 14 and 15 respectively illustrate the longitudinal response of distributional agents and analyze the average rewards and return distributions these agents experience during flight.

High pitch angles significantly affect aircraft aerodynamics, leading to decreased airspeed and increased altitude. At $t = 40\ s$, the aircraft is commanded to reach a 40° θ_{ref}, extending beyond the training conditions. As Figure 15 shows, this unfamiliar state is marked by a decline in expected returns and heightened variance, indicating increased uncertainty in agents' performance. Notably, the *Conservative* RUN-DSAC agent, with the highest expected returns and lowest variance, demonstrates the greatest confidence and tracking performance (reflected in the highest rewards) in this *Stall* scenario. Contrarily, although the *Risky* variant exhibits the lowest confidence based on its expected returns and variance, it outperforms DSAC in managing the high pitch angle reference at $t = 40\ s$, as evidenced by higher rewards. This suggests that while *Risky* RUN-DSAC's exploratory strategy is less effective in nominal conditions, it can be advantageous in unexpected scenarios.

While attempting to attain the high pitch angle reference, the agents encounter aerodynamic instabilities and stall-induced oscillations. Interestingly, the distributional agents display varying oscillation intensities in their responses: DSAC shows the most significant oscillations, followed by the *Conservative* RUN-DSAC, while *Risky* RUN-DSAC demonstrates the least oscillatory behavior. This pattern remains consistent across other near-stall flight scenarios, including those affected by wind gusts (*G1*) or ice-accretion faults leading to the reduction of maximal lift (*F1*).

The oscillation differences among agents can be explained by correlating each agent's actor DNN to a *gain parameter* K in classical control theory. K influences a system's reaction to error signals: a high gain K triggers aggressive responses, potentially causing instability or oscillations, whereas a low gain K yields conservative reactions. *Risky* RUN-DSAC seems to adopt a policy with a higher K, enhancing responsiveness to counteract the diminished aerodynamic efficiency and damping under decreased dynamic pressure in near-stall scenarios (caused by lower velocity and increased altitude). In contrast, DSAC, with a lower K, demonstrates less responsive behavior with larger, possibly delayed corrections. This results in inadequate error compensation, leading to more pronounced oscillations. *Conservative* RUN-DSAC exhibits intermediate characteristics.

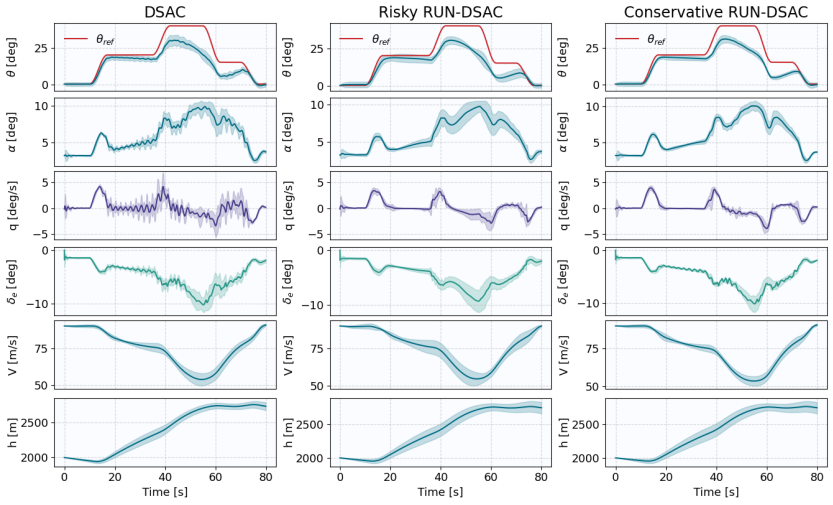


Fig. 14 Longitudinal state time traces of DSAC (left), *Risky* RUN-DSAC (middle), and *Conservative* RUN-DSAC (right) policies in the stall scenario.

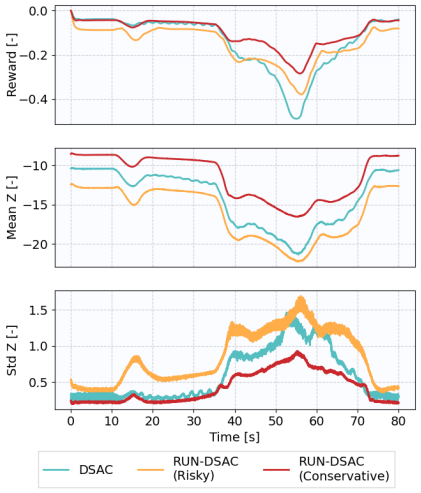


Fig. 15 Average rewards, state means and variances collected by distributional agents during flight in the stall scenario.

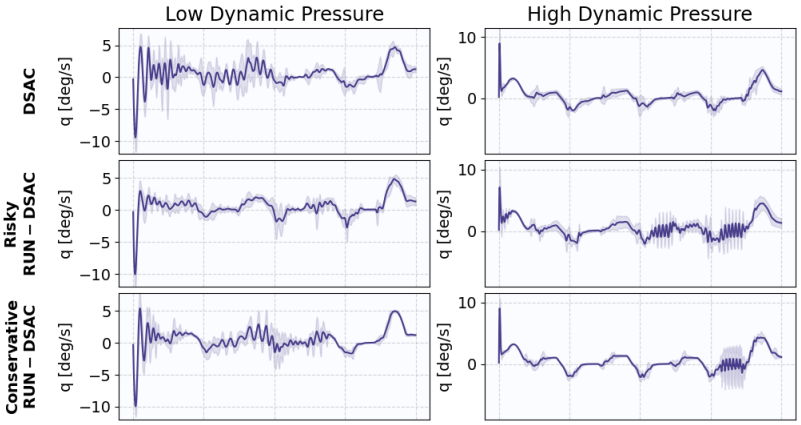


Fig. 16 Pitch rate time traces for DSAC (top row), *Risky* RUN-DSAC (middle row), and *Conservative* RUN-DSAC (bottom row) for the low dynamic pressure (left column) and high dynamic pressure (right column) scenarios.

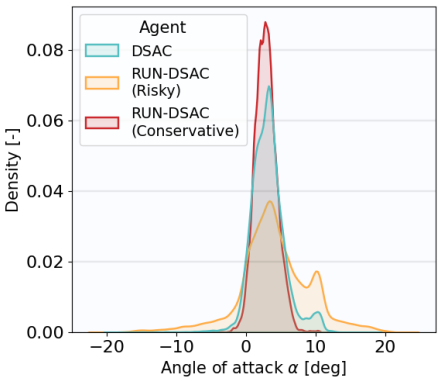


Fig. 17 Kernel density estimation of α experienced by various distributional agents during a single training run.

To validate the proposed *gain analogy*, a similar response pattern is anticipated in the low dynamic pressure scenario (*G4*), whereas the opposite is expected under conditions of increased dynamic pressure (*G3*) due to enhanced aerodynamic effectiveness. In the latter case, the aggressive, high-K control of the *Risky* variant could exacerbate oscillations, while the low-K control of DSAC and *Conservative* RUN-DSAC is likely to stabilize the system more effectively. This hypothesis is supported by Figure 16, which contrasts the pitch rate responses of the distributional agents under both low and high dynamic pressure conditions.

While the *gain analogy* partially explains the heightened oscillations of the *Conservative* RUN-DSAC relative to the *Risky* variant for low dynamic pressure scenarios, it falls short in explaining similar response disparities in scenarios with nominal dynamic pressure but altered aircraft dynamics (e.g., *F5* or *F7*). Hence, an alternative hypothesis is that the *Conservative* RUN-DSAC's focus on predictability may predispose it to overfit to training scenarios, leading to suboptimal and instability-prone policies under altered conditions. In contrast, the *Risky* RUN-DSAC's emphasis on uncertainty promotes exposure to a broader range of states and actions, bolstering robustness to novel scenarios and reducing susceptibility to instability, albeit with trade-offs in tracking precision and increased policy variance.

Supporting this hypothesis, a Gaussian kernel-smoothed density estimation of α distributions experienced by each agent during training is illustrated in Figure 17. This analysis reveals that the *Risky* RUN-DSAC has encountered a wider range of α, including extremes, which better prepares it for scenarios like $G5$. In contrast, the *Conservative* RUN-DSAC's experience is more restricted, with less frequent encounters of extremes than DSAC, underpinning its propensity to overfitting and increased vulnerability to instabilities in unforeseen scenarios.

D. Fault Tolerance

Assessing agents in diverse in-flight fault scenarios is imperative for ensuring safety, as modeling every conceivable failure for offline training is impractical. Figure 18 illustrates the performance of these agents in such evaluations. The *Conservative* RUN-DSAC consistently outperforms other agents in all seven fault scenarios, with statistically significant improvements over DSAC ($p < 0.05$) in all but *F7*. Conversely, the *Risky* variant underperforms relative to DSAC in every scenario, with statistically significant disparities in *F1*, *F3*, *F4*, and *F6*. SAC exhibits the lowest performance and the highest policy variance.

Despite its superior tracking performance and reduced policy variance, *Conservative* RUN-DSAC was observed to exhibit heightened sensitivity to changes in aircraft dynamics caused by failures, particularly those affecting control surface functionality, compared to the other distributional agents. This sensitivity results in marked disturbances in attitude states, contrasted by minimal oscillations in *Risky* RUN-DSAC, with DSAC exhibiting intermediate behavior. This reinforces the hypothesis of *Conservative* RUN-DSAC's overfitting tendency: policies refined during training become compromised under substantially altered dynamics.

The disparity in disturbance patterns is particularly evident in the *F7* scenario, which is characterized by high nMAE scores due to persistent sideslip bias. Figures 19 and 20 compare the state trajectories of the *Risky* and

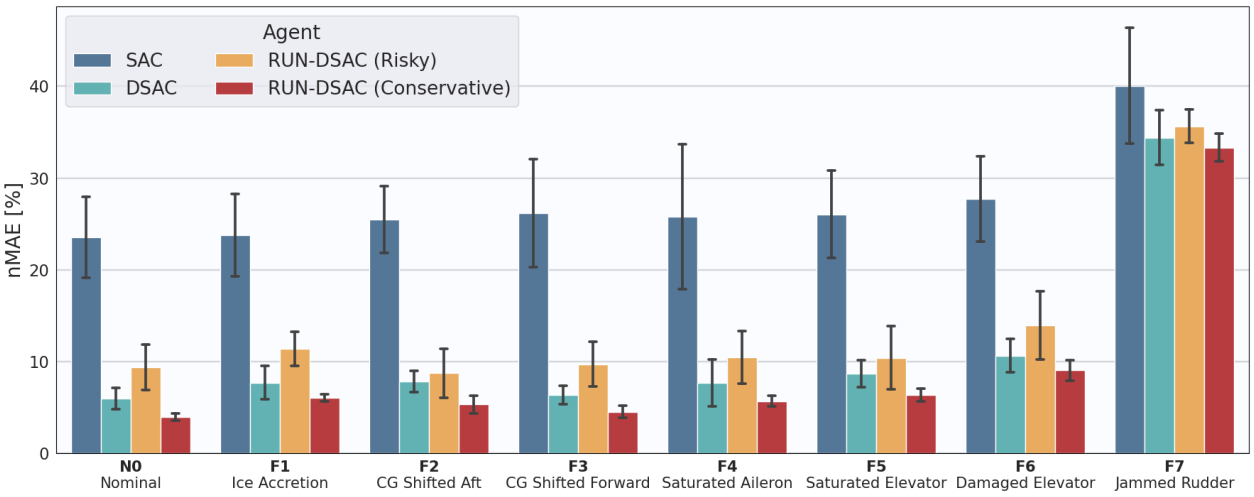


Fig. 18 Attitude tracking nMAE and its standard deviation across various in-flight fault scenarios.

Conservative RUN-DSAC variants, respectively. Here, both agents unsuccessfully attempt to counteract the stuck rudder by commanding deflections in the opposite direction. However, the rudder control signal triggers aileron deflections due to the learned roll-yaw coupling, counterbalancing the negative roll induced by the dihedral effect. This stabilizes β at $\sim 10^\circ$ and arrests further roll, but introduces a negative roll angle offset, subsequently impacting pitch tracking through pitch-roll coupling. These unusual flight dynamics cause pronounced oscillations in *Conservative* RUN-DSAC's response, in contrast to the minimal disturbances in *Risky* RUN-DSAC. Thus, while *Conservative* RUN-DSAC's policies are highly optimized for trained scenarios, they prove inadequate under substantially altered control conditions.

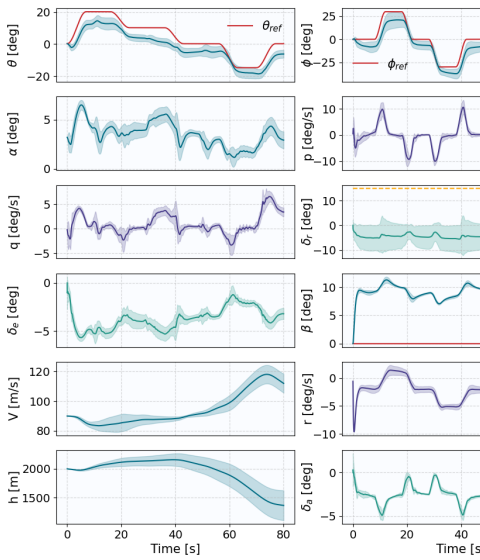


Fig. 19 State time-traces of the *Risky* RUN-DSAC policy in the immobilized rudder scenario.

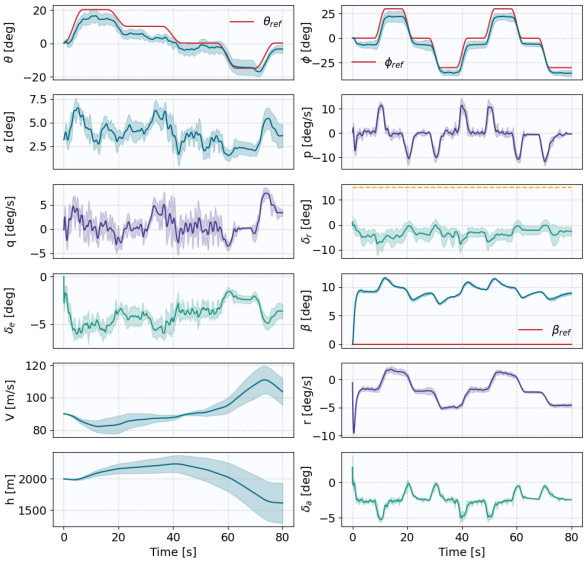


Fig. 20 State time-traces of the *Conservative* RUN-DSAC policy in the immobilized rudder scenario.

V. Conclusion

This paper introduces the Returns Uncertainty-Navigated Distributional Soft Actor-Critic (RUN-DSAC) algorithm to advance model-free flight controllers for non-linear, fully-coupled aerospace dynamics. The integration of uncertainty quantification within the RL decision-making framework contributes to the synthesis of intelligent, adaptive, and safety-centric autonomous flight controllers.

We show that RUN-DSAC, particularly its *Conservative* variant, significantly enhances learning efficiency and attitude tracking accuracy in trained flight conditions over state-of-the-art algorithms like SAC and DSAC. The improvement is attributed to the algorithm's ability to leverage the full distribution of returns, prioritizing actions with minimal uncertainty. This approach leads to more predictable and stable policies, which are crucial in the safety-critical domain of aviation.

In scenarios involving unforeseen flight conditions and in-flight system failures, *Conservative* RUN-DSAC continues to showcase superior tracking performance. However, its tendency to overfit to training conditions results in increased sensitivity and heightened oscillations under significantly altered flight dynamics. In contrast, the *Risky* RUN-DSAC variant, which promotes exploration by favoring actions with higher uncertainty, displays robustness in novel and unexpected scenarios at the cost of reduced precision and increased variance in trained policies. This dichotomy between the *Conservative* and *Risky* variants of RUN-DSAC underscores a fundamental trade-off in RL-based flight control: the balance between the tracking accuracy and predictability of learned policies in familiar scenarios, and their adaptability and robustness in unforeseen circumstances.

In conclusion, while RUN-DSAC demonstrates significant potential in model-free flight control, its real-world applicability necessitates comprehensive validation. The reliability of these results is contingent on the simulation model's fidelity and the breadth of scenarios tested. Critical model parameters, particularly those governing aircraft dynamics and environmental interactions, may fail to fully capture the complexities of actual flight conditions. The performance of RUN-DSAC in more extreme or untested scenarios also remains uncertain, and any unforeseen interactions or emergent behaviors could pose substantial safety risks. Therefore, the algorithm's deployment in a real Cessna Citation II would require rigorous validation under a broader spectrum of conditions, more detailed modeling of aircraft and sensor dynamics, and robust safeguards against the unpredictable nature of real-world aviation environments.

A. Significance of Contributions

Improving learning performance and robustness of offline RL algorithms by leveraging uncertainty quantification in decision-making, this research stands as a valuable advancement in AI-driven model-independent aerospace control technologies. RUN-DSAC paves the way for efficient, robust, and adaptive intelligent flight controllers requiring minimal human-domain knowledge, contributing to bridging the simulation-to-reality gap. This work not only enhances the safety and reliability of fault-tolerant autonomous flight controllers but also establishes a foundation for the application of AI in various safety-critical domains.

B. Recommendations

Online Flight Control: Given the promising attributes observed in *Conservative* RUN-DSAC, future research should explore its application in online flight control settings, assessing its real-time adaptability and effectiveness in dynamically adjusting to changing flight conditions and emergencies.

Hyperparameter optimization: Further optimization of RUN-DSAC's hyperparameters, particularly the newly introduced μ_{init} and N_μ, could enhance its performance. Adjusting μ_{init} to decay from a high positive value to a negative value across certain N_μ could balance the exploration needed for robust performance with the conservatism required for policy predictability.

6-DOF Flight Control: Leveraging the learning efficiency of *Conservative* RUN-DSAC, its application in full 6-DOF flight control is promising. A hierarchical approach, integrating uncertainty assessments at different control levels, could effectively address the *curse of dimensionality* inherent to high-dimensional control tasks.

Flight-Test Validation: With RUN-DSAC showing potential for robust real-system control, validation in rigorous flight tests is crucial to bridge the simulation-to-reality gap. Initial efforts should focus on employing techniques like domain randomization and robustness training in high-fidelity simulations, followed by trials on scaled-down models.

Application Beyond Flight Control: The successful application of RUN-DSAC in flight control suggests its potential in other complex domains, such as advanced Air Traffic Management. Its predictability and safety-centric approach could improve airspace efficiency by safely reducing aircraft separation standards.

Appendix

The RUN-DSAC is outlined in algorithm 1. First, the replay buffer $\mathcal{D}$, network parameters and the temperature parameter α_T are initialized. Furthermore, initial (pre-training) $\langle s, a, r, s', d \rangle$ transitions are collected using a randomly initialized policy and stored in $\mathcal{D}$. The algorithm is run for N_{eps} episodes, with each episode executing N_{trans} transitions using the current policy $\pi_w(\cdot|s)$.

Algorithm 1 RUN-DSAC algorithm

Hyperparameters: N_q, $|\mathcal{B}|$, ζ, $\bar{\mathcal{H}}$, λ_S, λ_T, σ, κ, μ_{init}, μ_{fin}, N_{RUN}
Input: $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$, N_{eps}, N_{trans}
Result: $Z_{\theta_k}(s, a)$, $\pi_w(s)$
Initialize replay buffer $\mathcal{D}$
Initialize actor and critic parameters θ_1, θ_2, w, target parameters $\bar{\theta}_1 \leftarrow \theta_1$, $\bar{\theta}_2 \leftarrow \theta_2$, $\bar{w} \leftarrow w$
Initialize temperature α_T
Collect initial trajectories $\mathcal{T}$ and store in the replay buffer $\mathcal{D}$
for *episode* $\leftarrow$ *1 to* N_{eps} **do**
  for *transition* $\leftarrow$ *1 to* N_{trans} **do**
    Collect a $\langle s, a, r, s', d \rangle$ transition using $\pi_w(\cdot|s)$
    Sample a mini-batch $\mathcal{B} \overset{i.i.d.}{\sim} \mathcal{D}$
    Generate quantile fractions $\tau_i, i = 0, \ldots, N_q$, $\tau_j, j = 0, \ldots, N_q$
    Sample $a' \sim \pi_{\bar{w}}(\cdot|s')$
    for $i \leftarrow 0$ *to* $N_q - 1$ **do**
      $\hat{\tau}_i \leftarrow \frac{\tau_i + \tau_{i+1}}{2}$
      for $j \leftarrow 0$ *to* $N_q - 1$ **do**
        $\hat{\tau}_j \leftarrow \frac{\tau_j + \tau_{j+1}}{2}$
        $y_i \leftarrow \min_{k=1,2} Z_{\hat{\tau}_i, \bar{\theta}_k}(s', a')$
        $\delta_{ij}^k \leftarrow r + \gamma \left[y_i - \alpha_T \log \pi_{\bar{w}}(a'|s') \right] - Z_{\hat{\tau}_j, \theta_k}(s, a)$, $k = 1, 2$
      end
    end
    $J_Z(\theta_k) \leftarrow \frac{1}{N_q} \sum_{i=0}^{N_q - 1} \sum_{j=0}^{N_q - 1} (\tau_{i+1} - \tau_i) \rho_{\hat{\tau}_j}^{\kappa}(\delta_{ij}^k)$, $k = 1, 2$
    Update θ_k with $\nabla J_Z(\theta_k)$, $k = 1, 2$ (using ADAM)
    Update $\bar{\theta}_k \leftarrow \zeta \theta_k + (1 - \zeta) \bar{\theta}_k$, $k = 1, 2$
    Sample new actions $\tilde{a} \sim \pi_w(\cdot|s)$ with reparametrization trick
    $Q(s, \tilde{a}) \leftarrow \sum_{i=0}^{N-1} (\tau_{i+1} - \tau_i) \min_{k=1,2} Z_{\hat{\tau}_i, \theta_k}(s, \tilde{a})$
    $\sigma_Q \leftarrow \sqrt{\sum_{i=0}^{N-1} (\tau_{i+1} - \tau_i) \left[Z_{\hat{\tau}_i, \theta}(s, a) - Q(s, a) \right]^2}$
    $\mu \leftarrow \max(\mu_{init} - (\mu_{init} - \mu_{fin}) \frac{episode}{N_{RUN}}, 0)$
    $\mathcal{L}_T \leftarrow \|\pi_w(s) - \pi_w(s')\|_2$ $\mathcal{L}_S \leftarrow \|\pi_w(s) - \pi_w(\tilde{s})\|_2$ with $\tilde{s} \sim \mathcal{N}(s, \sigma)$
    $J_\pi(w) \leftarrow \alpha_T \log(\pi_w(\tilde{a}|s)) - \left(Q(s, \tilde{a}) + \mu \sigma_Q \right) + \lambda_T \mathcal{L}_T + \lambda_S \mathcal{L}_S$
    Update w with $\nabla J_\pi(w)$ (using ADAM)
    Update $\bar{w} \leftarrow \zeta w + (1 - \zeta) \bar{w}$
    $J_{\mathcal{H}}(\alpha_T) \leftarrow \alpha_T \bar{\mathcal{H}} - \alpha_T \log(\pi_w(\tilde{a}|s))$
    Update α_T with $\nabla J_{\mathcal{H}}(\alpha_T)$ (using ADAM)
  end
  Store the collected trajectories $\mathcal{T}$ in the replay buffer $\mathcal{D}$
end

For critic network training, two quantile fraction sets are generated, one for the current and the other for the target value distribution estimates. New actions are then sampled from the target policy $\pi_{\bar{w}}(\cdot|s')$, followed by the computation of the temporal difference error δ_{ij}^k for each quantile fraction pair. The critic loss function $J_Z(\theta_k)$ is subsequently evaluated and used to update the critic networks' parameters. The target parameters are softly updated with a mixing coefficient ζ. Next, reparametrized action samples $\tilde{a}$ are drawn to calculate the expected return $Q(s, \tilde{a})$ and the return variance σ_Q is derived. Additionally, the uncertainty modulation factor μ is linearly decayed, and the smoothness losses $\mathcal{L}_T$ and $\mathcal{L}_S$ are computed. These components are used to formulate the policy loss function $J_\pi(w)$ and its gradient

$\nabla J_\pi(w)$ is used to update the policy network parameters w and to softly update the target parameters $\bar{w}$. Furthermore, the algorithm adaptively adjusts the temperature parameter α_T by minimizing a loss function that is designed to align the current policy's entropy with a predefined target level $\bar{\mathcal{H}}$. Finally, the newly acquired $\langle s, a, r, s', d \rangle$ transitions, collectively known as trajectories, are added to the replay buffer $\mathcal{D}$.

References

[1] Stevens, B. L., Lewis, F. L., and Johnson, E. N., *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*, John Wiley & Sons, 2015.

[2] Morelli, E. A., and Klein, V., *Aircraft System Identification: Theory and Practice*, 1st ed., American Institute of Aeronautics & Astronautics, Reston, VA, 2006.

[3] Benad, J., *The Flying V - A new Aircraft Configuration for Commercial Passenger Transport*, 2015. doi:10.25967/370094.

[4] Faggiano, F., Vos, R., Baan, M., and Dijk, R., *Aerodynamic Design of a Flying V Aircraft*, 2017. doi:10.2514/6.2017-3589.

[5] Ximin, L., Gu, H., Wang, Y., Li, Z., Shen, S., and Zhang, F., "Design and implementation of a quadrotor tail-sitter VTOL UAV," 2017. doi:10.1109/ICRA.2017.7989452.

[6] Bacchini, A., and Cestino, E., "Electric VTOL Configurations Comparison," *Aerospace*, Vol. 6, 2019, p. 26. doi:10.3390/aerospace6030026.

[7] Weisshaar, T. A., "Morphing Aircraft Systems: Historical Perspectives and Future Challenges," *Journal of Aircraft*, Vol. 50, No. 2, 2013, pp. 337–353. doi:10.2514/1.C031456, URL https://arc.aiaa.org/doi/10.2514/1.C031456.

[8] Mkhoyan, T., Thakrar, N., DeBreuker, R., and Sodja, J., "Morphing wing design using integrated and distributed trailing edge morphing," *Smart Materials and Structures*, Vol. 31, 2022. doi:10.1088/1361-665X/aca18b.

[9] de Croon, G., de Clercq, K., Ruijsink, R., Remes, B., and de Wagter, C., "Design, Aerodynamics, and Vision-Based Control of the DelFly," *International Journal of Micro Air Vehicles*, Vol. 1, No. 2, 2009, pp. 71–97. doi:10.1260/175682909789498288, URL https://doi.org/10.1260/175682909789498288.

[10] De Croon, G., Perçin, M., Remes, B., Ruijsink, R., and De Wagter, C., *The DelFly*, Springer Netherlands, Dordrecht, 2016. doi:10.1007/978-94-017-9208-0, URL http://link.springer.com/10.1007/978-94-017-9208-0.

[11] Wang, X., "Flexible Aircraft Gust Load Alleviation with Incremental Nonlinear Dynamic Inversion," , Feb. 2019. URL https://xueruiwangtud.github.io/publication/flexiblegla/.

[12] Wang, X., "Nonlinear Incremental Control for Flexible Aircraft Trajectory Tracking and Load Alleviation," , Aug. 2021. URL https://xueruiwangtud.github.io/publication/glider/.

[13] Gregory, P., and Center, U. S. A. F. W. A. D., *Proceedings of the Self Adaptive Flight Control Systems Symposium, Wright Air Development Center, 13 and 14 Jan., 1959*, WADC technical report, WADC, 1959. URL https://books.google.nl/books?id=0COgnQEACAAJ.

[14] Ignatyev, D. I., Shin, H.-S., and Tsourdos, A., "Two-layer adaptive augmentation for incremental backstepping flight control of transport aircraft in uncertain conditions," *Aerospace Science and Technology*, Vol. 105, 2020, p. 106051. doi:10.1016/j.ast.2020.106051, URL https://www.sciencedirect.com/science/article/pii/S1270963820307331.

[15] "Final Report: Accident to Airbus A330-203 registered F-GZCP, Air France AF 447 Rio de Janeiro - Paris, 1st June 2009 \textbar AAIU.ie," , ???? URL http://www.aaiu.ie/node/687.

[16] Khargonekar, P., Petersen, I., and Zhou, K., "Robust stabilization of uncertain linear systems: quadratic stabilizability and H/sup infinity / control theory," *IEEE Transactions on Automatic Control*, Vol. 35, No. 3, 1990, pp. 356–361. doi:10.1109/9.50357.

[17] Chen, D., "Reliable H control for uncertain affine nonlinear systems," *Proceedings of the 29th Chinese Control Conference*, 2010, pp. 1933–1938.

[18] Sieberling, S., Chu, Q. P., and Mulder, J. A., "Robust Flight Control Using Incremental Nonlinear Dynamic Inversion and Angular Acceleration Prediction," *Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 6, 2010, pp. 1732–1742. doi:10.2514/1.49978, URL https://arc.aiaa.org/doi/10.2514/1.49978.

[19] Skogestad, S., and Postlethwaite, I., "Multivariable Feedback Control: Analysis and Design," 2005.

[20] Caverly, R. J., Girard, A. R., Kolmanovsky, I. V., and Forbes, J. R., "Nonlinear Dynamic Inversion of a Flexible Aircraft," *IFAC-PapersOnLine*, Vol. 49, No. 17, 2016, pp. 338–342. doi:10.1016/j.ifacol.2016.09.058, URL https://www.sciencedirect.com/science/article/pii/S2405896316315300.

[21] Li, Y., Liu, X., Lu, P., He, Q., Ming, R., and Zhang, W., "Angular acceleration estimation-based incremental nonlinear dynamic inversion for robust flight control," *Control Engineering Practice*, Vol. 117, 2021, p. 104938. doi:10.1016/j.conengprac.2021.104938, URL https://www.sciencedirect.com/science/article/pii/S096706612100215X.

[22] Acquatella, P., Van Kampen, E.-J., and Chu, Q., *Incremental backstepping for robust nonlinear flight control*, 2013.

[23] Smeur, E. J. J., Chu, Q., and de Croon, G. C. H. E., "Adaptive Incremental Nonlinear Dynamic Inversion for Attitude Control of Micro Air Vehicles," *Journal of Guidance, Control, and Dynamics*, Vol. 39, No. 3, 2016, pp. 450–461. doi:10.2514/1.G001490, URL https://arc.aiaa.org/doi/10.2514/1.G001490.

[24] Sun, S., Wang, X., Chu, Q., and Visser, C. d., "Incremental Nonlinear Fault-Tolerant Control of a Quadrotor With Complete Loss of Two Opposing Rotors," *IEEE Transactions on Robotics*, Vol. 37, No. 1, 2021, pp. 116–130. doi:10.1109/TRO.2020.3010626.

[25] Cordeiro, R. A., Azinheira, J. R., and Moutinho, A., "Robustness of Incremental Backstepping Flight Controllers: The Boeing 747 Case Study," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 57, No. 5, 2021, pp. 3492–3505. doi:10.1109/TAES.2021.3082663.

[26] Goodfellow, I., Bengio, Y., and Courville, A., "Deep Learning," , Nov. 2016. URL https://mitpress.mit.edu/9780262035613/deep-learning/, publication Title: MIT Press.

[27] Sutton, R. S., and Barto, A. G., *Reinforcement learning: an introduction*, second edition ed., Adaptive computation and machine learning series, The MIT Press, Cambridge, Massachusetts, 2018.

[28] Hoogvliet, J., "Hierarchical Reinforcement Learning for Model-Free Flight Control: A sample efficient tabular approach using Q(lambda)-learning and options in a traditional flight control structure," 2019. URL https://repository.tudelft.nl/islandora/object/uuid%3Ad66efdb7-d7c7-4c44-9b50-64678ffdf60d.

[29] Gu, S., Yang, L., Du, Y., Chen, G., Walter, F., Wang, J., Yang, Y., and Knoll, A., "A Review of Safe Reinforcement Learning: Methods, Theory and Applications," , Feb. 2023. doi:10.48550/arXiv.2205.10330, URL http://arxiv.org/abs/2205.10330, arXiv:2205.10330 [cs].

[30] Heyer, S., "Reinforcement Learning for Flight Control: Learning to Fly the PH-LAB," 2019. URL https://repository.tudelft.nl/islandora/object/uuid%3Adc63cae7-4289-47c7-889e-253f7abd7c72.

[31] Feith, R., "Safe Reinforcement Learning in Flight Control: Introduction to Safe Incremental Dual Heuristic Programming," 2020. URL https://repository.tudelft.nl/islandora/object/uuid%3A07f53daa-b236-4bb9-b010-bc6654383744.

[32] Hutsebaut-Buysse, M., Mets, K., and Latré, S., "Hierarchical Reinforcement Learning: A Survey and Open Research Challenges," *Machine Learning and Knowledge Extraction*, Vol. 4, No. 1, 2022, pp. 172–221. doi:10.3390/make4010009, URL https://www.mdpi.com/2504-4990/4/1/9, number: 1 Publisher: Multidisciplinary Digital Publishing Institute.

[33] Gatti, G., "Shielded reinforcement learning for flight control," 2023. URL https://repository.tudelft.nl/islandora/object/uuid%3A1c4f10ef-c279-4969-8943-41c46e6b2d8f.

[34] Dally, K., and van Kampen, E.-J., "Soft Actor-Critic Deep Reinforcement Learning for Fault Tolerant Flight Control," *AIAA SCITECH 2022 Forum*, 2022. doi:10.2514/6.2022-2078, URL http://arxiv.org/abs/2202.09262.

[35] Seres, P., "Distributional Reinforcement Learning for Flight Control: A risk-sensitive approach to aircraft attitude control using Distributional RL," 2022. URL https://repository.tudelft.nl/islandora/object/uuid%3A6cd3efd1-b755-4b04-8b9b-93f9dabb6108.

[36] Vieira dos Santos, L., "Safe & Intelligent Control: Fault-tolerant Flight Control with Distributional and Hybrid Reinforcement Learning using DSAC and IDHP," 2023. URL https://repository.tudelft.nl/islandora/object/uuid%3A10f5fa68-f934-414a-9067-988f51f098cb.

[37] Bellemare, M. G., Dabney, W., and Munos, R., "A Distributional Perspective on Reinforcement Learning," , Jul. 2017. doi:10.48550/arXiv.1707.06887, URL http://arxiv.org/abs/1707.06887.

[38] Dabney, W., Rowland, M., Bellemare, M. G., and Munos, R., "Distributional Reinforcement Learning with Quantile Regression," , Oct. 2017. doi:10.48550/arXiv.1710.10044, URL http://arxiv.org/abs/1710.10044.

[39] Dabney, W., Ostrovski, G., Silver, D., and Munos, R., "Implicit Quantile Networks for Distributional Reinforcement Learning," , Jun. 2018. doi:10.48550/arXiv.1806.06923, URL http://arxiv.org/abs/1806.06923.

[40] Liu, C., van Kampen, E.-J., and de Croon, G. C. H. E., "Adaptive Risk-Tendency: Nano Drone Navigation in Cluttered Environments with Distributional Reinforcement Learning," , Sep. 2022. doi:10.48550/arXiv.2203.14749, URL http://arxiv.org/abs/2203.14749.

[41] Chan, A. J., and van der Schaar, M., "Scalable Bayesian Inverse Reinforcement Learning," , Mar. 2021. doi:10.48550/arXiv.2102.06483, URL http://arxiv.org/abs/2102.06483.

[42] Fortuin, V., Garriga-Alonso, A., Ober, S. W., Wenzel, F., Rätsch, G., Turner, R. E., van der Wilk, M., and Aitchison, L., "Bayesian Neural Network Priors Revisited," , Mar. 2022. doi:10.48550/arXiv.2102.06571, URL http://arxiv.org/abs/2102.06571.

[43] Buckman, J., "A Sober Look at Bayesian Neural Networks," , ???? URL https://jacobbuckman.com/2020-01-17-a-sober-look-at-bayesian-neural-networks/.

[44] Roman, I., Santana, R., Mendiburu, A., and Lozano, J. A., "Evolution of Gaussian Process kernels for machine translation post-editing effort estimation," *Annals of Mathematics and Artificial Intelligence*, Vol. 89, No. 8, 2021, pp. 835–856. doi:10.1007/s10472-021-09751-5, URL https://doi.org/10.1007/s10472-021-09751-5.

[45] Bertsekas, D., and Tsitsiklis, J., "Neuro-dynamic programming: an overview," *Proceedings of 1995 34th IEEE Conference on Decision and Control*, Vol. 1, 1995, pp. 560–564 vol.1. doi:10.1109/CDC.1995.478953, URL https://ieeexplore.ieee.org/document/478953, iSSN: 0191-2216.

[46] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M., "Playing Atari with Deep Reinforcement Learning," , Dec. 2013. doi:10.48550/arXiv.1312.5602, URL http://arxiv.org/abs/1312.5602.

[47] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D., "Human-level control through deep reinforcement learning," *Nature*, Vol. 518, No. 7540, 2015, pp. 529–533. doi:10.1038/nature14236, URL https://www.nature.com/articles/nature14236.

[48] Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M., "Deterministic Policy Gradient Algorithms," *Proceedings of the 31st International Conference on Machine Learning*, PMLR, 2014, pp. 387–395. URL https://proceedings.mlr.press/v32/silver14.html, iSSN: 1938-7228.

[49] Schulman, J., Levine, S., Moritz, P., Jordan, M. I., and Abbeel, P., "Trust Region Policy Optimization," , Apr. 2017. doi:10.48550/arXiv.1502.05477, URL http://arxiv.org/abs/1502.05477.

[50] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O., "Proximal Policy Optimization Algorithms," , Aug. 2017. doi:10.48550/arXiv.1707.06347, URL http://arxiv.org/abs/1707.06347.

[51] Miller, W. T., Sutton, R. S., and Werbos, P. J., "A Menu of Designs for Reinforcement Learning Over Time," *Neural Networks for Control*, MIT Press, 1995, pp. 67–95. URL https://ieeexplore.ieee.org/document/6300641, conference Name: Neural Networks for Control.

[52] Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S., "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor," , Aug. 2018. doi:10.48550/arXiv.1801.01290, URL http://arxiv.org/abs/1801.01290.

[53] Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., and Levine, S., "Soft Actor-Critic Algorithms and Applications," , Jan. 2019. doi:10.48550/arXiv.1812.05905, URL http://arxiv.org/abs/1812.05905.

[54] Fujimoto, S., van Hoof, H., and Meger, D., "Addressing Function Approximation Error in Actor-Critic Methods," , Oct. 2018. doi:10.48550/arXiv.1802.09477, URL http://arxiv.org/abs/1802.09477.

[55] Bellemare, M. G., Dabney, W., and Rowland, M., *Distributional Reinforcement Learning*, MIT Press, 2023. URL http://www.distributional-rl.org.

[56] McCaffrey, B. J., and 08/16/2021, "Wasserstein Distance Using C# and Python -," , ???? URL https://visualstudiomagazine.com/articles/2021/08/16/wasserstein-distance.aspx, publication Title: Visual Studio Magazine.

[57] Ma, X., Xia, L., Zhou, Z., Yang, J., and Zhao, Q., "DSAC: Distributional Soft Actor Critic for Risk-Sensitive Reinforcement Learning," , Jun. 2020. doi:10.48550/arXiv.2004.14547, URL http://arxiv.org/abs/2004.14547.

[58] Duan, J., Guan, Y., Li, S. E., Ren, Y., and Cheng, B., "Distributional Soft Actor-Critic: Off-Policy Reinforcement Learning for Addressing Value Estimation Errors," *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 33, No. 11, 2022, pp. 6584–6598. doi:10.1109/TNNLS.2021.3082568, URL http://arxiv.org/abs/2001.02811, arXiv:2001.02811 [cs, eess].

[59] Huber, P. J., "Robust Estimation of a Location Parameter," *The Annals of Mathematical Statistics*, Vol. 35, No. 1, 1964, pp. 73–101. doi:10.1214/aoms/1177703732, URL https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-35/issue-1/Robust-Estimation-of-a-Location-Parameter/10.1214/aoms/1177703732.full.

[60] Wang, S. S., "A Class of Distortion Operators for Pricing Financial and Insurance Risks," *The Journal of Risk and Insurance*, Vol. 67, No. 1, 2000, pp. 15–36. doi:10.2307/253675, URL https://www.jstor.org/stable/253675, publisher: [American Risk and Insurance Association, Wiley].

[61] Sobel, M. J., "The variance of discounted Markov decision processes," *Journal of Applied Probability*, Vol. 19, No. 4, 1982, pp. 794–802. doi:10.2307/3213832, URL https://www.cambridge.org/core/journals/journal-of-applied-probability/article/abs/variance-of-discounted-markov-decision-processes/AA4549BFA70081B27C0092F4BF9C661A, publisher: Cambridge University Press.

[62] Di Castro, D., Tamar, A., and Mannor, S., "Policy Gradients with Variance Related Risk Criteria," , Jun. 2012. doi:10.48550/arXiv.1206.6404, URL http://arxiv.org/abs/1206.6404, arXiv:1206.6404 [cs, math, stat].

[63] A., P. L., and Ghavamzadeh, M., "Variance-Constrained Actor-Critic Algorithms for Discounted and Average Reward MDPs," , Mar. 2015. doi:10.48550/arXiv.1403.6530, URL http://arxiv.org/abs/1403.6530, arXiv:1403.6530 [cs, math, stat].

[64] Ba, J. L., Kiros, J. R., and Hinton, G. E., "Layer Normalization," , Jul. 2016. doi:10.48550/arXiv.1607.06450, URL http://arxiv.org/abs/1607.06450.

[65] Kingma, D. P., and Ba, J., "Adam: A Method for Stochastic Optimization," , Jan. 2017. doi:10.48550/arXiv.1412.6980, URL http://arxiv.org/abs/1412.6980.

[66] Mysore, S., Mabsout, B., Mancuso, R., and Saenko, K., "Regularizing Action Policies for Smooth Control with Reinforcement Learning," , May 2021. doi:10.48550/arXiv.2012.06644, URL http://arxiv.org/abs/2012.06644.

[67] van den Hoek, M. A., de Visser, C. C., and Pool, D. M., "Identification of a Cessna Citation II Model Based on Flight Test Data," *Advances in Aerospace Guidance, Navigation and Control*, edited by B. Dołęga, R. Głębocki, D. Kordos, and M. Żugaj, Springer International Publishing, Cham, 2018, pp. 259–277. doi:10.1007/978-3-319-65283-2_14.

[68] Konatala, R., Van Kampen, E.-J., and Looye, G., "Reinforcement Learning based Online Adaptive Flight Control for the Cessna Citation II(PH-LAB) Aircraft," *AIAA Scitech 2021 Forum*, AIAA SciTech Forum, American Institute of Aeronautics and Astronautics, 2021. doi:10.2514/6.2021-0883, URL https://arc.aiaa.org/doi/10.2514/6.2021-0883.

[69] Grondman, F., Looye, G. H. N., Kuchar, R. O., Chu, Q. P., and van Kampen, E., "Design and flight testing of incremental nonlinear dynamic inversion based control laws for a passenger aircraft," *AIAA Guidance, Navigation, and Control*, , No. 210039, 2018. doi:10.2514/6.2018-0385, URL https://repository.tudelft.nl/islandora/object/uuid%3A964e1942-5c83-45e7-8ace-507a33ea3146, publisher: American Institute of Aeronautics and Astronautics Inc. (AIAA).

[70] Teirlinck, C., "Reinforcement Learning for Flight Control: Hybrid Offline-Online Learning for Robust and Adaptive Fault-Tolerance," 2022. URL https://repository.tudelft.nl/islandora/object/uuid%3Adae2fdae-50a5-4941-a49f-41c25bea8a85.

[71] Moorhouse, D. J., and Woodcock, R. J., "Background Information and User Guide for MIL-F-8785C, Military Specification - Flying Qualities of Piloted Airplanes," Tech. rep., Air Force Wright Aeronautical Laboratories, Jul. 1982. URL https://apps.dtic.mil/sti/citations/ADA119421, section: Technical Reports.