
Uncertainty Quantification for Flow-Based Vision-Language-Action Models

Ralf Römer
TU Munich

Maximilian Seeliger
ETH Zurich

Saida Liu
TU of Munich

Ben Sturgis
TU Munich

Marco Bagatella
ETH Zurich
MPI IS Tübingen

Daniel Marta
ETH Zurich

Andreas Krause
ETH Zurich

Angela P. Schoellig
TU Munich

Abstract

Vision-language-action models (VLAs) combine vision-language backbones with expressive generative action heads trained via flow matching on large-scale robotic datasets. Despite their strong empirical performance in robotic manipulation, VLAs lack mechanisms to quantify confidence in their predictions and to detect when their actions may be unreliable. This presents a critical limitation for real-world deployment in non-stationary environments, where models inevitably encounter scenarios outside their pretraining distribution and may fail without warning. To address this, we derive an efficient method for quantifying epistemic uncertainty in flow-matching models by leveraging velocity-field disagreement (VFD) across a small ensemble. We successfully use this uncertainty estimate for failure detection during deployment and active fine-tuning of flow-based VLAs. To this end, we propose SAVE, a framework for uncertainty-guided active multitask fine-tuning that reduces the number of costly expert demonstrations required to adapt VLAs to new tasks. Through extensive experiments on the LIBERO benchmark, we demonstrate that VFD yields better-calibrated uncertainty estimates predictive of downstream performance, that VFD achieves strong performance in detecting failures, and that uncertainty-guided data acquisition with SAVE requires at least 22% fewer samples than baselines. In summary, our work shows that quantifying epistemic uncertainty in flow-based VLAs improves both failure awareness and adaptation. Project website: tum-lsy.github.io/uq_vla/.

1 Introduction

The premise of vision-language-action models (VLAs) [67, 39, 54] is to bring the large-scale pre-training paradigm that has been successful in deep learning in recent years [17, 20, 6] into the physical world. Driven by the increasing availability of large-scale robotics datasets [38, 53], modern VLAs have demonstrated impressive performance in multitask learning, exhibiting remarkable robustness and emergent zero-shot capabilities [10], particularly for tasks that require a nuanced understanding of images and language. Architecturally, the state of the art has mostly converged on a powerful recipe [54, 64, 57, 10]: a large pre-trained vision-language backbone for semantic understanding, followed by an action expert trained via flow matching [45, 22] to produce smooth, continuous robot actions. Flow-based VLAs provide stable training, relatively fast inference, and the expressivity needed to model the complex, highly multimodal distributions of real-world physical demonstrations.

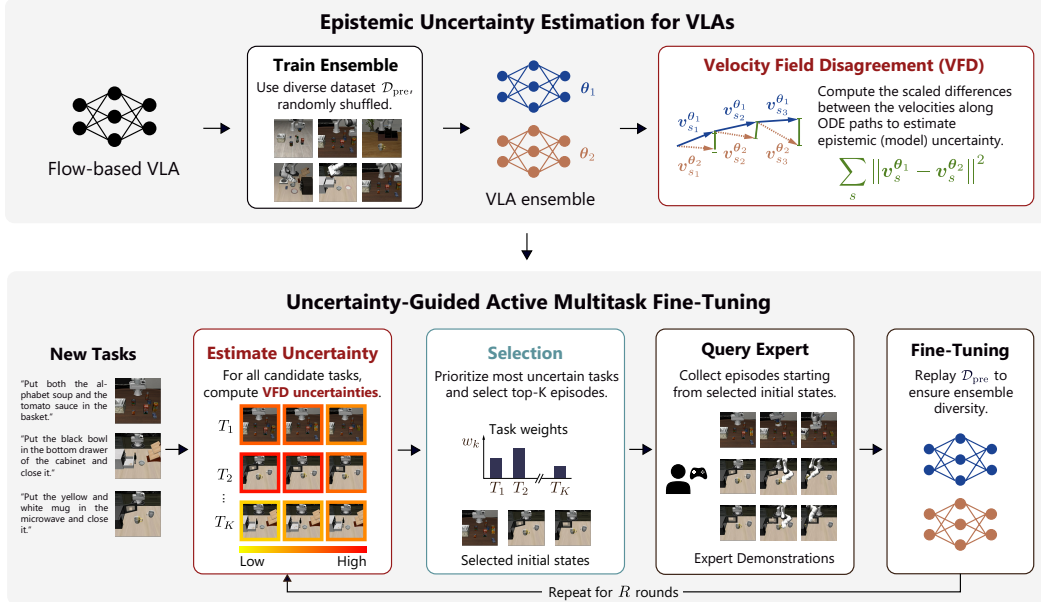


Figure 1: **Top:** VFD quantifies epistemic uncertainty by measuring scaled differences between ensembled velocity fields. **Bottom:** SAVE prioritizes tasks by their mean VFD uncertainty and, for the most uncertain initial observations within each sampled task, requests an expert demonstration. The models are then fine-tuned using new and replay data, yielding data-efficient multitask adaptation.

VLAs are pre-trained on large-scale static datasets that cover a broad distribution of objects and tasks [38, 53, 3]. However, deploying these models in the real world inevitably exposes them to non-stationarity. Users repurpose robots for novel tasks, object appearances may change, and environments evolve. In such scenarios, pre-trained VLAs may perform well on some tasks but fail completely on others, without explicit knowledge of which tasks are fully understood from pre-training and which require additional data. The inability of these models to communicate *what they do not know* prevents robust self-improvement and timely failure detection. VLAs confidently execute erratic actions in out-of-distribution scenarios [2, 27, 58] rather than abstaining or asking for help when uncertain [55], raising significant safety concerns [43, 27, 59]. Crucially, robustly adapting VLAs to new domains currently requires collecting large numbers of human expert demonstrations [52], which is prohibitively expensive.

To address these problems, our work focuses on uncertainty quantification for flow-based VLAs. Leveraging the generative architecture of these models, we propose an efficient method for estimating epistemic uncertainty by measuring divergence between ensembled velocity fields. Building on this estimator, we propose sample-efficient active fine-tuning via velocity-field epistemic uncertainty (SAVE) for VLAs. In summary, our main contributions are:

- We introduce a mathematically grounded epistemic uncertainty estimator for flow-matching models based on velocity field disagreement (VFD).
- We propose SAVE, a framework for uncertainty-guided active fine-tuning of flow-based VLAs that uses VFD to prioritize tasks and initial states for expert demonstration collection.
- We empirically evaluate the effectiveness of our methods, demonstrating that VFD predicts task performance and detects deployment failures more reliably than baselines, and that SAVE reduces the amount of expert data required for multitask adaptation by 22%.

2 Related Work

Vision-Language-Action Models. VLAs have emerged as a prevalent approach for learning manipulation policies, by fine-tuning pre-trained vision-language backbones on robotic data [39, 67, 54, 64].

Earlier VLAs map vision-language observations to discrete or continuous action tokens via regression or classification heads [39, 67]. In this work, we focus on recent flow matching-based VLAs [54, 64], which provide an expressive framework for modeling the highly multimodal action distributions in expert demonstrations by treating behavior cloning as a conditional generation problem [54, 64]. However, flow-based VLAs do not natively express uncertainty. Therefore, they cannot inherently identify situations in which they lack knowledge of which action to perform [2, 42], preventing safe, autonomous real-world deployment. Fine-tuning a VLA requires collecting many expensive expert demonstrations [52]. This motivates estimating *where* the model is most uncertain: before deployment, to prioritize situations for querying an expert, and during deployment, to detect failures.

Uncertainty Quantification for Generative Models. Uncertainty quantification in deep learning aims to estimate the trustworthiness of model outputs [1] and typically decomposes uncertainty into epistemic (model) and aleatoric (data) uncertainty [32]. While the latter is an irreducible property of the data generation process, epistemic uncertainty can be reduced by collecting new and, importantly, the right data. Depending on the model class, uncertainty quantification has been instantiated in many different ways. Test-time dropout [47] and inference on augmented inputs [7] may, for instance, be applied directly at test time and are mostly architecture-agnostic. More powerful approaches, such as deep ensembles [41] and Laplace approximations [16], require training multiple models or modifying the training procedure, in exchange for more accurate uncertainty estimates. With the rise of modern generative models [30, 45], which, despite their expressivity, often remain overconfident [51, 36], several recent works have investigated uncertainty quantification for this model class. Jazbec et al. [33] apply a last-layer Laplace approximation to diffusion models and measure variability in a semantic representation space using a Gaussian approximation, which may, however, be unsuitable for multi-modal distributions. DECU [9] trains an ensemble of latent diffusion models and performs pairwise distance estimation using the denoising means. Other recent works rely on similarity metrics in language space [24], leverage hypernetworks [13], or train explicit confidence predictors [50]. Closer to our setting, Ju et al. [34] estimate epistemic uncertainty for pre-trained VLMs through a Riemannian formulation of flow matching, but do not target action generation or active fine-tuning of VLAs. Uncertainty quantification is also an active area of research in autoregressive models [63], with domain-specific techniques spanning from differential entropy over vocabulary [44], to perplexity [56, 23], and Dirichlet evidence [48], but these methods primarily measure aleatoric uncertainty, i.e., data ambiguity. In this work, we focus on estimating epistemic uncertainty in flow-matching models, which have become the state of the art in generative modeling.

Active Learning. A direct application of reliable uncertainty estimates is in active data selection, a well-established research field with deep roots in experimental design [12] and active learning [62], including Bayesian disagreement-based acquisition for deep models [26, 40]. While a large body of literature has proposed domain-agnostic approaches [5, 31], several recent works have focused on active learning in MDPs, particularly for learning policies [35, 28]. Several existing approaches operate offline [29, 2], and filter a fixed dataset based on mutual information between states and actions or influence detection in a static setting. A long-running line of research [15, 37] operates instead in an online setting, and tries to infer *when* a human labeling effort is required. Closer to our approach, Bagatella et al. [8] also focus on an iterative setting but propose an algorithm that queries *entire* expert demonstrations to maximize information gain over multi-task expert trajectories. While this approach has strong regret guarantees, it does not scale to modern VLAs. In contrast, we directly leverage the generative formulation of state-of-the-art VLAs for uncertainty-guided active fine-tuning.

3 Preliminaries

Bayesian Uncertainty Quantification. We consider a dataset $\mathcal{D} = \{(\mathbf{x}^{(n)}, \mathbf{y}^{(n)})\}_{n=1}^N$ and a neural network with parameters $\theta \in \Theta$ trained to learn the relationship between $\mathbf{y} \in \mathcal{Y}$ and $\mathbf{x} \in \mathcal{X}$. Bayesian modeling considers the posterior predictive distribution

$$p(\mathbf{x} | \mathbf{y}, \mathcal{D}) = \int p(\mathbf{x} | \mathbf{y}, \theta) p(\theta | \mathcal{D}) d\theta, \quad (1)$$

which involves the posterior $p(\theta | \mathcal{D})$ that is generally intractable for deep neural networks [32]. For this reason, the integral (1) has to be approximated in practice by an MC estimate $p(\mathbf{x} | \mathbf{y}, \mathcal{D}) \approx \frac{1}{M} \sum_{i=1}^M p(\mathbf{x} | \mathbf{y}, \theta_i)$, $\theta_i \sim p(\theta | \mathcal{D})$, where approaches such as ensembling [41] or dropout [25] can be used to obtain $\theta_1, \dots, \theta_M \sim p(\theta | \mathcal{D})$. The total uncertainty

in the generation \mathbf{x} for an input \mathbf{y} is given by the entropy of the predictive distribution (1), $H[p(\mathbf{x} | \mathbf{y}, \mathcal{D})] = -\int p(\mathbf{x} | \mathbf{y}, \mathcal{D}) \log p(\mathbf{x} | \mathbf{y}, \mathcal{D}) d\mathbf{x}$, which includes both irreducible aleatoric uncertainty and epistemic uncertainty, which can be reduced by collecting additional data. Subtracting the aleatoric component, given by the expected entropy, yields the epistemic uncertainty [32]

$$\begin{aligned} I(\mathbf{x}, \boldsymbol{\theta} | \mathbf{y}, \mathcal{D}) &= H[p(\mathbf{x} | \mathbf{y}, \mathcal{D})] - \mathbb{E}_{\boldsymbol{\theta} \sim p(\cdot | \mathcal{D})} [H[p(\mathbf{x} | \mathbf{y}, \boldsymbol{\theta})]] \\ &= \mathbb{E}_{\boldsymbol{\theta} \sim p(\cdot | \mathcal{D})} [D_{\text{KL}}(p(\mathbf{x} | \mathbf{y}, \boldsymbol{\theta}) || p(\mathbf{x} | \mathbf{y}, \mathcal{D}))], \end{aligned} \quad (2)$$

which is equivalent to the mutual information between \mathbf{x} and $\boldsymbol{\theta}$ and can be viewed as the expected disagreement between a model learned from the data and the true posterior (1) at a test input \mathbf{y} .

Flow Matching. The goal of generative modeling is to learn a probability distribution q from a set of samples \mathcal{D} . We first consider unconditional generation, where $\mathcal{D} = \{\mathbf{x}^{(n)}\}_{n=1}^N \sim q$, $\mathbf{x}^{(n)} \in \mathbb{R}^d$. Flow matching models a time-dependent velocity field $\mathbf{u}_s : [0, 1] \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ that generates a probability path $(p_s)_{0 \leq s \leq 1}$ transforming a source distribution p_0 (e.g., a Gaussian) to the data distribution $p_1 = q$. This means the flow $\phi_s : [0, 1] \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ solving the ODE $\frac{d}{ds} \phi_s(\mathbf{x}) = \mathbf{u}_s(\phi_s(\mathbf{x}))$, $\phi_0(\mathbf{x}) = \mathbf{x}$ satisfies $\mathbf{x}_s = \phi_s(\mathbf{x}_0) \sim p_s$ for $\mathbf{x}_0 \sim p_0$. As $\mathbf{u}_s(\mathbf{x})$ is unknown, conditional flow matching [45] constructs p_s as the marginal of simpler conditional paths $p_s(\mathbf{x} | \mathbf{x}_1)$, where $\mathbf{x}_1 \sim q$. A common choice are optimal transport (OT) Gaussian conditional paths $p_s(\mathbf{x} | \mathbf{x}_1) = \mathcal{N}(\mathbf{x} | s\mathbf{x}_1, (1-s)^2 \mathbf{I})$, for which the conditional velocity field $\mathbf{u}_s(\mathbf{x} | \mathbf{x}_1)$ has an analytical expression. A neural network \mathbf{v}_s^θ is then trained to regress onto this target vector. Given a dataset $\mathcal{D} = \{(\mathbf{x}^{(n)}, \mathbf{y}^{(n)})\}_{n=1}^N \sim q$, a common training objective for learning $q(\mathbf{x} | \mathbf{y})$ is

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{s \sim \text{Unif}([0,1]), (\mathbf{x}_1, \mathbf{y}) \sim \text{Unif}(\mathcal{D}), \mathbf{x}_s \sim p_s(\cdot | \mathbf{x}_1)} \left[\left\| \mathbf{v}_s^\theta(\mathbf{x}_s, \mathbf{y}) - \mathbf{u}_s(\mathbf{x}_s | \mathbf{x}_1) \right\|_2^2 \right]. \quad (3)$$

New samples $\mathbf{x} \sim p_1(\cdot | \mathbf{y})$ from the learned distribution can be generated, for example, through Euler integration starting from $\mathbf{x}_0 \sim p_0$ with step size $\delta s = 1/N_s$ and $\mathbf{x}_{s+\delta s} = \mathbf{x}_s + \mathbf{v}_s^\theta(\mathbf{x}_s, \mathbf{y}) \delta s$.

Vision-Language-Action Models. A VLA is a robot policy $\pi_\theta : \mathcal{O} \rightarrow \Delta(\mathcal{A})$ that maps multimodal observations $\mathbf{o}_t \in \mathcal{O}$ to distributions over actions $\mathbf{a}_t \in \mathcal{A}$. The observations typically contain the robot’s proprioceptive state \mathbf{s}_t , camera images \mathbf{I}_t and language instructions \mathbf{l} , i.e., $\mathbf{o}_t = (\mathbf{s}_t, \mathbf{I}_t, \mathbf{l})$. Flow-based VLAs [54, 57, 64, 10] are typically trained through behavior cloning from large-scale data $\mathcal{D} = \{(\mathbf{o}^{(n)}, \mathbf{a}^{(n)})\}_{n=1}^N \sim q$ using the flow-matching loss (3) with $\mathbf{x} = \mathbf{A}_t = (\mathbf{a}_t, \mathbf{a}_{t+1}, \dots, \mathbf{a}_{t+H})$ and $\mathbf{y} = \mathbf{o}_t$, generating entire action chunks to increase temporal consistency and robustness against non-Markovian expert behavior [14, 10].

Problem Statement. At its core, our work focuses on uncertainty estimation to detect failures and guide data acquisition for VLAs. More formally, we consider a set of K tasks $\mathcal{T} = \{T_k\}_{k=1}^K$, each of which can be described by an MDP $\mathcal{M}_k = (\mathcal{O}, \mathcal{A}, P_k, R_k, \gamma)$, where \mathcal{O} and \mathcal{A} are observation and action spaces, $P_k : \mathcal{O} \times \mathcal{A} \rightarrow \Delta(\mathcal{O})$, $R_k : \mathcal{O} \rightarrow \mathbb{R}$ and $\rho_k \in \Delta(\mathcal{O})$ are task-specific dynamics, rewards and initial observation distributions, and γ is a discount factor. Given a policy $\pi : \mathcal{O} \rightarrow \Delta(\mathcal{A})$, its single-task performance is simply $J_k(\pi) = \mathbb{E}_{\pi, P_k, \rho_k} \sum_{t=0}^{\infty} \gamma^t R_k(\mathbf{o}_t)$, and its multitask performance is $J(\pi) = \frac{1}{K} \sum_{k=1}^K J_k(\pi)$. Our goal is to fine-tune a flow-based VLA π_θ (pre-trained on large-scale, diverse data) to maximize its average multitask performance. For each task $T_k \in \mathcal{T}$, we assume the availability of a pool of L candidate initial observations $\mathcal{O}_k = \{\mathbf{o}_{kl}\}_{l=1}^L \sim \rho_k$, where each \mathbf{o}_{kl} in practice contains an initial robot state, camera observations and language instructions. Without any environment interaction, a subset of observations $\hat{\mathcal{O}} \subset \cup_{k=1}^K \mathcal{O}_k$ needs to be selected, and an expert demonstrator $\pi^* \approx \text{argmax}_\pi J(\pi)$ will return a demonstration starting from each of the selected observations: $\tau_e = (\mathbf{o}_0, \mathbf{a}_0, \dots)$ with $\mathbf{a}_t \sim \pi^*(\mathbf{o}_t)$ and $\mathbf{o}_{t+1} \sim P_k(\mathbf{o}_t, \mathbf{a}_t)$ for each $\mathbf{o}_0 \in \hat{\mathcal{O}}$.

4 Epistemic Uncertainty Estimation in Flow-Matching Models

Training a flow-matching model via (3) yields an estimate $\boldsymbol{\theta}$ of the model parameters given the data \mathcal{D} . We aim to quantify whether the model, for a given conditioning \mathbf{y} , exhibits high epistemic uncertainty that could be reduced by collecting more data. A simple estimate of total predictive uncertainty is the conditional entropy $H[p(\mathbf{x} | \mathbf{y}, \boldsymbol{\theta})]$ [58]. As this quantity conflates epistemic and aleatoric uncertainty, we need to isolate the epistemic component (2). We denote $p^{\theta_i}(\mathbf{x} | \mathbf{y}) = p(\mathbf{x} | \mathbf{y}, \boldsymbol{\theta}_i)$ and

consider the MC approximation of (2) from a set of model parameters $\theta_1, \dots, \theta_M \sim p(\theta | \mathcal{D})$, i.e.,

$$I(\mathbf{x}, \theta | \mathbf{y}, \mathcal{D}) \approx \frac{1}{M} \sum_{i=1}^M D_{\text{KL}}(p^{\theta_i}(\mathbf{x} | \mathbf{y}) || p(\mathbf{x} | \mathbf{y}, \mathcal{D})) \quad (4a)$$

$$\approx \frac{1}{M} \sum_{i=1}^M D_{\text{KL}}\left(p^{\theta_i}(\mathbf{x} | \mathbf{y}) || \frac{1}{M} \sum_{j=1}^M p^{\theta_j}(\mathbf{x} | \mathbf{y})\right) \quad (4b)$$

$$\leq \frac{1}{M^2} \sum_{i,j=1}^M D_{\text{KL}}(p^{\theta_i}(\mathbf{x} | \mathbf{y}) || p^{\theta_j}(\mathbf{x} | \mathbf{y})), \quad (4c)$$

where (4c) follows by applying Jensen’s inequality to each term in the outer sum, using the convexity of the KL divergence in its second argument. Hence, the average pairwise KL divergence between flow-matching models trained on the same data \mathcal{D} , evaluated for a conditioning input \mathbf{y} , represents an approximate upper bound of the epistemic uncertainty.

In principle, the KL divergence between two flow-matching models can be estimated by computing the likelihood of individual samples by solving an augmented ODE that involves the divergence of the learned velocity field. However, computing the divergence is prohibitively expensive in high dimensions. Therefore, we derive a direct relationship between the pairwise KL divergence and the learned velocity fields. For the proof, we refer to Appendix A.2.

Theorem 4.1. *Consider two distributions $p^{\theta_1}(\mathbf{x} | \mathbf{y})$, $p^{\theta_2}(\mathbf{x} | \mathbf{y})$ induced by velocity fields $\mathbf{v}_s^{\theta_1}(\mathbf{x}, \mathbf{y})$, $\mathbf{v}_s^{\theta_2}(\mathbf{x}, \mathbf{y})$ of flow-matching models with OT Gaussian conditional probability paths. Define $\kappa_s = \frac{s}{1-s}$, and assume that, for all $s \in [0, 1]$, the marginal probability densities and velocity fields decay sufficiently fast at infinity such that $p_s^{\theta_1}(\mathbf{x} | \mathbf{y}) \mathbf{v}_s^{\theta_1}(\mathbf{x}, \mathbf{y}) \rightarrow 0$, $p_s^{\theta_2}(\mathbf{x} | \mathbf{y}) \mathbf{v}_s^{\theta_2}(\mathbf{x}, \mathbf{y}) \rightarrow 0$ as $\|\mathbf{x}\| \rightarrow \infty$. Then, the KL divergence between the two distributions is given by*

$$D_{\text{KL}}(p^{\theta_1}(\mathbf{x} | \mathbf{y}) || p^{\theta_2}(\mathbf{x} | \mathbf{y})) = \int_0^1 \kappa_s \mathbb{E}_{\mathbf{x}_s \sim p_s^{\theta_1}(\mathbf{x} | \mathbf{y})} \left[\|\mathbf{v}_s^{\theta_1}(\mathbf{x}_s, \mathbf{y}) - \mathbf{v}_s^{\theta_2}(\mathbf{x}_s, \mathbf{y})\|_2^2 \right] ds. \quad (5)$$

Note that by definition of κ_s , velocity differences at higher flow-matching times s , where the samples contain less noise, are more indicative of epistemic uncertainty. Theorem 4.1 allows us to efficiently approximate the KL divergence terms in (4c) by sampling $\mathbf{x}_0 \sim p_0$, performing Euler integration using the learned velocity field $\mathbf{v}_s^{\theta_1}$ and calculating the *velocity field disagreement (VFD)* at intermediate ODE states $\mathbf{x}_0, \mathbf{x}_{\delta s}, \dots$, i.e.,

$$D_{\text{KL}}(p^{\theta_1}(\mathbf{x} | \mathbf{y}) || p^{\theta_2}(\mathbf{x} | \mathbf{y})) \approx \frac{1}{N_s} \mathbb{E}_{\mathbf{x}_0 \sim p_0} \left[\sum_{\ell=0}^{N_s-1} \kappa_{s_\ell} \|\mathbf{v}_{s_\ell}^{\theta_1}(\mathbf{x}_{s_\ell}, \mathbf{y}) - \mathbf{v}_{s_\ell}^{\theta_2}(\mathbf{x}_{s_\ell}, \mathbf{y})\|_2^2 \right], \quad (6)$$

where $s_\ell = \ell \delta s$. Finally, denoting $\mathcal{V} = (\mathbf{v}_s^{\theta_1}, \dots, \mathbf{v}_s^{\theta_M})$, plugging (6) into (4c) and averaging over non-identical pairs, we define the VFD score for tractable epistemic uncertainty estimation as

$$u_e(\mathbf{y}; \mathcal{V}) = \frac{1}{M(M-1)N_s} \mathbb{E}_{\mathbf{x}_0 \sim p_0} \left[\sum_{\substack{i,j=1 \\ j \neq i}}^M \sum_{\ell=0}^{N_s-1} \kappa_{s_\ell} \|\mathbf{v}_{s_\ell}^{\theta_i}(\mathbf{x}_{s_\ell}^{(i)}, \mathbf{y}) - \mathbf{v}_{s_\ell}^{\theta_j}(\mathbf{x}_{s_\ell}^{(i)}, \mathbf{y})\|_2^2 \right], \quad (7)$$

where $\mathbf{x}_{s_\ell+\delta s}^{(i)} = \mathbf{x}_{s_\ell}^{(i)} + \mathbf{v}_{s_\ell}^{\theta_i}(\mathbf{x}_{s_\ell}^{(i)}, \mathbf{y})\delta s$, $\mathbf{x}_0^{(i)} = \mathbf{x}_0$. In practice, we approximate the expectation in (7) by forward integrating a small batch of B actions, leveraging GPU parallelization.

5 Uncertainty-Guided Active Multitask Fine-Tuning

Having proposed VFD for estimating epistemic uncertainty in flow-matching models, we now build on this and present SAVE for sample-efficient active multitask adaptation of flow-based VLAs. As established in Section 3, expert rollouts are unavailable for data acquisition. Instead, we must decide which demonstrations are most valuable to collect based solely on potential initial configurations for the given tasks. For this purpose, we calculate the VFD uncertainty for candidate initial observations. Higher values indicate the policy is more uncertain about how to act in a given scene, suggesting limited exposure during pre-training and a potential benefit from additional expert demonstrations.

Algorithm 1 Uncertainty-Based Active Fine-Tuning of Flow-Based VLAs with SAVE.

- 1: **Input:** Base VLA π_b , pool of initial configurations $\{\mathcal{O}_k\}_{k=1}^K$, diverse dataset \mathcal{D}_{pre} .
 - 2: **Output:** Final policy π_R .
 - 3: **Hyperparameters:** Rounds R , queries per round n_e , temperature τ , replay ratio λ .
 - 4: From π_b , train an ensemble Π_0 on \mathcal{D}_{pre} .
 - 5: Initialize buffer $\mathcal{D}_{\text{new}}^{(\leq 0)} \leftarrow \emptyset$.
 - 6: **for** $r = 0$ **to** $R - 1$ **do** \triangleright Active learning loop.
 - 7: **for** $k = 1$ **to** K **do**
 - 8: **for each** $\mathbf{o}_{kl} \in \mathcal{O}_k$ **do**
 - 9: Compute VFD uncertainty $u_{kl}^{(r)} = u_e(\mathbf{o}_{kl}; \Pi_r)$ via (7).
 - 10: Compute task uncertainty $U_k^{(r)}$ via (8).
 - 11: Compute task weights $W^{(r)}$ via (9).
 - 12: **for** $\xi = 1$ **to** n_e **do** \triangleright Data acquisition.
 - 13: Sample $k \sim \text{Cat}(W^{(r)})$.
 - 14: Select the most uncertain initial observation \mathbf{o}^* from task T_k via (10).
 - 15: Query the expert to collect a demonstration τ_e starting from \mathbf{o}^* and add τ_e to $\mathcal{D}_{\text{new}}^{(\leq r)}$.
 - 16: Construct training mixture $\mathcal{D}_{\text{train}}^{(r)}$ via (11).
 - 17: $\Pi_{r+1} \leftarrow \text{Fine-tune}(\Pi_r, \mathcal{D}_{\text{train}}^{(r)})$, $\mathcal{D}_{\text{new}}^{(\leq r+1)} \leftarrow \mathcal{D}_{\text{new}}^{(\leq r)}$.
-

VLA Ensemble. The high computational cost of pre-training foundation models makes it prohibitively expensive to train multiple VLAs from scratch to sample from the posterior. Instead, we take a single pre-trained base VLA π_b and fine-tune it M times on randomly shuffled versions of a diverse dataset \mathcal{D}_{pre} to obtain a VLA ensemble $\Pi_0 = (\pi_0^{(1)}, \dots, \pi_0^{(M)})$ for uncertainty estimation.

Uncertainty-Guided Data Acquisition. We train the VLA ensemble over R rounds of active fine-tuning. At each round r , we evaluate the current ensemble Π_r on all candidate initial observations and compute, for all tasks $k \in \{1, \dots, K\}$ and candidate initial observations $\mathbf{o}_{kl} \in \mathcal{O}_k$, $u_{kl}^{(r)} = u_e(\mathbf{o}_{kl}; \Pi_r)$, where the VFD score $u_e(\cdot)$ measuring epistemic uncertainty is defined in (7). A simple strategy would be to always select the single most uncertain candidate observation. However, this approach can overemphasize a small set of outlier scenarios, leading to poor task coverage and detrimental effects on multitask improvement. Thus, we first aggregate uncertainty at the task level using the mean score

$$U_k^{(r)} = \frac{1}{L} \sum_{l=1}^L u_{kl}^{(r)}. \quad (8)$$

It may then be tempting to directly request a batch of demonstrations from the most uncertain task; however, this is both theoretically flawed (as mutual information would need to be updated as soon as a single new demonstration is collected) and practically suboptimal (as an excessive number of demonstrations could be collected for an easy task). While posterior uncertainty can be easily computed under strong assumptions about the model class (e.g., analytically, without even observing x in the case of Gaussian Processes [8]), its estimation with neural networks remains, in general, intractable. As collecting demonstrations in batches remains practically necessary for computational reasons, we instead introduce diversity by defining a categorical sampling distribution $\text{Cat}(W^{(r)})$ with a temperature parameter $\tau \geq 0$ and weights $W^{(r)} = (w_1^{(r)}, \dots, w_K^{(r)})$ over tasks, where

$$w_k^{(r)} = \frac{\left(U_k^{(r)}\right)^\tau}{\sum_{k'=1}^K \left(U_{k'}^{(r)}\right)^\tau}. \quad (9)$$

Larger values of τ concentrate probability mass on the most uncertain tasks, and $\tau = 0$ corresponds to uniform sampling. We perform n_e expert queries per round and first sample a task index $k \sim \text{Cat}(W^{(r)})$, effectively prioritizing tasks with high estimated uncertainty while retaining exploration over all tasks. We then select the most uncertain initial observation within the sampled task,

$$\mathbf{o}_k^* = \arg \max_{\mathbf{o}_{kl} \in \mathcal{O}_k} u_{kl}^{(r)}, \quad (10)$$

and request an expert demonstration starting from $\mathbf{o}_0 = \mathbf{o}_k^*$, $\tau_e = (\mathbf{o}_0, \mathbf{a}_0, \dots)$. Repeating this procedure n_e times yields a batch of newly collected demonstrations $\mathcal{D}_{\text{new}}^{(r)}$.

Iterative Fine-tuning. Typically, fine-tuning only on newly collected demonstrations leads to forgetting previously acquired capabilities [19, 58]. To avoid this issue, we fine-tune the VLA ensemble on a mixture of pre-training data and queried data,

$$\mathcal{D}_{\text{train}}^{(r)} \sim \lambda \text{Unif}(\mathcal{D}_{\text{pre}}) + (1 - \lambda) \text{Unif}(\mathcal{D}_{\text{new}}^{(\leq r)}), \quad (11)$$

where $\mathcal{D}_{\text{new}}^{(\leq r)}$ denotes all demonstrations collected up to round r , and $\lambda \in [0, 1]$ controls the replay ratio. By determining which tasks currently need data most and then selecting the highest-uncertainty initial observations *within* those tasks, SAVE balances exploitation of uncertain scenes with exploration across tasks for sample-efficient multitask adaptation.

6 Experiments

With our experiments, we primarily aim to answer four research questions:

- Q1:** What is the best strategy for estimating epistemic uncertainty in flow-based VLAs?
- Q2:** How does uncertainty calibration affect the performance of active fine-tuning with SAVE?
- Q3:** How does uncertainty-guided data acquisition compare to optimizing for diversity alone?
- Q4:** Are uncertainty estimates informative for detecting failures during deployment?

6.1 Experimental Setup

Environments. We conduct our experiments on the LIBERO [46] benchmark, which simulates a Franka manipulator in various household environments. For active fine-tuning, we consider $K = 10$ long-horizon tasks from the most challenging suite, LIBERO-10. The initial VLA ensemble Π_0 is trained on 30 tasks from the LIBERO-Goal, Spatial, and Object suites, and we also include three active-learning tasks in \mathcal{D}_{pre} to instill prior task competence that modern VLAs increasingly possess.

Implementation Details. We use SmolVLA [64] in our experiments, which attaches an action expert trained via flow matching to a pre-trained SmolVLM-2 backbone and uses both cross- and self-attention during action generation. The policy is conditioned on two camera images (third-person and wrist), the proprioceptive state, and the task language instruction, generates action chunks of length $H = 50$, and replans every 25 environment steps, using an ODE step size $\delta s = 0.1$. We compute VFD with a batch size $B = 5$, perform $R = 15$ rounds of iterative active fine-tuning and query $n_e = 5$ expert demonstrations per round, using a replay ratio of $\lambda = 0.5$. All experiments are repeated across three random seeds, and we report the mean and standard deviation across seeds.

Baselines. We compare VFD against six baselines for uncertainty estimation in modern generative models. **Action-L2** computes the pairwise L_2 distance between independently generated action chunks from the ensemble members. **ACE** [58] computes the conditional entropy in the action chunk distribution. We adopt **DECU** [9], which performs pairwise distance estimation at a “branching” ODE timestep for denoising diffusion models (DDIM), for flow matching. Generative Uncertainty (**GU**) [33] generates samples from all ensemble members starting from the same initial noise to approximate the posterior predictive as a Gaussian, and calculates its entropy. **Entropy** [49] and **Perplexity** [23] operate on the VLM prefix, computing average per-token entropy and the exponentiated negative sum of the language tokens’ log-probabilities, respectively.

6.2 Calibration

Our uncertainty-guided data acquisition method relies on the epistemic uncertainty for an initial observation \mathbf{o}_{kl} being strongly negatively correlated with the probability of success. For this reason, we evaluate calibration using Spearman’s rank correlation coefficient ρ between the mean uncertainty and the success rate as the primary calibration metric; a value of $-\rho = 1$ indicates that the success rate decreases monotonically with increasing uncertainty. In addition, we report the Pearson correlation coefficient, measuring the strength of the linear relationship between the two quantities. We iteratively fine-tune a VLA ensemble to evaluate calibration over multiple rounds, collecting demonstrations at random each round to eliminate the effect of the data-acquisition strategy.

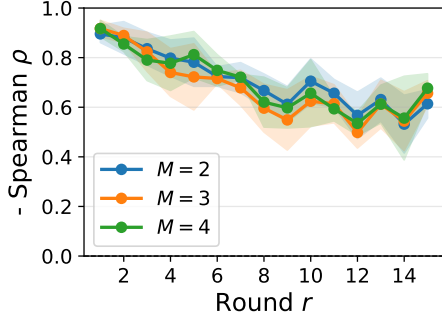


Figure 2: The VLA ensemble size has little impact on calibration, allowing for a lightweight two-member ensemble.

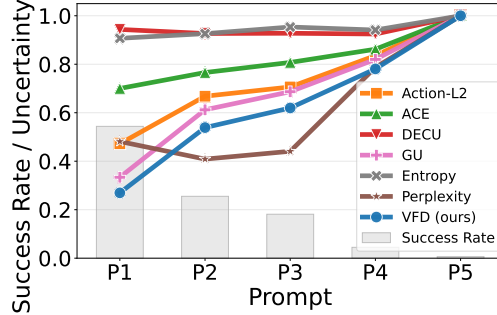


Figure 3: VFD is also well-calibrated when varying only the language prompt.

Table 1: **Calibration analysis.** Negative Spearman rank (\uparrow) and negative Pearson (\uparrow) correlation between uncertainty estimates and per-task success rates, averaged across iterative fine-tuning rounds.

Metric	Action-L2	ACE	DECU	GU	Entropy	Perplexity	VFD (ours)
-Spearman	0.50 ± 0.13	0.31 ± 0.12	0.31 ± 0.13	0.62 ± 0.00	0.10 ± 0.12	-0.04 ± 0.09	0.71 ± 0.03
-Pearson	0.48 ± 0.09	0.36 ± 0.08	0.23 ± 0.15	0.65 ± 0.02	0.23 ± 0.21	0.02 ± 0.15	0.71 ± 0.02

Small ensembles are sufficient. Since fine-tuning VLAs is expensive, it is highly desirable to have small ensembles. Figure 2 shows the impact of the ensemble size M on calibration for all fine-tuning rounds. The VLA ensembles consistently remain well-calibrated even for $M = 2$, indicating that they do not collapse to a single mode of the posterior. Due to the competitive performance and computational advantage of two-member ensembles, we adopt this strategy in all subsequent experiments.

VFD excels at estimating epistemic uncertainty. The overall calibration across all rounds is summarized in Table 1. Across all tasks and rounds, VFD is better calibrated than the baselines, achieving a 0.09 higher negative Spearman correlation than the second-best method, GU. These results establish VFD as a powerful uncertainty estimator to guide data acquisition in active multitask fine-tuning.

VFD attends to different input modalities. Since the initial observations within a task involve only variations in visual and proprioceptive input, we investigate whether VFD can capture epistemic uncertainty induced solely by varying the language command. We generate five increasingly semantically perturbed versions of each language prompt (P1 to P5) and evaluate the success rate and uncertainty for each, using the final policy π_{15} after iterative random data selection. The calibration curves in Figure 3, normalized for comparability, show that VFD is also well-calibrated with respect to the language instruction.

6.3 Active Fine-Tuning

Having shown superior calibration of VFD, we evaluate whether SAVE improves multitask performance more than diversity-based and uniform selection baselines, given the same expert demonstration budget. We compare our VFD-based acquisition rule against Action-L2 and GU, which performed best in the calibration experiments (cf. Table 1), as well as a diversity-based greedy baseline [61] that selects the most visually diverse initial observations. For the uncertainty-guided approaches, we sweep over temperature values $\tau \in \{1, 1.5, 2, 2.5, 3\}$. Further implementation details are provided in Appendix B, and additional experimental results are reported in Appendix C.

SAVE with VFD uncertainty guidance yields faster improvement. Table 2 reports the number of rounds it takes for different active fine-tuning approaches to achieve certain levels of success. SAVE w/ VFD surpasses the performance of random and diversity-based selection in the last three rounds with 50% and 45% fewer costly expert demonstrations, respectively. Compared to other approaches for uncertainty-guided data acquisition with SAVE, VFD requires at least 22% less data to achieve similar performance. After all rounds, SAVE with VFD-guided data acquisition achieves the highest success rate of 67% across tasks, outperforming the other methods by 3 to 12% given the same

Table 2: **Sample efficiency.** Number of active fine-tuning rounds (\downarrow) to reach certain success rates (SR) and final SR (\uparrow) for different demonstration selection strategies. Thresholds not reached within 15 rounds are marked as "—".

SR Threshold	Random	Diversity	SAVE w/ Action-L2	SAVE w/ GU	SAVE w/ VFD
$\geq 40\%$	5.7 ± 1.2	4.7 ± 0.5	5.0 ± 0.8	5.0 ± 1.4	5.0 ± 0.8
$\geq 45\%$	6.7 ± 1.2	7.3 ± 0.9	6.0 ± 1.4	5.7 ± 2.4	5.3 ± 0.9
$\geq 50\%$	8.7 ± 2.1	9.7 ± 1.7	9.0 ± 2.4	7.3 ± 1.9	6.0 ± 0.8
$\geq 55\%$	—	—	9.0 ± 1.0	11.0 ± 1.6	8.0 ± 1.6
$\geq 60\%$	—	—	—	12.7 ± 1.7	10.0 ± 0.8
$\geq 65\%$	—	—	—	—	12.5 ± 0.5
Final SR	54.6 ± 0.9	54.9 ± 1.3	56.8 ± 7.6	64.0 ± 2.6	67.1 ± 3.2

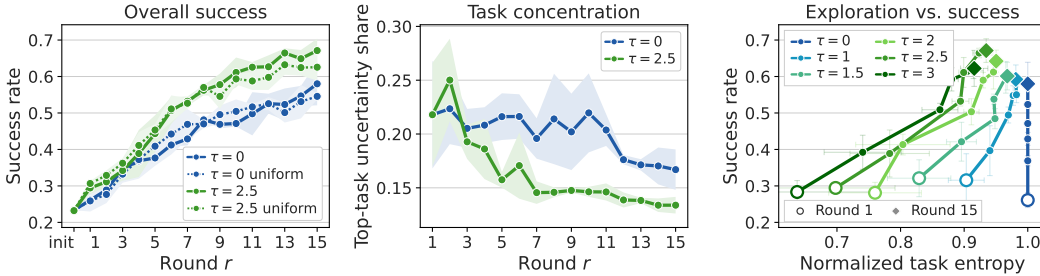


Figure 4: **Effect of the task-sampling temperature τ on SAVE.** Larger τ biases expert queries toward higher-uncertainty tasks, with uniform sampling for $\tau = 0$. **Left:** Uncertainty-guidance is beneficial both for sampling tasks and initial observations. Legend entries containing *uniform* correspond to sampling initial observations within a sampled task uniformly instead of uncertainty-guided. **Middle:** Uncertainty-based sampling more rapidly reduces the fraction of uncertainty concentrated in the most uncertain task, indicating better allocation of demonstrations to underperforming tasks. **Right:** SAVE reduces the difference in prior knowledge about tasks; for $\tau \leq 2.5$, the temperature controls an exploration–exploitation trade-off between task coverage and final success rate.

budget of expert demonstrations. Comparing the results to the calibration reported in Table 1, we find that the performance of uncertainty-guided active fine-tuning heavily depends on the quality of the uncertainty signals.

Task and initial state selection benefit from uncertainty guidance. To isolate the effects of task-level selection from choosing the exact initial configuration for the expert demonstration, we compare both parts with and without uncertainty guidance. Figure 4 (left) shows that most of the gain over random acquisition stems from the prioritization at the task level, while uncertainty-based initial observation selection yields a minor additional improvement. A key reason for this performance gain is shown in the middle of Figure 4 and in Figure 11. While pre-trained VLAs naturally exhibit different levels of uncertainty per task, our uncertainty-guided active fine-tuning strategy can reduce these differences and improve multitask performance more quickly by allocating the demonstration budget to tasks that require it most. Further, Figure 4 (right) complements this observation by showing that over multiple rounds, SAVE tends towards high-entropy categorical distributions over tasks (i.e., more diversity in the selection). But for moderate $\tau \leq 2.5$, it can actually trade off this diversity for faster improvement in multitask success by focusing on high-uncertainty tasks (cf. Figure 10).

6.4 Failure Detection

Lastly, we evaluate whether high epistemic uncertainty *during deployment* indicates imminent task failure. For this, we roll out the final VLA policy obtained after active fine-tuning 30 times per task and calculate VFD scores using the two-member ensemble at each action-generation timestep. Following prior works [65, 58], we calibrate task-specific thresholds from 10 successful rollouts using conformal prediction [4] and report accuracy and true-positive-rate (TPR).

We consider three recent baselines for detecting failures of generative policies: **ACE** [58] computes the conditional entropy of the action distribution, **STAC** [2] compares action distributions at consecutive timesteps, and **RND-OE** [58] detects OOD observations. As shown in Figure 5, high VFD is a strong signal for policy failures, achieving 67% accuracy and correctly predicting 79% of all failures (TPR). These results demonstrate the potential of our method to improve the reliability of VLAs by enabling them to express their own confidence.

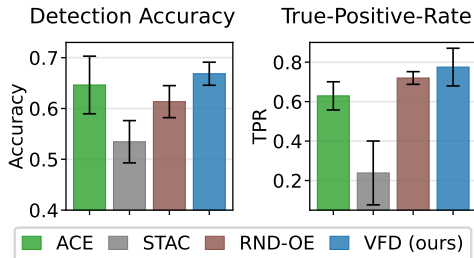


Figure 5: VFD epistemic uncertainty can detect failures during deployment.

7 Conclusion

We present VFD, a mathematically grounded and computationally tractable method for estimating epistemic uncertainty in flow-based VLAs. On that basis, we propose SAVE, a framework for multitask active fine-tuning of VLAs that allocates the demonstration budget to tasks and initial observations that require it most. Our experiments show that VFD is better-calibrated than baselines, and that using it for active fine-tuning with SAVE reduces data collection by at least 22% compared to uncertainty estimation baselines. These results, along with the superior failure-detection performance, demonstrate the capabilities of our methods to improve the reliability and adaptability of VLAs.

Limitations. While we have shown that VFD requires only a two-member ensemble to be well-calibrated, the reliance on training and maintaining a separate set of model weights is a limitation of our method. We use the uncertainty for the initial observation as a proxy for task difficulty. Our experiments empirically confirm this correlation, but there may be certain scenarios that “look simple” but are challenging for the policy (e.g., due to complex contact dynamics). Finally, treating uncertainty as the primary signal for determining which data to collect does not account for informational dependencies across tasks. Developing tractable methods to quantify these dependencies and maximizing expected information gain are interesting avenues for future work.

Acknowledgements

This work was supported by the German Federal Ministry of Research, Technology and Space (BMFTR) under the Robotics Institute Germany (RIG) funded by BMFTR grant 16ME0997K, the German Research Foundation (DFG) within the RTG project ConVeY funded by grant GRK 2428, the Humboldt Professorship for Robotics and Artificial Intelligence and the Swiss National Science Foundation under NCCR Automation, grant agreement 51NF40 180545. Marco Bagatella is supported by the Max Planck ETH Center for Learning Systems.

References

- [1] M. Abdar et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 76:243–297, 2021.
- [2] C. Agia, R. Sinha, J. Yang, Z. Cao, R. Antonova, M. Pavone, and J. Bohg. Unpacking failure modes of generative policies: Runtime monitoring of consistency and progress. In *Conference on Robot Learning (CoRL)*, 2025.
- [3] AgiBot-World et al. AgiBot World Colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. In *International Conference on Intelligent Robots and Systems (IROS)*, 2025.
- [4] A. N. Angelopoulos and S. Bates. Conformal prediction: A gentle introduction. *Foundations and Trends in Machine Learning*, 16(4):494–591, 2023.
- [5] J. T. Ash, C. Zhang, A. Krishnamurthy, J. Langford, and A. Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations (ICLR)*, 2020.
- [6] M. Awais, M. Naseer, S. Khan, R. M. Anwer, H. Cholakkal, M. Shah, M.-H. Yang, and F. S. Khan. Foundation models defining a new era in vision: a survey and outlook. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(4):2245–2264, 2025.
- [7] M. S. Ayhan and P. Berens. Test-time data augmentation for estimation of heteroscedastic aleatoric uncertainty in deep neural networks. In *Medical Imaging with Deep Learning*, 2018.
- [8] M. Bagatella, J. Hübotter, G. Martius, and A. Krause. Active fine-tuning of multi-task policies. In *International Conference on Machine Learning (ICML)*, 2025.
- [9] L. Berry, A. Brando, and D. Meger. Shedding light on large generative networks: Estimating epistemic uncertainty in diffusion models. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2024.
- [10] K. Black et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In *Conference on Robot Learning (CoRL)*, 2025.
- [11] R. Cadene, S. Alibert, F. Capuano, M. Aractingi, A. Zouitine, P. Kooijmans, J. Choghari, M. Russi, C. Pascal, S. Palma, et al. LeRobot: An open-source library for end-to-end robot learning. In *International Conference on Learning Representations (ICLR)*, 2026.
- [12] K. Chaloner and I. Verdinelli. Bayesian experimental design: A review. *Statistical science*, 10: 273–304, 1995.
- [13] M. Chan, M. Molina, and C. Metzler. Estimating epistemic and aleatoric uncertainty with a single model. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.
- [14] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *Robotics: Science and Systems (RSS)*, 2023.
- [15] Y. Cui, D. Isele, S. Niekum, and K. Fujimura. Uncertainty-aware data aggregation for deep imitation learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

- [16] E. Daxberger, A. Kristiadi, A. Immer, R. Eschenhagen, M. Bauer, and P. Hennig. Laplace redux - effortless Bayesian deep learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.
- [17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019.
- [18] J. Diquigiovanni, M. Fontana, S. Vantini, et al. The importance of being a band: Finite-sample exact distribution-free prediction sets for functional data. *STATISTICA SINICA*, 1:1–41, 2024.
- [19] S. Dohare, J. F. Hernandez-Garcia, Q. Lan, P. Rahman, A. R. Mahmood, and R. S. Sutton. Loss of plasticity in deep continual learning. *Nature*, 632:768–774, 2024.
- [20] A. Dosovitskiy et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.
- [21] B. Efron. Tweedie’s formula and selection bias. *Journal of the American Statistical Association*, 106:1602–1614, 2011.
- [22] P. Esser, S. Kulal, A. Blattmann, R. Entezari, J. Müller, H. Saini, Y. Levi, D. Lorenz, A. Sauer, F. Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning (ICML)*, 2024.
- [23] E. Fadeeva et al. Fact-checking the output of large language models via token-level uncertainty quantification. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 9367–9385, 2024.
- [24] G. Franchi, N. Belkhir, D. N. Trong, G. Xia, and A. Pilzer. Towards understanding and quantifying uncertainty for text-to-image generation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.
- [25] Y. Gal and Z. Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning (ICML)*, 2016.
- [26] Y. Gal, R. Islam, and Z. Ghahramani. Deep Bayesian active learning with image data. In *International Conference on Machine Learning (ICML)*, 2017.
- [27] Q. Gu, Y. Ju, S. Sun, I. Gilitschenski, H. Nishimura, M. Itkina, and F. Shkurti. SAFE: Multi-task failure detection for vision-language-action models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.
- [28] Z. He, Y. Cao, and M. Ciocarlie. Uncertainty comes for free: Human-in-the-loop policies with diffusion models. *arXiv preprint arXiv:2503.01876*, 2025.
- [29] J. Hejna, S. Mirchandani, A. Balakrishna, A. Xie, A. Wahid, J. Tompson, P. Sanketi, D. Shah, C. Devin, and D. Sadigh. Robot data curation with mutual information estimators. *Robotics: Science and Systems (RSS)*, 2025.
- [30] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:6840–6851, 2020.
- [31] D. Holzmüller, V. Zaverkin, J. Kästner, and I. Steinwart. A framework and benchmark for deep batch active learning for regression. *Journal of Machine Learning Research (JMLR)*, 24(164): 1–81, 2023.
- [32] E. Hüllermeier and W. Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110:457–506, 2021.
- [33] M. Jazbec, E. Wong-Toi, G. Xia, D. Zhang, E. Nalisnick, and S. Mandt. Generative uncertainty in diffusion models. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2025.
- [34] L. Ju, M. Nautiyal, A. Hellander, E. Vats, and P. Singh. Epistemic uncertainty quantification for pre-trained VLMs via Riemannian flow matching. *arXiv preprint arXiv:2601.21662*, 2026.

- [35] K. Judah, A. Fern, and T. G. Dietterich. Active imitation learning via reduction to IID active learning. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2012.
- [36] R. Karczewski, M. Heinonen, and V. Garg. Diffusion models as cartoonists: The curious case of high density regions. In *International Conference on Learning Representations (ICLR)*, 2025.
- [37] U. B. Karli, T. Kurumisawa, and T. Fitzgerald. Ask before you act: Token-level uncertainty for intervention in vision-language-action models. In *Second Workshop on Out-of-Distribution Generalization in Robotics at RSS*, 2025.
- [38] A. Khazatsky et al. DROID: A large-scale in-the-wild robot manipulation dataset. In *Robotics: Science and Systems*, 2024.
- [39] M. J. Kim et al. OpenVLA: An open-source vision-language-action model. In *Conference on Robot Learning (CoRL)*, 2024.
- [40] A. Kirsch, J. van Amersfoort, and Y. Gal. BatchBALD: Efficient and diverse batch acquisition for deep Bayesian active learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
- [41] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- [42] S.-W. Lee, X. Kang, and Y.-L. Kuo. Diff-DAGger: Uncertainty estimation with diffusion policy for robotic manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025.
- [43] Q. Li, B. Yin, W. Huang, R. Liu, B. Zou, R. Yu, J. Ye, W. Yu, and X. Wang. Vision-language-action safety: Threats, challenges, evaluations, and mechanisms. *arXiv preprint arXiv:2604.23775*, 2026.
- [44] C. Ling et al. Uncertainty quantification for in-context learning of large language models. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2024.
- [45] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.
- [46] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.
- [47] A. Loquercio, M. Segu, and D. Scaramuzza. A general framework for uncertainty estimation in deep learning. *IEEE Robotics and Automation Letters*, 2020.
- [48] H. Ma, J. Chen, J. T. Zhou, G. Wang, and C. Zhang. Estimating LLM uncertainty with evidence. *arXiv preprint arXiv:2502.00290*, 2025.
- [49] A. Malinin and M. Gales. Uncertainty estimation in autoregressive structured prediction. In *International Conference on Learning Representations (ICLR)*, 2021.
- [50] Z. Mei, T. Yin, M. Baker, O. Shorinwa, and A. Majumdar. World models that know when they don't know: Controllable video generation with calibrated uncertainty. *arXiv preprint arXiv:2512.05927*, 2025.
- [51] E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, and B. Lakshminarayanan. Do deep generative models know what they don't know? In *International Conference on Learning Representations (ICLR)*, 2019.
- [52] NVIDIA et al. GR00T N1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

- [53] A. O’Neill et al. Open X-embodiment: Robotic learning datasets and RT-X models: Open X-embodiment collaboration. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.
- [54] Physical Intelligence et al. π_0 : A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.
- [55] A. Z. Ren et al. Robots that ask for help: Uncertainty alignment for large language model planners. In *Conference on Robot Learning (CoRL)*, 2023.
- [56] J. Ren, J. Luo, Y. Zhao, K. Krishna, M. Saleh, B. Lakshminarayanan, and P. J. Liu. Out-of-distribution detection and selective generation for conditional language models. In *International Conference on Learning Representations (ICLR)*, 2023.
- [57] M. Reuss, H. Zhou, M. Rühle, Ö. E. Yağmurlu, F. Otto, and R. Lioutikov. FLOWER: Democratizing generalist robot policies with efficient vision-language-flow policies. In *Conference on Robot Learning (CoRL)*, 2025.
- [58] R. Römer, A. Kobras, L. Worbis, and A. P. Schoellig. Failure prediction at runtime for generative robot policies. *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.
- [59] R. Römer, J. Balletshofer, J. Thumm, M. Pavone, A. P. Schoellig, and M. Althoff. From demonstrations to safe deployment: Path-consistent safety filtering for diffusion policies. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2026.
- [60] R. Römer, Y. Zhang, and A. P. Schoellig. CLARE: Continual learning for vision-language-action models via autonomous adapter routing and expansion. *arXiv preprint arXiv:2601.09512*, 2026.
- [61] O. Sener and S. Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations (ICLR)*, 2018.
- [62] B. Settles. Active learning literature survey. Technical report, University of Wisconsin–Madison, 2009.
- [63] O. Shorinwa, Z. Mei, J. Lidard, A. Z. Ren, and A. Majumdar. A survey on uncertainty quantification of large language models: Taxonomy, open research challenges, and future directions. *ACM Computing Surveys*, 58:1–38, 2025.
- [64] M. Shukor et al. SmolVLA: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.
- [65] C. Xu, T. Khuong Nguyen, E. Dixon, C. Rodriguez, P. Miller, R. Lee, P. Shah, R. Ambrus, H. Nishimura, and M. Itkina. Can we detect failures without failure data? Uncertainty-aware runtime failure detection for imitation learning policies. In *Robotics: Science and Systems (RSS)*, 2025.
- [66] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer. Sigmoid loss for language image pre-training. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11975–11986, 2023.
- [67] B. Zitkovich et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning (CoRL)*, 2023.

Appendix

Table of Contents

A Theoretical Results	16
A.1 Flow Matching Fundamentals	16
A.2 Proof of Theorem 4.1	16
B Experimental Details	18
B.1 Computational Resources	18
B.2 Environments	18
B.3 Pre-Training	18
B.4 Uncertainty Quantification	18
B.5 Calibration Experiments	21
B.6 Active Fine-Tuning	23
B.7 Failure Detection	23
C Additional Results	23
C.1 Toy Example	23
C.2 Additional Results for Section 6.3	24
C.3 Ensembling vs. Laplace Approximation	24
C.4 Iterative Fine-Tuning vs. Retraining	26
C.5 Effect of τ on SAVE	26
C.6 Detailed Results on Failure Detection	26
D Broader Impact	28

A Theoretical Results

A.1 Flow Matching Fundamentals

Flow matching [45] formulates generative modeling as learning a time-dependent vector field \mathbf{u}_s that transports samples from a simple base distribution p_0 to the data distribution $p_1 = q$. We consider a noise distribution $p_0(\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0 | \mathbf{0}, \mathbf{I})$ and a data distribution $\mathbf{x}_1 \sim q$. The OT Gaussian conditional probability path is defined as

$$p_s(\mathbf{x} | \mathbf{x}_1) = \mathcal{N}(\mathbf{x} | s\mathbf{x}_1, (1-s)^2\mathbf{I}), \quad (12)$$

and the corresponding flow that pushes the noise distribution p_0 to $p_s(\mathbf{x} | \mathbf{x}_1)$ is given by the affine map

$$\phi_s(\mathbf{x}_0) = (1-s)\mathbf{x}_0 + s\mathbf{x}_1. \quad (13)$$

The conditional velocity field $\mathbf{u}_s(\mathbf{x} | \mathbf{x}_1)$ defining this flow via the ODE $\frac{d}{ds}\phi_s(\mathbf{x}_0) = \mathbf{u}_s(\phi_s(\mathbf{x}_0))$, and generating $p_s(\mathbf{x} | \mathbf{x}_1)$, can be derived as:

$$\mathbf{u}_s(\mathbf{x} | \mathbf{x}_1) = \frac{d}{ds}\phi_s(\mathbf{x}_0) = \mathbf{x}_1 - \mathbf{x}_0. \quad (14)$$

Marginalizing over the data distribution $q(\mathbf{x}_1)$ yields the marginal probability path

$$p_s(\mathbf{x}) = \int p_s(\mathbf{x} | \mathbf{x}_1)q(\mathbf{x}_1)d\mathbf{x}_1, \quad (15)$$

and the marginal velocity field is defined as

$$\mathbf{u}_s(\mathbf{x}) = \int \mathbf{u}_s(\mathbf{x} | \mathbf{x}_1) \frac{p_s(\mathbf{x} | \mathbf{x}_1)q(\mathbf{x}_1)}{p_s(\mathbf{x})} d\mathbf{x}_1. \quad (16)$$

Finally, these components must satisfy the continuity equation

$$\frac{\partial}{\partial s}p_s(\mathbf{x}) + \text{div}(p_s(\mathbf{x})\mathbf{u}_s(\mathbf{x})) = 0, \quad (17)$$

which ensures probability mass conservation.

A.2 Proof of Theorem 4.1

To simplify notation, we drop the fixed conditioning input \mathbf{y} and the parameterization of the velocity fields as neural networks in the following derivations, considering two terminal distributions $p^1(\mathbf{x}) = p_1^1(\mathbf{x})$, $p^2(\mathbf{x}) = p_1^2(\mathbf{x})$ induced by velocity fields $\mathbf{u}_s^1(\mathbf{x})$, $\mathbf{u}_s^2(\mathbf{x})$ and similar base distributions $p_0^1(\mathbf{x}) = p_0^2(\mathbf{x}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$. In the proof of Theorem 4.1, we will use the following lemma.

Lemma A.1. *Let $\mathbf{u}_s(\mathbf{x})$ be the marginal velocity field of a flow matching model trained with Gaussian OT conditional probability paths, and let $p_s(\mathbf{x})$ be the marginal probability path induced by $\mathbf{u}_s(\mathbf{x})$. Then, for all $s \in [0, 1)$,*

$$\nabla \log p_s(\mathbf{x}) = \frac{s\mathbf{u}_s(\mathbf{x}) - \mathbf{x}}{1-s}. \quad (18)$$

Proof. Applying Tweedie's formula [21] to (12) and (15) yields

$$\mathbb{E}[s\mathbf{x}_1 | \mathbf{x}_s = \mathbf{x}] = \mathbf{x} + (1-s)^2 \frac{\nabla p_s(\mathbf{x})}{p_s(\mathbf{x})} \quad (19)$$

$$\Leftrightarrow \nabla \log p_s(\mathbf{x}) = \frac{s\mathbb{E}[\mathbf{x}_1 | \mathbf{x}_s = \mathbf{x}] - \mathbf{x}}{(1-s)^2}. \quad (20)$$

Plugging the conditional velocity (14) into the marginal velocity (16) gives

$$\mathbf{u}_s(\mathbf{x}) = \int (\mathbf{x}_1 - \mathbf{x}_0)p_s(\mathbf{x}_1 | \mathbf{x}_s = \mathbf{x})d\mathbf{x}_1 \quad (21)$$

$$= \mathbb{E}[\mathbf{x}_1 - \mathbf{x}_0 | \mathbf{x}_s = \mathbf{x}] \quad (22)$$

$$= \frac{\mathbb{E}[\mathbf{x}_1 | \mathbf{x}_s = \mathbf{x}] - \mathbf{x}}{1-s} \quad (23)$$

$$\Leftrightarrow \mathbb{E}[\mathbf{x}_1 | \mathbf{x}_s = \mathbf{x}] = \mathbf{x} + (1-s)\mathbf{u}_s(\mathbf{x}), \quad (24)$$

where (23) follows from $\mathbf{x} = (1-s)\mathbf{x}_0 + s\mathbf{x}_1 \Leftrightarrow \mathbf{x}_1 - \mathbf{x}_0 = \frac{\mathbf{x}_1 - \mathbf{x}}{1-s}$ and taking the conditional expectation. Finally, plugging (23) into (20) yields

$$\nabla \log p_s(\mathbf{x}) = \frac{s\mathbf{x} + s(1-s)\mathbf{u}_s(\mathbf{x}) - \mathbf{x}}{(1-s)^2} \quad (25)$$

$$= \frac{s\mathbf{u}_s(\mathbf{x}) - \mathbf{x}}{1-s}, \quad (26)$$

which concludes the proof. \square

Proof of Theorem 4.1. Recall that we aim to express the KL divergence between two distributions $p^1(\mathbf{x})$ and $p^2(\mathbf{x})$ parameterized as flow matching models in terms of their time-dependent velocity fields $\mathbf{u}_s^1(\mathbf{x})$ and $\mathbf{u}_s^2(\mathbf{x})$. We write

$$D_{\text{KL}}(p^1(\mathbf{x}) \| p^2(\mathbf{x})) = D_{\text{KL}}(p^1(\mathbf{x}) \| p^2(\mathbf{x})) - D_{\text{KL}}(p_0^1(\mathbf{x}) \| p_0^2(\mathbf{x})) + D_{\text{KL}}(p_0^1(\mathbf{x}) \| p_0^2(\mathbf{x})) \quad (27)$$

$$= \int_0^1 \frac{\partial}{\partial s} D_{\text{KL}}(p_s^1(\mathbf{x}) \| p_s^2(\mathbf{x})) ds, \quad (28)$$

where the last step follows from the fact that $p_0^1 = p_0^2$ since both models use the same source distribution. Applying the product rule, we can reformulate the term in the integral as

$$\frac{\partial}{\partial s} D_{\text{KL}}(p_s^1(\mathbf{x}) \| p_s^2(\mathbf{x})) = \frac{\partial}{\partial s} \int p_s^1(\mathbf{x}) \log \frac{p_s^1(\mathbf{x})}{p_s^2(\mathbf{x})} d\mathbf{x} \quad (29)$$

$$= \int \frac{\partial}{\partial s} p_s^1(\mathbf{x}) \log \frac{p_s^1(\mathbf{x})}{p_s^2(\mathbf{x})} d\mathbf{x} + \int p_s^1(\mathbf{x}) \frac{\partial}{\partial s} \log \frac{p_s^1(\mathbf{x})}{p_s^2(\mathbf{x})} d\mathbf{x} \quad (30)$$

$$= \int \frac{\partial}{\partial s} p_s^1(\mathbf{x}) \log \frac{p_s^1(\mathbf{x})}{p_s^2(\mathbf{x})} d\mathbf{x} + \int p_s^1(\mathbf{x}) \left(\frac{\frac{\partial}{\partial s} p_s^1(\mathbf{x})}{p_s^1(\mathbf{x})} - \frac{\frac{\partial}{\partial s} p_s^2(\mathbf{x})}{p_s^2(\mathbf{x})} \right) d\mathbf{x} \quad (31)$$

$$= \int \frac{\partial}{\partial s} p_s^1(\mathbf{x}) \log \frac{p_s^1(\mathbf{x})}{p_s^2(\mathbf{x})} d\mathbf{x} + \underbrace{\int \frac{\partial}{\partial s} p_s^1(\mathbf{x}) d\mathbf{x}}_{=0} - \int \frac{p_s^1(\mathbf{x})}{p_s^2(\mathbf{x})} \frac{\partial}{\partial s} p_s^2(\mathbf{x}) d\mathbf{x}. \quad (32)$$

where the second term vanishes since $\int \frac{\partial}{\partial s} p_s^1(\mathbf{x}) d\mathbf{x} = \frac{\partial}{\partial s} \int p_s^1(\mathbf{x}) d\mathbf{x} = \frac{\partial}{\partial s} 1 = 0$. To reformulate the other terms in (32), we proceed as

$$\frac{\partial}{\partial s} D_{\text{KL}}(p_s^1(\mathbf{x}) \| p_s^2(\mathbf{x})) = \int \frac{\partial}{\partial s} p_s^1(\mathbf{x}) \log \frac{p_s^1(\mathbf{x})}{p_s^2(\mathbf{x})} d\mathbf{x} - \int \frac{p_s^1(\mathbf{x})}{p_s^2(\mathbf{x})} \frac{\partial}{\partial s} p_s^2(\mathbf{x}) d\mathbf{x} \quad (33)$$

$$= - \int \text{div}(p_s^1(\mathbf{x}) \mathbf{u}_s^1(\mathbf{x})) \log \frac{p_s^1(\mathbf{x})}{p_s^2(\mathbf{x})} d\mathbf{x} + \int \frac{p_s^1(\mathbf{x})}{p_s^2(\mathbf{x})} \text{div}(p_s^2(\mathbf{x}) \mathbf{u}_s^2(\mathbf{x})) d\mathbf{x} \quad (34)$$

$$= \int p_s^1(\mathbf{x}) \mathbf{u}_s^{1,\top}(\mathbf{x}) \nabla \log \frac{p_s^1(\mathbf{x})}{p_s^2(\mathbf{x})} d\mathbf{x} - \int p_s^1(\mathbf{x}) \mathbf{u}_s^{2,\top}(\mathbf{x}) \nabla \log \frac{p_s^1(\mathbf{x})}{p_s^2(\mathbf{x})} d\mathbf{x} \quad (35)$$

$$= \int p_s^1(\mathbf{x}) (\mathbf{u}_s^1(\mathbf{x}) - \mathbf{u}_s^2(\mathbf{x}))^\top \nabla \log \frac{p_s^1(\mathbf{x})}{p_s^2(\mathbf{x})} d\mathbf{x} \quad (36)$$

$$= \int p_s^1(\mathbf{x}) (\mathbf{u}_s^1(\mathbf{x}) - \mathbf{u}_s^2(\mathbf{x}))^\top (\nabla \log p_s^1(\mathbf{x}) - \nabla \log p_s^2(\mathbf{x})) d\mathbf{x}, \quad (37)$$

where (34) follows from the continuity equation (17), and (35) follows from the divergence theorem and the assumption that $p_s^m(\mathbf{x}) \mathbf{u}_s^m(\mathbf{x}) \rightarrow \mathbf{0}$ as $\|\mathbf{x}\| \rightarrow \infty$. Applying Lemma A.1 to the last term in the integral (37) yields

$$\nabla \log p_s^1(\mathbf{x}) - \nabla \log p_s^2(\mathbf{x}) = \frac{s}{1-s} (\mathbf{u}_s^1(\mathbf{x}) - \mathbf{u}_s^2(\mathbf{x})). \quad (38)$$

Defining $\kappa_s = \frac{s}{1-s}$ and plugging (38) into (37) results in

$$\frac{\partial}{\partial s} D_{\text{KL}}(p_s^1(\mathbf{x}) \| p_s^2(\mathbf{x})) = \int p_s^1(\mathbf{x}) (\mathbf{u}_s^1(\mathbf{x}) - \mathbf{u}_s^2(\mathbf{x}))^\top (\nabla \log p_s^1(\mathbf{x}) - \nabla \log p_s^2(\mathbf{x})) d\mathbf{x} \quad (39)$$

$$= \int p_s^1(\mathbf{x}) \kappa_s (\mathbf{u}_s^1(\mathbf{x}) - \mathbf{u}_s^2(\mathbf{x}))^\top (\mathbf{u}_s^1(\mathbf{x}) - \mathbf{u}_s^2(\mathbf{x})) d\mathbf{x} \quad (40)$$

$$= \kappa_s \mathbb{E}_{\mathbf{x} \sim p_s^1(\cdot)} [\|\mathbf{u}_s^1(\mathbf{x}) - \mathbf{u}_s^2(\mathbf{x})\|_2^2]. \quad (41)$$

Finally, we can insert (41) into (28) and obtain

$$D_{\text{KL}}(p^1(\mathbf{x}) \| p^2(\mathbf{x})) = \int_0^1 \kappa_s \mathbb{E}_{\mathbf{x} \sim p_s^1(\cdot)} [\|\mathbf{u}_s^1(\mathbf{x}) - \mathbf{u}_s^2(\mathbf{x})\|_2^2] ds, \quad (42)$$

concluding the proof. \square

Remark A.2. Although the weighting κ_s in (42) diverges as $s \rightarrow 1$, the VFD uncertainty estimate remains finite. In practice, our estimator (7) evaluates the velocity difference on a grid $s_\ell = \ell \delta s$ with $\ell \in \{0, \dots, N_s - 1\}$, so the largest weight is $\kappa_{1-\delta s} = \frac{1-\delta s}{\delta s}$, which is finite for $\delta s > 0$.

B Experimental Details

B.1 Computational Resources

All experiments¹ are run on a compute node with two NVIDIA RTX 4090 GPUs, 8 CPU cores, and 64 GB of RAM. With this hardware setup, one active learning experiment with 15 rounds of uncertainty quantification, episode selection, 4,000 gradient steps, and 30 evaluation rollouts per task takes approximately 12.5 hours. We run 25 different configurations for active learning, ablating temperature and uncertainty estimates, over three seeds, resulting in 75 total active learning experiment runs. This results in approximately 940 compute hours, i.e., with two GPUs in parallel, approximately 1,880 GPU hours. We estimate that about the same computing budget was used for developing the codebase and preliminary experiments.

B.2 Environments

We use the LIBERO simulation benchmark [46] (MIT License) in our experiments. The environments include a Franka robotic manipulator with a parallel-yaw gripper in a kitchen environment. For each task, up to 50 human expert demonstrations are available. The observations \mathbf{o}_t consist of two RGB images with spatial resolution 256×256 , one from a fixed workspace view and one from a camera mounted on the robot’s end-effector. Additionally, observations include the 8-dimensional proprioceptive state of the arm, comprising the end-effector position (3D), the end-effector orientation parameterized as an axis-angle (3D), and the gripper joint positions (2D). Actions \mathbf{a}_t are continuous and 7-dimensional: 3D delta end-effector translation, 3D delta end-effector orientation, and 1D gripper command.

B.3 Pre-Training

We pre-train a SmolVLA [64] (Apache License) model for 30,000 steps with a batch size of 32 on LIBERO-Spatial, LIBERO-Object, and LIBERO-Goal (10 tasks each), as well as tasks (0, 1, 2) from LIBERO-10 (also referred to as LIBERO-Long).

B.4 Uncertainty Quantification

The computation of VFD-based epistemic uncertainty estimation is summarized in Algorithm 2 and illustrated in Figure 6.

Action-L2 Implementation. Action-L2 estimates epistemic uncertainty from the disagreement between the sampler policy and an ensemble of M terminal action distributions. Given an observation

¹Our codebase builds on LeRobot [11], released under an Apache License.

Algorithm 2 Efficient Epistemic Uncertainty Estimation for Flow-Matching Models using Velocity Field Disagreement (VFD).

- 1: **Input:** Flow-matching model ensemble $p^{\theta_1}(\mathbf{x} | \mathbf{y}), p^{\theta_2}(\mathbf{x} | \mathbf{y})$ with vector fields $\mathcal{V} = (\mathbf{v}_s^{\theta_1}, \mathbf{v}_s^{\theta_2})$ trained via OT paths, conditioning input \mathbf{y} , ODE integration step size $\delta s = 1/N_s$.
 - 2: **Output:** Estimate of the epistemic uncertainty $u_e(\mathbf{y}; \mathcal{V})$ at the conditioning input \mathbf{y} .
 - 3: **Hyperparameters:** Batch size B .
 - 4: Sample $\mathbf{x}_0^{(1,1:B)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \mathbf{x}_0^{(2,1:B)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.
 - 5: Initialize $u_e \leftarrow 0$.
 - 6: **for** $\ell = 0$ **to** $N_s - 1$ **do** \triangleright ODE integration.
 - 7: Set $s = \ell \delta s$.
 - 8: **for each** $i \in \{1, 2\}$ **do**
 - 9: Compute the expected VFDs $d_{\ell,i}^2 = \frac{1}{B} \sum_{b=1}^B \|\mathbf{v}_s^{\theta_1}(\mathbf{x}_s^{(i,b)}, \mathbf{y}) - \mathbf{v}_s^{\theta_2}(\mathbf{x}_s^{(i,b)}, \mathbf{y})\|_2^2$.
 - 10: Update uncertainty estimate $u_e \leftarrow u_e + \frac{1}{2N_s} \frac{s}{1-s} d_{\ell,i}^2$.
 - 11: Forward integrate $\mathbf{x}_{s+\delta s}^{(i,1:B)} = \mathbf{x}_s^{(i,1:B)} + \mathbf{v}_s^{\theta_i}(\mathbf{x}_s^{(i,1:B)}, \mathbf{y}) \delta s$.
 - 12: **return** $u_e(\mathbf{y}; \mathcal{V}) = u_e$.
-

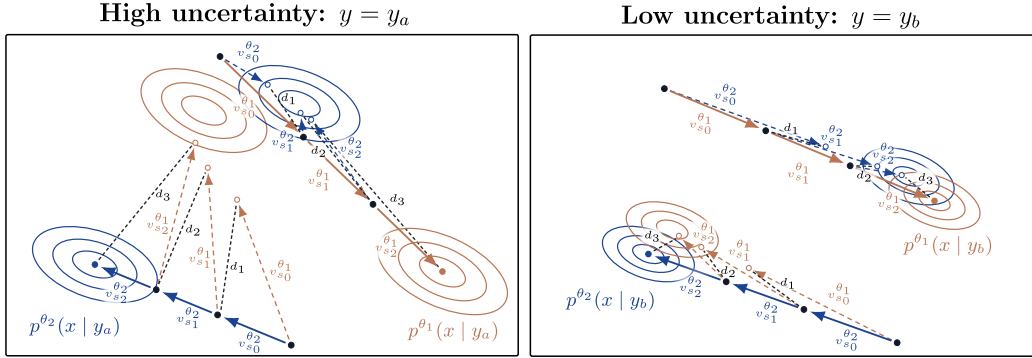


Figure 6: Visualization of the velocity-field disagreement (VFD) computation for two conditioning inputs with different levels of epistemic uncertainty.

\mathbf{o} , we draw independent Gaussian noise samples from p_0 and generate C terminal action chunks from the sampler policy and from each ensemble member. Let

$$\{\mathbf{A}_i^s(\mathbf{o})\}_{i=1}^C$$

denote the action chunks generated by the sampler policy, and let

$$\{\mathbf{A}_j^{(m)}(\mathbf{o})\}_{j=1}^C$$

denote the action chunks generated by ensemble member $m \in \{1, \dots, M\}$.

The discrepancy between the sampler policy and ensemble member m is computed as the mean pairwise Euclidean distance between their sampled terminal action chunks:

$$d_m(\mathbf{o}) = \frac{1}{C^2} \sum_{i=1}^C \sum_{j=1}^C \left\| \mathbf{A}_i^s(\mathbf{o}) - \mathbf{A}_j^{(m)}(\mathbf{o}) \right\|_2.$$

The Action-L2 epistemic uncertainty estimate is then the average discrepancy across ensemble members:

$$u_{\text{Action-L2}}(\mathbf{o}) = \frac{1}{M} \sum_{m=1}^M d_m(\mathbf{o}).$$

ACE Implementation. ACE [58] estimates epistemic uncertainty from the dispersion of action samples generated by a single policy. Given an observation \mathbf{o} , we draw C independent Gaussian

noise samples and generate C action chunks via flow-matching ODE integration. For each sampled chunk $c \in \{1, \dots, C\}$, we extract the first three action dimensions at each timestep k , corresponding to end-effector position deltas $\Delta \mathbf{p}_k^{(c)} \in \mathbb{R}^3$.

Starting from the current end-effector position \mathbf{p}_{curr} , these deltas are integrated to obtain absolute position trajectories:

$$\mathbf{p}_t^{(c)} = \mathbf{p}_{\text{curr}} + \sum_{k=1}^t \Delta \mathbf{p}_k^{(c)}, \quad t \in \{1, \dots, H\}.$$

At each timestep t in the prediction horizon H , the sampled positions $\{\mathbf{p}_t^{(c)}\}_{c=1}^C$ are discretized into a 3D grid. The grid cell size along each coordinate axis is set to 0.03 times the empirical range of the samples along that axis. Let \mathcal{B}_t denote a set of bins over the occupied grid at time t and let $P_t(b)$ be the empirical probability of a sample falling into bin $b \in \mathcal{B}_t$ at time t . The ACE uncertainty estimate is the Shannon entropy of these bin counts, averaged over the prediction horizon:

$$u_{\text{ACE}}(\mathbf{o}) = -\frac{1}{H} \sum_{t=1}^H \sum_{b \in \mathcal{B}_t} P_t(b) \log_2 P_t(b).$$

DECU Implementation. DECU [9] estimates epistemic uncertainty in diffusion models from the disagreement among ensemble members’ denoising mean at an intermediate point of the generative process. Given the close relationship between diffusion and flow matching, we adapt DECU to flow matching by replacing the denoising mean with the learned velocity field. Given an observation \mathbf{o} , we sample C independent Gaussian noise vectors $\mathbf{x}_0^{(c)} \sim p_0$ for $c \in \{1, \dots, C\}$. Each ensemble member integrates the flow-matching ODE from $s = 0$ to a branching time $s_b = 0.995$ to obtain intermediate states $\mathbf{x}_{s_b}^{(c)}$. At s_b , we evaluate the velocity fields of all M ensemble members, $\mathbf{v}_i^{(c)} = \mathbf{v}_{s_b}^{\theta_i}(\mathbf{x}_{s_b}^{(c)}, \mathbf{o})$ for $i \in \{1, \dots, M\}$.

Disagreement is measured using the Pairwise-Distance Estimator (PaiDE). For each sampled chunk c , define the squared pairwise velocity-field distances

$$D_{ij}^{(c)} = \left\| \mathbf{v}_i^{(c)} - \mathbf{v}_j^{(c)} \right\|_2^2, \quad i, j \in \{1, \dots, M\}.$$

The score for sample c is then

$$d_{\text{DECU}}^{(c)}(\mathbf{o}) = -\frac{1}{M} \sum_{i=1}^M \log \left(\frac{1}{M} \sum_{j=1}^M \exp(-D_{ij}^{(c)}) \right).$$

The DECU epistemic uncertainty estimate for observation \mathbf{o} is the mean score over the C sampled chunks:

$$u_{\text{DECU}}(\mathbf{o}) = \frac{1}{C} \sum_{c=1}^C d_{\text{DECU}}^{(c)}(\mathbf{o}).$$

Generative Uncertainty (GU) Implementation. As described in the main text, GU [33] approximates the posterior predictive action distribution as a Gaussian. Using our deep ensemble, we compute a raw uncertainty score for each initial observation \mathbf{o}_{kl} , where k indexes the task and l indexes the observation within that task. For a fixed initial noise sample, let $\mathbf{A}^{(i)}(\mathbf{o}_{kl}) \in \mathbb{R}^{H \times D_a}$ denote the action trajectory generated by ensemble member $i \in \{1, \dots, M\}$, and let $a_{kl,t,d}^{(i)}$ denote its action value at timestep $t \in \{1, \dots, H\}$ and action dimension $d \in \{1, \dots, D_a\}$.

The raw GU uncertainty score is obtained by summing the log-variance of the ensemble predictions across all timesteps and action dimensions:

$$u_{kl}^{\text{raw}} = \sum_{t=1}^H \sum_{d=1}^{D_a} \log \left(\text{Var}_{i=1}^M \left[a_{kl,t,d}^{(i)} \right] + \epsilon_{\text{var}} \right),$$

where $\epsilon_{\text{var}} = 10^{-8}$ is added for numerical stability.

Because the differential entropy of a continuous distribution can be negative, these raw scores may also be negative. To obtain non-negative uncertainty weights for use in (9), we first aggregate the observation-level scores into a raw task-level uncertainty score U_k^{raw} . We then shift the task-level scores by

$$U_k = U_k^{\text{raw}} - \min_{k' \in \{1, \dots, K\}} U_{k'}^{\text{raw}} + \epsilon_{\text{off}} \left(\max_{k' \in \{1, \dots, K\}} U_{k'}^{\text{raw}} - \min_{k' \in \{1, \dots, K\}} U_{k'}^{\text{raw}} \right).$$

Here, $\epsilon_{\text{off}} > 0$ is a small offset coefficient. This shift removes negative values and when the task scores are not all identical, the range-dependent offset moves the minimum-uncertainty task away from zero so that it retains non-zero sampling probability after normalization.

Entropy Implementation. Entropy [49] utilizes the extent to which the VLM (SmolVLA backbone) successfully predicted the next token of prefix. Given an observation \mathbf{o} , the policy constructs a prefix sequence of length S_{prefix} containing images, robot states and language. The VLM predicts the next-token probability distribution ψ_m at each valid prefix position $m \in \{1, \dots, S_{\text{prefix}}\}$, with a single forward pass. The uncertainty score is calculated by Shannon entropy of this distribution, averaged over all valid prefix tokens:

$$u_{\text{Entropy}} = -\frac{1}{S_{\text{prefix}}} \sum_{m=1}^{S_{\text{prefix}}} \sum_{z \in \mathcal{Z}} \psi_m(z) \log \psi_m(z),$$

where \mathcal{Z} is the VLM vocabulary and $\psi_m(z)$ is the predicted probability of token z at position m .

Perplexity Implementation. Perplexity [23] estimates uncertainty by measuring the likelihood of the language instruction under the VLM backbone. Similar to Entropy, it evaluates the prefix in a single forward pass without action generation. We extract the predicted next-token log-probabilities for the S_{lang} language tokens within their respective prefix. Let z_m be the true language token at sequence position m . The uncertainty score u is the exponentiated negative mean log-probability of these tokens:

$$u_{\text{Perp}} = \exp \left(-\frac{1}{S_{\text{lang}}} \sum_{m=1}^{S_{\text{lang}}} \log \psi_m(z_m) \right)$$

Diversity-Based Greedy. The diversity-based greedy baseline [61] selects initial observations to maximize visual diversity. Let \mathcal{O}_k denote the pool of candidate initial observations, and $\hat{\mathcal{O}}$ denote the set of selected observations. We extract visual features for each $\mathbf{o}_i \in \mathcal{O}_k$ using a SigLIP [66] vision encoder². We rank all candidates using the k -Center Greedy algorithm. At each step, we select \mathbf{o}^* that maximizes the minimum distance Δ to $\hat{\mathcal{O}}$:

$$\mathbf{o}^* = \arg \max_{\mathbf{o}_i \in \mathcal{O}_k \setminus \hat{\mathcal{O}}} \min_{\mathbf{o}_j \in \hat{\mathcal{O}}} \Delta(\mathbf{o}_i, \mathbf{o}_j),$$

where $\Delta(\mathbf{o}_i, \mathbf{o}_j)$ is the Euclidean distance between visual features. We update $\hat{\mathcal{O}} \leftarrow \hat{\mathcal{O}} \cup \{\mathbf{o}^*\}$ and repeat until all observations are ranked. In each active fine-tuning round, we select the top n_e observations from this ranking.

B.5 Calibration Experiments

We primarily measure calibration by comparing the uncertainty estimate for an initial observation with the success probability of rolling out the policy from there. More specifically, we calculate the mean uncertainty (8) per task and evaluate the per-task success rate by rolling out the policy multiple times for different initial states of the robot and the objects in the scene. To assess whether higher uncertainty correlates with a lower success rate, we calculate the Spearman rank correlation

²Weights downloaded from Hugging Face: huggingface.co/google/siglip-so400m-patch14-384.

Table 3: Prompts to evaluate calibration of the uncertainty estimators with respect to variations in the language input. Changes from the original prompt are marked in bold.

Task	P1 (orig.)	P2	P3	P4	P5
1	put both the alphabet soup and the tomato sauce in the basket	put the alphabet soup and the tomato sauce into the basket	place both the alphabet soup and the tomato sauce in the basket	move both the alphabet soup and tomato sauce to the basket	transfer the soup cans to the basket
2	put both the cream cheese box and the butter in the basket	place both the cream cheese box and the butter in the basket	put the cream cheese box and the butter into the basket	move the cream cheese and the butter into the basket	place the dairy products in the basket
3	turn on the stove and put the moka pot on it	switch on the stove and place the moka pot on it	start the stove and set the moka pot on the burner	activate the stove and put the moka pot on top of it	power on the cooktop and position the coffee maker on it
4	put the black bowl in the bottom drawer of the cabinet and close it	put the black bowl into the lower drawer of the cabinet and shut it	place the black bowl in the bottom cabinet drawer and close it	store the black bowl in the bottom cabinet drawer and close the drawer	move the dark bowl to the lowest cabinet compartment and shut it
5	put the white mug on the left plate and put the yellow and white mug on the right plate	set the white mug on the left plate and the yellow and white mug on the right plate	place the white mug on the left plate and place the yellow and white mug on the right plate	put the white cup on the left dish and the yellow-white cup on the right dish	arrange the mugs: white one on the left plate, yellow and white one on the right plate
6	pick up the book and place it in the back compartment of the caddy	grab the book and place it in the back compartment of the caddy	pick up the book and put it in the rear compartment of the caddy	take the book and place it in the back section of the caddy	move the book to the back compartment of the organizer
7	put the white mug on the plate and put the chocolate pudding to the right of the plate	place the white mug on the plate and put the chocolate pudding to the right of the plate	put the white cup on the dish and move the chocolate pudding to the right side of the plate	set the white mug on the plate and place the chocolate pudding to the right of the plate	position the mug on the plate and set the pudding to the right of it
8	place both the alphabet soup and the cream cheese box in the basket	put both the alphabet soup and the cream cheese box in the basket	put the alphabet soup and the cream cheese box into the basket	move both the alphabet soup and cream cheese box to the basket	transfer the soup and cheese box to the basket
9	put both moka pots on the stove	place both moka pots on the stove	put the two moka pots on the stove	set both coffee pots on the stove	move the two moka pots onto the cooktop
10	put the yellow and white mug in the microwave and close it	place the yellow and white mug in the microwave and close it	put the yellow and white mug into the microwave and shut it	move the yellow and white mug into the microwave and close the door	place the yellow-white mug in the microwave and shut the door

coefficient ρ between the two quantities. If and only if the success rate decreases monotonically with higher uncertainty, $\rho = -1$. As a secondary metric, we also report the Pearson correlation coefficient, which measures the strength of the linear relationship between the two quantities.

In Figure 3, we investigate whether the uncertainty scores are well-calibrated when a decrease in success rate stems solely from varying the language command. To this end, we use different language prompts for each task, listed in Table 3.

B.6 Active Fine-Tuning

The set of tasks \mathcal{T} over which we want to maximize multitask performance includes all $K = 10$ tasks from LIBERO-10. We perform active fine-tuning for $R = 15$ rounds, selecting $n_e = 5$ episodes per round, resulting in a total of 75 selected demonstrations. In each round, we warm-start the VLA ensemble’s model parameters with the final parameters from the previous fine-tuning round and perform 4,000 gradient steps with a batch size of 32. The learning rate is adapted using cosine decay, with 200 initial warmup steps followed by a decay from 5×10^{-5} to 55×10^{-5} over the remaining steps. We replay the ensemble training data \mathcal{D}_{pre} , using a replay ratio of $\lambda = 0.5$. This means that 50 % of the gradient steps are performed on data from episodes selected during active fine-tuning, and the remaining on \mathcal{D}_{pre} . This strategy avoids catastrophic forgetting [19, 60] and maintains sufficient diversity within the VLA ensemble.

B.6.1 Evaluation

In each active fine-tuning round r , we evaluate the current VLA ensemble by performing 30 environment rollouts with the first member $\pi_r^{(1)}$ for each LIBERO-10 task. We roll out the policy for up to 520 environment timesteps, terminating early upon task success.

Sample Efficiency Computation. The goal of active fine-tuning is to improve performance with as little data as possible. To quantify this capability, we measure *sample efficiency* as follows. Denote the success rate of method A after round r_1 by SR_{A,r_1} . Let $r_2 = \text{argmin}_r \text{ s.t. } \text{SR}_{B,r} \geq \text{SR}_{A,r_1}$ be the number of rounds required for method B to reach at least a similar success rate as SR_{A,r_1} . Then, if $r_2 < r_1$, method B requires $(1 - \frac{r_2}{r_1}) \times 100\%$ fewer samples to achieve the same performance, i.e., method B is $\frac{r_1}{r_1 - r_2} \times 100\%$ more sample-efficient than method A. To obtain robust results, we average across $r_1 \in \{R - 2, R - 1, R\}$. With this evaluation protocol, we find that SAVE w/ VFD requires 50 % fewer samples than Random, 45 % fewer samples than Diversity, 40 % fewer samples than SAVE w/ Action-L2, and 22 % fewer samples than SAVE w/ GU, corresponding to 101 %, 83 %, 68 %, and 29 % higher sample efficiency, respectively.

B.7 Failure Detection

For our experiments, we utilize the failure prediction framework [58], released under the MIT license. We use 10 successful rollouts per task for threshold calibration, i.e., to compute a one-sided conformal prediction band [18]. At each policy inference timestep, we compute the uncertainty score u_e and flag the rollout as *Fail* if u_e exceeds its corresponding threshold. The true-positive-rate (TPR) quantifies how many failures were correctly identified as such, and the TNR quantifies how many successful rollouts were correctly not flagged as Fail. We also report the detection time for rollouts correctly flagged as Fail, which corresponds to the timestep when the uncertainty score first exceeds its threshold. The detection time is normalized by the maximum episode length. Since failure detection is a trade-off between accuracy and early detection, we also report timestep-wise accuracy (TWA), which rewards correctly detected failures more the earlier they are flagged. Following [58], we do not cherry-pick a confidence value for conformal prediction and instead average results over quantiles (0.9, 0.91, ..., 0.99).

C Additional Results

C.1 Toy Example

We illustrate the capabilities of our VFD method for quantifying epistemic uncertainty in flow-matching models with a simple example.

Example 1. Consider a conditional generative modeling problem with $\mathbf{x} = (x_1, x_2)$, $\mathbf{y} = (y_1, y_2) \in \mathbb{R}^2$. The ground-truth conditional distribution is a bimodal Gaussian with input-dependent means; $q(\mathbf{x} | \mathbf{y}) = 0.5\mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_1(\mathbf{y}), \mathbf{I}) + 0.5\mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_2(\mathbf{y}), \mathbf{I})$, where $\boldsymbol{\mu}_1(\mathbf{y}) = (\sin(\pi y_1) + 0.5, \cos(\pi y_2) - 0.5)$ and $\boldsymbol{\mu}_2(\mathbf{y}) = (\cos(2\pi y_2) - 0.5, \sin(0.5\pi y_1) + 0.5)$. We draw a set of conditioning inputs $\{\mathbf{y}^{(n)}\}_{n=1}^N$ with $N = 4000$ from a uniform distribution over an annulus with inner radius 0.75 and outer radius 1.25 and sample $\mathbf{x}^{(n)} \sim q(\mathbf{x} | \mathbf{y}^{(n)})$. We parameterize the velocity field using an MLP and train an ensemble of $M = 5$ flow-matching models

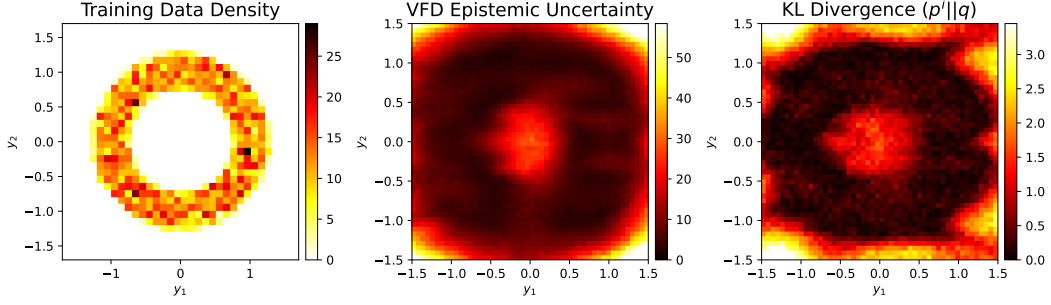


Figure 7: Epistemic uncertainty estimation for a 2D generative modeling problem. Our velocity field disagreement (VFD) uncertainty score is high for inputs far from the training distribution, similar to the KL divergence between the learned models’ conditional distributions and the ground truth.

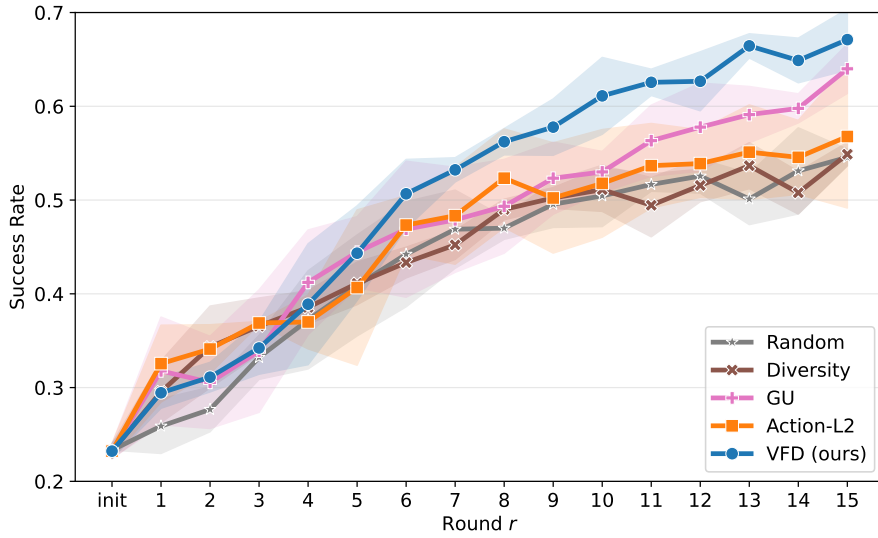


Figure 8: Learning curves corresponding to the active fine-tuning results reported in Table 2.

with a batch size of 256 for 300 epochs using the standard loss (3). As shown in Figure 7, our VFD score assigns regions far from the training data a high level of epistemic uncertainty, which closely resembles the average KL divergence $\sum_{i=1}^M D_{\text{KL}}(p^{\theta_i} || q)$ between the learned distributions and the ground truth (4a).

C.2 Additional Results for Section 6.3

We provide the full learning curves obtained in the active fine-tuning experiments in Figure 8. These plots correspond to the results reported in Tables 2 and 4.

C.3 Ensembling vs. Laplace Approximation

We compare training a VLA ensemble to computing the Laplace approximation [16] for sampling from the model posterior. Since computing the full Laplace approximation for the entire SmolVLA model is computationally intractable, we resort to the last-layer Laplace approximation, which the authors also use to compute GU in the original work [33]. We fit a diagonal last-layer Laplace approximation with `laplace-torch` [16] around each trained SmolVLA checkpoint. Concretely, we freeze the full VLA and place the posterior only on the final action projection layer, which maps

³This corresponds to sampling tasks and initial observations uniformly at random.

Table 4: Success rates in percent (\uparrow) after different active fine-tuning rounds for different demonstration selection strategies. The values reported in the summary Table 2 are marked by a * symbol.

	Round 1	Round 5	Round 10	Round 15
Diversity*	29.6 \pm 3.3	41.1 \pm 2.3	51.1 \pm 2.3	54.9 \pm 1.3
Action-L2 ($\tau = 0$)	24.1 \pm 2.7	40.6 \pm 2.4	51.1 \pm 3.4	54.0 \pm 3.4
Action-L2 ($\tau = 1$)	27.1 \pm 2.5	41.4 \pm 4.2	46.9 \pm 5.2	52.2 \pm 1.8
Action-L2 ($\tau = 1.5$)*	32.6 \pm 4.1	40.7 \pm 8.3	51.8 \pm 5.7	56.8 \pm 7.6
Action-L2 ($\tau = 2$)	31.8 \pm 4.1	40.7 \pm 9.0	53.0 \pm 5.7	55.0 \pm 2.6
Action-L2 ($\tau = 2.5$)	29.7 \pm 3.6	41.8 \pm 5.7	49.0 \pm 7.4	53.4 \pm 5.6
Action-L2 ($\tau = 3$)	27.4 \pm 3.0	41.8 \pm 6.8	46.4 \pm 8.3	53.6 \pm 5.1
GU ($\tau = 0$)	24.4 \pm 2.4	40.3 \pm 4.5	49.6 \pm 5.7	53.6 \pm 2.4
GU ($\tau = 1$)*	31.8 \pm 5.7	44.4 \pm 3.8	53.0 \pm 2.2	64.0 \pm 2.6
GU ($\tau = 1.5$)	31.4 \pm 3.7	42.0 \pm 1.2	57.4 \pm 4.3	59.2 \pm 5.5
GU ($\tau = 2$)	31.8 \pm 3.4	45.6 \pm 3.0	57.2 \pm 3.6	60.6 \pm 2.8
GU ($\tau = 2.5$)	31.4 \pm 4.2	46.7 \pm 3.8	59.3 \pm 3.4	60.1 \pm 2.2
GU ($\tau = 3$)	32.4 \pm 3.6	49.3 \pm 2.3	57.6 \pm 1.7	59.9 \pm 1.4
VFD ($\tau = 0$)	26.1 \pm 2.0	37.7 \pm 0.9	47.1 \pm 0.8	58.0 \pm 5.9
VFD ($\tau = 1$)	31.6 \pm 7.3	44.8 \pm 5.3	54.8 \pm 2.7	59.2 \pm 3.9
VFD ($\tau = 1.5$)	32.1 \pm 5.1	46.6 \pm 1.6	53.8 \pm 1.4	60.1 \pm 3.8
VFD ($\tau = 2$)	28.1 \pm 2.9	46.0 \pm 5.5	59.0 \pm 2.1	64.2 \pm 3.0
VFD ($\tau = 2.5$)*	29.4 \pm 1.6	44.3 \pm 5.0	61.1 \pm 4.1	67.1 \pm 3.2
VFD ($\tau = 3$)	28.3 \pm 3.1	45.8 \pm 2.8	59.2 \pm 3.2	62.2 \pm 4.9
VFD ($\tau = 0$, uniform)* ³	25.9 \pm 2.9	40.9 \pm 5.4	50.4 \pm 3.3	54.6 \pm 0.9
VFD ($\tau = 1$, uniform)	30.7 \pm 5.5	44.0 \pm 5.2	52.6 \pm 0.9	58.1 \pm 4.5
VFD ($\tau = 1.5$, uniform)	31.0 \pm 5.1	43.4 \pm 5.1	56.2 \pm 1.1	59.9 \pm 4.7
VFD ($\tau = 2$, uniform)	29.4 \pm 2.4	43.2 \pm 5.8	58.2 \pm 2.1	62.0 \pm 0.7
VFD ($\tau = 2.5$, uniform)	30.7 \pm 0.7	45.3 \pm 1.7	59.3 \pm 1.5	62.6 \pm 3.7
VFD ($\tau = 3$, uniform)	29.7 \pm 2.1	40.8 \pm 3.6	55.4 \pm 2.9	61.9 \pm 6.3

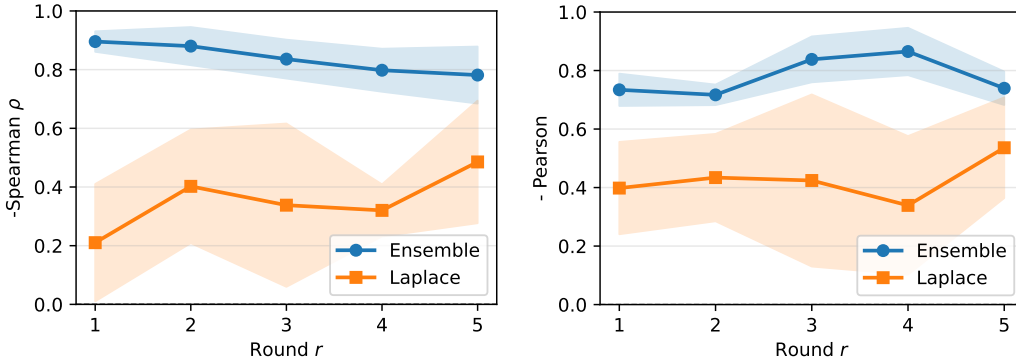


Figure 9: Ensembling vs. Laplace Approximation. Calibration is measured by the negative Spearman rank correlation (left) and the negative Pearson correlation (right) between the average task success rate and VFD uncertainty. Two separate ensemble members achieve much better calibration than sampling a second model from the last-layer Laplace approximation.

the action expert hidden states to the predicted flow-matching action velocity. The Laplace posterior is fit as a regression model on the same training episodes used for that active-learning round: for each calibration frame, we sample a flow-matching time s , a noise $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and a target action \mathbf{x}_1 , and use the corresponding velocity $\mathbf{x}_1 - \mathbf{x}_0$ as the regression target. We use all selected training episodes for the round, a calibration fraction of 1.0, and a diagonal Hessian approximation.

Even when limited to the last layer, the computational cost of the last-layer Laplace approximation remains considerable, requiring about 5 hours per checkpoint/round on an NVIDIA RTX 4090 GPU. This is much higher than the 30 minutes it takes to fine-tune a second ensemble member for 4,000 steps. Due to the high cost, we only fit the Laplace approximation for the first 5 rounds. As

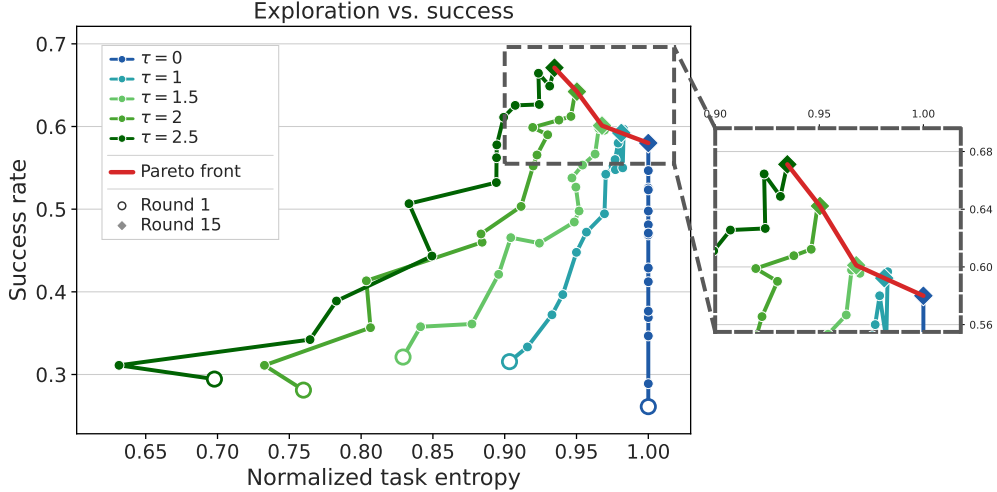


Figure 10: Pareto front of the exploration-exploitation trade-off.

shown in Figure 9, the two-member ensemble provides much better VFD score calibration than the Laplace approximation. Given also the lower computational cost of ensembling, we consider it the preferred option for posterior sampling in VLAs for uncertainty quantification.

C.4 Iterative Fine-Tuning vs. Retraining

We also test whether our iterative active-learning loop, designed to minimize data-collection effort, requires an additional separate full fine-tuning stage to yield optimal policy performance. To this end, we compare the final-round policy π_{15} to a policy retrained from the base VLA using a 50/50 split between the ensemble training data \mathcal{D}_{pre} and the actively selected data $\mathcal{D}_{\text{new}}^{(\leq 15)}$. For the selection of our best performing temperature $\tau = 2.5$, the retrained policy achieves an overall success rate of $70.6\% \pm 0.9$, which is only 3.5% higher than the iteratively fine-tuned policy. This demonstrates that uncertainty-guided acquisition produces a strong policy already during the iterative active learning process.

C.5 Effect of τ on SAVE

Figure 10 shows a detailed view of the trade-off between exploiting the uncertainty estimates and exploring the task space evenly. The exploitation is measured via the success rate of the rolled-out policy, and the diversity of the categorical task distribution is measured by its entropy. We can see that they form a Pareto front: higher temperatures increase the success rate at the cost of diversity. Naturally, this does not hold ad infinitum. In our experiments, we observe that for temperatures $\tau > 2.5$, the success rate starts declining, as too few tasks from the pool are explored (cf. Figure 11), making it harder to maximize the success rate across all tasks.

C.6 Detailed Results on Failure Detection

We provide more detailed results that include all metrics in Figure 12. Compared with baselines specifically developed for failure detection, our general VFD uncertainty estimation method demonstrates strong performance, achieving the highest accuracy (0.67), TPR (0.79), and TWA (0.54), as well as the second lowest detection time of 0.4. These results highlight the potential of VFD-based epistemic uncertainty estimation for online monitoring of VLAs, enabling targeted human intervention or activation of safety fallbacks.

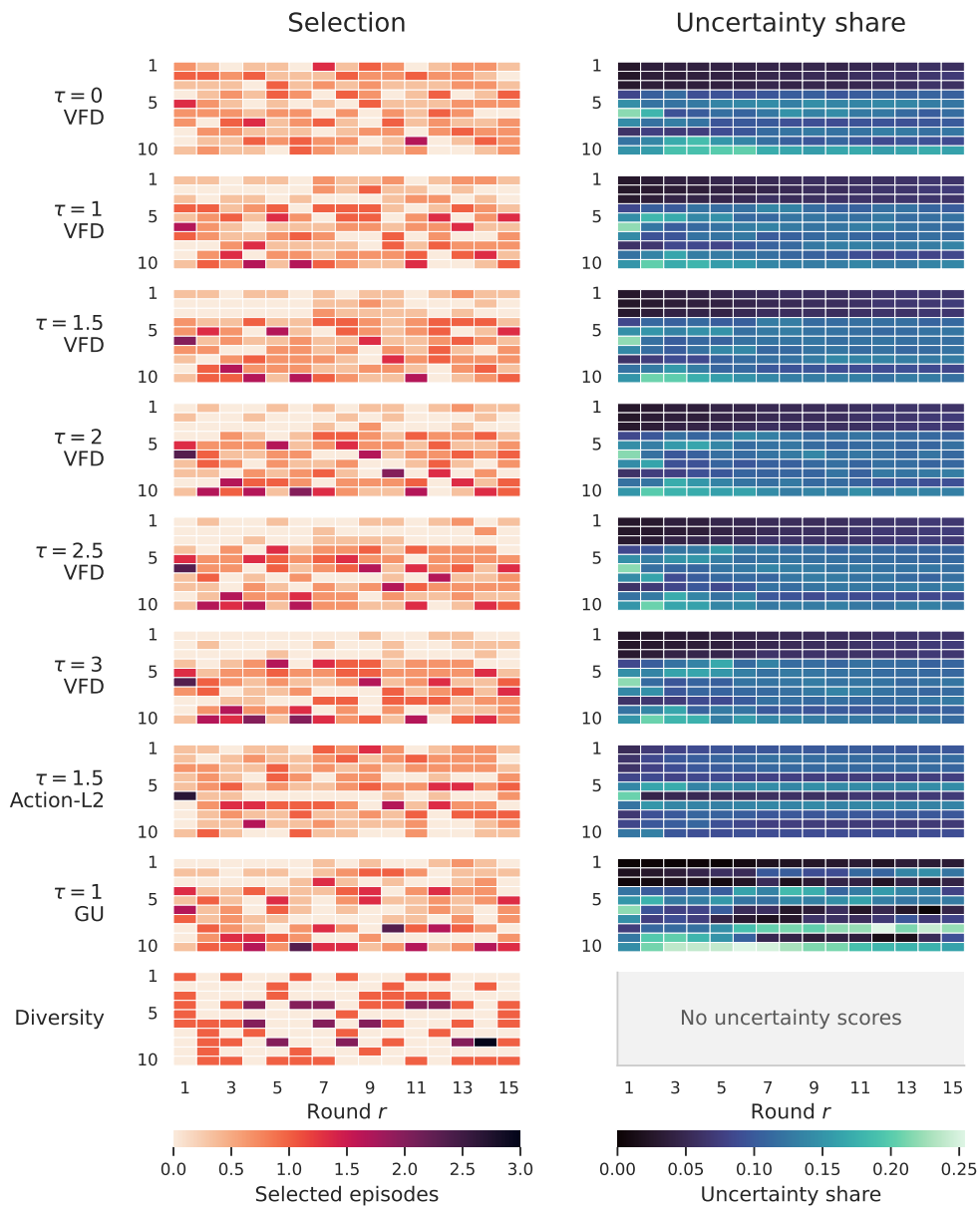


Figure 11: Behavior of our VFD-acquisition rule for different temperature values τ over all active fine-tuning rounds. Left: Number of selected episodes per task. Right: Relative share of overall uncertainty per task.

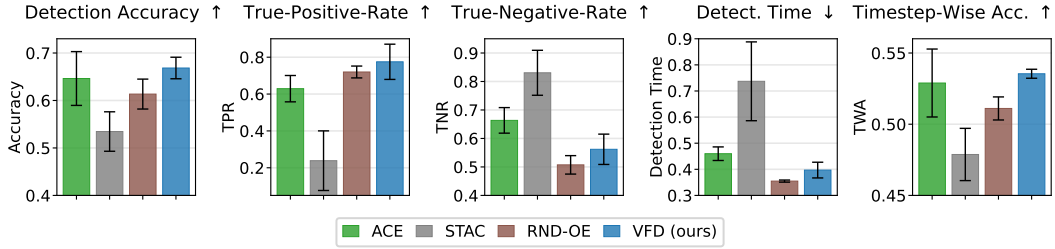


Figure 12: Full results of our failure detection experiments. VFD achieves the highest timestep-wise accuracy (TWA), which measures the capability to detect failures accurately *and* early.

D Broader Impact

This work may have positive societal impact by improving the reliability of VLAs in robotic systems. In particular, calibrated epistemic uncertainty can help identify unfamiliar situations, support more data-efficient adaptation, and enable detection of runtime failures before unsafe actions are executed. At the same time, uncertainty estimates are naturally imperfect and may pose risks if deployed as the sole safety mechanism; false confidence could lead to harmful robot behavior, while overly conservative estimates could reduce usability. These risks should be mitigated through conservative threshold design, careful human oversight, safe fallback policies, and rigorous evaluation under realistic deployment conditions before real-world use.