COHERENT MOMENT PROPAGATION: EPISTEMIC UNCERTAINTY FOR PRE-TRAINED ROBOT POLICIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Flow matching policies underpin the state-of-the-art in robot learning, with impressive results ranging from long-horizon planning to dexterous manipulation. However, they provide no measure of whether their predicted commands can be trusted, particularly in novel situations outside their training set. Epistemic uncertainty reflects uncertainty about the model's weights and quantifies this ignorance, but calculating it at inference time from a flow matching policy is not trivial. Actions are produced by an ODE solver, requiring uncertainty to be propagated through the entire integration process. BayesDiff addresses this problem for diffusion models by propagating the mean and variance of the state through the denoising chain under a last-layer Laplace posterior approximation, but it severely underestimates the epistemic uncertainty. In this paper we: i) prove that, for flow matching and diffusion ODEs, BayesDiff's total variance decays at rate $O(1/T)$ as the number of Euler steps T used by the solver increases; ii) introduce Coherent Moment Propagation (CoMP), two recursions inspired by the extended and unscented Kalman filters that fix this problem by carrying the cross-covariance between state and weights; and iii) widen the last-layer posterior with observation embeddings the policy already computes, improving uncertainty estimates without retraining for CoMP and several test-time estimators alike. We demonstrate these benefits on three manipulation benchmarks with applications in failure detection and demonstration selection for fine-tuning, where CoMP corrects BayesDiff's variance collapse while matching or outperforming several uncertainty estimators.

1 INTRODUCTION

Flow matching (Lipman et al., 2023; Liu et al., 2023b; Albergo et al., 2025) has become an integral component of robot stacks, powering action heads in state-of-the-art robot policies (Black et al., 2025b; Bjorck et al., 2025; Shukor et al., 2025). Despite their widespread success, these models lack an inherent mechanism to assess the reliability of their predictions. Epistemic uncertainty provides one such indicator, measuring how much predictions vary across parameter settings consistent with the training data (Kendall & Gal, 2017). However, obtaining this uncertainty at test time is increasingly important given the resources invested in foundation policies Bjorck et al. (2025), whose reliability on unfamiliar tasks remains difficult to assess (Xu et al., 2025; Gu et al., 2026).

Uncertainty quantification of a flow matching-based policy is not trivial, as it requires propagating model uncertainty through the ODE solver that generates robot commands. A common technique to quantify uncertainty is to compare the predictions from an ensemble of policies, whether trained (Lakshminarayanan et al., 2017; Pearce et al., 2020; Berry et al., 2024; Römer et al., 2026), generated by a hypernetwork (Chan et al., 2024) or drawn from a posterior fit after training (Jazbec et al., 2025; Huang et al., 2025). However, training and storing independent model copies is costly at the scale of foundation policies, motivating estimators that work from a single pre-trained policy.

Post-hoc Bayesian inference offers a way to obtain epistemic uncertainty from pre-trained models (Daxberger et al., 2021; Jazbec et al., 2025). Particularly, BayesDiff (Kou et al., 2024) combines a last-layer Laplace posterior with *moment propagation*, carrying the mean and variance of the denoised state through the sampling chain of a diffusion model (Ho et al., 2020), with extensions beyond the last layer (Gupta et al., 2026), to flow matching models (Han et al., 2026), and to multi-agent trajectory modeling (Capellera et al., 2025). However, we prove that BayesDiff exhibits

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

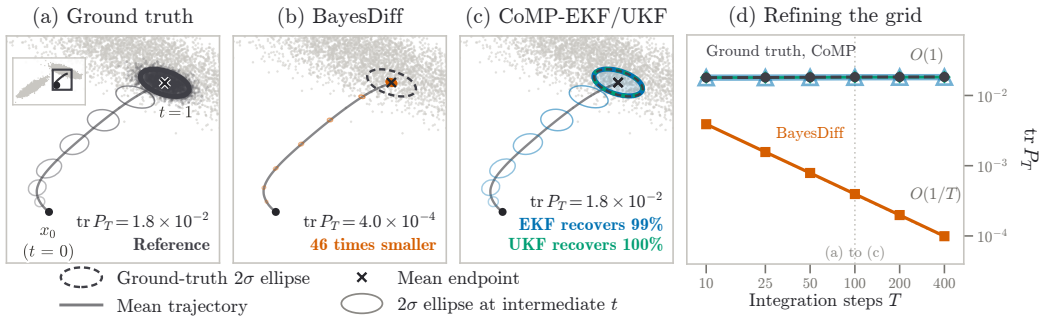

Figure 1: Variance collapse in moment propagation. State covariance P_T at the end of integration of a flow matching model on a two-mode mixture under a last-layer Laplace posterior for one sample x_0 at $T = 100$ Euler steps. (a) The reference is the empirical covariance of 4000 trajectories, one per posterior draw $\theta^{(m)} \sim \mathcal{N}(\theta_{\text{MAP}}, \Sigma_\theta)$. (b) BayesDiff returns $1/46$ of it, consistent with Theorem 1. (c) Our estimators CoMP-EKF and CoMP-UKF carry the cross-covariance between the state and the weights and recover the reference to within 1%. (d) The total variance $\operatorname{tr} P_T$ at the same seed for every estimator re-run on each grid from $T = 10$ to 400 Euler steps. BayesDiff halves with every doubling of T, while our methods track the reference regardless of the number of Euler steps.

variance collapse on flow matching and diffusion ODEs: under Lipschitz regularity and bounded predictive variance, its estimated total variance is upper bounded by $O(1/T)$ in the number of Euler steps T, so it vanishes as the solver is refined; see Fig. 1 (b) and (d). This creates an artificial tradeoff: finer integration better approximates the ODE solution, yet drives the reported uncertainty to zero.

To address this limitation, we introduce ***Co**herent **M**oment **P**ropagation* (CoMP), a method that carries the cross-covariance between state and weights, yielding an upper bound on total variance independent of T, see Fig. 1 (c) and (d). We show that BayesDiff's variance collapse results from treating the weights as independent of the state they generate, and derive two recursions inspired by the extended and unscented Kalman filters. Furthermore, to improve uncertainty estimates from a last-layer Laplace approximation (Deng et al., 2022; Calvo-Ordoñez et al., 2026), we draw inspiration from LULA (Kristiadi et al., 2021) and widen the last layer with observation embeddings already computed by foundation robot policies, without re-training or changing their action predictions.

In summary, our contributions are:

1. We prove BayesDiff's variance collapse and show its total variance is upper bounded by $O(1/T)$ for flow matching and diffusion ODEs with the number of Euler integration steps T;
2. We derive two new uncertainty estimators, **CoMP-EKF** and **CoMP-UKF**, from a linearized sampler recursion that carries state-weight cross-covariance, with a constant total variance bound; and
3. We improve last-layer Laplace uncertainty estimates by widening the head with existing observation embeddings, without re-training the policy or changing its action predictions.

We demonstrate the practical utility of our uncertainty estimators in the problems of runtime failure detection (Römer et al., 2025; Xu et al., 2025) and demonstration selection (Agia et al., 2025; Sirigiri et al., 2026) for fine-tuning (Bagatella et al., 2025; Römer et al., 2026). Across three robot policies and three manipulation benchmarks (robomimic (Mandlekar et al., 2022), LIBERO (Liu et al., 2023a), and RoboLab (Yang et al., 2026)), CoMP is better calibrated than BayesDiff and matches or outperforms several test-time uncertainty estimators for pre-trained robot policies. Lastly, we show that the augmented head further improves uncertainty estimates from CoMP and several baselines.

2 Preliminaries

Flow matching. Let $x \in \mathbb{R}^D$ denote the state and $u_\theta(x, t)$ a velocity field with parameters θ that transports noise to data through an ODE over $t \in [0, 1]$ with $x_0 \sim \mathcal{N}(0, I)$ and x_1 drawn from the data distribution. Following the conditional flow matching formulation (Lipman et al., 2023; Albergo et al., 2025), u_θ is trained on the affine interpolant $x_t = (1 - t)\, x_0 + t\, x_1$ by minimizing:

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t, x_0, x_1}\left[\left\| u_\theta(x_t, t) - (x_1 - x_0) \right\|^2\right], \tag{1}$$

with $t \sim \mathcal{U}[0,1]$. Sampling integrates $\mathrm{d}x/\mathrm{d}t = u_\theta(x,t)$ from a noise sample $x_0 \sim \mathcal{N}(0,I)$, using T Euler steps of size $\Delta t \triangleq 1/T$, with $t_i \triangleq i\Delta t$, so that the generated sample x_T results from iterating:

$$x_{i+1} = x_i + \Delta t\, u_\theta(x_i, t_i). \tag{2}$$

Flow-based robot policies. A vision-language-action (VLA) policy $\pi_\theta(a_{1:H} \mid o)$ maps an observation $o = (s, I, \ell)$ comprising proprioceptive state s, camera images I and a language instruction ℓ to a distribution over the next H actions (Black et al., 2025b; Bjorck et al., 2025; Shukor et al., 2025). For flow matching-based VLAs, a vision-language backbone encodes o once into $c(o)$, held fixed while an action expert uses (2) to integrate $u_\theta(x, t, c(o))$ from noise to an action chunk:

$$x_T \;=\; a_{1:H} \triangleq (a_1, \ldots, a_H) \in \mathbb{R}^{H \times d_a}, \tag{3}$$

where H is the predicted horizon and d_a the dimension of one action, e.g. an end-effector displacement. The state of the flow is therefore the chunk flattened to $D = H\, d_a$, and the policy executes its first $h \leq H$ actions before replanning. In the following we omit the conditioning on o.

Laplace approximation. The Laplace approximation (Daxberger et al., 2021) models the weight posterior $p(\theta \mid \mathcal{D}) \propto p(\mathcal{D} \mid \theta)\, p(\theta)$ of a network trained with data $\mathcal{D}$ as a Gaussian. This approximation allows us to quantify epistemic uncertainty, which reflects variation in the network's predictions for the same input across weights drawn from the posterior. The Gaussian is centered at a posterior mode, with covariance given by the inverse curvature of the negative log-posterior:

$$p(\theta \mid \mathcal{D}) \approx \mathcal{N}(\theta;\, \theta_{\mathrm{MAP}},\, \Sigma_\theta)\,, \qquad \Sigma_\theta \triangleq (\,\nabla^2_\theta \mathcal{L}(\mathcal{D};\theta)\big|_{\theta_{\mathrm{MAP}}})^{-1}, \tag{4}$$

where the objective $\mathcal{L}(\mathcal{D};\theta) \triangleq \mathcal{L}_{\mathrm{CFM}}(\mathcal{D};\theta) + (\lambda/2)\|\theta\|^2$ is a negative log-posterior, with the flow matching loss (1) as the negative log-likelihood $-\log p(\mathcal{D} \mid \theta)$ and weight decay as the negative log-prior under $p(\theta) = \mathcal{N}(0, \lambda^{-1} I)$. Computing the full Hessian is costly, so approximations restrict the posterior to the **last layer**, whose curvature can be estimated from a data subset $\mathcal{D}_{\mathrm{fit}}$, freezing the remaining weights at their MAP (Kristiadi et al., 2020). Here, θ denotes the last-layer weights, θ_{MAP} their pretrained value, and $\delta\theta \triangleq \theta - \theta_{\mathrm{MAP}} \sim \mathcal{N}(0, \Sigma_\theta)$ the deviation of a single posterior draw (4).

Augmented last-layer posterior. A posterior on the last layer can only capture what its input features $\phi(x,t)$ retain. A last layer trained to predict a velocity can over-compress observations, keeping only what the velocity needs and discarding the rest, potentially leading to feature collapse (van Amersfoort et al., 2021; Mukhoti et al., 2023). LULA (Kristiadi et al., 2021) addresses this by augmenting the features feeding the last layer to improve uncertainty calibration, while zero output weights on the added units preserve the network's predictions. For a linear last layer, this gives

$$u_\theta(x,t) = \theta\, \psi(x,t), \qquad \psi(x,t) \triangleq \big(\phi(x,t),\, \zeta\big), \qquad \theta_{\mathrm{MAP}} \leftarrow \big(\theta_{\mathrm{MAP}},\, 0\big), \tag{5}$$

where ζ contains the activations of the added hidden units. Here, θ_{MAP} denotes the pretrained weights with zero output weights appended. Thus, the output at θ_{MAP} is unchanged, while the Laplace covariance in (4) is computed from the loss curvature at θ_{MAP} using the augmented features ψ. However, LULA learns these added features using an out-of-distribution set, which a pre-trained policy does not provide.

The BayesDiff recursion. BayesDiff (Kou et al., 2024) introduced moment propagation for diffusion models, placing a last-layer Laplace posterior (4) on the denoiser and carrying the mean and variance of the state along the sampling chain, so that the terminal state variance captures the epistemic uncertainty of the generated sample. Here we express its recursion for the flow matching sampler (2), where different weights drawn from the posterior can predict different velocities at the same state x and time t. BayesDiff represents uncertainty through per-coordinate variances, retaining only the diagonal of the predictive covariance:

$$p(u \mid x, t, \mathcal{D}) \approx \mathcal{N}\big(u;\, u_{\theta_{\mathrm{MAP}}}(x,t),\, \mathrm{diag}\,\gamma^2(x,t)\big)\,, \tag{6}$$

with mean velocity produced by the MAP weights and variance $\gamma^2(x,t)$ given by the diagonal of $J^\theta \Sigma_\theta (J^\theta)^\mathsf{T}$, where $J^\theta(x,t) \triangleq \partial u/\partial\theta$ is the Jacobian of the velocity with respect to the weights. Writing $u_i \triangleq u_\theta(x_i, t_i)$ for the velocity at step i, the procedure carries the mean $m_i \triangleq \mathbb{E}[x_i]$ and the covariance of the state $P_i \triangleq \mathrm{Cov}(x_i)$, where all expectations, variances and covariances are taken

over x_i and u_i. Taking expectations and variances of the Euler update (2), which is affine in the velocity, gives an exact moment recursion. BayesDiff keeps the mean and diagonal of P_i:

$$m_{i+1} = m_i + \Delta t\, \mathbb{E}[u_i]\,, \tag{7}$$

$$P_{i+1} = P_i + 2\,\Delta t\ \mathrm{diag}\,\mathrm{Cov}(x_i, u_i) + \Delta t^2\ \mathrm{diag}\,\mathrm{Cov}(u_i)\,, \tag{8}$$

started from $m_0 = x_0$ and $P_0 = 0$ and iterated for T steps, with final state covariance P_T estimating the uncertainty of the generated sample. BayesDiff takes the conditional velocity variance from (6) and estimates the cross-covariance $\mathrm{Cov}(x_i, u_i)$ by Monte Carlo sampling S particles from $x_i^{(s)} \sim \mathcal{N}(m_i, P_i)$ and evaluating their MAP velocities. Every expectation above is taken under the product $p(u \mid x_i, t_i, \mathcal{D})\, p(x_i)$, whose first factor is the predictive distribution (6) with the weights integrated out, $p(u \mid x_i, t_i, \mathcal{D}) = \int p(u \mid x_i, t_i, \theta)\, p(\theta \mid \mathcal{D})\, \mathrm{d}\theta$. However, this marginalization treats the weights as independent of the state they generate, which is the key limitation of BayesDiff. A posterior trajectory uses one weight draw throughout integration, so x_i depends on that draw. Discarding this dependence leads to the variance collapse shown in the next section.

3 Coherent Moment Propagation

In this section, we prove that BayesDiff's estimated total variance collapses to zero as the integration grid of the solver is refined. We then introduce Coherent Moment Propagation (CoMP), which avoids this collapse by propagating the state-weight cross-covariance through the ODE sampler.

BayesDiff total variance recursion. To analyze BayesDiff's reported uncertainty, we consider the *total variance* of the generated sample, $\mathrm{tr}\, P_T = \sum_d (P_T)_{dd}$. Taking the trace of (8) collapses the covariance update into a single scalar recursion,

$$\mathrm{tr}\, P_{i+1} = \underbrace{\mathrm{tr}\,\mathrm{Cov}_x[x_i + \Delta t\, u_{\theta_{\mathrm{MAP}}}(x_i, t_i)]}_{\text{transport of the belief}} + \underbrace{\Delta t^2\, \bar{\sigma}_i^2}_{\text{epistemic injection}}\,, \tag{9}$$

where $\mathbb{E}_x \equiv \mathbb{E}_{x_i}$, $\mathrm{Cov}_x \equiv \mathrm{Cov}_{x_i}$, and $\bar{\sigma}_i^2 \triangleq \mathbb{E}_x[\sum_d \gamma_d^2(x_i, t_i)]$ averages the summed predictive variance (6) over the state belief. Each step pushes the belief through the deterministic Euler map $h_i(x_i) \triangleq x_i + \Delta t\, u_{\theta_{\mathrm{MAP}}}(x_i, t_i)$ and adds the epistemic injection $\Delta t^2 \bar{\sigma}_i^2$, through which the weight posterior enters the update. Since the chain takes $T = 1/\Delta t$ steps, the epistemic content of the reported spread is T injections of size Δt^2, and whether anything survives the refinement of the integration grid depends entirely on how much the transport term amplifies each contribution.

The estimated total variance is upper bounded by $O(1/T)$. We prove that BayesDiff's total variance is upper bounded by $O(1/T)$, so increasing the number of integration steps leads to an underestimated uncertainty. The result uses the joint law of (8), a fixed initial noise sample ($\mathrm{tr}\, P_0 = 0$), and two regularity conditions:

- **(A1)** *Lipschitz velocity field.* The velocity field $u_{\theta_{\mathrm{MAP}}}$ is globally Lipschitz in the state with constant L, as is normally assumed when analysing the convergence of diffusion and flow samplers (Chen et al., 2023a; Benton et al., 2024).
- **(A2)** *Bounded noise injection rate.* The summed predictive variance obeys $\sum_d \gamma_d^2(x, t) \le \bar{\sigma}_{\max}^2 < \infty$ at every state and time, with a bound independent of T such that $\bar{\sigma}_i^2 \le \bar{\sigma}_{\max}^2$.

Theorem 1 (Variance collapse)**.** *Under (A1) and (A2), the total variance under moment recursion* (8) *satisfies*

$$\mathrm{tr}\, P_T \ \le\ e^{2L}\, \frac{\bar{\sigma}_{\max}^2}{T} \ =\ O(1/T), \tag{10}$$

i.e., the total variance vanishes as the integration grid refines. See Appendix A.1 for the proof.

Proof sketch. Each step of (9) stretches the belief by at most $(1 + \Delta t\, L)^2$ under (A1) and adds at most $\Delta t^2 \bar{\sigma}_{\max}^2$ under (A2), so the T injections are bounded by $e^{2L} \bar{\sigma}_{\max}^2 / T$. Importantly, under (A1) and Lipschitz continuity in time, the Euler iterates converge to the solution of the continuous-time ODE $\mathrm{d}x/\mathrm{d}t = u_{\theta_{\mathrm{MAP}}}(x, t)$ as $T \to \infty$ at first order in the step size (Hairer et al., 2000). Corollary 1 shows that carrying full state covariance does not prevent the collapse. We also extend this bound to diffusion SDEs: Corollary 2 gives an $O(1/T)$ epistemic term in the upper bound alongside an $O(1)$

aleatoric term from Brownian noise (Appendix A.1). A finer integration grid can therefore collapse the reported total variance, underestimating epistemic uncertainty. To overcome this limitation, we introduce an alternative to BayesDiff called CoMP, which restores the omitted state-weight cross-covariance, giving an upper bound on total variance independent of the number of integration steps.

Our key idea is that rather than marginalize the weights at every step, we linearize the ODE sampler (2) about its mean trajectory and propagate the state and the weight deviation jointly, so the recursion accounts for their dependence instead of implicitly setting it to zero. We linearize jointly in (x_i, θ) about $(m_i, \theta_{\mathrm{MAP}})$, where (A1) bounds the state Jacobian by $\|J_i^x\| \le L$. For a linear last layer[1] the parameter Jacobian is available analytically. We develop two approximations for propagating state uncertainty through the ODE sampler.

Expanding the Euler update $x_i + \Delta t\, u_\theta(x_i, t_i)$ to first order in both x_i and θ about $(m_i, \theta_{\mathrm{MAP}})$ gives the linearized recursion

$$\delta x_{i+1} = A_i\, \delta x_i + B_i\, \delta\theta, \qquad A_i \triangleq I + \Delta t\, J_i^x, \qquad B_i \triangleq \Delta t\, J_i^\theta, \tag{11}$$

with $\delta x_i \triangleq x_i - m_i$ the state deviation under the linearized model, $\delta\theta$ the weight deviation of §2, drawn once and held fixed throughout integration, and J_i^x and J_i^θ the state and weight Jacobians of the velocity at $(m_i, t_i, \theta_{\mathrm{MAP}})$. Stacking the two deviations into $z_i \triangleq (\delta x_i, \delta\theta)$ turns (11) into the linear map $z_{i+1} = F_i z_i$ with F_i the block matrix with rows (A_i, B_i) and $(0, I)$, whose identity block carries $\delta\theta$ unchanged, so the joint moments propagate in closed form. We carry the state mean m_i, the full $D \times D$ covariance P_i, and the state-parameter cross-covariance $C_i \triangleq \mathrm{Cov}(\delta x_i,\, \delta\theta)$, which measures how much of the state's departure by step i is explained by the weights that drove it there. For this linearized model, the moment updates are exact. Written blockwise, Coherent Moment Propagation carries the triple (m_i, P_i, C_i) from $(x_0, 0, 0)$ to the end of the chain by

$$m_{i+1} = h_i(m_i), \tag{12}$$

$$C_{i+1} = A_i\, C_i + B_i\, \Sigma_\theta, \tag{13}$$

$$P_{i+1} = \underbrace{A_i P_i A_i^\mathsf{T}}_{\text{transported state spread}} + \underbrace{A_i C_i B_i^\mathsf{T} + B_i C_i^\mathsf{T} A_i^\mathsf{T}}_{\text{state-weight cross terms}} + \underbrace{B_i \Sigma_\theta B_i^\mathsf{T}}_{\text{weight contribution}}. \tag{14}$$

The mean (12) follows the MAP trajectory; (13) supplies the cross terms. At a fixed state, a linear last layer makes the velocity affine in θ, so J_i^θ and B_i can be computed analytically from its features.

The upper bound does not depend on the number of integration steps. The same two assumptions that gave Theorem 1 its upper bound give this recursion a different one, under the same integration grid and the same constant.

Theorem 2 (Upper bound independent of integration steps)**.** *Under (A1) and (A2), the total variance under recursion* (12)–(14) *from* $(x_0, 0, 0)$ *satisfies*

$$\operatorname{tr} P_T \;\le\; e^{2L}\, \bar\sigma_{\max}^2 \;=\; O(1), \tag{15}$$

whereas the same recursion with $C_i \equiv 0$ *has the same upper bound* (10) *as Theorem 1. See Appendix A.2 for the proof.*

The two upper bounds carry the same constant and differ by the factor $\Delta t = 1/T$. Unrolling the recursion collects T contributions of size Δt: carrying C_i adds them as amplitudes before squaring, while holding C_i at zero adds their squares as variances. We therefore attribute the vanishing spread of Theorem 1 to the cross-covariance that BayesDiff omits, and Theorem 2 says that carrying it removes the factor $1/T$ from the bound. Figure 1 compares the estimators' spread with that of a posterior ensemble on a flow matching model of a two-dimensional mixture. To realize this estimator, we still need the state transition A_i and transported spread $A_i P_i A_i^\mathsf{T}$. We derive two variants inspired by the extended and unscented Kalman filters.

CoMP-EKF. With the state Jacobian $J_i^x = \partial u_{\theta_{\mathrm{MAP}}}/\partial x_i$ evaluated at (m_i, t_i), running (12)–(14) as written is the extended Kalman filter's predict step (Särkkä, 2013) applied to the sampler, with batched autodiff supplying J_i^x at each step. We provide pseudocode in Algorithm 1.

[1]The action heads of recent VLAs are of this form, among them $\pi_{0.5}$ (Black et al., 2025a), GR00T (Bjorck et al., 2025) and SmolVLA (Shukor et al., 2025), each of which has a linear last layer.

CoMP-UKF. The unscented transform (Särkkä, 2013) represents the state mean m_i and covariance P_i with $2D + 1$ deterministic sigma points, which we propagate through the nonlinear Euler map h_i, capturing nonlinear effects beyond CoMP-EKF's local linearization. The same sigma points estimate A_i for the state-weight cross-covariance update in (13). Appendix A.4 gives the moment equations and pseudocode (Algorithm 2). Both CoMP estimators use the trace of P_T or of the chunk's endpoint covariance as the **uncertainty score** U (Appendix A.5).

Augmented head. We run CoMP with the plain last-layer posterior over features ϕ and the augmented posterior (5). LULA (Kristiadi et al., 2021) learns the added features using an out-of-distribution set, which we do not assume is available. We instead set $\zeta = c(o)$ in (5), reusing the image, language and proprioception embeddings already computed by VLAs such as $\pi_{0.5}$ (Black et al., 2025a) and SmolVLA (Shukor et al., 2025). Fitting this augmented Laplace posterior requires no policy retraining and it uses ψ instead of ϕ for B_i in (13)–(14).

4 Evaluation

To assess how CoMP's uncertainty estimates benefit pre-trained policies, we ask four questions:

- **Q1:** Does the covariance our estimator propagates match the posterior it approximates? (§4.1)
- **Q2:** Does the augmented head make uncertainty estimates better reflect input ambiguity, particularly in language instructions? (§4.2)
- **Q3:** Can CoMP's uncertainty estimates predict future failures during policy execution? (§4.3)
- **Q4:** Can uncertainty-guided demonstration selection improve sample efficiency for policy fine-tuning in new tasks? (§4.4)

Models and baselines. We evaluate three robot policies: a transformer-based Diffusion Policy (Chi et al., 2025) trained with flow matching (Lipman et al., 2023) on robomimic (Mandlekar et al., 2022), $\pi_{0.5}$ (Black et al., 2025a) on LIBERO (Liu et al., 2023a) and RoboLab (Yang et al., 2026), and SmolVLA (Shukor et al., 2025) on LIBERO. For $\pi_{0.5}$ and SmolVLA, we retain the default number of integration steps except in ablations (Appendix A.6). We adapt five post-hoc uncertainty estimators to flow matching policies. **BayesDiff** (Kou et al., 2024) is the Monte Carlo moment recursion described in §2. **DEUA** (Huang et al., 2025) re-noises the generated sample and scores velocity variance across posterior draws. **GU** (Jazbec et al., 2025) is the Gaussian entropy of the terminal state across posterior draws. **DECU** (Berry et al., 2024) estimates mutual information from the pairwise distances between member velocities at a branching point of the integration path. **VFD** (Römer et al., 2026) integrates the time-weighted disagreement between member velocity fields along the integration path. We also report DECU and VFD with independently trained ensembles as references (Römer et al., 2026), since they require policy training beyond our test-time setting. All post-hoc estimators use Kronecker-factored last-layer Laplace posteriors (4).

4.1 Faithfulness of the propagated covariance

To answer Q1, we test the calibration of the moment propagation estimators against a reference chunk covariance, $P^\star$, estimated via posterior ensembles on robomimic (Mandlekar et al., 2022). We train one transformer-based policy per task on the proficient-human demonstrations of *Lift*, *PickPlaceCan* and *NutAssemblySquare* (Figure 10). The posterior (4) is fit on the training split; training and sampler settings are given in Appendix A.7. We use three complementary metrics to assess calibration. The **total variance ratio** $\operatorname{tr} P / \operatorname{tr} P^\star$ measures recovered spread. The **variogram discrepancy** compares pairwise moments of order $1/2$ (Scheuerer & Hamill, 2015) and is sensitive to correlations. Both metrics are reported relative to their reference values, so a value of 1 corresponds to a perfect match. Total variance ratios below or above 1 indicate under- or overestimation, respectively, while variogram ratios above 1 indicate greater discrepancy with held-out posterior samples than the reference Gaussian. Finally, we use the **negative log likelihood (NLL)** to measure how much density the estimated Gaussian assigns to expert demonstrations; lower is better.

Figure 2 shows results with the plain head[2]. BayesDiff recovers only 6% to 23% of the reference variance depending on the task (Fig. 2, left), and its variogram discrepancy is 13 to 16 times the

[2]This keeps the focus on covariance propagation; augmented-head results are similar (Figure 6).

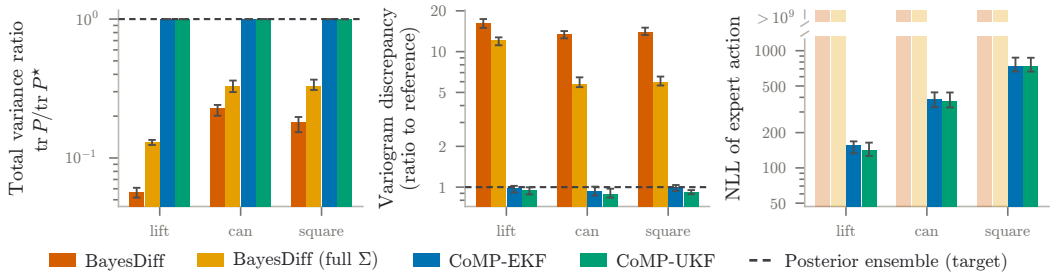

Figure 2: Calibration against the posterior ensemble on robomimic. Each bar is the median over 150 observation windows per task with a bootstrap 95% interval. Reference values in the first two panels use $M = 6000$ posterior draws. CoMP outperforms BayesDiff across all three metrics, with expert-action negative log likelihoods (NLLs) far below those of both BayesDiff variants ($> 10^9$).

reference value (Fig. 2, middle), i.e., the correlation across the chunk is misreported along with its scale. Propagating the full covariance with BayesDiff, i.e., *BayesDiff (full Σ)*, improves over *BayesDiff* but still misestimates the posterior distribution, as shown in Corollary 1. Both CoMP estimators recover the reference variance to within 0.2% and variogram discrepancy to within 10%, with lower NLLs indicating better agreement with expert demonstrations, showing better calibration overall.

4.2 Response of the Augmented Head to the Observation

To test whether the augmented head improves the response of estimated uncertainty to policy inputs, e.g., language instructions (Q2), we use RoboLab (Yang et al., 2026) for its high visual fidelity and three levels of instruction specificity: specific, default and vague (see Table 2). We evaluate ten tabletop tasks (Figure 7) from the same 100 initial configurations per task. We use openpi's $\pi_{0.5}$ fine-tuned on DROID Khazatsky et al. (2024), as released, which has *not* been trained on RoboLab data. We fit both Laplace heads, plain head and augmented head, from §3, once for all ten tasks, on up to ten successful rollouts of the *specific* wording mixed with DROID data. We evaluate uncertainty using only the first observation of each held-out rollout, varying only the instruction wording while keeping images and proprioception fixed. See Appendix A.8 for model and training details.

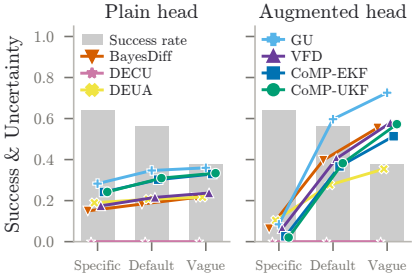

Figure 3: Language instruction shift on RoboLab. Success rate (grey bars) and first-decision uncertainty (lines, scaled to $[0, 1]$) averaged over the ten tasks by wording. On the plain head (left) estimators are mostly flat with the wording; on the augmented head (right) every estimator but DECU rises as language ambiguity increases.

As instructions become less specific, success rates should decline and uncertainty increase. However, all estimators report nearly constant uncertainty with the plain head (see Figure 3). With the augmented head, uncertainty rises with instruction ambiguity, for every estimator but DECU, which is saturated at $\ln M$, the ceiling of its mutual-information proxy. See per task results in Figure 8. This response shows that the augmented head can identify ambiguous language instructions with the vision language embeddings readily computed by the pre-trained policy. We also observe that BayesDiff's responsiveness to ambiguous instructions decreases with more integration steps, whereas CoMP is not affected (Fig. 9 in Appendix A.8), validating our theoretical results in §3.

4.3 Runtime Failure Detection

To evaluate early failure detection (Q3), we follow FIPER (Römer et al., 2025; 2026) on two simulation suites. On robomimic, we use the same tasks and policies of §4.1 with perturbed initial end-effector and object poses (Figure 10), with 50 rollouts per task and perturbation condition. On LIBERO (Liu et al., 2023a), we evaluate openpi's `pi05_libero` on nine held-out LIBERO-90

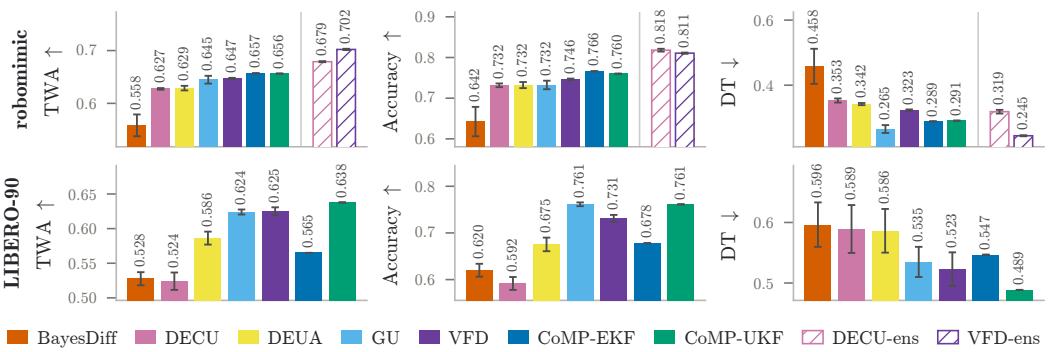

Figure 4: Runtime failure detection on robomimic and LIBERO-90. Timestep-wise accuracy (TWA), accuracy and normalized detection time (DT), averaged over tasks, with mean and standard deviation over five sampling seeds. CoMP uses our augmented head; post-hoc baselines use the plain head. The ensembles (hatched) have two independently trained members and are a reference only. CoMP leads across metrics on both suites; Table 3 reports comparisons with both heads.

tasks (Figure 11), chosen with a success rate between 0.3 and 0.7 as in FIPER. Motivated by the results from §4.2, we use CoMP with the augmented head and compare it against post-hoc baselines[3].

Following Römer et al. (2025), a rollout is flagged once the uncertainty score U exceeds a conformal threshold (Angelopoulos & Bates, 2023) calibrated on 20 rollouts per task. We report detection accuracy, detection time normalized by the maximum episode length (DT), and timestep-wise accuracy (TWA), an accuracy measure that penalizes late failure detections. Both CoMP variants outperform the post-hoc baselines in TWA and accuracy on robomimic. On LIBERO-90, CoMP-UKF achieves the highest TWA (0.638) and lowest DT (0.489), matching GU's accuracy (0.761), while CoMP-EKF lags behind (Figure 4), which we attribute to linearization errors. Although the trained ensembles (hatched) lead on robomimic, CoMP remains competitive at a fraction of their fitting and storage cost: its Laplace posterior stores $\lesssim 1\%$ of the ensemble parameters, fit in one pass over the training split (see Appendix A.9 for implementation details and additional results).

4.4 Uncertainty-guided fine-tuning

To test whether a pre-trained policy's uncertainty can inform demonstration selection for efficient fine-tuning (Q4), we adapt the budgeted selection setting of CUPID (Agia et al., 2025) and FAKTUAL (Sirigiri et al., 2026): we compare policies fine-tuned on X demonstrations selected by uncertainty or drawn at random. Following SAVE (Römer et al., 2026), we train SmolVLA (Shukor et al., 2025) on the LIBERO-Spatial, Object and Goal suites (Liu et al., 2023a) and three LIBERO-10 tasks; all LIBERO-10 demonstrations form the pool (Figure 12). Each estimator scores demonstrations at their first frame for uncertainty-guided sampling. We fine-tune once on the selection mixed with the original training data and evaluate on all ten tasks. We repeat this for five independently trained policies and budgets X from 25 to 250, with a two-member VFD as a reference. We compare CoMP with the same set of post-hoc baselines as in §4.3. Appendix A.10 presents results of post-hoc baselines with both heads (see Figure 13) and provides implementation details.

Figure 5 shows how many demonstrations each method needs to reach 90% of the success achieved with $X = 250$ randomly selected demonstrations (top), and its mean success-rate gain over random selection across budgets (bottom). On average, CoMP-UKF reaches the target success rate with the fewest demonstrations and yields the largest mean gain over random selection (3.5 pp), using a single policy and a 75 MB posterior instead of a second 1.2 GB policy. CoMP-EKF shows little benefit, possibly missing nonlinear effects that CoMP-UKF captures by propagating sigma points through the network. In the supplementary material we show how our augmented head improves fine-tuning results for every estimator (Figure 13), addressing the limited expressivity of the plain head. We also show that BayesDiff loses its fine-tuning gain as integration steps increase, whereas CoMP-UKF retains its gain (Table 4). Metric definitions and full curves are in Appendix A.10.

[3]The post-hoc baselines use the plain head by default, though we also evaluate them with our augmented head in Appendix A.9 to isolate its contribution.

5 Related Work

Epistemic uncertainty helps neural networks quantify when their own predictions may be unreliable (Kendall & Gal, 2017) and is commonly estimated with Bayesian weight approximations (Blundell et al., 2015; Gal & Ghahramani, 2016; Maddox et al., 2019) or ensembles (Lakshminarayanan et al., 2017; Pearce et al., 2020). In diffusion and flow matching models, ensembling is a common approach to estimating uncertainty (Berry et al., 2024; Römer et al., 2026), while other methods use Bayesian hypernetworks (Krueger et al., 2017; Chan et al., 2024), disagreement between existing expert paths (Berry et al., 2025), or Laplace posteriors (Jazbec et al., 2025; Huang et al., 2025). Other uncertainty proxies include input sensitivity, velocity-field divergence and velocity variance (De Vita & Belagiannis, 2025; Xing et al., 2026; de Kruiff et al., 2026). We focus on uncertainty estimation from a single pre-trained flow policy.

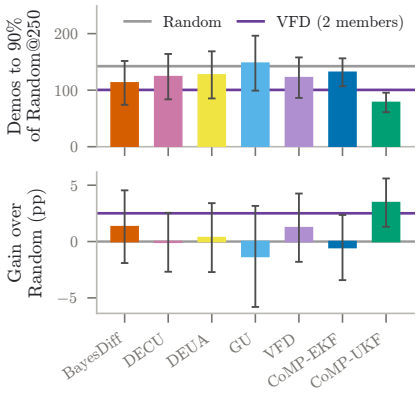

Figure 5: Data selection on LIBERO-10. Demonstrations needed to reach 90% of random's success at $X = 250$ (top) and mean gain over random (bottom). CoMP uses the augmented head; post-hoc baselines use the plain head.

Moment propagation instead carries moments through denoising. BayesDiff (Kou et al., 2024) applied it to diffusion models with a last-layer Laplace posterior (Kristiadi et al., 2020; Daxberger et al., 2021), followed by extensions to trajectory diffusion (Capellera et al., 2025) and flow matching (Han et al., 2026). However, these approximations have limited expressivity and are susceptible to feature collapse (van Amersfoort et al., 2021; Mukhoti et al., 2023). Extensions use broader parameter coverage (Gupta et al., 2026) or neural tangent kernel approximations (Deng et al., 2022; Calvo-Ordoñez et al., 2026), while LULA (Kristiadi et al., 2021) learns additional units without changing the model's predictions. CoMP addresses BayesDiff's variance shrinkage by retaining the state-weight cross-covariance, while our LULA-inspired widening reuses observation embeddings without policy re-training.

Uncertainty quantification helps robots anticipate failures and decide what to learn next. Runtime monitors use uncertainty or action consistency to flag failures early (Römer et al., 2025; Xu et al., 2025; Agia et al., 2024). To adapt policies with fewer demonstrations, uncertainty guides selection through information gain (Bagatella et al., 2025) or velocity-field disagreement (VFD) (Römer et al., 2026). Complementary criteria include influence functions (Agia et al., 2025), datamodels (Dass et al., 2026) and trajectory diversity (Sirigiri et al., 2026). CoMP supports these applications with uncertainty extracted from pre-trained foundation policies without re-training them.

6 Conclusion and Limitations

We present Coherent Moment Propagation (CoMP), an epistemic uncertainty estimator for pre-trained flow matching-based robot policies. We develop two novel uncertainty estimators, CoMP-EKF and CoMP-UKF, that propagate the state-weight cross-covariance through the ODE sampler. Through theoretical and empirical analysis, we show that BayesDiff underestimates uncertainty by omitting this cross-covariance. Specifically, its reported total variance is upper bounded by $O(1/T)$ in the number of Euler steps T, a phenomenon we call variance collapse: refining the solver drives uncertainty estimates to zero even as numerical accuracy of the solver improves. Inspired by LULA (Kristiadi et al., 2021), we further propose to augment the last-layer features with visual language embeddings readily computed by pre-trained policies, and show that the augmented head improves uncertainty estimators' responsiveness to policy input without re-training the policy. However, CoMP has two limitations: it relies on a linear last layer for analytic weight Jacobians, making deeper Laplace posteriors costly, and its state covariance storage scales quadratically with the output dimension. These limitations motivate low-rank covariance propagation and scalable posterior approximations beyond the last layer, which could broaden CoMP's applications, including to world action models. Despite these limitations, experiments across three simulated robotics benchmarks show that CoMP avoids BayesDiff's variance collapse and is on par with or better than established post-hoc uncertainty estimators on applications including failure detection and efficient fine-tuning.

AI USE STATEMENT

We used generative AI tools to polish the manuscript, write unit tests for our methods and baselines, create experiment launch scripts, check the consistency of mathematical notation, and improve figure presentation. We reviewed the AI-assisted material and take responsibility for the final manuscript.

REPRODUCIBILITY STATEMENT

We provide proofs of our theoretical results in Appendices A.1 and A.2, pseudocode in Appendix A.4, and experimental settings and implementation details in Appendices A.6–A.10. The code used to produce our results will be released publicly following publication.

REFERENCES

Christopher Agia, Rohan Sinha, Jingyun Yang, Ziang Cao, Rika Antonova, Marco Pavone, and Jeannette Bohg. Unpacking failure modes of generative policies: Runtime monitoring of consistency and progress. In *8th Annual Conference on Robot Learning*, 2024.

Christopher Agia, Rohan Sinha, Jingyun Yang, Rika Antonova, Marco Pavone, Haruki Nishimura, Masha Itkina, and Jeannette Bohg. Cupid: Curating data your robot loves with influence functions. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 2907–2932. PMLR, 27–30 Sep 2025.

Michael Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025.

Anastasios N. Angelopoulos and Stephen Bates. Conformal prediction: A gentle introduction. *Found. Trends Mach. Learn.*, 16(4):494–591, 2023. ISSN 1935-8237. doi: 10.1561/2200000101.

Marco Bagatella, Jonas Hübotter, Georg Martius, and Andreas Krause. Active fine-tuning of multi-task policies. In *Forty-second International Conference on Machine Learning*, 2025.

Joe Benton, George Deligiannidis, and Arnaud Doucet. Error bounds for flow matching methods. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

Lucas Berry, Axel Brando, and David Meger. Shedding light on large generative networks: Estimating epistemic uncertainty in diffusion models. In *The 40th Conference on Uncertainty in Artificial Intelligence*, 2024.

Lucas Berry, Axel Brando, Wei-Di Chang, Juan Camilo Gamboa Higuera, and David Meger. Emoe: Training-free expert disagreement for uncertainty-aware text-to-image diffusion. *arXiv preprint arXiv:2505.13273*, 2025.

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. PaliGemma: A versatile 3B VLM for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. GR00T N1 an open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, brian ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 27–30 Sep 2025a.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A Vision-Language-Action Flow Model for General Robot Control. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025b. doi: 10.15607/RSS.2025.XXI.010.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pp. 1613–1622. PMLR, 2015.

Sergio Calvo-Ordoñez, Jonathan Plenk, Richard Bergna, Alvaro Cartea, Yarin Gal, José Miguel Hernández-Lobato, and Kamil Ciosek. Richer bayesian last layers with subsampled NTK features. In *Forty-third International Conference on Machine Learning*, 2026.

Guillem Capellera, Antonio Rubio, Luis Ferraz, and Antonio Agudo. Unified uncertainty-aware diffusion for multi-agent trajectory modeling. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22476–22486. IEEE, 2025.

Matthew A Chan, Maria J Molina, and Christopher A Metzler. Estimating epistemic and aleatoric uncertainty with a single model. *Advances in neural information processing systems*, 37:109845–109870, 2024.

Sitan Chen, Sinho Chewi, Holden Lee, Yuanzhi Li, Jianfeng Lu, and Adil Salim. The probability flow ODE is provably fast. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a.

Sitan Chen, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, and Anru Zhang. Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions. In *The Eleventh International Conference on Learning Representations*, 2023b.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Shivin Dass, Alaa Khaddaj, Logan Engstrom, Aleksander Madry, Andrew Ilyas, and Roberto Martín-Martín. DataMIL: Selecting data for robot imitation learning with datamodels. In *The Fourteenth International Conference on Learning Representations*, 2026.

Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig. Laplace redux - effortless bayesian deep learning. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021.

Friso de Kruiff, Dario Coscia, Max Welling, and Erik Bekkers. Flowing with confidence. *arXiv preprint arXiv:2605.18472*, 2026.

Michele De Vita and Vasileios Belagiannis. Diffusion model guided sampling with pixel-wise aleatoric uncertainty estimation. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, pp. 3844–3854, February 2025.

Zhijie Deng, Feng Zhou, and Jun Zhu. Accelerated linearized laplace approximation for bayesian deep learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Qiao Gu, Yuanliang Ju, Shengxiang Sun, Igor Gilitschenski, Haruki Nishimura, Masha Itkina, and Florian Shkurti. Safe: Multitask failure detection for vision-language-action models. *Advances in Neural Information Processing Systems*, 38:40041–40076, 2026.

Aditi Gupta, Raphael A Meyer, Yotam Yaniv, Elynn Chen, and N. Benjamin Erichson. Quantifying epistemic uncertainty in diffusion models. In *The 29th International Conference on Artificial Intelligence and Statistics*, 2026.

E. Hairer, S.P. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I Nonstiff problems*. Springer, Berlin, second edition, 2000.

Juyeop Han, Lukas Lao Beyer, and Sertac Karaman. Flow matching with uncertainty quantification and guidance. *arXiv preprint arXiv:2602.10326*, 2026.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Yingsong Huang, Hui Guo, Jing Huang, Bing Bai, and Qi Xiong. Diffusion epistemic uncertainty with asymmetric learning for diffusion-generated image detection. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 17097–17107. IEEE, 2025.

Metod Jazbec, Eliot Wong-Toi, Guoxuan Xia, Dan Zhang, Eric Nalisnick, and Stephan Mandt. Generative uncertainty in diffusion models. In *The 41st Conference on Uncertainty in Artificial Intelligence*, 2025.

Felix Jimenez and Matthias Katzfuss. Probabilistic skip connections for deterministic uncertainty quantification in deep neural networks. *arXiv preprint arXiv:2501.04816*, 2025.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, Patrick Tree Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sungjae Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean Mercat, Abdul Rehman, Pannag R Sanketi, Archit Sharma, Cody Simpson, Quan Vuong, Homer Rich Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Z. Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Daphne Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan Paul Foster, Jensen Gao, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Donovon Jackson, Charlotte Le, Yunshuang Li, Roy Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Nguyen, Abigail O'Neill, Rosario Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Abhishek Gupta, Dinesh Jayaraman, Joseph J Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. DROID: A Large-Scale In-The-Wild Robot Manipulation Dataset. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.120.

Siqi Kou, Lei Gan, Dequan Wang, Chongxuan Li, and Zhijie Deng. BayesDiff: Estimating pixel-wise uncertainty in diffusion via bayesian inference. In *International Conference on Learning Representations*, volume 2024, pp. 17046–17063, 2024.

Agustinus Kristiadi, Matthias Hein, and Philipp Hennig. Being bayesian, even just a bit, fixes overconfidence in ReLU networks. In *International conference on machine learning*, pp. 5436–5446. PMLR, 2020.

Agustinus Kristiadi, Matthias Hein, and Philipp Hennig. Learnable uncertainty under laplace approximations. In *Uncertainty in Artificial Intelligence*, pp. 344–353. PMLR, 2021.

David Krueger, Chin-Wei Huang, Riashat Islam, Ryan Turner, Alexandre Lacoste, and Aaron Courville. Bayesian hypernetworks. In *Second Workshop on Bayesian Deep Learning at NeurIPS*, 2017.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, qiang liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023a.

Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023b.

Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. *Advances in neural information processing systems*, 32, 2019.

Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 1678–1690. PMLR, 08–11 Nov 2022.

Jishnu Mukhoti, Andreas Kirsch, Joost van Amersfoort, Philip H.S. Torr, and Yarin Gal. Deep deterministic uncertainty: A new simple baseline. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24384–24394, June 2023.

Tim Pearce, Felix Leibfried, and Alexandra Brintrup. Uncertainty in neural networks: Approximately bayesian ensembling. In *International conference on artificial intelligence and statistics*, pp. 234–244. PMLR, 2020.

Ralf Römer, Adrian Kobras, Luca Worbis, and Angela P. Schoellig. Failure prediction at runtime for generative robot policies. *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Ralf Römer, Maximilian Seeliger, Saida Liu, Ben Sturgis, Marco Bagatella, Daniel Marta, Andreas Krause, and Angela P. Schoellig. Uncertainty quantification for flow-based vision-language-action models. In *RSS 2026 Workshop on Diffusion for Robot Learning*, 2026.

Simo Särkkä. *Bayesian Filtering and Smoothing*. Cambridge University Press, USA, 2013. ISBN 1107619289.

Michael Scheuerer and Thomas M. Hamill. Variogram-based proper scoring rules for probabilistic forecasts of multivariate quantities. *Monthly Weather Review*, 143(4):1321–1334, 2015. doi: 10.1175/MWR-D-14-00269.1.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, et al. SmolVLA: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

Sreevardhan Sirigiri, Nathan Samuel de Lara, Christopher Agia, Florian Shkurti, and Fabio Ramos. Diversity you can actually measure: A fast, model-free diversity metric for robotics datasets. *arXiv preprint arXiv:2603.11634*, 2026.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

Joost van Amersfoort, Lewis Smith, Andrew Jesson, Oscar Key, and Yarin Gal. On feature collapse and deep kernel learning for single forward pass uncertainty. *arXiv preprint arXiv:2102.11409*, 2021.

Jiarui Xing, Song Wang, and Jian Wang. Divergence is uncertainty: A closed-form posterior covariance for flow matching. *arXiv preprint arXiv:2605.00941*, 2026.

Chen Xu, Tony Khuong Nguyen, Emma Dixon, Christopher Rodriguez, Patrick Miller, Robert Lee, Paarth Shah, Rares Andrei Ambrus, Haruki Nishimura, and Masha Itkina. Can We Detect Failures Without Failure Data? Uncertainty-Aware Runtime Failure Detection for Imitation Learning Policies. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.073.

Xuning Yang, Rishit Dagli, Alex Zook, Hugo Hadfield, Ankit Goyal, Stan Birchfield, Fabio Ramos, and Jonathan Tremblay. RoboLab: A High-Fidelity Simulation Benchmark for Analysis of Task Generalist Policies. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026. doi: 10.15607/RSS.2026.XXII.096.

A APPENDIX

A.1 VARIANCE COLLAPSE

This appendix proves Theorem 1. Proposition 1 shows that one denoising step of the recursion obeys the identity (9), Lemma 1 bounds the transport term in that identity, and the two combine into a covariance recursion that unrolls to the stated $O(1/T)$ upper bound.

Proposition 1 (Trace identity for one denoising step)**.** *Under the joint law of §2, the recursion* (8) *satisfies the identity* (9) *at every step i (re-stated here for convenience).*

$$\begin{aligned} \operatorname{tr} P_{i+1} &= \operatorname{tr} \operatorname{Cov}_x[x_i + \Delta t\, u_{\theta_{\mathrm{MAP}}}(x_i, t_i)] + \Delta t^2\, \bar{\sigma}_i^2 \\ &= \operatorname{tr} \operatorname{Cov}_x[h_i(x_i)] + \Delta t^2\, \bar{\sigma}_i^2. \end{aligned} \tag{9}$$

Proof. Taking the trace of (8) gives

$$\operatorname{tr} P_{i+1} = \operatorname{tr} P_i + 2\,\Delta t\ \operatorname{tr} \operatorname{Cov}(x_i, u_i) + \Delta t^2\ \operatorname{tr} \operatorname{Cov}(u_i), \tag{16}$$

and both velocity terms are handled by conditioning on the state. The subscript x denotes moments over x_i, omitting the step index for brevity.

The cross term. Conditioning on the state, the law of total covariance splits it in two,

$$\operatorname{Cov}(x_i, u_i) = \underbrace{\mathbb{E}_x[\operatorname{Cov}(x_i, u_i \mid x_i)]}_{=\,0} + \operatorname{Cov}_x(x_i,\, \mathbb{E}[u_i \mid x_i]) = \operatorname{Cov}_x(x_i,\, u_{\theta_{\mathrm{MAP}}}(x_i, t_i))\,, \tag{17}$$

the first term vanishes because x_i is deterministic given itself and the second collapses onto the MAP velocity under the joint law of §2, whose diagonal is exactly what the Monte Carlo estimator there targets.

The velocity variance. Conditioning on the state again, the same law splits it in two as well,[4]

$$\operatorname{tr} \operatorname{Cov}(u_i) = \underbrace{\operatorname{tr} \mathbb{E}_x[\operatorname{Cov}(u_i \mid x_i)]}_{=\,\bar{\sigma}_i^2} + \operatorname{tr} \operatorname{Cov}_x(\mathbb{E}[u_i \mid x_i]) = \bar{\sigma}_i^2 + \operatorname{tr} \operatorname{Cov}_x[u_{\theta_{\mathrm{MAP}}}(x_i, t_i)]\,, \tag{18}$$

the first term averages the predictive covariance of (6) over the belief and the second measures the spread of the MAP velocity as x_i varies under the current belief.

Recombining. Substituting (17) and (18) into (16) leaves $\Delta t^2 \bar{\sigma}_i^2$ alongside three terms involving the current state belief and MAP velocity, and bilinearity of the covariance collects those three into the Euler map $h_i(x_i) = x_i + \Delta t\, u_{\theta_{\mathrm{MAP}}}(x_i, t_i)$ of §3,

$$\operatorname{tr} P_i + 2\,\Delta t\ \operatorname{tr} \operatorname{Cov}_x(x_i,\, u_{\theta_{\mathrm{MAP}}}(x_i, t_i)) + \Delta t^2\ \operatorname{tr} \operatorname{Cov}_x[u_{\theta_{\mathrm{MAP}}}(x_i, t_i)] = \operatorname{tr} \operatorname{Cov}_x[h_i(x_i)]\,,$$

which is the transport term of (9). □

Lemma 1 (Transport barely grows the spread)**.** *Under (A1) the transport term of* (9) *obeys* $\operatorname{tr} \operatorname{Cov}_x[h_i(x_i)] \le (1 + \Delta t\, L)^2\ \operatorname{tr} P_i$.

Proof. For any states $x_i, y_i \in \mathbb{R}^D$, the triangle inequality and (A1) give

$$\|h_i(x_i) - h_i(y_i)\| \le \|x_i - y_i\| + \Delta t\ \|u_{\theta_{\mathrm{MAP}}}(x_i, t_i) - u_{\theta_{\mathrm{MAP}}}(y_i, t_i)\| \le (1 + \Delta t\, L)\, \|x_i - y_i\|\,.$$

Writing $m_i = \mathbb{E}_x[x_i]$, both $\mathbb{E}_x[h_i(x_i)]$ and $h_i(m_i)$ are deterministic quantities. Measuring squared distance from $h_i(m_i)$ instead of the transformed mean adds a nonnegative term:

$$\mathbb{E}_x \|h_i(x_i) - h_i(m_i)\|^2 = \operatorname{tr} \operatorname{Cov}_x[h_i(x_i)] + \|\mathbb{E}_x[h_i(x_i)] - h_i(m_i)\|^2\,.$$

Setting $y_i = m_i$ in the Lipschitz bound, squaring and taking expectations gives

$$\begin{aligned} \operatorname{tr} \operatorname{Cov}_x[h_i(x_i)] &\le \mathbb{E}_x \|h_i(x_i) - h_i(m_i)\|^2 \\ &\le (1 + \Delta t\, L)^2\, \mathbb{E}_x \|x_i - m_i\|^2 = (1 + \Delta t\, L)^2 \operatorname{tr} P_i. \quad \square \end{aligned}$$

[4] BayesDiff uses conditional predictive variance (Kou et al., 2024, after Eq. (8)). We retain both terms of (18); dropping the second, nonnegative term leaves our upper bound unchanged.

Proof of Theorem 1. Lemma 1 bounds the transport term of Proposition 1, while (A2) bounds the injection by $\Delta t^2 \bar{\sigma}^2_{\max}$, so (9) is dominated by the scalar recursion

$$\operatorname{tr} P_{i+1} \le \alpha \operatorname{tr} P_i + \beta, \qquad \alpha \triangleq (1 + \Delta t\, L)^2 \ge 1, \qquad \beta \triangleq \Delta t^2 \bar{\sigma}^2_{\max}. \tag{19}$$

Unrolling (19) from $\operatorname{tr} P_0 = 0$ leaves a geometric sum whose terms are each at most $\beta\alpha^T$, and substituting $\Delta t = 1/T$ gives:

$$\operatorname{tr} P_T \;\le\; \beta \sum_{i=0}^{T-1} \alpha^i \;\le\; T\,\beta\,\alpha^T \;=\; \frac{\bar{\sigma}^2_{\max}}{T}\left(1 + \frac{L}{T}\right)^{2T} \;\le\; e^{2L}\,\frac{\bar{\sigma}^2_{\max}}{T},$$

the last step because $1 + z \le e^z$ for every real z, which at $z = L/T$ raised to the T-th power gives $(1 + L/T)^T \le e^L$, and which is (10). □

Corollary 1 (Full covariance)**.** *Carrying the full covariance $P_{i+1} = \operatorname{Cov}(x_i + \Delta t\, u_i)$ in place of the diagonal of* (8) *leaves Theorem 1 unchanged.*

Proof. Expanding the full covariance of the Euler update (2) gives,

$$P_{i+1} = P_i + \Delta t \left[\operatorname{Cov}(x_i, u_i) + \operatorname{Cov}(u_i, x_i)\right] + \Delta t^2 \operatorname{Cov}(u_i), \tag{20}$$

and taking the trace, with $\operatorname{tr}\operatorname{Cov}(u_i, x_i) = \operatorname{tr}\operatorname{Cov}(x_i, u_i)^{\mathsf{T}} = \operatorname{tr}\operatorname{Cov}(x_i, u_i)$,

$$\operatorname{tr} P_{i+1} = \operatorname{tr} P_i + 2\,\Delta t\, \operatorname{tr}\operatorname{Cov}(x_i, u_i) + \Delta t^2\, \operatorname{tr}\operatorname{Cov}(u_i),$$

which is (16). Lemma 1 holds and (19) unrolls as before. □

Corollary 2 (Diffusion samplers)**.** *Let time run from noise at $t = 0$ to data at $t = 1$ as in §2. The reverse-time SDE through which BayesDiff propagates uncertainty (Song et al., 2021; Kou et al., 2024) then reads*

$$\mathrm{d}x_t = b_\theta(x_t, t)\,\mathrm{d}t + g(t)\,\mathrm{d}\omega_t, \qquad b_\theta(x, t) \triangleq -\Big[f(t)\,x + \frac{g(t)^2}{\sigma_t}\,\epsilon_\theta(x, t)\Big], \tag{21}$$

where f and g are the drift and diffusion coefficients respectively of the forward noising process, σ_t is the standard deviation of the noise at time t, ω_t is a Wiener process, and ϵ_θ is the noise prediction network, over whose last layer the Laplace posterior (4) *is placed. The drift b_θ plays the role of the velocity field in* (2) *and is affine in ϵ_θ, while the diffusion term $g(t)\,\mathrm{d}\omega_t$ does not depend on θ. We take f, g and g^2/σ_t bounded on $[0, 1]$, which holds when the sampled interval ends at a fixed positive distance from the point $\sigma_t = 0$, as done in continuous-time sampling (Song et al., 2021). Using the same integration steps as* (2)*, the Euler-Maruyama discretization of* (21) *gives*

$$x_{i+1} = x_i + \Delta t\, b_\theta(x_i, t_i) + \sqrt{\Delta t}\, g_i\, \eta_i, \qquad g_i \triangleq g(t_i), \qquad \eta_i \sim \mathcal{N}(0, I), \tag{22}$$

with η_i independent of x_i and θ, and the iteration rule of BayesDiff is the moment recursion (7)–(8) *of this step, written with the step size absorbed into the coefficients (Kou et al., 2024, Eq. 7). Under (A1) for $\epsilon_{\theta_{\mathrm{MAP}}}$, analogous to the Lipschitz assumption on the true score in (Chen et al., 2023b, Assumption 1), the MAP drift $b_{\theta_{\mathrm{MAP}}}$ is Lipschitz. Under (A2) for the noise predictor variance and the bounded schedule coefficients, the summed predictive variance of b_θ is also bounded. We again denote the resulting constants by L and $\bar{\sigma}^2_{\max}$, and the recursion reports*

$$\operatorname{tr} P_T \;\le\; e^{2L}\Bigg(\underbrace{\frac{\bar{\sigma}^2_{\max}}{T}}_{\textit{epistemic}} + \underbrace{D\,\Delta t \sum_{i=0}^{T-1} g_i^2}_{\textit{aleatoric}}\Bigg), \tag{23}$$

so the epistemic term in the upper bound decays as in Theorem 1, while the aleatoric term is $O(1)$.

Proof. Conditioned on x_i, the noise prediction has the Gaussian form of (6). In the drift (21), the term $-f(t_i)x_i$ is deterministic, so with $\sigma_i \triangleq \sigma_{t_i}$,

$$\operatorname{Cov}[b_\theta(x_i, t_i) \mid x_i] = \left(\frac{g_i^2}{\sigma_i}\right)^2 \operatorname{Cov}[\epsilon_\theta(x_i, t_i) \mid x_i] = \frac{g_i^4}{\sigma_i^2}\,\operatorname{diag}\gamma^2(x_i, t_i).$$

The drift is affine in the noise prediction, so its predictive distribution is Gaussian as in (6), with mean $b_{\theta_{\mathrm{MAP}}}(x_i, t_i)$:

$$p(b \mid x_i, t_i, \mathcal{D}) \approx \mathcal{N}\Big(b;\, b_{\theta_{\mathrm{MAP}}}(x_i, t_i),\, \frac{g_i^4}{\sigma_i^2}\, \mathrm{diag}\, \gamma^2(x_i, t_i)\Big). \tag{24}$$

Without the Brownian increment, the step (22) is the Euler step (2) with b_θ in place of u_θ. Proposition 1 therefore applies with (24) in place of (6), and Lemma 1 bounds the transport term,

$$\begin{aligned}\mathrm{tr}\,\mathrm{Cov}[x_i + \Delta t\, b_\theta(x_i, t_i)] &= \mathrm{tr}\,\mathrm{Cov}_x[x_i + \Delta t\, b_{\theta_{\mathrm{MAP}}}(x_i, t_i)] + \Delta t^2\, \bar\sigma_i^2 \\ &\le (1 + \Delta t\, L)^2\ \mathrm{tr}\, P_i + \Delta t^2\, \bar\sigma_{\max}^2,\end{aligned}$$

where $\bar\sigma_i^2 \triangleq \mathbb{E}_x\big[\frac{g_i^4}{\sigma_i^2} \sum_{d=1}^D \gamma_d^2(x_i, t_i)\big]$ is the variance of (24) summed over coordinates and averaged over the belief, the counterpart of the $\bar\sigma_i^2$ of (9), which is bounded by $\bar\sigma_{\max}^2$. Since the aleatoric increment $\sqrt{\Delta t}\, g_i \eta_i$ is zero-mean and independent of (x_i, θ), it adds its covariance without a cross term, $P_{i+1} = \mathrm{Cov}[x_i + \Delta t\, b_\theta(x_i, t_i)] + g_i^2 \Delta t\, I$, and (9) becomes

$$\mathrm{tr}\, P_{i+1} = \mathrm{tr}\,\mathrm{Cov}_x[x_i + \Delta t\, b_{\theta_{\mathrm{MAP}}}(x_i, t_i)] + \Delta t^2\, \bar\sigma_i^2 + D\, g_i^2\, \Delta t\ \le\ \alpha\ \mathrm{tr}\, P_i + \beta + D\, g_i^2\, \Delta t,$$

with α and β as in (19). Unrolling from $\mathrm{tr}\, P_0 = 0$ as in the proof of Theorem 1, with every power of α bounded by $\alpha^T \le e^{2L}$, gives

$$\mathrm{tr}\, P_T\ \le\ e^{2L} \sum_{i=0}^{T-1} \big(\beta + D\, g_i^2\, \Delta t\big)\ =\ e^{2L} \left(\frac{\bar\sigma_{\max}^2}{T} + D\, \Delta t \sum_{i=0}^{T-1} g_i^2 \right),$$

which is (23). The two sources of randomness enter (22) at different orders in the step size: the weight deviation through $\Delta t\, b_\theta$, with a variance of order Δt^2 per step and an $O(1/T)$ term in the bound, and the Brownian increment through $\sqrt{\Delta t}\, g_i \eta_i$, with a variance of order Δt per step and an $O(1)$ term in the bound. The ancestral sampler of DDPM (Ho et al., 2020) is a discretization of this form for the variance-preserving schedule (Song et al., 2021), and deterministic samplers of the probability-flow ODE drop the increment, halve the score term, and inherit the bound of Theorem 1 for their drift. □

A.2 Upper bound independent of integration steps

This appendix proves Theorem 2. Proposition 2 unrolls the recursion into a sum over the integration steps, and bounding that sum under (A1) and (A2) gives the two upper bounds.

Proposition 2 (The chain unrolls into one sum)**.** *Run from* $(x_0, 0, 0)$*, the recursion* (12)–(14) *reports*

$$P_T = M_T\, \Sigma_\theta\, M_T^\mathsf{T}, \qquad M_T \triangleq \Delta t \sum_{i=0}^{T-1} \Phi_i\, J_i^\theta, \qquad \Phi_i \triangleq A_{T-1} \cdots A_{i+1}, \tag{25}$$

with $\Phi_{T-1} = I$*, while the same recursion with* C_i *held at zero reports*

$$P_T = \Delta t^2 \sum_{i=0}^{T-1} \Phi_i\, J_i^\theta\, \Sigma_\theta\, J_i^{\theta\mathsf{T}}\, \Phi_i^\mathsf{T}. \tag{26}$$

Proof. Recall that $z_i = (\delta x_i, \delta\theta)$ stacks the state deviation $\delta x_i = x_i - m_i$ and the fixed weight deviation. Its covariance and transition matrix are

$$\mathrm{Cov}(z_i) = \begin{pmatrix} P_i & C_i \\ C_i^\mathsf{T} & \Sigma_\theta \end{pmatrix}, \qquad F_i = \begin{pmatrix} A_i & B_i \\ 0 & I \end{pmatrix}.$$

The upper-left block $P_i = \mathrm{Cov}(\delta x_i)$ is the state covariance, and the upper-right block $C_i = \mathrm{Cov}(\delta x_i, \delta\theta)$ captures its dependence on the weights. The joint update $\mathrm{Cov}(z_{i+1}) = F_i \mathrm{Cov}(z_i) F_i^\mathsf{T}$ gives (13)–(14) for these two blocks. Starting from $P_0 = C_0 = 0$, we unroll the recursion with and without the cross-covariance below.

With the cross-covariance. Applying (11) from $\delta x_0 = 0$ injects $B_i \delta\theta$ at step i and multiplies it by every A that follows, the same $\delta\theta$ appearing in all T of them, so the endpoint collects

$$\delta x_T = \sum_{i=0}^{T-1} A_{T-1} \cdots A_{i+1} \, B_i \, \delta\theta = \Delta t \sum_{i=0}^{T-1} \Phi_i \, J_i^\theta \, \delta\theta = M_T \, \delta\theta,$$

using $B_i = \Delta t \, J_i^\theta$, and since $\delta\theta \sim \mathcal{N}(0, \Sigma_\theta)$ its covariance is (25).

Without it. Holding $C_i \equiv 0$ leaves $P_{i+1} = A_i P_i A_i^\mathsf{T} + B_i \Sigma_\theta B_i^\mathsf{T}$, the transported state covariance and a weight contribution injected afresh, with no term relating the two. The recursion (8) of §2 carries no such term either, the weights having been marginalised out of the velocity field (6) before the step is taken, and we attribute the decay of Theorem 1 to that omission. Particularly, Kou et al. (2024) assume $\mathrm{Cov}(x_s, \epsilon_t) = 0$ for $s \neq t$, where ϵ_t is their noise prediction (Appendix A.2, after Eq. 32). Unrolling from $P_0 = 0$ transports the injection $B_i \Sigma_\theta B_i^\mathsf{T}$ through the same steps that follow,

$$P_T = \sum_{i=0}^{T-1} \Phi_i \, B_i \Sigma_\theta B_i^\mathsf{T} \, \Phi_i^\mathsf{T} = \Delta t^2 \sum_{i=0}^{T-1} \Phi_i \, J_i^\theta \, \Sigma_\theta \, J_i^{\theta\mathsf{T}} \, \Phi_i^\mathsf{T},$$

which is (26), the identical T contributions with their cross terms removed. □

Proof of Theorem 2. Both upper bounds follow from the same two bounds on a single summand. Under (A1) each factor of the transport obeys $\|A_i\| \leq 1 + \Delta t \, L$, so the product of at most T of them obeys $\|\Phi_i\| \leq (1 + \Delta t \, L)^T \leq e^L$. Under (A2), writing $\Sigma_\theta^{1/2}$ for the symmetric square root of Σ_θ gives

$$\|J_i^\theta \Sigma_\theta^{1/2}\|_F^2 = \mathrm{tr}[J_i^\theta \Sigma_\theta J_i^{\theta\mathsf{T}}] = \sum_d \gamma_d^2(m_i, t_i) \leq \bar{\sigma}_{\max}^2.$$

Taking square roots yields $\|J_i^\theta \Sigma_\theta^{1/2}\|_F \leq \bar{\sigma}_{\max}$. Here $\|\cdot\|$ denotes the spectral norm and $\|\cdot\|_F$ the Frobenius norm. Taking the trace of (25) gives $\mathrm{tr}\, P_T = \|M_T \Sigma_\theta^{1/2}\|_F^2$. Substituting $M_T = \Delta t \sum_i \Phi_i J_i^\theta$, the triangle inequality bounds the norm of the sum by the sum of the norms. We then apply $\|AB\|_F \leq \|A\| \, \|B\|_F$ to each term, separating Φ_i from $J_i^\theta \Sigma_\theta^{1/2}$, and use the bounds established above:

$$\begin{aligned}
\mathrm{tr}\, P_T = \left\| M_T \, \Sigma_\theta^{1/2} \right\|_F^2 \;&=\; \Delta t^2 \left\| \sum_{i=0}^{T-1} \Phi_i \, J_i^\theta \, \Sigma_\theta^{1/2} \right\|_F^2 \\
&\leq \left(\Delta t \sum_{i=0}^{T-1} \left\| \Phi_i \, J_i^\theta \, \Sigma_\theta^{1/2} \right\|_F \right)^2 \;\leq\; \left(\Delta t \sum_{i=0}^{T-1} \underbrace{\|\Phi_i\|}_{\leq e^L} \underbrace{\left\| J_i^\theta \, \Sigma_\theta^{1/2} \right\|_F}_{\leq \bar{\sigma}_{\max}} \right)^2 \\
&\leq \left(\Delta t \, T \, e^L \, \bar{\sigma}_{\max} \right)^2 \;=\; e^{2L} \, \bar{\sigma}_{\max}^2,
\end{aligned}$$

where the last equality uses $\Delta t \, T = 1$, giving (15).

Taking the trace of (26) gives a sum of squared Frobenius norms. Applying the same bounds to each term,

$$\begin{aligned}
\mathrm{tr}\, P_T &= \Delta t^2 \sum_{i=0}^{T-1} \|\Phi_i J_i^\theta \Sigma_\theta^{1/2}\|_F^2 \leq \Delta t^2 \sum_{i=0}^{T-1} \|\Phi_i\|^2 \, \|J_i^\theta \Sigma_\theta^{1/2}\|_F^2 \\
&\leq T \, \Delta t^2 \, e^{2L} \bar{\sigma}_{\max}^2 = e^{2L} \frac{\bar{\sigma}_{\max}^2}{T},
\end{aligned}$$

where $T \Delta t^2 = 1/T$, giving (10).

□

A.3 Supplementary Details: Augmented Head

For a linear last layer under the flow matching loss, the curvature in (4) is the Gram matrix of the last-layer input over the fit set $\mathcal{D}_{\text{fit}}$ plus λI, so widening the input from ϕ to ψ of (5) changes the Laplace posterior while preserving the policy's action predictions. Here, $\zeta = c(o)$ contains the observation embeddings already computed by the policy. Writing Σ_d for the block of Σ_θ that belongs to the d-th row of θ, the variance (6) of the d-th velocity coordinate expands as

$$\gamma_d^2(x,t) = \psi^\mathsf{T}\Sigma_d\,\psi = \begin{pmatrix}\phi\\ \zeta\end{pmatrix}^\mathsf{T}\begin{pmatrix}\Sigma_d^{\phi\phi} & \Sigma_d^{\phi\zeta}\\ \Sigma_d^{\zeta\phi} & \Sigma_d^{\zeta\zeta}\end{pmatrix}\begin{pmatrix}\phi\\ \zeta\end{pmatrix} = \phi^\mathsf{T}\Sigma_d^{\phi\phi}\,\phi + 2\,\phi^\mathsf{T}\Sigma_d^{\phi\zeta}\,\zeta + \zeta^\mathsf{T}\Sigma_d^{\zeta\zeta}\,\zeta, \tag{27}$$

where the first term uses the original features under the augmented posterior; the remaining terms involve the appended observation embeddings. Write $\theta_d = (\theta_d^\phi, \theta_d^\zeta)$, separating the weights on the original features ϕ from those on the appended observation embeddings ζ. To compare the two Laplace heads' predictive variances, we recover the plain head by fixing the appended weights θ_d^ζ to zero. With the same fit data and prior, its weight covariance is the conditional covariance under the augmented posterior:

$$\mathrm{Cov}(\theta_d^\phi \mid \theta_d^\zeta = 0) = (\Lambda_d^{\phi\phi})^{-1}, \qquad \Lambda_d \triangleq \Sigma_d^{-1}, \tag{28}$$

where $\Lambda_d^{\phi\phi}$ is the Gram matrix of ϕ plus λI, which is the precision of the plain Laplace head, and the conditional covariance of a Gaussian does not depend on the value conditioned on. The law of total variance then gives

$$\begin{aligned}\psi^\mathsf{T}\Sigma_d\,\psi &= \mathbb{E}\big[\,\mathrm{Var}(\psi^\mathsf{T}\theta_d \mid \theta_d^\zeta)\big] + \mathrm{Var}\left(\mathbb{E}[\psi^\mathsf{T}\theta_d \mid \theta_d^\zeta]\right)\\ &= \phi^\mathsf{T}(\Lambda_d^{\phi\phi})^{-1}\phi + \mathrm{Var}\left(\mathbb{E}[\psi^\mathsf{T}\theta_d \mid \theta_d^\zeta]\right) \;\geq\; \phi^\mathsf{T}(\Lambda_d^{\phi\phi})^{-1}\phi,\end{aligned} \tag{29}$$

where the right-hand side is the variance (6) of the plain head, so the augmented head's predictive velocity variance is no smaller than the plain head's at any input. Proposition 2 of Kristiadi et al. (2021) states this for a diagonal Laplace, and the argument holds equally for the Kronecker-factored posterior we use, whose input factor is the Gram matrix of ψ. Like LULA Kristiadi et al. (2021), our augmented head aims to alleviate feature collapse by widening the last-layer features, here using existing observation embeddings. Related approaches enrich Bayesian last layers with projected neural tangent kernel features (Calvo-Ordoñez et al., 2026) or fit probabilistic predictors to intermediate representations (Jimenez & Katzfuss, 2025).

A.4 Supplementary Details: Pseudocode

Algorithms 1 and 2 list the CoMP-EKF and CoMP-UKF recursions of §3 step by step; they differ in the propagated mean, transition A_i and transported state covariance.

CoMP-UKF moment updates. The state transition A_i in (11) and transported spread $A_i P_i A_i^\mathsf{T}$ in (14) can also be estimated from deterministic points instead of derivatives. The unscented transform (Särkkä, 2013) places $2D+1$ sigma points χ_j around m_i along the columns of a square root of P_i, with mean weights $w_j^{(m)}$ and covariance weights $w_j^{(c)}$ chosen so that the weighted points carry the mean and the covariance of the belief. Pushing them through the map, $y_j = h_i(\chi_j)$ with $\bar{y} \triangleq \sum_j w_j^{(m)} y_j$, gives

$$\Sigma_i^{yy} = \sum\nolimits_j w_j^{(c)}\,(y_j - \bar{y})(y_j - \bar{y})^\mathsf{T}, \qquad \Sigma_i^{xy} = \sum\nolimits_j w_j^{(c)}\,(\chi_j - m_i)(y_j - \bar{y})^\mathsf{T}, \tag{30}$$

which estimate the covariance of the transported state and its covariance with x_i. Two substitutions close the recursion: the transported spread $A_i P_i A_i^\mathsf{T}$ in (14) becomes Σ_i^{yy}, retaining curvature beyond the linear approximation, while elsewhere we use $A_i = (\Sigma_i^{xy})^\mathsf{T} P_i^{-1}$, the least-squares affine surrogate of h_i over the current belief (Särkkä, 2013). The mean advances as $m_{i+1} = \bar{y}$ and the weight terms are computed analytically using J_i^θ evaluated at m_i, so no sigma points are needed for θ. Algorithm 2 computes A_i by solving the regularized system $(P_i + \varepsilon_i I)A_i^\mathsf{T} = \Sigma_i^{xy}$, where $\varepsilon_i > 0$ is a small regularization parameter that keeps the solve defined even at $P_0 = 0$.

Algorithm 1 Coherent Moment Propagation with the extended Kalman filter (CoMP-EKF)

Require: sample $x_0 \sim \mathcal{N}(0, I)$, steps T with $\Delta t = 1/T$, MAP velocity $u_{\theta_{\mathrm{MAP}}}$ with features ϕ, last-layer posterior Σ_θ
Output: sample m_T and covariance P_T

Initialise $m_0 \leftarrow x_0$, $P_0 \leftarrow \mathbf{0}$, $C_0 \leftarrow \mathbf{0}$
for $i = 0 \to T-1$ **do**
  $J_i^x \leftarrow \partial u_{\theta_{\mathrm{MAP}}}(x_i, t_i)/\partial x_i$ at $x_i = m_i$
  $A_i \leftarrow I + \Delta t\, J_i^x$ and $B_i \leftarrow \Delta t\, J_i^\theta$ (analytic J_i^θ from ϕ for linear last-layer)
  $m_{i+1} \leftarrow m_i + \Delta t\, u_{\theta_{\mathrm{MAP}}}(m_i, t_i)$
  $C_{i+1} \leftarrow A_i C_i + B_i \Sigma_\theta$
  $P_{i+1} \leftarrow A_i P_i A_i^\mathsf{T} + A_i C_i B_i^\mathsf{T} + B_i C_i^\mathsf{T} A_i^\mathsf{T} + B_i \Sigma_\theta B_i^\mathsf{T}$
end for
return $\mathcal{N}(m_T, P_T)$

Algorithm 2 Coherent Moment Propagation with the unscented Kalman filter (CoMP-UKF)

Require: sample $x_0 \sim \mathcal{N}(0, I)$, steps T with $\Delta t = 1/T$, MAP velocity $u_{\theta_{\mathrm{MAP}}}$ with features ϕ, last-layer posterior Σ_θ
Output: sample m_T and covariance P_T

Initialise $m_0 \leftarrow x_0$, $P_0 \leftarrow \mathbf{0}$, $C_0 \leftarrow \mathbf{0}$
for $i = 0 \to T-1$ **do**
  Compute the $2D+1$ sigma points χ_j and weights $w_j^{(m)}, w_j^{(c)}$ carrying (m_i, P_i) by the unscented transform
  $y_j \leftarrow \chi_j + \Delta t\, u_{\theta_{\mathrm{MAP}}}(\chi_j, t_i)$ and $\bar{y} \leftarrow \sum_j w_j^{(m)} y_j$
  Σ_i^{yy} and Σ_i^{xy} from (30), then solve $(P_i + \varepsilon_i I) A_i^\mathsf{T} = \Sigma_i^{xy}$ for A_i and $B_i \leftarrow \Delta t\, J_i^\theta$ (closed-form B_i)
  $m_{i+1} \leftarrow \bar{y}$
  $C_{i+1} \leftarrow A_i C_i + B_i \Sigma_\theta$
  $P_{i+1} \leftarrow \Sigma_i^{yy} + A_i C_i B_i^\mathsf{T} + B_i C_i^\mathsf{T} A_i^\mathsf{T} + B_i \Sigma_\theta B_i^\mathsf{T}$
end for
return $\mathcal{N}(m_T, P_T)$

A.5 Supplementary Details: Uncertainty Scores

Algorithms 1 and 2 return the full covariance P_T of the predicted action chunk (3). We summarize it with two scalar uncertainty scores:

$$U_{\mathrm{chunk}} \triangleq \operatorname{tr} P_T, \qquad U_{\mathrm{endpoint}} \triangleq \operatorname{tr} \operatorname{Cov}\left(\sum_{k=1}^{H} a_k\right). \tag{31}$$

where H is the number of actions in the predicted chunk. The chunk score gives the total variance, summing the variances of all action coordinates. In contrast, the endpoint score also includes contributions from the off-diagonal blocks of P_T that couple commands at different steps.

The covariance of the accumulated commands can be written in terms of the blocks of P_T as

$$\Sigma_H \triangleq \operatorname{Cov}\left(\sum_{k=1}^{H} a_k\right) = \sum_{k=1}^{H}\sum_{k'=1}^{H} [P_T]_{kk'}, \qquad U_{\mathrm{endpoint}} = \operatorname{tr} \Sigma_H,$$

where $[P_T]_{kk'}$ is the covariance block coupling actions k and k'. Summing the commands treats the chunk as one accumulated action, whose uncertainty depends on both the individual command variances and their covariance across steps. We use $U = U_{\mathrm{chunk}}$ for direct comparison with BayesDiff in §4.1 and $U = U_{\mathrm{endpoint}}$ in all manipulation applications (§4.2, §4.3 and §4.4).

A.6 Experimental Settings

Table 1 summarizes the settings used for the baselines and our methods across experiments. Experiment-specific fitting, sampling and evaluation protocols are given in Appendices A.7–A.10.

Table 1: Settings of the uncertainty estimators. Baseline settings follow the released implementations, with adaptations to flow policies and experiment-specific sampling budgets. For each policy, we evaluate the post-hoc estimators with both the plain and augmented last-layer Laplace posteriors. K counts candidate chunks from independent initial noises per observation, M posterior weight draws (trained members for ensemble references), and S state particles; sampling budgets are given with each experiment.

Estimator	**Settings**	**Score** U
Common setup for post-hoc estimators		
Posterior	Kronecker-factored Laplace of the last layer, prior precision $\lambda = 1$; plain head on ϕ, augmented head on $\psi = (\phi, c(o))$; Σ_θ fixed during integration	
Sampler	Euler integration with $K = 1$ except for VFD; step counts, ablations and VFD candidate budgets are specified per experiment	
BayesDiff (Kou et al., 2024)	Diagonal moment recursion (7)–(8); state-velocity cross-covariance estimated from S state particles per step	$\operatorname{tr} P_T$
DEUA (Huang et al., 2025)	MAP sample re-noised to $\tau = 0.98$, one moment step to $\tau = 0.99$ ($\Delta t = 0.01$); velocity variance over the M draws at that state	$\Delta t^2 \sum_d \operatorname{Var}_\theta[u_d]$
GU (Jazbec et al., 2025)	M draws plus the MAP member integrated to $t = 1$; Gaussian entropy of the per-coordinate variance of the terminal state, floor $\sigma_0^2 = 10^{-8}$	$\frac{1}{2}\sum_d \log(\operatorname{var}_d + \sigma_0^2) + \frac{D}{2}\left[1 + \log(2\pi)\right]$
DECU (Berry et al., 2024)	Flow adaptation following SAVE (Römer et al., 2026): shared MAP trunk to the branching time $t = 0.995$, then one velocity $u^{(i)}$ for each of the M members; PaiDE mutual-information proxy with $D_{ij} = \|u^{(i)} - u^{(j)}\|^2$	$-\frac{1}{M}\sum_i \log\left(\frac{\sum_j e^{-D_{ij}}}{M}\right)$
VFD (Römer et al., 2026)	One reference member's path for each of the K candidates; V_ℓ sums the across-member velocity variances (normalized by the number of members) over the chunk and averages over the scored candidates; $t_\ell \in \{0, 0.1, \ldots, 0.9, 0.95\}$ and $\Delta t_\ell = t_{\ell+1} - t_\ell$, with the last interval ending at 1; members are the M draws, or two trained policies for the ensemble reference	$\sum_\ell \Delta t_\ell \, \frac{t_\ell}{1-t_\ell} \, V_\ell$
CoMP-EKF (ours)	Recursion (12)–(14) with exact J_i^x from batched reverse-mode autodiff; analytic J_i^θ and covariance products with Σ_θ	U_{chunk} or U_{endpoint} (Appendix A.5)
CoMP-UKF (ours)	$2D + 1$ state sigma points evaluated in batches through the action expert; regularized covariance solve for A_i (Appendix A.4); weight terms analytic in J_i^θ	U_{chunk} or U_{endpoint} (Appendix A.5)

A.7 Supplementary Details and Results: Faithfulness of the Propagated Covariance

For each task of §4.1, we train a 29M-parameter transformer variant of Diffusion Policy (Chi et al., 2025) with a ResNet (He et al., 2016) image encoder on the 200 proficient-human demonstrations, using the flow matching objective (1). Training runs for 200 epochs with AdamW at learning rate 10^{-4} and batch size 2048 for *Lift* and 1024 for the other tasks. The two-member ensembles of §4.3 use the same training recipe. Each policy predicts $H = 16$ actions, integrated for $T = 50$ Euler steps.

The plain head is the Kronecker-factored Laplace of the last layer on the training split of each policy. Each policy uses a history of two observations, combining proprioception with external and wrist-

camera images. The augmented head appends the two observation tokens formed from ResNet features and proprioception to condition the transformer, totaling 504 units, to the 252 last-layer features and is fit on the same windows. One robomimic policy trains for 200 epochs in 8 to 49 minutes on an RTX 5090 depending on the task, whereas the Kronecker-factored Laplace fit over the same training windows takes under half a minute for either head.

Reference and estimators. The 150 observation windows per task of §4.1 are drawn at random from the demonstration dataset. For each window, all estimators start from the same initial noise sample x_0. We construct the Gaussian reference using $M = 6000$ posterior draws integrated from x_0. BayesDiff uses $S = 24$ state particles and $M = 128$ posterior weight draws per step to estimate the velocity moments; its full-covariance ablation uses $S = 512$ particles per step. CoMP-EKF uses the exact state Jacobian and CoMP-UKF uses $2D+1 = 225$ state sigma points, evaluated in a single batch through the transformer and its output layer.

Metrics. We compare empirical pairwise moments using the variogram discrepancy

$$D_{\mathrm{VG}} \triangleq \sum_{d,e=1}^{D} \left[\frac{1}{N} \sum_{n=1}^{N} |z_d^{(n)} - z_e^{(n)}|^{1/2} - \frac{1}{N} \sum_{n=1}^{N} |y_d^{(n)} - y_e^{(n)}|^{1/2} \right]^2, \tag{32}$$

where $z^{(n)}$ are samples from the estimator's Gaussian and $y^{(n)}$ are held-out action chunks from the reference ensemble, with $N = 500$ for each. We report this discrepancy relative to that of the reference Gaussian on the same held-out chunks. The expert-action negative log-likelihood evaluates the ground-truth action chunk from each demonstration under each estimator's Gaussian; the ensemble reference for this metric uses $M = 1000$ posterior weight draws. Each bar of Figure 2 is the median over the windows with a bootstrap 95% interval of the median.

Additional results. Figure 6 repeats the study of §4.1 with the augmented head for all estimators. Both CoMP variants remain close to the posterior reference in total variance and variogram discrepancy, while both BayesDiff variants still underestimate spread. Compared with the plain head (Figure 2), the augmented Laplace head lowers expert-action NLL for both CoMP variants on all three tasks.

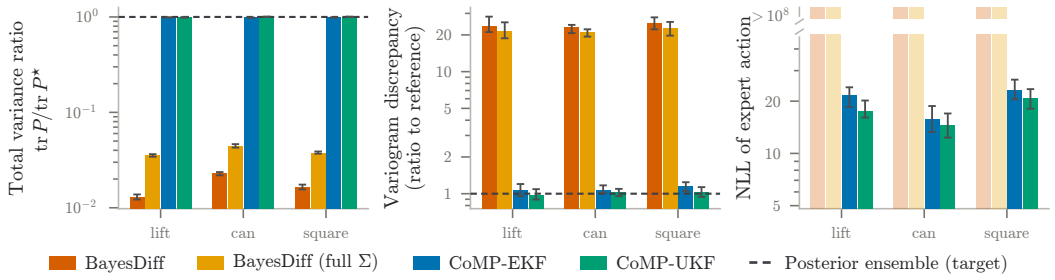

Figure 6: Calibration with the augmented head on robomimic. All estimators use the augmented last-layer Laplace posterior. Bars show medians over 150 observation windows per task with bootstrap 95% intervals; dashed lines mark reference values estimated from the augmented posterior. Both BayesDiff variants have expert-action NLLs above 10^8.

A.8 Supplementary Details and Results: Response of the Augmented Head

Policy and augmented head. We use openpi's $\pi_{0.5}$ fine-tuned on DROID with joint-position actions (release `pi05_droid_jointpos`). For $\pi_{0.5}$, we form $c(o)$ by separately averaging the image tokens of each of the two camera views and the language tokens of its PaliGemma vision-language backbone (Beyer et al., 2024). Appending these three 2048-dimensional averages to the action expert's 1024 last-layer features gives 7168 features in total. The augmented head increases posterior storage from 4 MB to 206 MB, less than 3% of $\pi_{0.5}$'s 7.2 GB of weights.

Fitting and evaluation. For each of the ten tasks, we run 100 episodes per wording on paired scene seeds to measure task success. The policy predicts and executes a chunk of $H = 15$ actions per decision at $T = 10$ Euler steps. Each Laplace posterior is fit once for all ten tasks, using up to ten successful rollouts per task selected in order from episodes 0 to 19 of the "specific" wording. We use an equally weighted mixture of about 2.2k observation frames from these RoboLab rollouts and 50k DROID frames for the posterior fit. The conditioning features are standardized per dimension using DROID statistics.

We evaluate uncertainty at the first decision of held-out episodes 20 to 99. The sampling estimators use $M = 16$ posterior weight draws, with VFD averaging $K = 5$ action candidates. BayesDiff uses $S = 10$ state particles per step. Figure 7 shows the ten scenes, and Table 2 lists their instruction wordings.

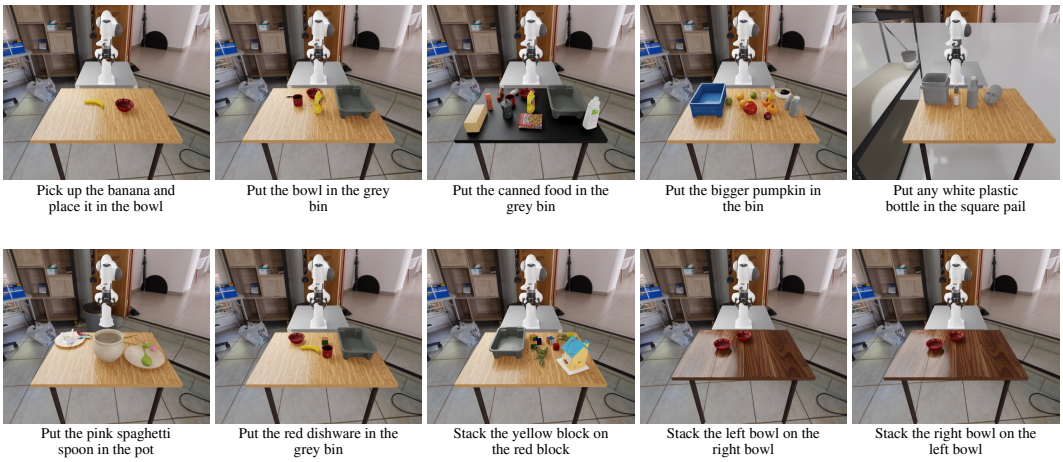

Figure 7: The ten RoboLab tasks of §4.2. Initial scene of each task from the front camera view, with its *default* instruction; the study runs every task under three wordings (specific, default and vague) from the same scenes.

Table 2: Instruction wordings for the ten RoboLab tasks. Each task is evaluated under specific, default and vague instructions from the same initial configurations in §4.2. Note how language instructions become more ambiguous from left to right.

Task	Language instruction wording		
	Specific	Default	Vague
banana in bowl	Grasp the yellow banana and place it inside the red bowl on the table	Pick up the banana and place it in the bowl	Put the fruit in the bowl
pumpkin in bin	Identify the larger pumpkin of the two pumpkins on the table and place it in the bin	Put the bigger pumpkin in the bin	Put the big pumpkin away
bowl in bin	Pick up the bowl from the table and place it inside the grey bin	put the bowl in the grey bin	put away bowl
stack L on R	Pick up the bowl to the left of the robot and place it on top of the bowl to the right of the robot	Stack the left bowl on the right bowl	Stack the left bowl on the right
stack R on L	Pick up the bowl to the right of the robot and place it on top of the bowl to the left of the robot	Stack the right bowl on the left bowl	Stack the right bowl on the left
canned food in bin	Put the small canned tuna in the grey bin	Put the canned food in the grey bin	Put away cans
bottle in pail	Pick up one of the three white plastic bottles and place it inside the square pail	Put any white plastic bottle in the square pail	Put a bottle in the square pail
spoon in pot	Pick up the pink spaghetti serving spoon that's on the left plate and place it inside the pot in the center of the table	Put the pink spaghetti spoon in the pot	Put the pink spoon in the pot
red dishes in bin	Pick up each red mug and bowl from the table and place them in the grey bin	Put the red dishware in the grey bin	Clean up red dishware
stack yellow	Pick up the yellow block on the left and place it precisely on top of the red block in front of the birdhouse	Stack the yellow block on the red block	Stack blocks: yellow on red

Additional results. Figure 8 shows the per-task results underlying Figure 3, for both the plain and augmented Laplace posteriors. With the plain head, uncertainty estimates remain mostly flat even as language instructions become more ambiguous. In contrast, the augmented head strengthens the response to these changes for every estimator except DECU.

Although the augmented head makes BayesDiff uncertainty estimates change with language ambiguity, it does not resolve the variance collapse issue outlined in §3. Figure 9 compares BayesDiff, CoMP-EKF and CoMP-UKF at 10, 25 and 50 Euler integration steps on the same recorded decisions. As the step count grows, BayesDiff's uncertainty shrinks and its response to instruction ambiguity weakens, while both CoMP variants retain their sensitivity. The default 10-step sampler of $\pi_{0.5}$ therefore makes this limitation less apparent, but increasing the number of integration steps reveals it even with the augmented head.

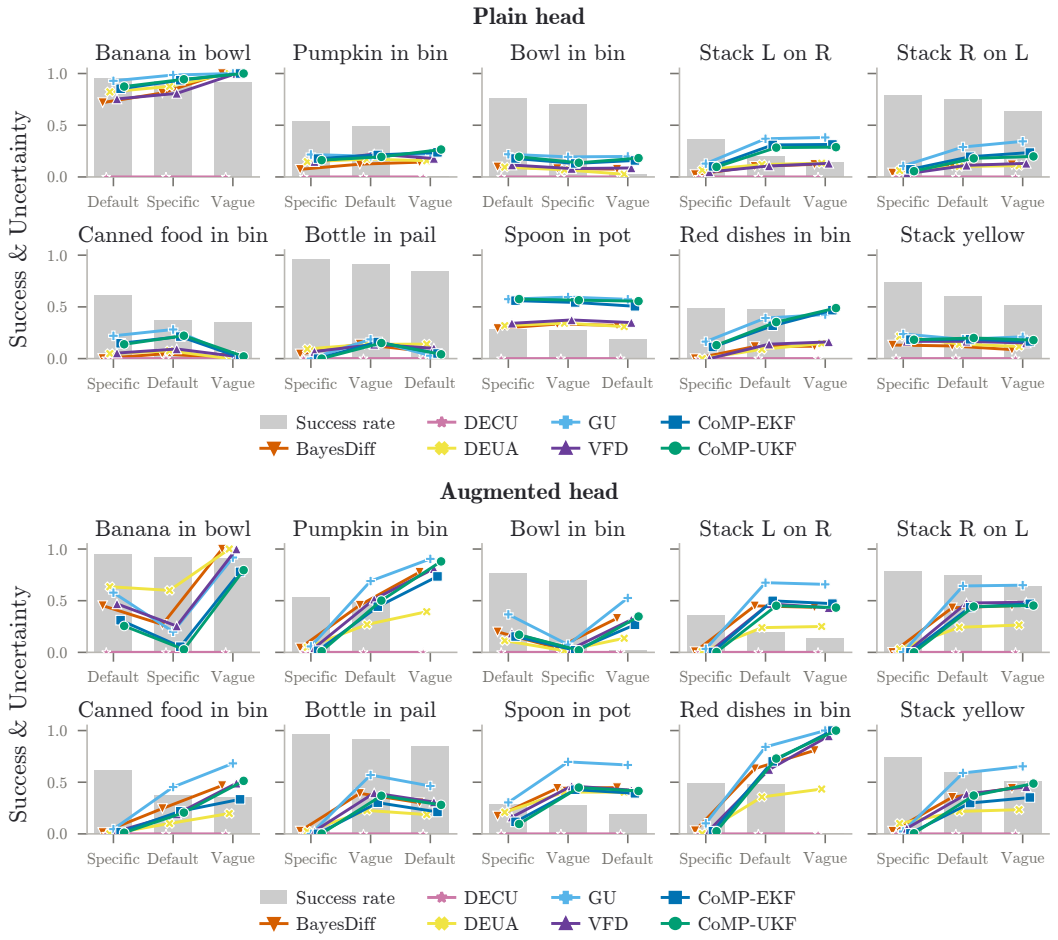

Figure 8: RoboLab instruction shift, per task. Success rate (grey) and first-decision uncertainty (lines, min-max scaled to $[0, 1]$ per estimator and Laplace fit across tasks and wordings) for the ten tasks and three wordings, under the plain (top) and augmented (bottom) Laplace heads. With the augmented head, every estimator except DECU generally reports higher uncertainty as language instructions become more ambiguous, while the plain head remains mostly flat, highlighting the benefits of our augmented Laplace posterior.

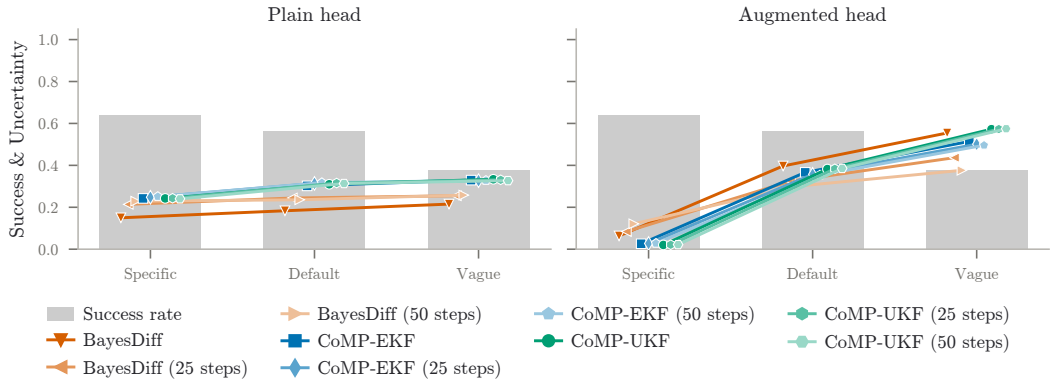

Figure 9: BayesDiff's response weakens with more integration steps. Success rate (grey) and first-decision uncertainty, averaged over the ten RoboLab tasks by wording, for BayesDiff, CoMP-EKF and CoMP-UKF at 10 Euler integration steps (as in Figure 3), 25 and 50. On both Laplace heads, each CoMP variant's response remains nearly unchanged as the number of integration steps increases, while BayesDiff's response to instruction ambiguity becomes flatter.

A.9 Supplementary Details and Results: Runtime Failure Detection

Robomimic. We use the policies and Laplace posteriors of Appendix A.7. We evaluate six perturbation magnitudes and an unperturbed condition, with 50 episodes per condition for a total of 350 test rollouts per task. On *Lift* and *NutAssemblySquare*, we displace both the target object in the table plane and the end effector from its initial pose along random directions. The respective magnitude ranges are 0.056–0.116 m and 0.050–0.120 m for *Lift*, and 0.005–0.045 m and 0.026–0.095 m for *NutAssemblySquare*. On *PickPlaceCan*, we perturb only the end effector by 0.057–0.202 m and leave the can's position unperturbed; its initial position already varies across the demonstrations.

An episode runs for at most 400 environment steps with an execution horizon of $h = 8$ and $T = 50$ Euler integration steps, giving at most $E = 50$ decisions. Thresholds are calibrated on a separate set of 20 successful, unperturbed episodes per task. BayesDiff estimates its cross term with $S = 24$ state particles per step, the sampling estimators draw $M = 16$ posterior weight samples, and VFD averages the scores of $K = 5$ action candidates. For comparison and following Römer et al. (2026), we also report DECU and VFD with ensembles of two independently trained policies, following the training recipe of Appendix A.7. Results for the sampling estimators are averaged over five random seeds. Figure 10 illustrates the perturbations on the three robomimic tasks.

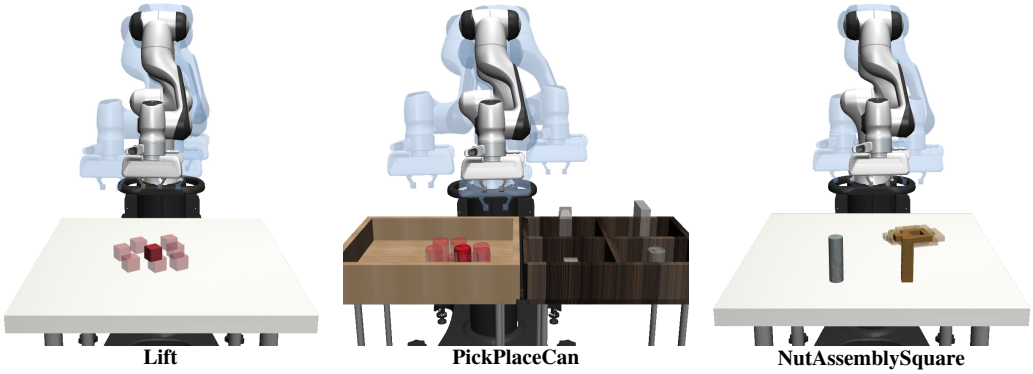

Figure 10: Perturbations on robomimic. The nominal initial state of each task from the front camera (solid), with perturbed resets overlaid as ghosts: the target object displaced in the table plane (translucent) and the end effector displaced from its reset pose (blue). Walls and floor are removed for clarity. The object displacements on PickPlaceCan are shown for illustration only, as its sweep perturbs the end effector alone.

LIBERO-90. We use openpi's `pi05_libero` checkpoint of $\pi_{0.5}$, fine-tuned on the LIBERO-Spatial, Object, Goal and 10 suites (Liu et al., 2023a). We select LIBERO-90 tasks 2, 12, 14, 29, 38, 59, 60, 64 and 70, for which the policy's success rate lies between 0.3 and 0.7, following the evaluation criterion in FIPER (Römer et al., 2025). Figure 11 shows their initial scenes and instructions. Both Laplace posteriors are fit once using 30k observations from the four training suites; the augmented head appends the same pooled image and language features as in Appendix A.8, standardized per dimension using these observations. Each Laplace fit takes 21 minutes on one A100.

We run 50 episodes per task, each capped at 400 environment steps. The policy predicts chunks of $H = 10$ actions with $T = 10$ Euler integration steps and executes $h = 5$ actions per decision, giving at most $E = 80$ decisions. Thresholds are calibrated on 20 successful episodes per task, drawn from two calibration runs on scenes disjoint from the test set. BayesDiff estimates its cross term with $S = 10$ state particles per step, the sampling estimators draw $M = 16$ posterior weight samples, and VFD averages the scores of $K = 5$ action candidates. Results for the sampling estimators are averaged over five random seeds.

Threshold calibration and evaluation. Following FIPER Römer et al. (2025), we aggregate uncertainty over a trailing window of $w = 5$ decisions to reduce sensitivity to isolated score fluctuations. Let U_t denote the resulting windowed score and $\tau \triangleq ((o_t, a^{(t)}_{1:H}))_{t=1}^{E_\tau}$ a rollout of E_τ decisions, with observation o_t and predicted action chunk $a^{(t)}_{1:H}$ at decision t. For each estimator

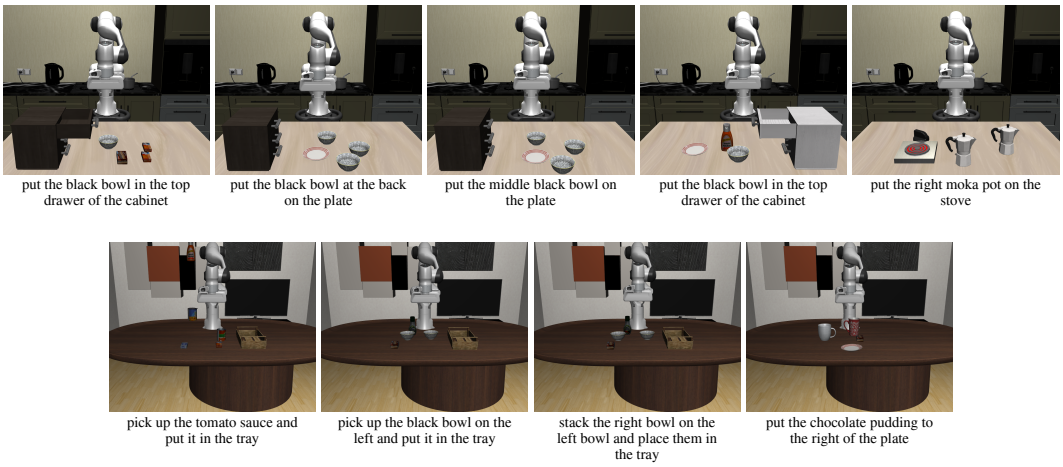

Figure 11: The nine LIBERO-90 tasks of §4.3. Initial scene and instruction of each task from the simulator's front camera.

and task, thresholds are fitted on a calibration set $\mathcal{D}_{\mathrm{c}} = \{\tau^i\}_{i=1}^{N}$ of N successful rollouts, disjoint from the test episodes. We use FIPER's constant conformal prediction threshold, computed as the empirical q-quantile of the per-rollout maxima:

$$\gamma_q \triangleq \mathrm{Quantile}_q\left(\left\{\max_t U_t^i\right\}_{i=1}^{N}\right), \qquad F(\tau_{:t}) \triangleq \mathbf{1}[U_t > \gamma_q], \tag{33}$$

where U_t^i is the windowed score at decision t of calibration rollout τ^i, and $F(\tau_{:t}) \in \{0, 1\}$ is the binary failure alarm based on the test rollout up to decision t. A test rollout is flagged at the first decision for which $F(\tau_{:t}) = 1$, with the threshold γ_q held fixed throughout the episode. We use the same window size and threshold rule for every estimator and average the evaluation metrics over ten quantiles $q \in \{0.90, 0.91, \ldots, 0.99\}$ so results are not cherry picked. Results for the sampling estimators are further averaged over five random seeds.

Metrics. Throughout the failure detection experiments, we use balanced accuracy, $\mathrm{Acc} \triangleq (\mathrm{TPR}+\mathrm{TNR})/2$, following FIPER (Römer et al., 2025). The true positive rate (TPR) is the fraction of failed rollouts correctly flagged, and the true negative rate (TNR) is the fraction of successful rollouts left unflagged. For a correctly flagged failure, normalized detection time is $\mathrm{DT} \triangleq (t_{\mathrm{det}} - 1)/(N_{\mathrm{steps}} - 1)$, where t_{det} is the policy step at which the first alarm is raised and N_{steps} is the maximum number of policy steps in a test episode for that task. The timestep-wise accuracy (TWA) discounts late failure detections as

$$\mathrm{TWA} \triangleq \frac{1}{2}\left[\mathrm{TNR} + \mathrm{TPR}\,(1 - \mathrm{DT})\right], \tag{34}$$

where DT is the mean normalized detection time over correctly flagged failures. All estimators score the same rollouts recorded with the MAP policy, so they receive identical observations and differences in detection arise from their uncertainty scores only.

Additional results. Table 3 compares all estimators with both Laplace heads on robomimic and LIBERO-90, allowing the effect of the augmented Laplace posterior to be assessed for each method. On robomimic, our augmented head improves TWA and accuracy for most estimators, including BayesDiff, and reduces detection time for all methods. On LIBERO-90, it raises CoMP-UKF's TWA from 0.613 to 0.638 and lowers DT from 0.512 to 0.489. With either head, both CoMP variants outperform BayesDiff on all three suite-averaged metrics, while CoMP-UKF remains competitive with the strongest post-hoc baselines.

Table 3: Runtime failure detection on robomimic and LIBERO-90 with both Laplace heads. Timestep-wise accuracy, balanced accuracy and normalized detection time using the constant threshold (33) with $w = 5$, as mean $\pm$ standard deviation over five sampling seeds. Rows show individual robomimic tasks, their average, and the average over nine LIBERO-90 tasks. All post-hoc estimators use the plain head (top) or our augmented head (bottom). The best post-hoc value in each row is bold and the second best underlined. Trained ensembles are references evaluated only on robomimic.

		Post-hoc, single model							Trained ensembles	
Task	Metric	BayesDiff	DECU	DEUA	GU	VFD	CoMP-EKF (ours)	CoMP-UKF (ours)	DECU-ens	VFD-ens
Plain head										
can	TWA ↑	0.605±0.061	0.766±0.004	0.766±0.006	0.739±0.013	0.768±0.004	0.742±0.042	**0.777±0.003**	0.797±0.003	0.789±0.002
	Acc. ↑	0.664±0.084	0.845±0.007	0.842±0.006	0.780±0.019	0.840±0.005	0.823±0.038	**0.855±0.005**	0.918±0.009	0.877±0.002
	DT ↓	0.330±0.080	0.205±0.010	0.199±0.009	**0.097±0.016**	0.178±0.001	0.234±0.059	0.196±0.013	0.258±0.012	0.206±0.004
lift	TWA ↑	0.566±0.021	0.580±0.009	0.579±0.007	**0.628±0.012**	0.621±0.004	0.607±0.014	0.618±0.006	0.612±0.006	0.636±0.002
	Acc. ↑	0.691±0.041	0.735±0.010	0.727±0.008	**0.763±0.016**	0.749±0.003	0.746±0.012	0.740±0.006	0.756±0.009	0.742±0.003
	DT ↓	0.543±0.060	0.365±0.006	0.347±0.007	0.318±0.009	0.300±0.005	0.378±0.080	**0.287±0.004**	0.332±0.013	0.242±0.004
square	TWA ↑	0.503±0.013	0.536±0.006	0.542±0.009	**0.568±0.002**	0.552±0.003	0.550±0.005	0.560±0.005	0.627±0.006	0.682±0.001
	Acc. ↑	0.572±0.038	0.615±0.012	0.628±0.016	0.654±0.004	0.650±0.006	0.662±0.025	**0.671±0.005**	0.781±0.006	0.814±0.001
	DT ↓	0.502±0.059	0.489±0.016	0.480±0.015	**0.381±0.016**	0.491±0.008	0.490±0.035	0.457±0.009	0.366±0.006	0.286±0.003
Average	TWA ↑	0.558±0.021	0.627±0.002	0.629±0.004	0.645±0.007	0.647±0.001	0.633±0.016	**0.652±0.003**	0.679±0.001	0.702±0.001
	Acc. ↑	0.642±0.036	0.732±0.005	0.732±0.007	0.732±0.010	0.746±0.002	0.744±0.014	**0.755±0.003**	0.818±0.003	0.811±0.001
	DT ↓	0.458±0.054	0.353±0.006	0.342±0.003	**0.265±0.012**	0.323±0.002	0.367±0.041	0.314±0.005	0.319±0.006	0.245±0.002
LIBERO-90	TWA ↑	0.528±0.009	0.524±0.013	0.586±0.009	0.624±0.004	**0.625±0.005**	0.587±0.000	0.613±0.000	–	–
	Acc. ↑	0.620±0.014	0.592±0.014	0.675±0.015	**0.761±0.004**	0.731±0.007	0.693±0.000	0.761±0.000	–	–
	DT ↓	0.596±0.037	0.589±0.040	0.586±0.036	0.535±0.025	0.523±0.028	0.514±0.000	**0.512±0.000**	–	–
Augmented head										
can	TWA ↑	0.690±0.038	**0.771±0.003**	0.757±0.005	0.699±0.004	0.759±0.006	0.758±0.000	0.761±0.000	0.797±0.003	0.789±0.002
	Acc. ↑	0.798±0.026	**0.832±0.003**	0.817±0.006	0.719±0.004	0.819±0.008	0.815±0.000	0.819±0.000	0.918±0.009	0.877±0.002
	DT ↓	0.317±0.109	0.147±0.003	0.143±0.004	**0.044±0.002**	0.142±0.004	0.139±0.000	0.142±0.000	0.258±0.012	0.206±0.004
lift	TWA ↑	0.609±0.002	**0.624±0.009**	0.599±0.001	0.608±0.005	0.598±0.002	0.607±0.000	0.607±0.000	0.612±0.006	0.636±0.002
	Acc. ↑	0.762±0.005	**0.781±0.008**	0.746±0.002	0.762±0.004	0.745±0.003	0.754±0.000	0.755±0.000	0.756±0.009	0.742±0.003
	DT ↓	0.345±0.007	0.354±0.020	0.349±0.003	0.352±0.017	0.349±0.004	**0.333±0.000**	0.336±0.000	0.332±0.013	0.242±0.004
square	TWA ↑	0.517±0.007	0.591±0.004	0.592±0.002	0.598±0.009	0.593±0.001	**0.606±0.000**	0.601±0.000	0.627±0.006	0.682±0.001
	Acc. ↑	0.614±0.037	0.692±0.006	0.694±0.002	0.691±0.009	0.699±0.003	**0.729±0.000**	0.706±0.000	0.781±0.006	0.814±0.001
	DT ↓	0.550±0.079	0.416±0.004	0.416±0.005	**0.319±0.013**	0.413±0.003	0.395±0.000	0.395±0.000	0.366±0.006	0.286±0.003
Average	TWA ↑	0.605±0.014	**0.662±0.004**	0.649±0.001	0.635±0.003	0.650±0.003	0.657±0.000	0.656±0.000	0.679±0.001	0.702±0.001
	Acc. ↑	0.725±0.020	**0.769±0.005**	0.752±0.002	0.724±0.003	0.754±0.003	0.766±0.000	0.760±0.000	0.818±0.003	0.811±0.001
	DT ↓	0.404±0.055	0.306±0.006	0.302±0.001	**0.238±0.005**	0.301±0.001	0.289±0.000	0.291±0.000	0.319±0.006	0.245±0.002
LIBERO-90	TWA ↑	0.540±0.012	0.527±0.005	0.570±0.006	0.611±0.005	0.534±0.007	0.565±0.000	**0.638±0.000**	–	–
	Acc. ↑	0.651±0.014	0.605±0.006	0.659±0.009	0.746±0.009	0.614±0.009	0.678±0.000	**0.761±0.000**	–	–
	DT ↓	0.613±0.026	0.634±0.050	0.591±0.045	0.534±0.019	0.531±0.042	0.547±0.000	**0.489±0.000**	–	–

A.10 Supplementary Details and Results: Uncertainty-Guided Fine-Tuning

Policy and posterior fitting. We use SmolVLA with the SmolVLM2-500M backbone and an action expert with 480-dimensional last-layer features, following SAVE's training recipe (Römer et al., 2026). The policy takes two 256×256 images, proprioception and a language instruction as input, predicts $H = 50$ actions and executes $h = 25$. We train it for 30k steps at batch size 32 on the LIBERO-Spatial, Object and Goal suites and LIBERO-10 tasks 0 to 2. Each policy occupies 1.2 GB (about 0.6B parameters), with a LIBERO-10 success rate of 20 to 25% before fine-tuning. We train ten independent policies, two for each of five evaluation seeds. All post-hoc estimators use one policy per seed, while the two-member VFD baseline (Römer et al., 2026) uses both to score $K = 5$ candidate chunks.

For the plain Laplace posterior, we fit a Kronecker-factored Laplace posterior over the expert's output layer with 480 input features, using 30k frames stratified over the training episodes. The augmented head appends 3840 features obtained by averaging the backbone's token features separately for each camera view, the language instruction and the proprioceptive state. This increases posterior storage from 0.95 MB to 75 MB. One SmolVLA member trains for 30k steps in 6.5 GPU-hours on a single A100, and each 30k-frame head fit takes 1.8 GPU-hours, of which the forward pass of the frozen policy is the bulk.

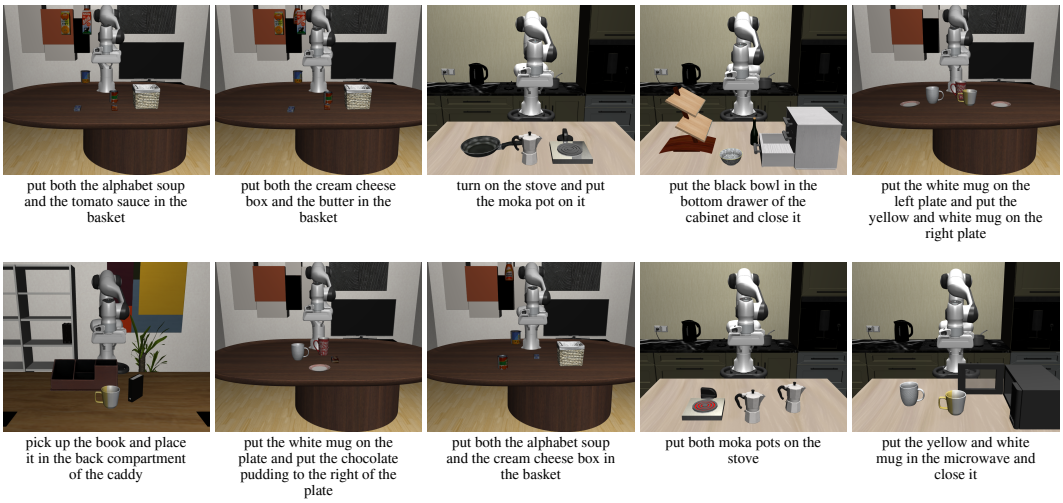

Figure 12: The ten LIBERO-10 tasks of §4.4. Initial scene and instruction of each task from the simulator's front camera.

Demonstration selection. The candidate pool contains all LIBERO-10 demonstrations from the HuggingFaceVLA release[5]. Each demonstration is scored once at its first frame with $T = 10$, $K = 5$ candidates for Laplace-based VFD, $M = 16$ posterior draws for the sampling estimators and $S = 24$ particles for BayesDiff. We select demonstrations once per run for budgets $X \in \{25, 50, 75, 100, 150, 200, 250\}$. Following SAVE (Römer et al., 2026), we compute task sampling probabilities as

$$\bar{U}_j \triangleq \frac{1}{|\mathcal{D}_j|} \sum_{d \in \mathcal{D}_j} U(d), \qquad p(j) = \frac{\bar{U}_j^{2.5}}{\sum_k \bar{U}_k^{2.5}}, \tag{35}$$

where $\mathcal{D}_j$ is the demonstration pool for task j and $U(d)$ is the uncertainty score at the first frame of demonstration d. We sample tasks according to $p(j)$ and select their highest-scoring demonstrations until the budget X is met, using uniform task sampling if all task scores are zero.

Fine-tuning and evaluation. For each seed, all fine-tuning runs start from the same pre-trained checkpoint. We use AdamW for 8000 steps at batch size 32, sampling equally from the selected demonstrations and the original training data. The learning rate warms up to 5×10^{-5} over 200 steps,

[5] https://huggingface.co/datasets/HuggingFaceVLA/libero

followed by cosine decay to 5×10^{-6}. Each fine-tuned policy is evaluated on all ten LIBERO-10 tasks (Figure 12).

Metrics and additional results. We assess demonstration selection with two metrics. The first measures the demonstrations needed to reach 90% of random selection's mean success over the five seeds at $X = 250$, using linear interpolation of each policy's success curve. The second measures the average improvement over random selection across budgets. For each method and seed, we compute the weighted average:

$$\text{Gain} \triangleq \sum_X w_X \left[\text{SR}(X) - \text{SR}_{\text{rand}}(X)\right], \tag{36}$$

where $\text{SR}(X)$ and $\text{SR}_{\text{rand}}(X)$ are the method's and random selection's success rates for the same seed, measured in percent and averaged over all ten LIBERO-10 tasks. The weights w_X are the normalized trapezoidal weights, accounting for the spacing between budgets and summing to one. Gain is therefore measured in percentage points. Both metrics are reported as means and standard deviations over the five trained policies.

Figure 13 extends Figure 5 to compare plain and augmented Laplace heads for every estimator; Figure 14 gives the full success curves. Our augmented Laplace head increases mean gain and reduces the average number of demonstrations needed for every estimator by reusing observation embeddings the policy already computes. With a single policy, CoMP-UKF and several augmented baselines exceed the two-member VFD ensemble's mean gain, while CoMP-EKF remains close to random selection. Although BayesDiff benefits from the augmented head at 10 integration steps, the following ablation shows that this gain disappears as the step count increases, consistent with its variance collapse.

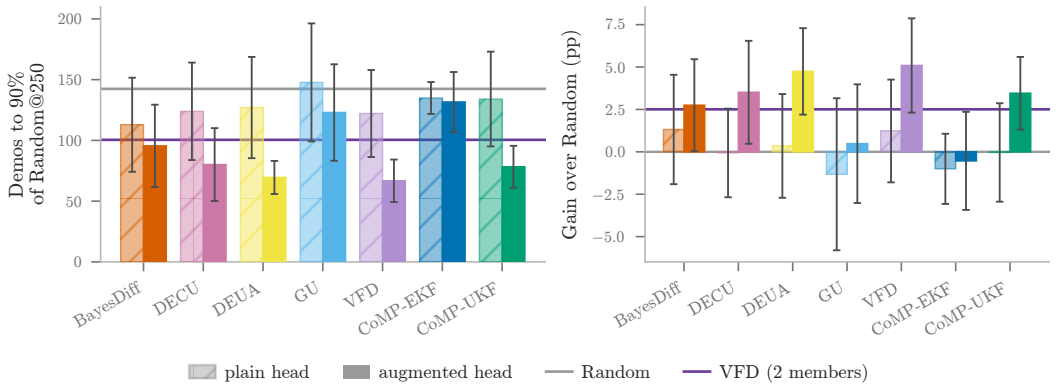

Figure 13: Effect of augmented Laplace head on data selection. Demonstrations needed to reach 90% of random selection's mean success at $X = 250$ (left; lower is better) and mean gain over random across budgets (right; higher is better), on all ten LIBERO-10 tasks. Paired bars compare plain and augmented heads for every estimator, with mean and standard deviation over five pre-trained policies. Horizontal lines mark random selection and two-member VFD. Overall, our augmented Laplace head improves both metrics for every estimator, with CoMP-UKF and several baselines exceeding the two-member VFD ensemble's mean gain using only one policy.

Integration steps. Consistent with the variance decay theorem in §3, BayesDiff's fine-tuning advantage over random selection disappears as more integration steps are used to score demonstrations (Table 4). We compare 10, 25 and 50 Euler integration steps, keeping the fine-tuning protocol fixed and always evaluating the resulting policies at 10 steps. Even with our augmented Laplace posterior, BayesDiff's average gain nearly vanishes at 25 steps and turns negative at 50, with the same declining trend under the plain Laplace posterior. In contrast, CoMP-UKF retains its gain with the augmented head across all three step counts. CoMP-EKF's gains are less consistent across step counts and repetitions, which may reflect its first-order linearization; CoMP-UKF instead propagates sigma points through the nonlinear velocity field avoiding this issue.

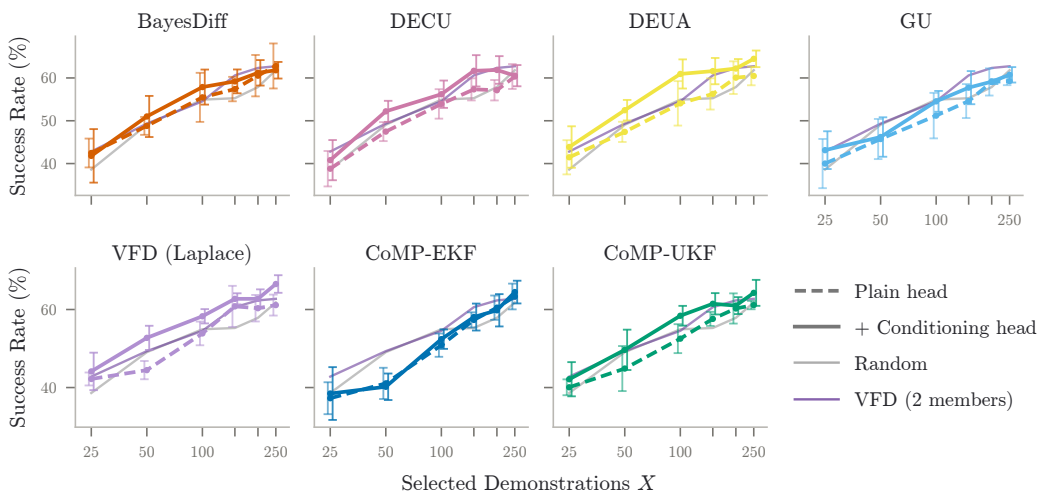

Figure 14: Data selection on LIBERO-10, per estimator. Success rate after fine-tuning on the ten LIBERO-10 tasks against the number of selected demonstrations X, for every estimator on the plain head (dashed) and on the augmented head (solid), with random selection and the two-member VFD as references; mean and standard deviation over five pre-trained policies. On the plain Laplace head every estimator tracks random selection; our augmented head brings most estimators above random selection, with several also surpassing the two-member VFD.

Table 4: BayesDiff's data selection weakens as integration steps increase. Average gain over random selection (pp) and demonstrations needed to reach 90% of random selection's mean success at $X = 250$ on the ten LIBERO-10 tasks. Selection scores use 10, 25 or 50 Euler integration steps; the fine-tuning protocol is fixed and evaluation always uses 10 integration steps. Mean and standard deviation over five pre-trained policies, with the plain (top) and augmented Laplace head (bottom). The best value in each row is bold and the second best underlined. BayesDiff's gain declines as integration steps increase, while CoMP-UKF retains its gain with our augmented head.

Metric	Steps	BayesDiff	CoMP-EKF (ours)	CoMP-UKF (ours)
Plain head				
Gain over random (pp) ↑	10	**+1.3 ± 3.2**	-1.0 ± 2.1	0.0 ± 2.9
	25	-2.6 ± 4.3	-0.5 ± 3.8	**+0.4 ± 3.2**
	50	-3.8 ± 5.4	-0.6 ± 2.9	**+1.1 ± 2.8**
Demos to 90% of random ↓	10	**113 ± 43**	135 ± 15	134 ± 44
	25	148 ± 58	130 ± 33	**126 ± 52**
	50	157 ± 45	142 ± 33	**119 ± 30**
Augmented head				
Gain over random (pp) ↑	10	+2.7 ± 2.7	-0.5 ± 2.9	**+3.5 ± 2.1**
	25	+0.5 ± 2.5	+2.5 ± 3.2	**+3.3 ± 2.2**
	50	-2.9 ± 3.8	+1.7 ± 4.1	**+3.7 ± 2.5**
Demos to 90% of random ↓	10	95 ± 38	132 ± 28	**78 ± 19**
	25	118 ± 44	102 ± 20	**74 ± 16**
	50	150 ± 31	108 ± 41	**84 ± 7**