Robotics: Science and Systems 2026
Sydney, Australia, July 13–17, 2026

When to Act, Ask, or Learn: Uncertainty-Aware Policy Steering

Jessie Yuan[1*], Yilin Wu[1*], Andrea Bajcsy[1]
[1]Carnegie Mellon University [*]Equal Contribution

***Abstract*—Policy steering is an emerging way to adapt robot behaviors at deployment-time: a learned verifier analyzes low-level action samples proposed by a pre-trained policy (e.g., diffusion policy) and selects only those aligned with the task. While Vision-Language Models (VLMs) are promising general-purpose verifiers due to their reasoning capabilities, existing frameworks often assume these models are well-calibrated. In practice, the overconfident judgment from VLM can degrade the steering performance under both high-level semantic uncertainty in task specifications and low-level action uncertainty or incapability of the pre-trained policy. We propose *uncertainty-aware policy steering* (UPS), a framework that jointly reasons about semantic task uncertainty and low-level action feasibility, and selects an uncertainty resolution strategy: execute a high-confidence action, clarify task ambiguity via natural language queries, or ask for action interventions to correct the low-level policy when it is deemed incapable at the task. We leverage conformal prediction to calibrate the composition of the VLM and the pre-trained base policy, providing statistical assurances that the verifier selects the correct strategy. After collecting interventions during deployment, we employ residual learning to improve the capability of the pre-trained policy, enabling the system to learn continually but with minimal expensive human feedback. We demonstrate our framework through experiments in simulation and on hardware, showing that UPS can disentangle confident, ambiguous, and incapable scenarios and minimizes expensive user interventions compared to uncalibrated baselines and prior human- or robot-gated continual learning approaches. Videos can be found at: https://jessie-yuan.github.io/ups/.**

I. Introduction

Imitation-based generative robot policies, such as diffusion policies and vision-language-action (VLA) models, are a promising way to model low-level action distributions conditioned on perceptual (and language) inputs. However, directly executing samples from the learned action distribution does not always lead to success at deployment time. This has motivated the paradigm of *policy steering* which adapts the behavior of a pre-trained policy online, e.g., via sampling and verification.

Recent works [30, 12] demonstrated that vision-language models (VLMs) can be harnessed as "open world" verifiers during policy steering, reasoning about the outcomes of action samples and selecting those which align most with task instructions. However, to-date, all approaches have implicitly assumed that the verifier is well-calibrated: i.e., it selects an appropriate action sample (or correctly rejects all candidates) with high confidence. In practice, this assumption often fails. VLM verifiers can be overconfident when task instructions are ambiguous or under-specified, leading to confident selection of misaligned behaviors. Moreover, when the low-level policy is fundamentally *incapable* of accomplishing the task under the current conditions, all candidate samples may be unsatisfactory. Our experiments show that an uncalibrated verifier may still select one of these faulty options rather than identifying the limitation and stopping execution.

In this work, we propose ***U**ncertainty-aware **P**olicy **S**teering* (**UPS**), a method that maps a VLM verifier's uncertainty over action samples into a resolution strategy: **execute** a high-confidence action sample, **clarify** task ambiguity through natural-language interactions, or request to **re-train** the low-level policy when it is incapable. Specifically, we calibrate the composition of the VLM verifier and the pre-trained policy using conformal prediction [2], returning a *set* of action sample(s) and possibly a "none of the above" option. This set is constructed such that, with a user-specified level $1-\epsilon$, the verifier can steer the low-level policy to success in $1-\epsilon$ of scenarios by either executing the right action sample or selecting the appropriate resolution strategy (asking clarification questions when ambiguous or eliciting human interventions when incapable). Our conformal prediction approach also minimizes the average size of prediction sets, ensuring that the robot selectively chooses to ask semantic clarification questions (when set size is > 1) or chooses to ask for low-level retraining (when set only contains "none of the above"). When low-level human intervention is required, we improve the policy via residual learning [10, 31], enabling continual improvement of the robot's capabilities.

We evaluate UPS in a series of simulation and hardware experiments with robotic manipulation. Compared to prior policy steering methods that do *not* calibrate the verifier, we show 30% improvement in ambiguous scenarios. On the uncertainty-quantification side, we show that our approach to conformal prediction, including a new score function with VLMs, yields strong empirical coverage guarantees with tight prediction sets compared to prior UQ methods for VLMs. On the continual learning side, we show that UPS's ability to reason about both high-level semantic uncertainty and low-level action uncertainty enables it to more effectively balance cheap language queries vs. expensive low-level action interventions from a human. Furthermore, compared to human-[11] or robot-gated [19] interactive imitation learning approaches, the data collected by UPS enables the robot to improve over successive deployment rounds while minimizing human effort.

II. Related Work

Policy Steering in Robotics. Recent work increasingly use VLMs as deployment-time verifiers to steer robot policies by

scoring multiple sampled trajectories [30, 12, 29]. However, these pipelines suffer from the same problems as LLM-based verification, including poor calibration and systematic agreement bias, where plausible but incorrect actions are blindly accepted [15, 1]. These failures are especially problematic in robotics because verifier outputs are often reduced to a scalar score (accept/reject decision), obscuring *what* is uncertain and *how* the system should respond. Our work addresses this gap by quantifying uncertainty in VLM verification and mapping different uncertainty modes to distinct resolution strategies, from high-level clarification to low-level data collection.

Uncertainty Quantification for Learned Robot Policies. While uncertainty quantification (UQ) has been a long-standing problem in robotics and machine learning [27, 7], in this paper, we focus on learning-based robot policies that generate low-level actions from multimodal inputs [6, 21] such as images and language instructions. Prior work typically addresses UQ at two decoupled levels of abstraction: semantic ambiguity and low-level action uncertainty. For high-level semantic uncertainty, approaches like KnowNo [21] and RCIP [13] employ conformal prediction to provide statistical guarantees that the generated plans (or intent predictions) capture the user's true instruction. However, these methods largely operate in discrete textual or symbolic space with an unrealistic assumption that the low-level policy can always execute the chosen plan. A separate body of work quantifies uncertainty directly within the continuous action space of learned policies, such as VLA or diffusion models [26, 34]. While these methods can flag distribution shifts or execution noise [32], low-level action uncertainty does not always correlate with task-level failures (e.g., there are multiple ways to successfully grasp a cup). Our work takes a step towards more tightly coupling the uncertainty of the low-level action policy (approximated by repeatedly sampling from the policy) with high-level semantic uncertainty about the task (quantified by how well the action outcomes align with the user's instruction).

Continual Learning via Interactive Imitation. Interactive Imitation Learning (IIL) improves policies through human feedback. The DAgger family of methods [23, 9] typically relies on the human operator to know when to intervene, or on action-level uncertainty signals to trigger interventions, which has challenges as described in the above section. In contrast, our approach leverages calibration to capture semantic uncertainty over different meaningful phases (e.g., grasping, placing) to minimize human feedback. Recent work also uses the imagination of a world model to evaluate long-horizon execution failures [14]. However, these methods typically evaluate a single policy trajectory. Because they do not sample multiple rollouts within the world model, they can't effectively "search" through the base policy's distribution to find an alternative action plan; this can result in asking for human interventions even when a feasible solution exists within the support of the policy distribution. Our method mitigates this by performing action sampling and calibrated verification. Finally, to adapt efficiently from human intervention, recent works utilize residual policy learning [10, 31] instead of fine-tuning the entire base model. We adopt this paradigm as it enables our system to integrate user corrections into the generation-imagination loop without destroying the diverse capabilities of the base policy.

III. Problem Formulation

Setup & Notation. We define the user's task instruction $\mathcal{L}$ as a single sentence defining the goal of a long-horizon task that involves multiple T-timestep subtasks. The robot's observation space $o \in \mathcal{O} := \mathcal{I} \times \mathcal{S}$ integrates RGB images I with proprioceptive states s, such as end-effector poses. Control is governed by a pre-trained base policy $\pi(a \mid o_t)$, typically a generative visuomotor model such as a diffusion policy [6] or a vision-language-action model [5]. At any timestep t, the policy generates a short action chunk $a_{t:t+H}$ ($H \ll T$), which is aggregated into a full sequence $\mathbf{a}_t = a_{t:t+T}$ with multiple continuous generations. Given an observation o_t, sampling from the policy $\mathbf{a}_t \sim \pi(\cdot \mid o_t)$ and executing the plan yields a corresponding future observation sequence $\mathbf{o}_t \sim \mathbb{P}(\cdot \mid o_t, \mathbf{a}_t)$. This formulation allows us to evaluate the policy's predicted outcomes against the high-level task instruction across multiple stages of execution.

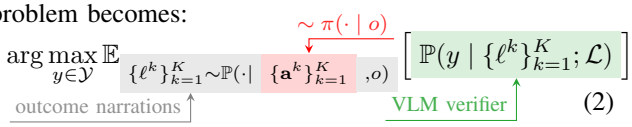
Problem. Given the robot's current observation o and K *i.i.d.* samples from the base policy, $\{\mathbf{a}^i\}_{k=1}^K \sim \pi(\mathbf{a} \mid o)$, our *uncertainty-aware policy steering* problem seeks to return the index $y \in \mathcal{Y} = \{1, ..., K, K+1\}$ of the action sample that best matches the textual task description $\mathcal{L}$ *or* return a $K+1$-th index which indicates "none of the above" (i.e., no samples accomplish the instructed task). More formally, we seek to solve the following optimization problem:

$$y^\star = \arg\max_{y \in \mathcal{Y}} \mathbb{P}(y \mid \{\mathbf{a}^k\}_{k=1}^K, o; \mathcal{L}). \tag{1}$$

In general, the distribution $\mathbb{P}(y \mid \{\mathbf{a}^k\}_{k=1}^K, o; \mathcal{L})$ above is very hard to characterize, which is why prior works [21, 30] have turned to vision-language models (VLMs) due to their ability to simultaneously reason about both natural language instructions and visual observations. However, prior work [30] shows that effective zero-shot use of VLMs to approximate $\mathbb{P}(y \mid \{\mathbf{a}^k\}_{k=1}^K, o; \mathcal{L})$ is not trivial, since the model must implicitly translate low-level actions into outcomes (i.e., predict $\{\mathbf{a}^k\}_{k=1}^K \to \{\mathbf{o}^k\}_{k=1}^K$), reason about the outcomes (i.e., alignment between the outcomes $\{\mathbf{o}^i\}_{k=1}^K$ and the text of the task $\mathcal{L}$), as well as select the correct index.

Following prior work [30, 29], we factorize the problem into two stages: first, predict the outcome of each low-level action and translate the outcomes into textual *outcome narrations*; given these narrations, we ask a VLM to solve a multiple-choice Q&A problem and select the action whose outcome narration best aligns with the task. Mathematically, our problem becomes:

$$y^\star = \arg\max_{y \in \mathcal{Y}} \mathbb{E}_{\{\ell^k\}_{k=1}^K \sim \mathbb{P}(\cdot \mid \{\mathbf{a}^k\}_{k=1}^K, o)} \left[\mathbb{P}(y \mid \{\ell^k\}_{k=1}^K; \mathcal{L}) \right]. \tag{2}$$



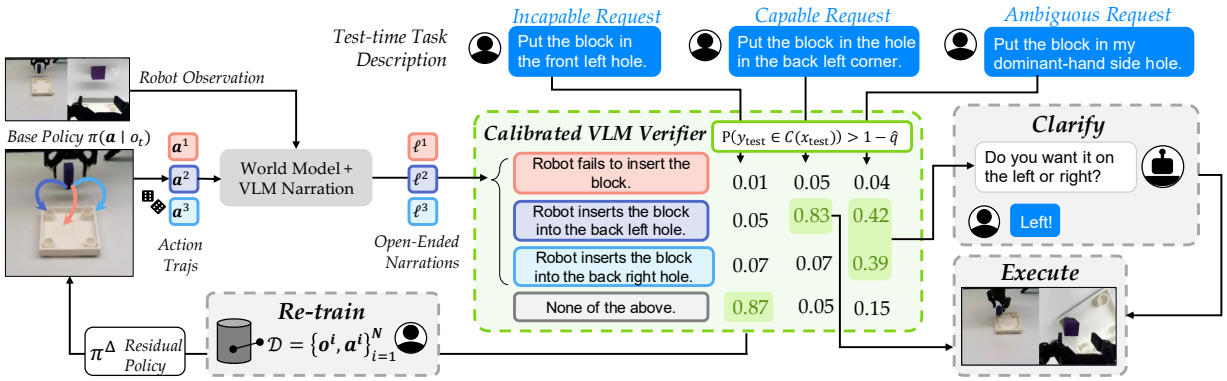


Fig. 1: **Uncertainty-Aware Policy Steering.** Our framework calibrates the VLM verifier used for policy steering via conformal prediction. This enables the VLM to select an appropriate way to resolve uncertainty, from querying the end-user in natural language to asking to re-train the low-level control policy.

Here, each ℓ^k is a language description of the outcome induced by the low-level action $\mathbf{a}^k$ executed from an initial observation. For example, ℓ^k = "The robot inserts the block into the back right hole." as shown in Fig. 1. In practice, these narrations are obtained by first *predicting future observations* $\mathbf{o}_t \sim \mathbb{P}(\cdot|o_t, \mathbf{a}_t)$ (e.g., via a world model, shown in Sec. IV), and then the predicted observations are described in text via video captioning [25]. By lifting the outcomes of low-level actions into textual descriptions, the VLM's reasoning capabilities can be used more effectively to verify which index $y \in \mathcal{Y}$ to choose; in other words, the VLM can be used as a proposal distribution in the optimization objective: $\mathbb{P}(y \mid \{\ell^k\}_{k=1}^K; \mathcal{L})$.

Challenges & Opportunities. A central challenge with Eq. 2 is the assumption of a well-calibrated VLM verifier. Even with perfect outcome narrations to choose from, VLMs are often overconfident evaluators [3, 18] which can undermine the quality of the policy steering. Moreover, the task descriptions $\mathcal{L}$ from the user may be inherently *ambiguous* or *underspecified*; a VLM verifier which confidently selects an incorrect behavior narration can lead to misaligned or even unsafe actions being executed. A second challenge is that when the policy is *incapable*, all K action samples from the policy in Eq. 2 will have unsatisfactory outcomes for the task instruction $\mathcal{L}$; an uncalibrated VLM verifier may choose one of the faulty samples instead of rejecting them all.

Finally, even if uncertainty or incapability is detected, the robot should *not* always stop operation and instead should generate an appropriate resolution strategy. However, actionable strategies for resolving uncertainty can range from simple clarifications with the user (e.g., "Did you want it on the left or right?") to far more demanding interventions such as re-training the low-level control policy (e.g., refining the precision of the skill to insert the block into the hole or learning a new skill for placing into a new hole). In other words, the source of uncertainty can lie at different levels of abstraction, from *semantic ambiguity* (e.g., multiple indices seem plausible for the task) to the base policy's *incapability*. Our unified framework seeks to both calibrate the VLM verifier and equip the robot with appropriate resolution strategies.

Aside: Comparison to KnowNo [21]. A closely related work [21] similarly performs uncertainty quantification on an LLM task planner that reasons about textual plans and asks questions when highly uncertain. However, it assumes an idealized low-level policy that executes any chosen plan accurately. We relax this assumption by grounding the uncertainty quantification in the capabilities of low-level policy. We achieve this because our textual plans (that the VLM verifier reasons about) directly correspond to outcomes induced by samples from low-level policy in Eq. 2, as well as an explicit "incapability" option. Thus, our uncertainty quantification jointly reasons about high-level semantic intent uncertainty and low-level action uncertainty. Together, our system not only resolves what the user wants, but also identifies whether the policy can reliably accomplish the task.

IV. Approach: Uncertainty-Aware Policy Steering

Our uncertainty-aware policy steering approach calibrates the VLM verifier to select, with high user-specified confidence, one of three uncertainty resolution strategies: **execute** a high-confidence action that fulfills the task; **clarify** task ambiguity when multiple actions seem plausible; or **re-train** the low-level policy via interactive imitation learning when it is incapable. Our overall framework is visualized in Fig. 1, and we describe each component in the following subsections.

A. VLM-in-the-loop Policy Steering

We follow the framework from [30, 29] and pose VLM-in-the-loop steering as an open-ended multiple-choice Q&A problem. This involves three stages: predicting the outcomes $\mathbf{o}$ of an action sample $\mathbf{a}$ with a world model, generating behavior narrations ℓ for the predicted outcomes, and selecting from these options with the VLM verifier.

Outcome Prediction. We leverage latent world models [8, 33] to predict the outcomes of low-level action samples directly from a high-dimensional observation, o_t. The world

model $\mathcal{W}_\phi$ consists of an encoder $\mathcal{E}_\phi : \mathcal{O} \to \mathcal{Z}$ which encodes an observation o_t, a latent dynamics model $f_\phi : \mathcal{Z} \times \mathcal{A} \to \mathcal{Z}$ which predicts latent state transitions, and a decoder $\mathcal{D}_\phi : \mathcal{Z} \to \mathcal{O}$ which reconstructs an observation from latent space. Recall that most low-level policies [6, 5] predict relatively short-horizon action "chunks". This presents a challenge for policy steering, since on very short time-horizons, the outcomes are often indistinguishable for the verifier.

To address this, we interleave action generation and imagination to obtain a longer-horizon action-outcome sequence that is informative for the verifier to reason about (visualized in Figure 2). Specifically, given the current *real* observation o_t, we query $a_{t:t+H} \sim \pi(\cdot \mid o_t)$ to generate an action chunk of length H and pass this to the world model to obtain predicted future observations: $\hat{o}_{t+1:t+H} = \mathcal{W}_\phi(o_t, a_{t+1:t+H})$. The last predicted observation $\hat{o}_{t+H}$ is provided as input to the policy $\pi(\cdot \mid \hat{o}_{t+H})$ to generate the next action chunk. We repeat this process T/H times and aggregate all action sequences together into one T-length sequence $a_{t:t+T}$ with corresponding imagined observations $\hat{o}_{t:t+T}$. By executing this process in parallel for K samples, we produce a set of predicted observation sequences $\{\hat{o}^k_{t:t+T}\}_{k=1}^K$ that are (approximate) future outcomes.

Narration. Next, the imagined outcomes $\{\hat{o}^k_{t:t+T}\}_{k=1}^K$ are translated into textual narrations by leveraging the strong reasoning capabilities of the VLM verifier. In our experiments, we utilize a Gemini model [25] to produce these narrations. Let $\mathcal{T}^{\text{VLM}}$ be the VLM narration model and for any $\hat{\mathbf{o}}^k_t \in \{\hat{\mathbf{o}}^k_t\}_{k=1}^K$, let the corresponding narration be $\ell^k_t = \mathcal{T}^{\text{VLM}}(\hat{\mathbf{o}}^k_t, \mathcal{L})$. We augment the set of narrations with a $(K+1)$-th option stating that "none of the above samples are correct" to ensure the system can detect incapability.

Verification. Finally, the VLM verifier selects an option from the set of narrations (along with the "none" option) by answering an open-ended multiple-choice problem over these narrations. Mathematically, this amounts to the VLM selecting the most likely single token corresponding to one of the multiple choice options: $y_t = \arg\max_{y \in \mathcal{Y}} \mathbb{P}^{\text{VLM}}(y \mid \{\ell^k_t\}_{k=1}^{K+1}, \mathcal{L})$. This formulation aligns with the VLM's native log-likelihood loss and, as noted in [21], and eliminates length bias when estimating the probability of open-ended textual generations.

B. VLM Verifier Uncertainty Quantification

We employ conformal prediction (CP) [2] to quantify the VLM verifier's uncertainty during policy steering. Specifically, our goal is to use the potentially unreliable model likelihoods in $\mathbb{P}^{\text{VLM}}$ to construct tight prediction sets which are provably guaranteed to contain the correct action with probability of at least $1-\epsilon$, where ϵ is a user-defined error budget.

Conformal Prediction. Given some *input* x to a prediction model, conformal prediction aims to construct a prediction set $C(x) \subseteq \mathcal{Y}$ on the *outputs* of that model that contains the true label y with a probability of at least $1-\epsilon$ while minimizing the set size $|C(x)|$. This calibration requires a dataset $(x, y) \in \mathcal{D}_{\text{calib}}$ of independent and identically distributed ($i.i.d.$) input-output pairs drawn from the same distribution as the test domain. However, because we are in the sequential-decision-making setting, any input to the VLM verifier depends on prior steering decisions and thus are not $i.i.d.$.

To maintain formal coverage guarantees despite these temporal dependencies, we perform sequence-level calibration. Assume that the VLM verifier is called M times during the task. This means we have M true observations that the verifier uses, M sets of behavior narrations, and—for calibration—M true labels about which behavior narration(s) are correct at each timestep. Let the sequence-level calibration dataset be structured as follows:

$$\{o_i, \{\{\ell^k\}_{k=1}^K\}_i, \mathcal{L}, \mathcal{Y}_i^*\}_{i=0}^M \in \mathcal{D}_{\text{calib}}, \quad |\mathcal{D}_{\text{calib}}| = N.$$

For notational simplicity, denote any single *input* $x_i = (o_i, \{\{\ell^k\}_{k=1}^K\}_i, \mathcal{L})$ and (set of) ground-truth label(s) $\mathcal{Y}_i^*$ obtained from an end-user. Then, denote a *sequence* of inputs and labels as $\bar{x} = (x_0, \ldots, x_M)$ and $\bar{y} = (\mathcal{Y}_0^*, \ldots, \mathcal{Y}_M^*)$. To construct $\mathcal{D}_{\text{calib}}$, we sample multiple initial observations and different task instructions, use our interleaved generation-imagination loop to obtain K candidate narrations and the "none" option, annotate the correct set of indices, and execute the correct action; this results in the next observation that the verifier sees and serves as the beginning of next generation-imagination loop, and so on.

Given this calibration dataset $\mathcal{D}_{\text{calib}}$, we use the VLM verifier's learned distribution $\mathbb{P}^{\text{VLM}}$ to compute the *non-conformity score*, which is defined for any data point $(\bar{x}, \bar{y}) \in \mathcal{D}_{\text{calib}}$ as:

$$\kappa(\bar{x}, \bar{y}) = 1 - \min_{x_i \in \bar{x}} \min_{y \in \mathcal{Y}_i^*} \mathbb{P}^{\text{VLM}}(y \mid x_i). \tag{3}$$

We compute this quantity for all N sequences in the calibration dataset. There are a few points worth noting here. First, intuitively, for any calibration data point, the higher the score, the less the data "conforms" to the training data of the verifier. Second, we design this score function to compute a *minimum* over the entire sequence (outer min in Eq. 3) to ensure coverage guarantees despite these temporal dependencies. Finally, the inner minimium in Eq. 3 is a conservative score designed to penalize the model for not including *all* suitable options in the prediction set. From a user perspective, this avoids scenarios where the verifier guesses the user's unstated intent and incentivizes the verifier to ask the user about the entire set of plausible choices.

Finally, CP calibrates the VLM verifier by selecting the $\hat{q} = \frac{\lceil (N+1)(1-\epsilon) \rceil}{N}$ empirical quantile of the nonconformity scores $\{\kappa^1, ..., \kappa^N\}$ and uses $\hat{q}$ to construct the prediction set $C(x)$. This includes all labels that the VLM verifier is at least $1-\hat{q}$ confident about, ensuring the $1-\epsilon$ coverage guarantee [27, 21].

Deployment Time. At deployment time, the robot is given a new task description $\mathcal{L}^{\text{test}}$. Given the current observation o_i^{test}, we obtain $x_i^{\text{test}} = (o_i^{\text{test}}, \{\{\ell^k\}_{k=1}^K\}_i, \mathcal{L}^{\text{test}})$ via the same action sampling, prediction, and narration pipeline described above. We query the VLM verifier to estimate the probability

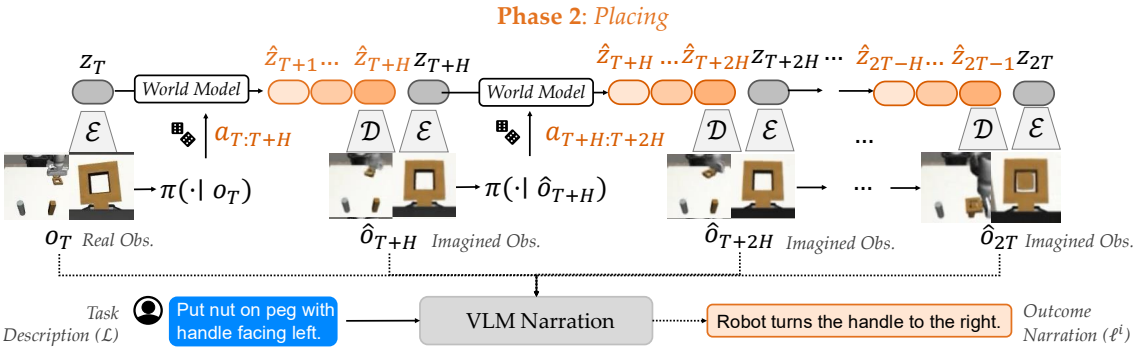


Fig. 2: **Outcome Prediction & Narration.** The policy and the world model are interleaved to predict long-horizon outcomes induced by the low-level policy. Decoded observations are fed into a VLM which narrates the outcomes in text.

$\mathbb{P}^{\text{VLM}}(y \mid x_i^{\text{test}})$ for each choice $y \in \mathcal{Y} = \{1, ..., K, K+1\}$, and the prediction set is generated via

$$C(x_i^{\text{test}}) = \{y \in \mathcal{Y} \mid 1 - \mathbb{P}^{\text{VLM}}(y \mid x_i^{\text{test}}) \leq \hat{q}\} \tag{4}$$

i.e., selecting any index whose probability is larger than the calibrated empirical quantile $1 - \hat{q}$. Assuming the narrations ℓ faithfully represent the outcomes of actions, we formally state the statistical assurance for our calibration procedure.

Theorem 1 (Verification Coverage Guarantee): Let $D_{\text{calib}} = \{\{(x_i^n, \mathcal{Y}_i^n)\}_{i=0}^M\}_{n=1}^N$ be the calibration dataset with non-conformity score:

$$\kappa^n := \kappa(\bar{x}^n, \bar{y}^n) = 1 - \min_{x_i \in \bar{x}^n} \min_{y \in \mathcal{Y}_i^n} \mathbb{P}^{\text{VLM}}(y \mid x_i^n)$$

and let $\hat{q}$ be the $\frac{\lceil (N+1)(1-\varepsilon) \rceil}{N}$–quantile of $\{\kappa^n\}_{n=1}^N$. For a test point x_i^{test} at any time when the VLM verifier is called $i \in [M]$, where the correct set of labels is $\mathcal{Y}_i^* \subseteq \mathcal{Y}$, define

$$C(x_i^{\text{test}}) = \{y : 1 - \mathbb{P}^{\text{VLM}}(y \mid x_i^{\text{test}}) \leq \hat{q}\}.$$

Then $\mathbb{P}(\mathcal{Y}_i^* \subseteq C(x_i^{\text{test}})) \geq 1 - \varepsilon$.

Proof. See Appendix A. We follow the sequence-level calibration proof from [21], but show that our non-conformity score includes all the ground-truth labels in our prediction set.

Shaping the VLM's Distribution *Before* CP. While in theory conformal prediction provides distribution-free coverage guarantees, the tightness of the resulting prediction sets depends on the non-conformity score. Our score in Eq. 3 directly depends on the VLM's distribution, $\mathbb{P}^{\text{VLM}}$. Naively querying a VLM for likelihoods over multiple choices results in significantly overconfident distributions, especially in ambiguous or incapable scenarios. Although CP can compensate for this, it leads to overly-large prediction sets that are not informative and result in the robot over-asking for help. Thus, the raw softmax probabilities from the logits of the VLM verifier are *not* an ideal way to inform our score function $\mathbb{P}^{\text{VLM}}$ from Eq. 3.

Our key idea is shape $\mathbb{P}^{\text{VLM}}$ even *before* calibration by *factorizing* the VLM's reasoning process such that the uncertainty is more explicit. Our factorization is inspired by Bayesian models of human intent [4, 20] which decouple inferring human intent (e.g., unstated aspects of the task) and the likelihood of behaviors given a specific intent:

$$\mathbb{P}^{\text{VLM}}(y \mid \{\ell^k\}_{k=1}^K; \mathcal{L}) = \sum_{\theta \in \Theta} \mathbb{P}(y \mid \{\ell^k\}_{k=1}^K, \theta) \mathbb{P}(\theta \mid \mathcal{L}). \tag{5}$$

Given the original task instruction $\mathcal{L}$ (e.g., "Put the block in my dominant-hand side hole"), we query the VLM three times to estimate each component of Eq. 5. First, we ask the model to hypothesize a set of hidden human intents $\theta \in \Theta$ (e.g., "Insert the block into with the Θ =[left/right] hole") alongside their respective probabilities, $\mathbb{P}(\theta \mid \mathcal{L})$. Then, we ask the VLM to infer the likelihood of each option, *given a hypothesized user intent*: $\mathbb{P}(y \mid \{\ell^k\}_{k=1}^K, \theta)$. The final distribution is obtained by marginalizing over the space of user intents.

Intuitively, this factorization leverages the commonsense reasoning abilities of the VLM in a structured way to re-calibrate $\mathbb{P}^{\text{VLM}}$ intrinsically even before non-conformity score computation. We find this to be especially useful in incapable scenarios or when one behavior mode is underrepresented in the action samples. For example, consider the scenario where all action samples insert the block into the right hole (i.e., ℓ^k = "Robot inserts the block into the back right hole", $\forall k \in \{1, \ldots, K\}$). Naively querying $\mathbb{P}^{\text{VLM}}$ results in an extremely biased distribution in favor of selecting the option with block in the right hole, ignoring the possibility that the user is left-handed. In our factorized approach, the VLM is asked to reason about hidden aspects of the task or user intents, and thus includes the possibility that the user is right *or* left-handed in Θ. This reshapes the final distribution over options, forcing the VLM to place non-trivial probability mass on the $K+1$th "none" option. In turn, this results in the reduced overconfidence of the VLM in incapable scenarios and enables the robot to ask for help only when necessary. In ambiguous scenarios, this same principle helps re-weight the distribution over several equally likely behaviors. In Sec. V, we show empirically that our factorization results in $\mathbb{P}^{\text{VLM}}$ that is substantially more calibrated even before conformal adjustment, and outperforms chain-of-thought reasoning (CoT) [28]. Ultimately, we obtain non-trivial prediction sets with high empirical coverage while also reducing the help rate.

C. Resolving Uncertainty: From High-level Clarifications to Low-Level Continual Learning

At deployment-time, the prediction sets constructed by our CP procedure provide the robot with a principled measure of both semantic task uncertainty and policy incapability, implying separate resolution strategies. Importantly, our statistical guarantee from Theorem 1 ensures that the VLM verifier can steer the low-level policy to success in $1-\epsilon$ proportion of scenarios by either executing the correct action sample or selecting the correct resolution strategy. We describe each strategy in detail below.

Confident and Capable Policy. If the prediction set is a singleton and contains an index corresponding to one of the low-level action plans, i.e. $|C(x_i^{\text{test}})| = 1$ and $K{+}1 \notin C(x_i^{\text{test}})$, then the steering system confidently executes this sequence of actions until the next verification step (bottom right, Fig. 1).

Resolving Task Uncertainty with Textual Clarifications. If the set size $|C(x_i^{\text{test}})| > 1$, the verifier has identified high-level semantic uncertainty in the task and asks a textual clarification question. For example, consider the right part of Fig. 1. Both inserting the block in the left hole and inserting into the right hole options are included in the prediction set when the user gives an ambiguous task description $\mathcal{L}$ ="*put the block in my dominant-hand side hole*". After asking the user to state their preferences about options, the VLM summarizes user's answer $\tilde{\mathcal{L}}$ and replaces the original ambiguous task instruction. This verify and clarify loop repeats and continues until $|C(x_i^{\text{test}})| = 1$. The right part of Fig. 1 demonstrates this process of resolving the task uncertainty.

Resolving Policy Incapability with Continual Learning. When the CP prediction set contains only "*none of the above*", i.e., $C(x_i^{\text{test}}) = \{K{+}1\}$, the VLM verifier detects that none of the action samples align with the user-requested task with high probability. This triggers a low-level resolution strategy via interactive imitation learning. Specifically, the robot randomly selects a trajectory and explicitly asks human supervision to correct potential failures, as shown in Fig 7. This approach improves upon standard interactive imitation learning methods [19, 11] in two ways. First, unlike human-gated methods [11] that demand constant monitoring of every trajectory, our robot only requests help when the policy is incapable in *all* samples. Second, unlike low-level action-uncertainty methods [19], our approach gates interventions based on semantic task alignment rather than low-level action variance; this avoids scenarios where low-level policy has noisy actions, but all result in good outcomes (i.e., the policy does not need to be re-trained).

We employ residual policy learning [24, 10] and train a lightweight model, $\Delta a \sim \pi^{\text{residual}}(\cdot \mid o, a)$, which predicts the difference between human corrections and base policy outputs alongside a gating classifier that determines when this correction is active. At re-deployment, we sample K actions from the frozen base policy and mix the residual policy into half of the samples (when triggered by the gating classifier), while half of the samples remain unmodified. This sampling strategy mitigates catastrophic forgetting by explicitly retaining the base policy's distribution. Finally, because the policy's action distribution shifts after residual policy learning, we recalibrate the VLM verifier, thereby closing the learning loop.

V. Simulation & Hardware Experiments

We evaluate our framework in both a simulation task and in robotic hardware with a Franka manipulator. In all our experiments, the base policy is an image-conditioned diffusion policy[1] [6] and the world model is Dreamer-v3 [8]. In both simulation and hardware **PnP Cup** task, we call the verifier twice ($M = 2$) throughout the task and use $M = 3$ for **Insert Block** task. We perform calibration at this sequence-level.

Simulation Setup. We use the Robomimic [16] benchmark, specifically the square nut-on-peg task. We chose this task because it requires high-precision and the human Robomimic demonstrations exhibit inherent diversity in *how* the nut is placed on the peg (e.g., handle facing left or right). We train our low-level diffusion policy with 120 demonstrations: 60 with the handle facing left and 60 facing right. We initially train the world model on 600 trajectories: the 120 demonstrations plus 480 rollouts from our base policy. We further fine-tune the model with 100 additional rollouts from the interleaved policy and world model prediction scheme.

Real World Setup. We use a Franka Emika robotic manipulator equipped with a Robotiq gripper and implement two real-world manipulation tasks: (1) **PnP Cup** has the robot picking a green cup and placing it into one of two bins (labeled "clean" and "dirty"), and (2) **Insert Block** has the robot inserting a purple table leg (block with circular peg) into the top left or top right holes of a tabletop. Images are perceived from a wrist camera and an external camera (see Appendix C). The second task is more challenging because it requires high insertion precision. Task uncertainty comes from the user's instructions as to where to place the cup or insert the block. For both tasks, the base policy is a diffusion policy that controls the cartesian pose and gripper. We train it with 100 demonstrations, covering two behavioral modes: placing the cup in the left/right bin or inserting the block into the left/right hole. The world model is trained on 350 trajectories: the 100 expert demonstrations plus 250 policy rollouts.

VLM Narration & Verification. We use the same set of VLM models in simulation and hardware. For narrating the imaginations, we use Gemini-3-flash-preview due to its strong video summarization capabilities. For verification, we use Gemini-2.0-flash because it exposes the token logits via the API for ablations of conformal prediction in Appendix E.

A. Uncertainty-Aware Steering

Calibration Dataset. We collect 80 pairs of initial observation o_0 and language instruction $\mathcal{L}$. Our tasks have multiple

[1]Although we focus on diffusion models, our framework can be applied to any base policy that represents a stochastic action distribution and from which diverse action samples can be drawn, e.g., a vision-language-action model [5].

phases $M = 2$ or 3 and we obtain new observation o_T after the each action sequence is executed, gathering data of all phases as our calibration set. 40 of these instructions are ambiguous, such as *"move the cup to a bin"*, and 40 are straightforward, such as *"can you move it to the dirty bin?"* for **PnP Cup** task. Among straightforward instructions, sometimes the policy does not generate any target actions, leading the scenario to be labeled incapable. From the observation, we obtain the behavior narrations for the imagined rollouts of $K = 10$ action samples. After narration, we group semantically identical narrations and randomly select a narration from each group to appear as a multiple choice option. We choose $1-\epsilon = 0.85$ for calibration. Details of instructions are in Appendix C.

Evaluation Dataset. Similar to our calibration dataset, we maintain a set of 40 samples in the test set sampled from the same distribution as the calibration dataset. During evaluation, we predict the set for each sample and compare that to the ground-truth set. If the ground-truth option(s) lie in the prediction set, we have achieved coverage. If the set size is larger than 1, we need clarification.

Metrics. We define three metrics commonly used in prior CP work to evaluate our calibrated VLM verifier. **Coverage** is the proportion of test samples whose ground-truth option(s) are included in the prediction set. The **Clarification rate** is the proportion of times that the prediction set is larger than 1 and thus the robot asks the human a question. The **Set size** is the size of the prediction set. For coverage, we want to achieve the desired coverage rate $1-\epsilon$ in all three scenarios. For the clarification rate, we want a high clarification rate in ambiguous cases and a low one in the other two. For the prediction set, we want the set size to be 2 for ambiguous cases and 1 for other cases.

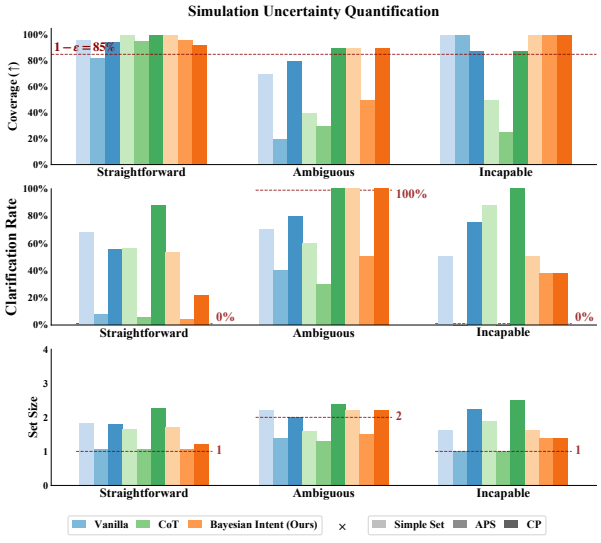


Fig. 3: **Uncertainty Quantification Results: Simulation**. We compare the combination of Vanilla, CoT and Bayesian Intent (Ours) models for UQ. Dashed lines indicate the target coverage ($1-\epsilon = 0.85$), clarification rate, and set size.

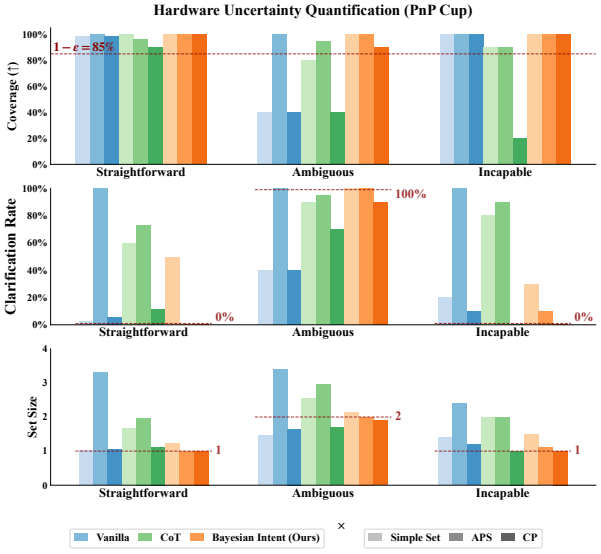


Fig. 4: **Uncertainty Quantification Results: Hardware (PnP Cup)**. Combination of Vanilla, CoT and Bayesian Intent (Ours) models for UQ. Dashed lines indicate the target coverage rate ($1-\epsilon = 0.85$), clarification rate, and set size.

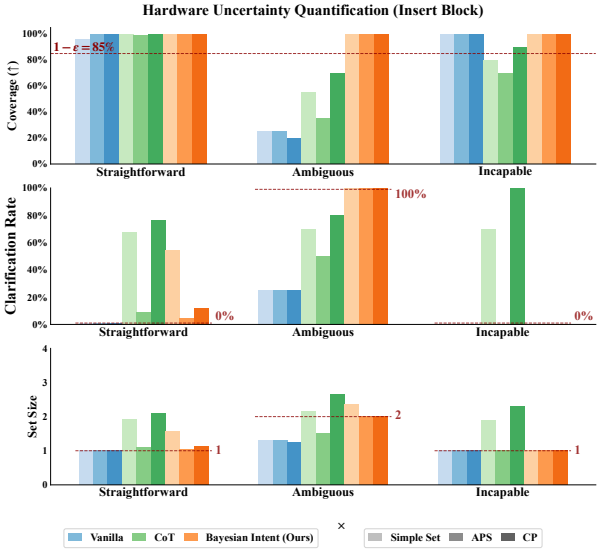


Fig. 5: **Uncertainty Quantification Results: Hardware (Insert Block)**. We compare the combination of Vanilla, CoT and Bayesian Intent (Ours) models for UQ. Dashed lines indicate the target coverage, clarification rate, and set size.

1) Our Score Function Balances Coverage & Clarification:

Methods. We evaluate three different UQ methods: Simple Set, APS [22], CP [21], and three different score functions including Vanilla, CoT and Bayesian Intent (Ours). This results in 3x3 methods we compare. **CP**: we adopt the same conformal prediction scheme from [21] but modify the score function to be Eq 3. **Simple Set**: sorts the options from high scores to low and adds the options until the sum exceeds $1-\epsilon$. **APS** sorts the options from high to low scores and adds the options until the sum exceeds $1-\hat{q}$. **Vanilla** uses the distribu-

tion $\mathbb{P}^{\text{VLM}}$ by asking the model to self-generate the probability scores between 0 and 1. **CoT** shapes $\mathbb{P}^{\text{VLM}}$ by first prompting the VLM to reason step-by-step and then, conditioned on the reasoning, generates the self-reported probability. **Bayesian Intent** computes intent-conditioned scores $\mathbb{P}^{\text{VLM}}$ as in Eq. 5.

Evaluation. We collect the evaluation dataset in Sec. V-A. When the policy is incapable under straightforward instruction, it is counted as an incapable scenario. Details of the dataset composition are in Appendix E.

Results. Across simulation and hardware, Fig. 3, Fig. 4 and Fig. 5 show that our proposed Bayesian intent score function with CP can achieve strong coverage ($\geq 85\%$) while minimizing the clarification rate compared to Vanilla and CoT. Our proposed Bayesian intent score function achieves the best balance between coverage and clarification rate across all three tasks. In contrast, CoT-based methods perform poorly due to conservative reasoning; they frequently overestimate ambiguity (leading to near 100% clarification rates) or exhibit sharp drops in coverage when set sizes are reduced (APS CoT). Similarly, Vanilla methods struggle with poor calibration, resulting in either overconfidence or severe undercoverage. This indicates that shaping the VLM's uncertainty via factorization leads to better calibration. With our intent-conditioned score function, CP strikes the best balance: it avoids Simple Set's overconservativeness in straightforward cases and APS's undercoverage in ambiguous ones.

2) Uncertainty Improves Policy Steering Performance: Next, we study if our well-calibrated VLM verifier (CP + Bayesian Intent) results in closed-loop performance improvements within our policy steering framework.

Methods. We compare three methods in this section including **Base Policy**, the low-level diffusion policy, **Forewarn** [30], an *uncalibrated* variant of VLM-in-the-loop policy steering, and **UPS w/ Clarification** which uses the calibrated threshold $\hat{q}$ to select the prediction set. When set size > 1, it communicates with user through Q & A and updates its task instructions accordingly to verify again as described in Sec. IV-C.

Evaluation. We take the same dataset of 40 samples as in Sec.V-A1. We compute a confusion matrix where true positive (TP) indicates the selected action sample leads to the correct outcome; true negative (TN) indicates the system properly elected to ask for help, false positive (FP) indicates the selected action actually fails; false negative (FN) indicates the system chose to ask for unnecessary help. We report the success rate as the number of true positives divided by total number of samples. The detailed analysis is in Appendix C and D.

Results. Fig. 6 shows that although Forewarn improves the success rate in both straightforward scenarios and ambiguous scenarios by steering the base policy to better align with the task instructions, it achieves much smaller gain in ambiguous cases in both tasks (e.g., in the **PnP Cup** task improvement in straightforward = 45%, ambiguous = 15%) because the VLM verifier is overconfident. In contrast, UPS + w/ Clarification addresses uncertainty in ambiguous cases, further increasing success rate by 15%. We note that with UQ and clarification, we can already achieve the desired coverage rate of 85% in straightforward cases for the **PnP Cup** task.

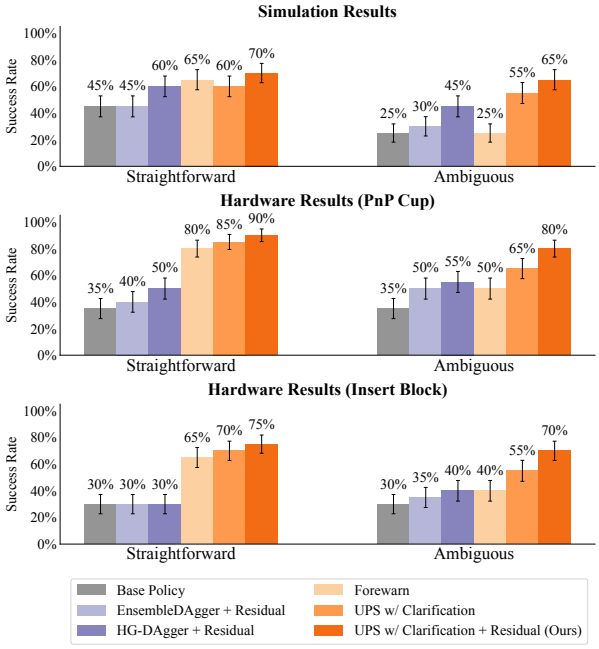


Fig. 6: **Success Rates Pre- and Post-Continual Learning: Hardware and Simulation.** We deploy the robot with 20 straightforward (left) and 20 ambiguous (right) instructions. We average the success rate over 20 trials for each scenario. UPS solicits data in a way which maximizes the final success rate after residual policy training, compared to baselines.

B. Continual Learning

Finally, we close the loop and evaluate how our uncertainty-aware steering method informs low-level data collection for improving the base policy, and how this influences human intervention rate and re-deployment performance.

1) Semantic-level UQ Minimizes Human Feedback:

Evaluation. Similar to Sec. V-A2, we evaluate the policy for 40 trials where each trial has different initial condition and different language instructions. During intervention, the human corrects the robot by controlling the end-effector and gripper through a teleoperation device (e.g., SpaceMouse). We use the same success rate metric as in Sec. V-A2.

Methods. We compare our proposed high-level semantic guided human interventions **UPS + w/ Clarification + Residual** against two foundational approaches, shown in Fig. 7: **HG-DAgger** [11], where a human intervenes whenever they see fit, and **EnsembleDAgger** [19], which asks for human interventions when the policy ensemble disagreement is high.

Metrics. For each trajectory, we divide the number of human intervention steps by trajectory length, and then we average this normalized rate across all trajectories to measure overall

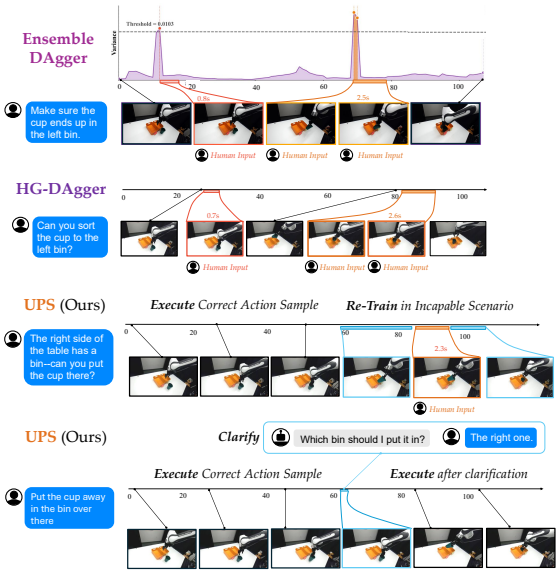


Fig. 7: **Hardware: Robot Asking for Interventions.** EnsembleDAgger (top): the human intervenes whenever the ensemble disagreement exceeds a threshold. Human-Gated (HG) DAgger (middle): a human monitors and intervenes when the robot's behavior deviates from their intention. Our approach (bottom two) asks for "cheap" clarifications and only requests interventions when no action sample achieves the task.

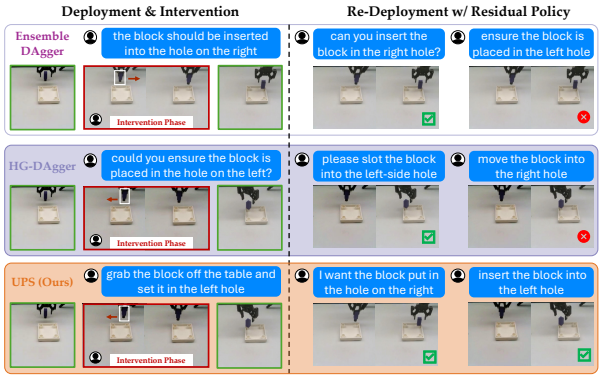


Fig. 8: **Intervention & Re-deployment for Insert Block Task**. (left) Different strategies to elicit human interventions: HG-DAgger (top), EnsembleDAgger (middle) and UPS (bottom). (right) UPS maintains the multi-modality, while other methods exhibit failures or mode collapse.

human effort. When it asks a clarification question *without* low-level interventions, we count this as one human intervention step since Q&A is easier than physical intervention.

Results. **UPS** has the lowest human intervention rate because it asks for low-level intervention data only when the policy cannot generate a low-level behavior that is "semantically aligned" with the task. Our method achieves significantly lower intervention rates than the baselines in both settings. For **PnP Cup** task, UPS requires interventions at a rate of 0.06, compared to 0.1 for HG-DAgger and 0.16 for EnsembleDAgger, whose low-level uncertainty quantification demands even more human assistance. Similarly, UPS' intervention rate is 0.05, compared to HG-DAgger's rate of 0.06 and EnsembleDAgger's rate of 0.26 in **Insert Block**. In simulation, the gap widens: UPS's rate is 0.058, while HG-DAgger has an intervention rate of 0.20 and EnsembleDAgger has 0.27.

2) UPS Improves Re-Deployment Performance:

Residual Policy Training. For fair comparison among different intervention methods, we fix an intervention budget with the same number of intervention trajectories for all methods. We compute the delta action between the intervention and the base action and train the residual policy π^{residual} with observation and base action as inputs, as shown in Appendix F.

Methods. Similar to Sec. V-B1, we learn a residual policy from interventions collected with different methods: **EnsembleDAgger + Residual**, **HG-DAgger + Residual** and **UPS + w/ Clarification + Residual**. We also compare to base policy without the residual with the metric described in Sec. V-A2.

Results. Fig. 6 shows that our residual policy continues to improve from **UPS w/ Clarification** only because it reduces the proportion of incapable scenarios; it obtains a 15% increase in success rate in ambiguous settings for both hardware tasks. Across both types of scenarios, our final success rate is 85% for **PnP Cup** task. This empirical result corroborates Theorem 1, demonstrating that the verifier can steer the low-level policy to success in $1-\epsilon$ of scenarios by either executing the right sample or selecting the appropriate strategy (asking clarification questions when ambiguous or eliciting human interventions when incapable). In simulation and **Insert Block** task, we achieve slightly lower coverage due to the challenges of simulating precise contact-rich manipulation tasks with the current world model. On the other hand, both baseline DAgger methods struggle to learn new behaviors from intervention data while maintaining the original capabilities, as shown in Fig. 8. We hypothesize the learned residual policy may bias the final policy distribution towards certain behavior modes of doing the task, or may push the model towards new incapable modes.

VI. Conclusion & Limitations

In this work, we propose an uncertainty-aware policy steering system which rigorously calibrates a verifier's uncertainty over low-level action samples. Our approach also resolves uncertainty with high-level clarifications or low-level re-training to achieve overall success rate in $1-\epsilon$ of scenarios. Since we focus on uncertainty quantification of VLM verifiers, we assumed the imaginations from world model always match future outcomes and behavior narrations are correct and complete. Interesting future work incorporates the uncertainty of the world model (and narrations) [17], for more robust system.

VII. Acknowledgements

The authors were partially supported by the National Science Foundation (NSF) award [#2246447], NSF CAREER award [#2441014], and Samsung Research through the LEAP-U program. The views expressed are those of the authors and do not necessarily reflect those of NSF or Samsung. We thank Junwon Seo and Michelle Zhao for helpful discussions.

References

[1] Moises Andrade, Joonhyuk Cha, Brandon Ho, Vriksha Srihari, Karmesh Yadav, and Zsolt Kira. Let's think in two steps: Mitigating agreement bias in mllms with self-grounded verification. In *International Conference on Learning Representations (ICLR)*, 2026.

[2] Anastasios N Angelopoulos, Stephen Bates, et al. Conformal prediction: A gentle introduction. *Foundations and trends® in machine learning*, 16(4):494–591, 2023.

[3] Zechen Bai, Pichao Wang, Tianjun Xiao, Tong He, Zongbo Han, Zheng Zhang, and Mike Zheng Shou. Hallucination of multimodal large language models: A survey. *arXiv preprint arXiv:2404.18930*, 2024.

[4] Chris L Baker, Joshua B Tenenbaum, and Rebecca R Saxe. Goal inference as inverse planning. In *Proceedings of the annual meeting of the cognitive science society*, volume 29, 2007.

[5] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0 : A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[6] 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*, 2024.

[7] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, New York, New York, USA, 20–22 Jun 2016. PMLR.

[8] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *Nature*, 2023.

[9] Ryan Hoque, Ashwin Balakrishna, Ellen Novoseller, Albert Wilcox, Daniel S Brown, and Ken Goldberg. Thriftydagger: Budget-aware novelty and risk gating for interactive imitation learning. In *5th Annual Conference on Robot Learning*, 2021.

[10] Yunfan Jiang, Chen Wang, Ruohan Zhang, Jiajun Wu, and Li Fei-Fei. Transic: Sim-to-real policy transfer by learning from online correction. In *Conference on Robot Learning*, pages 1691–1729. PMLR, 2025.

[11] Michael Kelly, Chelsea Sidrane, Katherine Driggs-Campbell, and Mykel J Kochenderfer. Hg-dagger: Interactive imitation learning with human experts. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8077–8083. IEEE, 2019.

[12] Jacky Kwok, Christopher Agia, Rohan Sinha, Matt Foutter, Shulu Li, Ion Stoica, Azalia Mirhoseini, and Marco Pavone. Robomonkey: Scaling test-time sampling and verification for vision-language-action models. *arXiv preprint arXiv:2506.17811*, 2025.

[13] Justin Lidard, Hang Pham, Ariel Bachman, Bryan Boateng, and Anirudha Majumdar. Risk-calibrated human-robot interaction via set-valued intent prediction. *Robotics: Science and Systems*, 2024.

[14] Huihan Liu, Yu Zhang, Vaarij Betala, Evan Zhang, James Liu, Crystal Ding, and Yuke Zhu. Multi-task interactive robot fleet learning with visual world models. In *8th Annual Conference on Robot Learning*, 2024.

[15] Yuxuan Liu, Tianchi Yang, Shaohan Huang, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, and Qi Zhang. Calibrating llm-based evaluator. In *Proceedings of the 2024 joint international conference on computational linguistics, language resources and evaluation (lrec-coling 2024)*, pages 2638–2656, 2024.

[16] 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 *arXiv preprint arXiv:2108.03298*, 2021.

[17] Zhiting Mei, Tenny Yin, Micah Baker, Ola Shorinwa, and Anirudha Majumdar. World models that know when they don't know: Controllable video generation with calibrated uncertainty. *arXiv preprint arXiv:2512.05927*, 2025.

[18] Zhiting Mei, Christina Zhang, Tenny Yin, Justin Lidard, Ola Shorinwa, and Anirudha Majumdar. Reasoning about uncertainty: Do reasoning models know when they don't know? *arXiv preprint arXiv:2506.18183*, 2025.

[19] Kunal Menda, Katherine Driggs-Campbell, and Mykel J Kochenderfer. Ensembledagger: A bayesian approach to safe imitation learning. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5041–5048. IEEE, 2019.

[20] James F Mullen and Dinesh Manocha. Lbap: Improved uncertainty alignment of llm planners using bayesian inference. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 18716–18723. IEEE, 2025.

[21] Allen Z Ren, Anushri Dixit, Alexandra Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila Takayama, Fei Xia, Jake Varley, et al. Robots that ask for help: Uncertainty alignment for large language model planners. *Conference on Robot Learning*, 2023.

[22] Yaniv Romano, Matteo Sesia, and Emmanuel Candes. Classification with valid and adaptive coverage. *Advances in neural information processing systems*, 33: 3581–3591, 2020.

[23] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[24] Tom Silver, Kelsey Allen, Josh Tenenbaum, and Leslie Kaelbling. Residual policy learning. *arXiv preprint arXiv:1812.06298*, 2018.

[25] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[26] Pablo Valle, Chengjie Lu, Shaukat Ali, and Aitor Arrieta. Evaluating uncertainty and quality of visual language action-enabled robots. *arXiv preprint arXiv:2507.17049*, 2025.

[27] Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*. Springer, 2005.

[28] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[29] Yilin Wu, Anqi Li, Tucker Hermans, Fabio Ramos, Andrea Bajcsy, and Claudia P'erez-D'Arpino. Do what you say: Steering vision-language-action models via runtime reasoning-action alignment verification. *arXiv preprint arXiv:2510.16281*, 2025.

[30] Yilin Wu, Ran Tian, Gokul Swamy, and Andrea Bajcsy. From foresight to forethought: Vlm-in-the-loop policy steering via latent alignment. *Robotics: Science and Systems*, 2025.

[31] Wenli Xiao, Haotian Lin, Andy Peng, Haoru Xue, Tairan He, Yuqi Xie, Fengyuan Hu, Jimmy Wu, Zhengyi Luo, Linxi Fan, et al. Self-improving vision-language-action models with data generation via residual rl. *arXiv preprint arXiv:2511.00091*, 2025.

[32] Michelle D. Zhao, Henny Admoni, Reid Simmons, Aaditya Ramdas, and Andrea Bajcsy. Conformalized interactive imitation learning: Handling expert shift and intermittent feedback. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=Ym2RNPX6la.

[33] Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. Dino-wm: World models on pre-trained visual features enable zero-shot planning. In *Forty-second International Conference on Machine Learning*, 2025.

[34] Thomas P Zollo and Richard Zemel. Confidence calibration in vision-language-action models. *arXiv preprint arXiv:2507.17383*, 2025.