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

ReSteer: Quantifying and Refining the Steerability of Multitask Robot Policies

Zhenyang Chen†, Alan Tian*, Liquan Wang*, Benjamin Joffe, Yingyan Celine Lin,
Yuxiao Chen, Siddharth Karamcheti‡, Danfei Xu‡
Georgia Institute of Technology *Equal contribution ‡Equal advising
resteer-vla.github.io

***Abstract*—Despite strong multi-task pretraining, existing policies often exhibit poor task steerability. For example, a robot may fail to respond to a new instruction "put the bowl in the sink" when moving towards the oven, executing "close the oven", even though it can complete both tasks when executed separately. We propose ReSteer, a framework to quantify and improve task steerability in multitask robot policies. We conduct an exhaustive evaluation of state-of-the-art policies, revealing a common lack of steerability. We find that steerability is associated with limited overlap among training task trajectory distributions, and introduce a proxy metric to measure this overlap from policy behavior. Building on this insight, ReSteer improves steerability via three components: (i) a steerability estimator that identifies low-steerability states without full-rollout evaluation, (ii) a steerable data generator that synthesizes motion segments from these states, and (iii) a self-refinement pipeline that improves policy steerability using the generated data. In simulation on LIBERO, ReSteer improves steerability by 11% over 18k rollouts. In real-world experiments, we show that improved steerability is critical for interactive use, enabling users to instruct robots to perform any task at any time. We hope this work motivates further study on quantifying steerability and data collection strategies for large robot policies.**

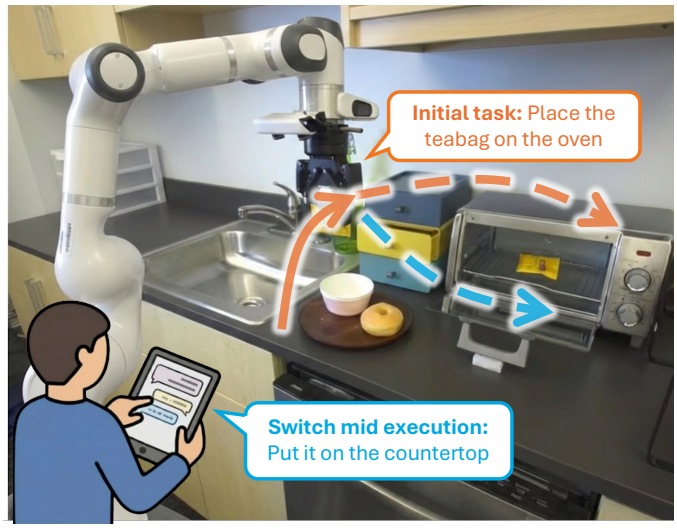

Fig. 1. Daily manipulation is diverse and time-critical, and often requires **interruptible** behavior, with users revising their intent after execution begins. This demands **interactive, steerable** policies that can switch behaviors **from any intermediate state** in response to a new task prompt. We propose *ReSteer*, a framework to quantify and improve the steerability of multitask robot policies.

I. Introduction

Language-conditioned multitask robot policies such as Vision-Language-Action (VLA) models [1–7] have emerged as a promising paradigm for robotic control, unifying language-specified goals and visual understanding. These policies use language instructions as an intuitive and flexible task representation, leveraging pretrained multimodal backbones [8] to encode grounded priors about the physical world. While these models have demonstrated immense capability, fitting dozens of tasks [3, 5], generalizing across new embodiments [1, 4], and unlocking different types of embodied reasoning [4, 9], a key failure mode remains: *steerability*.

Steerability captures a policy's ability to properly condition on new task instructions in arbitrary states. Consider the example in Fig. 1, where we initially instruct the robot to put the tea bag on the oven, then ask it to place it on the countertop. Existing policies [1, 2, 4] tend to ignore the new instruction and continue executing the original task, operating as if they were conditioned on state alone. This deficit is particularly problematic as it prevents users from issuing new instructions during ongoing execution, severely restricting when and where a robot can be meaningfully controlled.

In this work, we argue that a policy's steerability is a function of its training data distribution, with most existing datasets exhibiting pathologies that encourage learned models to ignore task conditioning. Concretely, existing datasets are collected subject to the following recipe: we identify a set of tasks, then collect demonstrations on each task independently. This process yields data where language is only informative for a small fraction of total task execution, leading to policies that over-index on state information to dictate behavior, a form of shortcut learning [10]. As an initial contribution, we formalize this intuition via conditional mutual information (CMI): given a dataset and learned policy, steerability can be approximated by $I(A; L|S)$. In other words, incorporating language prompts reduces the variance of the predicted action distribution, indicating that instructions meaningfully constrain action selection. Empirically, we validate our proxy metric through experiments across multiple state-of-the-art VLAs, showing a strong correlation between our proposed metric and "steerable" success rate.

Motivated by this diagnosis, we argue that incorporating data where the task instruction changes mid-execution—eliciting corresponding behavioral switches—is critical for improving the steerability of the learned policy. To this end, we introduce *ReSteer*, a simple yet effective data generation framework that

comprises two stages: *stage-aware steering data generation* and *self-refinement*. In stage-aware steering data generation, for each switch from task A to task B, we (i) use CMI to identify low-steerability states along task A rollouts, (ii) pick a target state from task B, and (iii) synthesize a transition trajectory between the two for training. While this first stage introduces new task-switching trajectories, it does not always guarantee reliable execution, and the steering success rate can remain low. To address this gap, we subsequently apply a self-refining behavioral cloning stage, where the policy is iteratively deployed to collect rollouts and finetuned solely on its own successful trajectories, thereby reinforcing in-distribution steerability induced by the newly injected steering data. By combining these two stages, *ReSteer* provides an efficient way to expand the steerability coverage and make steering behavior robust.

We evaluate *ReSteer* through comprehensive experiments in both simulation and the real world. In simulation on the LIBERO benchmark [11], we observe that baseline policies exhibit limited steerability, with task switching feasible only early in execution and substantial performance degradation as the rollout progresses. In contrast, *ReSteer* achieves a **11%** absolute improvement in successfully steering among the 10 benchmark tasks from diverse intermediate states. These gains indicate that *ReSteer* improves language steerability and makes mid-execution task switching more reliable in practice. To isolate the contribution of each component of *ReSteer*, we perform controlled ablations to analyze how sampling strategy and dataset size affect performance. Furthermore, we conduct extensive real-world experiments in a daily kitchen scene, where *ReSteer* improves steerability by $\mathbf{2.2}\times$ compared to a policy fine-tuned only on teleoperation data. Lastly, we show that *ReSteer*-enhanced policies can be chained to execute long-horizon task sequences. For example, a user can instruct the robot to switch from closing the oven to put the tea bag in the drawer, and then switch to pick up a bowl afterward (see Fig. 8). This flexibility enables more interactive and reactive robotic assistants in real-world deployments. Across our experiments, *ReSteer* improves steerability, enabling more reactive multitask robot policies that better support human interaction. We argue that quantifying and understanding steerability is important for reliable policy deployment, and *ReSteer* offers a simple, practical way to do so. We will release all code, data, and models to reproduce our results.

II. Related Work

A rich body of work studies learning multitask, language-conditioned policies for robot control [12, 13]. While early approaches typically focus on following a fixed, task-specific instruction set, recent VLA policies are pretrained at scale and can generalize zero-shot to a much broader range of natural-language commands.

From Language Conditioning to Interactive Prompting. Multitask policies such as BC-Z and RT-1/RT-X [5, 14, 15] demonstrate that large-scale language-conditioned policies can solve many manipulation tasks from diverse instructions. More recent works have targeted language following more directly. Interleave-VLA [16] re-labels Open-X-Embodiment [15] with interleaved image–text instructions to improve grounding of referring expressions. ChatVLA/ChatVLA-2 [17, 18] jointly train a VLM head and an action head to retain open-world reasoning while producing instruction-conditioned actions. In interactive systems (e.g., Hi-Robot [19], Yell-at-your-robot [20], Robix [21]), a VLM decomposes high-level user requests into short prompts for a downstream VLA. SwitchVLA [22] studies switching by conditioning on execution context, but relies on segmented phases and engineered behavior labels, resetting to pre-contact stages and resulting in discontinuous motion. CAST [23] synthesizes counterfactual language for similar observations, improving navigation instruction following by increasing semantic diversity. A complementary line of work targets steerability at inference time: ITPS [24] selects among behaviors already present in the learned action distribution via human interaction, while DAgger-style approaches [25] collect on-policy supervision along trajectories of the *same* task. In contrast, ReSteer focuses on *where* and *how* to construct cross-task supervision: it identifies low-steerability states and pairs them across tasks to expand the training distribution, so that the behaviors required for mid-episode switching are learnable in the first place. Overall, these methods either presume robust primitive instruction following or restrict switching to predefined modes; in contrast, we *measure* and *improve* language steerability under mid-episode prompt changes via targeted data generation and self-refinement.

Steerability and Reasoning Beyond Language. Most existing multitask approaches categorize control through two primary lenses: *latent-space manipulation* and *reasoning-based planning*. Latent-space methods utilize pretrained language-driven embeddings or internal latents [26, 27] to interpolate between behaviors. While effective for representation transfer [28–30], these are often limited to predefined skill boundaries or treated as static inputs. Parallel work [4, 9, 31, 32] in reasoning-based planning augments policies with textual plans or spatial traces to improve execution. However, both directions generally assume a single fixed intent per rollout. Because they treat language as a static episode-level input or restrict task updates to predefined phase boundaries, they lack a mechanism for high-frequency, user-driven redirection from arbitrary intermediate states. *ReSteer* addresses these drawbacks by treating steerability as an instant, state-dependent interruption, enabling a policy to remain interactive and responsive to intent changes at any point during execution.

Our Perspective: Steerability of Multitask Policies. Later work on reasoning, interactive prompting, counterfactual relabeling, task switching, and text-latent control improves robustness and usability, but often assumes a single intent per episode, limits updates to predefined phase boundaries, or depends on external planners and engineered state machines. In contrast, we target *steerability*: a policy should be interruptible and re-steerable by language at any execution state, without fixed task boundaries or hand-designed switching modes. Accordingly, we treat language following as a state-dependent

property of the closed-loop policy and introduce evaluation protocols and training mechanisms designed specifically for this continuous notion of steerability.

III. Quantifying Steerability

A. Definitions

Consider an MDP with state space $\mathcal{S}$, action space $\mathcal{A}$, and transition dynamics $\mathcal{P}$. We assume a set of tasks $k \in \mathcal{K}$, with corresponding language instruction l_k. Let $\pi(a \mid s, \ell)$ denote a language-conditioned policy that outputs an action distribution given state s and instruction ℓ. For a fixed instruction l_k and initial state distribution $p(s_0)$, the closed-loop execution of π induces a trajectory distribution $p_k^\pi(\tau \,|\, s) \triangleq p_\pi\big(s_0, a_0, s_1, a_1, \ldots, s_T \mid l_k\big), \quad \tau = (s_0, a_0, \ldots, s_T)$, where $a_t \sim \pi(\cdot \mid s_t, l_k)$.

We focus on pretrained policies that already exhibit strong multitask performance under fixed instructions. Let $\mathbb{I}_k(\tau) \in \{0, 1\}$ denote the task-k success indicator, and define the *success probability from state s* as

$$P_k^\pi(s) \;\triangleq\; \Pr_{\tau \sim p_\pi(\cdot|s_0=s,\, l_k)} \big[\mathbb{I}_k(\tau) = 1\big] \tag{1}$$

We assume an imitation learning setting, where expert demonstrations are collected under instruction l_k, yielding trajectories $\{\tau^{(i)}\}_{i=1}^{N_k}$ with N_k demos. We construct the imitation dataset as the collection of state-action pairs along the recorded trajectories: $D_k \;\triangleq\; \Big\{(s_t^{(i)}, a_t^{(i)})\Big\}_{i=1,\ldots,N_k;\; t=0,\ldots,T_i-1}$. Let μ_k denote the empirical distribution over states visited in the expert demonstrations of task k. We assume the existence of thresholds $\alpha \in (0, 1)$ and $\delta \in [0, 1)$ such that

$$\Pr_{s \sim \mu_k} \big[P_k^\pi(s) \geq \alpha\big] \;\geq\; 1 - \delta, \qquad \forall k \in \mathcal{K} \tag{2}$$

This assumption ensures that the policy is competent at executing tasks from in-distribution states, and that failures observed under mid-execution instruction changes are attributable to limited steerability rather than poor single-task execution.

State Visitation Sets. Given a policy π and task k, we define the *policy-induced state visitation set* as the set of states from which the policy can successfully complete task k with high probability:

$$\mathcal{S}_k(\pi) \;\triangleq\; \Big\{s \in \mathcal{S} \;\Big|\; P_k^\pi(s) \geq \alpha\Big\}, \tag{3}$$

where $P_k^\pi(s)$ denotes the task-k success probability of policy π when initialized from state s, and $\alpha \in (0, 1)$ is the success threshold defined previously.

Union of Task Visitation Sets. For a task pair $i, j \in \mathcal{K}$ and policy π, we define the *union of task visitation sets* as

$$\mathcal{S}_{i,j}^{\text{union}}(\pi) \;\triangleq\; \mathcal{S}_i(\pi) \cup \mathcal{S}_j(\pi). \tag{4}$$

A state s belongs to $\mathcal{S}_{i,j}^{\text{union}}(\pi)$ if the policy can successfully complete *at least one* of the two tasks when initialized from s under the corresponding instruction. The union $\mathcal{S}_{i,j}^{\text{union}}(\pi)$ characterizes overall task feasibility, while the intersection $\mathcal{S}_{i,j}^{\text{inter}}(\pi)$ identifies states where both tasks are individually achievable and where steering between tasks is well-defined.

Steerable States. For a task pair (i, j) and policy π, we first define *directional steerability*. A state $s \in \mathcal{S}$ is said to be *steerable from task i to task j* if, when the policy is executing task i and the language instruction is switched to l_j at state s, the policy can still complete task j with high probability. Formally, define the post-switch success probability

$$P_{\to j}^\pi(s) \;\triangleq\; \Pr_{\tau \sim p_\pi(\cdot|s_0=s,\, l_j)} \big[\mathbb{I}_j(\tau) = 1\big]. \tag{5}$$

We call s steerable to j if $P_{\to j}^\pi(s) \geq \alpha$. The set of directionally steerable states from i to j is

$$\mathcal{S}_{i \to j}^{\text{steer}}(\pi) \;\triangleq\; \Big\{s \in \mathcal{S}_i \;\Big|\; P_{\to j}^\pi(s) \geq \alpha\Big\}. \tag{6}$$

We further define *bidirectional steerability* as:

$$\mathcal{S}_{i \leftrightarrow j}^{\text{steer}}(\pi) \;\triangleq\; \mathcal{S}_{i \to j}^{\text{steer}}(\pi) \;\cap\; \mathcal{S}_{j \to i}^{\text{steer}}(\pi). \tag{7}$$

Steerability Coverage Ratio (SCR). While bidirectional steerability characterizes whether instruction switching is possible at a given state, we seek a policy-level metric that quantifies how broadly such steering behavior is supported across the policy's feasible state space.

We define the *steerability coverage ratio (SCR)* for a task pair (i, j) and policy π as

$$\text{SCR}_{i \leftrightarrow j}(\pi) \;\triangleq\; \frac{\big|\mathcal{S}_{i \leftrightarrow j}^{\text{steer}}(\pi)\big|}{\big|\mathcal{S}_{i,j}^{\text{union}}(\pi)\big|}. \tag{8}$$

The denominator represents the full set of states at which the policy can successfully execute at least one of the two tasks, and therefore remains fixed under steerability-improving training. A higher SCR indicates that the policy can respond to task changes across a broader range of states during execution.

Data Dependence of Steerability. In standard multitask datasets, each trajectory is executed under a single fixed instruction. Let $\mathcal{D}$ denote the training dataset, and let $\pi_\mathcal{D}$ denote the policy obtained after training on $\mathcal{D}$. Such data primarily determines the set of states from which the trained policy can successfully execute at least one task. Formally, the dataset induces the *union of task visitation sets* under the trained policy:

$$\mathcal{S}_{i,j}^{\text{union}}(\pi_\mathcal{D}) \;\triangleq\; \mathcal{S}_i(\pi_\mathcal{D}) \;\cup\; \mathcal{S}_j(\pi_\mathcal{D}), \tag{9}$$

However, because $\mathcal{D}$ contains only single-instruction trajectories, it provides no instruction-contrasting supervision at the same underlying state. As a result, although $\mathcal{S}_{i,j}^{\text{union}}(\pi_\mathcal{D})$ may be large, the bidirectionally steerable set $\mathcal{S}_{i \leftrightarrow j}^{\text{steer}}(\pi_\mathcal{D})$ can remain small, leading to poor steerability coverage.

Improving steerability therefore requires augmenting $\mathcal{D}$ with *steering data* that explicitly injects paired supervision at shared execution states. Concretely, for a state s_t, we add samples of the form $(s_t, a_t^{(i)}, l_i)$ and $(s_t, a_t^{(j)}, l_j)$, which expose how actions should differ under the two instructions at the same state. This instruction-contrasting augmentation enables the policy to learn task-dependent switching behavior under mid-execution prompt changes, thereby expanding $\mathcal{S}_{i \leftrightarrow j}^{\text{steer}}(\pi_\mathcal{D})$.

B. Evaluation of Steerability

To evaluate steerability, we quantify the ability of a policy to complete any target task τ_j (specified by prompt l_j) from any trained states from task τ_i. A rollout is marked as successful if the policy, when conditioned on the provided new prompt l_j, completes the corresponding task t_j according to the task's success metric. To evaluate policy steerability at each task, we sample different timesteps to switch different task prompts during policy execution. At each sampled state, we prompt the policy with all task prompts from the dataset. In our experiment, we test each (state, prompt) pair with 10 rollouts ($n_{repeat} = 10$) to account for policy stochasticity. The whole evaluation requires $n \times (n-1) \times |k_i| \times n_{repeat}$ number of rollouts, where k_i is the number of sampled states to steer from at a certain originally executing task τ_i. Even though expensive, it provides a precise and exhaustive way to quantify steerability and directly matches the definition of steerability. We use this metric to compare against other proxies and benchmark different multitask policies.

C. Case Study: Evaluating the Steerability of Multitask Policies

Using steerability evaluation, we analyze state-of-the-art VLA policies and reveal state-dependent failures to respond to instruction changes, motivating targeted data generation. We quantify steerability for representative open-source VLA models: $\pi_{0.5}$ [1], OpenVLA-OFT [33], and MolmoACT [4]. For each model, we evaluate steerability by switching the language instruction mid-execution while keeping the environment state in-distribution, yielding a controlled *in-distribution steering* setting. Because we focus on cross-task steering within a shared scene, we use LIBERO-Goal as the benchmark, a commonly used small-scale multi-task dataset. Although all three VLAs achieve saturated single-task performance on LIBERO-Goal (over 90% success rate), none exhibits reliable cross-task steering under prompt switches, as shown in I. This reveals a lack of *in-domain* steerability in current VLAs, consistent with policies that overfit to single-task behaviors despite high nominal success.

	OpenVLA-OFT	**MolmoACT**	$\boldsymbol{\pi_{0.5}}$
Steerability Score	0.252	0.295	0.403

TABLE I
VLAs STEERABILITY COMPARISON ON LIBERO.

IV. ReSteer: A Data-Centric Framework for Refining Steerability

To improve the steerability coverage ratio (SCR), we show in the previous section that additional data is necessary. We therefore propose *ReSteer*, a framework that progressively expands the steerable set through targeted data generation and policy refinement.

As illustrated in Fig. 4, ReSteer first performs stage-aware steering data generation, which introduces instruction-contrasting transitions at compatible execution states. This expands the steerable sets $\mathcal{S}^{\text{steer}}_{i \to j}(\pi)$ and increases the portion

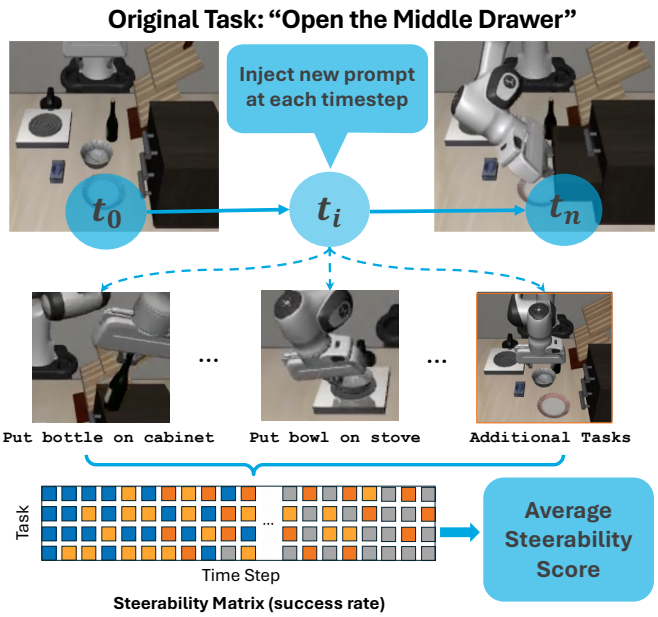

Fig. 2. Steerability evaluation. We run an original task i, then at sampled timesteps t we switch the language prompt to a target task j and measure whether the policy completes j from that state. Repeating this over all sampled states and all target tasks yields a steerability matrix (time×target task), whose average success rate gives the overall steerability score in task i.

of feasible states where task switching is possible. However, exhaustively generating steering data for all task pairs and states is prohibitive, requiring $\mathcal{O}(n^2 k_i n_r)$ rollouts, where n is the number of tasks, k_i denotes the number of states visited under task i, and n_r is the number of rollout trials per state. To address this, *ReSteer* leverages conditional mutual information (CMI) between action and language as a principled signal to identify states with low language–action coupling, where steering data is most needed.

Following this, *ReSteer* applies self-refining behavior cloning (SRBC) to finetune the policy on its own successful steering rollouts. As shown in Fig. 4 (Right), this refinement enlarges the steerable region within the same feasible state space, further increasing SCR by strengthening in-distribution steering.

A. SteerGen: Stage-Aware Steering Data Generation

Prior work mitigates poor language steering through counterfactual data generation [23], but has primarily been demonstrated in navigation settings and does not directly extend to long-horizon, contact-rich manipulation. Inspired by recent data-generation pipelines [34, 35], we instead exploit task-stage structure to synthesize cross-task steering motions. Our *stage-aware* procedure (i) segments demonstrations into semantic stages; (ii) samples a start state from a source task and selects a stage-matched end state from a target task by minimizing a shortest-path cost that favors short, smooth transitions; and (iii) generates a feasible interpolation segment via motion planning.

We emphasize that the ReSteer formulation and the CMI-based quantification in Sec. III apply to arbitrary states and do not rely on any task segmentation. The stage-aware procedure described below is one concrete instantiation that produces

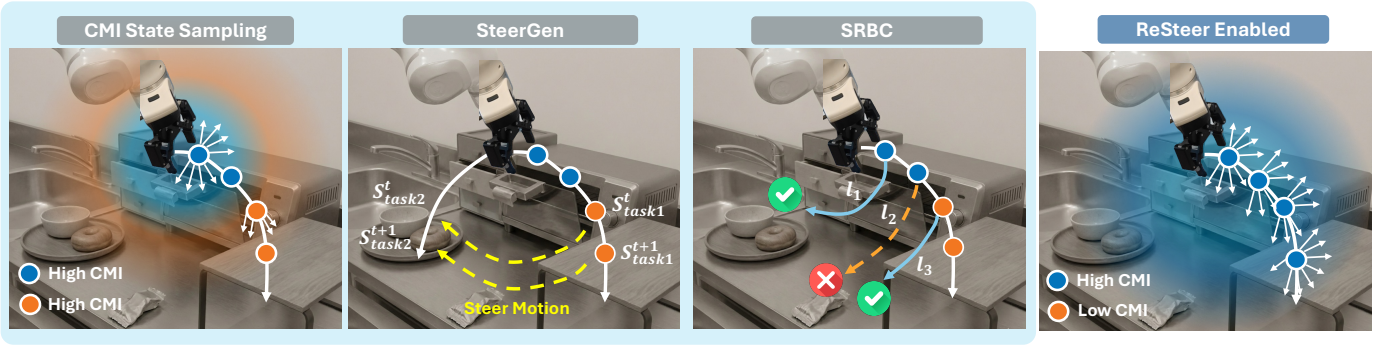

Fig. 3. We propose an online learning framework to improve the steerability of multitask policies. The framework comprises three components. First, we propose a CMI-based state sampling strategy that prioritizes data collection at the most unsteerable states, improving the sample efficiency of data generation. Second, we introduce a stage-aware steering data generation pipeline that synthesizes cross-task steering motions ($s^t_{task1} \rightarrow s^t_{task2}$), thereby expanding the steerable states, $\mathcal{S}^{\text{steer}}_{i \leftrightarrow j}$. Third, we develop a self-refining behavior cloning (SRBC) scheme that finetunes the policy using successful steered trajectories given different instructions l and consequently increases steerability coverage ratio $\text{SCR}_{i \leftrightarrow j}(\pi)$.

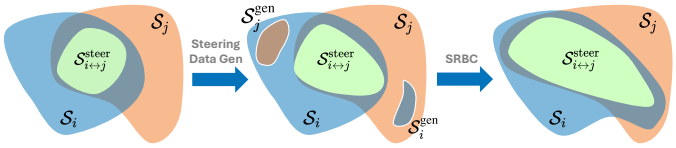

Fig. 4. Illustration of the steerability improvement afforded by ReSteer. Left: the bidirectionally steerable set $\mathcal{S}^{\text{steer}}_{i \leftrightarrow j}$ (green) occupies only a small subset of the policy-induced feasible states $\mathcal{S}_i$ and $\mathcal{S}_j$. Middle: stage-aware steering data generation ($\mathcal{S}^{\text{gen}}_i$ and $\mathcal{S}^{\text{gen}}_j$) introduces instruction-contrasting transitions, expanding the steerable region. Right: self-refining behavior cloning (SRBC) further enlarges steerability within the same feasible state space.

feasible cross-task steering trajectories at the capability level of current VLAs, and supports non-trivial behaviors beyond pick-and-place (see Sec. VI).

Stage-aware segmentation We assume a pick-and-place task structure and assign each state s to one of three semantic stages (equivalently, contact modes):

$$\sigma(s) \in \{\text{pre-grasp}, \text{transport}, \text{place}\} \tag{10}$$

where σ is a stage-labeling function. In practice, σ is computed using simple kinematic and contact cues, including end-effector–object distance, gripper opening, and object support height. The stage label $\sigma(s)$ is precomputed for each demonstration state. These labels restrict steering to stage-consistent regions (e.g., pre-grasp $\rightarrow$ pre-grasp across tasks), simplifying trajectory synthesis and improving feasibility.

Shortest-path end-state selection Steering data consists of instruction-contrasting supervision at the same execution state, i.e., paired samples $(s, l_i, a^{(i)})$ and $(s, l_j, a^{(j)})$ that specify how actions should change when the task instruction is switched. To obtain such data, we connect a state s from a source-task trajectory to a compatible state s' from a target-task demonstration, after which execution follows the original target rollout. This produces a feasible steering trajectory that provides task-j supervision at state s.

Given a start state s and target task τ', we pick a stage-matched state s' with the same stage label $\sigma(s') = \sigma(s)$ from target-task demonstrations such that the interpolation from s to s' is shortest.

Steering trajectory generation A steering trajectory consists of a short transition from the source state s to the selected target state s', followed by execution under the target instruction $l_{\tau'}$. We instantiate this transition using simple linear interpolation. On the physical system, the robot is reset to s, tracks the interpolation segment using a low-level controller, and then executes the policy conditioned on $l_{\tau'}$ from s' onward. Successful executions are recorded as steering transitions and added to the training set for subsequent finetuning of π.

B. Mutual-information-based Steering Trajectory Sampling

As the number of tasks and execution horizon grow, exhaustively evaluating steerability and generating steering trajectories at all states becomes computationally infeasible. Moreover, steerability is highly non-uniform across the state space: only a subset of states consistently fail to respond to instruction changes. We therefore focus data collection on *low-SCR* regions, prioritizing states where the policy is least responsive to language.

Since direct rollout-based evaluation at every state is impractical, we introduce *conditional mutual information* (CMI) between language and action as an efficient offline proxy for steerability. At a fixed state s, the CMI is defined as

$$I(A; L \mid S = s) \;=\; H(A \mid S = s) \;-\; H(A \mid S = s, L) \tag{11}$$

which quantifies the reduction in action uncertainty when the instruction is known.

Intuitively, $H(A \mid S = s)$ captures the diversity of actions the policy may produce from the same state across different instructions, while $H(A \mid S = s, L)$ measures how deterministically the policy behaves under a fixed instruction. A large CMI indicates that language strongly modulates the action distribution at state s, reflecting high steerability. Conversely, low CMI implies weak language–action coupling, revealing states where instruction switches are unlikely to induce meaningful behavioral change.

Theoretical Link: CMI as a Proxy for SCR To evaluate steerability without the computational burden of rollouts, we establish that CMI serves as a **necessary condition** for the Steerability Coverage Ratio (SCR). For a task pair (i, j),

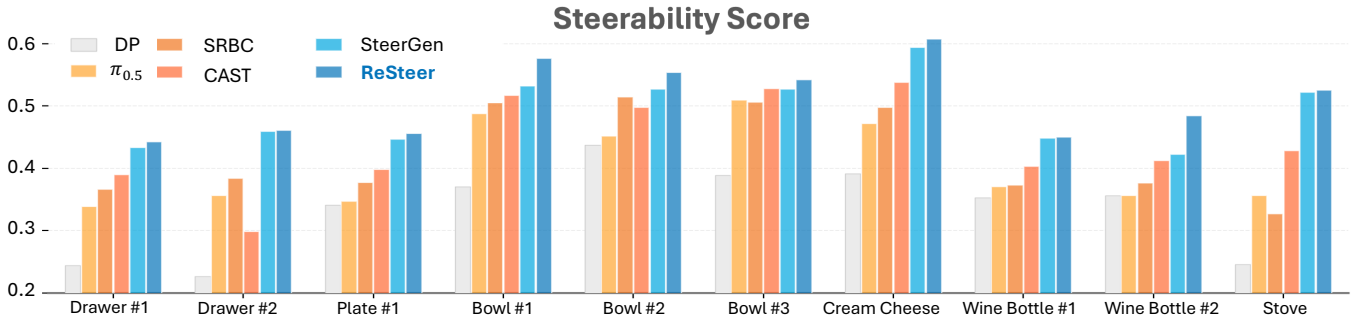

Fig. 5. **Evaluating Steerability on LIBERO-Goal.** For each source task (x-axis), we sample intermediate execution states and measure the average success rate of switching to each of the other nine target tasks under the corresponding language prompt (bars). SteerGen and SRBC are ablations of ReSteer that finetune $\pi_{0.5}$ on the steering-data-generation and self-refinement components alone, respectively. *ReSteer* achieves the highest steerability across all ten tasks and consistently outperforms strong baselines such as CAST [23], demonstrating the effectiveness of our two-stage data generation pipeline.

the CMI at state s is equivalent to the Jensen–Shannon (JS) divergence between instruction-conditioned action distributions:

$$I^{\pi}_{i,j}(s) = \mathrm{JS}\left(\pi(\cdot \mid s, L_i) \,\|\, \pi(\cdot \mid s, L_j)\right) \tag{12}$$

By applying *Pinsker's Inequality*, we can lower-bound this information by the squared Total Variation (TV) distance. Assuming a steerable state must exhibit a minimum behavioral shift ε such that $\mathrm{TV}(\pi_i, \pi_j) \geq \varepsilon$, we obtain:

$$s \in \mathcal{S}^{\mathrm{steer}}_{i \leftrightarrow j}(\pi) \implies I^{\pi}_{i,j}(s) \geq \tau, \quad \text{where } \tau = \tfrac{1}{2}\varepsilon^2 \tag{13}$$

This indicates that states with $I^{\pi}_{i,j}(s) < \tau$ are effectively "instruction-blind" and cannot support task switching. Let ν denote a reference distribution over $\mathcal{S}^{\mathrm{union}}_{i,j}(\pi)$. By taking the expectation over $s \sim \nu$, we establish that the SCR is upper-bounded by the probability of encountering high-CMI states:

$$\mathrm{SCR}_{i \leftrightarrow j}(\pi) \leq \Pr_{s \sim \nu}\left[I^{\pi}_{i,j}(s) \geq \tau\right] \tag{14}$$

This relationship justifies CMI as a *rollout-free proxy* for identifying low-steerability regions without environment simulation. For the complete formal proof and further technical details, please refer to the Supplementary Material.

CMI-Guided State Prioritization To operationalize the CMI defined in Eq. 11, we estimate the entropy terms using Gaussian kernel density estimation (KDE) [36]. For a state s_t, the conditional entropy $\widehat{H}(A \mid s_t, l)$ is computed from the empirical policy density $\widehat{p}(a \mid s_t, l)$:

$$\widehat{H}(A_t \mid s_t, l) = -\frac{1}{NK}\sum_{i,j} \log \widehat{p}(a^{(i)}_j \mid s_t, l) \tag{15}$$

where $a^{(i)}_j$ are action samples drawn from the policy. The marginalized entropy $\widehat{H}(A \mid s_t)$ is estimated by aggregating these samples across the set $\mathcal{L}$.

We use the resulting CMI to identify "instruction-blind" regions where the policy fails to differentiate behavior. Each state s in the buffer $\mathcal{B}$ is assigned a weight $w(s) \propto g(\widehat{I}(A; L \mid s))$, where the shaping function g is chosen to prioritize low-steerability (low-CMI) states. We then sample start states for new rollouts proportional to these weights. This mechanism allows us to trigger targeted steering rollouts *online* during collection or curate high-difficulty states *offline* before finetuning.

C. Self-Refining Behavioral Cloning

While stage-aware data generation increases task overlap by injecting new transitions, we observe that the resulting policy often fails to fully utilize these regions, leading to inconsistent task-switching. As illustrated in Fig. 4, while the first stage expands the feasible intersection of tasks, the policy-induced steerable set $\mathcal{S}^{\mathrm{steer}}_{i \leftrightarrow j}(\pi)$ may still only occupy a small portion of this region.

To close this gap and maximize the steerability coverage ratio (SCR), we propose *Self-Refining Behavioral Cloning* (SRBC). Starting from the initial finetuned policy, we collect rollouts involving task switching and retain only the successful trajectories $\mathcal{D}_{\mathrm{succ}}$. We then iteratively finetune the policy on the augmented dataset $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_{\mathrm{succ}}$ by minimizing:

$$\mathcal{L}_{\mathrm{BC}}(\pi) = \mathbb{E}_{(o,a)\sim\mathcal{D}}[-\log \pi(a \mid o)] \tag{16}$$

By reinforcing only successful steering behaviors, SRBC "sharpens" the policy's response to instruction changes, ensuring that the theoretical steerability provided by the generated data is reliably realized in practice.

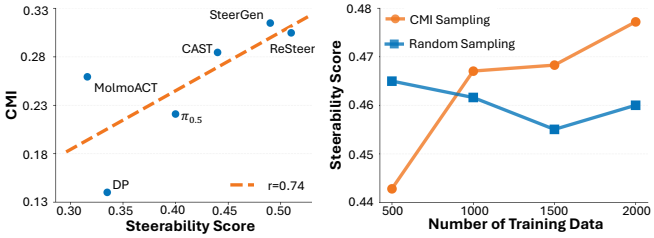

Fig. 6. To evaluate CMI as a proxy for steerability, we track CMI and steerability scores across training checkpoints. As shown in left figure, CMI is positively correlated with steerability, with particularly consistent trends within a single model family (the $\pi_{0.5}$ variants). Motivated by this relationship, we use CMI to prioritize low-steerability states for data collection and fine-tuning. Empirically, MI-guided sampling yields higher sample efficiency than uniform random sampling.

V. Simulation Experiments

We test the following hypotheses to illustrate the effectiveness of the proposed metrics and evaluation on steerability, as well as how our methods improve the steerability.

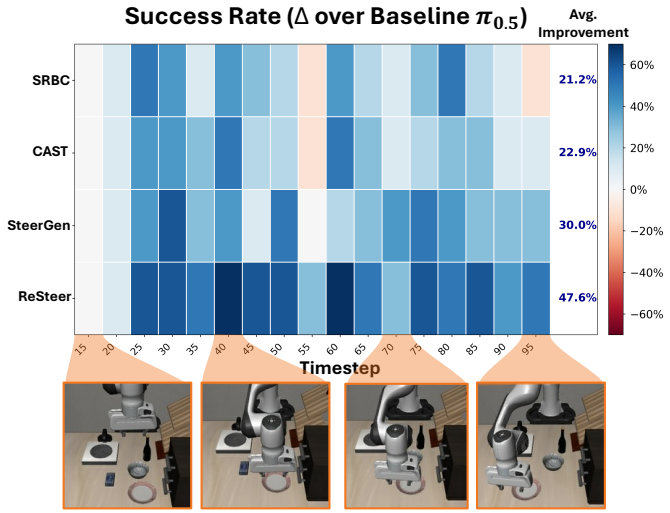

Fig. 7. We conduct a case study on task steering from *Push the Plate to the Front of the Stove* $\rightarrow$ *Put the Bowl on the Plate*. The prompt is switched at the indicated timestep, with representative LIBERO environment states shown at select timesteps. SteerGen and ReSteer consistently outperform all baseline methods, with particularly strong gains at later timestep switches. ReSteer further improves upon SteerGen by sharpening the policy in regions where SteerGen cannot consistently complete the task.

- **H1**: Stage-aware data augmentation increases overlap between task state distributions, leading to improved in-domain steerability across tasks.
- **H2**: CMI serves as an effective proxy for steerability, reducing the need for exhaustive rollout-based evaluation.
- **H3**: CMI-guided state sampling improves the sample efficiency of steering data generation.
- **H4**: SRBC improves the success rate of steerable behaviors, bridging the gap between induced steerability from dataset overlap and realized steerability at execution time.

A. Setup

Dataset We perform experiments with the LIBERO-Goal setup [11]. LIBERO-Goal provides a single-scene environment containing versatile multi-task language labels, offering a controlled setting that is well-suited for evaluating steerability across tasks. We use $\pi_{0.5}$ as a base model and finetune it based on different settings.

Baselines We compare ReSteer against two representative multitask visuomotor learning approaches: (i) **Diffusion Policy (DP)** [37], which we adapt for multitask control using FiLM-based instruction conditioning; and (ii) **CAST** [23], a data-augmentation baseline that improves instruction following by pairing existing observations with counterfactual language labels. To ensure a fair comparison, we implement a CAST-style variant using our stage-aware generator, extracting initial action snippets from generated steering motions to augment the training set with instruction-contrasting supervision.

B. Results

H1: Stage-aware data augmentation improves steerability between tasks. Figure 5 shows that fine-tuning on our augmented data (SteerGen only) improves steerability by 8.8%, indicating that the resulting policy responds more reliably to language instructions across intermediate execution states than the baseline. We also observe that the CAST variant increases steerability; however, because it does not provide complete trajectories to realize full steering behaviors, it still underperforms ReSteer.

H2: Conditional Mutual Information is a good proxy to reflect steerability. We evaluate the validity of CMI as a rollout-free proxy by examining its correlation with the ground-truth steerability score. As shown in Fig. 6, we observe a strong positive correlation ($r = 0.74$) across diverse model architectures and data augmentation strategies. This trend is even more pronounced within specific model families; for instance, the $\pi_{0.5}$ variants exhibit a Pearson correlation exceeding 0.9. These results empirically validate that CMI effectively captures a policy's sensitivity to instruction changes, which matches our theory. This high correlation justifies using CMI as a principled signal to identify "instruction-blind" states and prioritize them for targeted data collection, significantly improving sample efficiency over uniform sampling.

H3: MI-based sampling scheme improves the sample efficiency for data generation. Instead of augmenting states uniformly along each rollout, we concentrate data generation on low CMI states. We compare this CMI-guided strategy against random sampling, which selects task pairs $(a \rightarrow b)$ uniformly at random and uses the corresponding steering trajectories for training. We first collect 50 short steering motion segments for each task pair by uniformly sampling from all task trajectories. Using a single rollout per task pair, we then estimate CMI for candidate states and construct two training sets of equal size: (i) a low-CMI subset obtained by down-sampling the segments with the smallest CMI, and (ii) a randomly down-sampled subset. Figure 6 (right) shows that CMI-guided sampling yields more consistent improvements in steerability as the dataset scales, indicating higher sample efficiency. In contrast, random sampling exhibits substantially higher variance.

H4: SRBC sharpens the success rate of steerable tasks. We hypothesize that reinforcing the policy on its own *successful* steering executions will increase robustness and thus improve overall steerability. We evaluate *Self-Refining Behavioral Cloning* (SRBC), which iteratively collects successful task-switching rollouts and finetunes the policy on these trajectories. In Figure 5, we report steerability for (i) the base $\pi_{0.5}$ policy and (ii) our full pipeline, each with and without SRBC, keeping all other components fixed. SRBC consistently improves steerability in both settings: it yields a robust gain for $\pi_{0.5}$ and a further gain when added to our method, achieving the highest overall success rate. The improvement is consistent across tasks, indicating that SRBC provides a general, reliable boost to steerability rather than a task-specific effect.

Data efficiency and scalability. Because the number of task pairs grows combinatorially with the number of tasks, a natural concern is whether ReSteer requires prohibitive amounts of data. We find this is not the case. On LIBERO-Goal, the ReSteer steering-motion data is only **64%** the size of the original

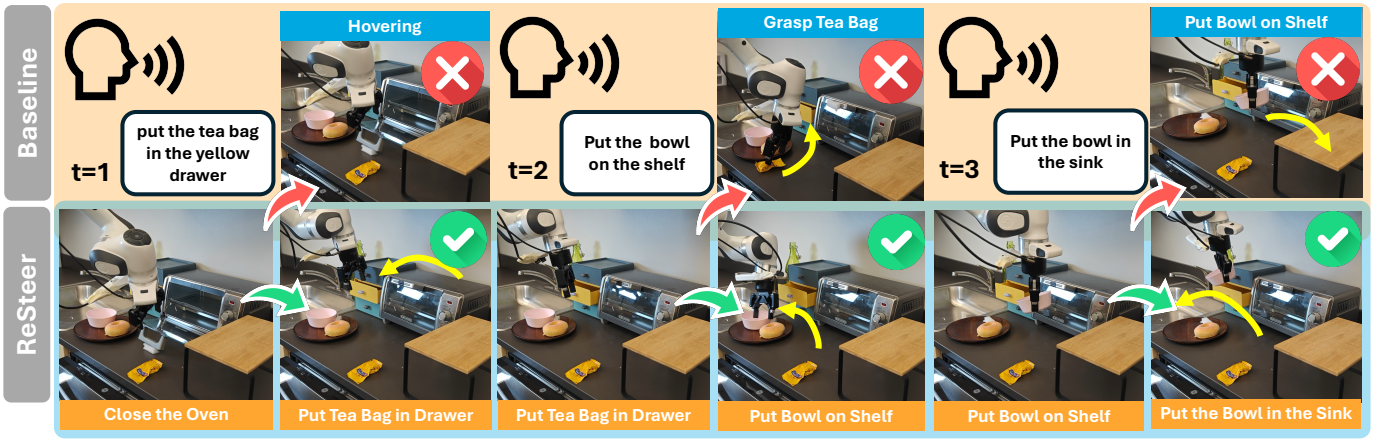

Fig. 8. In an example with four single tasks and three steering scenarios, ReSteer (bottom row) successfully switches to the requested next task upon user instruction, whereas the baseline (top row) exhibits failure modes such as getting stuck or continuing the original task (keep grasping the tea bag, put the bowl to the shelf instead of sink) despite the new prompt.

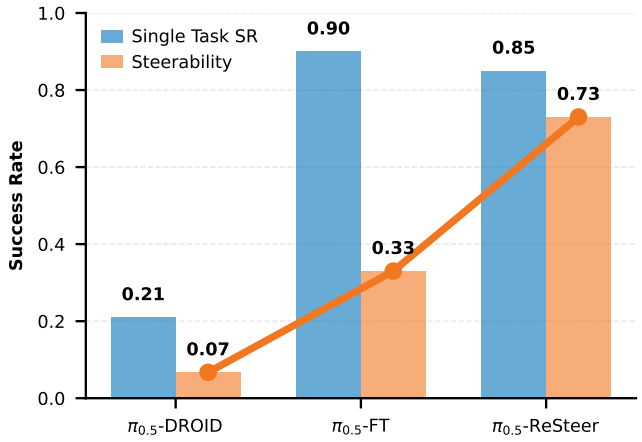

Fig. 9. We evaluate policies on the DROID platform across four single-task evaluations and three steering scenarios. In-domain teleoperation data improves single-task success over the baseline, but yields limited steerability. In contrast, ReSteer achieves $\mathbf{2.2}\times$ higher steerability while preserving single-task performance.

multitask dataset, yet substantially improves steerability across a broader set of states via CMI-based state prioritization. To further probe scalability, we trained a variant using a ReSteer subset containing only five task pairs (20 steering cases out of 90 for 10 tasks). Even with this limited data, the resulting policy improves steering success by 3.9% on seen pairs and by 3.7% on *unseen* pairs over the baseline, suggesting that the benefits of ReSteer generalize beyond the specific task pairs used during training and that steerability can be improved with substantially less data than the worst-case combinatorial estimate. We additionally validate that these gains transfer to a different LIBERO-Object scene; full results are reported in Appendix H.

VI. Real-World Experiments

We conduct real-world experiments using the DROID [38] platform and the $\pi_{0.5}$ DROID checkpoint. The scene is a kitchen-style setup designed to reflect everyday use. It contains an oven, a countertop shelf, a plate, and a bowl, enabling multiple manipulation tasks and within-episode task switching.

A. Task Design and Data Collection

Teleoperated Demonstrations. We construct four different manipulation tasks based on the kitchen setup:

- **Place bowl on shelf:** grasp the bowl, put it on the shelf.
- **Place bowl in sink:** grasp the bowl, put it in the sink.
- **Store tea bag in drawer:** open the yellow drawer, grasp the tea bag, and close the drawer.
- **Close oven:** grasp the oven door and close it.

We curate the finetuning datasets by teleoperating the robot and collecting a dataset of expert trajectories.

***ReSteer* Data Generation.** Specifically, we sample states from the same motion stage across different demonstrations and generate steering motions between those states. Due to the cost of large-scale real-world rollouts, SRBC is treated as an optional refinement on hardware and reported on the two lowest-performing steering cases (Transport $t3 \rightarrow t4$, Place $t1 \rightarrow t2$).

To illustrate how *ReSteer* enables more interactive manipulation with multitask policies, we design a long-horizon task sequences composed of the tasks described above, with mid-execution task steering:

1) Close oven. After closing, steer to (from Place stage):
2) Put the tea bag in the yellow drawer. After opening the drawer, steer to (from Pregrasp stage):
3) Put the white bowl in the sink. After picking up the bowl, steer to (from Transport stage):
4) Put the white bowl to the countertop shelf.

B. Results

We evaluate *ReSteer* on the DROID platform across four manipulation tasks to determine if it can resolve the steerability failures observed in state-of-the-art policies. As illustrated in Fig. 9, while standard teleoperation finetuning ($\pi_{0.5}$-FT) saturates single-task success at 90%, it yields only marginal

Test Case	$\pi_{0.5}$	$\pi_{0.5}$-FT	$\pi_{0.5}$-*ReSteer*
Single-task success (%)			
t1-close the oven	0	**100**	80
t2-put tea bag in drawer	4	60	**70**
t3-put white bowl on the shelf	20	**100**	**100**
t4-put white bowl in the sink	60	**100**	**100**
Steering success (%)			
Pregrasp t1→t3	**100**	0	80
Pregrasp t3→t1	0	40	**100**
Pregrasp t2→t3	0	20	**100**
Transport t3→t4	20	80	**100**
Place t1→t2	0	60	**85**

TABLE II
SUCCESS RATES FOR SINGLE-TASK EXECUTION AND PREGRASPING-STAGE STEERING ACROSS THREE METHODS.

improvements in steerability (33%) compared to the 7% of the zero-shot baseline. This indicates a "specialization pathology" where the model over-indexes on state information to complete a task, effectively ignoring mid-execution instruction changes. In contrast, *ReSteer* achieves a 73% steering success rate—a $2.2\times$ improvement over the finetuned baseline. Average single-task success decreases slightly from 90% to 85%, a modest trade-off attributable to the shift in training distribution introduced by the synthesized steering motions, which present a more challenging supervision target than the standard per-task demonstrations. Table II reveals that the most significant improvements occur during *pregrasp-stage transitions*, such as $t3 \rightarrow t1$ (white bowl on shelf $\rightarrow$ close oven) and $t2 \rightarrow t3$ (tea bag in drawer $\rightarrow$ white bowl on shelf), where *ReSteer* achieves 100% success compared to the baseline's 40% and 20%. These peak gains suggest that *ReSteer* successfully addresses high-conflict decision points where the robot must break its commitment to an initial object to re-orient toward a new goal.

VII. DISCUSSION & CONCLUSION

Limitations. Even though we provide a detailed and thorough analysis on steerability of VLA in a same test scene setting and illustrate how data distribution affects the steerability of the model. Such analysis is costly to run in real-world and test under different scenarios.

Future Works. A promising direction is *online interactive learning*: during deployment, the robot can accept natural-language corrections and use MI-style signals to decide when to query the user, log corrective segments, and prioritize updates on low-steerability states. To keep this practical and safe, we will study lightweight on-robot adaptation. A complementary direction concerns *system safety* under instruction switching: not every prompt change at every state corresponds to a physically feasible or safe transition (e.g., releasing a grasped object mid-transport over a fragile surface). Future work should incorporate feasibility and safety checks—through model-based predictions, constraint-aware planners, or learned reachability estimators—so that the policy can either execute, defer, or query the user when a requested switch is unsafe.

ACKNOWLEDGMENTS

The authors thank the members of the Georgia Tech Robot Learning and Reasoning Lab for helpful discussions and feedback throughout this project.

References

[1] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael 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, 2025. URL https://arxiv.org/abs/2504.16054.

[2] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model, 2024. URL https://arxiv.org/abs/2406.09246.

[3] TRI LBM Team, Jose Barreiros, Andrew Beaulieu, Aditya Bhat, Rick Cory, Eric Cousineau, Hongkai Dai, Ching-Hsin Fang, Kunimatsu Hashimoto, Muhammad Zubair Irshad, Masha Itkina, Naveen Kuppuswamy, Kuan-Hui Lee, Katherine Liu, Dale McConachie, Ian McMahon, Haruki Nishimura, Calder Phillips-Grafflin, Charles Richter, Paarth Shah, Krishnan Srinivasan, Blake Wulfe, Chen Xu, Mengchao Zhang, Alex Alspach, Maya Angeles, Kushal Arora, Vitor Campagnolo Guizilini, Alejandro Castro, Dian Chen, Ting-Sheng Chu, Sam Creasey, Sean Curtis, Richard Denitto, Emma Dixon, Eric Dusel, Matthew Ferreira, Aimee Goncalves, Grant Gould, Damrong Guoy, Swati Gupta, Xuchen Han, Kyle Hatch, Brendan Hathaway, Allison Henry, Hillel Hochsztein, Phoebe Horgan, Shun Iwase, Donovon Jackson, Siddharth Karamcheti, Sedrick Keh, Joseph Masterjohn, Jean Mercat, Patrick Miller, Paul Mitiguy, Tony Nguyen, Jeremy Nimmer, Yuki Noguchi, Reko Ong, Aykut Onol, Owen Pfannenstiehl, Richard Poyner, Leticia Priebe Mendes Rocha, Gordon Richardson, Christopher Rodriguez, Derick Seale, Michael Sherman, Mariah Smith-Jones, David Tago, Pavel Tokmakov, Matthew Tran, Basile Van Hoorick, Igor Vasiljevic, Sergey Zakharov, Mark Zolotas, Rares Ambrus, Kerri Fetzer-Borelli, Benjamin Burchfiel, Hadas Kress-Gazit, Siyuan Feng, Stacie Ford, and Russ Tedrake. A careful examination of large behavior models for multitask dexterous manipulation, 2025. URL https://arxiv.org/abs/2507.05331.

[4] Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, et al. Molmoact: Action reasoning models that can reason in space. *arXiv preprint arXiv:2508.07917*, 2025.

[5] Anthony Brohan, Noah Brown, Justice Carbajal, et al. Rt-1: Robotics transformer for real-world control at scale. In *Conference on Robot Learning*, 2023.

[6] NVIDIA, :, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025. URL https://arxiv.org/abs/2503.14734.

[7] Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. Smolvla: A vision-language-action model for affordable and efficient robotics, 2025. URL https://arxiv.org/abs/2506.01844.

[8] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

[9] Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. *arXiv preprint arXiv:2407.08693*, 2024.

[10] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, November 2020. ISSN 2522-5839. doi: 10.1038/s42256-020-00257-z. URL http://dx.doi.org/10.1038/s42256-020-00257-z.

[11] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning, 2023. URL https://arxiv.org/abs/2306.03310.

[12] Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew Walter, Ashis Banerjee, Seth Teller, and Nicholas Roy. Understanding natural language commands for robotic navigation and mobile manipulation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 25 (1):1507–1514, Aug. 2011. doi: 10.1609/aaai.v25i1.7979. URL https://ojs.aaai.org/index.php/AAAI/article/view/7979.

[13] Simon Stepputtis, Joseph Campbell, Mariano Phielipp, Stefan Lee, Chitta Baral, and Heni Ben Amor. Language-conditioned imitation learning for robot manipulation tasks, 10 2020.

[14] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea

Finn. Bc-z: Zero-shot task generalization with robotic imitation learning, 2022. URL https://arxiv.org/abs/2202.02005.

[15] Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, Albert Tung, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anchit Gupta, Andrew Wang, Andrey Kolobov, Anikait Singh, Animesh Garg, Aniruddha Kembhavi, Annie Xie, Anthony Brohan, Antonin Raffin, Archit Sharma, Arefeh Yavary, Arhan Jain, Ashwin Balakrishna, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Blake Wulfe, Brian Ichter, Cewu Lu, Charles Xu, Charlotte Le, Chelsea Finn, Chen Wang, Chenfeng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Christopher Agia, Chuer Pan, Chuyuan Fu, Coline Devin, Danfei Xu, Daniel Morton, Danny Driess, Daphne Chen, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dinesh Jayaraman, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Ethan Foster, Fangchen Liu, Federico Ceola, Fei Xia, Feiyu Zhao, Felipe Vieira Frujeri, Freek Stulp, Gaoyue Zhou, Gaurav S. Sukhatme, Gautam Salhotra, Ge Yan, Gilbert Feng, Giulio Schiavi, Glen Berseth, Gregory Kahn, Guangwen Yang, Guanzhi Wang, Hao Su, Hao-Shu Fang, Haochen Shi, Henghui Bao, Heni Ben Amor, Henrik I Christensen, Hiroki Furuta, Homanga Bharadhwaj, Homer Walke, Hongjie Fang, Huy Ha, Igor Mordatch, Ilija Radosavovic, Isabel Leal, Jacky Liang, Jad Abou-Chakra, Jaehyung Kim, Jaimyn Drake, Jan Peters, Jan Schneider, Jasmine Hsu, Jay Vakil, Jeannette Bohg, Jeffrey Bingham, Jeffrey Wu, Jensen Gao, Jiaheng Hu, Jiajun Wu, Jialin Wu, Jiankai Sun, Jianlan Luo, Jiayuan Gu, Jie Tan, Jihoon Oh, Jimmy Wu, Jingpei Lu, Jingyun Yang, Jitendra Malik, João Silvério, Joey Hejna, Jonathan Booher, Jonathan Tompson, Jonathan Yang, Jordi Salvador, Joseph J. Lim, Junhyek Han, Kaiyuan Wang, Kanishka Rao, Karl Pertsch, Karol Hausman, Keegan Go, Keerthana Gopalakrishnan, Ken Goldberg, Kendra Byrne, Kenneth Oslund, Kento Kawaharazuka, Kevin Black, Kevin Lin, Kevin Zhang, Kiana Ehsani, Kiran Lekkala, Kirsty Ellis, Krishan Rana, Krishnan Srinivasan, Kuan Fang, Kunal Pratap Singh, Kuo-Hao Zeng, Kyle Hatch, Kyle Hsu, Laurent Itti, Lawrence Yunliang Chen, Lerrel Pinto, Li Fei-Fei, Liam Tan, Linxi "Jim" Fan, Lionel Ott, Lisa Lee, Luca Weihs, Magnum Chen, Marion Lepert, Marius Memmel, Masayoshi Tomizuka, Masha Itkina, Mateo Guaman Castro, Max Spero, Maximilian Du, Michael Ahn, Michael C. Yip, Mingtong Zhang, Mingyu Ding, Minho Heo, Mohan Kumar Srirama, Mohit Sharma, Moo Jin Kim, Muhammad Zubair Irshad, Naoaki Kanazawa, Nicklas Hansen, Nicolas Heess, Nikhil J Joshi, Niko Suenderhauf, Ning Liu, Norman Di Palo, Nur Muhammad Mahi Shafiullah, Oier Mees, Oliver Kroemer, Osbert Bastani, Pannag R Sanketi, Patrick "Tree" Miller, Patrick Yin, Paul Wohlhart, Peng Xu, Peter David Fagan, Peter Mitrano, Pierre Sermanet, Pieter Abbeel, Priya Sundaresan, Qiuyu Chen, Quan Vuong, Rafael Rafailov, Ran Tian, Ria Doshi, Roberto Martín-Martín, Rohan Baijal, Rosario Scalise, Rose Hendrix, Roy Lin, Runjia Qian, Ruohan Zhang, Russell Mendonca, Rutav Shah, Ryan Hoque, Ryan Julian, Samuel Bustamante, Sean Kirmani, Sergey Levine, Shan Lin, Sherry Moore, Shikhar Bahl, Shivin Dass, Shubham Sonawani, Shubham Tulsiani, Shuran Song, Sichun Xu, Siddhant Haldar, Siddharth Karamcheti, Simeon Adebola, Simon Guist, Soroush Nasiriany, Stefan Schaal, Stefan Welker, Stephen Tian, Subramanian Ramamoorthy, Sudeep Dasari, Suneel Belkhale, Sungjae Park, Suraj Nair, Suvir Mirchandani, Takayuki Osa, Tanmay Gupta, Tatsuya Harada, Tatsuya Matsushima, Ted Xiao, Thomas Kollar, Tianhe Yu, Tianli Ding, Todor Davchev, Tony Z. Zhao, Travis Armstrong, Trevor Darrell, Trinity Chung, Vidhi Jain, Vikash Kumar, Vincent Vanhoucke, Vitor Guizilini, Wei Zhan, Wenxuan Zhou, Wolfram Burgard, Xi Chen, Xiangyu Chen, Xiaolong Wang, Xinghao Zhu, Xinyang Geng, Xiyuan Liu, Xu Liangwei, Xuanlin Li, Yansong Pang, Yao Lu, Yecheng Jason Ma, Yejin Kim, Yevgen Chebotar, Yifan Zhou, Yifeng Zhu, Yilin Wu, Ying Xu, Yixuan Wang, Yonatan Bisk, Yongqiang Dou, Yoonyoung Cho, Youngwoon Lee, Yuchen Cui, Yue Cao, Yueh-Hua Wu, Yujin Tang, Yuke Zhu, Yunchu Zhang, Yunfan Jiang, Yunshuang Li, Yunzhu Li, Yusuke Iwasawa, Yutaka Matsuo, Zehan Ma, Zhuo Xu, Zichen Jeff Cui, Zichen Zhang, Zipeng Fu, and Zipeng Lin. Open x-embodiment: Robotic learning datasets and rt-x models, 2025. URL https://arxiv.org/abs/2310.08864.

[16] Cunxin Fan, Xiaosong Jia, Yihang Sun, Yixiao Wang, Jianglan Wei, Ziyang Gong, Xiangyu Zhao, Masayoshi Tomizuka, Xue Yang, Junchi Yan, and Mingyu Ding. Interleave-vla: Enhancing robot manipulation with interleaved image-text instructions. *arXiv preprint arXiv:2505.02152*, 2025.

[17] Zhongyi Zhou, Yichen Zhu, Minjie Zhu, Junjie Wen, Ning Liu, Zhiyuan Xu, Weibin Meng, Ran Cheng, Yaxin Peng, Chaomin Shen, and Feifei Feng. Chatvla: Unified multimodal understanding and robot control with vision-language-action model, 2025. URL https://arxiv.org/abs/2502.14420.

[18] Zhongyi Zhou, Yichen Zhu, Junjie Wen, Chaomin Shen, and Yi Xu. Chatvla-2: Vision-language-action model with open-world embodied reasoning from pretrained knowledge, 2025. URL https://arxiv.org/abs/2505.21906.

[19] Lucy Xiaoyang Shi, Brian Ichter, Michael Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolo Fusai, Adrian Li-Bell, Danny Driess, Lachy Groom, Sergey Levine, and Chelsea Finn. Hi robot: Open-ended instruction following with hierarchical vision-language-action models, 2025. URL https://arxiv.org/abs/2502.19417.

[20] Lucy Xiaoyang Shi, Zheyuan Hu, Tony Z. Zhao, Archit Sharma, Karl Pertsch, Jianlan Luo, Sergey Levine, and

Chelsea Finn. Yell at your robot: Improving on-the-fly from language corrections, 2024. URL https://arxiv.org/abs/2403.12910.

[21] Huang Fang, Mengxi Zhang, Heng Dong, Wei Li, Zixuan Wang, Qifeng Zhang, Xueyun Tian, Yucheng Hu, and Hang Li. Robix: A unified model for robot interaction, reasoning and planning, 2025. URL https://arxiv.org/abs/2509.01106.

[22] Meng Li, Zhen Zhao, Zhengping Che, Fei Liao, Kun Wu, Zhiyuan Xu, Pei Ren, Zhao Jin, Ning Liu, and Jian Tang. Switchvla: Execution-aware task switching for vision-language-action models, 2025. URL https://arxiv.org/abs/2506.03574.

[23] Catherine Glossop, William Chen, Arjun Bhorkar, Dhruv Shah, and Sergey Levine. Cast: Counterfactual labels improve instruction following in vision-language-action models, 2025. URL https://arxiv.org/abs/2508.13446.

[24] Yanwei Wang, Lirui Wang, Yilun Du, Balakumar Sundaralingam, Xuning Yang, Yu-Wei Chao, Claudia Perez-D'Arpino, Dieter Fox, and Julie Shah. Inference-time policy steering through human interactions. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025. URL https://arxiv.org/abs/2411.16627.

[25] Stéphane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[26] Siddharth Karamcheti, Suraj Nair, Annie S. Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-driven representation learning for robotics, 2023. URL https://arxiv.org/abs/2302.12766.

[27] Quanyi Li. Task reconstruction and extrapolation for π_0 using text latent. *arXiv preprint arXiv:2505.03500*, 2025.

[28] Seungjae Lee, Yibin Wang, Haritheja Etukuru, H. Jin Kim, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Blade: Decomposed behavior cloning for language-conditioned robotic manipulation. *arXiv preprint arXiv:2403.03181*, 2024.

[29] Simon Stepputtis, Joseph Campbell, Mariano Phielipp, Stefan Lee, Chitta Baral, and Heni Ben Amor. Language-conditioned imitation learning for robot manipulation tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 13139–13150. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/9909794d52985cbc5d95c26e31125d1a-Paper.pdf.

[30] Liquan Wang, Ankit Goyal, Haoping Xu, and Animesh Garg. Discovering robotic interaction modes with discrete representation learning, 2024. URL https://arxiv.org/abs/2410.20258.

[31] Zhekai Duan, Yuan Zhang, Shikai Geng, Gaowen Liu, Joschka Boedecker, and Chris Xiaoxuan Lu. Fast ecot: Efficient embodied chain-of-thought via thoughts reuse, 2025. URL https://arxiv.org/abs/2506.07639.

[32] Anhong Gu, Ahmed Kirmani, et al. Rt-trajectory: Robotic task generalization via hindsight trajectory sketches. In *Proceedings of the Conference on Robot Learning*, 2023.

[33] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success, 2025. URL https://arxiv.org/abs/2502.19645.

[34] Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations, 2023. URL https://arxiv.org/abs/2310.17596.

[35] Zhengrong Xue, Shuying Deng, Zhenyang Chen, Yixuan Wang, Zhecheng Yuan, and Huazhe Xu. Demogen: Synthetic demonstration generation for data-efficient visuomotor policy learning. *arXiv preprint arXiv:2502.16932*, 2025.

[36] Lingxiao Guo, Zhengrong Xue, Zijing Xu, and Huazhe Xu. Demospeedup: Accelerating visuomotor policies via entropy-guided demonstration acceleration, 2025. URL https://arxiv.org/abs/2506.05064.

[37] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[38] 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, Vitor Guizilini, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Muhammad Zubair Irshad, Donovon Jackson, Charlotte Le, Yunshuang Li, Kevin Lin, 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, 2025. URL https://arxiv.org/abs/2403.12945.

SUPPLEMENTARY MATERIAL

The supplementary material is structured as follows:

- **ReSteer Framework and Algorithm** (§A): this section describes the detailed workflow of the *ReSteer* framework and implementation of each component.
- **Steerability Plot and Comparison for Tasks** (§B): this section provides steerability plots for individual tasks and further analysis.
- **Analysis on Conditional Mutual Information Result** (§C)
- **Visualization and Discussion of LIBERO Trajectories**: we visualize the LIBERO tasks' trajectories to discuss the potential loss of steerability when finetuned on a smaller dataset.(§D)
- **Inspecting Language Information Restoration from VLA Latent** (§E)
- **Properties of Steerable States** (§F): we derive the properties of steerable states based on the definitions.
- **Proof: CMI is a Proxy for Steerability** (§G): this section shows a proof sketch and theoretical link between CMI and steerability.
- **Hyperparameters** (§I)

APPENDIX A
SYSTEM OVERVIEW

ReSteer Framework. ReSteer is an iterative data-centric refinement procedure that progressively expands and consolidates the steerable state set of a pretrained multitask policy. The framework alternates between two stages. First, it performs CMI-guided steering data generation (*SteerGen*), where states with weak language–action coupling are identified and augmented with instruction-contrasting transitions to enlarge the feasible steering region. Second, it applies self-refining behavioral cloning (SRBC), which reinforces successful task-switching rollouts collected from the updated policy to improve the reliability of steering behavior. By repeatedly injecting targeted steering supervision and reinforcing realized switching success, ReSteer increases steerability coverage while preserving single-task competence.

We detail the *ReSteer* framework in 1.

Algorithm 1 ReSteer Framework

Require: Pretrained policy π, Dataset $\mathcal{D}$, Number of Action Samples N_a

1: **while** True **do**
  $\mathcal{D}_{\text{SteerGen}=\{\}}$
  Sample states from task i: $s_0, s_1, \ldots, s_t \sim p_k^{\pi}(\tau \mid s_0)$
2: **for** $s_0, s_1, \ldots, s_t$ **do** ▷ Calculate CMI for rollout States
3: Sample N_a action chunks for each l_j: $a_j \sim p(a|s_t, l_j)$
4: Calculate $\widehat{I}(A; L \mid s_t)$ ▷ Eq. 15
5: **end for**
6: **for** $s_i, l_j \sim q(s_t, l)$ **do** ▷ Eq. 20: Sampling from Low CMI States
7: Select end state s_j
8: $\tau_j^i = \text{SteerGen}(s_i, s_j)$
9: $\mathcal{D}_{\text{SteerGen}} \leftarrow \tau_j^i$
10: **end for**
11: $\pi_{\text{SteerGen}} \leftarrow \text{Finetune}(\pi, \mathcal{D} \cup \mathcal{D}_{\text{SteerGen}})$
12: Sample n rollouts: $\tau_{success} \sim \pi_{\text{ReSteer}}$
13: $\mathcal{D}_{\text{SRBC}} \leftarrow \tau_{success}$
14: $\pi_{\text{ReSteer}} \leftarrow \text{Finetune}(\pi, \mathcal{D} \cup \mathcal{D}_{\text{SteerGen}} \cup \mathcal{D}_{\text{SRBC}})$
15: $\pi = \pi_{\text{ReSteer}}$
16: **end while**

CMI-Guided State Prioritization. In Algorithm 1 (Lines 3–7), we estimate the conditional mutual information (CMI) at each rollout state to identify instruction-blind regions where steering supervision is most needed.

To compute the conditional action entropy $H(A \mid S = s_t, L = l)$, we approximate the instruction-conditioned action density using Gaussian kernel density estimation (KDE) [36]. Given N_a sampled action chunks of horizon K under instruction l, the density at timestep t is estimated as

$$\widehat{p}(A_t \mid s_t, l) = \frac{1}{N_a K h} \sum_{j=t}^{t+K-1} \sum_{i=1}^{N_a} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{\|a_j - a_j^{(i)}\|_2^2}{2h^2} \right), \tag{17}$$

where h is the KDE bandwidth.

The conditional entropy at state s_t is then estimated as

$$\widehat{H}(A_t \mid s_t, l) = -\frac{1}{N_a K} \sum_{j=t}^{t+K-1} \sum_{i=1}^{N_a} \widehat{p}\Big(a_j^{(i)} \mid s_t, l\Big) \log \widehat{p}\Big(a_j^{(i)} \mid s_t, l\Big). \tag{18}$$

To estimate the language-marginalized entropy $H(A_t \mid s_t)$, we aggregate samples across prompts $l' \in \mathcal{L}$ using the same KDE procedure. The empirical CMI used in Line 4 of the algorithm is then

$$\widehat{I}(A; L \mid s_t) = \widehat{H}(A_t \mid s_t) - \widehat{H}(A_t \mid s_t, L).$$

In practice, instead of flattening the entire action chunk (which leads to high-dimensional density estimation), we compute entropy using the difference between the final action in the chunk and the current robot state. Empirically, this lower-dimensional representation yields more stable density estimates.

Sampling States Based on CMI. After computing $\widehat{I}(A; L \mid s_t)$ for rollout states (Line 4), we assign each candidate pair (s_t, l) a sampling weight,

$$w(s_t, l) \;\propto\; g\Big(\widehat{I}(A; L \mid s_t)\Big), \tag{19}$$

where $g(\cdot)$ is a shaping function that emphasizes low-steerability states (low CMI).

As shown in Line 6 of Algorithm 1, start states for steering generation are sampled from

$$q(s_t, l) = \frac{w(s_t, l)}{\sum_{s' \in \mathcal{B}} \sum_{l \in \mathcal{L}} w(s', l)}, \tag{20}$$

where $\mathcal{B}$ is a buffer of states collected across tasks.

This CMI-guided prioritization focuses steering data generation on states where the policy exhibits weak language–action coupling, thereby improving sample efficiency and accelerating expansion of the steerable set.

APPENDIX B
STEERABILITY PLOT AND COMPARISON FOR TASKS

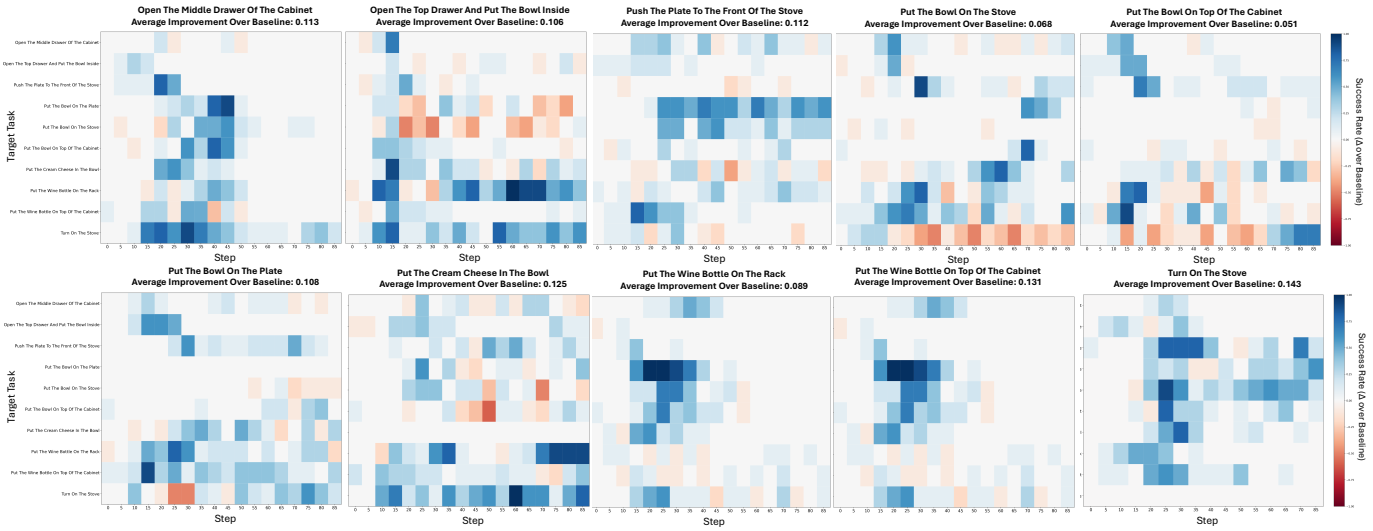

Fig. 10. Task-level steerability improvement on LIBERO-Goal. Each subplot corresponds to a source task (executed initially), and reports the improvement in steering success rate (Δ over the baseline policy) when switching to all other target tasks at different timesteps. The x-axis shows the execution timestep of task switching, the y-axis lists target tasks, and color encodes the success-rate difference, where blue indicates improvement. The number above each subplot indicates the average improvement across all target tasks for that source task ReSteer consistently improves steerability across tasks, with particularly large gains at later timesteps where task commitment is stronger and baseline policies tend to ignore prompt changes.

Figure 10 visualizes steerability improvement across all LIBERO-Goal tasks. For each source task, we execute the original instruction and sample intermediate states along the rollout. At each sampled timestep, we switch the prompt to every other target task and measure the steering success rate. The x-axis denotes the timestep at which the task switch occurs during execution, the y-axis enumerates the target tasks, and the color of each cell indicates the difference in success rate, with blue regions corresponding to improvement in success rate.

a) Temporal Structure of Steerability.: Across tasks, we observe a consistent temporal pattern: steerability improvement is modest at early timesteps and becomes more pronounced at later timesteps. This trend reflects the fact that early execution states often remain compatible with multiple task objectives, whereas later states exhibit stronger task commitment (e.g., object grasping or contact), making mid-execution switching more challenging. ReSteer yields particularly large gains in these high-conflict regions, demonstrating that the generated steering data effectively expands the feasible switching set.

b) Task-Dependent Asymmetry.: The magnitude of improvement varies across source tasks. Tasks involving object placement or contact-rich manipulation (e.g., *Turn On The Stove*, *Put The Wine Bottle On Top Of The Cabinet*) show the largest average gains (up to 0.143). These tasks require decisive behavioral redirection, and baseline policies tend to over-commit to the original goal. ReSteer substantially mitigates this commitment bias.

In contrast, tasks with less constrained object geometry (e.g., *Put The Bowl On Top Of The Cabinet*) show smaller improvements, suggesting that these tasks already exhibit moderate baseline steerability.

c) Implications for Steerability Coverage.: The consistent positive Δ across all source tasks indicates that ReSteer enlarges the steerable state set without sacrificing feasibility. Importantly, gains are not confined to specific task pairs; improvements are broadly distributed across targets, suggesting that the framework enhances general language–action coupling rather than overfitting to particular transitions.

Overall, the visualization confirms that ReSteer systematically improves mid-execution prompt switching, with the largest benefits emerging in states where baseline policies exhibit strong task commitment.

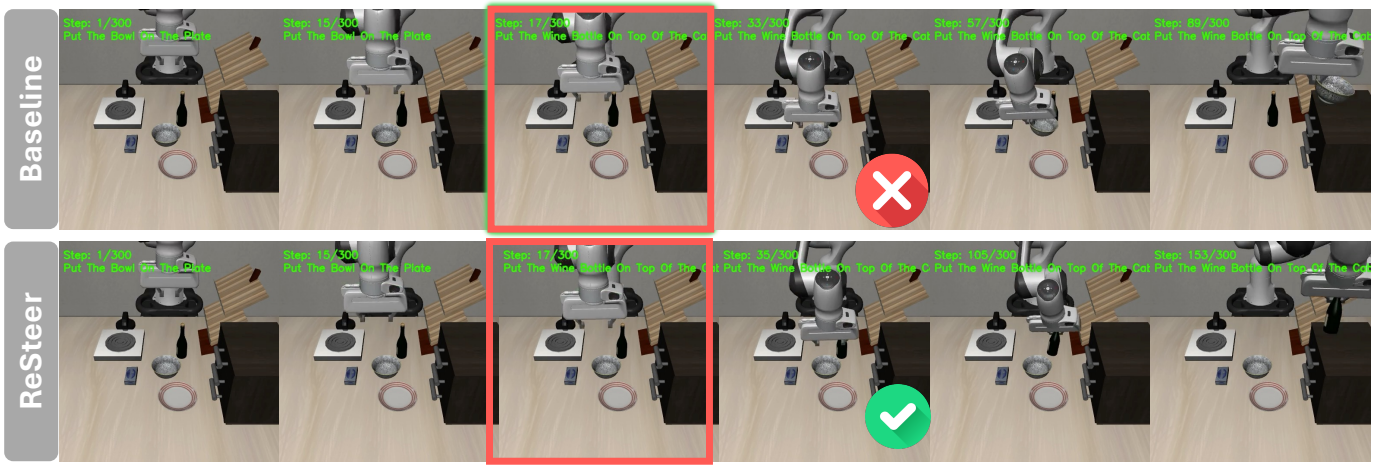

Fig. 11. We prompt the model to switch from the "put the bowl on the stove" to "put the wine bottle on top of the cabinet" at $t = 17$. While baseline follows "put the bowl on the stove" throughout the rollout, *ReSteer* manages to switch to the execution of the "put the wine bottle on top of the cabinet."

APPENDIX C
ANALYSIS ON CONDITIONAL MUTUAL INFORMATION RESULT

Conditional mutual information (CMI) serves as a necessary indicator of language-sensitive behavior: a policy that ignores language will satisfy $I(A; L \mid S_t) \approx 0$. However, our empirical results reveal that large conditional MI alone does not guarantee successful or task-aligned steering.

Specifically, we observe two failure modes in which CMI can be misleading:

a) High influence but low competence.: A model may exhibit large marginal action entropy $H(A \mid S_t)$ and correspondingly large $I(A; L \mid S_t)$ because different prompts shift the action distribution toward distinct but incorrect modes. In this regime, the policy is *sensitive* to language, yet the induced behaviors are not aligned with task success. The mutual information is inflated by unstructured or noisy variability rather than meaningful goal-directed control.

b) Confident collapse to incorrect actions.: Conversely, conditioning on language may sharply reduce $H(A \mid S_t, L)$, producing a deterministic but incorrect action. In this case, $I(A; L \mid S_t)$ can remain nontrivial even though the policy is consistently wrong. Here, language influences behavior, but not in a task-consistent manner.

To better quantify the fraction of action uncertainty resolved by language, we introduce a normalized variant:

$$\widetilde{I}(A; L \mid S_t) \;=\; \frac{I(A; L \mid S_t)}{H(A \mid S_t)}. \tag{21}$$

This normalization discounts states with inherently large action variance and more directly reflects the relative strength of language conditioning.

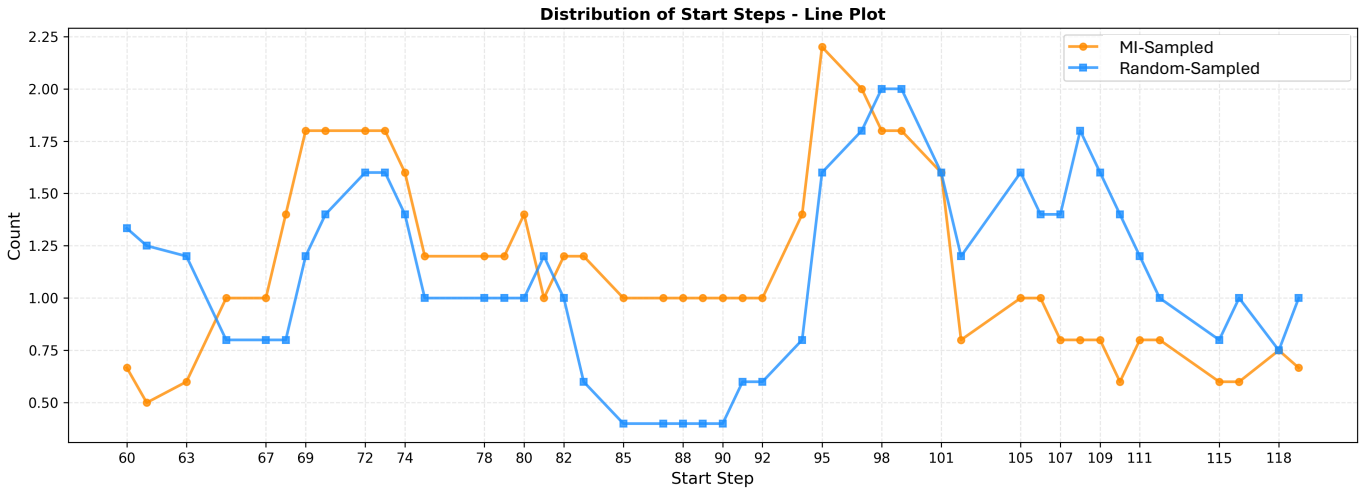

Fig. 12. Comparison of start-step distributions under MI-guided sampling and uniform random sampling. The x-axis denotes the timestep selected as the starting state for steering data generation, and the y-axis shows the sampling count. MI-guided sampling concentrates selections in specific temporal regions corresponding to low-CMI (instruction-blind) states, where language–action coupling is weak. In contrast, random sampling distributes selections more uniformly across the rollout. This targeted concentration demonstrates the improved sample efficiency of CMI-based prioritization, focusing data generation on states that most require steering supervision.

Fig. 13. Visualization of MI-guided sampled states on the DROID platform. Highlighted frames correspond to rollout states selected using the CMI-based prioritization strategy. The sampled states concentrate around high-conflict or decision-critical phases (e.g., object approach, grasp transition, and pre-placement), where baseline policies exhibit weak language–action coupling. This confirms that CMI-guided sampling identifies instruction-blind regions in real-world trajectories and focuses data generation on states that most require steering supervision, thereby improving sample efficiency compared to uniform sampling.

c) Visualization of MI-Based State Selection.: Fig. 14 visualizes the CMI values along a rollout of the task *"put the wine bottle on top of the cabinet."* Each row corresponds to a target prompt, and each column corresponds to a timestep. Cells are color-coded by MI magnitude, and blue boxes highlight the bottom 50% of states selected for steering data generation.

Several structural patterns are evident. First, low-MI states cluster in contiguous temporal segments, particularly during mid-rollout phases where the policy's behavior is dominated by state dynamics (e.g., reaching or transport) rather than instruction-dependent branching. These states are precisely where baseline policies exhibit instruction-blind behavior.

Second, high-MI regions tend to occur near decision points (e.g., object interaction or goal placement), where language meaningfully modulates the action distribution. However, as discussed above, not all high-MI states correspond to successful steering; some reflect unstable or misaligned behavioral divergence.

The CMI-guided sampling mechanism in Algorithm 1 therefore focuses on the instruction-blind regions (blue-highlighted states), injecting instruction-contrasting supervision exactly where the baseline policy fails to differentiate tasks. This targeted expansion of language–action coupling improves steerability coverage while avoiding redundant data generation in already language-sensitive regions.

d) CMI-Guided Sampling for DROID Experiment: Based on the above observation, we compare CMI-guided state prioritization against uniform random sampling on the DROID platform. We use SteerGen to collect 150 steering trajectories for switching from *close the oven* to *put the white bowl in the sink*. We then construct two equal-sized training sets by downsampling to 50 trajectories: (i) a CMI-guided subset that prioritizes low-CMI (instruction-blind) start states, and (ii) a uniformly random

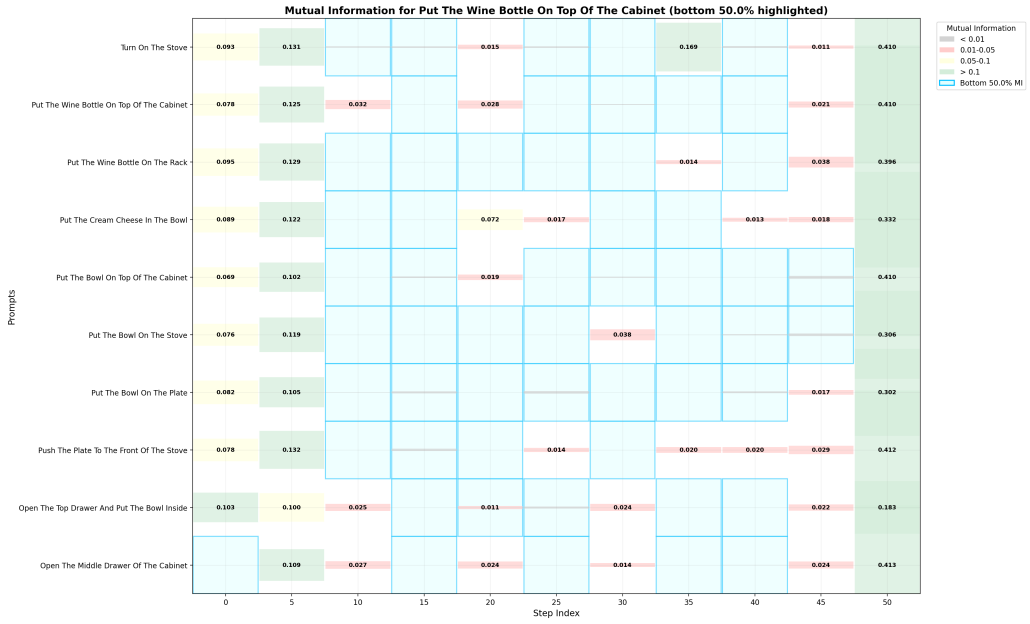

Fig. 14. Visualization of states that were sampled according to the CMI Metric for rollout of task p̈ut the wine bottle on top of the cabinet". The blue boxes indicate the states selected with mutual information.

subset. Fig. 12 reports the resulting start-timestep distributions and we visualize representative start states selected by CMI sampling Fig. 13. Finally, we fine-tune separate policies on the two subsets and evaluate steering from *close the oven* to *put the white bowl in the sink*. CMI-guided sampling achieves higher steering success (80%) than random sampling (75%), suggesting better sample efficiency by focusing supervision on states where language–action coupling is weakest.

	CMI-Sampling	**Random Sampling**
Steerability Score	0.8	0.75

TABLE III
STEERABILITY SCORE FOR DIFFERENT SAMPLING STRATEGY ON DROID

APPENDIX D
VISUALIZATION AND DISCUSSION OF LIBERO TRAJECTORIES

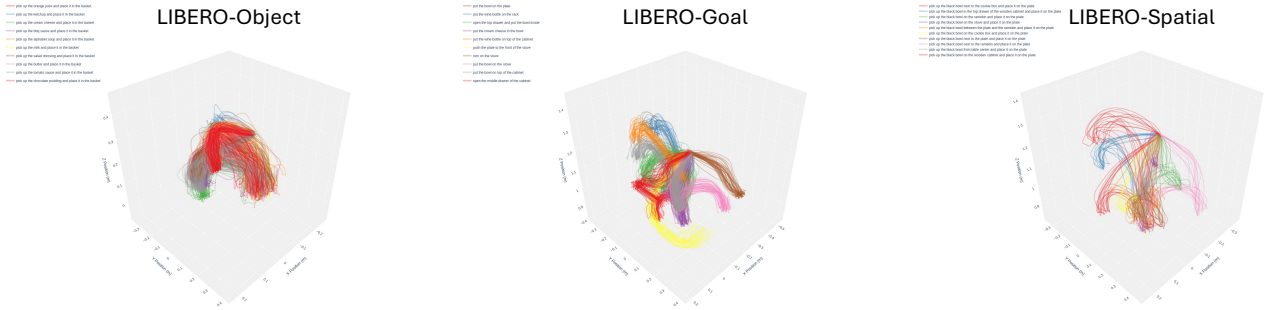

Fig. 15. We visualize the LIBERO benchmark trajectories. Most of the task trajectories from the dataset are clustered into a small space and lead to a relatively small $\mathcal{S}^{\text{steer}}$, which potentially makes the learn policies less steerable.

Visualization of LIBERO Trajectories. Figure Fig. 15 visualizes the end-effector trajectories for LIBERO tasks. We observe that tasks exhibiting substantial spatial overlap in end-effector trajectories tend to be more amenable to steering. Intuitively, high overlap suggests shared underlying motion primitives, which facilitates transferring control signals across tasks.

Discussion. Tasks that are already closely related in the original dataset (e.g., put bowl on plate and put bowl on stove) naturally exhibit high steerability. As a result, the relative gains from our method on these tasks are smaller, since much of the transferable structure is already present. In contrast, tasks with less initial overlap benefit more from steering, leading to larger relative improvements.

APPENDIX E
INSPECTING TASK INFORMATION RESTORATION FROM VLA LATENT

We study whether semantic information associated with task instructions is already encoded in the visual representation learned by the policy. Concretely, we test whether different language prompts are *linearly separable* from visual latents alone, without access to language embeddings. A positive result would indicate that the policy can infer task identity directly from visual context, providing evidence for a shortcut in which language conditioning is partially ignored.

Hypothesis. Visual observations (and their latent representations) may contain sufficient semantic cues to identify the task being executed. As a result, the policy can recover task intent from vision alone, reducing reliance on explicit language input.

A. Experiment 1: Multitask Linear Separability from Visual Latents

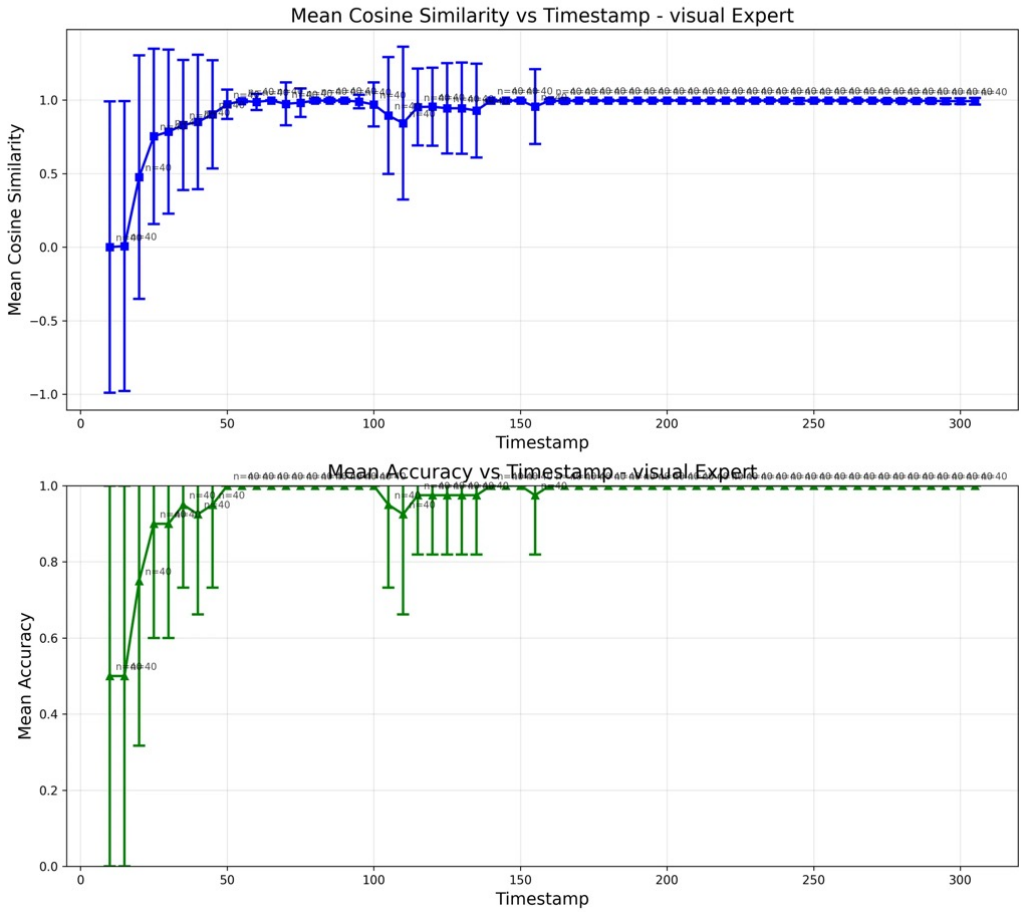

Fig. 16. **Stage-dependent linear separability of task semantics from visual latents.** We report the performance of a linear probe trained to predict task identity from visual latents as a function of rollout timestep. **Top:** mean cosine similarity between predicted and ground-truth task prototypes. **Bottom:** mean classification accuracy. Early in the rollout (pre-grasp stage), both metrics are low and exhibit high variance, indicating that visual latents are not linearly separable across tasks when the scene is visually similar. As execution progresses and physical interaction occurs (post-grasp and transport stages), both cosine similarity and accuracy rapidly increase and saturate near 1.0, with reduced variance. This demonstrates that task semantics become strongly encoded in visual latents once task-specific effects are reflected in the environment state, enabling task inference from vision alone.

Setup. We consider a multitask setting with $K = 10$ tasks in LIBERO-Object. For each task, we collect visual latents produced by the policy encoder during closed-loop rollouts. Each latent corresponds to a visual observation o_t and is extracted *without* conditioning on language embeddings.

To stress-test linear separability, we aggregate latents across multiple sources of variation: (i) different task identities, (ii) different rollout timesteps, and (iii) multiple camera viewpoints (agent-view and wrist camera). We explicitly exclude the right camera, which is not consistently available across all rollouts.

We train a linear classifier (logistic regression) to predict the task identity from the visual latent. This is the hardest probing setting, as the classifier must remain invariant to time, viewpoint, and trajectory-specific variations.

Metrics. We report (i) classification accuracy on a held-out evaluation set and (ii) cosine similarity between predicted and ground-truth class prototypes, which reflects the geometric separability of task clusters in latent space.

Results. Training on agent-view and wrist-camera latents ($\sim$500 training samples, $\sim$100 evaluation samples), the linear probe achieves an accuracy of 95.8% and an average cosine similarity of 0.91. This indicates that task identity is highly linearly decodable from visual latents alone, even under substantial nuisance variation.

Interpretation. These results suggest that, across different scenes and execution stages, the visual latent implicitly encodes task semantics. Consequently, the policy can often infer "what to do" from vision alone, which helps explain why language conditioning becomes weak or redundant in multitask execution.

B. Experiment 2: Single-Task Prompt Separability Across Execution Stages

Setup. We next analyze a more controlled setting to understand *when* visual latents encode instruction-specific information. We focus on a single task executed in a fixed scene and collect visual latents from two different trajectories that share the same initial setup but diverge due to different language prompts.

We assign different labels to the two trajectories and train a linear probe to distinguish them based solely on visual latents. This isolates prompt-related effects while holding task identity and scene constant.

Results. As shown in Fig. 16, we find that visual latents are *not* linearly separable during early execution stages (e.g., pre-grasp), where observations are nearly identical across prompts. However, after physical interaction occurs (e.g., post-grasp or object motion), linear separability emerges.

Interpretation. This stage-dependent behavior indicates that visual latents encode prompt-specific semantics only after the environment state begins to reflect the consequences of the instruction. Prior to interaction, language-conditioned differences are not visually observable, forcing the policy to rely on language input. After interaction, the visual state itself reveals task intent, enabling prompt inference from vision alone.

Summary. Together, these experiments show that visual latents can strongly encode task semantics, especially in multitask settings and later execution stages. This supports the hypothesis that VLA policies may over-index on visual context and under-utilize language, contributing to reduced steerability under mid-execution prompt changes.

APPENDIX F
PROPERTIES OF STEERABLE STATES

The steerable states definitions in §III-A imply several structural relationships between task competence, feasibility, and steerability.

Steerability implies task competence. For any policy π and task pair (i, j),

$$\mathcal{S}^{\text{steer}}_{i\to j}(\pi) \;\subseteq\; \mathcal{S}_j(\pi). \tag{22}$$

Explanation. Directional steerability from i to j requires that, after switching the instruction to l_j, the policy can complete task j with high probability. Therefore, any steerable state must also be a state from which task j is individually feasible.

Bidirectional steerability implies joint competence. For any task pair $i, j \in \mathcal{K}$ and a policy π, we define the *intersection of task visitation sets* as

$$\mathcal{S}^{\text{inter}}_{i,j}(\pi) \;\triangleq\; \mathcal{S}_i(\pi) \cap \mathcal{S}_j(\pi). \tag{23}$$

A state s belongs to $\mathcal{S}^{\text{inter}}_{i,j}(\pi)$ if the policy can successfully complete *both* tasks i and j with high probability when initialized from s under their respective language instructions. For any policy π,

$$\mathcal{S}^{\text{steer}}_{i\leftrightarrow j}(\pi) \;\subseteq\; \mathcal{S}^{\text{inter}}_{i,j}(\pi) \;=\; \mathcal{S}_i(\pi) \cap \mathcal{S}_j(\pi). \tag{24}$$

Explanation. Bidirectional steerability requires that the policy successfully execute both tasks after instruction switches in either direction. Consequently, steerability is only well-defined at states where both tasks are individually achievable, namely the states contained in the intersection of task visitation sets. This constraint fundamentally distinguishes steerability from spatial or temporal generalization: rather than requiring robustness across unseen states or time steps, steerability concerns the policy's ability to adapt its behavior when the task instruction is changed at a specific state during execution.

Steering is more restrictive than task feasibility. In general,

$$\mathcal{S}^{\text{steer}}_{i\to j}(\pi) \;\subseteq\; \mathcal{S}_j(\pi), \qquad \mathcal{S}^{\text{steer}}_{i\leftrightarrow j}(\pi) \;\subseteq\; \mathcal{S}^{\text{inter}}_{i,j}(\pi), \tag{25}$$

with strict inclusion commonly observed in practice. *Explanation.* Although a policy may be capable of executing a task from a given state, it may fail to adapt its behavior when the instruction is changed mid-execution. Empirically, we observe many states where both tasks are feasible, yet instruction switching does not reliably alter the policy's behavior, resulting in non-steerable states within the feasible region.

Steering is undefined outside the union of task visitation sets. For any policy π,

$$\mathcal{S}^{\text{steer}}_{i\to j}(\pi) \;\subseteq\; \mathcal{S}^{\text{union}}_{i,j}(\pi) \;\triangleq\; \mathcal{S}_i(\pi) \cup \mathcal{S}_j(\pi). \tag{26}$$

Explanation. Steering presupposes that at least one of the tasks is executable at the current state. Outside the union of task visitation sets, neither task can be completed, and instruction switching is therefore ill-defined.

Directional asymmetry. Directional steerability is not symmetric in general:

$$\mathcal{S}^{\text{steer}}_{i\to j}(\pi) \;\neq\; \mathcal{S}^{\text{steer}}_{j\to i}(\pi). \tag{27}$$

Explanation. Switching from task i to task j may succeed while the reverse transition fails, due to asymmetries in task structure, environment dynamics, or policy representations. For example, transitioning from a pre-grasping behavior to a placing behavior may be feasible, while reversing the instruction after placing may not restore a valid grasping configuration.

APPENDIX G
PROOF: CMI IS A PROXY FOR STEERABILITY

We now formalize the relationship between state-conditional mutual information and steerability coverage.

Without the loss of generality, fix a task pair (i, j) and a policy π. For a given state s, recall that the instruction-conditioned action distributions are given by $\pi(\cdot \mid s, l_i)$ and $\pi(\cdot \mid s, l_j)$. We define the state-conditional mutual information between actions and task instructions as

$$I^{\pi}_{i,j}(s) \;\triangleq\; I(A; L \mid S = s), \tag{28}$$

restricted to the two instructions l_i and l_j.

a) Mutual information and action distribution separation.: For a fixed state s, the conditional mutual information admits the closed-form expression

$$I^{\pi}_{i,j}(s) \;=\; \mathrm{JS}(\pi(\cdot \mid s, l_i) \parallel \pi(\cdot \mid s, l_j)), \tag{29}$$

where $\mathrm{JS}(\cdot\|\cdot)$ denotes the Jensen–Shannon divergence between the two instruction-conditioned action distributions.

By Pinsker's inequality, Jensen–Shannon divergence lower-bounds the squared total variation distance. Specifically, for any two probability measures p and q,

$$\mathrm{JS}(p\|q) \;\geq\; \tfrac{1}{2}\,\mathrm{TV}(p, q)^2, \tag{30}$$

where $\mathrm{TV}(p, q) = \frac{1}{2}\|p - q\|_1$ denotes total variation distance. Applying this inequality yields

$$I^{\pi}_{i,j}(s) \;\geq\; \tfrac{1}{2}\,\mathrm{TV}(\pi(\cdot \mid s, l_i), \pi(\cdot \mid s, l_j))^2\,. \tag{31}$$

b) Steerability implies instruction-dependent action separation.: We assume the existence of a constant $\varepsilon > 0$ such that for any state s that is bidirectionally steerable between tasks i and j, the policy must exhibit a nontrivial change in its action distribution under instruction switching:

$$s \in \mathcal{S}^{\text{steer}}_{i\leftrightarrow j}(\pi) \implies \mathrm{TV}(\pi(\cdot \mid s, l_i), \pi(\cdot \mid s, l_j)) \;\geq\; \varepsilon. \tag{32}$$

This assumption formalizes the requirement that successful task switching must induce meaningfully different control behaviors at the switching state.

Combining the above inequalities, we obtain the implication

$$s \in \mathcal{S}^{\text{steer}}_{i\leftrightarrow j}(\pi) \implies I^{\pi}_{i,j}(s) \;\geq\; \tau, \qquad \tau \triangleq \tfrac{1}{2}\varepsilon^2. \tag{33}$$

Equivalently,

$$I^{\pi}_{i,j}(s) < \tau \implies s \notin \mathcal{S}^{\text{steer}}_{i\leftrightarrow j}(\pi). \tag{34}$$

c) Implication for steerability coverage.: Let $\nu_{i,j}$ denote a reference distribution over feasible states (e.g., uniform over the task union set). Taking expectation over $s \sim \nu_{i,j}$ yields

$$\mathrm{SCR}_{i\leftrightarrow j}(\pi) \;\leq\; \Pr_{s\sim\nu_{i,j}}\left[I^{\pi}_{i,j}(s) \geq \tau\right]. \tag{35}$$

This result establishes that state-conditional mutual information provides a necessary-condition proxy for steerability: states with low mutual information cannot be reliably steered between tasks, and the fraction of high-mutual- information states upper-bounds the steerability coverage ratio. Consequently, mutual information enables rollout-free identification of low-steerability states and serves as a principled proxy for optimizing steerability in practice.

APPENDIX H
ADDITIONAL EXPERIMENT ON (LIBERO-OBJECT)

To verify that the improvements are not specific to LIBERO-Goal, we additionally evaluate ReSteer on three supplementary tasks in the LIBERO-Object scene. As shown in Table IV, ReSteer improves both single-task and steering success over the baseline, demonstrating that the framework transfers to a different scene and task set.

	Single Task (%)	Steerability (%)
Baseline	90.3	25.8
ReSteer	**97.2**	**44.1**

TABLE IV
SINGLE-TASK AND STEERING SUCCESS ON THREE SUPPLEMENTARY LIBERO-OBJECT TASKS.

APPENDIX I
HYPERPARAMETERS

We provide the hyperparameters used in our training and evaluation setup.

Training Configuration	
Base model	$\pi_{0.5}$
Training steps	2000
Observation history	1
Action horizon	10
Batch size	128
Finetuning method	Full finetuning
Data co-training ratio	LIBERO : SteerGen = 1 : 3 SteerGen : SRBC = 1 : 5
Learning rate schedule	Cosine decay
Warmup steps	10,000
Peak learning rate	5×10^{-5}
Decay steps	1,000,000
Final learning rate	5×10^{-5}
Evaluation Configuration	
Evaluation timestep range	$[0, 100]$
Evaluation step length	5
Rollouts per evaluation step	10
CMI sample size N	32

TABLE V
TRAINING AND EVALUATION HYPERPARAMETERS USED IN ALL EXPERIMENTS.