DEMOPRIOR: LEARNING UNSEEN ROBOT TASKS FROM A SINGLE DEMONSTRATION WITH ACTION-SPACE PRIORS

Anonymous authors
Paper under double-blind review

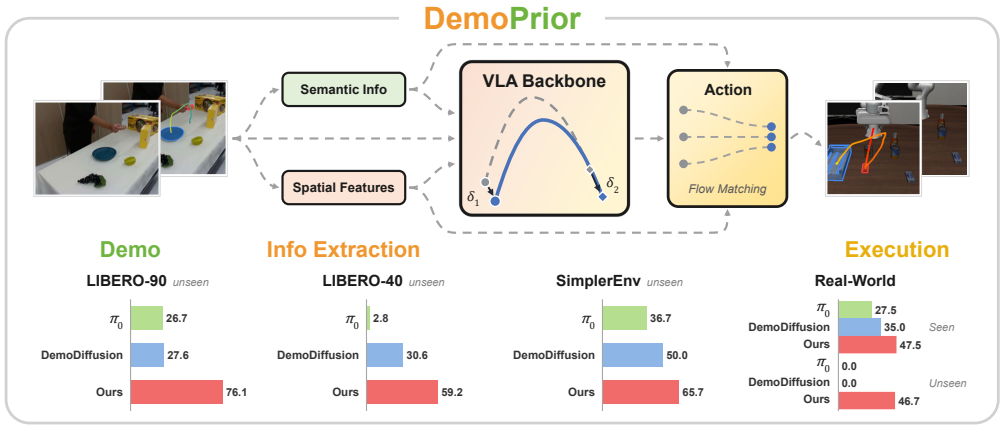

Figure 1: **DemoPrior is a one-shot video-to-skill framework that learns a reusable demo-to-skill transformation.** Inspired by how humans acquire new skills from demonstrations, DemoPrior extracts semantic structure and task-relative spatial structure from a single video demonstration, and learns a reusable *demo-to-skill transformation* that converts this information into a new executable skill. A vision–language–action (VLA) backbone learns the skill transformation to the current execution scene through closed-loop flow refinement. On real-world unseen tasks, DemoPrior achieves a 46.7% success rate using only a single demonstration video.

ABSTRACT

We propose DemoPrior, a one-shot video-to-skill framework that enables a learned vision-language-action (VLA) policy to acquire unseen manipulation skills from a single demonstration video. Existing approaches either learn adaptation mechanisms whose transfer remains tied to the tasks and variations covered during training, or use a test-time demonstration mainly as a weak conditioning signal or trajectory initialization. We instead propose to learn a reusable demo-to-skill transformation that converts information extracted from a new demonstration into executable behavior. Our key insight is that a demonstration provides two complementary forms of task information: semantic structure specifying *what* entities and interactions matter, and task-relative spatial structure specifying *how* the behavior should unfold in space. DemoPrior extracts these semantic and spatial cues from the video, transfers them to the current scene, and maps them into an action-space skill prior. A flow-based VLA is trained to refine this prior using closed-loop observations, learning how to turn video-derived structure into robot actions rather than generating a new skill from scratch. At test time, a single demonstration can therefore introduce a new skill without any task-specific policy update. Across our unseen-task evaluations, DemoPrior consistently improves demonstration-based skill transfer, and achieves a 46.7% success rate on real-world unseen tasks.

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

1 INTRODUCTION

Humans can often acquire a new manipulation skill from a single demonstration by understanding not only the desired outcome, but also which entities matter, how they interact, and how the behavior should unfold in space (Argall et al., 2009; Duan et al., 2017; Finn et al., 2017; Yu et al., 2018; Dasari & Gupta, 2021). For robots, the corresponding challenge is not simply to imitate the observed motion, but to convert task-relevant information from the demonstration into executable behavior under different scenes, object configurations, or embodiments. When the demonstrated skill is absent from the policy's training data, this requires inferring new behavior while retaining the pretrained policy's perception and closed-loop control capabilities. We thus ask: *can a robot learn a reusable mechanism that turns information extracted from a single demonstration into an executable skill for previously unseen tasks, without task-specific policy updates?*

Large-scale vision-language-action (VLA) models have substantially improved general-purpose robot control through robot data and vision-language pretraining (Zitkovich et al., 2023; Kim et al., 2025b; Black et al., 2025b;a). A growing line of work further studies one-shot robot learning, where a robot must acquire a new behavior from only one or a few demonstrations. These approaches span in-context or task adaptation for pretrained policies, single-video imitation, and cross-embodiment skill transfer (Sridhar et al., 2025; Zhao et al., 2025; Chen et al., 2025; Park et al., 2026; Kim et al., 2025a; Li et al., 2025a). However, demonstrations are still commonly used as task-specific conditions, representations, or trajectories, rather than as sources of reusable task structure that can be systematically learned and transformed into executable behavior. How to learn such a general demo-to-skill transformation therefore remains underexplored.

To address this problem, we propose DemoPrior, a one-shot video-to-skill framework that learns this transformation. As shown in Fig. 1, DemoPrior first extracts *semantic structure*, describing task-relevant entities and interactions, and *task-relative spatial structure*, describing the associated motion geometry. These structures are then grounded in the execution scene and mapped into an action-space skill prior. A flow-based VLA learns to transform this demonstration-derived structure into executable closed-loop behavior, allowing a new skill specification without any task-specific parameter update.

Concretely, DemoPrior establishes semantic correspondences between the demonstration and execution scenes and transfers motion in task-relative rather than absolute coordinates. The resulting action-space structure directly initializes flow-based action generation, allowing the VLA backbone to refine it using current observations (Black et al., 2025b; Lipman et al., 2023). The refinement is constrained to preserve reliable demonstration-derived structure while leaving execution-sensitive components adaptable through closed-loop control. In short, the VLA learns to refine a skill structure provided by the demonstration rather than generating an unseen skill from scratch.

We evaluate the generalization of DemoPrior across tasks, scenes, and embodiments on LIBERO (Liu et al., 2023), SimplerEnv (Li et al., 2025b), cross-domain transfer from human demonstrations, and a physical UR5 robot. On real-world tasks unseen during policy training, it achieves 46.7% success from a single demonstration without task-specific test-time updates.

In summary, our main contributions are:

- We introduce a one-shot demo-to-skill formulation that learns a reusable transformation from demonstration-derived semantic and task-relative spatial structure to executable robot skills.
- DemoPrior maps the demonstration-derived structure into an action-space skill prior and trains a flow-based VLA to refine it into closed-loop behavior while preserving reliable task structure.
- DemoPrior demonstrates cross-task and cross-domain transfer across LIBERO, SimplerEnv, human demonstration videos, and a real UR5 robot, achieving 46.7% success on real-world unseen tasks from a single demonstration with no task-specific test-time update.

2 RELATED WORK

Vision-language-action models. Vision-language-action (VLA) models combine visual perception, language understanding, and low-level control through large-scale robot data and vision-language pretraining (Li et al., 2024; Shukor et al., 2025; Reuss et al., 2025). Generalist policies such as RT-1 (Brohan et al., 2023), Octo (Ghosh et al., 2024), RT-2 (Zitkovich et al., 2023), Open-

VLA (Kim et al., 2025b), and π_0 (Black et al., 2025b) broaden task, scene, and embodiment coverage. Recent work further improves new-task adaptation through in-context learning, composable skill representations, demonstration conditioning, and additional generalization training (Sridhar et al., 2025; Zhao et al., 2025; Chen et al., 2025; Bu et al., 2025; Black et al., 2025a).

These approaches improve VLA generalization, but their transfer capabilities are typically learned from the task and variation coverage available during training. A less explored direction is to fine-tune a pretrained VLA into a reusable policy that converts task-relevant structure from a test-time demonstration into an executable skill. DemoPrior targets this capability through a reusable *demo-to-skill transformation*.

Learning robot skills from videos. Videos provide information beyond task goals by capturing object interactions, motion, and spatial structure (Sermanet et al., 2018; Wen et al., 2024; Bharadhwaj et al., 2024). They have therefore been used to learn transferable representations, states, rewards, and action-related knowledge from action-free data (Xu et al., 2025; Ko et al., 2024; Nair et al., 2023; Shao et al., 2021; Ma et al., 2023a; Sontakke et al., 2023; Ma et al., 2023b). More recent methods learn latent actions and transferable skill representations from heterogeneous or cross-embodiment videos (Ye et al., 2025; Bu et al., 2025; Xu et al., 2023; Kim et al., 2025a).

These representations provide useful abstractions for transfer, but compact video representations can discard task-specific spatial and interaction structure needed to instantiate a new skill. DemoPrior instead preserves semantic and task-relative spatial structure from the demonstration and transfers them into the action space for skill generation.

One-shot skill learning from demonstrations. One-shot imitation studies how a robot can acquire a manipulation skill from a single demonstration (Duan et al., 2017; Finn et al., 2017; Yu et al., 2018; Dasari & Gupta, 2021). Geometry- and correspondence-based approaches transfer demonstrated behavior using object correspondences, 3D structure, trajectories, or functional relations (Zhu et al., 2026; Kerr et al., 2025; Li et al., 2025a; Tang et al., 2025a;b). Recent methods further combine one-shot demonstrations with learned policies through trajectory initialization, demonstration conditioning, latent planning, or test-time adaptation (Park et al., 2026; Chen et al., 2025; Son et al., 2026; Feng et al., 2026).

Across these approaches, the demonstration is commonly used as a representation, conditioning signal, initialization, or input to a transfer procedure. What remains less explored is a reusable policy-level mapping from demonstration-derived task structure to executable behavior. DemoPrior addresses this gap by extracting semantic and task-relative spatial structure from a single demonstration, converting it into an action draft, and fine-tuning a flow-based VLA to learn a reusable *demo-to-skill transformation*. At test time, a new skill is introduced through the demonstration alone, without task-specific parameter updates.

3 METHOD

We propose DemoPrior, a one-shot video-to-skill framework that learns a reusable transformation from demonstration-derived information to executable skills, as shown in Fig. 2. Because demonstration and execution may differ in scene layout, object locations, viewpoint, and embodiment, DemoPrior first transfers semantic and task-relative spatial structure from the video to the current scene, producing an *action draft*. A pretrained flow-based VLA is then fine-tuned to convert this draft into closed-loop actions. The method consists of demonstration-to-draft transfer (Sec. 3.1) and draft-to-skill generation (Sec. 3.2).

Given a demonstration video V, the current visual observation o, and a language instruction ℓ, our goal is to execute the demonstrated skill in the current scene, including skills not covered during policy training. DemoPrior extracts *semantic structure*, representing task-relevant entities and their interactions, and *task-relative spatial structure*, representing the associated spatial motion. These structures are transferred to the current scene and expressed in the native VLA action space $\mathcal{A} \subseteq \mathbb{R}^{d_a}$, where d_a denotes the action dimensionality of the current embodiment and control interface, yielding an *action draft* $\mathbf{p} \in \mathbb{R}^{d_a}$. The VLA then learns to map $\mathbf{p}$, together with o and ℓ, to executable actions. This learned mapping provides the reusable *demo-to-skill transformation* used to introduce new skills at test time without task-specific parameter updates.

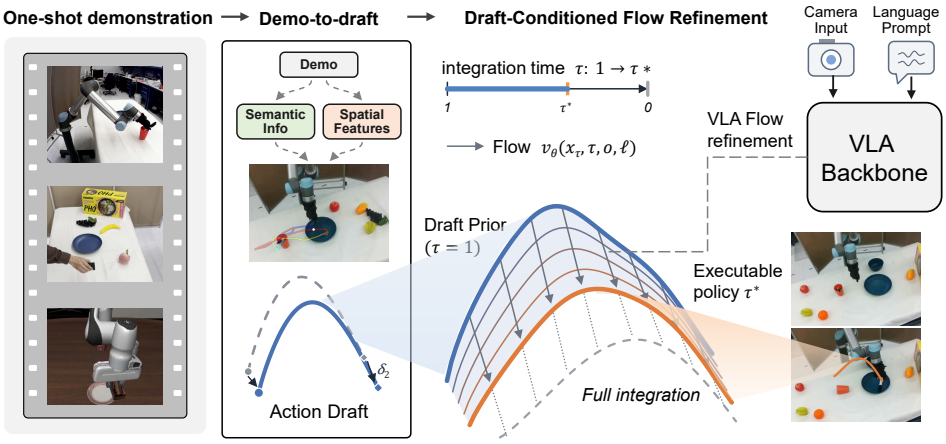

Figure 2: **Overview of DemoPrior.** Given a single demonstration, DemoPrior first extracts task-relevant semantic structure and task-relative spatial structure, transfers them to the current scene, and constructs an action-space draft. Crucially, DemoPrior trains a VLA backbone to learn a reusable *demo-to-skill transformation* that maps this action draft directly into a new executable skill in the current scene, rather than learning a separate policy for each new task. This transformation is jointly constrained by the demonstration-derived prior and the pretrained VLA's generative capability: the former provides the task structure of the new skill, while the latter adapts execution using the current observation. The stopping time τ^* controls their relative influence, preventing useful demonstration structure from being overwritten and guiding generation toward a more executable trajectory.

3.1 From Demonstration to Action Draft

This stage converts the demonstration video and current scene into an action-space draft $\mathbf{p}$. Since object locations and interaction geometry may change across scenes, we transfer task-relative structure rather than directly replaying the demonstrated motion.

Semantic and motion grounding. We identify task-relevant entities, recover their motion in the demonstration, and locate their semantic counterparts in the current scene. We use SAM2 (Ravi et al., 2025) for object segmentation, CoTracker (Karaev et al., 2025) for temporal motion tracking, and SigLIP2 (Tschannen et al., 2025) for cross-scene semantic correspondence. An interaction location is defined as the contact location on an object surface where the end-effector or gripper performs a task-relevant interaction. These locations are first identified in the 2D demonstration video and then grounded to corresponding locations in the execution scene based on perceived depth. We denote the resulting matched interaction anchors as $\{\mathbf{q}_m^D, \mathbf{q}_m^E\}_{m=1}^M$, where $\mathbf{q}_m^D, \mathbf{q}_m^E \in \mathbb{R}^{d_s}$ represent the same semantic interaction in a shared task-relative spatial representation, and d_s denotes the spatial dimensions that can be reliably transferred from the demonstration.

Task-relative spatial transfer. The matched locations specify where the interactions should occur in the current scene, and then we recover the motion specifying how the skill unfolds between them. Let $\mathbf{e}(t) \in \mathbb{R}^{d_s}$ denote the demonstrated task-relative motion and t_m the trajectory position associated with $\mathbf{q}_m^D$. The spatial adjustment for interaction location m is $\Delta_m = \mathbf{q}_m^E - \mathbf{q}_m^D$. For $t_m \le t < t_{m+1}$, we transfer the motion as

$$\tilde{\mathbf{e}}(t) = \mathbf{e}(t) + (1 - \alpha_m(t))\,\Delta_m + \alpha_m(t)\Delta_{m+1}, \tag{1}$$

$$\alpha_m(t) = \frac{L(t) - L(t_m)}{L(t_{m+1}) - L(t_m)}, \tag{2}$$

where $L(t)$ is the cumulative arc length of the demonstrated motion.

This transformation adapts the interaction geometry while preserving the task-relative motion structure. The transferred motion and other reliable interaction information are then converted into the native VLA action representation to form the action draft $\mathbf{p}$. Some action dimensions may not be reliably inferred from a monocular demonstration, such as dimensions requiring depth or embodiment-specific information; these dimensions are left for the VLA to generate (Sec. 3.2).

3.2 From Action Draft to Executable Skill

The action draft captures the structure of the new skill but may not be directly executable. We therefore fine-tune a pretrained flow-based VLA to transform the draft, conditioned on the current observation and language instruction, into executable actions.

Action draft as flow source. Standard flow-based VLAs generate actions from an unstructured Gaussian noise source (Black et al., 2025b; Lipman et al., 2023). DemoPrior instead initializes supported action dimensions from the demonstration-derived draft. Let $\bar{\mathbf{p}} \in \mathbb{R}^{d_a}$ denote the normalized action draft and $\mathcal{F} \subseteq \{1, \ldots, d_a\}$ the set of action dimensions that cannot be reliably specified by the demonstration. For action dimension j, we define

$$s_j = \begin{cases} \bar{p}_j + \sigma \epsilon_j, & j \notin \mathcal{F}, \\ \epsilon_j, & j \in \mathcal{F}, \end{cases} \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{3}$$

Thus, supported dimensions begin from task-specific structure, while unsupported dimensions begin from the Gaussian noise source. We fine-tune the VLA with the standard flow-matching objective (Black et al., 2025b; Lipman et al., 2023), so that its conditional velocity field $v_\theta(\mathbf{x}_\tau, \tau, o, \ell) \in \mathbb{R}^{d_a}$ maps demonstration-derived structure toward executable actions.

Structure-constrained action generation. During action generation, the draft provides the structure of the new skill, while the VLA uses the current observation to correct execution-related errors. Excessive flow integration, however, may overwrite useful demonstration structure. We therefore introduce a stopping parameter τ_j^* for each action dimension:

$$\frac{dx_{\tau,j}}{d\tau} = \mathbb{I}[\tau \geq \tau_j^*]\, v_{\theta,j}(\mathbf{x}_\tau, \tau, o, \ell), \qquad \tau : 1 \to 0, \tag{4}$$

where $v_{\theta,j}$ is the j-th component of the flow-matching velocity field.

A larger τ_j^* retains more demonstration-derived structure, while a smaller value allows stronger VLA correction. Unsupported dimensions $j \in \mathcal{F}$ use the full flow. Thus, τ^* regulates the balance between the demonstration prior and VLA generation, allowing the action to move toward an executable trajectory without unnecessarily overwriting useful skill structure.

Implementation details. DemoPrior can be instantiated with flow-matching VLAs that expose their native action representation and generation process. We use the pretrained OpenPI π_0 backbone (Black et al., 2025b) and fine-tune it with the draft-conditioned flow objective described above. At test time, the policy parameters remain fixed and new skills are introduced only through demonstration videos. Additional preprocessing, geometric transfer, inverse-kinematics, flow, τ setup, and robot-specific details are provided in the appendix.

4 Experiments

We evaluate DemoPrior along three dimensions: **1)** whether a single demonstration can introduce skills unseen during policy training; **2)** whether the learned video-to-skill transformation generalizes across scenes, demonstration domains, and embodiments; and **3)** whether performance is robust to variation in the provided demonstration. We conduct experiments on LIBERO, SimplerEnv, self-recorded human demonstration videos, and a physical UR5 robot.

Unless otherwise specified, each task is evaluated over 10 rollouts. Demonstrations are provided only at test time, with no task-specific policy update on unseen tasks. Detailed experimental settings are provided in the appendix.

4.1 Experimental Setup

LIBERO. We evaluate two settings from LIBERO (Liu et al., 2023). For **LIBERO-40**, we directly evaluate the policy trained on the LIBERO-90 training split on 40 tasks from LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, and LIBERO-10, without any additional training. None of these 40 tasks appears in LIBERO-90. For **LIBERO-90**, we use 72 of the 90 tasks for policy training, covering diverse manipulation instructions such as "put the red mug on the plate to the left." The

Table 1: Success rate on LIBERO. LIBERO-40 contains only tasks excluded from policy training, while LIBERO-90 reports both seen and held-out tasks. For DemoPrior, DemoDiffusion, and Skill Prior, each task is evaluated using five independently selected demonstrations, with 10 rollouts per demonstration, and we report the mean across demonstrations.

	LIBERO-40 Disjoint Tasks					**LIBERO-90**	
Method	Spatial	Object	Goal	Long	Avg.	Seen	Unseen
π_0	1.0	0.0	6.0	4.0	2.8	85.0	26.7
UniVLA	2.0	0.0	7.0	2.0	2.8	**90.0**	27.2
DemoDiffusion	51.8	18.0	43.3	9.4	30.6	68.1	27.6
Skill Prior (Open-loop)	41.8	35.0	58.0	11.2	36.5	41.6	57.7
DemoPrior (Ours)	**85.4**	**70.8**	**62.7**	**18.0**	**59.2**	76.2	**76.1**

remaining 18 disjoint tasks are held out from training and used for unseen-task evaluation. We report performance on both the 72 seen tasks and the 18 unseen tasks.

SimplerEnv. We evaluate on the WidowX–Bridge environment in SimplerEnv (Li et al., 2025b). The policy is fine-tuned on Bridge V2, which contains diverse manipulation instructions such as "put the broccoli in the pot." During training, we hold out 10% of instruction groups and additionally remove all instruction families associated with the four standard SimplerEnv WidowX tasks. We evaluate on these four standard tasks and five additional tasks whose instructions never appear in the Bridge V2 training split. Because the released UniVLA checkpoint has seen the Bridge tasks during training, we report it separately as a seen-task reference.

High-budget LLM reasoning. We compare DemoPrior with LLM-assisted control on ten held-out LIBERO-90 tasks, evaluating each task over five episodes. Claude Opus 5.5 and GPT-6 Astra are paired with a π_0 policy fine-tuned on LIBERO and are allowed to intervene at any time during execution. The LLMs have access to the same observations available to π_0, including camera inputs and robot states such as joint angles. Whenever an intervention is deemed necessary, execution can be paused to allow offline LLM reasoning; the resulting intervention is then applied before resuming.

Learning from human demonstrations. We additionally evaluate self-recorded human demonstration videos to test whether skill transfer remains effective under changes in embodiment and scene configuration. We use six simulation tasks: pudding-left-of-plate, butter-to-tray, black-bowl-to-plate, butter-to-basket, mug-to-right-plate, and mug-right-of-caddy. We further evaluate one real-world task, banana-to-drawer-top. For each task, we provide a single human demonstration video and extract task-relevant semantic and spatial information using SAM2 (Ravi et al., 2025), SigLIP2 (Tschannen et al., 2025), and CoTracker (Karaev et al., 2025). Because the other baselines considered in our study do not learn new skills from demonstration videos, we evaluate only our method in this setting.

Real-world UR5. We further evaluate DemoPrior on a physical UR5 robot. The policy is trained on a self-collected dataset containing 333 demonstrations across eight manipulation tasks. We evaluate on four seen tasks included in the training set and three unseen tasks, where the unseen skills are introduced through demonstration videos only at test time.

Baselines. We compare against four baselines. π_0 is the language-conditioned VLA backbone without demonstration input. UniVLA (Bu et al., 2025) represents video demonstrations using learned latent actions. DemoDiffusion (Park et al., 2026) uses the same π_0 backbone and demonstration-derived information as DemoPrior, but incorporates the demonstration during inference-time generation. We also include Skill Prior (Open-loop), which directly executes the transferred skill prior without closed-loop refinement. Implementation details and baseline-specific settings are provided in the appendix.

4.2 Main Results

We evaluate DemoPrior on LIBERO and SimplerEnv, focusing on generalization to skills excluded from policy training. Across both benchmarks, the gains are most significant on unseen tasks, supporting our central goal of acquiring new skills from a single demonstration

LIBERO. Table 1 first reports results on LIBERO-40, where all evaluated tasks are excluded from policy training. DemoPrior achieves 59.2% average success and consistently outperforms the baselines across all four task suites, indicating stronger transfer to tasks outside the training distribution.

Table 2: Success rate on SimplerEnv. UniVLA is reported as a seen-task reference, while the remaining methods are evaluated on tasks excluded from their policy training.

		Official-4	**Additional-5**					
Method	Setting	Avg.	Egg-Plate	Egg-Towel	Cube-Plate	Cube-Towel	Y-G	Avg.
UniVLA†	Seen	51.0	62.5	50.0	91.7	87.5	12.5	60.8
π_0	Unseen	32.3	54.2	37.5	37.5	37.5	16.7	36.7
DemoDiffusion	Unseen	46.9	45.8	50.0	70.8	62.5	20.8	50.0
Skill Prior (Open-loop)	Unseen	31.9	35.8	33.3	60.0	59.2	37.5	45.2
DemoPrior (Ours)	Unseen	**51.2**	**70.0**	**64.6**	**73.3**	**72.5**	**48.3**	**65.7**

†UniVLA uses the released Bridge/Simpler checkpoint, whose training data cover the evaluated Bridge tasks.

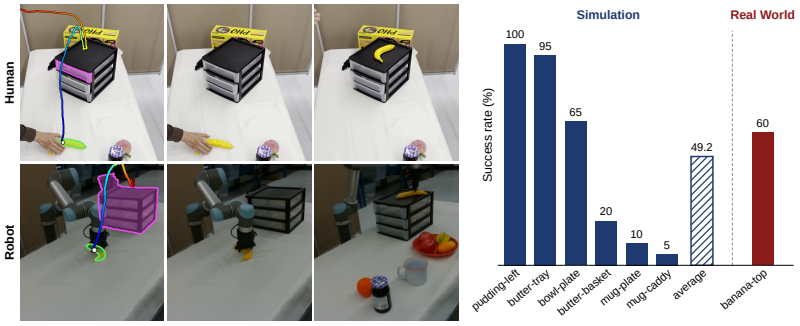

Figure 3: Cross-domain transfer from human demonstration videos to robot execution. Left: representative frames from a self-recorded human demonstration. Right: average success rates over six manipulation tasks.

On the held-out LIBERO-90 tasks, DemoPrior further reaches 76.1% and substantially improves over both the open-loop skill prior and other baselines. This result shows that preserving the transferred skill structure while refining it through closed-loop control is important for executing unseen skills. On seen tasks, DemoPrior performs below π_0 and UniVLA. This is because DemoPrior learns a general transformation from video-derived skill structure to executable actions, rather than directly optimizing expert trajectories for individual training tasks.

SimplerEnv. Table 2 evaluates transfer to the WidowX–Bridge domain. DemoPrior achieves 51.2% on the official tasks and 65.7% on the additional tasks, outperforming all baselines evaluated under the unseen-task protocol. It also matches or exceeds the released UniVLA checkpoint despite UniVLA having seen the Bridge tasks during training. These results indicate that DemoPrior learns a reusable transformation from information extracted from a demonstration to executable skills in a new scene, rather than a transformation tied to the training domain.

Learning from human demonstrations. As shown in Fig. 3, we further evaluate DemoPrior using self-recorded human demonstration videos. DemoPrior achieves 49.2% average success across six simulation tasks and 60.0% success on the real-world task. These results show that DemoPrior can extract transferable skill information from videos, and that this capability extends to human demonstration videos despite changes in scene and embodiment.

Comparison with high-budget LLM reasoning. Table 3 compares DemoPrior with high-budget test-time LLM reasoning systems on ten held-out LIBERO-90 tasks. DemoPrior achieves 72.0% success from a single demonstration, compared with 76.0% for Claude Opus 5.5 + π_0 and 58.0% for GPT-6 Astra + π_0. The LLM-based systems achieve this performance with substantial test-time reasoning cost. Claude Opus 5.5 consumes an average of 12.5M input and 114K output tokens per episode, taking 21.1 minutes, while GPT-6 Astra uses 5.1M input and 26K output tokens and takes 17.1 minutes per episode. In contrast, DemoPrior extracts task-relevant skill information from a single demonstration video and requires no iterative LLM reasoning during execution. These results show that video-based skill transfer can achieve competitive performance on unseen tasks with substantially lower test-time reasoning overhead.

Real-world robot results. We evaluate DemoPrior on a real-world UR5 using four seen and three unseen tasks, with unseen skills introduced through a single test-time demonstration. As shown in

Table 3: Comparison on 10 unseen manipulation tasks, with 5 trials per task. DemoPrior approaches the performance of high-budget LLM-assisted systems using only a single demonstration and without iterative LLM reasoning. GPT-6 Astra + π_0 and Claude Opus 5.5 + π_0 require substantial test-time reasoning time and token consumption.

Method	π_0	DemoDiffusion	**Ours**	GPT-6 Astra + π_0	Claude Opus 5.5 + π_0
Success Rate (%)	22.0	30.0	**72.0**	58.0	76.0
LLM Reasoning Time (min/episode)	–	–	–	17.1	21.1
Input / Output Tokens	–	–	–	5.1M / 26K	12.5M / 114K

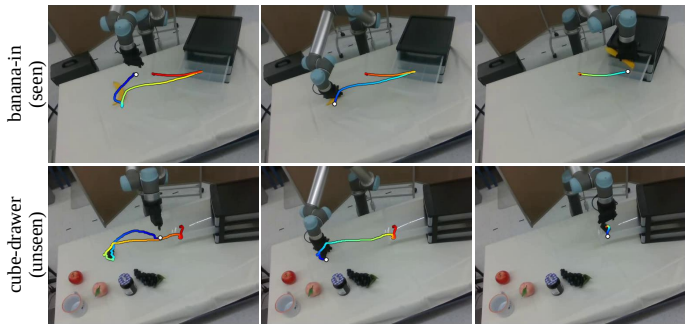

Figure 4: Qualitative real-world UR5 results on a seen task (top) and an unseen task (bottom). Colored curves show the executed end-effector trajectories. DemoPrior preserves the demonstration-derived motion structure while adapting to the current scene, including on skills unseen during policy training.

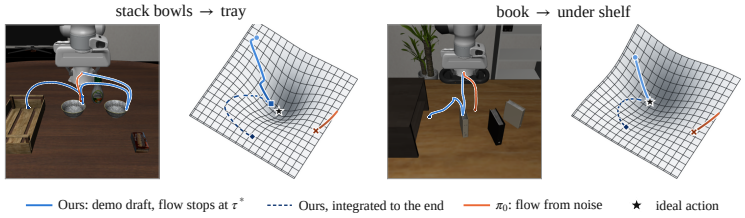

Figure 5: Flow refinement on two unseen LIBERO-90 tasks. Each statistical landscape corresponds to one real flow trajectory and is constructed from empirical failure rates, with lower regions indicating better execution. DemoPrior transforms the video-derived draft toward the desired low-failure region, while τ^* stops the flow near its best-performing point before further integration moves it away.

Table 4, DemoPrior achieves 47.5% success on seen tasks and 46.7% on unseen tasks, while π_0 and DemoDiffusion fail on all unseen tasks. This is partly because these baselines do not explicitly learn a reusable mapping from demonstration-derived skill information to executable behavior, which becomes more challenging under real-world perception and control shifts.

Fig. 4 further shows that DemoPrior preserves the overall motion structure of the demonstration while adapting the executed trajectory to the current scene through closed-loop control. Together, these results demonstrate transfer of video-derived skills to real-world execution, including skills absent from policy training.

5 Analysis

From action generation to action refinement. Figure 5 visualizes how the action evolves during flow integration on unseen LIBERO-90 tasks. The landscape is constructed from empirical failure rates, with lower regions corresponding to better executions. A clear pattern emerges from the trajectories. DemoPrior moves progressively from a high-failure region toward the low-failure basin around the desired execution. This directional movement suggests that the learned flow is transforming the semantic and spatial information extracted from the demonstration toward an executable skill trajectory. In contrast, the original π_0 flow remains in a nearby region and does not show the same

Table 4: Success rate on real-world UR5 tasks. DemoPrior achieves the strongest overall performance, demonstrating that the learned demo-to-skill transformation extends to real-world robot execution, including tasks unseen during policy training.

	Seen Tasks					*Unseen Tasks*			
Method	Cube-Bowl	Banana-Top	Banana-Drawer	Marker-Plate	Avg.	Cube-Drawer	Banana-Plate	Cube-Plate	Avg.
π_0	40.0	**30.0**	20.0	20.0	27.5	0.0	0.0	0.0	0.0
UniVLA	0.0	0.0	10.0	10.0	5.0	10.0	10.0	0.0	6.7
DemoDiff.	30.0	20.0	**60.0**	30.0	35.0	0.0	0.0	0.0	0.0
Ours	**60.0**	**30.0**	40.0	**60.0**	**47.5**	**40.0**	**60.0**	**40.0**	**46.7**

Table 5: Success rate of ablations on different uses of the demonstration-derived action draft, averaged over five demonstrations per task. Using the action draft as the flow source with partial integration performs best across all settings.

	LIBERO-90 Unseen	LIBERO-40	SimplerEnv-Official	SimplerEnv-Additional
No draft (π_0)	26.7	2.8	32.3	36.7
Open-loop action draft	57.7	36.5	31.9	45.2
Conditioning	19.4	3.3	23.1	22.3
Residual	27.8	1.8	19.4	23.0
Flow source + full integration	46.1	27.9	21.9	13.3
Flow source + partial integration (Ours)	**76.1**	**59.2**	**51.2**	**65.7**

movement toward the desired execution. The stopping point is equally important. The semantic and spatial structure carried by the draft constrains the integration horizon through τ^*, allowing the flow to stop near the best-performing region. Without this constraint, continuing the flow can carry the trajectory beyond that region and away from the desired action. We further analyze the effect of the stopping point τ in Appendix A. Together, the trajectory and stopping behavior show that DemoPrior learns a flow from video-derived skill information toward executable behavior, while preserving that information at the point where it is most useful for execution.

How should the action draft be used? Table 5 compares different ways of incorporating the demonstration-derived action draft. *Open-loop action draft* directly executes the draft without policy refinement. *Conditioning* provides the action draft as an additional conditioning signal to the original policy, while *Residual* uses it to predict an additive correction to the policy action. Both indirect variants provide limited gains, while open-loop execution remains sensitive to execution errors. A more effective strategy is to use the action draft directly as the *flow source*. With *full integration*, however, the flow can move away from the task-specific structure carried by the draft. In contrast, *flow source + partial integration*, used by DemoPrior, performs best across all evaluated settings. These results support the two key design choices of DemoPrior: the action draft should directly guide action generation, while refinement should stop before the transferred skill structure is weakened.

Limitations. Our current evaluation focuses on single-arm manipulation, and extending DemoPrior to dual-arm or more complex multi-effector systems remains an interesting direction for future work. In addition, we currently assume that each demonstration video contains a single task, rather than multiple temporally composed skills. The flow-matching refinement also introduces additional inference steps, and improving its runtime efficiency would make the method more practical for higher-frequency control.

6 Conclusion

We introduced DemoPrior, a one-shot video-to-skill framework for extending pretrained VLAs to manipulation skills outside their training distribution. The central idea is to treat a demonstration not merely as an additional conditioning signal, but as a source of task-specific action structure. DemoPrior transfers this structure into the robot action space and incorporates it directly into flow-based action generation, allowing the pretrained policy to refine a demonstrated skill through closed-loop interaction. Experiments across LIBERO, SimplerEnv, and a real-world UR5 robot show substantial gains on unseen tasks and demonstrate transfer across scenes and embodiments. Overall, our results suggest that demonstration-derived action priors provide a practical interface for combining one-shot skill acquisition with the general execution capabilities of pretrained VLAs.

REPRODUCIBILITY STATEMENT.

We provide detailed experimental settings, training configurations, evaluation protocols, data splits, and additional implementation details in the appendix. We also include an anonymized implementation of the key components of DemoPrior in the supplementary material. The datasets and benchmarks used in our experiments are publicly available, and the complete training and evaluation pipeline will be released upon publication.

AI USE STATEMENT

We used generative AI tools to assist with literature search and discovery and to edit and polish the manuscript for clarity and readability. All AI-assisted content was independently reviewed and verified by at least two authors to ensure accuracy and consistency with the intended scientific content. The authors take full responsibility for the final manuscript.

REFERENCES

Brenna D. Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009. ISSN 0921-8890.

Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation. In *European Conference on Computer Vision (ECCV)*, 2024.

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

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

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael S Ryoo, Grecia Salazar, Pannag R Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan H Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: Robotics Transformer for Real-World Control at Scale. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023.

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Learning to Act Anywhere with Task-centric Latent Actions. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025.

Guangyan Chen, Meiling Wang, Qi Shao, Zichen Zhou, Weixin Mao, Te Cui, Minzhao Zhu, Yinan Deng, Luojie Yang, Zhanqi Zhang, Yi Yang, Hua Chen, and Yufeng Yue. See once, then act: Vision-language-action model with task learning from one-shot video demonstrations, 2025.

Sudeep Dasari and Abhinav Gupta. Transformers for one-shot visual imitation. In Jens Kober, Fabio Ramos, and Claire Tomlin (eds.), *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pp. 2071–2084. PMLR, 16–18 Nov 2021.

Yan Duan, Marcin Andrychowicz, Bradly Stadie, OpenAI Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. One-shot imitation learning. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Yusen Feng, Bingchen Han, Jiangran Lyu, Kai Liu, Yixin Zheng, Yuxuan Wan, Weiheng Liu, Sun Han, Ruiqin Li, Yulong Zhang, Fangfu Liu, Xuesong Shi, Libin Liu, Yizhou Wang, Zhizheng Zhang, and He Wang. Wam-ttt: Steering world-action models by watching human play at test time, 2026.

Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot visual imitation learning via meta-learning. In Sergey Levine, Vincent Vanhoucke, and Ken Goldberg (eds.), *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pp. 357–368. PMLR, 13–15 Nov 2017.

Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Quan Vuong, Ted Xiao, Pannag R Sanketi, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An Open-Source Generalist Robot Policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024.

Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 18–35, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-73033-7.

Justin Kerr, Chung Min Kim, Mingxuan Wu, Brent Yi, Qianqian Wang, Ken Goldberg, and Angjoo Kanazawa. Robot see robot do: Imitating articulated object manipulation with monocular 4d reconstruction. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 587–603. PMLR, 06–09 Nov 2025.

Hanjung Kim, Jaehyun Kang, Hyolim Kang, Meedeum Cho, Seon Joo Kim, and Youngwoon Lee. Uniskill: Imitating human videos via cross-embodiment skill representations. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 4269–4294. PMLR, 27–30 Sep 2025a.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R 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. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 06–09 Nov 2025b.

Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B Tenenbaum. Learning to act from actionless videos through dense correspondences. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 40938–40958, 2024.

Jinhan Li, Yifeng Zhu, Yuqi Xie, Zhenyu Jiang, Mingyo Seo, Georgios Pavlakos, and Yuke Zhu. Okami: Teaching humanoid robots manipulation skills through single video imitation. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 299–317. PMLR, 06–09 Nov 2025a.

Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, Hang Li, and Tao Kong. Vision-language foundation models as effective robot imitators. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 26703–26721, 2024.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Oier Mees, Karl Pertsch, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating real-world robot manipulation policies in simulation. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 3705–3728. PMLR, 06–09 Nov 2025b.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling, 2023.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 44776–44791. Curran Associates, Inc., 2023.

Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. LIV: Language-image representations and rewards for robotic control. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 23301–23320. PMLR, 2023a.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. VIP: Towards universal visual reward and representation via value-implicit pre-training. In *The Eleventh International Conference on Learning Representations*, 2023b.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. In Karen Liu, Dana Kulic, and Jeff Ichnowski (eds.), *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 892–909. PMLR, 14–18 Dec 2023.

Sungjae Park, Homanga Bharadhwaj, and Shubham Tulsiani. Demodiffusion: One-shot human imitation using pre-trained diffusion policy. In *International Conference on Robotics and Automation (ICRA)*, 2026.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollar, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 28085–28128, 2025.

Moritz Reuss, Hongyi Zhou, Marcel Rühle, Ömer Erdinç Yağmurlu, Fabian Otto, and Rudolf Lioutikov. Flower: Democratizing generalist robot policies with efficient vision-language-flow models. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 3736–3761. PMLR, 27–30 Sep 2025.

Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, and Sergey Levine. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1134–1141. IEEE, 2018.

Lin Shao, Toki Migimatsu, Qiang Zhang, Karen Yang, and Jeannette Bohg. Concept2robot: Learning manipulation concepts from instructions and human demonstrations. *The International Journal of Robotics Research*, 40(12-14):1419–1434, 2021.

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.

Jaehyeon Son, Junhyun Kim, Kyle Kam, Jeremiah Coholich, Seok Joon Kim, Jinhoo Kim, Chris Dongjoo Kim, Jaemin Cho, Dieter Fox, and Zsolt Kira. Seetraceact: Visibility-aware latent planning from cross-embodiment demonstration videos, 2026.

Sumedh Sontakke, Jesse Zhang, Séb Arnold, Karl Pertsch, Erdem Bı yık, Dorsa Sadigh, Chelsea Finn, and Laurent Itti. Roboclip: One demonstration is enough to learn robot policies. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 55681–55693. Curran Associates, Inc., 2023.

Kaustubh Sridhar, Souradeep Dutta, Dinesh Jayaraman, and Insup Lee. Ricl: Adding in-context adaptability to pre-trained vision-language-action models. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 5022–5038. PMLR, 27–30 Sep 2025.

Chao Tang, Anxing Xiao, Yuhong Deng, Tianrun Hu, Wenlong Dong, Hanbo Zhang, David Hsu, and Hong Zhang. Functo: Function-centric one-shot imitation learning for tool manipulation. *arXiv preprint arXiv:2502.11744*, 2025a.

Chao Tang, Anxing Xiao, Yuhong Deng, Tianrun Hu, Wenlong Dong, Hanbo Zhang, David Hsu, and Hong Zhang. Mimicfunc: Imitating tool manipulation from a single human video via functional correspondence. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 4473–4492. PMLR, 27–30 Sep 2025b.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, Olivier Hénaff, Jeremiah Harmsen, Andreas Steiner, and Xiaohua Zhai. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features, 2025.

Chuan Wen, Xingyu Lin, John Ian Reyes So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point Trajectory Modeling for Policy Learning. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024.

Mengda Xu, Zhenjia Xu, Cheng Chi, Manuela Veloso, and Shuran Song. Xskill: Cross embodiment skill discovery. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 3536–3555. PMLR, 06–09 Nov 2023.

Mengda Xu, Zhenjia Xu, Yinghao Xu, Cheng Chi, Gordon Wetzstein, Manuela Veloso, and Shuran Song. Flow as the cross-domain manipulation interface. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2475–2499. PMLR, 06–09 Nov 2025.

Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Se June Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, Lars Liden, Kimin Lee, Jianfeng Gao, Luke Zettlemoyer, Dieter Fox, and Minjoon Seo. Latent action pretraining from videos. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 28213–28239, 2025.

Tianhe Yu, Chelsea Finn, Sudeep Dasari, Annie Xie, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot imitation from observing humans via domain-adaptive meta-learning. In *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

Ruihan Zhao, Tyler Ingebrand, Sandeep Chinchali, and Ufuk Topcu. Mos-vla: A vision-language-action model with one-shot skill adaptation, 2025.

Yifeng Zhu, Arisrei Lim, Peter Stone, and Yuke Zhu. Vision-based manipulation from single human video with open-world object graphs. *Autonomous Robots*, 50(2):27, 2026.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao

Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 06–09 Nov 2023.

Appendix

A Ablation Studies

We conduct additional ablations on LIBERO-90 to study the robustness of DemoPrior and the effect of structure-preserving integration. All ablations are performed at test time without additional policy training.

A.1 Gaussian Perturbations

We perturb the action-space prior with Gaussian noise of different strengths, using

$$\sigma \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5\}.$$

Table 6 reports the results on unseen LIBERO-90 tasks.

Performance decreases gradually as the action-space prior is increasingly perturbed. This trend indicates that the spatial structure carried by the action draft is important for successful transfer, while the gradual degradation suggests that DemoPrior remains tolerant to moderate errors in the extracted draft. A similar trend is observed on the additional SimplerEnv tasks, where performance also decreases as the perturbation strength increases.

A.2 Structure-Preserving Integration

We vary the stopping time

$$\tau^* \in \{0, 0.3, 0.55, 0.75, 0.95\},$$

and additionally compare with open-loop execution. The value used in the main experiments is selected on seen training tasks and then fixed for unseen-task evaluation. Table 7 reports the results.

An intermediate stopping point performs best on both seen and unseen tasks. Integrating the flow to its endpoint ($\tau^* = 0$) can move the action away from the useful structure provided by the demonstration, whereas stopping too early leaves insufficient opportunity for the policy to correct the draft. The best performance at $\tau^* = 0.75$ supports the role of partial integration: it allows the pretrained policy to refine the action draft while preserving the skill structure transferred from the demonstration. Notably, the same stopping point selected on seen tasks also performs best on unseen tasks, suggesting that this integration behavior transfers beyond the tasks used for selection.

Table 6: LIBERO-90 unseen-task success rate (%) under Gaussian perturbations.

σ	0	0.1	0.2	0.3	0.4	0.5
Success (%)	**76.1**	72.2	67.2	60.0	52.2	39.4

Table 7: Effect of the stopping time τ^* on LIBERO-90. Success rates are reported in %.

τ^*	0	0.30	0.55	0.75	0.95	Open-loop
Seen	65.0	73.3	75.0	**76.2**	62.8	41.6
Unseen	46.1	57.2	68.9	**76.1**	67.2	57.7

B LIBERO Experimental Details

B.1 Task Splits and Evaluation

We use LIBERO-90 as the policy-training domain. The 90 tasks are shuffled with seed 0, after which 72 tasks are used for training and the remaining 18 tasks are held out for unseen-task evaluation. The training split contains 3,600 demonstrations in total. For seen-task evaluation, we randomly sample 18 tasks from the 72 training tasks.

We additionally evaluate on **LIBERO-40**, consisting of 40 tasks from LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, and LIBERO-10. These tasks do not overlap with LIBERO-90. No additional training is performed on LIBERO-40; we directly evaluate the policy trained on the 72-task LIBERO-90 split.

For demonstration-based methods, each evaluation task is paired with five independently selected demonstrations. Each demonstration is evaluated on 10 initial states (init 0–9), and we report the mean success rate across the five demonstrations.

B.2 Model and Input Configuration

Table 8 summarizes the model and input configuration used for LIBERO.

Table 8: Model and input configuration for LIBERO experiments.

Configuration	Value
Backbone	PyTorch π_0, initialized from π_0 base
Visual observations	Third-person + wrist cameras, 224×224
State	8D: position (3) + axis-angle (3) + finger joints (2)
Action	7D end-effector delta action
Action components	translation (3) + rotation (3) + gripper (1)
Internal action dimension	32
Action chunk length H	50
Normalization	Mean/std. from the 72 training tasks only

The policy receives two camera views together with an 8-dimensional state observation. The native LIBERO action consists of a 7-dimensional end-effector delta action, which is padded to the 32-dimensional action representation used internally by the model. For simulation, we use simulator-provided object IDs for segmentation in place of SAM2 on synthetic observations.

Table 9: Training configuration for DemoPrior on LIBERO.

Configuration	Value
Training steps	30,000
Batch size	32
Hardware	$2\times$ A100 80GB
Per-GPU batch size	8
Gradient accumulation	2
Optimizer	AdamW
Adam β_1, β_2	$0.9,\ 0.95$
Weight decay	10^{-10}
Gradient clipping	Global norm 1.0
Learning-rate schedule	Cosine
Warmup	1,000 steps
Peak learning rate	2.5×10^{-5}
Final learning rate	2.5×10^{-6}
Precision	bf16
EMA	None
Training seed	42
Draft noise σ_{train}	0.1
Gaussian-source samples	10%

B.3 Training Details

Table 9 gives the training configuration. All models trained on LIBERO use the same 72-task training split unless otherwise specified.

For each training sample, the action draft is constructed from a *different demonstration of the same task*, rather than from the expert trajectory used as the training target. Gaussian noise with $\sigma_{\text{train}} = 0.1$ is added to the action draft during training. For 10% of training samples, the draft is removed

and the flow instead starts from Gaussian noise, retaining the original generation capability of the pretrained VLA.

B.4 Inference Details

Table 10 summarizes the inference settings used throughout the LIBERO experiments.

Table 10: Inference configuration for LIBERO.

Configuration	Value
Flow integration steps	10
Stopping point τ^*	0.75
Partially integrated dimensions	translation (3) + gripper
Fully integrated dimensions	rotation (3)
Test-time draft noise	$\sigma = 0$
Execution before replanning	8 steps
LIBERO-90 episode limit	520
LIBERO-Spatial limit	220
LIBERO-Object limit	280
LIBERO-Goal limit	300
LIBERO-10 limit	520
Initial states per demonstration	10 (init 0–9)

At inference time, the translational and gripper dimensions use $\tau^* = 0.75$, while rotational dimensions are integrated through the full flow and generated by the VLA. No additional noise is added to the action draft. The policy executes eight actions before replanning, at which point the draft is realigned using the current end-effector position. All methods use the same episode horizon within each benchmark setting.

B.5 Baseline Configurations

Table 11 summarizes the baseline configurations. Whenever retraining is required, we use the same 72-task LIBERO-90 training split as DemoPrior.

Table 11: Baseline configurations for the LIBERO experiments.

Method	Configuration
π_0 (language-only)	Initialized from π_0 base and fine-tuned for 30K steps on the same 72-task LIBERO-90 split using the same training recipe, without demonstration input.
UniVLA	Trained for 30K steps on the same 72-task LIBERO-90 split following the official training recipe.
DemoDiffusion	Applied on top of the language-only π_0 using SDEdit-style inference and the same action draft as DemoPrior. We use $\tau = 0.5$ on LIBERO-90 and $\tau = 0.3$ on LIBERO-40.
Skill Prior (Open-loop)	Directly executes the demonstration-derived action draft without VLA refinement.

C SimplerEnv Experimental Details

For SimplerEnv (Li et al., 2025b), we use Bridge V2 and construct a 90/10 split over normalized task instructions using seed 0. Task families containing the four official evaluation tasks are excluded from the policy-training split. We additionally construct five evaluation tasks whose instructions do not match those in the policy-training data.

The DemoPrior training set contains 18,817 trajectories. The language-only π_0 baseline is trained on 49,746 trajectories.

D LEARNING FROM HUMAN DEMONSTRATIONS

We further evaluate whether DemoPrior can learn from human manipulation videos recorded in a different domain from policy execution. We use self-recorded human demonstrations and evaluate six tasks in simulation and one task on the real robot. For each task, the policy receives a single RGB human demonstration video together with the task instruction.

E BASELINE AND TRAINING CONFIGURATIONS

Table 12 summarizes the configurations used for the main baselines. For π_0, UniVLA, and DemoDiffusion, we follow the same task splits used by DemoPrior whenever retraining is required. The language-only π_0 baseline is fine-tuned on the corresponding LIBERO and Bridge training splits, while UniVLA follows its official training recipe on LIBERO and uses the released Bridge checkpoint on SimplerEnv. DemoDiffusion is built on the same language-only π_0 backbone and applies its SDEdit-style inference procedure at test time. The Skill Prior baseline directly executes the demonstration-derived action draft without policy refinement. We additionally include task-specific single-demonstration finetuning on LIBERO as a reference for direct adaptation to individual tasks.

Table 12: Baseline configurations used in the main experiments.

Method	LIBERO	SimplerEnv
Language-only π_0	π_0 base, same split, 30K steps	π_0 base, same Bridge split, 30K steps
UniVLA (Bu et al., 2025)	Official recipe on the same 72-task split, 30K steps	Released Bridge checkpoint
DemoDiffusion (Park et al., 2026)	Language-only π_0 with SDEdit-style inference	Language-only SimplerEnv π_0 with the same inference procedure
Skill Prior (Open-loop)	Direct open-loop execution	Direct open-loop execution

Table 13: Training data and optimization budgets for the policy variants.

Model	Training data	Steps	Evaluation
DemoPrior (LIBERO)	72 LIBERO-90 tasks	30K	LIBERO-90 and LIBERO-40
Language π_0 (LIBERO)	Same 72 tasks	30K	LIBERO baselines
DemoPrior (SimplerEnv)	Bridge 90% split	5K	SimplerEnv
Language π_0 (SimplerEnv)	Same Bridge split	30K	SimplerEnv baselines
Conditioning / residual variants	72 LIBERO-90 tasks	30K each	Ablation

F COMPARISON WITH LLM-IN-THE-LOOP CONTROL

F.1 PROTOCOL

We compare DemoPrior with two high-budget test-time reasoning baselines that augment the same language-only π_0 policy with Claude Opus 5.5 or GPT-6 Astra. The underlying π_0 policy is trained for 30K steps on the 72 LIBERO-90 training tasks.

At each policy-query step, the LLM receives the task instruction, third-person and wrist-camera observations, the robot state, and the 15×7 action chunk proposed by π_0. The interaction history is retained throughout the episode.

The LLM can either accept the proposed action chunk, modify a short prefix of the proposed actions, or temporarily take control through bounded end-effector commands. Control is returned to π_0 once the LLM accepts a subsequent policy proposal. The LLM has no access to trajectory previews, simulator states beyond the provided observations, task-definition files, or code-execution tools.

Episodes use a 520-step horizon at 20 Hz. We evaluate 10 held-out LIBERO-90 tasks using initial states 0–4, for a total of 50 episodes per method. Both LLMs use their highest available reasoning-effort setting.

F.2 CLAUDE OPUS 5.5 + π_0

Table 14: Performance and test-time reasoning cost of Claude Opus 5.5 + π_0 on held-out LIBERO-90 tasks. Decisions and interventions are reported per episode.

Task	Success	Decisions	Interventions	Time (min)
Butter → Tray	5/5	19.0	6.8	9.6
Tomato Sauce → Tray	5/5	36.4	19.6	21.3
Ketchup → Basket	2/5	72.4	38.0	37.6
Stack Bowls → Tray	0/5	73.6	40.6	40.1
Pudding → Plate-Left	5/5	34.8	19.8	15.8
Book → Lower Shelf	1/5	63.4	35.8	35.1
Open Top Drawer	5/5	6.6	0.6	3.1
Book → Back Compartment	5/5	27.6	9.8	19.0
Close Microwave	5/5	30.6	16.4	18.8
Pan → Cabinet Top	5/5	24.6	10.0	10.1
Average	38/50	38.9	19.7	21.1

Claude Opus 5.5 + π_0 succeeds in 38 of 50 episodes. However, it requires an average of 38.9 LLM decisions and 19.7 active interventions per episode, corresponding to 21.1 minutes of test-time reasoning on average. This comparison highlights the substantial inference-time interaction required by an LLM-in-the-loop controller.

F.3 GPT-6 ASTRA + π_0

Table 15: Per-episode averages for GPT-6 Astra + π_0.

Task	Success	Decisions	Modify + Takeover	Time (min)	Input (M)	Output (K)
Butter → Tray	5/5	20.6	6.6	10.0	1.87	15.3
Tomato Sauce → Tray	5/5	21.6	9.8	10.7	2.04	16.5
Ketchup → Basket	4/5	29.6	7.6	15.0	3.84	21.7
Stack Bowls → Tray	0/5	48.2	17.8	24.8	8.00	37.7
Pudding → Plate-Left	0/5	50.8	19.2	25.7	8.38	37.6
Book → Lower Shelf	0/5	55.2	25.4	29.1	10.07	44.1
Open Top Drawer	5/5	6.2	0.0	2.8	0.29	4.3
Book → Back Compartment	4/5	20.6	5.4	10.4	2.54	15.9
Close Microwave	1/5	58.8	32.0	30.7	11.04	45.0
Pan → Cabinet Top	5/5	26.2	6.6	11.6	2.74	18.1
Average	29/50	33.8	13.0	17.1	5.08	25.6

Across all 50 episodes, the Astra system makes 1,689 decisions, consumes approximately 254M input tokens and 1.28M output tokens, and spends an average of 17.1 minutes of LLM reasoning per episode.

F.4 SUMMARY AND EVALUATION DISCLOSURE

Table 16: Summary of the LLM-in-the-loop comparison.

	DemoPrior	Opus 5.5 +π_0	Astra +π_0
Success	36/50	**38/50**	29/50
LLM calls / episode	–	38.9	33.8
Input tokens / episode	0	12.51M	5.08M
Output tokens / episode	0	113.6K	25.6K
LLM time / episode	0	21.1 min	17.1 min
Modify/takeover ratio	–	50.7%	38.6%
Affected-step ratio	–	33.2%	18.8%

G Real-World Experimental Setup

G.1 Robot Platform

Experiments are conducted on a Universal Robots UR5 (CB3) running PolyScope 3.15.8 with a Gimatic KIT-UR-J two-finger parallel gripper. The robot is controlled through RTDE using `servoJ` at 30 Hz.

We use three third-person RGB cameras: a top-view Intel RealSense D435i, a side-view RealSense D435, and a front USB camera, all recording at 640×480 and 30 fps. No wrist-mounted camera is used. Depth from the top and side RealSense cameras is used only for skill-prior construction and is not provided to the policy.

Robot demonstrations are collected using an SO-101 leader arm and retargeted to the UR5.

G.2 Data and Task Split

Robot demonstrations are collected by teleoperation and stored in LeRobot v2.0 format. After trimming, the dataset contains 107,881 frames at 30 fps, corresponding to approximately 60 minutes of interaction.

The policy input consists of two 224×224 RGB views (top and side) and a 7-dimensional robot state containing six joint angles and the previous gripper command. The action consists of six absolute joint targets and a binary gripper command.

Table 17: Real-world evaluation tasks.

Task	Manipulation	Split
cube_bowl	Black cube → bowl	Seen
banana_top	Banana → drawer top	Seen
banana_in	Banana → top drawer	Seen
marker_plate	Marker → plate	Seen
cube_drawer	Black cube → drawer	Unseen
banana_plate	Banana → plate	Unseen
cube_plate	Black cube → plate	Unseen

G.3 Policy Training

The policy is initialized from the pretrained π_0 base model and trained for 15K steps using bf16 precision on four A100-80GB GPUs. The global batch size is 32. We use fused AdamW with a cosine learning-rate schedule, 1K warmup steps, a peak learning rate of 2.5×10^{-5}, and a final learning rate of 2.5×10^{-6}. The 15K-step checkpoint is used for evaluation.

The policy predicts action chunks of length 50. Actions use the backbone's 32-dimensional action representation, with unused dimensions padded. Training prior noise is set to $\sigma = 0.1$.

Table 18: Real-world training configuration.

Setting	Value
Initialization	π_0-Base (PyTorch)
Hardware	$4\times$ A100-80GB
Precision	bf16
Global batch size	32
Training steps	15K
Optimizer	AdamW (fused)
LR schedule	cosine
Warmup	1K steps
Peak / final LR	2.5×10^{-5} / 2.5×10^{-6}
Action chunk length	50
Training prior noise	$\sigma = 0.1$

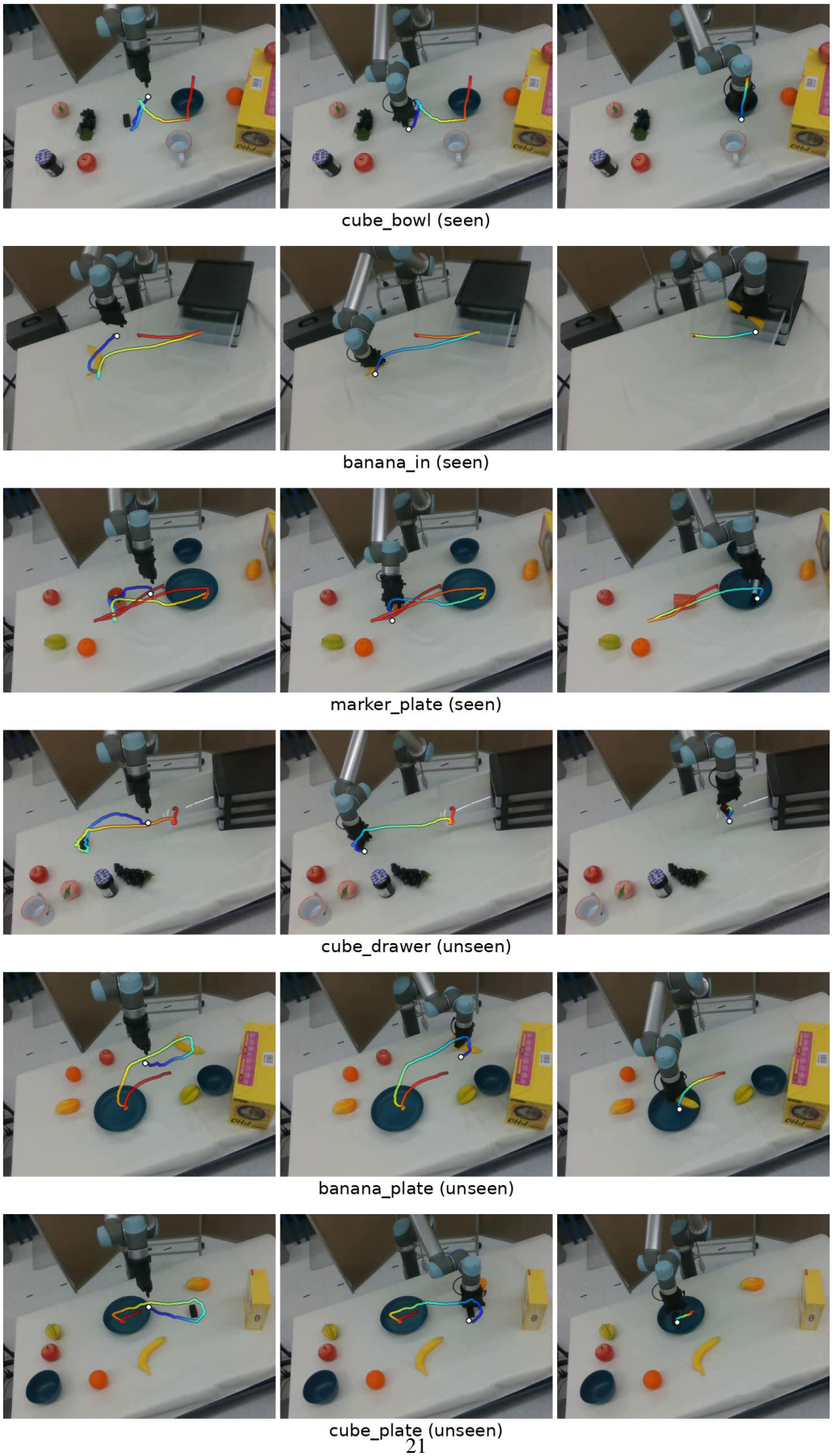

Figure 6: Real-world UR5 setup and representative manipulation tasks.

G.4 Skill-Prior Construction and Execution

SAM2 (Ravi et al., 2025) and SigLIP2 (Tschannen et al., 2025) are used to establish task-relevant correspondences between the demonstration and the current scene. CoTracker (Karaev et al., 2025) is used for temporal motion tracking

The real-world execution settings are:

$$\tau^* = 0.7, \qquad \sigma_{\text{test}} = 0.$$

Flow integration uses 10 Euler steps. The policy generates an action chunk at each query and executes the first eight actions before replanning. Robot control runs at 30 Hz, with a maximum joint-command change of 3° per control step. No task-specific policy update is performed during evaluation.

H DemoDiffusion Hyperparameter Selection

DemoDiffusion (Park et al., 2026) initializes denoising from

$$\mathbf{x}_\tau = (1-\tau)\mathbf{p} + \tau\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}).$$

Here, smaller τ retains more of the transferred prior, while $\tau = 1$ corresponds to initialization from pure noise.

H.1 Simulation

Table 19: DemoDiffusion success rate (%) under different SDEdit noise levels τ in simulation. Bold indicates the selected setting.

Setting	$\tau = 0.3$	$\tau = 0.5$	$\tau = 0.7$	$\tau = 1$
LIBERO-90 Unseen	21.1	**27.6**	5.0	26.7
LIBERO-90 Seen	25.0	**68.1**	18.9	85.0
LIBERO-40	**30.6**	5.6	1.0	2.8
SimplerEnv Official	**46.9**	29.2	41.7	32.3

H.2 Real-Robot Tuning

We additionally evaluate DemoDiffusion sensitivity to τ on the UR5 setup. Only DemoDiffusion results are reported here in order to isolate the effect of the SDEdit noise level.

The real-robot setting exhibits substantially different sensitivity from the simulation benchmarks. The $\tau = 0.5$ setting fails on all four tested seen tasks, whereas $\tau = 0.9$ reaches 35.0% aggregate success. We therefore use $\tau = 0.9$ for the real-robot DemoDiffusion baseline. The selected DemoDiffusion τ is reported explicitly for each evaluation domain because its effective value varies substantially across settings.