TRAINING-FREE ADAPTATION OF GENERALIST ROBOT POLICIES VIA DISTRIBUTIONAL POLICY STEERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Pretrained vision-language-action (VLA) policies possess broad motor skills but can struggle to apply them under semantic and spatial distribution shifts. Vision-language models (VLMs) offer complementary reasoning capabilities for out-of-distribution (OOD) inference, given its strong capability on semantic understanding and commonsense knowledge. However, how to formalize and integrate VLM reasoning into visuomotor action sampling while leveraging pretrained motor skills remains an open challenge. We introduce Distributional Policy Steering (DPoS), a training-free, inference-time steering method that provides distributional alignment guidance from VLM-derived spatial information to a frozen VLA policy. DPoS constructs a shell-supported target distribution around a reasoned spatial estimate and jointly guides sampled action chunks toward it. An energy function is designed based on the squared maximum mean discrepancy measures between the sampled action candidates cloud and the target distribution. DPoS enables the repulsion among sampled candidates, aiming for a better coverage among the action space. The attraction to the target distribution aims for a reasoned spatial target alignment for the task success. Evaluated on LIBERO-PRO simulator and 9 real-world OOD challenging tasks, DPoS achieves the highest success rate among all leading baselines. It improves over the base model $\pi_{0.5}$ by **44.4** and **19.9** percentage points on real-world and LIBERO-PRO tasks respectively, demonstrating distributional guidance as an effective way to integrate reasoning prior into pretrained robot control without additional model training.

1 INTRODUCTION

Pre-trained vision-language-action (VLA) policies (Black et al., 2024; Intelligence et al., 2025) can acquire the motor skills needed to complete a task, yet often fail under distribution shifts involving object identity, spatial or semantic layouts, or language instructions (Zhou et al., 2025). Such failures can arise from various sources, including perception errors (Dey et al., 2025), incorrect semantic grounding (Fang et al., 2026), and limitations in physical control (Parés-Morlans et al., 2026). A common adaptation strategy is to fine-tune the policy on domain-specific demonstrations, which is time-consuming and labor-costly (Kim et al., 2025). How to efficiently adapt the pre-trained VLA policy for OOD tasks with no finetuning or verifier training, while preserving the board motor skills from the pre-training remains a problem to solve.

External vision-language models (VLMs) provide a natural source of semantic and spatial guidance for end-to-end robot planning (Feng et al., 2025). Recent methods incorporate such guidance into robot execution through task decomposition (Liu et al., 2026; Li et al., 2025), generated objectives (Nakamoto et al., 2024; Wu et al., 2025; Liu et al., 2024), and action sampling guidance (Song et al., 2026; Tang et al., 2026; Kao et al., 2026; Wagenmaker et al., 2025; Qu et al., 2025). For example, VLS uses a VLM to determine different planning stages and synthesize differentiable trajectory rewards (Liu et al., 2026), while OmniGuide (Song et al., 2026) constructs spatial energy fields around a VLM-pointed target. These results indicate that a reasoning model can help improve a policy's inference time performance on certain OOD tasks. Inspired by this finding, *we investigate how reasoning outputs can be represented to guide OOD task execution while leveraging the policy's learned motor skills.*

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

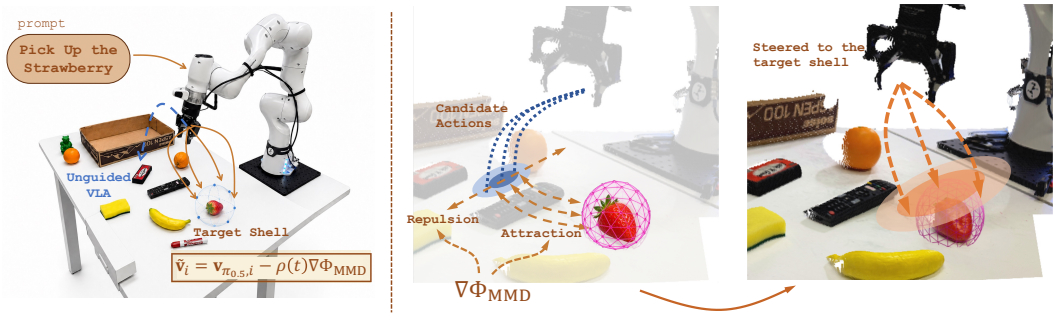


Figure 1: **Distributional Policy Steering (DPoS) guides a frozen VLA toward a reasoned spatial target with board action sampling coverage and task-specific target distribution attraction.** VLA fails to handle out-of-distribution settings. DPoS constructs a target shell around the VLM-localized object and applies Shell-MMD guidance, whose attraction term moves candidate samples toward the shell while its sample-wise repulsion term spreads candidates apart. The resulting action candidates approach the intended object from diverse directions, allowing the policy to reuse its pretrained motor skills without fine-tuning.

We propose DPoS, a training-free, inference-time guidance method that project point-wise VLM reasoning preference to a set of task-space targets. A key insight underlying our approach is that reasoning should express the spatial preference as an estimate of an optimal solution during the guidance but not as a unique endpoint. We study whether a neighborhood around the estimated target can retain near-optimal task performance theoretically and experimentally. Proposition 1 shows that, under mild regularity and estimation-quality assumptions, a non-trivial neighborhood of the reasoning estimate admits a bounded task cost. This finding provides a theoretical basis for extending point-wise reasoning target into a distribution of valid targets for steering.

Specifically, DPoS constructs a shell-supported target distribution around a reasoned spatial estimate and guide the sampled action chunk candidates cloud toward this distribution. DPoS defines the guidance energy based on the maximum mean discrepancy (MMD) (Gretton et al., 2012) between the action-sample distribution and the target-shell distribution. The derived gradient naturally combines attraction to the target distribution and pairwise replusion without requiring a separately manually added diversity objective. The target distribution term encourages alignment with the task-specific target support. Pairwise sample repulsion term discourages concentration among candidates and promotes coverage of alternative approaches. Maintaining sampling coverage helps avoid collapse among similar wrong action proposals (Corso et al., 2024). As illustrated qualitatively in Fig. 7, multiple candidates can otherwise follow the same incorrect direction, limiting the alternatives available for execution.

We evaluate DPoS in both simulated and real-world robotic settings and both acheived the highest success rate among strong leading baselines. Across 9 challenging real-world OOD tasks on Droid (Khazatsky et al., 2024) setup, DPoS achieves a success rate of **71.11%**, exceeding the strongest baseline, OmniGuide, by **36.67** percentage points and the frozen $\pi_{0.5}$ policy by **44.44** percentage points. On LIBERO-PRO, DPoS (VLM) achieves **51.1%** success, exceeding the strongest baseline, Point Attraction (45.1%), by approximately 6 percentage points and $\pi_{0.5}$ (31.2%) by 19.9 percentage points. We conduct ablation studies on the attraction-repulsion balance, target-shell radius, and guidance strength to assess their contributions to task performance and the method's sensitivity to these design choices. We also qualititively show that DPoS steer the frozen VLA but still sample actions close to the pre-trained policy distribution, well utilizing and preserving the pre-trained motor skills.

2 Related Work

Generalist Robot Policies and Vision-Language-Action Models. Large-scale imitation learning has shifted robot learning from task-specific policies toward generalist models trained across tasks, scenes, and embodiments. Early scaling efforts such as Brohan et al. (2022), Brohan et al. (2023), Kim et al. (2024) demonstrated that transformer policies benefit from diverse real-robot

data and made large-scale pretraining and downstream adaptation broadly accessible, establishing the paradigm of Vision Language Action (VLA) model.

In parallel, generative action models have become a strong interface for multimodal robot behavior. Chi et al. (2025) models visuomotor control as conditional denoising and combines action chunking with receding-horizon execution. Building on this formulation, Black et al. (2024) and Intelligence et al. (2025) couple a pretrained VLM to a flow-matching action expert, co-training across robot data, web data, object detections, and high-level semantic prediction (Team et al., 2024; Collaboration et al., 2023). These models provide broad semantic and motor priors, but their behavior is still shaped by the support and correlations of the demonstration distribution (De Haan et al., 2019; Fang et al., 2026). Under novel object appearances, layouts, spatial relations, or instructions, the requisite skill can be present in the model yet not be reliably elicited. DPoS leaves the generalist backbone frozen and treats OOD generalization as a problem of controlling its action distribution at inference time.

Inference-Time Steering of Generative Policies. Diffusion guidance modifies pretrained generative sampling without retraining (Ho & Salimans, 2022; Dhariwal & Nichol, 2021), while particle guidance introduces interactions among samples to promote diversity (Corso et al., 2024). Robotics methods adapt these ideas at different stages of action generation. Value-Guided Policy Steering (Nakamoto et al., 2024) and FOREWARN (Wu et al., 2025) select among candidate actions using value estimates or predictive evaluation. Other approaches guide intermediate denoising states through human feedback or learned dynamics (Wang et al., 2025; Du & Song, 2026), or modify trajectory initialization and latent noise to influence policy outputs (Yoneda et al., 2023; Park et al., 2025; Wagenmaker et al., 2025). General Policy Composition (Cao et al., 2026) instead combines multiple pretrained policy scores. DPoS jointly steers samples from a single frozen policy by matching the action candidates distribution to a reasoning-conditioned target distribution.

Recent methods instantiate distinct forms of inference-time steering. VLS (Liu et al., 2026) performs *reward-guided sampling*, using VLM-derived gradients together with repulsion and resampling, while OmniGuide (Song et al., 2026) applies *geometric energy guidance* through differentiable kinematics. Both primarily guide individual trajectory proposals. Flow Reversal Steering (Tang et al., 2026) instead encodes guidance into the *latent initialization* by reversing a coarse action into flow noise, whereas Flow Control (Kao et al., 2026) directly modifies the flow initial condition with user inputs. In contrast, our method does not treat task attraction and sample diversity as separate mechanisms. DPoS jointly guides particles toward a distribution of VLM-estimated targets while preserving action sampling coverage, leaving fine-grained manipulation to the pretrained VLA.

3 Near-Optimal Target Regions

3.1 Problem Formulation

We consider inference-time guidance of a pretrained vision-language-action (VLA) policy with a flow-matching action head $\pi_\theta(A \mid o, \ell)$, where o denotes the sensor observation, ℓ is a natural-language instruction, and $A \in \mathbb{R}^{H \times D}$ is an action chunk with horizon H and action dimension D. We target test-time out-of-distribution (OOD) generalization, where

$$\mathrm{supp}(\mathcal{D}_{\mathrm{test}}) \nsubseteq \mathrm{supp}(\mathcal{D}_{\mathrm{train}}).$$

In particular, $(o, \ell) \sim \mathcal{D}_{\mathrm{test}}$ may contain novel target objects or appearances, unseen object poses and spatial relations, or language instructions requiring compositional scene and spatial reasoning that were absent or rare in $\mathcal{D}_{\mathrm{train}}$. Our method does not access the original training data or update the parameters θ.

We focus on tasks in which an external target estimate can help the policy approach the instructed object or region. Given (o, ℓ), π_θ needs to approach a task-relevant object. At its start, π_θ maps (o, ℓ) to a target point $g_w \in \mathbb{R}^3$ in world frame as $F(o, \ell) = g_w$. The target provides the directional guidance; it need not specify a grasp contact, a unique end-effector endpoint, or a complete action trajectory. Our objective is to use this information to improve task execution while making the most of the pretrained policy's action prior.

3.2 NONUNIQUENESS OF VLM TARGET POINT

We treat VLM reasoning output as a directional guidance. A VLM prediction is an estimate of a useful spatial target, rather than necessarily the unique optimal target. We characterize a non-singleton solution region around this estimate under which sampled targets can retain comparable task quality.

Let $x^\star$ denote an optimal target for a downstream robotic task, with task cost $J : \mathcal{X} \to \mathbb{R}$:

$$x^\star \in \arg\min_{x \in \mathcal{X}} J(x), \qquad J^\star = J(x^\star). \tag{1}$$

In practice, a VLM does not necessarily recover the exact optimum. Instead, we model its prediction as an estimate of the optimal target:

$$\hat{x} = x^\star + e, \tag{2}$$

where e denotes the estimation error of the VLM.

Rather than treating $\hat{x}$ as a unique target, we consider the set of solutions that are *approximately optimal*. For a tolerance $\epsilon > 0$, define the ϵ-optimal solution set as

$$\mathcal{S}_\epsilon = \{x \in \mathcal{X} \mid J(x) \le J(x^\star) + \epsilon\}. \tag{3}$$

We show that, under mild smoothness assumptions on the task cost, the VLM estimate induces a non-trivial neighborhood of near-optimal solutions.

Proposition 1 (A neighborhood of near-optimal targets)**.** *Suppose J is L-Lipschitz on $\mathcal{X}$ with respect to the ℓ_2 norm,*

$$|J(x_1) - J(x_2)| \le L\|x_1 - x_2\|_2, \qquad x_1, x_2 \in \mathcal{X}. \tag{4}$$

Suppose further that the estimated target is δ_{est}-suboptimal,

$$J(\hat{x}) \le J^\star + \delta_{\text{est}}, \qquad \delta_{\text{est}} \ge 0. \tag{5}$$

Then, for every $\varepsilon > \delta_{\text{est}}$,

$$\overline{B}_2\left(\hat{x}, \frac{\varepsilon - \delta_{\text{est}}}{L}\right) \cap \mathcal{X} \subseteq \mathcal{S}_\varepsilon, \tag{6}$$

where $\overline{B}_2(\hat{x}, r)$ is the closed Euclidean ball of radius r centered at $\hat{x}$. If $\hat{x}$ lies in the interior of $\mathcal{X}$, this inclusion contains a positive-radius neighborhood and hence infinitely many ε-optimal targets.

The proof of this proposition and can be found in Supplementary B. We also show a special case under task symmetry where the solution region contains an exact manifold of equivalent optima in Supplementary C.

Empirical motivation. Our simulation and real-world evaluations provide evidence consistent with retaining multiple spatial targets during guidance: single-target attraction trails DPoS (VLM) by approximately 6 percentage points on LIBERO-PRO and 42.22 percentage points in real-world trials. Qualitatively, single-target guidance exhibits abrupt corrections during the approach-to-grasp transition, with particularly visible consequences on the real robot, including unintended tabletop contact and failed grasps. These observations motivate retaining spatial alternatives during guidance. Detailed comparisons and failure analyses are presented in Fig. 2 and Fig. 5.

4 SHELL-MMD GUIDANCE

DPoS uses the external target to construct a distribution of task-space candidates and steers multiple policy samples towards that distribution. At every re-plan, we draw K policy samples to form denoised chunks $\{\hat{A}_0^{(i)}\}_{i=1}^K$, and map each chunk to a task-space displacement $y_i \in \mathbb{R}^{d_g}$, where d_g is the guided task-space dimension. We then construct a target distribution $\nu_R(g)$ around the current goal displacement $g \in \mathbb{R}^{d_g}$, guide the joint sampler by a designed shell-MMD energy function, enabling both replusion between particles for sampling diversity and distributional target attraction for a task-constrained exploration.

4.1 REASONING THROUGH A PRE-TRAINED PRIOR

At the start of a reach stage, DPoS leverages the VLM's pre-trained semantics prior to reason through the OOD targets. It points to the target object on the agent-view camera's observation as pixels, and then be projected to the world frame as $F(o, \ell) = g_w$, and $g_w \in \mathbb{R}^3$, given the depth image and known camera intrinsics and extrinsics.

The guidance target is the displacement from the current end-effector position p_{ee} to the goal as:

$$g = (g_w - p_{\text{ee}}) \in \mathbb{R}^3. \tag{7}$$

$F(o, \ell)$ is queried once per reach stage, while the same target g_w is re-expressed relative to the current p_{ee} at every re-plan. Let $T_\tau : \mathbb{R}^3 \to \mathbb{R}^3$ maps the translational component of policy action $\widehat{A}_0[\tau, 1:3]$ into the measured displacement in world frame at step τ as:

$$y(\widehat{A}_0) = \sum_{\tau=1}^{H_g} T_\tau \Big(\widehat{A}_0[\tau, 1{:}3] \Big) \in \mathbb{R}^{d_g}, \tag{8}$$

where $H_g \leq H$ is the guidance horizon and $y(\widehat{A}_0)$ is the predicted end-effector displacement from the denoised action chunk.

4.2 SHELL-MMD OBJECTIVE

DPoS constructs M shell support points as $q_m = g + Rd_m$, where d_m is a unit direction vector. Each point lies at distance R from the goal displacement g. The corresponding empirical target distribution is

$$q_m = g + Rd_m, \qquad \nu_R(g) = \frac{1}{M} \sum_{m=1}^{M} \delta_{q_m}. \tag{9}$$

In order to have a probabilistic measure between the sampled particles cloud δ_{y_i} and the shell δ_{q_m}, we construct an energy function aiming for both shell attraction and sample-wise repulsion. It moves the predicted samples toward the shell but also preserves the samples' action space coverage. More details about the Maximum and Mean Discrepancy and kernels are in Supplementary A.2. For K samples, the shell-MMD's energy is

$$\Phi_{\text{shell}}(Y; g) := K \ \text{MMD}^2_{\kappa_h} \left(\frac{1}{K} \sum_{i=1}^{K} \delta_{y_i}, \nu_R(g) \right) \tag{10}$$

$$= \frac{1}{K} \sum_{i=1}^{K} \sum_{j=1}^{K} \kappa_h(y_i, y_j) - \frac{2}{M} \sum_{i=1}^{K} \sum_{m=1}^{M} \kappa_h(y_i, q_m) + C, \tag{11}$$

where C is independent of samples, κ_h denotes to the similarity kernel. (Details see Supplementary D). We thus have $\phi(Y; g)$ as the derivative of this energy function with respect to each particle's displacement y_i that has both attractions to the target distribution and sample-wise replusion for sampling coverage.

$$\phi(Y; g) = -\nabla_{y_i} \Phi_{\text{shell}} = \underbrace{\frac{4}{Mh^2} \sum_{m=1}^{M} \kappa_h^2(y_i, q_m)(q_m - y_i)}_{\text{shell attraction}} + \underbrace{\frac{4}{Kh^2} \sum_{\substack{j=1 \\ j \neq i}}^{K} \kappa_h^2(y_i, y_j)(y_i - y_j)}_{\text{sample-wise repulsion}}. \tag{12}$$

In our implementation, we normalize the attraction and repulsion gradients separately for each particle before weighting them. Let ϕ_i^{att} and ϕ_i^{rep} denote the action-chunk gradients induced by the two terms in Eq. 12. The implemented direction is

$$u_i^c = \frac{\phi_i^c}{\|\phi_i^c\|_2 + \epsilon}, \quad \bar{\phi}_i = \frac{w_{\text{att}} u_i^{\text{att}} + w_{\text{rep}} u_i^{\text{rep}}}{\|w_{\text{att}} u_i^{\text{att}} + w_{\text{rep}} u_i^{\text{rep}}\|_2 + \epsilon}, \quad c \in \{\text{att}, \text{rep}\}. \tag{13}$$

Separate normalization prevents either term's raw scale from dominating, so the ratio $w_{\text{att}}{:}w_{\text{rep}}$ directly controls the attraction–diversity trade-off.

Table 1: **Quantitative simulation results on LIBERO-PRO.** DPoS improves task success across six suites with position and task shifts. Overall success is the unweighted mean of the six suite success rates. The VLM-guided variant reaches 51.1% overall success, surpassing all listed baselines; oracle 3D targets further increase success to 54.6%.

Method	object swap	object task	spatial swap	spatial task	goal swap	goal task	all six suites
$\pi_{0.5}$	17.8	10.8	49.0	53.2	35.0	21.6	31.2
OmniGuide	21.0	12.4	49.8	54.4	36.8	24.6	33.2
VLS	45.0	41.0	42.0	54.0	38.0	33.6	42.3
Best-of-N	10.0	7.6	51.0	54.6	34.6	22.2	30.0
Point Attraction	41.6	42.6	44.0	78.6	35.2	28.6	45.1
DPoS (VLM)	58.4	47.2	**52.4**	76.2	41.0	31.6	51.1
DPoS (Oracle3d)	**59.2**	**48.8**	51.4	**92.6**	**41.3**	**34.6**	**54.6**

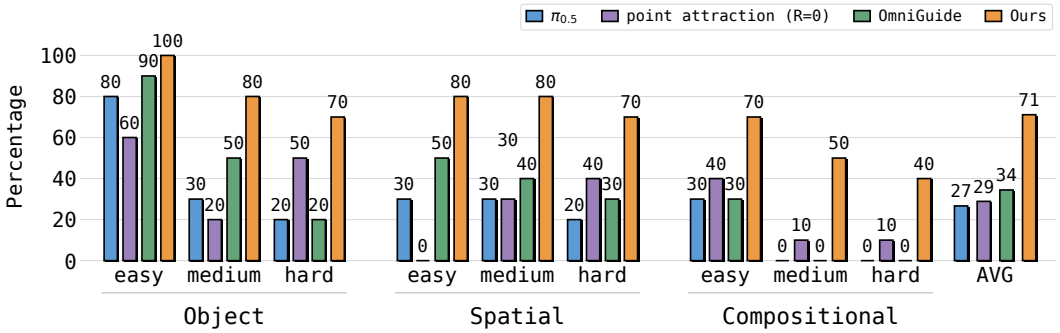


Figure 2: **Shell guidance substantially improves real-world task success.** Across the Object, Spatial, and Compositional suites, DPoS outperforms both the frozen $\pi_{0.5}$ policy and single-point attraction on all nine tasks, achieving 71.1% average success.

4.3 Shell-MMD guidance of flow policies

Guidance alters the noise forward integration without retraining. Given $\bar{\phi}(Y; g)$, and the velocity field from pre-training, the energy-guided forward integration step via Euler method is:

$$x_{t-\delta}^{(i)} = x_t^{(i)} - \delta v_\theta^{(i)} + \delta\rho\bar{\phi}(Y; g), \tag{14}$$

where $\delta > 0$ denotes to the numerical integration step size and ρ is guidance strength. Larger ρ apply stronger shell guidance, while $\rho = 0$ recovers the original policy. We ablate the guidance strength ρ and the attraction–repulsion ratio w_{att}:w_{rep} in Sec. 5.3. More Flow head definition details are in Supplementary A.1. **Particle selection and execution.** Shell-MMD shapes the joint particle distribution but does not determine which action chunk is executed. After the final denoising step, we score each particle by the directional alignment between its predicted displacement and the current goal displacement g as cosine similarity between y_i and g:

$$s_i = \frac{y_i^\top g}{(\|y_i\|_2 + \epsilon)(\|g\|_2 + \epsilon)}, \qquad i^\star = \underset{i\in\{1,\dots,K\}}{\arg\max}\ s_i, \tag{15}$$

where $\epsilon > 0$ provides numerical stability. The selected chunk $\hat{A}_0^{(i^\star)}$ is executed for the first n_{exec} control steps before the policy replans.

5 Experiments

We evaluate DPoS in closed-loop simulation and real-world robot experiments, addressing four quesitons: (1) Can DPoS generalize better to OOD task with a pre-trained frozen VLA? (2) Do the improvements transfer to physical manipulation under real-world perception and calibration? (3) Does DPoS adhere to the pre-trained action prior? (4) Which components of DPoS drive its performance gains, and how sensitive is the method to key design choices?

5.1 Closed-loop Simulator Evaluation

We evaluate DPoS on LIBERO-PRO (Zhou et al., 2025; Liu et al., 2023) using the frozen official $\pi_{0.5}$-LIBERO checkpoint. We test Position (`swap`) and Task (`task`) shifts across LIBERO-Object,



Figure 3: **Real-world tasks probe increasingly complex target grounding.** We propose three real-world task suites to evaluate generalization across object identity, spatial relations, and compositional instructions. The Object suite tests recognition of unfamiliar named objects. The Spatial suite requires selecting targets by relative location. The Compositional suite requires interpreting indirect references involving multiple objects. Rows progress from easy to hard through more cluttered scenes and more complex instructions. Full prompts are provided in Sec. F.

LIBERO-Spatial, and LIBERO-Goal, yielding six suites. The main comparison uses 50 rollouts per task (500 per suite), while ablations use 20 per task (200 per suite). Detailed evaluation and execution settings are provided in Appendix E.

Baselines. We compare against frozen $\pi_{0.5}$ with default sampling, VLS (Liu et al., 2026) with VLM-generated stage rewards, particle repulsion, and resampling, and OmniGuide (Song et al., 2026) with geometric energy guidance. We also evaluate *Best-of-N*, which selects the closest-to-target endpoint among N independently sampled action chunks without guidance, and *Single-Point Target*, which retains our K-particle guided sampler but sets $R = 0$.

Our Implementation. We evaluate two variants of DPoS to separate the effect of shell-MMD steering from perception errors. *DPoS (Oracle3D)*, directly obtains the privileged target 3D information from the simulator, providing DPoS's performance upper bound. *DPoS (VLM)*, a deployable version, uses Qwen3-VL-32B to provide semantic and spatial reasoning and identify the instructed object in the agent-view image. The predicted 2D point is lifted into the world frame using the rendered depth map and calibrated camera geometry.

Quantitative Results. In Table 1, the frozen $\pi_{0.5}$ policy achieves 31.2% overall success under distribution shifts, compared with 33.2% for OmniGuide and 42.3% for the published VLS result. Our deployable variant, *DPoS (VLM)*, achieves 51.1%, improving over $\pi_{0.5}$ by 19.9 percentage points. Its largest gains over $\pi_{0.5}$ occur on Object-Swap (40.6 percentage points) and Object-Task (36.4 percentage points), supporting effective steering under object displacement and novel task–target combinations. Best-of-N and Point Attraction achieve 30.0% and 45.1%, respectively. These comparisons support joint guidance during action sampling over post-hoc candidate selection, and a shell-supported target distribution over a single-point target. **Oracle Target Analysis.** Using simulator-provided 3D targets yields 54.6% overall success, compared with 51.1% for VLM-estimated targets, a gain of approximately 3.5 percentage points. This gap suggests room for further improvement through more accurate spatial grounding.

5.2 Real-world Experiments

We evaluate DPoS on a Franka Research3 using the official DROID (Khazatsky et al., 2024) setup. We devise nine tasks span Object, Spatial, and Compositional suites, each with easy, medium, and hard levels, as shown in Fig. 3. Each method is evaluated over 10 trials per task (90 total), with success requiring completion of the full instruction. Hardware, control settings, and task construction are detailed in Appendix E; task prompts and rollout examples appear in Appendices F and G.

Quantitative Results. As shown in Fig. 2, DPoS achieves 71.1% average success, compared with 26.7% for $\pi_{0.5}$, 28.9% for Point Attraction, and 34.4% for OmniGuide. Across the Object, Spatial, and Compositional suites, DPoS reaches 83.3%, 76.7%, and 53.3% success, respectively. Although

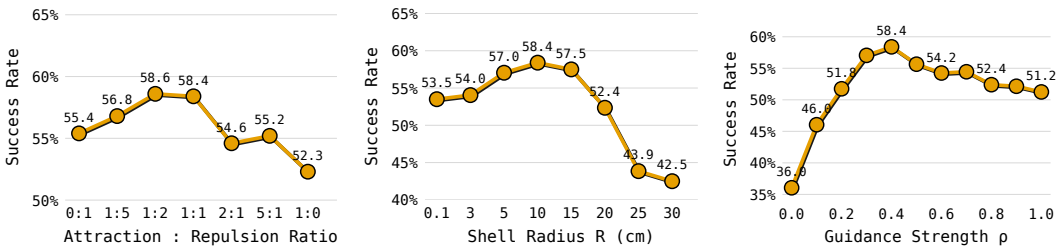


Figure 4: **Balanced shell guidance improves task success.** (a) Attraction and repulsion play complementary roles. (b) A moderate shell radius preserves useful approach flexibility. (c) Excessive guidance can interfere with pretrained manipulation behavior.

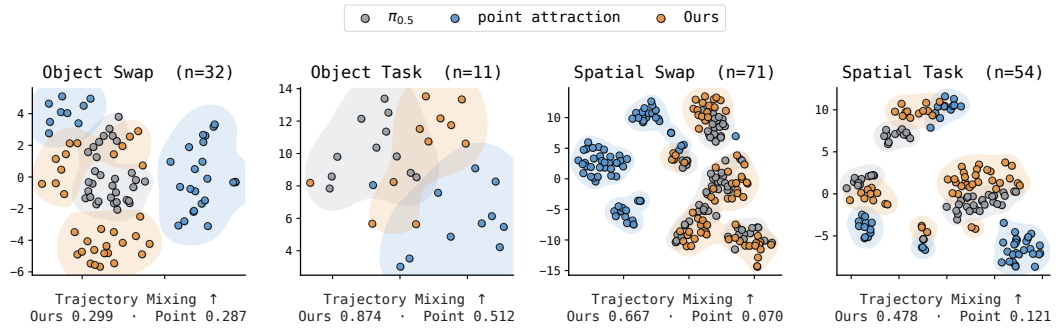


Figure 5: **Shell guidance better adheres to the pretrained action distribution.** We visualize trajectories from successful episodes across all three methods using per-suite t-SNE projections and 10- Nearest Neighbors. Across all four suites, trajectories generated by DPoS exhibit greater visual overlap with those of the pretrained $\pi_{0.5}$ policy than trajectories generated by single-point attraction. This pattern suggests that our steered action chunk is close to the pre-trained action distribution, persevering and utilizing the pre-trained motor skills.

absolute performance decreases as target grounding becomes more demanding, the advantage over the frozen policy and Point Attraction persists across all three suites. In contrast, Single-point Attraction improves overall success only marginally over the frozen policy, suggesting that direct attraction toward the target location is insufficient for reliable manipulation; guidance must also preserve the policy's learned approach and grasp behavior.

5.3 Ablation and Analysis

We examine how the design choices in DPoS affect task success and manipulation behavior. We ablate the attraction-repulsion balance, shell radius, and guidance strength (Fig. 4), and compare shell guidance with single-point attraction through quantitative evaluation and qualitative analysis in both simulation and real-world experiments.

Steering improves candidate quality beyond best-of-N selection. Best-of-N can only select among independently sampled actions, leading to mode collapse. As is shown in Figure 7, the eight independently sampled candidates collapse to wrong action mode and move away from the target, whereas DPoS produces action with better action space coverage.

Attraction vs repulsion trade-off. In Fig. 4(a), the plotted success rate reaches 58.6% at $w_{\text{att}}{:}w_{\text{rep}} = 1{:}2$, compared with 55.4% for repulsion alone and 52.3% for attraction alone. These results support combining both components: attraction encourages alignment with the spatial target, while repulsion promotes diversity among candidate approaches. The strongest performance occurs with moderately greater weight on repulsion, rather than either component alone.

A moderate shell radius improves task success. Figure 4(b) shows that an appropriate shell radius improves task success. Increasing R from $0.1\,\text{cm}$ to $10\,\text{cm}$ raises success from 53.5% to 58.4%, whereas further enlarging it to $30\,\text{cm}$ reduces success to 42.5%. These results suggest that a moder-

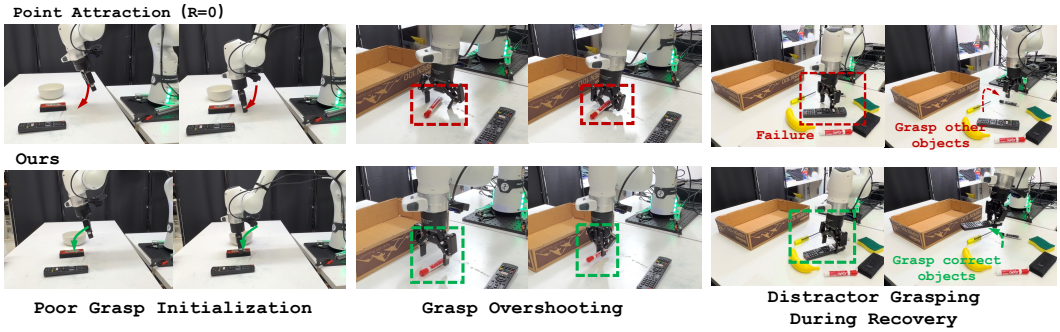


Figure 6: **Shell guidance avoids common failures of single-point attraction.** Point attraction can induce poor grasp initialization, overshoot the grasp target, or redirect recovery toward a distractor, whereas DPoS maintains a success approach to the intended object.

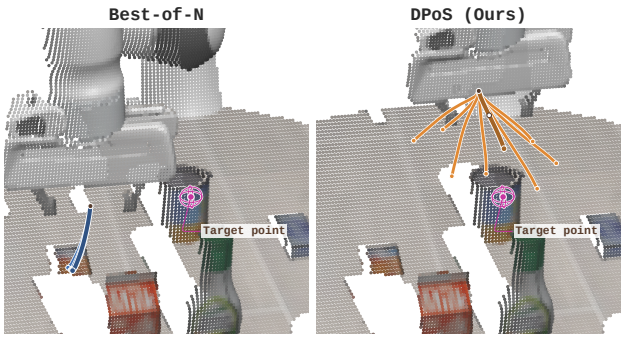


Figure 7: **Shell guidance produces diverse, target-directed action candidates.** Best-of-N candidates cluster tightly and move away from the target, whereas DPoS candidates approach it along diverse paths.

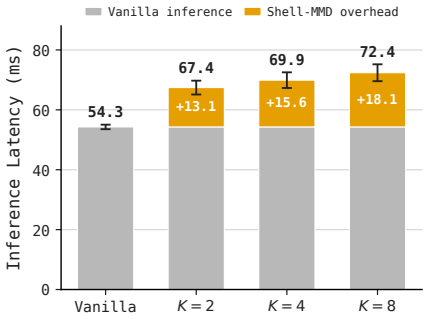


Figure 8: **Batched Shell-MMD adds modest latency.** Mean ± standard deviation over 100 warmed calls; orange denotes overhead over vanilla.

ate radius provides useful spatial flexibility, whereas an excessively large shell places target support too far from the estimated target.

DPoS samples actions close to the pre-trained action prior. Figure 5 compares trajectories from matched successful episodes using per-suite t-SNE visualizations and *Trajectory Mixing*, a local label-mixing metric computed in the original standardized trajectory space. Across all four tasks, DPoS exhibits greater mixing with frozen $\pi_{0.5}$ trajectories than single-point attraction, suggesting closer alignment with pretrained behavior among successful rollouts. Details see Supplementart H.

Stronger guidance is not always better. In Fig. 4(c), success increases from 36.0% at $\rho = 0$ to 58.4% at $\rho = 0.4$, but drops to 51.2% at $\rho = 1.0$. This suggests that moderate guidance improves target alignment while retaining the policy's learned manipulation behavior. As ρ increases, the geometric objective has greater influence relative to the pretrained flow. Excessive guidance may interfere with the reach-to-grasp transition.

Batched particle guidance remains efficient in inference speed. Figure 8 measures 100 calls on ten recorded robot observations. At the deployed $K = 8$, inference takes 72.4 ± 2.8 ms, compared with 54.3 ± 0.8 ms for vanilla, while the complete client decision takes 79.1 ms. Increasing K from 2 to 8 adds only 5.0 ms, demonstrating efficient batched inference; the benchmark excludes compilation, VLM queries, since these are all one-time overhead.

6 Conclusion

We introduced DPoS, a training-free method for steering frozen VLA policies through distribution matching. The MMD energy naturally yields attraction for target alignment and repulsion for proposal coverage. Experiments demonstrate improved OOD task success and trajectory overlap consistent with retaining pretrained action patterns.

REFERENCES

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

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Jiahang Cao, Yize Huang, Hanzhong Guo, Qiang Zhang, Rui Zhang, Weijian Mai, Mu Nan, Jiaxu Wang, Hao Cheng, Jingkai SUN, et al. Compose your policies! improving diffusion-based or flow-based robot policies via test-time distribution-level composition. In *International Conference on Learning Representations*, volume 2026, pp. 152825–152879, 2026.

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

Open X Collaboration, Rehman Abby, Gupta Abdul, Maddukuri Abhinav, Gupta Abhiram, Padalkar Abhishek, Lee Abhishek, Pooley Abraham, Gupta Acorn, Mandlekar Agrim, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *CoRR*, 2023.

Gabriele Corso, Yilun Xu, Valentin De Bortoli, Regina Barzilay, and Tommi Jaakkola. Particle guidance: non-iid diverse sampling with diffusion models. In *International Conference on Learning Representations*, volume 2024, pp. 22480–22507, 2024.

Pim De Haan, Dinesh Jayaraman, and Sergey Levine. Causal confusion in imitation learning. *Advances in neural information processing systems*, 32, 2019.

Sombit Dey, Jan-Nico Zaech, Nikolay Nikolov, Luc Van Gool, and Danda Pani Paudel. Revla: Reverting visual domain limitation of robotic foundation models. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8679–8686. IEEE, 2025.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Maximilian Du and Shuran Song. Dynaguide: Steering diffusion polices with active dynamic guidance. *Advances in Neural Information Processing Systems*, 38:44192–44221, 2026.

Yu Fang, Yuchun Feng, Dong Jing, Jiaqi Liu, Yue Yang, Zhenyu Wei, Daniel Szafir, and Mingyu Ding. When vision overrides language: Evaluating and mitigating counterfactual failures in vlas. *arXiv preprint arXiv:2602.17659*, 2026.

Bowen Feng, Zhiting Mei, Julian Ost, Filippo Ghilotti, Baiang Li, Roger Girgis, Anirudha Majumdar, and Felix Heide. Verdi: Vlm-embedded reasoning for autonomous driving. *arXiv preprint arXiv:2505.15925*, 2025.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The journal of machine learning research*, 13:723–773, 2012.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Jonathan C Kao, Jason Chan, and Andy Wang. Flow control: Steering vision-language-action models with simple real-time inputs. *arXiv preprint arXiv:2606.10180*, 2026.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

Yi Li, Yuquan Deng, Jesse Zhang, Joel Jang, Marius Memmel, Caelan Garrett, Fabio Ramos, Dieter Fox, Anqi Li, Abhishek Gupta, et al. Hamster: Hierarchical action models for open-world robot manipulation. In *International Conference on Learning Representations*, volume 2025, pp. 24040–24068, 2025.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Fangchen Liu, Kuan Fang, Pieter Abbeel, and Sergey Levine. Moka: Open-world robotic manipulation through mark-based visual prompting. *arXiv preprint arXiv:2403.03174*, 2024.

Shuo Liu, Ishneet Sukhvinder Singh, Yiqing Xu, Jiafei Duan, and Ranjay Krishna. Vls: Steering pretrained robot policies via vision-language models. *arXiv preprint arXiv:2602.03973*, 2026.

Mitsuhiko Nakamoto, Oier Mees, Aviral Kumar, and Sergey Levine. Steering your generalists: Improving robotic foundation models via value guidance. *arXiv preprint arXiv:2410.13816*, 2024.

Carlota Parés-Morlans, Nils Kuhn, Isabel Liu, Alberta Longhini, and Jeannette Bohg. Demystifying when and why vlas fail in contact-rich tasks and how to fix them. *arXiv preprint arXiv:2608.01402*, 2026.

Sungjae Park, Homanga Bharadhwaj, and Shubham Tulsiani. Demodiffusion: One-shot human imitation using pre-trained diffusion policy. *arXiv preprint arXiv:2506.20668*, 2025.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, et al. Spatialvla: Exploring spatial representations for visual-language-action model. *arXiv preprint arXiv:2501.15830*, 2025.

Yunzhou Song, Long Le, Yong-Hyun Park, Jie Wang, Junyao Shi, Lingjie Liu, Jiatao Gu, Eric Eaton, Dinesh Jayaraman, and Kostas Daniilidis. Omniguide: Universal guidance fields for enhancing generalist robot policies. *arXiv preprint arXiv:2603.10052*, 2026.

Andy Tang, William Chen, Andrew Wagenmaker, Chelsea Finn, and Sergey Levine. Improving robotic generalist policies via flow reversal steering. *arXiv preprint arXiv:2606.13675*, 2026.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. *arXiv preprint arXiv:2506.15799*, 2025.

Yanwei Wang, Lirui Wang, Yilun Du, Balakumar Sundaralingam, Xuning Yang, Yu-Wei Chao, Claudia Pérez-D'Arpino, Dieter Fox, and Julie Shah. Inference-time policy steering through human interactions. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 15626–15633. IEEE, 2025.

Yilin Wu, Ran Tian, Gokul Swamy, and Andrea Bajcsy. From foresight to forethought: Vlm-in-the-loop policy steering via latent alignment. *arXiv preprint arXiv:2502.01828*, 2025.

Takuma Yoneda, Luzhe Sun, Ge Yang, Bradly Stadie, and Matthew Walter. To the noise and back: Diffusion for shared autonomy. *arXiv preprint arXiv:2302.12244*, 2023.

Xueyang Zhou, Yangming Xu, Guiyao Tie, Yongchao Chen, Guowen Zhang, Duanfeng Chu, Pan Zhou, and Lichao Sun. Libero-pro: Towards robust and fair evaluation of vision-language-action models beyond memorization. *arXiv preprint arXiv:2510.03827*, 2025.

A PRELIMINARIES

A.1 FLOW-MATCHING ACTION POLICIES

VLA maps an observation o and a natural language instruction ℓ to an action chunk $A \in \mathbb{R}^{H\times D}$ of H future actions and D dimension action space. Specially, the VLA with flow-matching action head (Lipman et al., 2022) uses the VLM as reasoning backbone and a flow head that predict the action chunk as $p(A|o,\ell)$. With ϵ sampled from normal distribution as $\epsilon \sim \mathcal{N}(0,I)$, and flow time $t \in [0,1]$, the interpolant is $x_t = t\epsilon + (1-t)A$. A network approximates the velocity as $dx_t/dt = \epsilon - A \approx v_\theta(x_{t,t,o,\ell})$. Sampling intergarates the learned ODE from $x_1 = \epsilon$ to x_0 with N forward integration steps of size $\delta = \frac{1}{N}$ as:

$$x_{t-\delta} = x_t - \delta\, v_\theta(x_t, t, o, \ell) \tag{16}$$

The one-step estimate of the denoised action chunk is

$$\widehat{A}_0(x_t) = x_t - t\, v_\theta(x_t, t, o, \ell). \tag{17}$$

A.2 KERNELS AND MAXIMUM MEAN DISCREPANCY

Let $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be a positive-definite kernel with reproducing kernel Hilbert space (RKHS) $\mathcal{H}_k$. The kernel mean embedding of a distribution P is $\mu_p = \mathbb{E}_{x\sim P}[k(x,\cdot)] \in \mathcal{H}_k$, and the squared maximum mean discrepancy (Gretton et al., 2012) between P and Q is

$$\mathrm{MMD}_k^2(P,Q) = \|\mu_P - \mu_Q\|_{\mathcal{H}_k}^2 \tag{18}$$

$$= \mathbb{E}_{x,x'\sim P} k(x,x') + \mathbb{E}_{y,y'\sim Q} k(y,y') - 2\mathbb{E}_{\substack{x\sim P\\ y\sim Q}} k(x,y). \tag{19}$$

Such MMD defines a metric on probability measures between two manifold and operates as $\mathrm{MMD}_k(P,Q) = 0$ iff $P = Q$, and it satisfies the triangle inequality.

B PROOF FOR PROPOSITION

Proof. For any $x \in \mathcal{X}$ satisfying $\|x - \hat{x}\|_2 \le (\varepsilon - \delta_{\text{est}})/L$, Lipschitz continuity implies

$$J(x) \le J(\hat{x}) + L\|x - \hat{x}\|_2 \tag{20}$$

$$\le J^\star + \delta_{\text{est}} + \varepsilon - \delta_{\text{est}} \tag{21}$$

$$= J^\star + \varepsilon. \tag{22}$$

Thus $x \in \mathcal{S}_\varepsilon$. If $\hat{x}$ is an interior point of $\mathcal{X}$, a sufficiently small positive-radius ball around $\hat{x}$ also lies in $\mathcal{X}$, establishing the final claim. □

C EXACT MANIFOLD OF EQUIVALENT OPTIMA

Proposition 2. *Solution region as an exact Manifold of Equivalent Optima*

Let $\mathcal{F} \subseteq \mathbb{R}^3$ be an open feasible approach domain, and let $G \subseteq \mathrm{SO}(3)_c$ be a compact subgroup of rotations about an object reference center c. Assume $g\mathcal{F} = \mathcal{F}$ and $J(g\cdot x) = J(x)$ for every $g \in G$ and $x \in \mathcal{F}$, where $J : \mathcal{F} \to \mathbb{R}$ is an approach cost. If $x^\star$ is a global minimizer of J, then

$$G \cdot x^\star \subseteq \arg\min_{x\in\mathcal{F}} J(x).$$

The orbit is a compact embedded manifold of dimension $\dim G - \dim G_{x^\star}$, where $G_{x^\star} = \{g \in G : g\cdot x^\star = x^\star\}$. Positive orbit dimension implies infinitely many optimal positions. For $G = \mathrm{SO}(3)_c$ and $x^\star \neq c$, the orbit is

$$\mathcal{S}_R(c) = \{x : \|x - c\|_2 = R\}, \qquad R = \|x^\star - c\|_2.$$

For axial rotational symmetry, an optimal point off the rotation axis generates a horizontal circle of optimal approach positions.

Proof. For every $g \in G$, feasibility invariance gives $g \cdot x^\star \in \mathcal{F}$, and cost invariance gives

$$J(g \cdot x^\star) = J(x^\star) = \min_{x \in \mathcal{F}} J(x).$$

Thus the orbit is optimal. The orbit theorem for compact Lie group actions implies that $G \cdot x^\star$ is an embedded manifold diffeomorphic to $G/G_{x^\star}$, with the stated dimension. Compactness follows from continuity of the orbit map. Full rotations about c generate the sphere of radius $\|x^\star - c\|_2$; axial rotations generate a circle for off-axis points. □

D Similarity Kernel

$\kappa_h(a, b)$ increases when a and b are close and decreases when they are far apart as:

$$\kappa_h(a, b) = \left(1 + \frac{\|a - b\|_2^2}{h^2}\right)^{-1} = \frac{h^2}{h^2 + \|a - b\|_2^2}. \tag{23}$$

Here, $h > 0$ is the kernel-bandwidth hyperparameter controlling the spatial range of attraction and repulsion. We keep h fixed during inference and select its value empirically on a validation set.

E Experimental Setup Details

Simulation benchmark and evaluation. We evaluate on LIBERO-PRO (Zhou et al., 2025; Liu et al., 2023), a benchmark designed to measure out-of-distribution generalization beyond memorized training configurations. Among its five perturbation dimensions, we focus on Position (`swap`) and Task (`task`), which test adaptation to displaced objects and novel task–target combinations. Applying these perturbations to LIBERO-Object, LIBERO-Spatial, and LIBERO-Goal gives six suites. Each suite contains 10 tasks. For the main comparison, we execute one rollout from each of 50 prescribed initial states per task, yielding 500 episodes per suite and 3,000 episodes per method across all six suites. Ablation studies use 20 rollouts per task, totaling 200 rollouts per suite.

Simulation policy and execution. We build DPoS on the official $\pi_{0.5}$-LIBERO checkpoint and keep the policy frozen throughout evaluation. All locally evaluated methods use the same checkpoint, paired initial states, and BDDL instructions. Images are resized to 224×224, and the policy replans after executing five actions from each predicted 10-step action chunk.

Real-world hardware and control. We use a physical Franka Research3 with the official DROID (Khazatsky et al., 2024) setup. The policy receives one external RGB view, a wrist-camera view, and robot proprioception; both images are resized to 224×224. The robot operates at 15 Hz with joint-velocity control for the arm and position control for the gripper.

Real-world tasks and evaluation. The nine tasks in Fig. 3 are organized into Object, Spatial, and Compositional suites, each with easy, medium, and hard levels. The Object suite refers by name to objects absent from the DROID training set. The Spatial suite specifies targets through relative positions rather than names. The Compositional suite uses multi-stage instructions with indirect references involving multiple objects. Difficulty increases with scene clutter and reasoning complexity. We conduct 10 trials per task for each method, totaling 90 trials per method and 360 physical rollouts across $\pi_{0.5}$, Point Attraction, OmniGuide, and DPoS. A rollout succeeds only if the robot completes the entire language-specified task. Task instructions and representative rollout sequences are provided in Appendices F and G, respectively.

F Real-world Task Prompt Reference

We provide the task instructions corresponding to the nine settings in Fig. 3.

Table 2: **Task prompt reference for the real-world evaluation.** Instructions are grouped by suite and difficulty, matching Fig. 3.

Suite	Difficulty	Task instruction
Object	Easy ★	Pick up the controller and put it in the box.
	Medium ★★	Pick up the marker and put it in the box.
	Hard ★★★	Pick up the board eraser and put it in the box.
Spatial	Easy ★	Pick up the object far from the box and put it in the box.
	Medium ★★	Pick up the object on the left of the three objects and put it in the box.
	Hard ★★★	Pick up the object in the middle of the pile and put it in the box.
Compositional	Easy ★	Pick up the object pointed by the SpongeBob and put it in the bowl.
	Medium ★★	Pick up the object next to the plate and put it in the hat. Then, pick up the controller and put it on the plate.
	Hard ★★★	Pick up the object next to the board eraser and put it in the hat. Then, pick up the board eraser and put it in the drawer.

G Real-world Task Instructions and Rollouts

Figures 9, 10, and 11 show selected physical rollouts for the Object, Spatial, and Compositional suites, respectively. Each figure covers the easy, medium, and hard tasks and compares the frozen $\pi_{0.5}$ policy with DPoS. Five frames are sampled uniformly within each rollout and ordered from left to right; corresponding columns therefore indicate relative progress, rather than synchronized times across methods. Outcome markers refer to the full task, and these selected examples supplement the aggregate success rates in Fig. 2.

Object identity and spatial relations. The Object tasks ask the robot to place a named controller, marker, or board eraser in a box. The Spatial tasks instead identify the target by its distance from the box, its leftmost position among three objects, or its position in the middle of a pile. These examples make explicit the distinction between recognizing a named object and grounding a target through its surroundings.

Compositional instructions. The easy task refers to an object through the direction indicated by a SpongeBob figure. The medium task first moves the object next to a plate to a hat, then moves the controller to the plate. The hard task first moves the object next to the board eraser to a hat, then moves the board eraser to a drawer. The latter two instructions are abbreviated with arrows in Fig. 11, where each arrow denotes an object-to-destination assignment and the assignments are executed in sequence.

H Detals about T-SNE and Nearest-neighbor results

DPoS samples actions close to the pre-trained action prior. Fig. 5 compares trajectories from matched successful episodes across three methods. We represent each trajectory by concatenating 16 uniformly sampled 7D actions into a 112-dimensional vector, standardize these representations, and visualize them using per-suite t-SNE embeddings. We additionally report *Trajectory Mixing*, a normalized inverse-Simpson label-mixing score computed over 10-nearest-neighbor neighborhoods in the original standardized 112-dimensional space. When evaluated against frozen $\pi_{0.5}$ trajectories, DPoS achieves higher mixing than point attraction on all four tasks. These results suggest that, DPoS's sampled actions remain close to the pre-trained action distribution. In Fig. 5, DPoS does the exploration on the contour of the pre-trained action distribution for task success while not shifting too far toward a distinctly separated set of behaviors.

I LIMITATIONS

DPoS steers a frozen policy but does not explicitly learn new motor skills. Its effectiveness therefore depends on the pretrained policy's manipulation capabilities. For example, in a charger-cable connection task, the robot successfully grasped and positioned the charger near the cable but failed to complete insertion. This illustrates a limitation of spatial guidance: reaching the correct region alone may not be sufficient for fine-grained, contact-rich manipulation requiring precise alignment and control.

J AI USE DISCLOSURE

We used AI tools to correct grammatical errors, polish final writings and assist with coding. All AI-assisted code was verified. The authors take responsibility for the final manuscript, including the correctness of its claims, experimental results, code, and visualizations, whether or not they were produced with AI assistance.

K REPRODUCIBILITY STATEMENT

We provide the core code as detailed implementation of the DPoS algorithm built on $\pi_{0.5}$ in the supplementary materials. The code isolates Shell-MMD guidance and its integration into the policy's sampling procedure to facilitate inspection and reimplementation of the method described in Sec. 4. The main paper describes the experimental setups and evaluation protocols, while Appendix B provides the proof of the near-optimality result under the stated assumptions. Real-world task instructions and representative rollout sequences are provided in Appendices F and G, respectively.

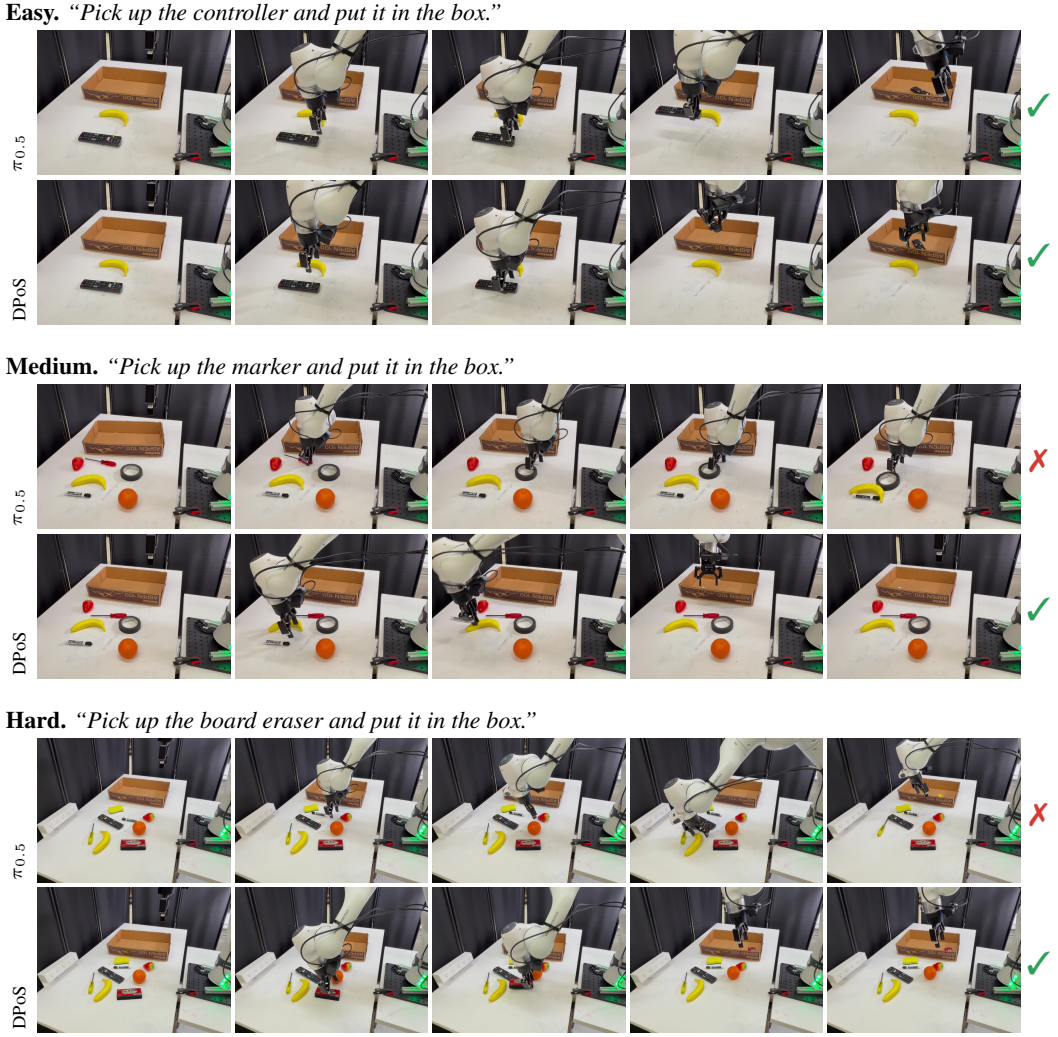


Figure 9: **Real-world rollouts, Object suite.** For each task, the top row shows the frozen $\pi_{0.5}$ policy and the bottom row shows $\pi_{0.5}$ with DPoS. Frames are sampled uniformly over each rollout, from start (left) to end (right); a green check denotes success and a red cross denotes failure. These selected rollouts illustrate behavior and do not estimate task success rates.

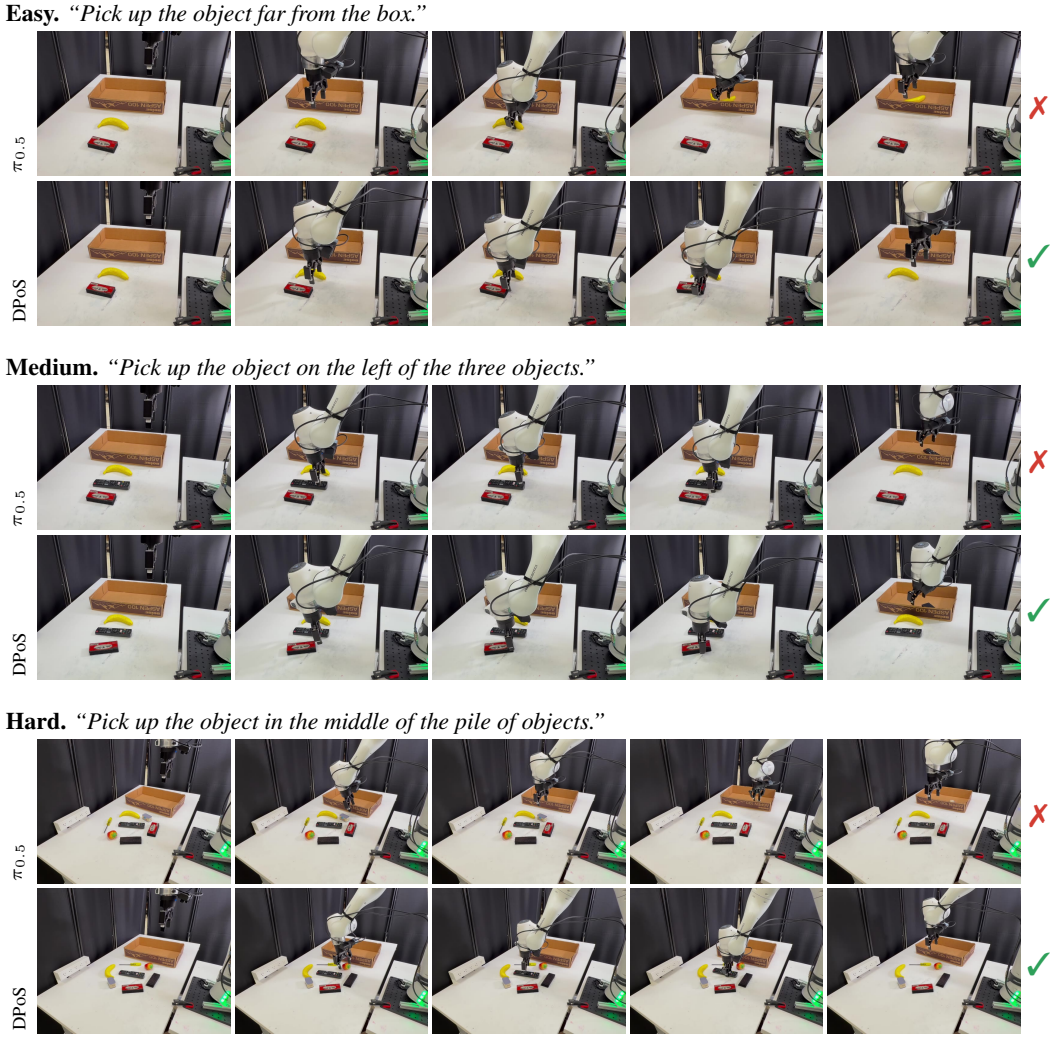


Figure 10: **Real-world rollouts, Spatial suite.** For each task, the top row shows the frozen $\pi_{0.5}$ policy and the bottom row shows $\pi_{0.5}$ with DPoS. Frames are sampled uniformly over each rollout, from start (left) to end (right); a green check denotes success and a red cross denotes failure. These selected rollouts illustrate behavior and do not estimate task success rates.

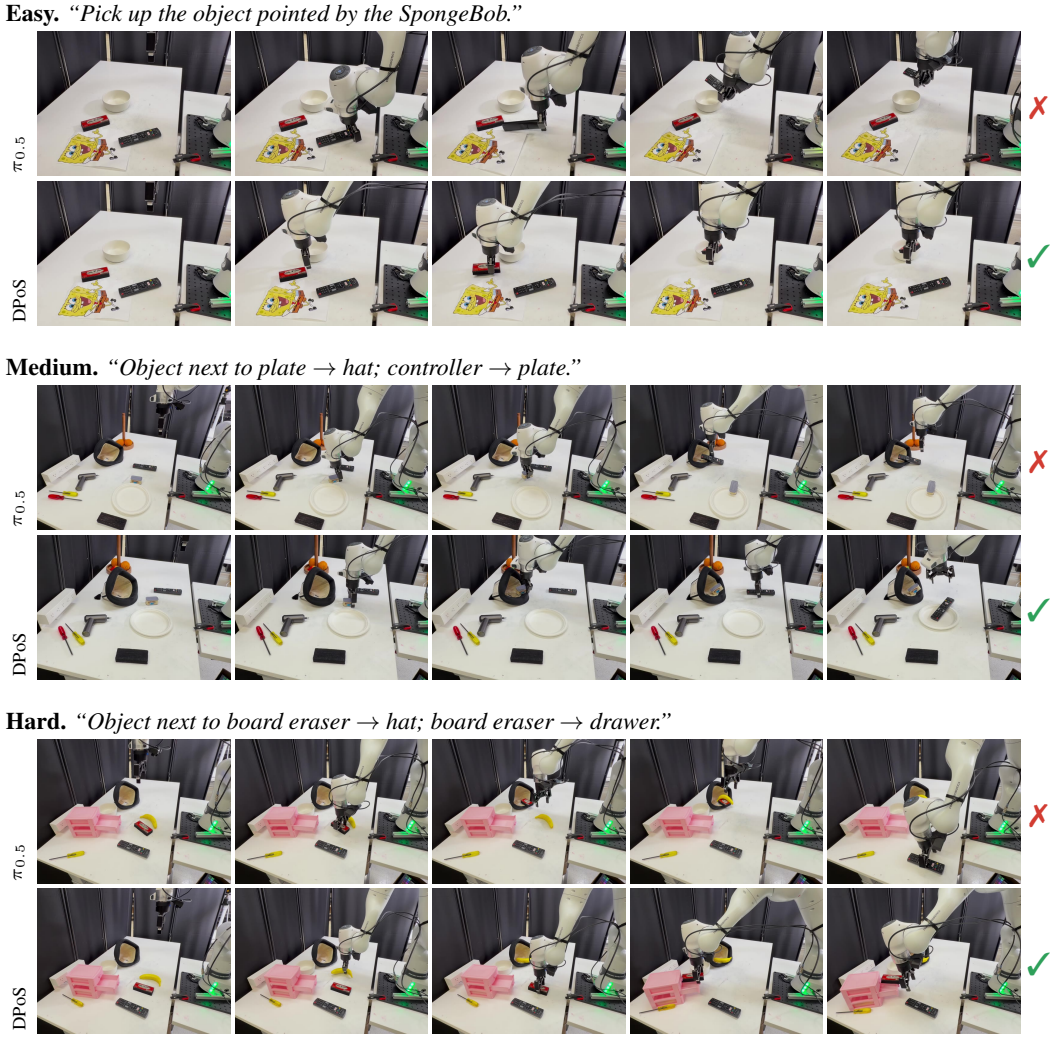


Figure 11: **Real-world rollouts, Compositional suite.** For each task, the top row shows the frozen $\pi_{0.5}$ policy and the bottom row shows $\pi_{0.5}$ with DPoS. Frames are sampled uniformly over each rollout, from start (left) to end (right); a green check denotes success and a red cross denotes failure. These selected rollouts illustrate behavior and do not estimate task success rates.