DISCERN: LANGUAGE-GUIDED BEHAVIORAL SEARCH WITH WORLD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

World models offer a way to evaluate policy behaviors by predicting action consequences before real-world execution. For test-time policy steering, however, their effectiveness depends not only on predicting outcomes but also on proposing sufficiently diverse behaviors to search over. Existing steering methods based on world models primarily sample actions from a policy under a fixed instruction, restricting the search to the behaviors induced by that language conditioning. We introduce DISCERN, which steers a frozen Vision-Language-Action policy at test time by searching over task-preserving rephrasings of the original task instruction. For each rephrasing, an action-conditioned world model predicts the policy's behavior over several action chunks in closed loop, and a vision-language verifier selects the instruction whose predicted outcome best satisfies the task. On LIBERO-PRO, DISCERN improves frozen-policy success by 18.7 percentage points on tasks with lock-in failures, where the policy ignores the instruction and repeats a motion from training, significantly outperforming all baselines. Our analyses further show that varying the instruction reaches successful behaviors that action sampling with a fixed instruction misses, and that closed-loop previews improve language selection over a single action chunk prediction. These results identify language as an effective way to sample behaviors for policy steering with world models.

1 INTRODUCTION

Natural language offers a high-level interface for commanding general-purpose robot policies, allowing users to specify desired behaviors without prescribing low-level actions or retraining the policy. Vision-Language-Action (VLA) policies trained on large-scale manipulation data make this increasingly practical, bridging broad manipulation skills with strong language-conditioning derived from internet-scale pretraining (Kim et al., 2024; Black et al., 2026; Intelligence et al., 2025). Yet, specifying the desired behavior in language does not guarantee that the policy will follow this instruction. The same skill may succeed under one phrasing or scene configuration and fail under another, as the policy can be biased toward particular behaviors or interpretations of an instruction (Huang et al., 2026a). As a result, steering a VLA is not only a matter of stating the task correctly at the start, but also of finding instructions that reliably induce the desired behavior.

Policy steering can be broadly categorized into two lines of work: language-based steering and world-model-based behavioral search. Language-based steering shows that varying the instruction can generate diverse behaviors from a VLA, but existing methods rely on task-specific training and real-world policy rollouts to guide this search (Jiang et al., 2026; Jeong et al., 2026; Bhatia et al., 2026). World models provide a natural intermediary for evaluating policy behavior before costly real-world interactions (Yang, 2026). However, their usefulness depends on the diversity of policy behaviors that can be predicted and evaluated. Existing world-model steering methods primarily search over actions sampled from a policy conditioned on a fixed instruction (Cui et al., 2026; Seo et al., 2026), restricting action selection to a narrow region of the policy's behavioral distribution. This motivates an approach combining the complementary strengths of language and world models: using language to sample a broader set of behaviors from a frozen VLA, and a world model to evaluate those behaviors before execution.

Our key insight is that effective world-model search requires not only accurate prediction, but also a proposal mechanism that exposes behaviorally distinct policy modes and a preview horizon long

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.

enough for those modes to diverge. We introduce DISCERN, which uses task-preserving language variations to sample diverse behaviors from a frozen VLA and evaluates them in a closed-loop with an action-conditioned world model. For every candidate instruction, the policy is repeatedly queried on the world model's predicted observations, so later actions respond to the predicted consequences of earlier ones. This is important because different instructions may induce similar immediate action chunks yet diverge substantially after several decision steps. A vision-language verifier then compares the resulting previews and selects the instruction that is executed in the real environment.

On the *task* and *swap* perturbations of LIBERO-PRO (Zhou et al., 2026), DISCERN raises the success of a frozen $\pi_{0.5}$ from 10.9% to 29.6% on tasks where the policy ignores the original instructions, significantly outperforming all baselines. Our analyses support the two design choices underlying this gain: task-preserving language proposals reach substantially more desired behaviors than action resampling with a fixed instruction, while closed-loop world-model previews improve selection over open-loop prediction by allowing candidate behaviors to separate over multiple policy decisions.

Our main contribution is DISCERN, a test-time behavioral search framework that combines task-preserving language proposals with closed-loop world-model evaluation to improve a frozen VLA, without trying candidates in the environment or updating the policy.

2 RELATED WORK

Training VLAs to follow instructions. Vision-language-action models (VLAs) trained on large robotic datasets generalize across objects, scenes, and tasks (Kim et al., 2024; Black et al., 2026; Intelligence et al., 2025; NVIDIA et al., 2025). However, despite internet-scale vision and language pretraining, their instruction following remains brittle: semantically equivalent instructions produce different behavior, and small changes to the scene can break it altogether (Liu et al., 2026; Kwok et al., 2026b). Prior work addresses this at training time, either by regularizing the policy to preserve pretrained vision-language knowledge (Driess et al., 2025; Pertsch et al., 2025; Intelligence et al., 2025; Huang et al., 2026a) or through denser supervision and steerability-preserving post-training (Kim et al., 2026; Chen et al., 2026b; Huang et al., 2026a). These methods all modify the underlying policy. DISCERN aims to recover the instruction-following capabilities of VLAs, but keeps the policy frozen: we change the instruction the policy receives, not its weights.

Steering frozen policies at inference time. Rather than modifying the policy, inference-time steering keeps it fixed and spends additional compute at deployment to improve its behavior. A common approach is to steer in action space (Wagenmaker et al., 2025; Longhini et al., 2026). Other methods sample action candidates and select among them with a value function (Nakamoto et al., 2025), a trained verifier (Kwok et al., 2025; 2026a), or a vision-language verifier (Wu et al., 2025; 2026). Some guide the denoising process itself, using dynamics-model gradients (Du & Song, 2025), a scene-specific reward (Li et al., 2026; Liu et al., 2026), or a learned latent (Tan et al., 2026; Shah et al., 2025). Both bodies of work search only over actions the current instruction already proposes, which, in many cases, is insufficient for successful behavior. This motivates steering through the instruction instead. Prior work executes instruction candidates, either as an exploration axis during reinforcement learning (Jiang et al., 2026; Bhatia et al., 2026) or to learn offline which instructions a policy responds to (Jeong et al., 2026). Instructions can also be ranked with a trained contrastive verifier (Kwok et al., 2026b), but such a verifier scores only the first sampled action chunk and not the behavior that follows. In contrast, DISCERN evaluates each instruction by the closed-loop behavior it induces in a world model, which requires neither real executions nor a trained verifier.

Evaluating policy behaviors with world models. World models predict the consequences of a candidate behavior instead of executing it, replacing real rollouts with predicted ones. In policy steering, world models have mainly been used to evaluate actions under a specific language instruction. (Cui et al., 2026) samples action chunks and motion primitives from a frozen policy, predicts their outcomes with an action-conditioned latent world model, and ranks them with a language-conditioned value model. Similarly, (Wu et al., 2025) predicts latent outcomes for sampled action plans and aligns a language model to these latents so that a vision-language model can evaluate them. These methods are the closest to ours, but each trains an additional component and, like the action-space methods above, searches over actions under a fixed instruction. DISCERN instead uses the world model to evaluate *different* instructions: we roll out the frozen policy closed-loop under each can-

didate instruction and compares the resulting behaviors rather than individual action chunks. The policy, world model, and verifier are all frozen and generic, with no task- or scene-specific training.

3 Problem Formulation

Policy and task alignment. We consider a robot policy π_θ that maps the current observation o_t and a deployment-time control variable u_t to robot actions, $a_t \sim \pi_\theta(\cdot \mid o_t, u_t)$. Here, u_t denotes any input or sampling choice available at test time that can influence the behavior produced by the fixed policy. For example, in a VLA, u_t may be the language instruction, while in diffusion- or flow-based policies, it may also include the latent noise used to sample actions. A task q specifies the desired behavior or goal. We write $G_q(\tau) \in \{0, 1\}$ for whether a complete trajectory successfully accomplishes the task.

Test-time behavioral optimization. At a given observation o_t, different choices of the deployment-time control variable u_t may induce different future behaviors, not all of which are equally successful for the current task. Our goal is to choose these variables at test time, without observing real-world rollouts, so as to maximize the probability of task success. We seek

$$u_t^\star \in \arg\max_{u_t} \mathbb{E}_{\tau \sim P_\theta^\pi(\cdot \mid o_t, u_t)} \left[G_q(\tau) \right], \tag{1}$$

where $P_\theta^\pi(\cdot \mid o_t, u_t)$ denotes the distribution over future trajectories induced by the fixed policy and environment from the current observation o_t under control input u_t.

Deployment constraint. At deployment, a steering method must search the optimal control variable $u_t^\star$ that maximizes the objective in Equation 1, without executing candidate behaviors in the environment before committing to one nor having access to the simulator's state or task reward. It may, however, query the frozen policy and an action-conditioned world model f_ϕ, which predicts the next policy observation after executing an action a_t, $\hat{o}_t = f_\phi(o_t, a_t)$.

4 Method

We introduce DISCERN, a language-guided method to steer policies through test-time behavioral optimization with world models. Our key insight is to use language as a deployment-time control variable to broaden the behavioral distribution exposed by a frozen policy. We instantiate this idea with a VLA policy, using the language instruction ℓ as the control variable, such that $u_t = \ell_t$. Actions are sampled as $a_t \sim \pi_\theta(\cdot \mid o_t, \ell)$, where $o_t = [I_t^1, \ldots, I_t^n]$ comprises n camera views. Figure 1 provides an overview of DISCERN, which steers a frozen policy toward the desired task behavior by proposing task-preserving language instructions (Sec. 4.1), predicting the behaviors they induce through closed-loop policy–world-model rollouts (Sec. 4.2), and using a vision-language verifier to select the instruction whose predicted outcome best aligns with the task (Sec. 4.3). The policy then acts in the environment under the selected instruction.

4.1 Language as a Control Variable

For a task q, we denote by $\mathcal{L}(q)$ the set of language instructions that preserve the task objective, with the original instruction used for training denoted as $\ell^0 \in \mathcal{L}(q)$. With language as the control variable, the objective in Equation 1 is to find an instruction $\ell^\star \in \mathcal{L}(q)$ that maximizes the probability of task success under the frozen policy given the current observation o_t. To obtain a finite set of candidate instructions $\mathcal{L}(q)$, we follow prior work that uses VLMs to propose instructions for robot policies (Bhatia et al., 2026; Chen et al., 2026b; Shi et al., 2025) and prompt a VLM once per task, using the initial image and ℓ^0, to generate $K-1$ task-preserving alternatives (prompt in Appendix C). These may refer to an object by its appearance, restate a spatial relation, or describe the goal at a different level of abstraction, while keeping the task unchanged, as highlighted in Figure 1. Together, they form the candidate set $\mathcal{L}(q) = \{\ell^k\}_{k=0}^{K-1}$, which remains fixed throughout the episode.

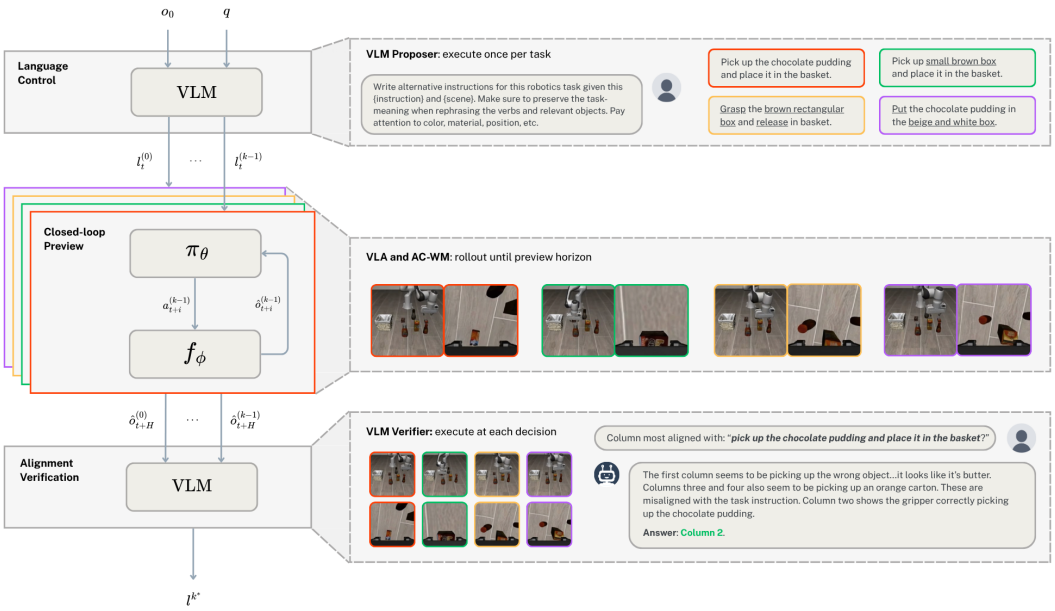


Figure 1: Overview of DISCERN. At deployment, a VLM proposer uses the initial observation o_0 and the task description q to generate a fixed set of task-preserving language controls $\mathcal{L}(q)$. At each decision boundary, each candidate ℓ^k conditions the same frozen policy π_θ, whose actions are rolled out in closed loop with the frozen action-conditioned world model f_ϕ. The resulting terminal predicted observations are presented jointly to a VLM verifier together with q, which selects $\ell^{k^\star}$ based on predicted task alignment.

4.2 Closed-Loop Behavior Prediction

Different candidate instructions $\ell^k \in \mathcal{L}(q)$ often yield nearly identical first action chunks. Evaluating the first chunk alone therefore cannot easily separate such candidates. We therefore roll out the frozen policy in closed loop with a world model over multiple policy queries.

For each candidate ℓ^k, $k \in \{0, \dots, K-1\}$, starting from $\hat{o}_0^k = o_t$, the policy predicts an action chunk from the latest predicted observation, and a pre-trained, action-conditioned world model f_ϕ predicts its outcome:

$$\hat{a}_j^k \sim \pi_\theta\big(\cdot \mid \hat{o}_j^k, \ell^k\big), \qquad \hat{o}_{j+1}^k = f_\phi\big(\hat{o}_j^k, \hat{a}_j^k\big), \qquad j = 0, \dots, J-1, \tag{2}$$

where $J = \lceil H/H \rceil$ queries cover a horizon of H low-level actions. We call this predicted trajectory the *preview* of ℓ^k. Since all previews share the same initial observation, policy, and world model, they differ only in the instruction. The final observation $\hat{o}_J^k$ is passed to the verifier to assess how well the progress of each candidate aligns with the task q.

4.3 Verifying Task Alignment

The objective in Equation 1 is defined by task success, but the world-model previews may not cover the full episode length. We therefore select among candidates by assessing which predicted outcome shows the most progress toward the task, using a vision-language verifier conditioned on the task description q.

We tile the K final predicted observations into a single image with one column per candidate, each column stacking the camera views of $\hat{o}_J^k$ (third-person and wrist; Figure 1). The canonical instruction ℓ^0 always occupies the first column and serves as the default, while the remaining candidates are shuffled to reduce position bias. The verifier returns the index of the candidate with the strongest progress,

$$k^\star = \text{Verify}\left(\hat{o}_J^0, \dots, \hat{o}_J^{K-1}; q\right), \tag{3}$$

and the policy is conditioned on $\ell^\star = \ell^{k^\star}$ in the real environment. Presenting all candidates jointly lets the verifier compare them directly and requires a single query per decision. The verifier is

instructed to keep ℓ^0 unless another candidate shows clearly greater progress, so an uninformative prediction does not trigger an unnecessary switch (prompt in Appendix C). Once the verifier switches away from ℓ^0, the selected instruction is kept for the rest of the episode, as we empirically found that selecting at each time step the best language candidate does not provide significant gains.

5 EXPERIMENTS

In this section, we evaluate DISCERN to study whether language sampling and world models are effective for test-time behavioral search. We design our experiments to answer the following key research questions:

RQ1 Does DISCERN lead to improved performance over existing test-time steering methods?

RQ2 Can language-guided world model search help us recover successful behaviors effectively?

RQ3 When are world-model previews useful for selecting desired candidate behaviors?

RQ4 What are the contributions and design choices of each component in DISCERN?

5.1 EXPERIMENTAL SETUP

Environment and Metrics. We evaluate DISCERN on LIBERO-PRO (Zhou et al., 2026), an extended LIBERO (Liu et al., 2023) benchmark that perturbs tasks to create challenging generalization scenarios for existing policies. Following previous work (Liu et al., 2026), we test over *swap* and *task* perturbations. Swap perturbations change the positions of objects, and task perturbations redefine the instruction-relevant objects. We report binary, episode-level success rate as our primary metric, averaged over 10 initial states for each of the 80 tasks. To isolate the contribution of language steering, we additionally report results on 23 *locked-in* tasks, in which the policy is steerable towards the right behavior while the original language is not effective. Specifically, in these tasks the policy ignores the original language instruction and defaults to a training-associated behavior, such as grasping or placing an object at the corresponding training location (Figure 2). For these tasks, there exists at least one rephrasing that improves the outcome relative to the canonical instruction. On the remaining tasks, no rephrasing improves the outcome, as the policy is not steerable, which dilutes aggregate gains over the full benchmark. Improving language steerability through policy training (Chen et al., 2026b; Shi et al., 2025) could broaden the effectiveness of our method on all tasks, but is complementary to our test-time approach and outside the scope of this work.

Models. All of our baselines use the same three frozen components: a $\pi_{0.5}$ policy trained on LIBERO (Chen et al., 2026a), the A2World action-conditioned world model (Huang et al., 2026b), and Gemini Robotics ER 2 as the verifier (full configuration in Appendix A.1). Since A2World was trained on demonstrations from the LIBERO scenes used to derive LIBERO-PRO tasks, we do not claim to test for world-model generalization.

Baselines. We compare DISCERN against baseline techniques that also aim to induce greater steerability of the base $\pi_{0.5}$ model. Table 1 summarizes the main differences between these baselines. Additional implementation details can be found in Appendix A.

- **Canonical instruction:** Feeds the original language command used for training directly to the policy, without the world model in the loop.
- **Random instruction:** Executes one of the eight language candidates, drawn uniformly and held constant throughout the episode. This isolates the gain of selection from rephrasing.
- **CoVer** (Kwok et al., 2026b): Samples jointly over languages and actions and uses a learned contrastive verifier to select action candidates given language, observation, and action tuples. We use the same set of language rephrases to isolate the selection mechanism.
- **FOREWARN** (Wu et al., 2025): Uses a latent world model to predict the outcomes of candidate action plans. A learned narrator converts the predicted latent futures into language, which a VLM then evaluates for task alignment.
- **RL^2-VLA** (Tan et al., 2026): Learns an offline-RL policy over VLA latents and composes its steering signal with the frozen VLA during inference. A learned failure detector activates this intervention selectively.

Table 1: Comparison to other steering methods. *Varies instruction*: the candidates are induced by language sampling. *Predicts outcomes*: the scorer sees the predicted consequences of each candidate's actions. *Off-the-shelf verifier*: the selection mechanism is training-free and generalizable.

Method	Varies instruction	Predicts outcomes	Off-the-shelf verifier
CoVer	✓	✗	✗
FOREWARN	✓	✓	✗
RL^2-VLA	✓	✗	✗
VLS	✗	✗	✓
DISCERN	✓	✓	✓

- **VLS** (Liu et al., 2026): uses a VLM-derived reward function to guide the denoising process. This tests the ability of differentiable steering methods to reach successful behaviors.
- **Simulator preview:** Replaces the world model previews with the ground-truth forward-simulated futures. This isolates DISCERN from world-model predictive errors.
- **Best fixed instruction:** executes the best language instruction from start to end of the episode. We choose the best in hindsight by rolling out 20 trials for each rephrase until termination. This upper-bounds success on only one decision point at start of episode.
- **Oracle switch:** Starts from the canonical instruction and always switches to the best language instruction at the optimal decision point. The oracle is chosen in hindsight by rolling out a branch of conditional futures at each decision point. This upper-bounds our method by showing what perfect decision-making could achieve.

5.2 DISCERN IS EFFECTIVE FOR TEST-TIME STEERING (RQ1)

We first ask whether DISCERN enables test-time behavioral search without real-world rollouts. We compare our approach to prior methods that build upon an invariant base policy to steer behavior, as well as established bounds showing that, given human-level judgment, world-model steering can be used to reach improved task performance.

Our results show that DISCERN outperforms all other baselines (Table 1), leading with a 5.8 pp improvement in success rate over the canonical instruction. The gains are most evident on lock-in tasks (Figure 2), where the policy picks up objects at the training object's position or places them at the training placement location instead of following the instruction. In this setting, DISCERN beats canonical instructions by 18.7 pp and random instructions by 9.5 pp. Our method's gain over random selection shows that its success depends on not only generating candidate behaviors but also selecting effectively among them. The larger gap to the oracle switch upper bound suggests that selection quality is the main bottleneck and that our method could benefit from a stronger VLM verifier. These results support our hypothesis that world models can be used as behavioral search engines for test-time steering. However, they do not yet explain why our mechanisms for generating and selecting candidate behaviors are more effective than existing methods, motivating the analyses that we perform below.

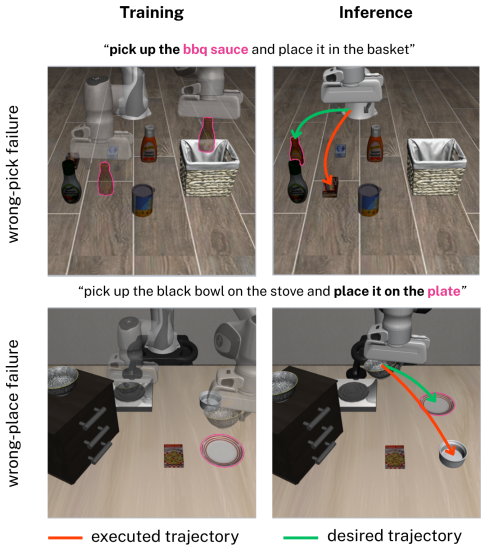


Figure 2: Lock-in failures in baselines. Top: asked to pick up the bbq sauce, the policy grasps the object at the training object's position. **Bottom:** asked to place the bowl on the plate, it places the bowl at the training placement location. Red and green arrows denote executed and desired trajectories, respectively.

Table 2: LIBERO-PRO results. We evaluate all methods using the same frozen $\pi_{0.5}$ policy on 80 LIBERO-PRO tasks (Zhou et al., 2026), with 10 trials per task. These episodes cover the *task* and *swap* perturbations across the four original LIBERO (Liu et al., 2023) task suites. *Lock-in tasks*: the 23 tasks in which the canonical policy exhibits lock-in (Sec. 5.1); *All*: full 80 tasks. Values report mean success rates, with standard errors computed across the tasks. An asterisk marks a baseline for which DISCERN is significantly better on the lock-in tasks under a paired sign-flip permutation test over tasks ($p < 0.05$). Rows below the horizontal rule are oracle-based and not deployable: † denotes simulator previews, and ‡ denotes hindsight upper bounds that evaluate every candidate.

	Lock-in tasks			All		
Method	Task	Swap	Avg.	Task	Swap	Avg.
Canonical instruction	12.2	10.0	10.9* ±5.0	19.5	7.0	13.2 ±3.3
Random instruction	12.4	25.0	20.1* ±4.5	15.5	12.7	14.1 ±2.8
CoVer (Kwok et al., 2026b)	11.1	15.7	13.9* ±5.6	21.3	9.2	15.3 ±3.4
FOREWARN (Wu et al., 2025)	17.8	20.0	19.1* ±4.9	17.2	10.8	14.0 ±2.9
RL^2-VLA (Tan et al., 2026)	10.0	22.9	17.8 ±5.8	18.2	12.3	15.2 ±3.1
VLS (Liu et al., 2026)	18.4	14.3	15.9* ±6.0	20.9	9.2	15.1 ±3.3
DISCERN (world model)	**34.4**	**26.4**	**29.6** ±6.7	**23.5**	**14.5**	**19.0** ±3.5
DISCERN (simulator)†	36.7	32.9	34.3 ±6.3	23.5	16.2	19.9 ±3.4
Best fixed instruction‡	45.6	67.9	59.1	33.3	29.8	31.5
Oracle switch‡	71.1	85.0	79.6	39.8	37.0	38.4

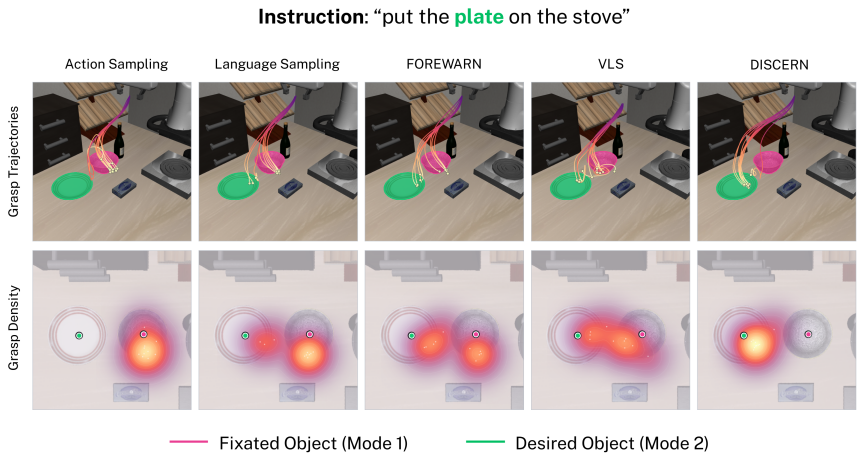


Figure 3: Recoverability of Successful Behavioral Modes. A representative lock-in failure. The policy defaults to a spurious correlation, ignoring the desired object and fixating on the object at the training position. Top: grasp trajectories induced by sampled candidates, shown from a third-person view. Bottom: spatial density of grasp locations, shown from a top-down view of the same scene. Pink dots mark the fixated object (Mode 1) and green dots mark the desired target (Mode 2). Action-space sampling remains concentrated around the fixated mode, whereas language-conditioned sampling shifts behavior toward the desired target.

5.3 Language-Guided World Model Search Improves Recoverability (RQ2)

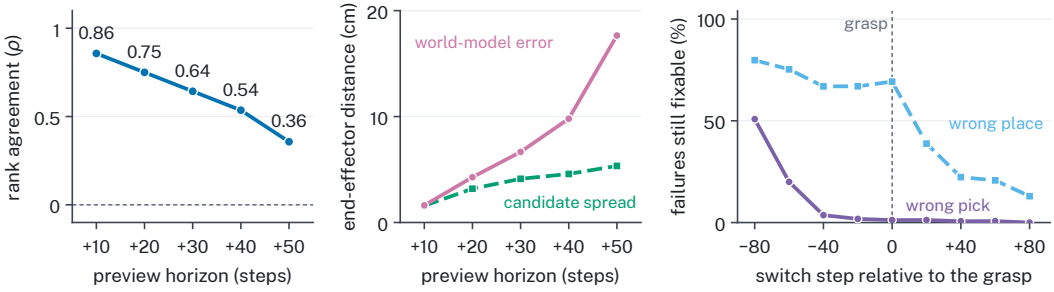
We next ask whether language-guided search with world models is the best way to reach the desired behavioral modes. We use a qualitative example in Figure 3 to examine behavior recoverability. Specifically, we compare random instruction sampling with action sampling via the VLA, along with FOREWARN and RL^2-VLA as representative baselines for instruction sampling and reward-guided steering, respectively. On locked-in tasks, Table 3 shows that grasps from action-level sampling remain concentrated around the fixated object, while those from instruction-level sampling reach the desired object. This confirms that language can induce global behavior-altering perturbations that action sampling cannot. Among instruction-based methods, DISCERN reaches the desired behavioral mode most consistently. While the VLS baseline achieves broad distributional coverage from differentiable steering, it tends to grasp intermediate locations between the fixed and desired

Table 3: Reaching desired objects. Percentage of rollouts whose grasp or release lands within 5 cm of the instructed target (Mode 2, Figure 3) and success (%), on all lock-in tasks.

Method	Mode 2 ↑	Success ↑
Action Sampling	20.1 ±6.6	13.5 ±4.8
Language Sampling	36.5 ±5.0	20.1 ±4.5
FOREWARN	31.7 ±5.9	19.1 ±4.9
VLS	33.7 ±7.7	15.9 ±6.0
DISCERN	**37.4** ±6.9	**29.6** ±6.7

Table 4: Ablation studies. We compare variants of DISCERN without closed-loop previews, repeated decision making, and rephrase retention. Every row uses the same policy, initial states, candidates and verifier.

Method	Task	Swap	Avg.
w/o closed-loop (open-loop preview)	17.5	13.5	15.5
w/o repeated decisions (step 0 only)	20.0	11.5	15.8
w/o lock (may switch again)	23.0	**14.5**	18.7
DISCERN (ours)	**23.5**	**14.5**	**19.0**



(a) Ranking agreement **(b)** Spread vs. error **(c)** Recoverable failures

Figure 4: When world-model previews are effective. (a) The Spearman correlation between ground truth and world model candidate rankings over different preview horizons. (b) Median spread between candidate futures and world-model error. (c) Share of failures that can be recovered by switching to the desired behavioral modes, relative to when the object is grasped. Wrong-pick failures must be corrected before the grasp, while wrong-place failures remain fixable while the object is carried.

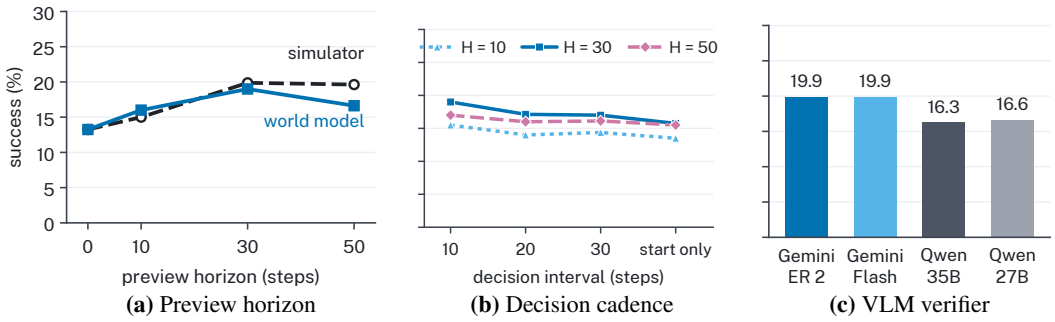
locations due to premature guidance release. Thus, broader action coverage does not necessarily translate to greater recoverability of successful behavior.

5.4 When World Model Previews are Effective (RQ3)

We next evaluate when world-model previews provide a useful selection signal. We hypothesize that previews support selection only when two conditions are satisfied: (1) the world model's behavior prediction preserves the ground truth ordering of language-candidate outcomes, and (2) the language-candidate behaviors diverge sufficiently for their differences to outweigh prediction error.

In Figure 4, we examine these conditions by comparing rankings between simulation ground truth and world model previews. We observe that, at short preview horizons, the two sets of rankings are highly correlated, with Spearman $\rho = 0.64$ at 30 steps (Figure 4a). However, as the horizon increases, prediction error begins to dominate candidate spread (Figure 4b), making the previews less informative for distinguishing among candidate behaviors. As a result, the choice of preview horizon creates a tradeoff between distinguishability of candidate actions and reliability of world model previews. Longer previews can reveal larger behavior differences, but at the cost of predictive drift. Qualitative examples of the world model predictions against the ground truth are shown in Appendices B.2 and B.3.

We observe that this relationship imposes an additional temporal constraint on when world model predictions can support successful verifier intervention: the candidate behaviors must become distinguishable while switching away from the canonical instruction can still recover the desired outcome. The timing analysis in Figure 4c illustrates this limited window of opportunity. Wrong-pick failures must be corrected early, so that that another instruction can steer the policy toward the desired object before the grasp takes place. Once the policy commits to picking up an incorrect object, it becomes very difficult for a corrected instruction to recover from the mistake. Wrong-place failures, however, remain mostly recoverable throughout the object pick phase and require fast correction after



(a) Preview horizon **(b)** Decision cadence **(c)** VLM verifier

Figure 5: Key design choices. (a) Effect of preview horizon H over simulator and world-model previews. (b) Effect of how often we make decisions during deployment. (c) Effect of the VLM verifier. Appendix B.4 ablates the split between instruction and action-sample candidates.

the grasp. These results show that world model previews must also identify the desired behavior quickly, while the relevant candidate behaviors may still be only partially expressed.

5.5 Ablation study over the design choices (RQ4)

We now evaluate ablated variants of DISCERN to examine how the established constraints on recoverability and separability influence its design choices. Table 4 shows that closed-loop previews and repeated decisions both improve selection, indicating that candidate differences must be tracked as they emerge and acted on while the episode remains recoverable. The horizon study in Figure 5 further supports the separation-error tradeoff of world-model previews. Short previews end before candidate behaviors become distinguishable, while longer previews accumulate enough prediction error to cancel those differences; simulator previews do not exhibit the same long-horizon degradation. We also find that open-weight verifiers underperform proprietary frontier models, suggesting that selection requires sufficient visuospatial and physical reasoning to interpret the task-relevant differences exposed by predicted futures. Together, these results show that effective behavioral search depends on matching preview depth, decision timing, and verifier capability to the distinctions that the world model can reliably expose.

6 Discussion and Conclusion

We introduced DISCERN, a framework for test-time optimization of frozen policies through language-conditioned behavioral search and closed-loop world-model evaluation. The results show that language proposals and world-model evaluation play complementary roles in steering a frozen policy. Rephrasing the instruction exposes successful behaviors that action sampling misses, while evaluating their predicted outcomes helps determine which instruction to execute. In practice, we found that effective selection depends on predicting far enough ahead for candidate behaviors to diverge, while keeping the closed-loop preview short enough for the world model to predict those differences reliably.

These findings suggest a path toward more reliable test-time alignment between robot behavior and human intent: language specifies the desired behavior, while predicted outcomes showcase how the policy would act on that specification before execution. Although we instantiate DISCERN with language-conditioned VLAs, the same search principle could extend to other control variables that expose distinct policy behaviors with predictable consequences, assuming the policy is steerable with the selected control variables. Applying it to agentic systems that directly issue executable robot commands is a promising direction for future work.

Limitations. First, DISCERN can only recover failures for which its candidate instructions can elicit a successful behavior; rephrasing cannot compensate for capabilities absent from the frozen policy. Second, world-model previews and VLM verification introduce nontrivial inference latency, limiting how frequently the system can reconsider its instruction during execution. Finally, our evaluation covers a single policy in simulation. Whether these gains transfer to physical robots and other VLA policies remains to be established.

AI USE STATEMENT

We used LLM coding assistants for implementation (porting baselines to our evaluation harness, evaluation and analysis code), feedback on experiment design, and iterations on results interpretation. We did not use them to generate or clean datasets or for translation; mathematical claims and proofs are not applicable to this work. We additionally used them to write figure code, search related work, draft and edit parts of the paper. We have reviewed all AI-assisted work and take responsibility for the final content of this paper.

REPRODUCIBILITY STATEMENT

All components of DISCERN are publicly available: the frozen policy ($\pi_{0.5}$, RLinf LIBERO SFT checkpoint), the world model (A2World, LIBERO checkpoint), the benchmark (LIBERO-PRO), and the proposer and verifier (Gemini Robotics ER 2, accessed through its public API). Sec. 4 specifies the method, and Sec. 5.1 specifies the tasks and the evaluation protocol. Appendix A.1 lists the full configuration of DISCERN (Table 5), Appendix A.2 describes how each baseline was configured and where it departs from its released code (Table 6), and Appendix C gives the complete proposer and verifier prompts.

REFERENCES

Jagdeep Singh Bhatia, Andrew Wagenmaker, William Chen, and Sergey Levine. Adapting generalist robot policies with semantic reinforcement learning, 2026. URL https://arxiv.org/abs/2606.31958.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael 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, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2026. URL https://arxiv.org/abs/2410.24164.

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, Xiang Li, Quanlu Zhang, Zhaofei Yu, Guoliang Fan, Tiejun Huang, Yu Wang, and Chao Yu. π_{RL}: Online rl fine-tuning for flow-based vision-language-action models, 2026a. URL https://arxiv.org/abs/2510.25889.

William Chen, Jagdeep Singh Bhatia, Catherine Glossop, Nikhil Mathihalli, Ria Doshi, Andy Tang, Danny Driess, Karl Pertsch, and Sergey Levine. Steerable vision-language-action policies for embodied reasoning and hierarchical control, 2026b. URL https://arxiv.org/abs/2602.13193.

Hanchen Cui, Sergio Arnaud, Arjun Majumdar, Daniel Dugas, Elie Aljalbout, Karthik Desingh, Krishna Murthy Jatavallabhula, and Franziska Meier. Dreamsteer: Latent world models can steer vla policies during deployment without any finetuning, 2026. URL https://arxiv.org/abs/2607.02865.

Danny Driess, Jost Tobias Springenberg, Brian Ichter, Lili Yu, Adrian Li-Bell, Karl Pertsch, Allen Z. Ren, Homer Walke, Quan Vuong, Lucy Xiaoyang Shi, and Sergey Levine. Knowledge insulating vision-language-action models: Train fast, run fast, generalize better, 2025. URL https://arxiv.org/abs/2505.23705.

Maximilian Du and Shuran Song. Dynaguide: Steering diffusion polices with active dynamic guidance, 2025. URL https://arxiv.org/abs/2506.13922.

Suning Huang, Jiaqi Shao, Ke Wang, Qianzhong Chen, Jiankai Sun, Yanjiang Guo, Mac Schwager, and Jeannette Bohg. Breaking lock-in: Preserving steerability under low-data vla post-training, 2026a. URL https://arxiv.org/abs/2604.23121.

Ze Huang, Jiahui Zhang, Hairuo Liu, Chenxi Zhang, Ran Cheng, and Li Zhang. Learning transferable dynamics priors from action to world modeling, 2026b. URL https://arxiv.org/abs/2606.29501.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

Hyun Joe Jeong, Gokul Swamy, and Andrea Bajcsy. Learning what to say to your vla: Mostly harmless vision language action model steering, 2026. URL https://arxiv.org/abs/2606.12299.

Sunshine Jiang, John Marangola, David Zhang, Raghuram Kowdeed, Ruiyang Luo, Nitish Dashora, Richard Li, Pulkit Agrawal, and Zhang-Wei Hong. Prompt-driven exploration: Language as an exploration space for vla reinforcement learning, 2026. URL https://arxiv.org/abs/2607.08837.

Bosung Kim, Ruiyi Wang, David Acuna, Jaehun Jung, Alexander Trevithick, Brandon Cui, Yejin Choi, and Prithviraj Ammanabrolu. How to instruct your robot: Dense language annotations power robot policy learning, 2026. URL https://arxiv.org/abs/2605.17077.

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

Jacky Kwok, Christopher Agia, Rohan Sinha, Matt Foutter, Shulu Li, Ion Stoica, Azalia Mirhoseini, and Marco Pavone. Robomonkey: Scaling test-time sampling and verification for vision-language-action models, 2025. URL https://arxiv.org/abs/2506.17811.

Jacky Kwok, Shulu Li, Pranav Atreya, Yuejiang Liu, Yixing Jiang, Chelsea Finn, Marco Pavone, Ion Stoica, and Azalia Mirhoseini. Llm-as-a-verifier: A general-purpose verification framework, 2026a. URL https://arxiv.org/abs/2607.05391.

Jacky Kwok, Xilun Zhang, Mengdi Xu, Yuejiang Liu, Azalia Mirhoseini, Chelsea Finn, and Marco Pavone. Scaling verification can be more effective than scaling policy learning for vision-language-action alignment, 2026b. URL https://arxiv.org/abs/2602.12281.

Zhuo Li, Junjia Liu, Zhipeng Dong, Tao Teng, Quentin Rouxel, Darwin Caldwell, and Fei Chen. Towards deploying vla without fine-tuning: Plug-and-play inference-time vla policy steering via embodied evolutionary diffusion, 2026. URL https://arxiv.org/abs/2511.14178.

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

Shuo Liu, Ishneet Sukhvinder Singh, Yiqing Xu, Jiafei Duan, and Ranjay Krishna. Vls: Steering pretrained robot policies via vision-language models, 2026. URL https://arxiv.org/abs/2602.03973.

Alberta Longhini, David Emukpere, Jean-Michel Renders, and Seungsu Kim. Behavioral mode discovery for fine-tuning multimodal generative policies, 2026. URL https://arxiv.org/abs/2605.11387.

Mitsuhiko Nakamoto, Oier Mees, Aviral Kumar, and Sergey Levine. Steering your generalists: Improving robotic foundation models via value guidance, 2025. URL https://arxiv.org/abs/2410.13816.

NVIDIA, :, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu,

Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025. URL https://arxiv.org/abs/2503.14734.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models, 2025. URL https://arxiv.org/abs/2501.09747.

Junwon Seo, Sushant Veer, Ran Tian, Wenhao Ding, Apoorva Sharma, Karen Leung, Edward Schmerling, Marco Pavone, and Andrea Bajcsy. Stressdream: Steering video world models for robust policy evaluation and improvement, 2026. URL https://arxiv.org/abs/2606.00267.

Ameesh Shah, William Chen, Adwait Godbole, Federico Mora, Sanjit A. Seshia, and Sergey Levine. Learning affordances at inference-time for vision-language-action models, 2025. URL https://arxiv.org/abs/2510.19752.

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

Derek Ming Siang Tan, Shailesh Shailesh, Srikrishna Iyer, William Wei Jie Teo, Yuanliang Ju, Qiao Gu, and Guillaume Sartoretti. Rl^2-vla: Adaptive rl latent compositional steering with test-time scaling for vision-language-action models, 2026. URL https://arxiv.org/abs/2607.26991.

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, 2025. URL https://arxiv.org/abs/2506.15799.

Yilin Wu, Ran Tian, Gokul Swamy, and Andrea Bajcsy. From foresight to forethought: Vlm-in-the-loop policy steering via latent alignment, 2025. URL https://arxiv.org/abs/2502.01828.

Yilin Wu, Anqi Li, Tucker Hermans, Fabio Ramos, Andrea Bajcsy, and Claudia Pérez-D'Arpino. Do what you say: Steering vision-language-action models via runtime reasoning-action alignment verification, 2026. URL https://arxiv.org/abs/2510.16281.

Sherry Yang. World models as an intermediary between agents and the real world, 2026. URL https://arxiv.org/abs/2602.00785.

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, 2026. URL https://arxiv.org/abs/2510.03827.

A Implementation Details

We provide the configuration of DISCERN and each baseline in Appendices A.1 and A.2, additional results to our experiments in Appendix B, and the prompts used to query the VLM proposer and verifier in Appendix C.

A.1 Configurations

Table 5: Configuration of DISCERN. We Policy: $\pi_{0.5}$ with the RLinf LIBERO SFT checkpoint (Chen et al., 2026a); world model: A2World, LIBERO checkpoint (Huang et al., 2026b); tasks: LIBERO-PRO (Zhou et al., 2026), scored by the LIBERO success predicate (Liu et al., 2023).

Policy (frozen, all methods)		***World-model preview***	
Checkpoint	$\pi_{0.5}$, RLinf LIBERO SFT	Model	A2World, LIBERO
Camera views	third-person, wrist, 256^2	Views	third-person, wrist, 256^2
Proprioception	none	Horizon H	30 steps, 3 policy queries
Action chunk	50 generated, 10 executed	Sampling steps	16 (release default 35)
Denoising steps	3	Guidance	none
Settle steps	10 (LIBERO-10: 20)	Context	first frame, last 4 frames
Step cap	280 / 280 / 300 / 520 [a]	Batching	8 candidates, 3 passes of 10
Candidate instructions		***Verifier***	
Candidates K	8: canonical and 7 others	Model	Gemini Robotics ER 2
Proposer	Gemini Robotics ER 2	Sampling	temperature 0, one sample
Queried	once per task, before rollouts	Thinking level	medium
Sampling	temperature 1.0, thinking off	Input	one image, 8 columns $\times$ 2 views
Input	instruction, state-0 image	Per view	final previewed frame, 256 px
Kept	first 7 distinct of 36	Column order	current first, others shuffled
Evaluation		***Decision rule***	
Tasks	80 (task and swap)	Decision steps	every 10, from step 0
Lock-in tasks	23	Hold	column 1 or unparsed reply
Initial states	10 per task (0 to 9)	Commitment	first switch locks

[a] LIBERO-Spatial / Object / Goal / 10.

A.2 Baseline Implementations

Baselines share the policy, tasks and initial states of Table 5. Those that sample over instructions share the same rephrases. Since no previous method releases components trained for our setting, we re-implement each of the baselines on LIBERO data with the authors' hyperparameters (Table 6).

CoVer. We train its contrastive verifier on the 40 LIBERO tasks, whose scenes LIBERO-PRO perturbs. The original paper's batch of 32,768 exceeds our corpus (30,551 frames). We find that a batch of 2,048 holds each each task around 50 times.

FOREWARN. We use the generated action chunks of our policy as candidates, and execute decisions according to our commitment rule, since the released code re-plans at defined task-specific phase ends. We retrain its latent world model and learned latent-text alignment narrator on rollouts of the 40 unperturbed LIBERO tasks, using behavioral labels from the simulator for evaluation.

RL^2-VLA. We retrain QAM on the policy's own LIBERO demonstrations, setting the horizon to the length of the executed chunk (10 actions) and SAFE on 1,200 rollouts of the 40 unperturbed tasks. We choose its conformal level α based on a held-out validation split.

VLS. To find keypoints, we use Gemini 3.8 Flash to name task objects, Grounding-DINO to box them, SAM to mask them, and their released keypoint detector to propose keypoints from depth-projected masks. We also use use RAFT optical flow to track objects. Using this perception pipeline, we find that masks reach a median IoU of 0.97 and tracked keypoints reach a median error of around 0.02 cm. Since VLS's gradient acts in normalized action space, we scale the guidance and diversity weights by the ratio of action-normalization ranges between its checkpoint.

Table 6: Baseline configurations, with hyperparameters named as in CoVer (Kwok et al., 2026b), VLS (Liu et al., 2026), FOREWARN (Wu et al., 2025) and RL^2-VLA (Tan et al., 2026). *Paper* is the original value (RL^2-VLA: its π_0 setting; † released code, where the paper gives none; – not reported).

CoVer			***VLS***		
	ours	*paper*		*ours*	*paper*
Rephrases K	8	8	Particles K	20	10
Action samples M	5	5	Denoising steps	50 (10 unguided)	10†
Verifier ensemble	3	3	Guidance scale	30	80†
Scored actions	first 5 of 10	–	Diversity scale	10	20†
Backbone	SigLIP2 ViT-L/16	SigLIP2 ViT-L/16	Executed actions	particle 0, first 10	particle 0, first 10
Camera views	2	–	Reward program	one per episode	one per episode
Training data	LIBERO	BridgeV2	Program VLM	Gemini 3.8 Flash	GPT-5.1†
Demos / frames	1,631 / 30,551	–	Stage VLM	Gemini 3.8 Flash	Gemini 3 Flash†
Instructions / task	13	16	Stage queries	uncapped	50†
Batch size	2,048	32,768	Keypoints	perceived	simulator†
Training steps	2k	2k	Prompt rules	generic	task-specific†
FOREWARN			***RL^2-VLA***		
	ours	*paper*		*ours*	*paper*
Candidate plans	8	6 of 100	$K \times M$	8×5	8×5
Plan horizon	50	64	Weight w (clipped)	$\mathcal{N}(0.5, 0.25)$	$\mathcal{N}(0.5, 0.25)$
Imagined latents	16	16	Embedding dim.	1024	1024
Decision steps	0, 10, 20, 30	phase ends	QAM batch / LR	256 / 3e−4	256 / 3e−4
Commitment	first switch	–	QAM critics	10	10
Narration classes	9, simulator	3, hand	QAM flow steps	10	10
Task condition	generic	per task	QAM horizon	10	4
World model (64×64)	DreamerV3	DreamerV3	QAM inverse temp.	0.1	0.1
Batch × length	16 × 64	16 × 64	QAM steps	500k	500k
Demos per task	50	100	QAM data	LIBERO	BridgeV2
Rollouts per task	250 (training)	250 (deployed)	SAFE LR / λ_{reg}	1e−3 / 0.1	1e−3 / 0.1
WM steps	100k	100k–150k	SAFE batch / epochs	512 / 2k	512 / 2k
Narrator (LoRA)	Llama-3.2-11B	Llama-3.2-11B	SAFE rollouts/task	30	300
LoRA rank / α	8 / 32	8 / 32	SAFE split	60/20/20	60/40
LoRA batch / LR	24 / 1e−4	10 / 1e−4	CP level α	0.05	best of top 3
LoRA training	6k steps	10–15 epochs	Selection loop	released	released

B Additional Results

B.1 Recoverability of Successful Behavioral Modes on More Lock-In Tasks

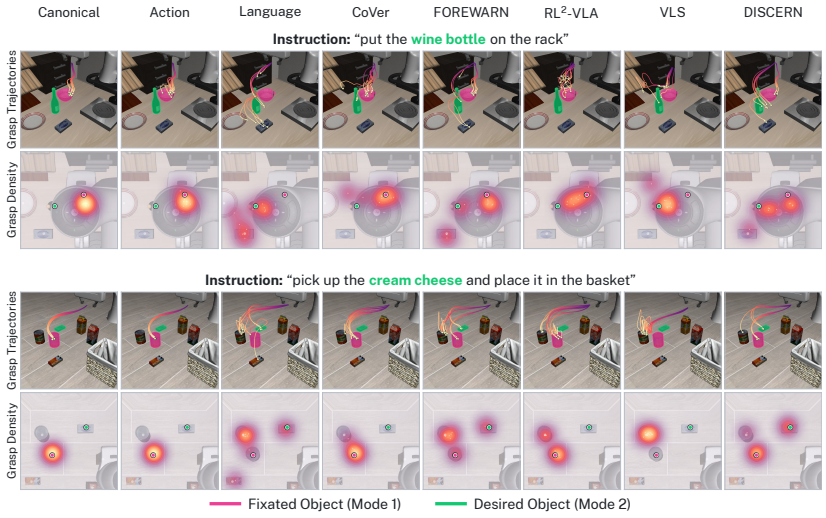


Figure 6: Behavioral modes on more lock-in tasks. Rows as in Figure 3: grasp trajectories (top) and grasp density (bottom); pink marks the fixated object, green the desired one. Action stays with the canonical grasps, while language reaches the desired objects.



Figure 7: Behavioral modes on more lock-in tasks (continued from Figure 6).

B.2 World-Model Previews

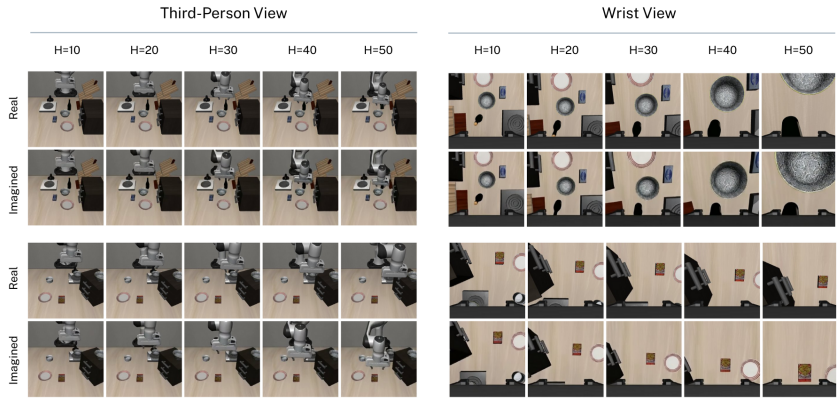


Figure 8: World-model previews against simulator futures, At short horizons, we observe close alignment between world model predictions and the ground truth. As horizon length increase, the previews begin to drift.

B.3 Separating Candidates with Previews

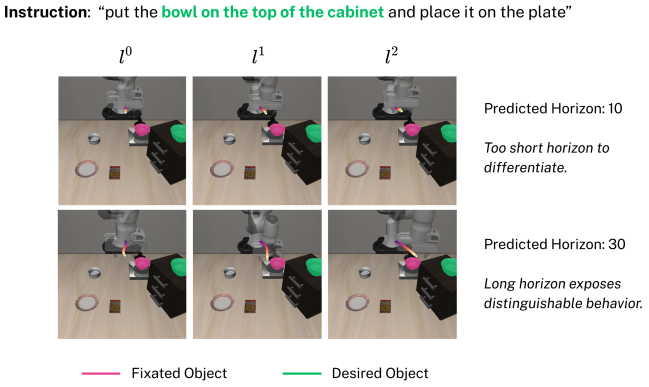


Figure 9: Longer-horizon previews expose distinguishable behaviors. Third-person views of three candidate instructions at. The language-candidates are behaviorally indistinguishable at $H{=}10$ and begin to separate at $H{=}30$. ℓ^0 and ℓ^1 fixate on the bowl on the top of the stove, while ℓ^2 successfully steers towards the desired bowl on the top of the cabinet.

B.4 Number of Instructions and Action Samples

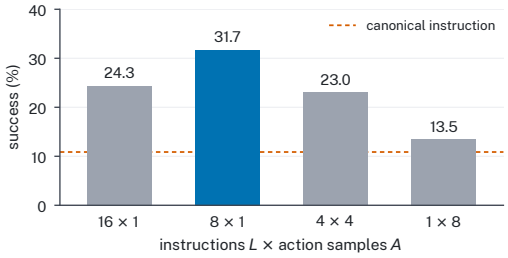


Figure 10: Number of instructions and action samples. Success on the lock-in tasks when the candidates are L instructions $\times$ A action samples. Action samples explore only locally and add little over the canonical instruction (1×8); trading instructions for action samples (4×4) loses behavioral diversity, and doubling the instructions (16×1) makes selection harder.

C SYSTEM PROMPTS

VLM Proposer (Gemini Robotics ER 2)

You are helping a robot succeed at a manipulation task. You will see the scene and the instruction the robot was given. Write alternative instructions for the SAME physical goal. You may describe the target object however it actually looks in the image -- by colour, material, position, or the kind of appliance it is -- rather than reusing the noun in the original instruction. Every alternative must still command the same physical outcome.

The robot was given: "{instruction}". The image shows the initial scene, overhead view on top and wrist view below.

Write {k} alternative instructions for the same physical goal. Vary how you NAME the target object as well as how you phrase the action: name it by colour, by material, by where it sits, or by what kind of thing it is, as the image supports. Do not feel obliged to reuse any noun from the original instruction. Keep each instruction short and imperative, and make sure it commands the same physical outcome.

Reply with exactly {k} lines, one instruction per line, numbered "1." to "{k}". No preamble, no commentary.

VLM Verifier (Gemini Robotics ER 2)

This is a simulated LIBERO benchmark scene (a Franka arm on a tabletop with household objects). The robot's task: {task}. You see {K} columns; each is a different possible future from the SAME current state, as the third-person camera (top row) and the wrist camera (bottom row) at the end of that future.

Work in three steps. Step 1, identify the target: say what the target object named in the task looks like (colour, shape, any label) and where it is on the table right now, using the third-person row; the same object must be at the same table location in every column that has not moved it, so cross-check columns against each other and do not assume the target is where a similar-looking object is.

Step 2, for every column, say where the gripper is heading relative to the target: the third-person row gives the global direction of the arm (left/right, forward/back), the wrist row looks down along the gripper: the two dark finger tips are at the bottom edge, whatever sits between them at the bottom-centre is what the gripper is over, and anything higher up in the wrist frame is AHEAD of the gripper, further away. Apparent size is NOT a heading: every object grows in the wrist view as the arm descends, so a target that looks larger but has moved UP the frame while a neighbouring object fills the bottom between the fingers means the gripper is descending onto the neighbour, not the target. Name, per column, the object between the finger tips; that is the column's heading, whatever else is visible. Before grouping, check whether the columns differ materially at all: if the arm pose in the third-person row is the same in every column and the wrist rows differ only by zoom, the futures are indistinguishable, they form ONE group, and the answer is 1. Columns that share a heading form a group; name the groups. Be sceptical of 'already holding the target' unless BOTH rows agree and the target has left its table location.

Step 3, decide. If column 1 is among the columns heading to the target, answer 1: switching only pays when column 1 is heading somewhere else. Progress beats neatness: a column whose gripper is already low at the target, with the target large or partly cut off between the finger tips, is FURTHER ALONG than a column that still shows the target small and neatly centred from above; a clean centred view is not a better heading. Only when column 1 is over the wrong object or empty table, choose the column whose gripper is heading most directly toward the target with a plausible approach for grasping it. If no column is clearly heading toward the target, answer 1.

Reply with the column number alone on the first line, then exactly three short lines: 'target: <what it looks like, where it is>', 'headings: <groups of columns and the object each gripper is centred over>', 'why: <one sentence>'.