CORRECTING STRATEGY SELECTION IN GENERALIST ROBOT POLICIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Large-scale robot datasets record multiple *strategies*, or ways of completing the same task: a microwave can be opened by pulling its handle or by pressing its release button, and a lid can be taken off by lifting its knob or by pushing its rim. A given object, however, often affords only one of them. We find that generalist policies trained on such data can carry out an *unsupported* strategy, one the object does not afford, such as reaching for a handle on a microwave that does not have one. The training data demonstrate the requested strategy about as often as or more often than the strategy the policy carries out, and stating the requested strategy in the instruction does not fix the failure. We therefore ask what drives the selection, and identify two features of the input that matter: the color of the object and the object nouns in the instruction. Recoloring the object to the most common color of the objects on which the training data demonstrate the requested strategy, and deleting object nouns that draw the policy's attention away from the part this strategy acts on, both shift the policy toward the requested strategy. We turn these findings into a method that decides for each input whether and how to edit it, without retraining the policy. Across three simulated tasks, the edits raise the average success of $\pi_{0.5}$-DROID from 8% to 57%. Same edits also improve two other DROID-trained policies.

1 INTRODUCTION

Large-scale robot datasets such as DROID (Khazatsky et al., 2024) record multiple *strategies*, or ways of completing the same task: operators open a microwave by pulling its handle in some episodes and by pressing a release button in others, and take a lid off by lifting its knob in some and by pushing its rim in others. Such variety arises naturally, since the demonstrations come from many operators across many labs, each completing a task in their own way. At test time, however, a particular object often affords only one of these strategies: a microwave without a handle opens only when its button is pressed, and a lid set into its box comes off only when its knob is lifted. We find that a policy trained on such multi-strategy data can ignore the object's affordance and carry out an *infeasible* strategy, one the object does not afford. On the handleless microwave, the policy reaches for the place on the door where a handle would be; on the box, it pushes along the rim of the lid; and on a rice cooker whose lid lifts only by a handle, it presses a round mark on the lid that looks like a button (Figure 1).

Clarifying the requested strategy in the instruction, as in *open the microwave by pressing the release button*, does not reliably fix this failure: generalist policies have been shown to often fail to follow the instruction as intended (Fei et al., 2026; Zhou et al., 2025) Indeed, in our own experiments in simulation, on a button-only-microwave, $\pi_{0.5}$-DROID reaches for the missing handle in 23 of 30 rollouts, and naming the button press in the instruction only lowers this number to 20 (Section 5.2). To use such a generalist policy reliably in the real world, a deployer therefore needs another way to steer it toward the strategy the object affords.

Most existing works steer a trained policy either by intervening in how it generates actions (Wang et al., 2025a; Du & Song, 2025; Wagenmaker et al., 2025; Miao et al., 2026) or by editing its inputs. Most of the former require a steering signal, and the latter have so far edited only what surrounds the task object (Hancock et al., 2025; Song et al., 2026) or how the task is worded (Jeong et al., 2026). More fundamentally, none of these methods asks why the policy selects a strategy the object does

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.

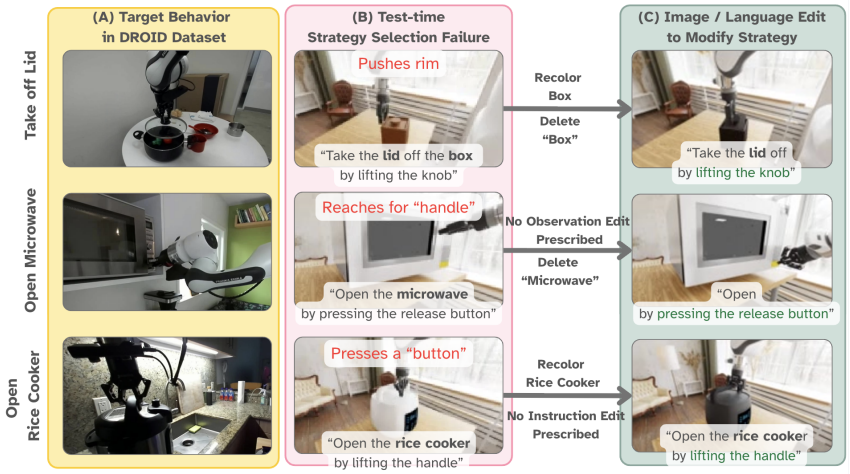

Figure 1: **A generalist policy trained on data with multiple strategies per task can carry out a strategy the object does not afford, and editing the object's appearance or the instruction turns it to the requested one.** Left: DROID episodes that demonstrate the requested strategy. Middle: on a test object that affords only that strategy, $\pi_{0.5}$-DROID carries out an unsupported one. Right: under edits prescribed by our method, the same policy carries out the requested strategy on the same object.

not afford, reaching for a handle that is not there on a microwave that opens only by its button. We tackle this question.

The natural first answer, that the policy overfits to whichever strategy the training data demonstrates most often, does not hold: DROID demonstrates the button press on microwaves about as often as the handle pull (54 against 57 episodes). Nor is the policy missing the ability to perform the requested strategy: on a pot similar to those on which DROID demonstrates the knob lift, $\pi_{0.5}$-DROID lifts the knob in 29 of 30 rollouts (Appendix A).

Instead, we find that two features of the input change which strategy the policy selects. First, the color of the object alone can shift the policy toward the requested strategy: most knob-lift episodes involve black pots, and simply recoloring a brown wooden box to black, without changing its shape or material, shifts the policy from pushing the rim toward lifting the knob (Section 5.4). Second, we find that some object nouns in the instruction carry a prior about where to act: with *microwave* in the instruction, the policy's attention concentrates on the door edge where a handle would be; deleting the noun moves the attention onto the button and raises success (Section 5.5).

Building on these findings, we propose a steering method that decides for each input channel whether and how to edit it, without retraining the policy. The observation edit recolors the test object to the color most common among the objects on which the training data demonstrate that strategy, unless its color is already typical of them. The instruction edit deletes each object noun whose deletion raises the policy's attention on the part the requested strategy acts on. The prescribed edits raise the success of $\pi_{0.5}$-DROID from 0 to 15 of 30 for taking the lid off a box, from 3 to 12 for opening a microwave and from 4 to 24 for opening a rice cooker (Section 5.3). The same edits also raise success for MolmoAct2-DROID (Fang et al., 2026a) and Cosmos3-Nano-Policy-DROID (NVIDIA, 2026), by amounts that differ by task (Section 5.6).

Our contributions are as follows.

- We identify a strategy selection failure in policies trained on multi-strategy data: on objects that afford only one strategy, the policy carries out an unsupported one. How often each strategy appears in the training data does not explain the failure, and stating the strategy in the instruction does not fix it (Section 5.2).
- We find two features of the input that change which strategy the policy selects: the color of the object and the object nouns in the instruction (Sections 5.4 and 5.5).

- We turn the two findings into a test-time steering method that prescribes input edits without retraining the policy. On all three tasks, the edits raise the success of three DROID-trained policies, by amounts that differ across policies and tasks (Sections 4, 5.3 and 5.6).

2 Related Work

Strategy selection in robot policies. Several works study how a robot manipulation policy selects among multiple strategies demonstrated for one task. Wang et al. (2025b) study objects whose correct strategy cannot be read from the image, such as a microwave whose door opens directly when unlocked and needs a button press when locked; since the lock state is invisible, their policy learns from demonstrated recoveries to try one strategy and switch after a failure. In our case the test object only supports one strategy, and the strategy can be read from the image, but the policy still selects the unsupported one. Fang et al. (2026b) report a related failure: a VLA executes the behavior it associates with the scene and disregards the instruction. In their setting the instruction changes and the scene stays fixed, and the error is which object is acted on. In ours the error is which part of the object is acted on. Other works also report failures of language conditioning in VLA policies (**??**)., and related failures have been linked to limited diversity within sub-datasets and fragmentation across them (Xing et al., 2025), as well as to language varying far less than vision (Xu et al., 2025). Instead of explaining the behavior by properties of the training data or of the model, we identify two features of the input at test time, the object's color and the object noun in the instruction, that change which strategy the policy selects. A recent work Zhao et al. (2026) also notes that committing to the wrong action mode can produce actions the scene does not support. They make action modes explicit with a vector-quantized action representation and steer mode selection during inference. We instead steer a given trained policy through its inputs, without changing its action representation.

Steering within the policy. Most methods that steer a trained policy at deployment act on how it generates actions. They guide the denoising of a diffusion policy toward a target such as a user sketch or a goal image (Wang et al., 2025a; Du & Song, 2025), choose the initial noise the policy starts from, with a noise policy trained from reward or by running a coarse action backward through a flow-matching policy (Wagenmaker et al., 2025; Tang et al., 2026), rank sampled actions with a verifier (Kwok et al., 2026; Wu et al., 2026; 2025), or edit hidden activations or weights (Miao et al., 2026). Each of these methods requires a steering signal: a target given in a form other than the instruction, such as a sketch, a goal image or a coarse action; a component trained on additional data or rollouts; or a verifier queried during execution. Our method derives the edits from the training data and the policy's attention, which does not require any rollouts or training.

Steering through the inputs. Methods that act only on the inputs edit the observation or the instruction. On the observation side, existing works focus on using run-time image editing methods to removes or replaces what the policy should ignore: For example, Hancock et al. (2025) alter task-irrelevant regions to which a VLA is sensitive, Song et al. (2026) inpaint distractors identified from the instruction, and Chen et al. (2024) paint the robot seen in training over an unseen one. None of these edits changes the task object, and none has been shown to change the strategy applied to it. On the instruction side, Jeong et al. (2026) and Kwok et al. (2026) both search over instruction wordings scored by downstream behavior, using closed-loop rollouts and a trained verifier, respectively. Their search resembles automatic prompt optimization for language models (Zhou et al., 2023; Yang et al., 2024; Prasad et al., 2023). Such a search adopts whatever wording scores best and gives no account of why it works. We delete a noun only for a measured reason, its effect on the policy's attention.

3 Problem Formulation

We consider a trained policy $\pi(a \mid o, l)$ that maps an observation o and a language instruction l to actions. A *task* τ specifies the outcome the robot must achieve, such as an open microwave door. We name a task by the object whose state changes, so taking a lid off a pot and taking a lid off a box are the same task. The training data often demonstrate a task with more than one *strategy* $\sigma \in \Sigma_\tau$, a way of achieving the outcome that is determined by which part of the object the robot makes contact and what motion it applies there, such as opening a microwave by pressing its release button. An instruction can word the same task and strategy in many ways.

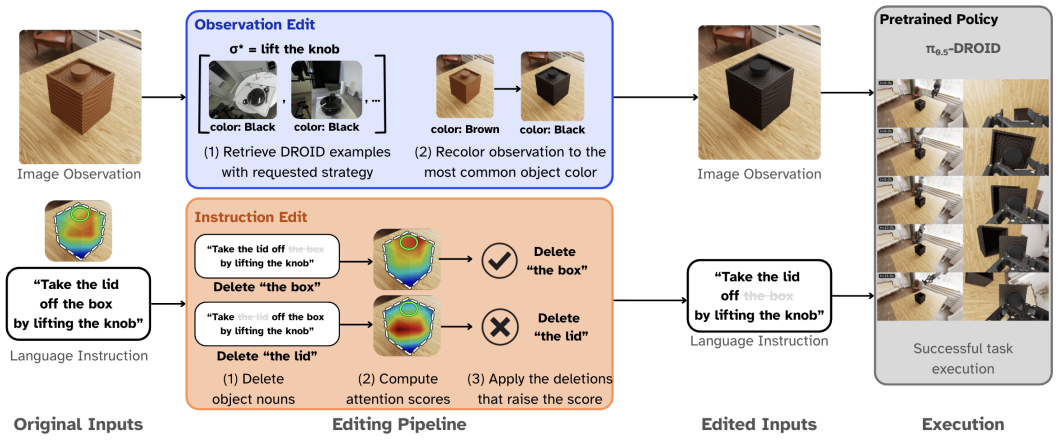

Figure 2: **Overview of our method, demonstrated on one of our tasks.** Our method is a procedure that decides, for the observation channel and the instruction channel, whether and how to edit each. The observation edit retrieves the training episodes that demonstrate the requested strategy $\sigma^\star$ and recolors the test object to the most common color of their objects, unless its own color is already typical of those objects. The instruction edit checks whether each object noun should be deleted by deleting one object noun at a time, pooling the attention-score change over held-out poses, and applying the deletions that raise the pooled score.

At test time, the robot faces an object that affords only one strategy of the task, the *requested strategy* $\sigma^\star$, and we call every other strategy of the task *unsupported*. We assume that both τ and $\sigma^\star$ are given in l; this assumption is mild, since which strategy an object affords is usually evident from the observation (Appendix B). Given π, o and l, our goal is to construct an edited pair (o', l') under which π carries out $\sigma^\star$. We do not modify the weights of π, and we prescribe (o', l') before any rollout. We assume access to the data the policy was trained on.

4 Method

To return (o', l') under which π carries out $\sigma^\star$, our method is a procedure that decides, for each input channel, whether and how to edit it. Each decision builds on one of the two findings introduced in Section 1 and tested in Sections 5.4 and 5.5. On the observation side, we recolor the test object to the most common color among the objects on which the training data demonstrates $\sigma^\star$, unless its own color is already typical of them (Section 4.1). On the instruction side, we check for each object noun whether deleting it raises the policy's attention on the part $\sigma^\star$ acts on, and delete the nouns that have this effect (Section 4.2). Figure 2 illustrates the procedure. We use the task *take the lid off the box* (pick-place(lid)) as a running example in Figure 2 and throughout this section.

4.1 Observation Edit

Retrieving demonstrations of the requested strategy. To recolor the test object toward the objects on which the training data demonstrates $\sigma^\star$, we first retrieve the corresponding demonstrations. In order to do so, we build an index of the whole training data offline: Following the definitions in Section 3, we assign each training data a task (e.g., take the lid off) and then label every episode of a task with one of its strategies (e.g., take the lid off by lifting the knob); Appendix C gives the full pipeline. At test time, given the requested strategy, we simply retrieve its demonstrations from the index.

Recoloring. The goal of the observation edit is to make the test object resemble the retrieved objects without changing its geometry or physics. Our analysis of DROID's annotations shows that color is the attribute annotators most often use to tell objects apart: of the annotations that describe the object with at least one modifier, 72% include a color (Appendix C). Prior work has also found color to be among the perturbations to which manipulation policies are most sensitive (Pumacay et al.,

2024). We therefore choose to edit color, which changes how an object looks without changing how it responds to manipulation.

We first label the color of the object in each retrieved demonstration. If the annotation names a color (e.g., *take the lid off the black pot*), we take the label directly from it; otherwise, we use a vision-language model (GPT-5.5 (OpenAI, 2026)) to name the color from the video of the demonstration. We label the test object with the same model. We call a color *typical* of a set of objects if it belongs to the smallest set of labels that together cover more than half of the episodes. If the label of the test object is not typical of the retrieved objects, we recolor the test object to their most frequent label; otherwise we leave the observation unchanged.

We apply the recoloring online, to every frame of both cameras. Specifically, we first use SAM 3 (Carion et al., 2026) to track the object in each view. Completely replacing the pixels inside the mask with the target color would turn the object into a flat patch of color, obfuscating the identity of that object. We instead recolor the object in the CIELAB color space (CIE, 2019) by changing its chromaticity and mean lightness to reflect the target color, while maintaining pixel-to-pixel variation in lightness. This method encourages shading, highlights and surface texture to be similar following the edit. The implementation details can be found in Appendix D.

4.2 Instruction Edit

The instruction edit builds on our finding that some object nouns carry priors about where to act, which can draw the policy away from the part $\sigma^\star$ acts on and toward the part an unsupported strategy acts on (Section 5.5). To identify such nouns, we delete one object noun n at a time from the full instruction l (a compound such as *rice cooker* counts as one). This gives a reduced instruction $l_{\setminus n}$. For example, for $l =$ *take the lid off the box by lifting the knob*, $l_{\setminus \mathrm{box}}$ is *take the lid off by lifting the knob* and $l_{\setminus \mathrm{lid}}$ is *take off the box by lifting the knob*. We then compare $l_{\setminus n}$ with l to test n. Note that we leave the nouns in the phrase that states the strategy (e.g., *knob* in *by lifting the knob*) out of this test, since deleting them would change the strategy the instruction requests.

For each pair $(l, l_{\setminus n})$, we run two forward passes of the policy on the same scene observation, joint and gripper state and sampled noise, and read the attention that the action tokens pay to the same observation under the different instructions. Specifically, at the final denoising step of a given layer, we average the attention from the action queries to the keys of the side-view image over heads and queries, which gives one saliency map per instruction and layer (Figure 3). The check therefore needs two forward passes on the scene observation without any rollout.

We score a saliency map by how well it ranks the part $\sigma^\star$ acts on above the rest of the object, using AUC-Judd (Judd et al., 2009) with the pixels of the part as positives and the remaining pixels within the object as negatives. We average the per-layer scores over selected layers (Appendix E) to compute the overall score for each instruction (l and $l_{\setminus n}$), and delete n if $l_{\setminus n}$ scores higher.

We note that because this prior we check is a property of the noun as well as the policy rather than of a particular scene, we make the instruction edit decision once per task instead of once per scene. Specifically, we pool the score differences between l and $l_{\setminus n}$ over a set of held-out scenes, disjoint from those on which we evaluate our method, to generate l'. We then use the same l' for every rollout of the task under different scenes. Details are in Section 5.5.

5 Experiments

5.1 Setup

Dataset. In this work, we choose to work on DROID (Khazatsky et al., 2024), whose multi-collector, multi-scene nature readily gives rise to multiple strategies for the same task, making it well suited to the problem we study.

Tasks. We choose three tasks in DROID with demonstrations of more than one strategy. For each task, we build a test scene in Isaac Lab (NVIDIA, 2025) with an object that affords only one of the strategies: **Task 1: Take the lid off the box**. In this task, the lid comes off only by lifting its knob. The lid is set into the box and can only move upward, so pushing its rim does not open it. **Task 2: Open the microwave**. The microwave door has no handle and opens only when its yellow release

Table 1: **Success rates and where $\pi_{0.5}$-DROID mostly acts on, without any edit to the inputs.**

			Part the gripper mostly acts on			
Task	Instruction	Success	Requested	Unsupported	Both	Neither
Lid	task only	1/30	1/30	29/30	0/30	0/30
	task + strategy	0/30	0/30	30/30	0/30	0/30
Microwave	task only	1/30	4/30	23/30	0/30	3/30
	task + strategy	3/30	5/30	20/30	3/30	2/30
Rice cooker	task only	6/30	2/30	27/30	1/30	0/30
	task + strategy	4/30	1/30	26/30	3/30	0/30

button is pressed. **Task 3: Open the rice cooker**. The lid opens only by a handle on top. The rice cooker asset we use originally has a steam valve that looks like a button on the rice cooker's cover; we remove it, but a round mark remains, which the policy could mistake for an opening button. Appendix D describes in detail how we build the scenes, and Appendix C lists the strategies DROID contains for these tasks. In the tables in the following sections, we abbreviate the three tasks as **Lid**, **Microwave** and **Rice cooker**.

Evaluation protocol. For each task, we evaluate each (observation, instruction) pair on the same 10 initial poses of the object, with three episodes per pose, and report the number of successes out of 30. The attention check discussed in Section 4.2 uses ten separate poses of each object, disjoint from these.

Policies. The main policy we study is $\pi_{0.5}$-DROID (Black et al., 2025), one of the most widely used open generalist policies in the literature. In Section 5.6, we additionally test two policies with different architectures that are also (post-)trained on DROID, MolmoAct2-DROID (Fang et al., 2026a) and Cosmos3-Nano-Policy-DROID (NVIDIA, 2026), to check whether our findings and the proposed steering procedure hold across architectures.

5.2 Q1: Does the Generalist Policy Select the Strategy the Object Affords?

We test $\pi_{0.5}$-DROID on the three tasks with unedited observations and two unedited instructions: one that states only the task (e.g., *open the microwave*) and one that also states the strategy (e.g., *open the microwave by pressing the release button*). Under both, the policy consistently carries out the same unsupported strategy: it pushes the rim of the box instead of lifting the knob on the lid task, reaches for the edge of the microwave door where a handle would be instead of pressing the release button, and presses the round mark on the rice cooker's lid instead of lifting the handle. To quantify this failure, we use GPT-5.5 (OpenAI, 2026) to label each rollout from its video with the part that the gripper acts on for most of its duration, and one author checks these labels on a random third of the rollouts (Cohen's $\kappa = 0.91$; Appendix F). As Table 1 shows, with the task-only instruction, the policy goes for the unsupported strategy's part in 29, 23 and 27 of 30 rollouts on the lid, the microwave and the rice cooker, respectively, and succeeds only in 1, 1 and 6 of them. Stating the strategy barely changes this: 30, 20 and 26 rollouts still go for the unsupported part, and task success is 0, 3 and 4. The failure also persists when we test on different objects with varied geometry (Appendix G).

This failure cannot be predicted simply by each strategy's share in the training data. DROID demonstrates the button press and the handle pull on microwaves about equally often (54 against 57 episodes), and the handle lift and the button press on rice cookers equally often (9 episodes each). On lids, DROID demonstrates the knob lift, which our object requires, far more often than the rim push (2119 against 370 episodes), and the policy still pushes the rim. [1]

[1] The strategy classifications of the microwave and rice cooker tasks have been verified manually. The counts of episodes of different strategies for the lid task are given by our automatic DROID index (Appendix C), as the number is too large to be checked by hand.

Table 2: **What our method prescribes.**

Task	Requested strategy	Observation edit	Instruction edit
Lid	lift the knob	recolor to black	delete "box", keep "lid"
Microwave	press the release button	none	delete "microwave"
Rice cooker	lift the handle	recolor to black	keep "rice cooker"

Table 3: **Success rates of $\pi_{0.5}$-DROID with and without the prescribed edits.** Bold marks the instruction, the observation and the cell our method prescribes. *Asset* means we directly recolor the asset in the simulator, which serves as an error-free reference.

(a) Lid.

Instruction	brown (original)	**black, online**	black, asset
take the lid off the box by lifting the knob	0/30	10/30	18/30
take the lid off by lifting the knob	0/30	**15**/30	23/30

(b) Microwave.

Instruction	**original**
open the microwave by pressing the release button	3/30
open by pressing the release button	**12**/30

(c) Rice cooker.

Instruction	white (original)	**black, online**	black, asset
open the rice cooker by lifting the handle	4/30	**24**/30	24/30

5.3 Q2: Does Our Method Recover the Requested Strategy?

Table 2 lists what our method prescribes for the three tasks following the procedure described in Section 4. Two of the six prescriptions are no edit: The procedure leaves the microwave as it is, since its color is already typical of the objects on which DROID demonstrates the button press (Appendix C). Our method also keeps the instruction for the rice cooker task unchanged, as *rice cooker* is the only object noun in its instruction, and deleting it lowers the attention score on the requested strategy's part (the handle). We test both decisions by overruling them in Sections 5.4 and 5.5. In Section 5.5 we also discuss in detail how each object noun shifts the policy's attention and how this shift relates to the change in success when the noun is deleted.

Table 3 reports the success rates of $\pi_{0.5}$-DROID for each instruction and observation. Our method runs the recoloring online, and we additionally test $\pi_{0.5}$-DROID's performance when the simulator asset is directly recolored, which shows the success rate an ideal recoloring free of segmentation and tracking errors would reach.

As shown in Table 3, our prescribed edits raise success on all three tasks. On the lid, both edits contribute: while the box is wood brown, success stays at 0/30 under both instructions; recoloring alone raises it to 10/30 online and 18/30 in the asset, and deleting *box* on top of the recoloring raises it further, to 15/30 online and 23/30 in the asset. On the microwave, where the procedure prescribes no recoloring, deleting the object noun alone raises success from 3/30 to 12/30. On the rice cooker, recoloring alone raises it from 4/30 to 24/30, online and in the asset alike. We additionally run an ablation that keeps only the strategy phrase (e.g., *lift the knob*); it loses most of the effect on all three tasks (Appendix H).

5.4 Q3: Does the Object's Color Alone Change the Selected Strategy?

As Section 5.3 shows, a change of color alone is enough to shift the policy toward the requested strategy: recoloring the box and the rice cooker black, the typical color of the objects on which DROID demonstrates their requested strategies (Appendix C), raises success on both objects.

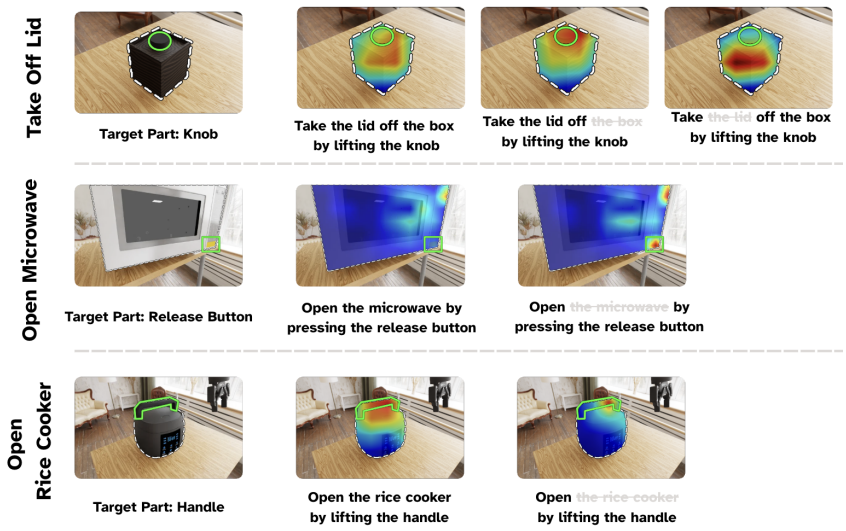

Figure 3: **Deleting one object noun moves where the policy looks.** The first image marks the part the requested strategy acts on (solid outline) and the object silhouette (dashed outline). Each map shows the attention of the action tokens at layer 11 of $\pi_{0.5}$-DROID inside the silhouette, under the full request or with one object noun deleted (struck through).

Table 5: **Success rates and attention scores when one object noun is deleted.** Brackets mark the deleted noun. Numbers are computed on scenes with recolored assets (the microwave keeps its original color). The computation of the attention score is described in Section 4.2. For every deletion, success changes in the same direction as the attention score.

		Success		Attention score	
Task	Instruction	kept	deleted	kept	deleted
Microwave	open the [microwave] by pressing the release button	3/30	12/30	0.63	0.84
Lid	take the lid off the [box] by lifting the knob	18/30	23/30	0.74	0.87
	take the [lid] off the box by lifting the knob	18/30	16/30	0.74	0.56
	take the [lid] off by lifting the knob	23/30	18/30	0.87	0.76
Rice cooker	open the [rice cooker] by lifting the handle	24/30	15/30	0.83	0.78

We test whether the target color matters in Table 4. Red, which labels 2.5% of the lid episodes and none of the rice cooker episodes, leaves both objects near their original success. On the microwave, whose color is already typical of the objects on which DROID demonstrates the button press, recoloring it black lowers success, which supports the decision to leave it unedited. The vision-language model labels the microwave white although its frame and window have other colors, so we also recolor the whole microwave white; this barely changes success, so moving an object further toward a color that is already typical brings no gain.

Table 4: **Success rates after recoloring the object in the asset**, under the instruction our method prescribes.

Task (original)	Original	Black	Red	White
Lid (brown)	0/30	23/30	0/30	–
Microwave (white)	12/30	9/30	–	11/30
Rice cooker (white)	4/30	24/30	5/30	–

5.5 Q4: Do Object Nouns Carry Priors About Where to Act?

In this section, we examine the priors of the four object nouns in our three tasks, i.e., *microwave*, *box*, *lid* and *rice cooker*, with the process discussed in Section 4.2.

Figure 3 visualizes the attention of the action tokens at layer 11 of $\pi_{0.5}$-DROID under different instructions. The attention score of Section 4.2 itself averages over layers 11 to 13; Appendix E

Table 6: **Success rates of three DROID-trained policies with the edits prescribed in Section 5.3.** Recolorings are rendered in the asset.

Task	Edit	$\pi_{0.5}$-DROID	MolmoAct2-DROID	Cosmos3-Nano-Policy-DROID
Lid	original input	0/30	11/30	11/30
	recoloring	18/30	24/30	12/30
	recoloring + noun deletion	23/30	26/30	12/30
Microwave	original input	3/30	2/30	2/30
	noun deletion	12/30	7/30	5/30
Rice cooker	original input	4/30	1/30	2/30
	recoloring	24/30	2/30	5/30

explains how we select these layers and shows that the effect is not sensitive to layer choice. As shown in Figure 3, the four nouns fall on two sides. Two of them, *microwave* and *box*, draw attention away from the requested part, and deleting them moves it back: with *microwave* in the instruction, attention concentrates on the door edge where a handle would be, and without the noun it moves onto the release button; deleting *box* concentrates attention on the knob. The other two, *lid* and *rice cooker*, draw attention toward the requested part: deleting *lid* moves attention off the knob and onto the body of the box, and deleting *rice cooker* lowers the attention on the handle.

We show the attention scores of different instructions in Table 5. As shown in the table, for all four nouns, deleting the noun changes success in the same direction as it changes the score. Deleting *microwave* quadruples success as attention moves from the door edge onto the release button. Deleting *rice cooker*, the noun our method keeps, lowers success from 24/30 to 15/30, which supports the decision to keep it. For the lid task, deleting *box* from the full request raises success and deleting *lid* lowers it, matching the decision to delete the first and keep the second. Although our method needs only these two comparisons on the full request, we additionally compare *take the lid off by lifting the knob* with *take off by lifting the knob*, because deleting *lid* from the full request also changes what the verb takes as its object. The score and success both fall again (Table 5, last lid row), which indicates that the effect of *lid* holds in both instructions that contain it.

As discussed in Section 4.2, because the prior we study is a property of the noun and the policy rather than of a particular scene, we compute the scores once per task instead of once per scene. The scores in Table 5 are averaged over ten held-out poses, disjoint from the evaluation poses on which we roll out the method. Single held-out poses mostly agree with the average: deleting a noun moves the score in the same direction as the average on all 10 poses for *microwave* and *lid*, on 8 for *rice cooker* and *box*. An ablation that chooses the instruction separately for each evaluation pose does not improve success (Appendix E).

5.6 Q5: Do the Edits Carry Over to Other Policies Trained on DROID?

We apply the same edits discussed in Section 5.3 to MolmoAct2-DROID (Fang et al., 2026a) and Cosmos3-Nano-Policy-DROID (NVIDIA, 2026). Table 6 reports the success rates of the three policies. Every edit raises success or leaves it unchanged for every policy, but the size of the gain varies: recoloring helps MolmoAct2-DROID more than Cosmos3-Nano-Policy-DROID on the lid; deleting *microwave* helps both, while deleting *box* helps only MolmoAct2-DROID.

6 Conclusion

We identify a strategy selection failure in generalist policies trained on multi-strategy data: on objects that afford only one strategy, these policies can carry out an unsupported strategy, even when the instruction states the requested one. We trace this failure to two features of the input: the color of the object and the object nouns in the instruction. We turn these findings into a steering procedure that prescribes whether and how to edit each input. The edits raise success on all three simulated tasks for all three DROID-trained policies, by amounts that differ across policies and tasks. Future directions include studying where the noun priors come from (Appendix I).

AI USE STATEMENT

Generative AI tools were used in three ways. (1) As part of the method: GPT-5.5 assigns strategies and color labels to DROID episodes, labels the part that each base rollout acts on, and names the strategy a test object affords (Sections 4.1 and 5.2; Appendices B, C and F). This use is described and evaluated in the paper, including an agreement check of the rollout labels against an author's labels (Cohen's $\kappa = 0.91$) and a manual check of the microwave and rice cooker assignments. (2) Writing: AI assistants were used to polish drafts of the paper and to check them for clarity and consistency; the authors reviewed and revised all text. (3) Literature: during the research and the writing, AI assistants were used to find related papers; the authors read and checked every cited work. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

All three policies are public checkpoints: $\pi_{0.5}$-DROID from openpi, and MolmoAct2-DROID and Cosmos3-Nano-Policy-DROID from their authors' releases (Section 5.1). The dataset we read is the public DROID release (Khazatsky et al., 2024). Appendix C gives the rules that build the task and strategy index from its annotations, with a manual check of the assignments, and Table 2 lists every edit our method prescribes, so each edit can be checked without rebuilding the index. The scenes are built in Isaac Lab (NVIDIA, 2025); the microwave and the rice cooker are ArtVIP assets (Jin et al., 2026), and every modification to them is stated in Section 5.1 and Appendix D. Appendix E specifies the attention read-out, including the layers, the flow step, the shared noise sample and the part masks. Every success rate comes from the same ten object poses with three episodes each, and every attention score that sets a decision comes from ten further poses held out from evaluation. Code will be released to reproduce all reported numbers.

REFERENCES

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

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, et al. SAM 3: Segment anything with concepts. In *International Conference on Learning Representations (ICLR)*, 2026.

Lawrence Yunliang Chen, Kush Hari, Karthik Dharmarajan, Chenfeng Xu, Quan Vuong, and Ken Goldberg. Mirage: Cross-embodiment zero-shot policy transfer with cross-painting. In *Robotics: Science and Systems (RSS)*, 2024.

CIE. Colorimetry – part 4: CIE 1976 L*a*b* colour space. Technical Report ISO/CIE 11664-4:2019, International Commission on Illumination, 2019.

Zeyuan Ding, Wenhai Liu, Yang Xu, Jiayu Hu, Yinda Chen, Yi Zhang, Yong Dai, Jian Tang, and Xiaozhu Ju. Pelican-VLA 0.5: Attending before acting benefits generalization. *arXiv preprint arXiv:2607.06655*, 2026.

Maximilian Du and Shuran Song. DynaGuide: Steering diffusion polices with active dynamic guidance. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2506.13922.

Haoquan Fang, Jiafei Duan, Donovan Clay, Sam Wang, Shuo Liu, Weikai Huang, Xiang Fan, Wei-Chuan Tsai, Shirui Chen, Yi Ru Wang, Shanli Xing, Jaemin Cho, Jae Sung Park, Ainaz Eftekhar, Peter Sushko, Karen Farley, Angad Wadhwa, Cole Harrison, Winson Han, Ying-Chun Lee, Eli

VanderBilt, Rose Hendrix, Suveen Ellawela, Lucas Ngoo, Joyce Chai, Zhongzheng Ren, Ali Farhadi, Dieter Fox, and Ranjay Krishna. MolmoAct2: Action reasoning models for real-world deployment. *arXiv preprint arXiv:2605.02881*, 2026a.

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*, 2026b.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, Jinlan Fu, Jingjing Gong, and Xipeng Qiu. LIBERO-Plus: A progressive robustness benchmark for visual-language-action models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 38574–38583, 2026. arXiv:2510.13626.

Asher J. Hancock, Allen Z. Ren, and Anirudha Majumdar. Run-time observation interventions make vision-language-action models more visually robust. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9499–9506, 2025. arXiv:2410.01971.

Arhan Jain. DROID sim evaluation. `https://github.com/arhanjain/sim-evals`, 2025. GitHub repository.

Sarthak Jain and Byron C. Wallace. Attention is not explanation. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019.

Hyun Joe Jeong, Gokul Swamy, and Andrea Bajcsy. Learning what to say to your VLA: Mostly harmless vision language action model steering. *arXiv preprint arXiv:2606.12299*, 2026.

Zhao Jin, Zhengping Che, Tao Li, Zhen Zhao, Kun Wu, Yuheng Zhang, Yinuo Zhao, Zehui Liu, Qiang Zhang, Xiaozhu Ju, Jing Tian, Yousong Xue, and Jian Tang. ArtVIP: Articulated digital assets of visual realism, modular interaction, and physical fidelity for robot learning. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2506.04941.

Tilke Judd, Krista Ehinger, Frédo Durand, and Antonio Torralba. Learning to predict where humans look. In *IEEE International Conference on Computer Vision (ICCV)*, 2009.

Alexander Khazatsky, Karl Pertsch, et al. DROID: A large-scale in-the-wild robot manipulation dataset. In *Robotics: Science and Systems (RSS)*, 2024.

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. *arXiv preprint arXiv:2602.12281*, 2026.

Lightwheel. Lightwheel kitchen: A SimReady kitchen scene for Isaac Sim. `https://github.com/LightwheelAI/Lightwheel_Kitchen`, 2025. Version 1.1.

Miranda Muqing Miao, Subin Kim, Brandon Yang, and Lyle Ungar. Contrastive conceptor activation steering (COAST): Unlocking vision-language-action models through hidden states. *arXiv preprint arXiv:2605.17144*, 2026.

NVIDIA. Isaac Lab: A GPU-accelerated simulation framework for multi-modal robot learning. *arXiv preprint arXiv:2511.04831*, 2025.

NVIDIA. Cosmos 3: Omnimodal world models for physical AI. *arXiv preprint arXiv:2606.02800*, 2026.

OpenAI. GPT-5.5 system card. `https://openai.com/index/gpt-5-5-system-card/`, 2026.

Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. GrIPS: Gradient-free, edit-based instruction search for prompting large language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pp. 3845–3864, 2023.

Wilbert Pumacay, Ishika Singh, Jiafei Duan, Ranjay Krishna, Jesse Thomason, and Dieter Fox. THE COLOSSEUM: A benchmark for evaluating generalization for robotic manipulation. In *Robotics: Science and Systems (RSS)*, 2024.

Haoyuan Shi, Xiancong Ren, Yingji Zhang, Qinfan Zhang, Jiayu Hu, Haozhe Shan, Han Dong, Jinpeng Lu, Yinda Chen, Yi Zhang, Yong Dai, and Xiaozhu Ju. VLA-Trace: Diagnosing vision-language-action models through representation and behavior tracing. *arXiv preprint arXiv:2605.30117*, 2026.

Sangmim Song, Sarath Kodagoda, Marc Carmichael, and Karthick Thiyagarajan. Overcoming visual clutter in vision language action models via concept-gated visual distillation. *arXiv preprint arXiv:2603.10340*, 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.

Andrew Wagenmaker, Yunchu Zhang, Mitsuhiko Nakamoto, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. In *Conference on Robot Learning (CoRL)*, volume 305 of *Proceedings of Machine Learning Research*, pp. 258–282, 2025. arXiv:2506.15799.

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 *IEEE International Conference on Robotics and Automation (ICRA)*, 2025a.

Yuanfei Wang, Xiaojie Zhang, Ruihai Wu, Yu Li, Yan Shen, Mingdong Wu, Zhaofeng He, Yizhou Wang, and Hao Dong. AdaManip: Adaptive articulated object manipulation environments and policy learning. In *International Conference on Learning Representations (ICLR)*, 2025b. arXiv:2502.11124.

Yilin Wu, Ran Tian, Gokul Swamy, and Andrea Bajcsy. From foresight to forethought: VLM-in-the-loop policy steering via latent alignment. In *Robotics: Science and Systems (RSS)*, 2025. arXiv: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. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2026. arXiv:2510.16281.

Youguang Xing, Xu Luo, Junlin Xie, Lianli Gao, Heng Tao Shen, and Jingkuan Song. Shortcut learning in generalist robot policies: The role of dataset diversity and fragmentation. In *Conference on Robot Learning (CoRL)*, volume 305 of *Proceedings of Machine Learning Research*, pp. 3239–3266, 2025. arXiv:2508.06426.

Kechun Xu, Zhenjie Zhu, Anzhe Chen, Shuqi Zhao, Qing Huang, Yifei Yang, Haojian Lu, Rong Xiong, Masayoshi Tomizuka, and Yue Wang. Seeing to act, prompting to specify: A Bayesian factorization of vision language action policy. *arXiv preprint arXiv:2512.11218*, 2025.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *International Conference on Learning Representations (ICLR)*, 2024.

Zhigen Zhao, Mark Leggiero, Yipu Chen, Haoran Liu, Yifan Wu, Huishu Xue, Sirui Zhan, and Ye Zhao. VQActFlow: Vector-quantized action mode steering for multi-task robot manipulation. *arXiv preprint arXiv:2606.21600*, 2026.

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.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. In *International Conference on Learning Representations (ICLR)*, 2023.

Table 7: **Strategies that the index assigns to the episodes of our three tasks.** The assignment is automatic, and the list is not claimed to be exhaustive.

Task	Strategy
Open the microwave	by pulling the door handle
	by pressing the release button
	by pulling the door rim
	by pulling the door handle with a tool-in-hand
Open the rice cooker	by pressing the lid-release button
	by lifting the lid handle
Take the lid off	by pinching-and-lifting the handle/knob
	by pushing the lid rim/edge
	by sliding the slider
	by rotating/twisting the lid body/top surface
	by pressing the lid body/top surface

A A Reference Object Resembling the Training Objects

With the wooden box swapped for a black pot similar to those whose lids are lifted by the knob in DROID, $\pi_{0.5}$-DROID lifts its knob in 29 of 30 rollouts. The pot is Pot057 from the KitchenRoom set of Lightwheel Kitchen v1.1 (Lightwheel, 2025), scaled from its centimeter mesh to meters, and the instruction is *take the lid off*. For the microwave and the rice cooker we have no asset on which the policy performs the requested strategy without an edit, so on those two tasks the evidence that the policy can perform the requested strategy comes from the edits themselves (Sections 5.4 and 5.5).

B Inferring the Afforded Strategy with a Vision-Language Model

Section 3 takes the requested strategy as given. To check that this assumption is mild, we ask a vision-language model to name the strategy each test object affords. The model receives the third-person image at the start of an episode, the task, and every strategy that the index (Section 4.1) lists for that task, and it returns one of these strategies. We query it once on each of the ten poses of each of the three tasks, the same poses used throughout the evaluation (Section 5.1). It returns the strategy the object affords in all 30 queries. The model is GPT-5.5 (OpenAI, 2026), the same one we use for labeling.

C Training Data Statistics and Labeling

Building the index. GPT-5.5 assigns each episode a task label of the form $\mathrm{verb}(\mathrm{object})$ from its language annotation, such as $\mathrm{open}(\mathrm{microwave})$ or $\mathrm{pick\text{-}place}(\mathrm{lid})$. The object in the label is the one whose state changes, not its container or support, so taking a lid off a pot and taking a lid off a box receive the same label, $\mathrm{pick\text{-}place}(\mathrm{lid})$ (Section 3). GPT-5.5 then proposes candidate strategies for each task from the task name and assigns each episode to one of them, from its annotation when the annotation names both a contact site and a motion and from side-view and wrist keyframes otherwise; when no candidate fits, it adds a new strategy. Over DROID's 75,144 episodes, the index covers 737 objects and 1,496 tasks. We check the strategy assignments of every microwave and rice cooker episode by hand; the lid task has too many episodes to check, so its counts come from the index alone.

Strategies of our three tasks. Table 7 lists the strategies that the index assigns to the episodes of our three tasks. The assignment is automatic, so the list is what GPT-5.5 recovers from the annotations and keyframes, and we do not claim that it is exhaustive.

Color labels. Table 8 gives, for each task, the color labels of the objects on which DROID demonstrates the requested strategy, counted over episodes, together with the label of our test object and the edit that follows from the criterion of Section 4.1. Every episode labeled black among the 2,119

Table 8: **Color labels of the objects on which DROID demonstrates each requested strategy.** Share of episodes per label, in percent; the typical labels, which together cover more than half of the episodes, are in bold. *Clear* names a material and is never a target. The microwave and rice cooker labels are verified by hand; the lid episodes are too many to check, so their labels come from the automatic index and may contain errors.

Task (episodes)	Labels (share of episodes, %)	Test object	Edit
Lid (2119)	**black** 52.3, silver 23.7, clear 7.3, white 5.4, blue 5.3, red 2.5, other 3.3	brown	recolor to black
Microwave (54)	**white** 51.9, silver 42.6, gray 5.6	white	none
Rice cooker (9)	**black** 100	white	recolor to black

episodes that demonstrate lifting a lid by its knob shows a pot, so black pots make up 52.3% of these episodes, which supports the description in Section 1.

Color modifiers. We parse every annotation with spaCy (`en_core_web_sm`) and count a word as a modifier of the object if it stands to the left of an object noun and is either an adjectival modifier of it, a compound that appears in our closed lists of color, material and size words, or the token right before the noun when that token is in these lists or is an adjective. Of the 225,432 annotations in the three instruction slots of DROID, 130,612 describe the object with at least one modifier, and 93,731 of these (71.8%) include a color word; an annotation can contain more than one modifier.

D Assets and Implementation Details

Test scenes. We build the test scenes in Isaac Lab (NVIDIA, 2025) on the living-room scene of Jain (2025), which is tuned so that policies trained on the real-world DROID data run in it zero-shot, and replace only the objects on the table; the middle column of Figure 1 shows the three scenes.

Test objects. The box of the lid task is a custom-built asset whose lid is set into the box and can only move upward before being lifted. The microwave and the rice cooker are articulated assets from ArtVIP (Jin et al., 2026). The door of the microwave has no handle and opens only when its release button is pressed, and we recolor the button yellow so that it stands out on the control panel. The original handle of the rice cooker sits on the back and cannot move; we replace it with a handle on top of the lid, where the rice cookers in DROID have theirs, and remove the raised steam valve from the lid, leaving the round mark it sat on. A rollout succeeds when the lid is lifted clear of the box, when the door of the microwave opens past 30 degrees, and when the lid of the rice cooker is raised. Every result uses the joint-position version of $\pi_{0.5}$-DROID released with openpi (`pi05_droid_jointpos`), since the scene of Jain (2025) works best with joint-position policies.

Tracking the object. SAM 3 tracks the object in each camera independently. It starts the track on the first frame whose detection score exceeds a threshold, propagates the mask from there, and re-detects the object locally when the track degrades.

Recoloring. Inside the tracked mask, we convert each pixel to CIELAB and replace its chromaticity (a^*, b^*) with that of the target color. For lightness, we keep each pixel's deviation from the mean lightness of the object and move the mean to that of the target. The target color is a fixed CIELAB value for each color label.

E Attention Analysis

Layer selection. The layers are chosen by a quantity separate from the one the check evaluates. For each layer we measure the share of the action tokens' attention to the third-person image that falls inside the object's silhouette, averaged over the ten held-out poses, which indicates whether the layer attends to the object at all; the check then asks a different question, whether the requested part gains attention relative to the rest of the object when a noun is deleted. Across the three tasks this share is highest at layers 11 to 13, and the check is scored as the mean AUC over those three layers, computed per layer and then averaged rather than by averaging the attention maps first. Table 9 gives, on the ten held-out poses from which our method takes its decisions, the change in score for

Table 9: **The deletion check at each layer read.** Change in the AUC on the requested part when the noun is deleted, averaged over the ten held-out poses, on the observation our method prescribes. The decisions use the mean over layers 11 to 13; the mean over all 18 layers gives the same sign for every noun. *lid* is deleted from the full request, and in the last lid row from the request with *box* already deleted.

Task	Noun	Layer 11	Layer 12	Layer 13	Mean 11–13	Mean, all layers
Lid	*box*	$+0.040$	$+0.080$	-0.025	$+0.032$	$+0.013$
	lid	-0.363	-0.024	-0.160	-0.182	-0.084
	lid (after *box*)	-0.248	-0.128	-0.158	-0.178	-0.063
Microwave	*microwave*	$+0.275$	$+0.252$	$+0.241$	$+0.256$	$+0.124$
Rice cooker	*rice cooker*	-0.089	-0.020	$+0.008$	-0.034	-0.029

each noun at each of these layers and under the mean over all 18 layers, which involves no selection. The all-layer mean gives the same sign as the mean over layers 11 to 13 for every noun, so the decisions do not rest on the choice of layers. Single layers agree with it for *microwave* and *lid* at all three layers, and for *box* and *rice cooker* at layers 11 and 12 but not at layer 13, where these two nouns, whose pooled effects are the smallest, change the score by less than 0.03. We do not claim that these layers are the right choice beyond the policy and the objects studied here.

Read-out. For each instruction we run one forward pass of the policy on the first frame of a pose and read the attention from the action tokens to the image tokens of the third-person camera, averaged over heads and over action tokens, at the last of the ten flow-matching steps that the policy runs at deployment. The instructions compared on a pose share the first frame, the robot state and one noise sample drawn with a fixed seed, so only the text differs. The requested part and the object silhouette are annotated by hand once on the 3D model of each object and rendered into a mask for every pose, which we resize to the grid of image tokens. The score of a map is its AUC-Judd with the pixels of the requested part as positives and the rest of the silhouette as negatives (Judd et al., 2009); the per-layer maps for layers other than 11 to 13 come from a second forward pass on the same inputs.

We use attention only to check which part of the object an edit draws the policy toward, and we do not read it as an explanation of the policy's decision (Jain & Wallace, 2019); the effect of each deletion on behavior is measured separately, in success (Section 5.5). Prior work has also examined the attention of the action pathway: Ding et al. (2026) report that in current open-source VLAs the action pathway attends diffusely, spreading over the arm, the background and task-irrelevant objects, and argue that concentrating it on the manipulation-relevant region benefits generalization. Shi et al. (2026) find, on LIBERO with a $\pi_{0.5}$ finetuned there, that $\pi_{0.5}$ spreads its action attention more broadly than OpenVLA, with lower peak-hit rates. Because action attention is spread out rather than peaked, a threshold on its absolute mass would be arbitrary, and we score the map by how it ranks the requested part against the rest of the object instead.

Single poses. Table 10 lists, for each held-out pose, the change in the layers-11-to-13 score when a noun is deleted. Every pose for *microwave* and for *lid* has the sign of the pooled decision; two poses for *rice cooker* and two for *box* have the opposite sign, each by less than 0.03, and one pose for *box* leaves the score unchanged. Table 11 compares choosing the instruction once per task, as our method does, with choosing it separately for each evaluation pose from that pose's own scores.

F Annotation of the Base Rollouts

Table 1 reports the labels of GPT-5.5, which labels all 180 base rollouts. One author labels a random third of them, 60 rollouts, without seeing the labels of GPT-5.5, and we report the agreement between the two on this subset. The two labels agree on 58 of the 60 rollouts (Cohen's $\kappa = 0.91$). In one of the two others, GPT-5.5 labels the requested part and the author labels both parts; in the other, GPT-5.5 labels the unsupported strategy's part and the author labels neither part. In no rollout does one label the requested part and the other the unsupported strategy's part, and the two disagreements err in opposite directions, so the labels do not systematically favor the failure. Table 12 lists, for each task, the parts the labels refer to.

Table 10: **Per-pose change in the attention score on the held-out poses when a noun is deleted.** Mean AUC over layers 11 to 13 after deletion minus before, on one first frame per pose, with the robot state and the sampled noise held fixed. *lid* is deleted from the full request in the fourth column and from the request with *box* already deleted in the fifth. Poses are numbered 1 to 10; the first row of each column is the pooled difference.

Pose	*microwave*	*box*	*lid*	*lid* (after *box*)	*rice cooker*
Mean	$+0.209$	$+0.134$	-0.172	-0.108	-0.053
1	$+0.141$	$+0.050$	-0.353	-0.210	-0.048
2	$+0.403$	$+0.012$	-0.036	-0.153	-0.052
3	$+0.186$	$+0.344$	-0.106	-0.008	-0.108
4	$+0.119$	-0.001	-0.018	-0.028	$+0.003$
5	$+0.152$	0.000	-0.051	-0.023	-0.095
6	$+0.203$	$+0.390$	-0.241	-0.085	-0.082
7	$+0.119$	$+0.122$	-0.291	-0.017	-0.025
8	$+0.158$	$+0.223$	-0.123	-0.097	-0.073
9	$+0.311$	-0.023	-0.336	-0.397	$+0.019$
10	$+0.294$	$+0.227$	-0.169	-0.057	-0.068

Table 11: **Choosing the instruction once per task or separately for each pose.** Success on the evaluation poses of $\pi_{0.5}$-DROID. The per-pose choice deletes a noun on a pose only if deleting it raises that pose's own attention score.

Task	Once per task	Per pose	Poses where the choices differ
Lid	23/30	22/30	3 (*box* kept on poses 4, 5 and 9)
Microwave	12/30	12/30	0
Rice cooker	24/30	24/30	2 (*rice cooker* deleted on poses 4 and 9)

G Geometric Variants of the Objects

The failure is not tied to the particular geometry of the test object. Each row of Table 13 changes the geometry of the object while keeping its appearance, and is run with the full request.

The policy still fails on every variant under the full request. On the microwave, the variant makes the requested part more prominent, which does not help. Changing the geometry is not an option at deployment in any case, since the deployer cannot rebuild the object in front of the robot.

H Keeping Only the Part Phrase

Naming the part does not account for the effect of the instruction edit. Keeping only the part phrase, as in *lift the knob* or *press the release button*, loses most of the effect on all three tasks (Table 14): success falls from 23/30 to 11/30 on the lid, from 12/30 to 5/30 on the microwave and from 24/30 to 6/30 on the rice cooker. The effect of the edited instruction therefore does not reduce to naming the right part.

I Additional Discussion

Why the two edits help differently on different tasks. Recoloring alone accounts for the gain on the rice cooker, deleting the object noun alone accounts for it on the microwave, and the lid needs the recoloring and gains further from the deletion (Section 5.3). The white rice cooker differs in color from every object on which DROID demonstrates the handle lift, all of which are black, whereas the white microwave already has the typical color of the objects on which DROID demonstrates the button press (Table 8), so recoloring has nothing to change there. One explanation consistent with both tasks is that the object noun decides the strategy when the appearance of the object leaves the choice open. Three tasks cannot establish this explanation, but it is testable without any rollout, since the overlap between the appearance distributions of two strategies of a task can be read from

Table 12: **Annotation.** Each base rollout is labeled with the part that the gripper acts on for most of its duration. GPT-5.5 labels every rollout, and one author labels a random third of them independently.

Task	Requested part	Unsupported strategy's part
Lid	knob	rim
Microwave	release button	door seam, where a handle would be
Rice cooker	handle on top of the lid	mark left by the removed valve

Table 13: **The failure persists under changes to the geometry.** Each row is a geometric variant of the test object, evaluated with the full request on the original appearance ($n = 30$, ten poses $\times$ three episodes). The edits are evaluated on the test objects only.

Task	Variant	Success
Lid	stoneware jar with a knob in place of the box	2/30
Microwave	release button at twice its size	3/30
Rice cooker	steam valve restored on the lid	8/30

the index. If it holds, demonstrating a strategy on more varied objects would also reduce how much a policy relies on the object noun.

Where the noun priors come from. The origin of the noun priors is open. They are unlikely to come from the instructions in the robot data, since DROID describes both strategies of a task with the same nouns. A natural candidate is the vision-language pretraining of the backbone, in which *microwave* may co-occur with handles more often than with release buttons. Running the same paired comparison on the backbone before robot training would test this candidate; we have not run it.

Beyond color. We edit color because it can be changed on a live image without changing the geometry or the physics of the object. Other attributes, such as material, texture and shape, may also separate the objects of different strategies in the training data, and editing them at run time is a question of tooling rather than of principle. As such tools mature, the observation edit can draw on a larger set of attributes and cover more objects.

Scope. Our results are limited in four respects. First, the observation edit reads the training data, which assumes that the data are public, as DROID is. Second, the attention check uses the interface of one policy; the two other policies receive the edits computed for $\pi_{0.5}$-DROID without a check of their own. Third, the agreement between attention and behavior rests on four nouns across three tasks. Fourth, all experiments run in simulation: we compare the online edit with the same edit rendered in the asset, but real lighting, segmentation and contact remain untested.

Table 14: **Success rates with the instruction our method prescribes and with only its part phrase.** On the observation our method prescribes, with recolorings rendered in the asset.

	Instruction		Success	
Task	Our method	Part phrase	Our method	Part phrase
Lid	take the lid off by lifting the knob	lift the knob	23/30	11/30
Microwave	open by pressing the release button	press the release button	12/30	5/30
Rice cooker	open the rice cooker by lifting the handle	lift the handle	24/30	6/30