# A PHENOMENOLOGY OF NEURAL GEOMETRY IN VISION-LANGUAGE-ACTION MODELS

**Anonymous authors**
Paper under double-blind review

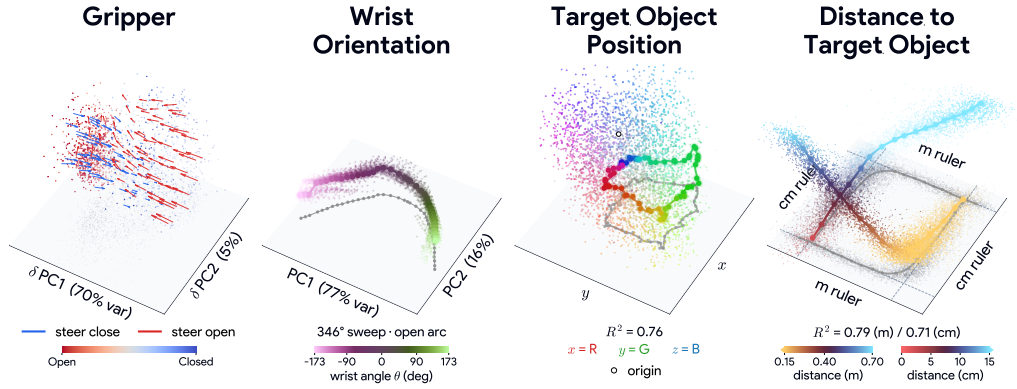


Figure 1: **Control variables realize distinct geometries in a VLA policy.** MolmoBot (Deshpande et al., 2026) activations over many states (clouds) and controlled sweeps varying one physical quantity (traces). Clouds and traces are activations projected onto a low-dimensional basis (principal components for the gripper and wrist; the span of a linear probe's weights for target position); for distance, the activations along the two ruler directions are plotted against the true distance (z axis). (**Gripper**) Open and closed states from natural rollouts separate along one axis; arrows show interventions reversing the command. (**Wrist**) Rotation sweeps pooled over 64 houses trace an open arc, not the circle unconstrained rotation would give. (**Target position**) A structured trajectory within the cloud of teleported target placements. (**Distance**) Two orthogonal rulers, coarse for approach and fine for grasp.

## ABSTRACT

Robotic manipulation is inherently geometric, but what shape does the world take inside the policy that performs it? Prior work on vision-language-action (VLA) policies establishes that task-relevant variables are linearly readable from activations, yet readability certifies only that a variable is present and says nothing about the shape its values trace, which may be curved and multi-dimensional. Using controlled sweeps, natural rollouts, linear probes, and activation interventions, we build a taxonomy of seven task-relevant concepts in VLA models. We find that realized geometry is predominantly multi-dimensional and is shaped by the constraints under which the policy acts. Physical joint limits, for example, leave wrist orientation an open arc rather than a circle, and distance to the target, a single scalar, is realized in two orthogonal subspaces, a coarse ruler for the approach and a fine ruler for the grasp. The same geometries reappear in two further VLA models on real-world robot data, indicating that they are neither model-specific nor artifacts of simulation. We demonstrate how the findings are actionable by steering along the geometry a concept realizes. Overall, this work lays out an initial phenomenology of realized geometry in VLA policies and demonstrates that a concept's shape carries as much importance as its readability.

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

# 1 INTRODUCTION

Robotic manipulation is inherently geometric. Robotic systems operate in three-dimensional space through articulated kinematic structures, while the objects they manipulate and the tasks they perform have their own geometric organization, including symmetries, constraints, and phases of interaction. Any general manipulation policy must contend with this structure, and vision-language-action (VLA) models have become the dominant paradigm for building such policies (Black et al., 2025; Brohan et al., 2023; Deshpande et al., 2026; Fang et al., 2026; Kim et al., 2024; Lee et al., 2025; Octo Model Team et al., 2024; Physical Intelligence et al., 2025b). VLAs inherit visual and semantic priors from backbones pretrained on internet-scale data, including vision encoders (Oquab et al., 2024; Radford et al., 2021; Zhai et al., 2023), language models (Gemma Team et al., 2024; Touvron et al., 2023), and full vision-language models (Beyer et al., 2024; Deitke et al., 2025; Driess et al., 2023; Wang et al., 2024). This data has grown by orders of magnitude in volume and diversity (Bjorck et al., 2025; Bu et al., 2025; Deshpande et al., 2026; O'Neill et al., 2024; Walke et al., 2023), with generalization improving along power-law-like trends (Lin et al., 2025). The resulting systems are increasingly capable, general, and even deployed in the real world (Figure AI, 2025a;b; 2026; Gemini Robotics Team, 2026; Gemini Robotics Team et al., 2025; Physical Intelligence et al., 2025a;b).

How such competence is reflected in the representations of these VLAs is the subject of a nascent literature (Bhardwaj et al., 2026; Buurmeijer et al., 2026; Goyal, 2026; Grant et al., 2026; Häon et al., 2025; Khan et al., 2025; Lu et al., 2025; Molinari et al., 2025; Shi et al., 2026; Swann et al., 2026; Yang & Mokalla, 2026). In particular, one line of work uses probes, testing whether a quantity of interest can be read out from the model's activations (Bhardwaj et al., 2026; Buurmeijer et al., 2026; Grant et al., 2026; Lu et al., 2025; Molinari et al., 2025; Yang & Mokalla, 2026). A second trains sparse autoencoders (SAEs) (Bhalla et al., 2026b; Bricken et al., 2023; Bussmann et al., 2024; Costa et al., 2025; Cunningham et al., 2024; Fel et al., 2025; Gao et al., 2025; Hindupur et al., 2025; Lieberum et al., 2024; Lubana et al., 2026; Makhzani & Frey, 2014; Rajamanoharan et al., 2024a;b; Templeton et al., 2024; Thasarathan et al., 2025) to discover concepts without supervision (Grant et al., 2026; Khan et al., 2025; Swann et al., 2026), and the discovered features even provide control over the policy via steering.

Both lines of work share a limitation. A probe can ensure that a variable is decodable, yet says nothing about the structure through which the model "realizes" it, and an SAE inherits the linear representation hypothesis (LRH) (Costa et al., 2025; Park et al., 2024), which casts each concept as a single direction in activation space and therefore fixes that structure in advance. Interpretability for language and vision models, conscious of this limitation, has moved beyond the single direction to multi-dimensional features and curved geometry (Bhalla et al., 2026a; Chung et al., 2018; Cohen et al., 2020; Costa et al., 2025; Dhimoila et al., 2026; Diego-Simón et al., 2024; Dooms & Gauderis, 2025; Engels et al., 2025; Fel et al., 2025; 2026a;b; Feucht et al., 2026; Gorton, 2024; Gurnee et al., 2026; Hosseini et al., 2026; Kantamneni & Tegmark, 2025; Karkada et al., 2026; Lubana et al., 2026; Modell et al., 2025; Park et al., 2025a;b; 2026; Pearce et al., 2025; Sarfati et al., 2026; Sun et al., 2025; Tvetkova et al., 2025; Wurgaft et al., 2026; Yocum et al., 2025), a line of work now gathered under the name of *Neural Geometry* (Geiger et al., 2026).

Do VLAs realize their variables through similarly rich geometry? Two facts suggest they might. First, vision models, including the backbones VLAs inherit, already carry colors, shadows, and position as multi-dimensional features rather than single directions (Fel et al., 2026a;b; Pach et al., 2026; Uselis et al., 2026). Second, representations tend to inherit the symmetries of their training data, so that cyclic concepts, for example, come to be arranged on circles (Karkada et al., 2026). The variables a manipulation policy must track, such as poses, distances, contacts, and task phase, are geometric objects with symmetries of their own before the network ever represents them.

We take up this question and make the following contributions:

- **Realized geometry as a lens for VLA interpretability.** We formalize the distinction between *concept realization*—the capacity to extract information about a concept under a given readout—and its *realized geometry*—the structure activations trace as the concept varies. Linear readability establishes only that a concept is present, while its realized geometry determines how its values are structured and whether they can be manipulated.
- **A taxonomy of realized geometry in MolmoBot.** Combining controlled sweeps, natural rollouts, linear probes, and activation interventions, we chart the realized geometry of seven control concepts:

gripper state, end-effector position, distance to the target, target object position, contact, task phase, and wrist orientation. We find this geometry is predominantly multi-dimensional and shaped by the constraints under which the policy acts: joint limits leave wrist orientation an open arc rather than a circle, while the numerical precision of the embedding splits distance into orthogonal coarse (meter-scale) and fine (centimeter-scale) rulers.

- **Real-world transfer.** We reproduce these geometries in $\pi_{0.5}$ (Physical Intelligence et al., 2025b) and MolmoAct2 (Fang et al., 2026) on the real-world DROID dataset (Khazatsky et al., 2024), indicating they are neither model-specific nor artifacts of simulation.

## 2 Realization of Control Geometry

Before specifying the seven concepts of interest that this study will examine and documenting their realized geometry, we first clarify what is meant by realized and by geometry. In particular, our aim here is to distinguish two questions that are frequently conflated in the literature: whether a concept can be recovered from a policy's activations – which we term *realized* – and what geometric shape that recovered concept takes within activation space – which we term *realized geometry*. The first question asks about recoverability, the second about structure; we now formalize those notions.

**Notation.** Throughout, bold symbols denote vector or matrix valued quantities and calligraphic symbols their domains. We model observations as arising from a data-generating process whose latent state collects $m$ concepts, writing $[m] = \{1, \dots, m\}$ and $\boldsymbol{z} = (\boldsymbol{z}_1, \dots, \boldsymbol{z}_m) \sim p_{\boldsymbol{z}}$ for a state drawn from $\boldsymbol{\mathcal{Z}} = \mathcal{Z}_1 \times \cdots \times \mathcal{Z}_m$. Borrowing the language of conceptual spaces (Gärdenfors, 2000), we take each concept $i \in [m]$ to be a structured domain $(\mathcal{Z}_i, d_i)$ rather than a single vector, so that $\mathcal{Z}_i$ supplies the values the concept can assume; a domain may be discrete, continuous, or manifold-valued, with an intrinsic dimension $k_i = \dim(\mathcal{Z}_i)$ when smooth. We write $\boldsymbol{x}$ and $\boldsymbol{a}$ for a single input and activation, and $\boldsymbol{X}$ and $\boldsymbol{A} \in \mathbb{R}^{n \times d}$ for $n$ of them stacked row-wise.

**Data-generating process.** We consider the minimal data-generating process

$$\boldsymbol{z} \sim p_{\boldsymbol{z}}, \qquad \boldsymbol{x} = \boldsymbol{g}(\boldsymbol{z}) \in \mathcal{X}, \qquad \boldsymbol{a} = \boldsymbol{f}(\boldsymbol{x}) \in \mathcal{A} \subseteq \mathbb{R}^d, \tag{1}$$

in which the (potentially stochastic) generator $\boldsymbol{g} : \boldsymbol{\mathcal{Z}} \to \mathcal{X}$ combines the concepts $\boldsymbol{z}$ and instantiates an observation $\boldsymbol{x}$ (e.g. an image), while $\boldsymbol{f} : \mathcal{X} \to \mathcal{A}$ maps that observation to an activation $\boldsymbol{a}$ at a fixed layer of a VLA. In this work, we will focus on a particular set of concepts which are control-relevant and are modeled by both a *quantity* and its *possible values*[1], so that $\mathcal{Z}_i$ may describe a binary gripper state, a scalar distance in $\mathbb{R}_+$, an end-effector position in $\mathbb{R}^3$, or a wrist orientation in $\mathbb{S}^1$. These are properties of the world or of the control process, and the policy accesses them only through the observation that the generator $\boldsymbol{g}$ produces, so that nothing guarantees a concept present in $\boldsymbol{z}$ remains recoverable from $\boldsymbol{a}$. Whether it does is the first thing to establish, and it is a question that shall become well posed only once the means of recovery is fixed.

**Definition 1** (Concept realization)**.** *Let $(\mathcal{Z}_i, d_i)$ be the concept domain of interest and $\mathcal{V}$ be a class of readouts $\boldsymbol{\gamma} : \mathcal{A} \to \mathcal{Z}_i$, for $\varepsilon \geq 0$, we say that concept $i$ is $(\mathcal{V}, \varepsilon)$-realized in $\boldsymbol{a}$ if*

$$\inf_{\boldsymbol{\gamma} \in \mathcal{V}} \mathbb{E}\left(d_i\left(\boldsymbol{\gamma}(\boldsymbol{a}), \boldsymbol{z}_i\right)\right) \leq \varepsilon, \tag{2}$$

*where the expectation is taken over the process in Equation 1.*

This definition is relative to both a readout class and an error tolerance. It therefore specifies not only whether information about $\boldsymbol{z}_i$ is present, but also how "easily" it can be extracted. This follows the same operational principle as predictive $\mathcal{V}$-information (Xu et al., 2020): usable information is defined relative to a class of predictors. In particular, unless specified, our main case is the class of affine readouts $\boldsymbol{\gamma}(\boldsymbol{a}) = \boldsymbol{W}\boldsymbol{a} + \boldsymbol{b}$. When concept $i$ is realized under this class, we say that it is *linearly readable*. We reserve *realization* for the general, readout-relative notion and use *linear readability* only for this particular case.

[1]This mirrors the recent notion of a feature field (Yocum et al., 2025), where a feature is defined over a topological space – rather than at a point – and the geometry of that space embedded in activation space determines which functions of the concept are linearly representable.

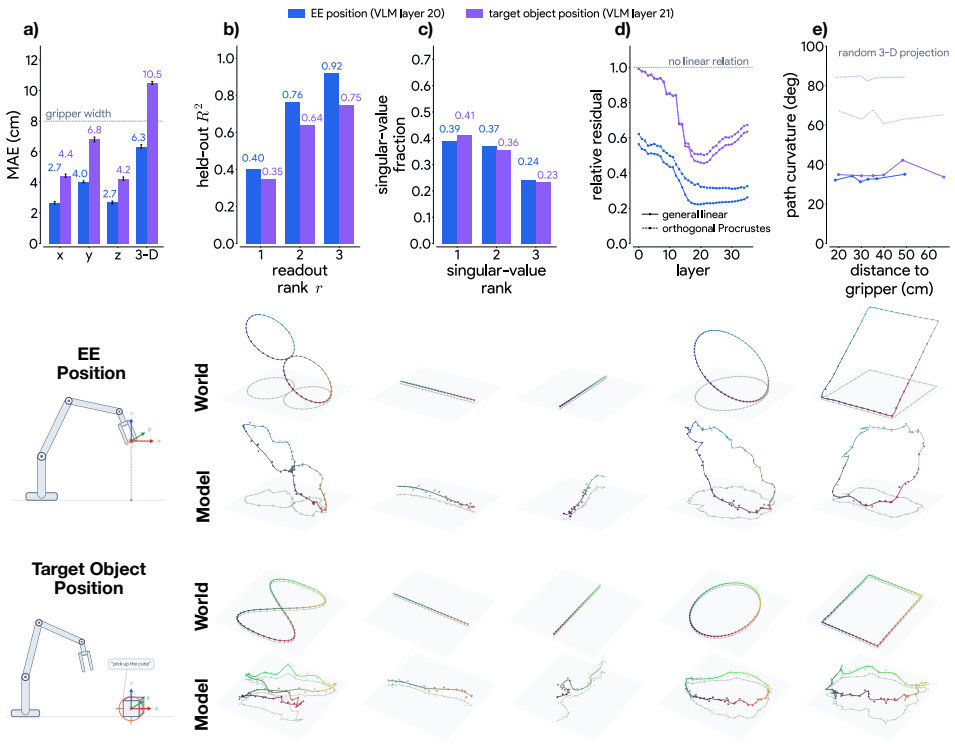


Figure 2: **MolmoBot's VLM encodes its end-effector (layer 20) and the target object (layer 21) in separate, approximately Euclidean 3-D subspaces. Top:** (a) held-out per-axis error relative to gripper width; (b) held-out $R^2$ vs. readout rank; (c) singular values of the rank-3 readout; (d) residual of a linear map from world to activation displacements across layers (solid: general linear; dashed: orthogonal; reference line: no linear relation); (e) curvature of straight world paths inside each subspace vs. distance to the gripper, compared with a random 3-D projection. **Middle, bottom:** five gripper paths through the workspace (middle) and five target paths across its support surface (bottom), shown in world coordinates (upper) and projected into the corresponding subspace (lower) at a shared scale. Subspaces are fit with the tested house held out.

An important remark before jumping to the realized geometry is that realization does not imply that the activations are isomorphic to the concept domain[2] or that the policy causally "uses" the recoverable information. Finally, it also lacks a description of the geometry of the conceptual space itself, which we now turn our attention to.

## 3 A Taxonomy of Realized Geometry in MolmoBot

We now build a taxonomy of how a model realizes the variables of pick and pick-and-place tasks. Our subject is MolmoBot (Deshpande et al., 2026), which pairs a VLM that reads the language instruction and camera images of the scene with an action expert that reads the robot's proprioceptive state and generates action chunks via flow matching, cross-attending to the VLM at each layer (Appendix B.1). We choose MolmoBot for three reasons: it is a relatively strong policy on numerous benchmarks (Deshpande et al., 2026), and we hypothesize that stronger models realize task geometry more faithfully; its weights are open, so we can record and edit activations; and it runs in MolmoSpaces (Kim et al., 2026), which provides ground-truth task variables and lets us intervene on the world mid-rollout. We study seven variables in three groups: spatial geometry (EE position, target object position, and distance to the target), task structure (contact and task phase), and action geometry (wrist orientation and gripper aperture). For each, we ask whether the model represents it and with what geometry.

[2]It does not require (1) preservation of distances or (2) topology, nor (3) an efficient or minimal representation (see App. A for an example).

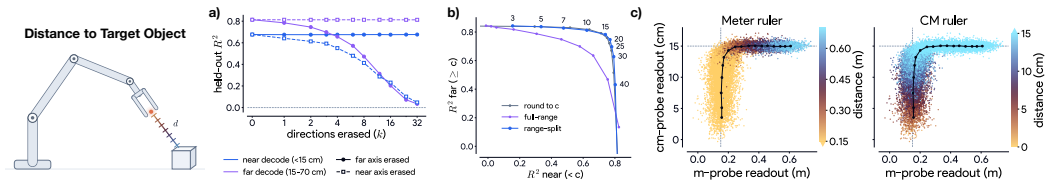


Figure 3: **MolmoBot's VLM encodes distance to the target in nearly orthogonal near- and far-range codes (layer 21).** (a) Near (<15 cm) and far (15–70 cm) probe $R^2$ after erasing the top-$k$ directions of the other probe, computed against each range's own variance. Erasing far directions leaves near decoding intact while far decoding collapses, and vice versa. The intact ($k = 0$) probes have errors of $\sim$1.3 cm (near) and $\sim$4.0 cm (far). (b) Near $R^2$ ($<c$) vs. far $R^2$ ($\geq c$) as the cutoff $c$ is swept. Clipped ('round to c') probes match probes fit separately on each range and outperform a single full-range probe. (c) Activations plotted by their meter- and centimeter-probe readouts, colored by true distance in meters (left) and centimeters (right). Far samples vary along the meter axis while the centimeter readout saturates, and near samples do the opposite, tracing an L.

### 3.1 Spatial Geometry

We begin with spatial geometry in the VLM backbone, which receives no proprioception and must infer these variables from images: EE position, target object position, and distance to the target.

**End-effector and target object position.** Since both positions live in $\mathbb{R}^3$, we expect a faithful code for each to occupy a three-dimensional subspace. We teleport the arm through poses above the workspace and the target through a uniform sweep of positions on the workspace surface, record VLM activations after each teleport, and fit ridge probes onto $(x, y, z)$, evaluated on held-out houses (Appendix B.3). For both variables, three dimensions are necessary and sufficient: held-out $R^2$ rises sharply up to rank 3 and does not improve beyond it, reaching 0.92 for EE position (layer 20) and 0.76 for the target (layer 21), with per-axis error within the gripper's width (Fig. 2a–c). These subspaces also preserve spatial geometry: translating the gripper or target by a known world vector shifts the activations in a way well fit by an orthogonal (Procrustes) map (Fig. 2d), and straight world paths stay nearly straight inside each subspace (Fig. 2e). Position thus sets the taxonomy's baseline: from pixels alone, the VLM re-encodes a Euclidean quantity at its native dimension and approximately its native metric.[3] We next turn to a variable that couples the two: distance to the target.

**Distance to target object.** Distance to the target is a scalar, but the policy needs it at two very different scales: it must tolerate noise during the approach yet resolve the grasp to within a centimeter. A single direction with finite precision may struggle to carry both, so we hypothesize that the model encodes distance in more than one subspace. We fit separate linear probes on near (<15 cm) and far (15–70 cm) states and iteratively erase one probe's directions from the activations (Appendix B.4). Each code survives removal of the other: erasing 32 far directions drops far $R^2$ from 0.80 to near zero while near $R^2$ stays at 0.66, and vice versa (Fig. 3a).

To isolate the two codes, we fit a meter probe with targets clipped below 15 cm and a centimeter probe with targets clipped above 15 cm, a cutoff that is Pareto-optimal for both ranges (Fig. 3b). Each probe reads its own range well ($R^2 = 0.79$ and 0.71) and saturates outside it, so activations projected onto the two readouts trace an L (Fig. 3c). The two probe directions are nearly orthogonal, with a cosine of 0.07 against a split-half ceiling of $\sim$0.27 (Appendix Fig. 12). Clipping does not impose this orthogonality, since a probe can only select directions the activations contain. On synthetic activations with a single planted distance axis, the clipped probes stay nearly parallel (cosine 0.95), and probes fit separately on each range, without clipping, are also orthogonal (0.04 vs. $\sim$0.18; range-split in Appendix Fig. 12). Distance thus departs from the position baseline: the model realizes a single scalar along separate directions for coarse and fine scales, consistent with the limited precision of a representation bending how a concept is encoded.

[3]The action expert also carries rank-3 subspaces for both variables (Appendix Figs. 9 and 10).

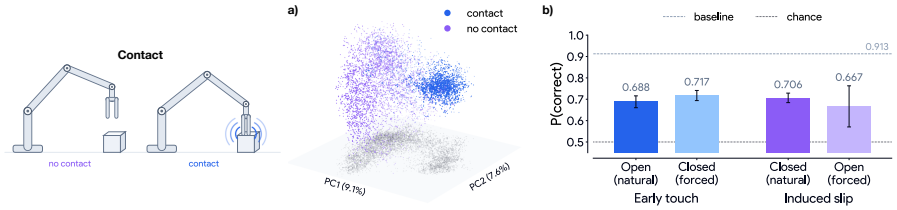


Figure 4: **MolmoBot's VLM encodes contact with the target along a direction that survives off-schedule contact (layer 24).** (a) Natural rollouts projected onto their first three principal components, colored by contact; the two states separate cleanly. (b) An LDA probe trained on natural rollouts decodes contact with 91.3% accuracy. When we force an early touch or an induced slip by teleporting the object, the same probe still decodes contact well above chance.

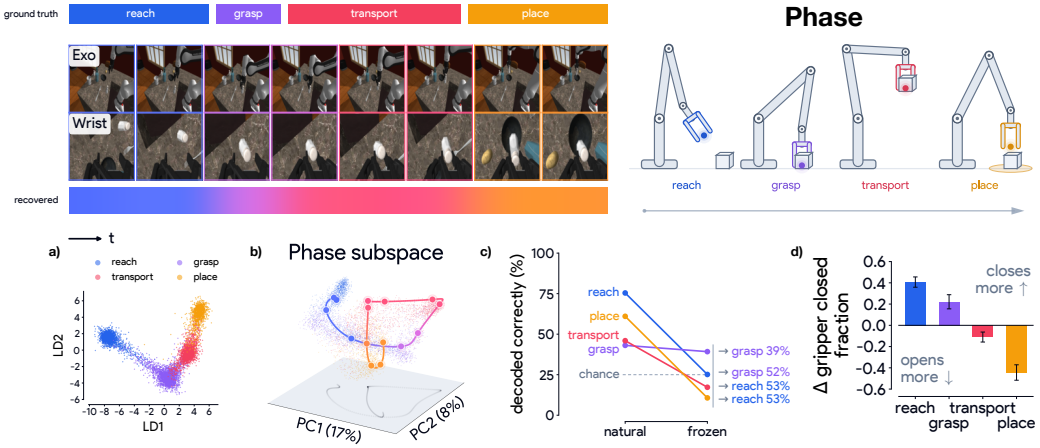


Figure 5: **MolmoBot's action expert encodes task phase, but infers it from motion, so pauses are read as phase transitions (layer 24). Top:** an example pick-and-place trajectory and its path through the phase subspace. **Bottom:** (a) LDA projection of natural rollouts, colored by phase. (b) First three principal components of natural activations, colored by LDA phase prediction. (c) Freezing the arm shifts the decoded phase to grasp during reach and grasp, and to reach during transport and place. (d) Change in the fraction of executed-chunk steps commanding a closed gripper (frozen − natural; mean ± 95% CI across landmarks). Frozen reach and grasp moments close the gripper, while frozen transport and place moments open it; transport's shift builds late in the chunk (Fig. 20).

### 3.2 Task Structure

We next turn to variables that describe the structure of the task rather than the geometry of the scene: contact with the target object, in the VLM, and task phase, in the action expert.

**Contact.** Contact with the target is binary, so we treat it as a two-class problem and use linear discriminant analysis (LDA) to find the direction that separates the classes. We align rollouts to the simulator's contact time and fit the discriminant on VLM activations (layer 24). Contact and no-contact states separate cleanly even in an unsupervised PCA of natural activations (Fig. 4a), and the discriminant decodes contact with 91.3% accuracy. Because contact co-occurs with other variables in natural rollouts, high accuracy alone does not show that the direction tracks contact itself. We therefore force contact off schedule by teleporting the object into the gripper early (an early touch) or away from it during a grasp (an induced slip). Even though both interventions are out of distribution for the model, the same direction still decodes contact well above chance (Fig. 4b), indicating that the model maintains a real contact code rather than some proxy for task timing.

**Phase.** We now move to the action expert, where we expect variables coupled to actions to appear most strongly. Unlike contact, task phase is defined by the whole trajectory rather than a single state, so we fit a multi-class LDA on activations along full pick-and-place trajectories (AE, layer 24; labels

**Box 1: Belief vs. Action.** Probe accuracy does not distinguish the two regimes.

**Belief.** Some structures hold the model's estimate of its own state or of the scene, such as the angle of its wrist or the distance to the object. Editing such a structure affects behavior only *indirectly*: the policy treats the edited value as a perception and acts on it. The policy then moves to close the gap between the perceived state and its goal, so empirically, the behavior shifts *against* the edit. We call this a *compensation effect*.

**Action.** Other structures hold the command itself. Editing these changes behavior *directly*, by the written amount, with other outputs unchanged. We find such edits from the action side, by pulling an objective on the output back into activation space.

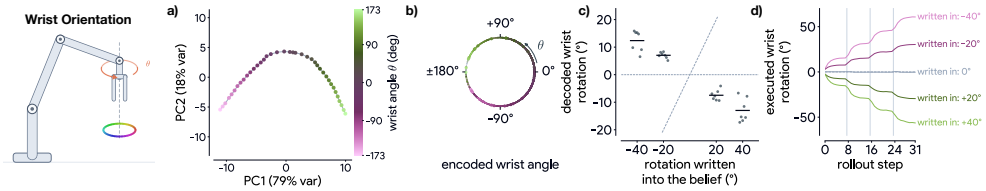


Figure 6: **The action expert encodes wrist orientation as an open arc, and edits to it act as beliefs (layer 9).** We sweep the wrist's roll joint through its full range ($-172.8°$ to $+172.8°$), holding other joints fixed, and record activations at each angle (camera views in Fig. 21). (a) The sweep traces an open curve: its endpoints, nearly the same physical pose, lie far apart. (b) An autoencoder maps activations onto an exact circle. (c) A single $\pm 20°$ or $\pm 40°$ edit produces a decoded rotation opposing the written angle (dots: seven held-out scenes; ticks: means; dashed diagonal: the response a command would produce). (d) Holding the edit at every step compounds the corrections, rotating the wrist opposite the written angle by about its magnitude (vertical lines mark action chunks).

in Appendix B.7). Natural activations separate into distinct clusters for reach, grasp, transport, and place, both in the LDA projection and in the first three principal components (Fig. 5a–b). To test how the model infers phase, we freeze the arm for several actions. Freezing during reach or grasp shifts the decoded phase to grasp, and freezing during transport or place shifts it to reach (Fig. 5c). The shifts are also behavioral: freezing raises the fraction of steps commanding a closed gripper by 41 percentage points during reach and lowers it by 11 and 44 points during transport and place (Fig. 5d).

MolmoBot thus represents contact and task phase with different robustness. While contact remains decodable when forced off schedule, phase does not survive a pause. Since the model does not receive previous actions, it cannot keep an explicit time index and must re-derive phase from motion at each chunk. In demonstrations, the arm stops when the gripper actuates, so a still arm is read as a phase transition, and the model issues the corresponding gripper command.

### 3.3 Action Geometry

So far, we have studied variables by reading them out. For the last two, wrist orientation and gripper aperture, we also write to them, which reveals a distinction that readout alone cannot: whether an edit lands on what the model believes or on what it commands (Box 1).

**Wrist orientation.** Orientation is cyclic, so its most compact code would be a circle. However, the wrist's roll joint is limited to $\pm 172.8°$ and never completes a full turn, and the action expert receives this angle directly as a raw, non-wrapping joint reading. Consistent with this, sweeping the joint through its range (Appendix Fig. 21) traces an open arc in the action expert (layer 9), whose ends lie far apart in activation space despite being only $14°$ apart physically (Fig. 6a). The action expert thus encodes an unwrapped angle: after a full turn past the joint limit (overridden in simulation), the images are identical but its activation does not return to the start, while the VLM's does (Appendix Fig. 23). To write to this code, we cannot add one fixed vector: on an open arc, the direction that advances the rotation depends on where the activation sits (Fig. 6a). We instead train a small MLP autoencoder that maps activations onto an exact circle (Fig. 6b), so that rotating by $\theta$ is the same operation at every state; the edit is the decoded difference between the rotated and unrotated

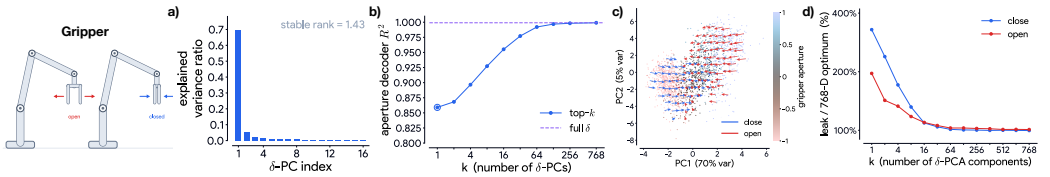


Figure 7: **Gripper control is one-dimensional (AE, layer 32).** (a) The first principal component of the pullback steering vectors captures $\sim$70% of their variance. (b) One principal component of the pullback steering vectors decodes the commanded gripper aperture nearly as well as the full $\delta$. (c) Natural-rollout activations colored by gripper state; the steering vectors point from each cluster toward the other. (d) One component flips the gripper but leaks into the arm joints relative to using the full 768-dimensional steering vector; more components remove the leakage.

points (Appendix B.8). This edit is built from the representation alone and does not command motion, making it a belief edit (Box 1). Applied once mid-rollout, it produces a compensation effect: the decoded action rotates $0.34 \pm 0.09$ against each written degree (Fig. 6c), as if the policy were closing the gap between its believed angle and its goal. Held at every step, the corrections compound, and the wrist rotates opposite the edit by 1.0–1.4$\times$ the written angle (Fig. 6d). Thus, even a geometry-matched edit lands on a belief, so for the gripper we work from the action side.

**Gripper.** The gripper command is an action variable (see Box 1), so a natural question is, then, how can we edit just the gripper action? Here, we derive the edit from the action output. For each sampled rollout state, we use pullback (Wurgaft et al., 2026) to find a steering vector $\boldsymbol{\delta}$, added as $\boldsymbol{h} \leftarrow \boldsymbol{h} + \boldsymbol{\delta}$ in the action expert (layer 32), optimized with gradients from the action output until the decoded gripper command flips while the arm joints stay fixed. Natural activations form two clouds by gripper state, and each successful steering vector points from its own cloud toward the other (Fig. 7c), so edits derived only from action gradients recover the axis that separates the states. These edits are largely one-dimensional: their first principal component captures $\sim$70% of their variance (Fig. 7a) and is sufficient to accurately decode the commanded gripper state (Fig. 7b). This direction is not fully isolated, however: steering along it also moves the arm joints, and suppressing this leakage requires additional components (Fig. 7d).

Together, the wrist and gripper show two things. First, physical limits shape realized geometry: a variable we hypothesized would be circular is encoded as an arc. Second, recovering a variable's geometry does not determine how editing it affects behavior: edits along the wrist's arc land on a belief and elicit compensation, whereas edits derived from gripper action gradients change the action.

## 4 Representations in Real Robots

All experiments in Section 3 were run in simulation. Since the gap between simulation and reality remains a central open problem in robotics (Jakobi et al., 1995; Kadian et al., 2020; Tobin et al., 2017), we ask whether the geometry we found carries over to real data. We repeat our analyses on MolmoBot and two additional VLAs, $\pi_{0.5}$ (Physical Intelligence et al., 2025b) and MolmoAct2 (Fang et al., 2026), using episodes from the real-world DROID dataset (Khazatsky et al., 2024). Real data lacks the privileged information of simulation, such as ground-truth object poses and mid-rollout interventions, so we focus on four variables we can measure: gripper aperture, wrist orientation, EE position, and distance to target, for which we use distance to the eventual grasp location as a proxy.

All four variables are recovered in every model, with cross-model Procrustes similarities well above a shuffled-variable null (Fig. 8). The gripper structure is the most consistent across models: pullback steering vectors again separate closing from opening, with the first principal component carrying 31–59% of their variance. EE position is linearly decodable in all three models ($R^2$ of 0.78–0.92), and the probed activations follow the arm smoothly through an episode. Wrist orientation is decoded less well ($R^2$ of 0.52–0.69) and agrees least across models. Most notably, the two-scale distance code reappears: in all three models, far points spread along the meter ruler while near points split off onto the centimeter ruler. This structure is sharper in $\pi_{0.5}$ and MolmoAct2 than in MolmoBot, consistent with MolmoBot being evaluated on real data outside its simulated training distribution. Together,

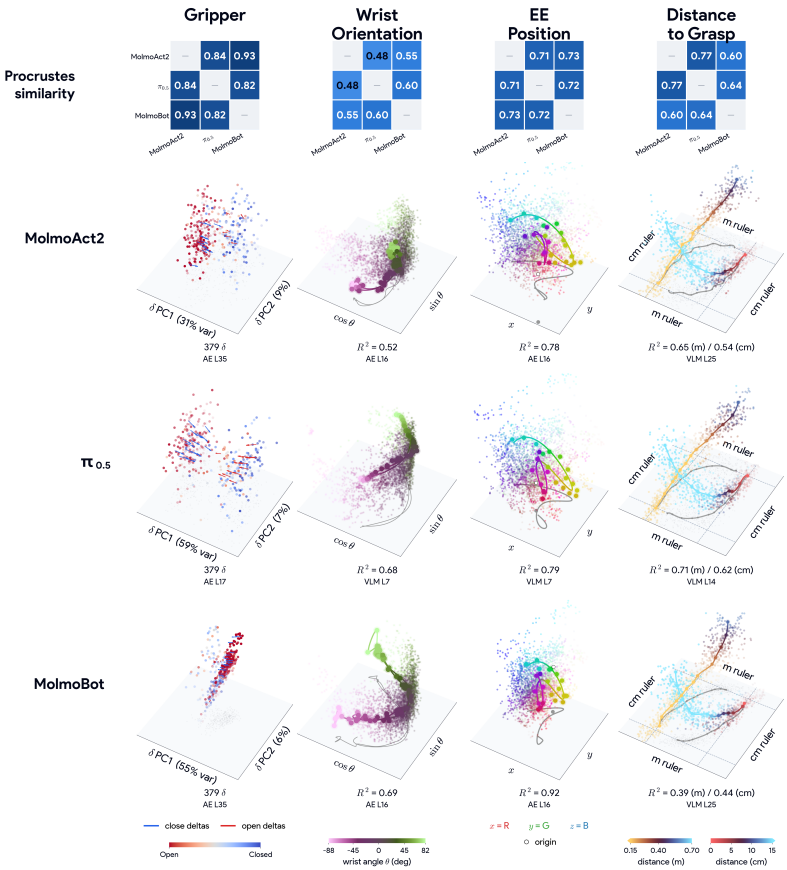


Figure 8: **The geometry recovered in MolmoBot largely reappears in $\pi_{0.5}$ and MolmoAct2 on real-world DROID data.** Procrustes similarity between each pair of models is highest for the gripper and lowest for wrist orientation, and all values exceed a shuffled null (0.00–0.02 for gripper and distance; 0.11–0.18 for wrist and EE position). (**Gripper**) Pullback steering vectors (379 per model) on their top three principal components, colored by whether they close (blue) or open (red) the gripper. (**Wrist orientation**) Activations on a $(\cos\theta, \sin\theta)$ probe, colored by angle. (**EE position**) Activations on the $(x, y, z)$ probe. (**Distance to grasp**) The real-world proxy for distance to target: activations on the meter and centimeter rulers of Fig. 3, with far (cool) and near (warm) points spreading along different rulers. Episode trajectories are overlaid in the last three columns.

these results suggest that the geometry we identified is neither an artifact of simulation nor specific to one model, but a recurring way in which VLAs organize the variables needed for manipulation.

## 5 Conclusion

In this work, we asked what geometry manipulation variables realize inside VLAs. In MolmoBot, the answer is rarely a single direction: position occupies a near-Euclidean 3-D subspace, wrist orientation an open arc imposed by joint limits, and distance two nearly orthogonal rulers, and this structure largely recurs in $\pi_{0.5}$ and MolmoAct2 on real data. Two lessons follow. First, decodability is only a starting point: a variable can be linearly readable yet require a geometry-matched edit to steer, and even then the edit may land on a belief the policy compensates against rather than an action it executes. Interpretability tools for VLAs should therefore match both the geometry a concept realizes and its role in control. Second, this geometry is shaped by the policy's inputs and physical limits, so design choices such as how joint angles are encoded shape how the model represents them. Our taxonomy is supervised, but it provides ground truth against which unsupervised methods, such as block-sparse featurizers (Fel et al., 2026a), can be validated. We believe the geometry of VLAs is a promising direction for both interpretability and better policies.

### AI USE STATEMENT

In this work, we used generative AI tools to help develop the conceptual framework, formulate the mathematical definitions and claims in our formalization, propose and refine hypotheses, design and provide feedback on experiments, implement methods, build the simulation datasets from policy rollouts, support data analysis, and interpret results. We have not used generative AI tools to generate synthetic data (all data come from simulator rollouts or real-robot recordings) or assist with translation, and assistance with proofs and qualitative or thematic data analysis are not applicable to this work. Additionally, we used generative AI tools to brainstorm, identify relevant literature and research gaps, summarize existing literature, suggest experimental parameters, create and edit code, create and modify figures, suggest a structure for the paper, draft and edit parts of the paper for readability, format references, and propose title options. We have reviewed all AI-assisted work. All citations suggested by LLMs were manually verified against the original sources, LLM-generated code was verified by having the model explain its implementation and checking that it matched the intended method, and all reported results were checked by the authors. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

### REPRODUCIBILITY STATEMENT

All models we study have publicly released weights (MolmoBot, MolmoAct2, and $\pi_{0.5}$), and all simulation experiments run in the open-source MolmoSpaces simulator on ProcTHOR houses. Appendix B gives the details needed to reproduce the results of the main text: the model architecture and where activations are read (Appendix B.1), how the four simulation datasets are collected (Appendix B.2), and, for each variable, the data, probes, controls, and interventions behind each main-text figure (Appendices B.3–B.9). Appendix B.10 describes the real-robot analyses, which use the public DROID dataset, together with the one additional episode we recorded for the wrist roll sweep. We intend to release our source code for data collection, probing, and steering upon publication.

## REFERENCES

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. PaliGemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

Usha Bhalla, Thomas Fel, Can Rager, Sheridan Feucht, Tal Haklay, Daniel Wurgaft, Siddharth Boppana, Matthew Kowal, Vasudev Shyam, Jack Merullo, et al. Do sparse autoencoders capture concept manifolds? *arXiv preprint arXiv:2604.28119*, 2026a.

Usha Bhalla, Alex Oesterling, Claudio Mayrink Verdun, Hima Lakkaraju, and Flavio Calmon. Temporal sparse autoencoders: Leveraging the sequential nature of language for interpretability. In *International Conference on Learning Representations*, volume 2026, pp. 141147–141175, 2026b.

Atiksh Bhardwaj, Edward Weiyi Duan, Prithwish Dan, Wei-Chiu Ma, and Preston Culbertson. Decoding task progress from vla representations. *arXiv preprint arXiv:2608.13474*, 2026.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi 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. *arXiv preprint arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. $\pi_0$: A vision-language-action flow model for general robot control. *Robotics: Science and Systems (RSS)*, 2025.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. *Proceedings of the Conference on Robot Learning (CoRL)*, 2023.

Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Shenyuan Gao, Xindong He, Xuan Hu, Xu Huang, et al. AgiBot World Colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025.

Bart Bussmann, Patrick Leask, and Neel Nanda. BatchTopK sparse autoencoders. *arXiv preprint arXiv:2412.06410*, 2024.

Hugo Buurmeijer, Carmen Amo Alonso, Aiden Swann, and Marco Pavone. Observing and controlling features in vision-language-action models. *arXiv preprint arXiv:2603.05487*, 2026.

SueYeon Chung, Daniel D Lee, and Haim Sompolinsky. Classification and geometry of general perceptual manifolds. *Physical Review X*, 8(3):031003, 2018.

Uri Cohen, SueYeon Chung, Daniel D Lee, and Haim Sompolinsky. Separability and geometry of object manifolds in deep neural networks. *Nature communications*, 2020.

Valérie Costa, Thomas Fel, Ekdeep Singh Lubana, Bahareh Tolooshams, and Demba Ba. From flat to hierarchical: Extracting sparse representations with matching pursuit. *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Procthor: Large-scale embodied ai using procedural generation. *Advances in neural information processing systems*, 35:5982–5994, 2022.

Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and PixMo: Open weights and open data for state-of-the-art vision-language models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 91–104. IEEE, 2025.

Abhay Deshpande, Maya Guru, Rose Hendrix, Snehal Jauhri, Ainaz Eftekhar, Rohun Tripathi, Max Argus, Jordi Salvador, Haoquan Fang, Matthew Wallingford, et al. MolmoB0T: Large-scale simulation enables zero-shot manipulation. *arXiv preprint arXiv:2603.16861*, 2026.

Gregoire Dhimoila, Thomas Fel, Victor Boutin, and Agustin Picard. Cross-modal redundancy and the geometry of vision-language embeddings. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2026.

Pablo Diego-Simón, Stéphane d'Ascoli, Emmanuel Chemla, Yair Lakretz, and Jean-Rémi King. A polar coordinate system represents syntax in large language models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Thomas Dooms and Ward Gauderis. Finding manifolds with bilinear autoencoders. *arXiv preprint arXiv:2510.16820*, 2025.

Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. PaLM-E: An embodied multimodal language model. *Proceedings of the International Conference on Machine Learning (ICML)*, 2023.

Josh Engels, Eric Michaud, Isaac Liao, Wes Gurnee, and Max Tegmark. Not all language model features are one-dimensionally linear. In *International Conference on Learning Representations*, volume 2025, pp. 84591–84622, 2025.

Haoquan Fang, Jiafei Duan, Donovan Clay, Sam Wang, Shuo Liu, Weikai Huang, Xiang Fan, Wei-Chuan Tsai, Shirui Chen, Yi Ru Wang, et al. MolmoAct2: Action reasoning models for real-world deployment. *arXiv preprint arXiv:2605.02881*, 2026.

Thomas Fel, Ekdeep Singh Lubana, Jacob S Prince, Matthew Kowal, Victor Boutin, Isabel Papadimitriou, Binxu Wang, Martin Wattenberg, Demba Ba, and Talia Konkle. Archetypal SAE: Adaptive and stable dictionary learning for concept extraction in large vision models. *Proceedings of the International Conference on Machine Learning (ICML)*, 2025.

Thomas Fel, Matthew Kowal, Mozes Jacobs, Dron Hazra, Usha Bhalla, Lee Sharkey, Lucius Bushnaq, Satchel Grant, Tal Haklay, Thomas Icard, et al. Structuring sparsity: Block-sparse featurizers capture visual concept manifolds. *arXiv preprint arXiv:2606.25234*, 2026a.

Thomas Fel, Binxu Wang, Michael Lepori, Matthew Kowal, Andrew Lee, Randall Balestriero, Sonia Joseph, Ekdeep Singh Lubana, Talia Konkle, Demba Ba, et al. Into the rabbit hull: From task-relevant concepts in DINO to Minkowski geometry. In *International Conference on Learning Representations*, volume 2026, pp. 154538–154584, 2026b.

Sheridan Feucht, Tal Haklay, Usha Bhalla, Daniel Wurgaft, Can Rager, Raphaël Sarfati, Jack Merullo, Thomas McGrath, Owen Lewis, Ekdeep Singh Lubana, et al. Arithmetic in the wild: Llama uses base-10 addition to reason about cyclic concepts. *arXiv preprint arXiv:2605.01148*, 2026.

Figure AI. F.02 contributed to the production of 30,000 cars at BMW. https://www.figure.ai/news/production-at-bmw, November 2025a. Accessed September 2026.

Figure AI. Helix: A vision-language-action model for generalist humanoid control. https://www.figure.ai/news/helix, February 2025b.

Figure AI. Introducing Helix 02: Full-body autonomy. https://www.figure.ai/news/helix-02, January 2026.

Leo Gao, Tom Dupre la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. In *International Conference on Learning Representations*, volume 2025, pp. 26721–26754, 2025.

Peter Gärdenfors. *Conceptual spaces: The geometry of thought*. MIT press, 2000.

Atticus Geiger, Ekdeep Singh Lubana, Thomas Fel, Jack Merullo, Michael Byun, Owen Lewis, and Thomas McGrath. The world inside neural networks: How neural geometry will unlock understanding and control of AI. Goodfire Research, May 2026. URL https://www.goodfire.com/research/the-world-inside-neural-networks.

Gemini Robotics Team. Gemini Robotics 2 brings whole-body intelligence to robots. Google DeepMind blog, July 2026. URL https://deepmind.google/blog/gemini-robotics-2-brings-whole-body-intelligence-to-robots/. Accessed: 2026-08-25.

Gemini Robotics Team, Abbas Abdolmaleki, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Ashwin Balakrishna, Nathan Batchelor, Alex Bewley, Jeff Bingham, et al. Gemini Robotics 1.5: Pushing the frontier of generalist robots with advanced embodied reasoning, thinking, and motion transfer. *arXiv preprint arXiv:2510.03342*, 2025.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on Gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

Liv Gorton. The missing curve detectors of InceptionV1: Applying sparse autoencoders to InceptionV1 early vision. *arXiv preprint arXiv:2406.03662*, 2024.

Aryan Goyal. The hitchhiker's guide to mechanistic interpretability of vision-language-action models. In *Mechanistic Interpretability Workshop at ICML 2026*, 2026.

Bryce Grant, Xijia Zhao, and Peng Wang. Not all features are created equal: A mechanistic study of vision-language-action models. *arXiv preprint arXiv:2603.19233*, 2026.

Wes Gurnee, Emmanuel Ameisen, Isaac Kauvar, Julius Tarng, Adam Pearce, Chris Olah, and Joshua Batson. When models manipulate manifolds: The geometry of a counting task. *arXiv preprint arXiv:2601.04480*, 2026.

Bear Häon, Kaylene Stocking, Ian Chuang, and Claire Tomlin. Mechanistic interpretability for steering vision-language-action models. *arXiv preprint arXiv:2509.00328*, 2025.

Sai Sumedh R Hindupur, Ekdeep Singh Lubana, Thomas Fel, and Demba Ba. Projecting assumptions: The duality between sparse autoencoders and concept geometry. *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Eghbal A Hosseini, Yuxuan Li, Yasaman Bahri, Declan Campbell, and Andrew Kyle Lampinen. Context structure reshapes the representational geometry of language models. *arXiv preprint arXiv:2601.22364*, 2026.

Nick Jakobi, Phil Husbands, and Inman Harvey. Noise and the reality gap: The use of simulation in evolutionary robotics. In *European conference on artificial life*, pp. 704–720. Springer, 1995.

Abhishek Kadian, Joanne Truong, Aaron Gokaslan, Alexander Clegg, Erik Wijmans, Stefan Lee, Manolis Savva, Sonia Chernova, and Dhruv Batra. Sim2real predictivity: Does evaluation in simulation predict real-world performance? *IEEE Robotics and Automation Letters*, 5(4):6670–6677, 2020.

Subhash Kantamneni and Max Tegmark. Language models use trigonometry to do addition. *arXiv preprint arXiv:2502.00873*, 2025.

Dhruva Karkada, Daniel J Korchinski, Andres Nava, Matthieu Wyart, and Yasaman Bahri. Symmetry in language statistics shapes the geometry of model representations. *arXiv preprint arXiv:2602.15029*, 2026.

Momin Ahmad Khan, Novak Boskov, Fatima M Anwar, and Manzoor A Khan. Controlling vision–language–action policies through sparse latent directions. In *Mechanistic Interpretability Workshop at NeurIPS 2025*, 2025.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. DROID: A large-scale in-the-wild robot manipulation dataset. *Robotics: Science and Systems (RSS)*, 2024.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Pannag Sanketi, et al. OpenVLA: An open-source vision-language-action model. *Proceedings of the Conference on Robot Learning (CoRL)*, 2024.

Yejin Kim, Wilbert Pumacay, Omar Rayyan, Max Argus, Winson Han, Eli VanderBilt, Jordi Salvador, Abhay Deshpande, Rose Hendrix, Snehal Jauhri, et al. MolmoSpaces: A large-scale open ecosystem for robot navigation and manipulation. *arXiv preprint arXiv:2602.11337*, 2026.

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

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. Gemma Scope: Open sparse autoencoders everywhere all at once on Gemma 2. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pp. 278–300, 2024.

Fanqi Lin, Yingdong Hu, Pingyue Sheng, Chuan Wen, Jiacheng You, and Yang Gao. Data scaling laws in imitation learning for robotic manipulation. In *International Conference on Learning Representations*, volume 2025, pp. 54877–54910, 2025.

Hong Lu, Hengxu Li, Prithviraj Singh Shahani, Stephanie Herbers, and Matthias Scheutz. Probing a vision-language-action model for symbolic states and integration into a cognitive architecture. In *2025 IEEE International Conference on AI and Data Analytics (ICAD)*, pp. 1–8. IEEE, 2025.

Ekdeep Singh Lubana, Can Rager, Sai Sumedh R Hindupur, Valérie Costa, Oam Patel, Sonia Murthy, Thomas Fel, Greta Tuckute, Daniel Wurgaft, Eric Bigelow, et al. Priors in time: Missing inductive biases for language model interpretability. In *International Conference on Learning Representations*, volume 2026, pp. 42231–42290, 2026.

Alireza Makhzani and Brendan Frey. K-sparse autoencoders. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

Alexander Modell, Patrick Rubin-Delanchy, and Nick Whiteley. The origins of representation manifolds in large language models. *arXiv preprint arXiv:2505.18235*, 2025.

Marco Molinari, Leonardo Nevali, Saharsha Navani, and Omar G Younis. Emergent world representations in OpenVLA. *arXiv preprint arXiv:2509.24559*, 2025.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. In *Robotics: Science and Systems (RSS)*, 2024.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *The Journal of Transactions on Machine Learning Research (TMLR)*, 2024.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

Mateusz Pach, Jessica Bader, Quentin Bouniot, Serge Belongie, and Zeynep Akata. The latent color subspace: Emergent order in high-dimensional chaos. *arXiv preprint arXiv:2603.12261*, 2026.

Core Francisco Park, Andrew Lee, Ekdeep Singh Lubana, Yongyi Yang, Maya Okawa, Kento Nishi, Martin Wattenberg, and Hidenori Tanaka. ICLR: In-context learning of representations. In *International Conference on Learning Representations*, 2025a.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. *Proceedings of the International Conference on Machine Learning (ICML)*, 2024.

Kiho Park, Yo Joong Choe, Yibo Jiang, and Victor Veitch. The geometry of categorical and hierarchical concepts in large language models. In *International Conference on Learning Representations*, volume 2025, pp. 76441–76463, 2025b.

Kiho Park, Todd Nief, Yo Joong Choe, and Victor Veitch. The information geometry of softmax: Probing and steering. *Proceedings of the International Conference on Machine Learning (ICML)*, 2026.

Michael Pearce, Elana Simon, Michael Byun, and Daniel Balsam. Finding the tree of life in Evo 2. *Goodfire Research*, August 2025. URL https://www.goodfire.com/research/phylogeny-manifold.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, et al. $\pi^*_{0.6}$: a VLA that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025a.

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

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Tom Lieberum, Vikrant Varma, János Kramár, Rohin Shah, and Neel Nanda. Improving sparse decomposition of language model activations with gated sparse autoencoders. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024a.

Senthooran Rajamanoharan, Tom Lieberum, Nicolas Sonnerat, Arthur Conmy, Vikrant Varma, Janos Kramar, and Neel Nanda. Jumping ahead: Improving reconstruction fidelity with JumpReLU sparse autoencoders. *arXiv preprint arXiv:2407.14435*, 2024b.

Raphaël Sarfati, Eric Bigelow, Daniel Wurgaft, Siddharth Boppana, Jack Merullo, Atticus Geiger, Owen Lewis, Tom McGrath, and Ekdeep Singh Lubana. The shape of beliefs: Geometry, dynamics, and interventions along representation manifolds of language models' posteriors. *arXiv preprint arXiv:2602.02315*, 2026.

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

Xiaoqing Sun, Alessandro Stolfo, Joshua Engels, Ben Wu, Senthooran Rajamanoharan, Mrinmaya Sachan, and Max Tegmark. Dense SAE latents are features, not bugs. *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Aiden Swann, Lachlain McGranahan, Hugo Buurmeijer, Monroe Kennedy III, and Mac Schwager. Sparse autoencoders reveal interpretable and steerable features in vla models. *arXiv preprint arXiv:2603.19183*, 2026.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from Claude 3 Sonnet. *Transformer Circuits Thread*, 2024.

Harrish Thasarathan, Julian Forsyth, Thomas Fel, Matthew Kowal, and Konstantinos Derpanis. Universal sparse autoencoders: Interpretable cross-model concept alignment. *Proceedings of the International Conference on Machine Learning (ICML)*, 2025.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30. IEEE, 2017.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Lenka Tvetkova, Thea Bruesch, Teresa Dorszewski, Fabian Martin Mager, Rasmus Ortoft Aagaard, Jonathan Foldager, Tommy Sonne Alstrom, and Lars Kai Hansen. On convex decision regions in deep network representations. *Nature Communications*, 2025.

Arnas Uselis, Andrea Dittadi, and Seong Joon Oh. Compositional generalization requires linear, orthogonal representations in vision embedding models. *arXiv preprint arXiv:2602.24264*, 2026.

Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on robot learning*, pp. 1723–1736. PMLR, 2023.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-VL: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

Daniel Wurgaft, Can Rager, Matthew Kowal, Vasudev Shyam, Sheridan Feucht, Usha Bhalla, Tal Haklay, Eric Bigelow, Raphael Sarfati, Thomas McGrath, et al. Manifold steering reveals the shared geometry of neural network representation and behavior. *arXiv preprint arXiv:2605.05115*, 2026.

Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. A theory of usable information under computational constraints. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

Tony Yang and Aidan Mokalla. Interpretable signals for vision-language-action model failure. In *Mechanistic Interpretability Workshop at ICML 2026*, 2026.

Julian Yocum, Cameron Allen, Bruno Olshausen, and Stuart Russell. Neural manifold geometry encodes feature fields. *NeurIPS 2025 Workshop on Symmetry and Geometry in Neural Representations*, 2025.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2023.

# A Realization does not determine Geometric Realization.

For example, linear readability does not imply linear realized geometry. Consider the map

$$\boldsymbol{\varphi}(t) = \begin{pmatrix} t \\ \cos t \\ \sin t \end{pmatrix}. \tag{3}$$

The value $t$ is exactly linearly readable through $\gamma(\boldsymbol{a}) = a_1$, yet its realized geometry is a helix. The concept is intrinsically one-dimensional, but its trace is curved and occupies several activation dimensions. Thus, the subspace used by a linear readout should not be confused with the shape traced by the concept itself.

This distinction also matters for control. A single fixed steering direction $\boldsymbol{v}$ can follow a concept globally only when

$$\boldsymbol{\varphi}_{i,\boldsymbol{c}}(\boldsymbol{u} + \boldsymbol{\delta}) - \boldsymbol{\varphi}_{i,\boldsymbol{c}}(\boldsymbol{u}) \approx \boldsymbol{v}\boldsymbol{\delta} \tag{4}$$

throughout the relevant range. For a curved or multiscale geometry, the appropriate activation steering vector instead changes with the current concept value. Another way to put it is that a readout specifies how to measure a concept from an activation but it does not necessarily specify how to move the activation while remaining on the geometry realized by that concept. Linear readability therefore does not imply linear steerability.

# B Experimental Details

## B.1 MolmoBot Architecture

MolmoBot (Deshpande et al., 2026) consists of two streams: a vision-language model (VLM) and an action expert. The VLM takes in the language instruction and images from two cameras: an exocentric camera providing a fixed third-person view of the workspace, and a wrist camera mounted on the end effector providing a first-person view that moves with the gripper. At each query, the VLM receives the current frame and a frame from eight steps earlier from each camera. The VLM receives no proprioceptive input, so any information it carries about the robot's state must be recovered from these observations.

The action expert takes in the robot's proprioceptive state together with noisy action tokens, which it denoises into an action chunk over 10 steps of flow matching. The two streams contain 36 layers each, and each layer of the action expert cross-attends to the VLM layer at the same depth, so visual and language information can enter the action computation at every level. Information flows in one direction only: the action expert reads from the VLM, but the VLM does not attend to the action expert. The action expert can therefore access the target object, which appears only in the camera images, solely through this cross-attention.

## B.2 Data

Simulation experiments run in MolmoSpaces on ProcTHOR (Deitke et al., 2022) houses. All use the *pick* task ("Pick up the X") except phase, which uses *pick-and-place* ("Pick up the X and place it in or on the Y"). Every probe is 5-fold cross-validated with houses held out. We collect four datasets:

1. *Natural pick rollouts* (1,011 episodes, 196 houses), with action-expert activations at every action-chunk query. Used for wrist orientation, the gripper pullback, and contact (for which we replay 299 episodes to record VLM activations).
2. *Object teleports* (906 scenes). The target is teleported to up to 16 placements per scene (Figure 16) and activations are recorded at the first four chunk boundaries, with an additional near-range set of placements within 35 cm of the hand (69,534 states in total). Used for distance and direction.
3. *Crossed arm×object teleports* (945 scenes). Five object placements are crossed with ten arm poses, recorded immediately after the teleport (q0) and after one executed chunk (q1). Used for both position variables.
4. *Pick-and-place rollouts* ($\sim$525 episodes, 181 houses, 71% success). Used for phase.

Position, distance, direction, and contact are read from the VLM residual stream, mean-pooled over the token positions given below. Phase, wrist orientation, and the gripper are read from the action expert, averaged over the 16 action tokens at the final flow step unless noted. Layers are indexed from 0 in both streams.

### B.3 End-Effector and Target Object Position

**Data.** In each scene we pick ten gripper targets at random in a box in front of the robot (20–60 cm forward, up to 30 cm to either side, 5–40 cm above the table) and move the arm there with inverse kinematics. We keep a target only if the gripper gets within 1.5 cm of it without the arm intersecting anything, and we drop arm–object pairs in which the gripper is within 12 cm of the object. Each arm pose is paired with the scene's first five placements from the object-teleport dataset. After each teleport the robot holds its new arm configuration, so the policy only sees frames taken after the move. Labels are the gripper and object positions read from the simulator after the teleport, in the robot's base frame.

**Probes.** For each variable we average the activations over a group of tokens and fit a ridge regression onto its $(x, y, z)$ position, with houses held out; both variables use the same rows and folds. We use the token group and layer that decode each variable best: for EE position, the short labels that introduce each camera image ("Image 1 View 1" plus the image start and end markers) at layer 20; for the target, the instruction tokens at layer 21. To count how many dimensions a readout uses, we restrict its predictions to their top one, two, or three principal directions (the rank curves) and report the singular values of the predictions. Table height differs between scenes but not within one, so part of the $z$ signal only identifies the scene; subtracting each scene's mean from the true and predicted positions removes it (within-scene $R^2$: 0.92 for EE, 0.57 for the target). Neither probe predicts the other variable ($R^2 \leq 0.01$).

**Equivariance.** To test whether a subspace preserves spatial geometry, we take pairs of states from the same scene that differ in one variable only: two arm poses with the object fixed, or two object placements with the arm fixed (the object moves only horizontally, so its displacement is 2-D). On training scenes we fit a map from the world displacement to the resulting change in activation within the rank-3 subspace, and on held-out scenes we report the median relative error of its predictions, once for any linear map and once for a rotation with uniform scaling (Procrustes). Shuffling which world displacement goes with which activation change gives an error of 1.00.

**Curvature.** We move the hand along four straight lines (forward, sideways, up, and toward the object; 48 scenes) and the object along straight lines with the arm at its home pose (43 scenes), 24 points per line, and project the activations onto the subspace fit without those scenes. Curvature is the angle between consecutive $\sim$7 cm segments of the projected path, binned by hand–object distance and compared with the same activations projected onto random 3-D subspaces (median of eight). The paths in Figure 2 come from one of these held-out scenes.

**Action expert (Figure 9).** Applied to the action expert's activations, the same pipeline decodes EE position best at layer 12 ($R^2 = 0.998$, 1.1 cm error). There, a general linear map predicts the displacements almost exactly (error 0.04) but a rotation with uniform scaling does not (0.29), so the subspace is a stretched rather than a rigid copy of world coordinates.

### B.4 Distance to Target Object

**Data.** We use the object-teleport dataset, keeping the 69,161 states in which the object is within 70 cm of the gripper (905 scenes; 8,707 states below 15 cm). Distance $d$ is the straight-line distance between the gripper and the object, read from the simulator. Activations are the VLM's final token at layer 21; the action expert's last layer (35) gives the same main results (Figure 14). The erasure test, the left panel of Figure 13, and the EE control below use only the first round of teleports, recorded before the extra near-range placements were added (52,217 states; 2,922 below 15 cm).

**Erasure (Figure 3a).** We split states into near (below 15 cm) and far (15–70 cm). Within each cross-validation fold, we fit a linear probe for distance on one range, remove its direction from all

activations, and repeat, fitting each new probe on the already-cleaned activations, up to 32 times. After each removal we refit a probe on each range and report its held-out $R^2$, computed against the variance within that range.

**Rulers and cutoff sweep.** The meter ruler is a ridge probe fit on all states to distance with every value below 15 cm set to 15 cm; the centimeter ruler is the mirror image, with every value above 15 cm set to 15 cm. Each is scored against the true distance within its own range. To judge whether two probes point in different directions, we compare the cosine between them with the cosine expected if both measured the same direction: for each probe, the cosine between two copies fit on disjoint halves of the houses (median of three random splits), taking the lower of the two probes' values. The cutoff sweep repeats this with the cutoff moved from 3 to 56 cm, fitting the pair either only on the states on each side of the cutoff (range-split) or on all states with values clipped at the cutoff (round-to-$c$), and compares both with a single probe fit on the whole range and then rescaled and shifted separately on each side of the cutoff.

**Controls.** The near axis (the direction of the near-range probe) is not a disguised copy of EE position: EE position predicts nothing about distance among near states ($R^2 \approx 0$), and less than 3% of the near axis falls within the subspaces that encode EE and target position. It does not detect contact either: the centimeter ruler's error is the same on states in which the object touches nothing but its supporting surface (1.40 vs. 1.41 cm). For the single-direction control, we build synthetic activations in which distance is written along one random direction, plus noise and a per-scene offset, and run the same probes on them.

### B.5 Direction to Target Object

We use the same data and layer as for distance. The target is the unit bearing $\boldsymbol{u}$ from EE to object in the base frame, decoded by ridge regression onto its three components. Error is the angle between decoded and true bearing, compared to a training-mean baseline ($32.6°$) and a scene-mean oracle ($18.1°$). Since the bearing is a nonlinear function of the two positions, we check that the model does not simply recompute it from them: the residual of $\boldsymbol{u}$ after regressing out $[\boldsymbol{p}_{\mathrm{EE}}, \boldsymbol{p}_{\mathrm{obj}}, d]$ remains decodable ($R^2$ of 0.3–0.5 per component), and the bearing probe has little energy in the position and distance subspaces (0.01–0.03). In Figure 15, decoded bearings are averaged in 6 cm cells of a target-centered frame, and the cutoff sweep follows the distance protocol.

### B.6 Contact

**Data and probe.** We record VLM activations by replaying 299 natural pick episodes. A replay does not exactly reproduce the original trajectory because the model runs in reduced precision (bf16), so each state is labeled from the replay itself: contact means the gripper's fingers touch the object with nonzero contact force. We average the layer-24 activations over the instruction tokens and fit a linear discriminant (LDA) for contact, with houses held out. It reaches 91.3% held-out accuracy against a 65% majority baseline (AUC 0.954); the final token alone gives the same accuracy (AUC 0.957). This probe is used for Figure 4 and for all tests below.

**Off-schedule interventions.** To create contact at moments it would not naturally occur, we pause 120 replayed episodes at fixed poses and move the object in small steps along the line to the gripper: toward the open gripper until it first touches (early touch), or away from a closed gripper holding it until contact breaks (induced slip). At each pose we count detection as correct if the probe's mean score is higher on the contact states than on the no-contact states; Figure 4b reports the fraction of poses scored correctly (chance 0.5), with the gripper in its natural state or forced closed during touches and open during slips.

### B.7 Phase

**Labels.** Each step is labeled from the simulator state: the gripper command, the object's height, and whether the object rests on the receptacle. Reach lasts until four steps before the first command to close the gripper. Grasp runs from there until transport begins, so failed grasps and retries count as grasp. Transport begins once the object, held after the final close, has risen 3 mm. Place begins

where the object starts its last uninterrupted descent (at least 1 cm) before being released onto the receptacle, or at the release if there is no such descent, and runs to the end of the episode.

**Decoding.** We fit a four-class linear discriminant (LDA) on the successful episodes, with houses held out. Balanced accuracy (the mean of the four per-phase accuracies; chance 0.25) is 0.92 for the action expert at layer 24, 0.86 for the VLM's image tokens of the two current camera frames at layer 31, and 0.87 for the VLM's non-image tokens at layer 29. The VLM's 3-D view (Figure 18) shows the top three LDA directions rather than principal components, because the VLM's first principal component (78% of variance) tracks the scene rather than the phase; the action expert's view in Figure 5b uses principal components.

**Freezing.** The policy sees two frames per query, taken eight steps apart. At landmark moments in each phase, taken from re-simulations of successful episodes, we query the policy twice at the same moment: once with its normal two frames (natural), and once with the current frame shown twice, so that the world appears to have stopped (frozen; "2-query pause" in Figure 20). Re-simulations that end in failure are kept. The decoded phase of both queries (Figures 5c, 18, and 19) comes from a separate four-class LDA fit on the landmarks' natural queries, holding out each landmark's house (876 landmarks from 122 episodes). The gripper effect comes from a second capture (900 landmarks from 126 episodes): the fraction of the eight executed steps that command a closed gripper, frozen minus natural (Figure 5d; per step in Figure 20).

### B.8 Wrist Orientation

**Roll sweep (Figures 1 and 6).** Starting from one arm pose, we set the wrist joint (Franka joint 7) to 39 evenly spaced angles across its range, keep all other joints fixed, and re-render both cameras at each angle. We record the action expert at layer 9, using the first action token at the final flow step, and plot the top two principal components of the 39 activations. For Figure 1, we run the same sweep in 64 houses, subtract each house's mean activation, and fit one PCA to all of them.

**Readout and precession (Figures 25 and 24).** The natural-rollout readout uses the first action token at the final flow step. In the precession sweep, each joint is kept within the range of the policy's training actions (1st to 99th percentile) and the hand within 5 cm of its starting position; in each direction, the orbit stops when the hand can no longer be held there.

**Steering (Figure 6b–d).** The autoencoder is a pair of small MLPs (768–512–256–32 and back), trained on layer-9 activations averaged over the action tokens, from controlled wrist-rotation sweeps with a third of the houses held out. Three pairs of its 32 code dimensions are each constrained to a circle, one per rotation axis, and the other 26 are unconstrained; we rotate only the pair for rotation about the vertical axis, where one degree on the circle corresponds to one degree of physical rotation on held-out sweeps. To rotate by $\theta$, we encode the current activation, rotate its point on that circle by $\theta$, and decode. The edit is the decoded rotated point minus the decoded unrotated one, so a zero-degree rotation gives no edit; it is added to all action tokens at layer 9 at the final flow step and is $19\% \pm 7\%$ of the activation's norm. One-shot edits of $\pm 20^\circ$ and $\pm 40^\circ$ are applied at a single query in the middle of a rollout, in eight scenes held out from training the autoencoder, and the response is read from the decoded 8-step action chunk; one scene in which the wrist rotates strongly on its own is excluded, leaving 28 conditions. Held edits are recomputed from the live activation at every query over 32-step rollouts in six held-out scenes (five of these seven plus the excluded one), and the rotation is measured relative to the unedited rollout of the same scene.

### B.9 Gripper Pullback

**Landmarks.** In 311 grasping episodes (170 houses), we take two landmarks relative to the moment the grasp becomes stable (contact held for three consecutive queries): a few queries before it (typically 3–9, gripper open) and a few after (gripper closed and holding the object). At each landmark we pick up to three evenly spaced queries whose executed gripper command is opposite the target, giving 1,697 edits. We cache each query's inputs so that the edits can be refit at layers 9, 18, 27, and 32 without re-simulating.

**Objective.** We fit an additive edit $\boldsymbol{\delta} \in \mathbb{R}^{768}$ to all 16 action tokens at the output of action-expert block $L$, at the final flow step only, with the model's weights fixed. Let $a(\boldsymbol{\delta}) \in \mathbb{R}^{H\times 8}$ be the decoded chunk ($H$=16), with seven arm channels $a^{\text{arm}}$ and a gripper channel $g$. We minimize

$$\begin{aligned}\mathcal{L}(\boldsymbol{\delta}) \;=\; & \tfrac{w_g}{H}\textstyle\sum_t \big(g_t(\boldsymbol{\delta}) - g^*\big)^2 \;+\; \tfrac{w_a}{7H}\textstyle\sum_{t,j}\big(a^{\text{arm}}_{t,j}(\boldsymbol{\delta}) - \bar{a}_j\big)^2 \\ & +\; \tfrac{w_s}{7(H-1)}\textstyle\sum_{t,j}\big(a^{\text{arm}}_{t+1,j}(\boldsymbol{\delta}) - a^{\text{arm}}_{t,j}(\boldsymbol{\delta})\big)^2 \;+\; \lambda\|\boldsymbol{\delta}\|_2^2,\end{aligned}$$

where $g^* \in \{+1, -1\}$ is the target gripper command (close, open) and $\bar{a}$ is the current joint configuration; the terms flip the gripper, hold the arm still, and keep the chunk smooth. We use $(w_g, w_a, w_s, \lambda) = (1, 300, 3000, 10^{-4})$, with $w_a$ and $w_s$ chosen by grid search at $w_g = 1$, and run Adam (learning rate 0.1) for 1,000 steps, keeping the lowest-loss edit. An edit succeeds if the gripper command, averaged over the chunk, has the sign of $g^*$; all measurements use the decoded chunk, which is not executed.

**Low-rank edits and leakage.** For the leakage curves we refit the same objective with the edit restricted to $\boldsymbol{\mu} + V\boldsymbol{w}$, where $\boldsymbol{\mu}$ and $V$ are the mean and top $k$ principal components of the edits from all other houses, at one query per landmark per episode. Leakage is the root-mean-square gap between the arm commands and the current joint angles over the chunk, reported as the median over queries relative to the median for the unrestricted 768-dimensional edits.

**Probe and steering field.** The aperture probe is a ridge regression, with houses held out, from each state's pre-edit activation, projected onto the top $k$ edit components, to the gripper command averaged over its decoded chunk. The field panel places pre-edit states on the top two components and averages the edits in each grid cell, separately for closing and opening edits.

### B.10 Real-Robot Analyses

**Data.** We use DROID (Khazatsky et al., 2024), giving each model one frame per camera at each query (no earlier frame). The first 100 episodes, queried every eighth step (3,321 queries), are used for Figures 8, 29, and 33; the wrist traces in the first two instead show the roll-sweep episode described below. A further 2,000 pick-and-place episodes, drawn at random from those with a single grasp-and-release cycle and at least six recorded queries after the release, are used for Figures 31 and 32. Action-expert activations are averaged over the action tokens at the final flow step. VLM activations are averaged over the image tokens, with each token first scaled to unit length for MolmoAct2 and MolmoBot; the MolmoAct2 distance rulers instead average the non-image tokens. Reported scores are 5-fold cross-validated with episodes held out. *Closure onset* is the first query at which the measured gripper position leaves zero, ignoring blips that stay below 30% of its holding value.

**EE position and wrist orientation.** Labels are computed from the recorded joint angles with the robot's kinematic model; the wrist angle is the gripper's rotation about the vertical axis (the last angle of its $xyz$ Euler decomposition). Probes read action-expert layer 16 of MolmoAct2 and MolmoBot and VLM layer 7 of $\pi_{0.5}$. For the roll sweep (Figure 30), an operator rotated joint 7 from $+69°$ to $-79°$ and back to $+84°$ in one episode while the other joints moved by up to $15°$; Figure 30 colors points by this joint angle. The $\pi_{0.5}$ action-expert panel uses one fixed noise seed for all queries, the third of the ten flow steps, and the first action token (with a new seed per query at the final step, no arc is visible).

**Distance to grasp.** We use the queries from the start of each episode up to closure onset, keeping episodes with at least three such queries, and define $d$ as the gripper's distance from where it is at closure onset. The rulers are fit as in App. B.4, on VLM layer 25 of MolmoAct2 and MolmoBot and layer 14 of $\pi_{0.5}$.

**Phase.** Labels come from the measured gripper position: reach until closure onset, grasp until the gripper reaches 90% of its holding value, transport while it stays there, and place for the six queries (3.2 s) after that; later queries are not scored. The bars in Figure 32 come from four one-vs-rest linear discriminants, one per phase, each fit on all other episodes.

**Gripper pullback.** We follow App. B.9, adding $\boldsymbol{\delta}$ after the final action-expert block, with 15 action tokens for MolmoAct2 and $\pi_{0.5}$ (whose activations are 1,024-dimensional) and a gripper weight of 16 for MolmoAct2 and 4 for $\pi_{0.5}$ and MolmoBot. MolmoAct2's gripper command lies in $[0, 1]$, so its targets are 0 and 1 and a flip means crossing 0.5; $\pi_{0.5}$ outputs joint velocities, so its arm target is zero velocity. All models share 379 landmarks: queries 3–9 before closure onset (gripper open) or 0–9 after (closing), at most two of each per episode. Rank-$k$ edits project each $\boldsymbol{\delta}$ onto the mean and top $k$ principal components of all 379 edits instead of refitting.

**Cross-model similarity.** MolmoBot is run on the same DROID queries. For each variable, we project each model's standardized activations onto that model's own ridge-probe directions for the variable, out of fold (aperture on the 379 landmark states; $(\cos\theta, \sin\theta)$ and $(x, y, z)$ on the 3,321 queries; the two rulers on the reach queries of the first 100 episodes). Similarity is one minus the Procrustes disparity between two models' coordinates on identical rows, after centering, scaling to unit norm, and the best rotation, reflection, and uniform scaling; the null repeats this with the target shuffled across rows. Every coordinate is fit to the same labels in each model, so part of the agreement is built in.

# C EXTENDED RESULTS

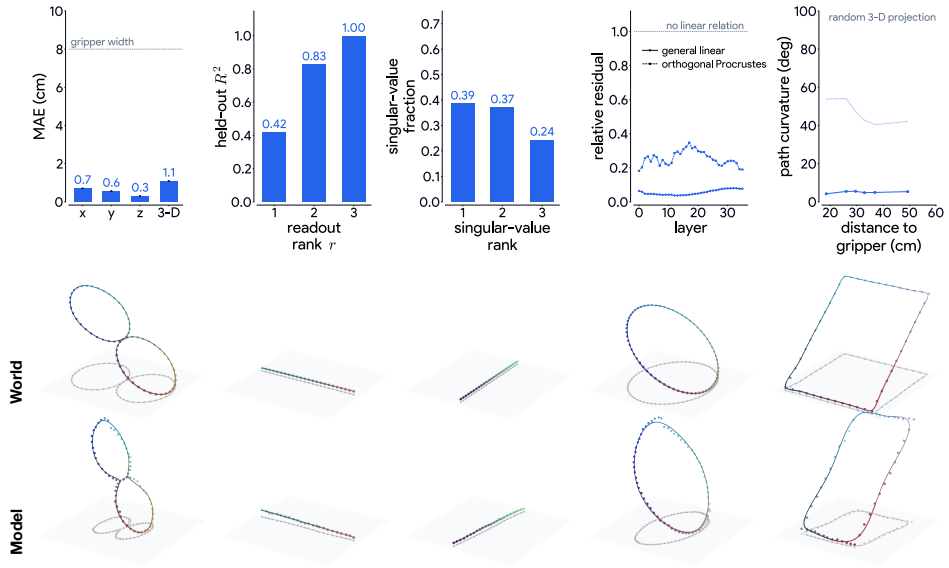


Figure 9: **The action expert (layer 12) also maintains a rank-3 EE position subspace.** Panels as in the top and middle rows of Figure 2, for EE position read from the action expert, which receives proprioception; the paths are from a single scene excluded when fitting the subspace. The readout is near-exact and the paths are recovered almost without distortion. The gap between the general-linear and orthogonal residuals shows the subspace is a scaled, not isometric, copy of world coordinates.

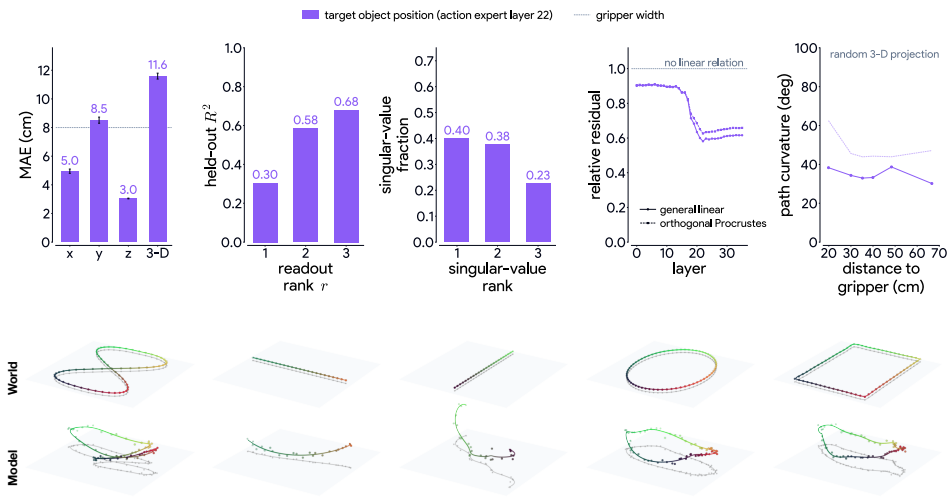


Figure 10: **The action expert (layer 22) also maintains a rank-3 target object position subspace.** Panels as in Figure 9, for the target object's position; the paths are from a single scene excluded when fitting the subspace. The action expert receives the object only through cross-attention to the VLM, and it carries target position with larger error than the VLM (Figure 2), recovering the paths with more distortion.

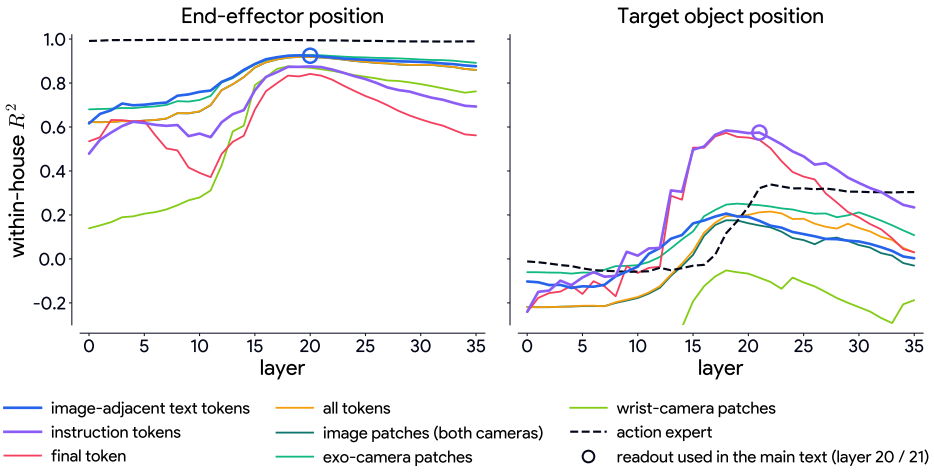


Figure 11: **Within a scene, target object position is computed on the VLM's instruction and final tokens between layers 12 and 21 and appears in the action expert from layer 17; EE position is present from the first layer in both.** Held-out $R^2$ of the rank-3 ridge readout of end-effector position (left) and target object position (right) at every layer, scored after removing each held-out house's mean from the true and predicted positions, so only position changes within a scene count; one line per VLM token pool, the action expert dashed, rings on the readouts of Figure 2, values below the axis floor clipped. The target's height is constant within a scene, so the target panel scores its horizontal position, and scores fall below zero where a readout's predictions vary within a scene without tracking the object. On the instruction and final tokens the target read stays at or below zero until layer 12 and peaks at layers 20–21, while the image patches carry little of it at any layer. The action expert tracks EE position near-perfectly at every layer and the target only from layer 17 on.

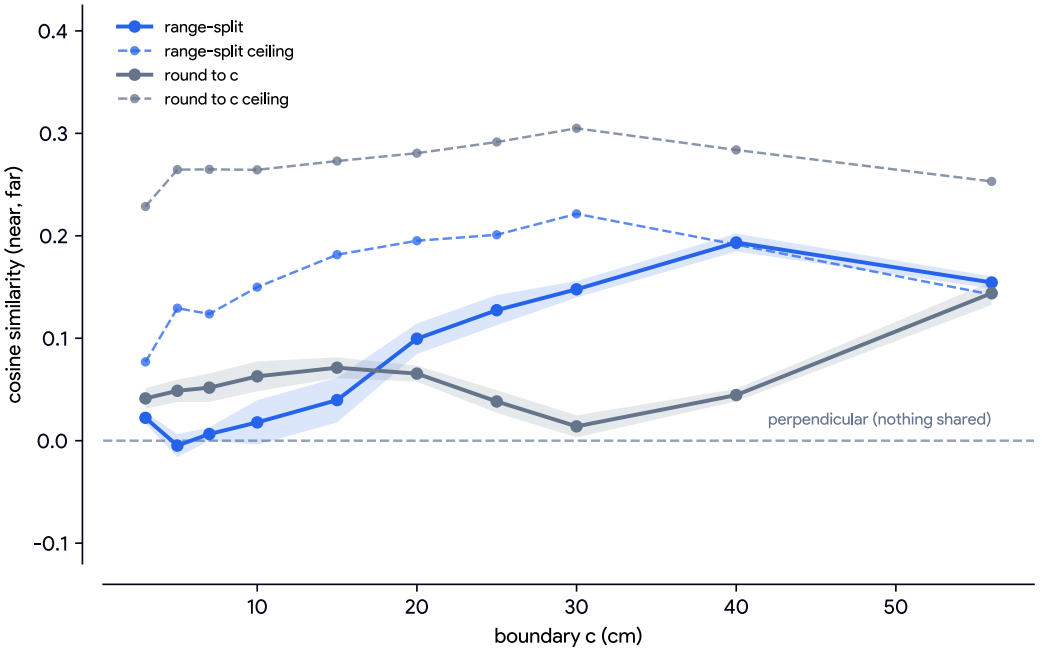


Figure 12: **Near and far distance probes are nearly orthogonal.** Per boundary $c$, a probe pair is fit either on close and far distances separately (range-split) or on all distances with targets saturated at $c$ (round-to-$c$); lines show the cosine between the pair's weight vectors (fold mean $\pm$ s.d.). Dashed: each pair's split-half ceiling — the cosine expected if both probes measured the same axis. At the 15 cm cutoff, cosines are $0.04$ and $0.07$ against ceilings of $\sim$0.18 and $\sim$0.27; the range-split cosine reaches its ceiling only at $c \gtrsim 40$, where the near fit degenerates to the global axis.

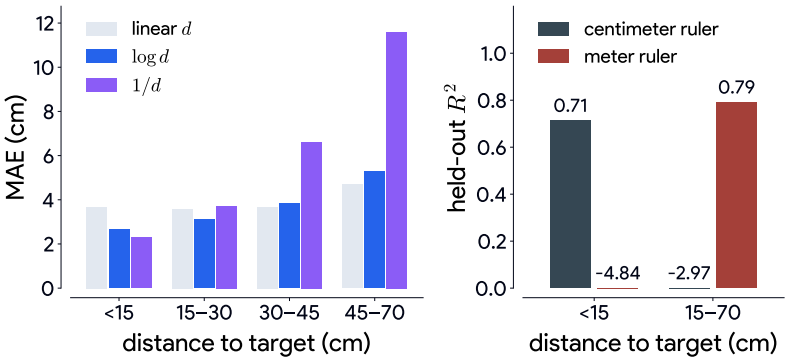


Figure 13: **Distance is compressive within the near range and closest to linear beyond 30 cm, and each ruler reads only its own range (VLM, layer 21).** (Left) Held-out mean absolute error per bin of true distance for a single full-range ridge probe fit to $d$, $\log d$, or $1/d$ and mapped back to metres; the compressive links have the lower error below 15 cm and the linear link in the farthest bins. (Right) Held-out $R^2$ against true distance for the centimeter ruler and the meter ruler of Figure 3, scored separately on states below and above that cutoff; negative (saturated) values are drawn at zero and printed. Each ruler saturates outside its own range, as does a single unclipped metre-scale probe (not shown) on the near-range states.

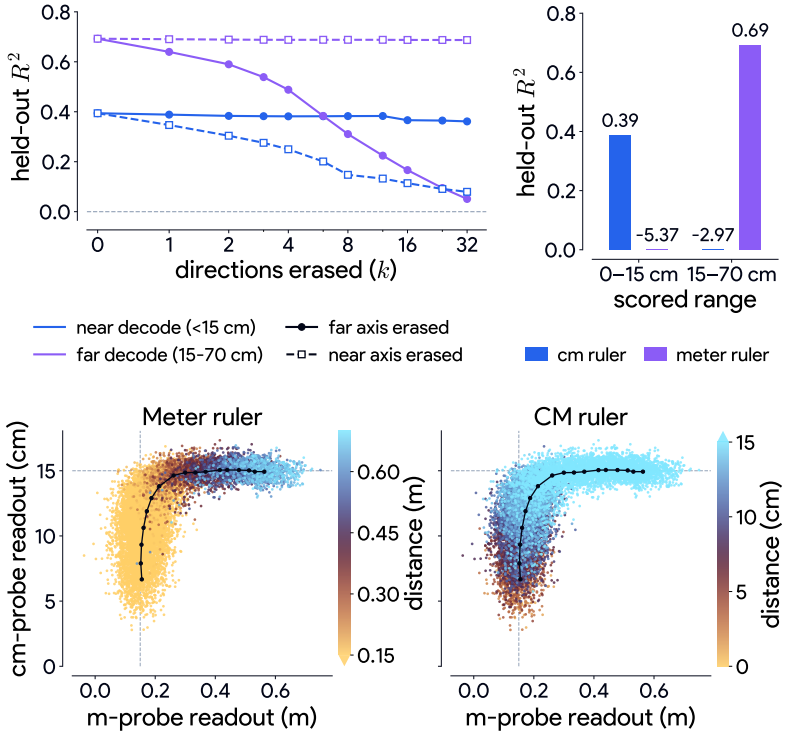


Figure 14: **The action expert (layer 35) reproduces the two-subspace distance code.** Same store, folds, and constructions as Figure 3, read from the action expert instead of the VLM. (Top left) Held-out $R^2$ of the near and far decodes as far-axis (solid) or near-axis (dashed) directions are erased; each decode drops only when its own axis is removed, and the near decode retains a $\sim$20% floor of its $R^2$ that the VLM cell does not show. (Top right) Held-out $R^2$ of the centimeter and meter rulers on each side of the 15 cm cutoff; each ruler scores only its own range, and bars below zero are drawn at zero with the value printed. (Bottom) The two ruler readouts plotted against each other, colored by true distance on the meter (left) and centimeter (right) scales; the meter ruler resolves the far arm and the centimeter ruler the near arm, as in the VLM cell.

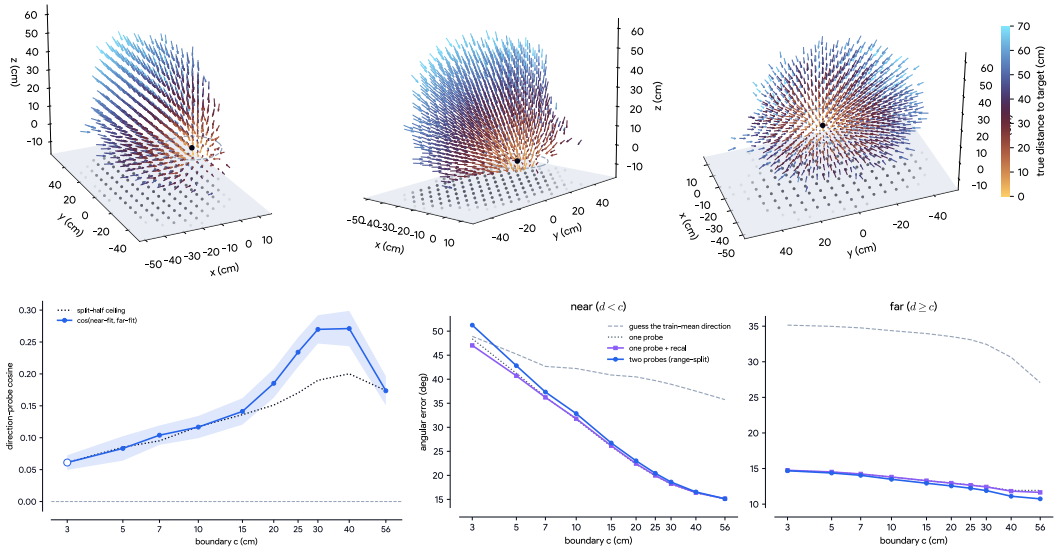


Figure 15: **Direction to target object is observable with one direction, in contrast to the two distance rulers. Top:** held-out decoded directions (VLM layer 21), averaged in 6 cm cells in a target-centered frame; arrow length is cell coherence. Decoded directions point toward the target throughout the workspace (median angular error $11.5°$ vs. a $32.6°$ floor); within 15 cm, per-read error increases but cell means remain accurate (coherence $0.96 \rightarrow 0.52$). **Bottom left:** as in Fig. 12, but for direction probes: the near/far cosine matches its ceiling at every boundary ($0.14$ vs. $\sim 0.13$ at $c = 15$). The near- and far-fit probes recover the same axes. **Bottom right:** the range-split pair does not outperform a single recalibrated probe on near reads (up to $4°$ worse) and improves on it by $< 1°$ on far reads — a single probe suffices at all ranges.

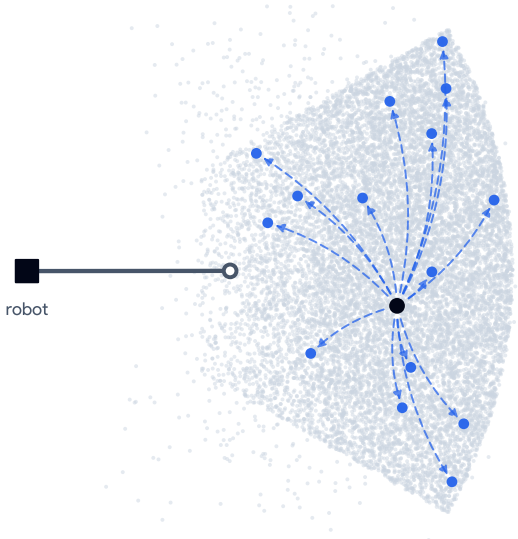


Figure 16: **Object teleportation across the workspace for distance to target object experiments.** Object placements in the robot-base frame, viewed from above (robot at left, forward to the right). Gray points are all placements in the teleport datasets (906 houses). For one example house, the black point is the object's natural placement and the blue points its 15 teleported placements, sampled uniformly over the supporting surface within the benchmark placement envelope; dashed arrows connect the natural placement to each teleport. Only the object is moved; the arm's initial pose is unchanged.

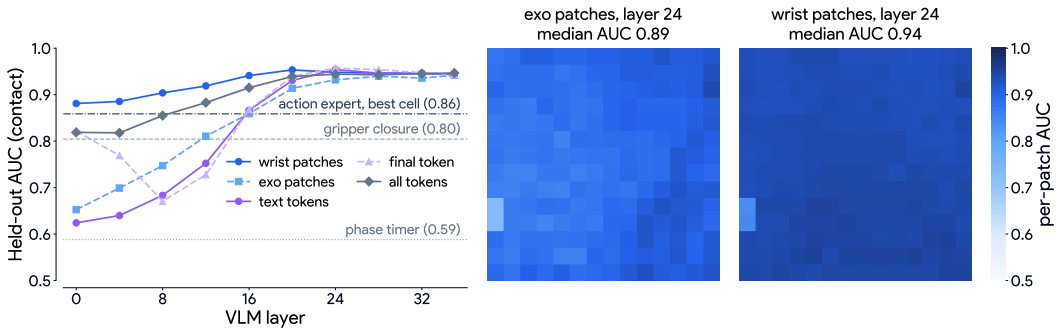


Figure 17: **Contact is decodable from the wrist-camera patches at the first layer and from the other tokens only from the middle layers on (VLM).** (Left) Held-out AUC of the contact discriminant at each layer, fit separately on five token groups: wrist-camera patches, exo-camera patches, text tokens, the final token, and the whole-sequence mean; reference lines mark a phase-timer baseline, a gripper-closure baseline, and the action expert. (Right) Per-patch AUC at layer 24 for the two cameras: the signal is spread over the whole image rather than confined to the fingers.

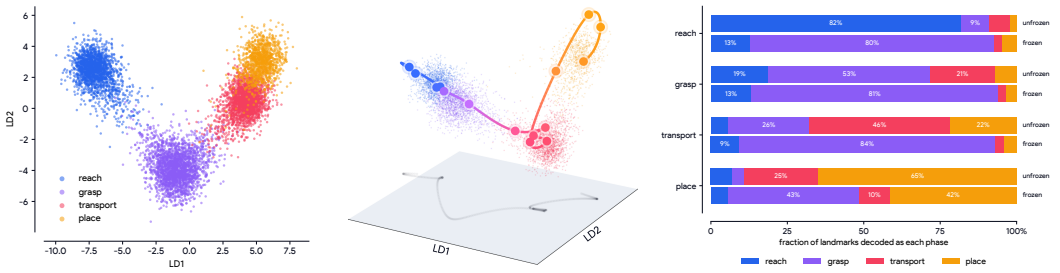


Figure 18: **MolmoBot's task phase subspace in the VLM backbone (vision tokens, Layer 31).** (Left) LDA projection of natural rollouts, colored by phase (0.86 balanced accuracy across held-out houses; AE: 0.92). (Middle) The full three-dimensional LDA subspace, with a held-out episode's trajectory traversing the four phase regions in order. (Right) Freezing shifts the VLM's decoded phase to grasp in every phase (place only partially), whereas the action expert's frozen transport and place reads shift to reach (Figure 5).

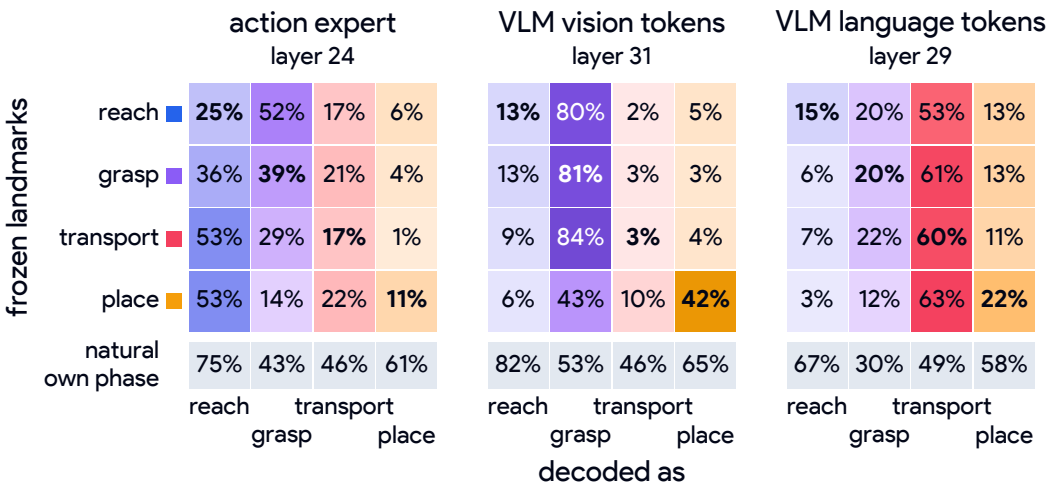


Figure 19: **Under the frozen read, each stream's decoded phase collects in a different phase: grasp for the VLM vision tokens, transport for the VLM language tokens, and reach for the action expert's transport and place landmarks.** Left to right: action expert, VLM vision tokens, VLM language tokens. Each row gives the percentage of one phase's landmarks decoded as each phase when the policy is shown the current frame twice (bold: the true phase); the strip beneath each matrix gives the percentage decoded correctly with the usual two frames, at the same moments.

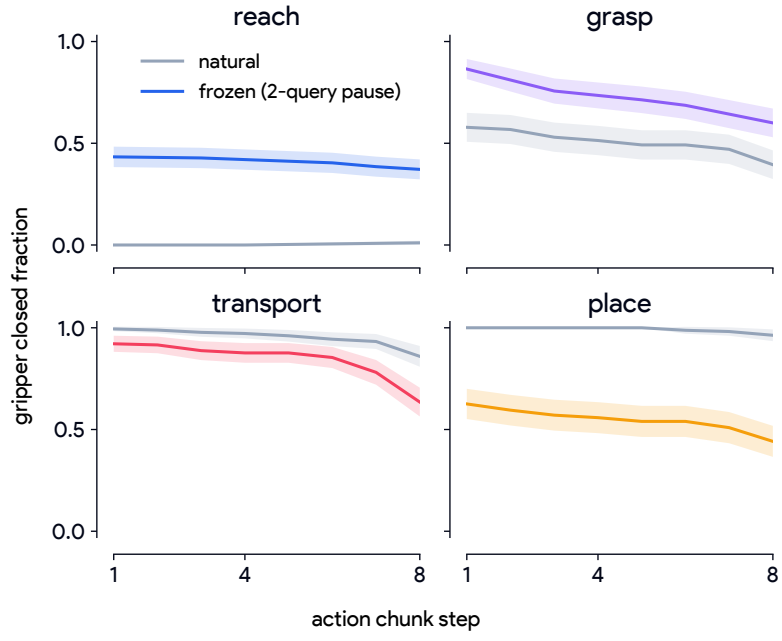


Figure 20: **Per-step time course of the freezing effect on gripper commands.** Fraction of chunks commanding a closed gripper at each step of the executed action chunk (shaded: 95% CI), natural vs. frozen, per phase.

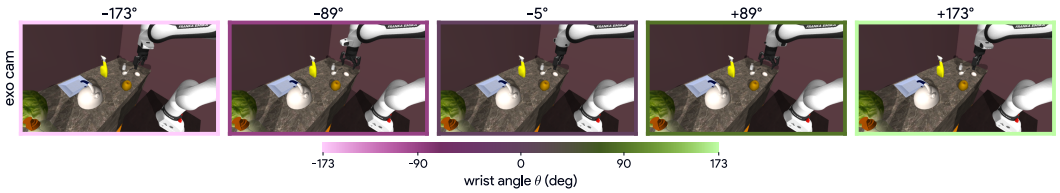


Figure 21: **Wrist roll sweep (Figure 6).** Exo (third-person) camera images at several points of the sweep through the wrist's full roll range ($-172.8°$ to $+172.8°$), with all other joints fixed. Borders are colored by wrist angle $\theta$.

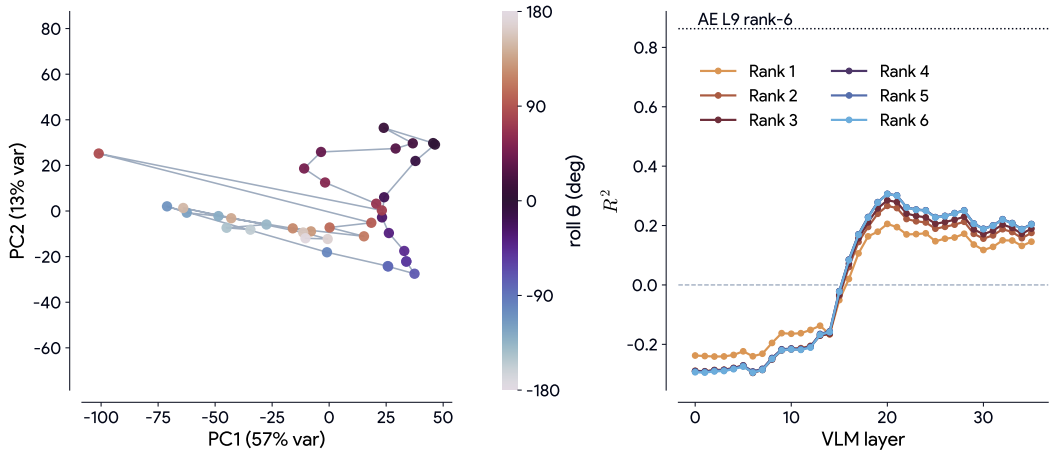


Figure 22: **The VLM maintains only a weak, pixel-derived trace of wrist orientation.** The same roll sweep as Figure 6, but we study the VLM activations instead; the VLM receives only images and text as input. (Left) PCA over the sweep activations (layer 35). The trace wraps — $-173°$ and $+176°$ land close together does not form a clean circle. PC1 is likely dominated by the visual response and not the rotation angle. (Right) The reduced-rank regression of Figure 6 at every VLM layer. Decoding peaks mid-stack at $R^2 = 0.31$ (layer 20), a third of the action expert's 0.86 (dotted) on the same data.

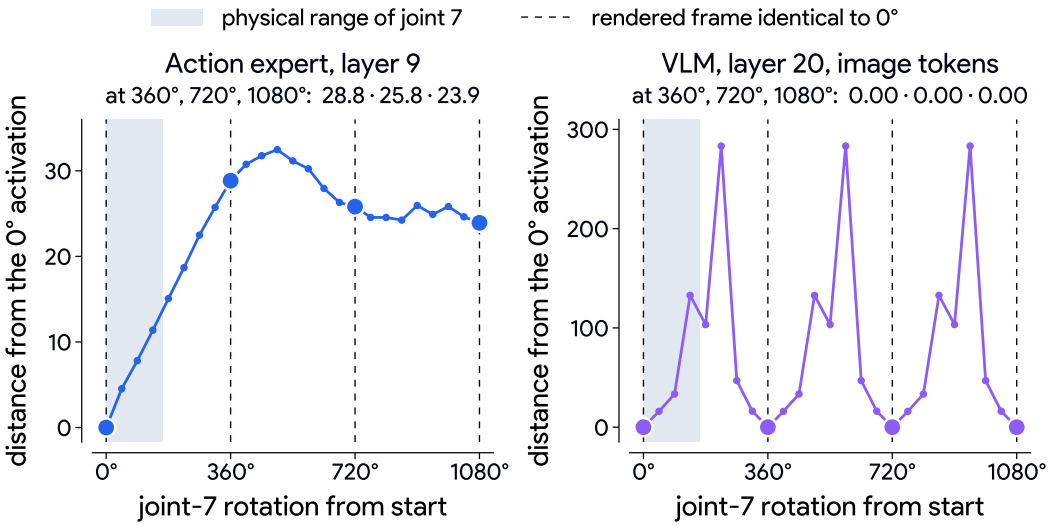


Figure 23: **After a full turn of the wrist joint the action expert's activation does not return to its start; the VLM's returns exactly.** From one pose of the roll sweep (App. B.8; a single house), the wrist joint (Franka joint 7) is driven through successive full turns by writing the joint value directly into the simulator, far past the joint's physical range (shaded band), so the sweep tests the models' inputs rather than a reachable motion. At every full turn (dashed lines, large markers) the rendered frames and text tokens are identical to the start and only the raw joint value differs, and only the action expert receives that value. **(Left)** Action expert, layer 9, first action token: Euclidean distance of the activation at each step from the start activation; after one turn it is 28.8 units away and does not return over the following turns. **(Right)** VLM, layer 20, mean over image tokens: the same distance, which varies within each turn and is exactly 0 at every full turn; the two panels are in different activation units.

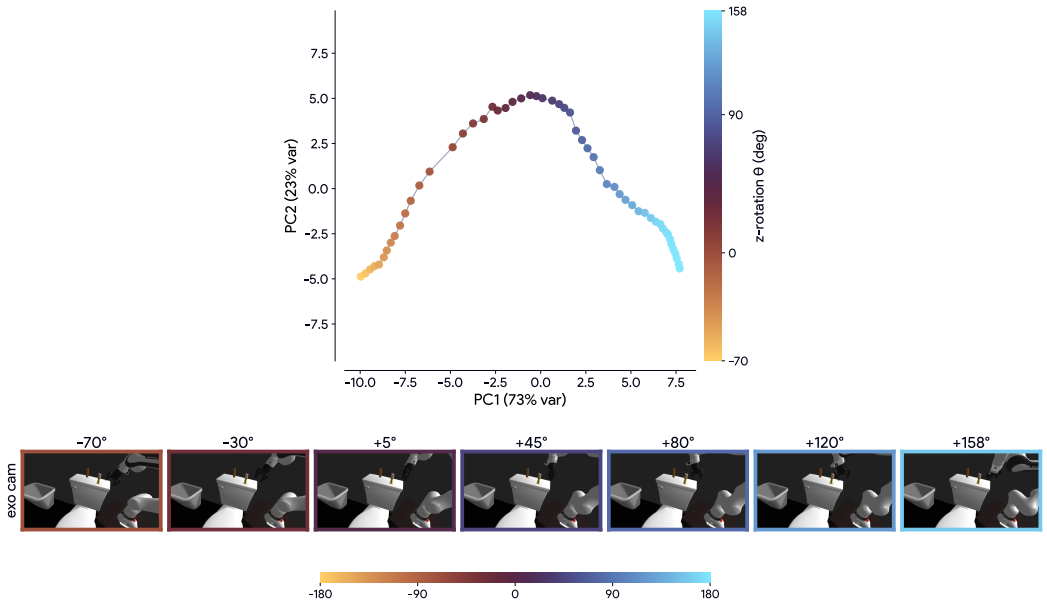


Figure 24: **Even the most circular reachable wrist rotation does not close.** Instead of rolling about the gripper's approach axis, we tilt the gripper $30°$ off that axis and orbit the tilt about the vertical. If driven through a full turn, this precession returns the wrist orientation exactly to its start (a closed circle in $SO(3)$). The gripper position is held fixed while the whole arm reconfigures, and we record activations at each step. The arm's joint limits stop the orbit partway: sweeping both directions from the start covers $-70°$ to $+158°$ ($228°$ of the turn; angles are the realized rotation about the vertical axis). (Top) PCA over the sweep activations. Precession remains an open arc, with PC1 carrying $73\%$ of the variance and PC2 $23\%$. (Bottom) The exo camera image at different rotation angles along the sweep.

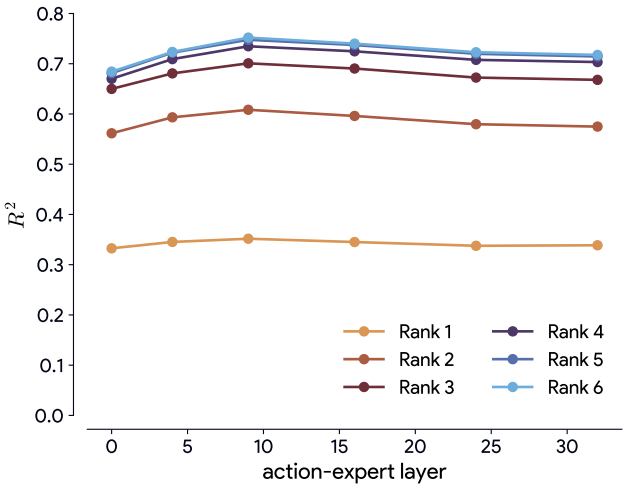


Figure 25: **Full 3-DOF wrist orientation is decodable from natural rollouts at low rank (AE).** Unlike Fig. 6, which rotates a single roll joint and reads the code with PCA, here "wrist orientation" means the complete 3-DOF orientation of the hand, represented by the first two columns of its rotation matrix (a 6-D target). We fit reduced-rank ridge regressions of rank 1 to 6 from the activations to this target on natural rollouts (6,555 timesteps, 191 houses) and report $R^2$ on held-out houses at each action-expert layer. Rank 2 recovers most of the decodable variance and rank 3 nearly all of it; higher ranks add little. Decodability peaks at layer 9.

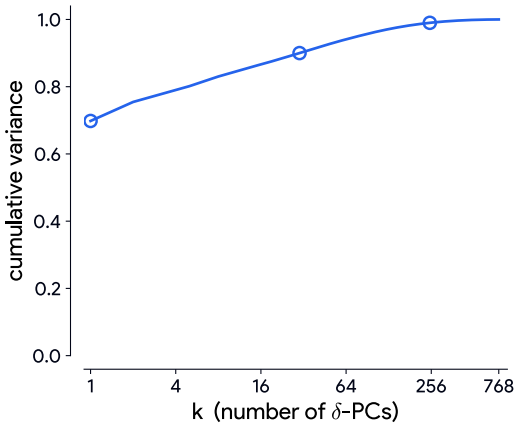


Figure 26: **Cumulative variance of gripper steering vectors.** Cumulative explained variance from PCA on the pullback steering vectors $\delta$ (App. B.9) as a function of the number of principal components. The first component captures $\sim$70% of the variance, complementing the per-component view in Figure 7.

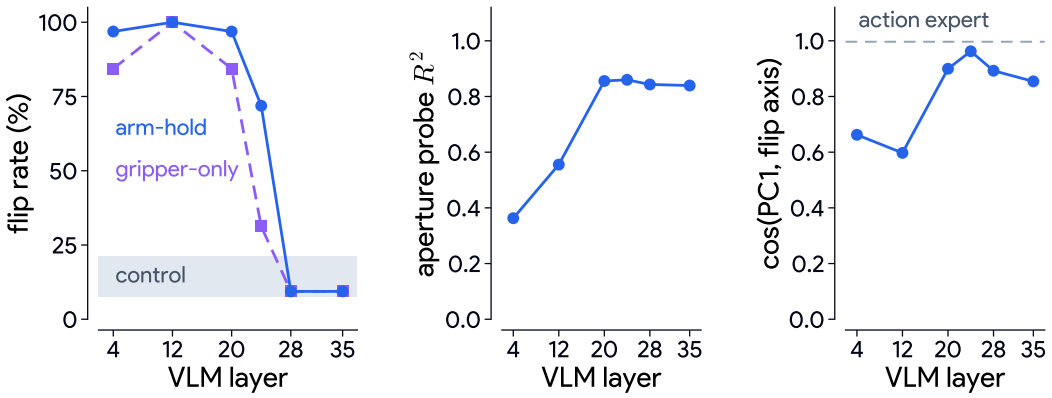


Figure 27: **In the VLM backbone, gripper steerability falls to the control floor by layer 28 while aperture readability rises.** The gripper pullback of App. B.9 with the edit $\delta$ added to every sequence token after VLM block $L$, at screen scale (a small edit catalog per layer and objective). (Left) Fraction of edits that flip the commanded gripper under the arm-hold objective of App. B.9 and under the gripper-only objective (the arm-hold and smoothness terms dropped); the gray band spans the flip rates of a control edit that does not flip the gripper (mean of the action-expert catalog, App. B.9). (Middle) Held-out $R^2$ of a ridge probe from the pre-edit activation to the commanded aperture, best of the token pools at each layer; readability rises to 0.86 over the same layers. (Right) Absolute cosine between the first principal component of the arm-hold edits, a sample-limited estimate from the small per-layer catalog, and the close–open flip axis (dashed: the action-expert catalog of Figure **??**); at the shallowest flipping layers (4 and 12) the leading component is only partly aligned with the flip axis.

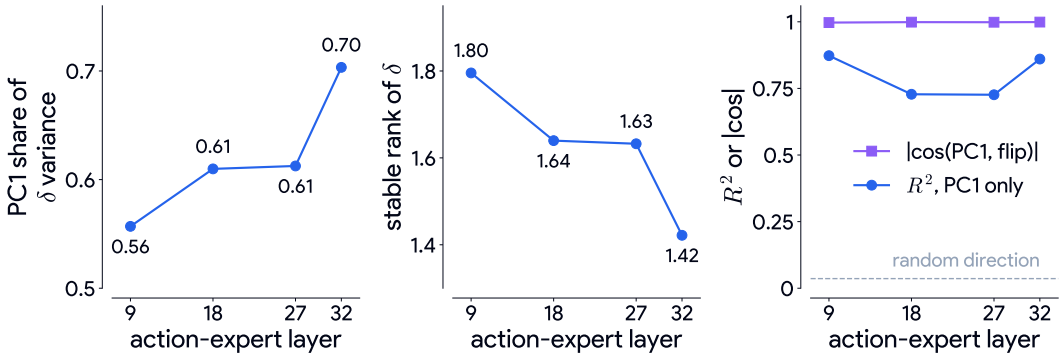


Figure 28: **The gripper steering vectors stay one-dimensional and aligned with the flip axis at every action-expert layer.** The pullback catalog of App. B.9, re-fit at layers 9, 18, 27, and 32. (Left) Share of steering-vector variance on the first principal component. (Middle) Stable rank of the steering vectors. (Right) Cosine between the first principal component and the flip axis (mean close edit minus mean open edit), and held-out $R^2$ of the aperture probe restricted to that component; dashed: a random direction.

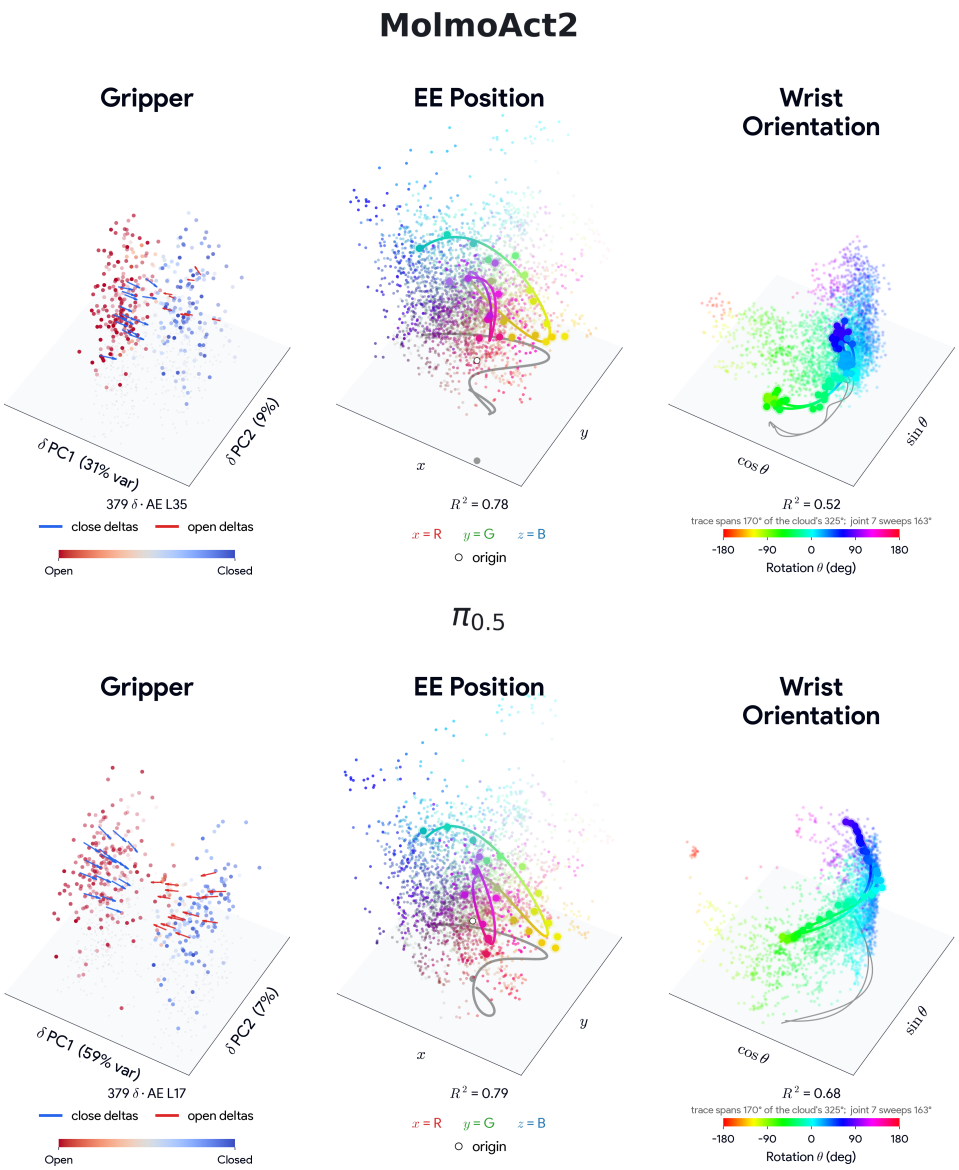


Figure 29: **DROID real world behavioral manifolds.** Here we show the behavioral manifolds realized by MolmoAct2 and $\pi_{0.5}$. Each point clouds shows the activations of 3,000 timesteps sampled from each models final action-expert block. For EE position and wrist orientation $\pi_{0.5}$ and MolmoAct2 activations are sampled from AE L16 of 36 and VLM L7 of 18 demonstrating the structures appear across multiple streams. We plot a natural episode trajectory in the EE position and Wrist Orientation subplots. In the Gripper plot we project pullback steering vectors into the PCA space.

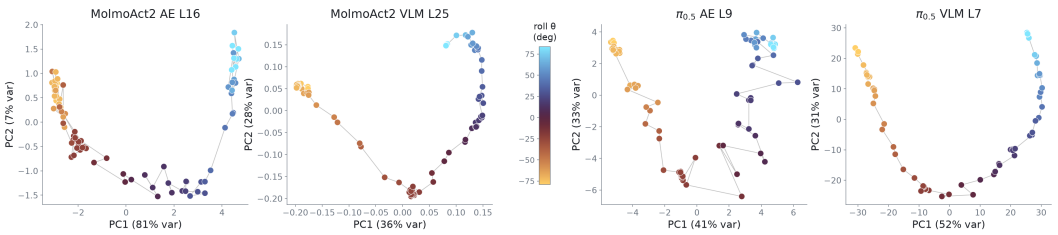


Figure 30: **Wrist orientation PCA across models and streams.** Following Fig. 6, we record a real-robot episode in which the wrist roll joint is swept from $-79°$ to $+84°$ while the other joints and the scene are held approximately fixed, and plot the top two principal components of the activations for MolmoAct2 and $\pi_{0.5}$ in both the VLM and the action expert (color: roll angle). As in MolmoBot, the sweep traces an open, semicircular arc. Because real-world data are noisier, the trajectories are considerably less smooth.

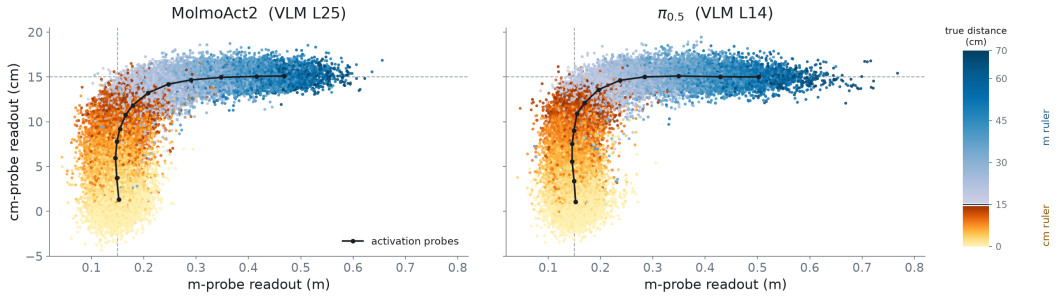


Figure 31: **MolmoAct2 and $\pi_{0.5}$ encode distance to target object in 2 orthogonal subspaces.** We find that similar to MolmoBot, as shown in Fig. 3 the tested DROID model also encode distance to target object with a multi-scale ruler. We sample 26,675 timesteps from 1,998 pick and place episodes. The distance is how far the gripper is from its initial grasp location. Activations are sampled from MolmoAct2 VLM L25 of 36 and $\pi_{0.5}$ VLM L14 of 18.

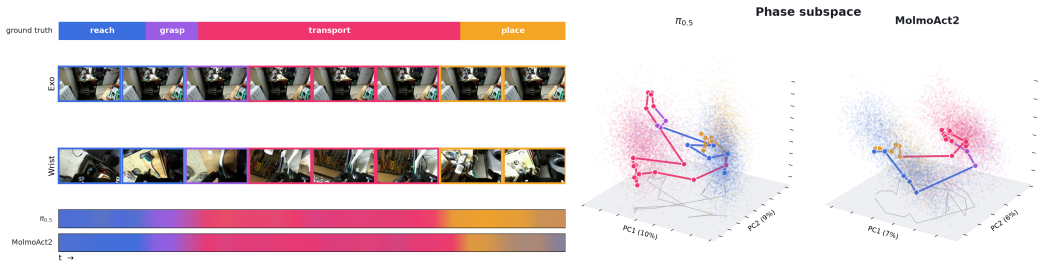


Figure 32: **Task phase subspace in $\pi_{0.5}$ and MolmoAct2.** (Left) Example trajectory for a simple pick-and-place task, "Pick up the white and green brush from the table and hang it on the wall," along with the predicted task phases. The ground-truth bar is derived from the measured gripper aperture alone, using a simple heuristic: reach until the gripper starts to close, grasp while it closes, transport while it stays closed, place after it opens. A linear classifier is trained on the top 48 principal components of activations collected across 2,000 episodes. (Right) The recovered task subspace in both models, colored by phase. Activations are taken from the $\pi_{0.5}$ action expert at layer 9 of 18 and the MolmoAct2 action expert at layer 22 of 36.

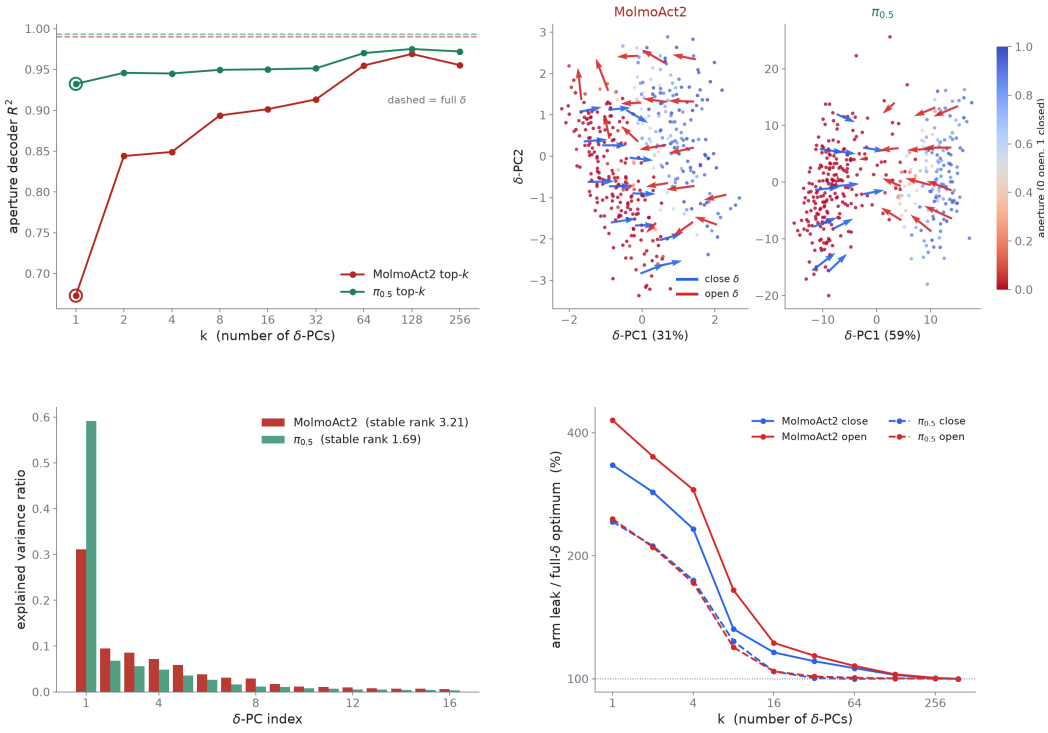


Figure 33: **Gripper steering vectors on real DROID episodes**. (Top left) A linear probe decodes gripper aperture from the top-k principal components of the steering vectors. Dashed lines: each model's full-activation ceiling. (Top right) Activations colored by gripper state (red open, blue closed); the steering vectors point from each cluster toward the other. (Bottom left) One component captures 59 % of $\pi_{0.5}$'s steering-vector variance (stable rank 1.69) but only 31% of MolmoAct2's (stable rank 3.21).(Bottom right) One component flips the gripper (94–97% of attempts, both models) but leaks into the arm joints at 2.4–4.3× the full-$\delta$ optimum. Activations are recorded and steered from the penultimate layer of each AE.

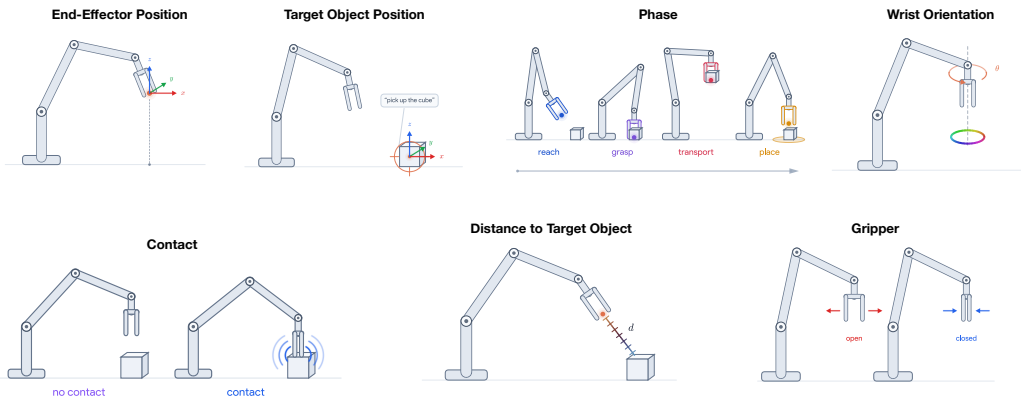


Figure 34: **The seven task variables.** An overview of each variable we study.

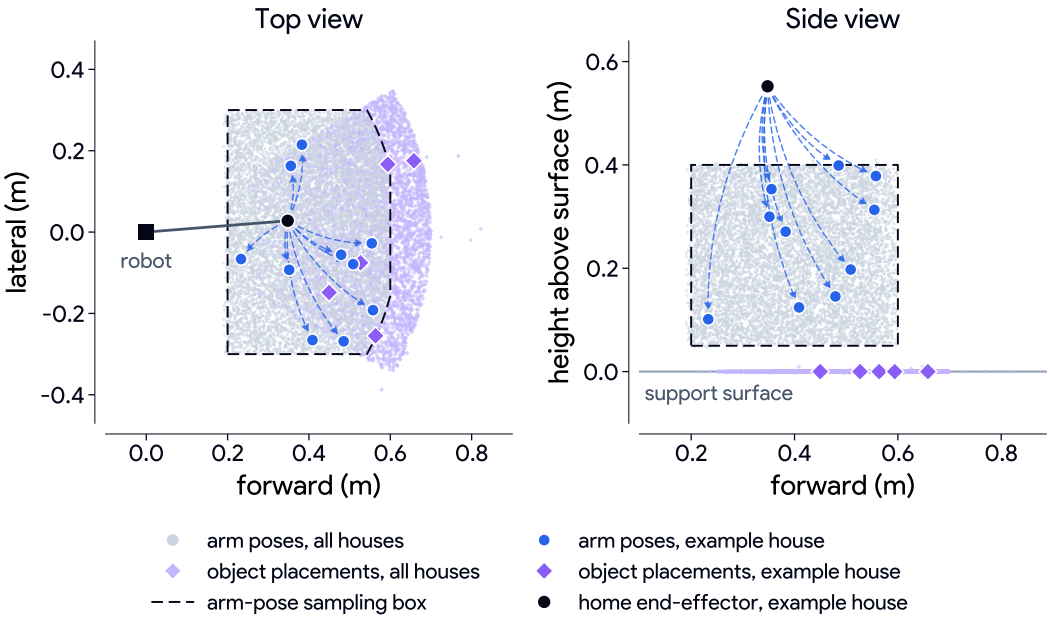


Figure 35: **Arm and object placements in the crossed teleport harvest used for EE and target position.** Robot-base frame viewed from above (left) and from the side, with height above the table (right). Gray points are arm poses and violet diamonds object placements across all houses; the dashed outline is the box arm poses are sampled in. For one example house, the black point is the home pose, blue points its ten arm poses, and violet diamonds its five object placements; unlike Figure 16, both the arm and the object move.

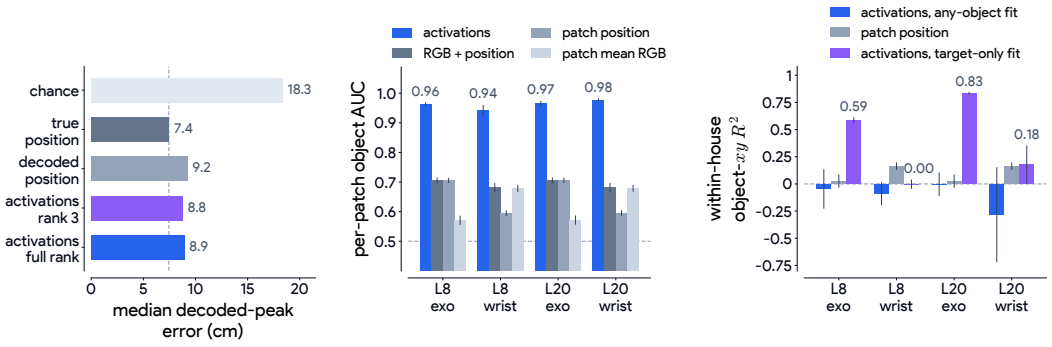


Figure 36: **The VLM keeps the target's position but no map of the workspace; other objects are encoded per image patch.** (Left) A table-occupancy map of the target, read from VLM activations (last token, layer 21), is no more accurate than the same readout applied to the target's three true coordinates (dashed). (Middle) One readout shared across all image-patch tokens detects whether an object the instruction does not name is visible in that patch, well above patch-position and patch-color baselines (exo and wrist cameras, layers 8 and 20). (Right) With scene means removed, a readout of table position fit on all visible objects does not track the target as it moves, whereas one fit on the instructed target alone does.