

Lost in Reconstruction: Aligning Action Representations with Language in Vision-Language-Action Models

Li Wenjie* Yash Jangir Ignacy Stepka Yash Agarwal Marion Kipsang Yonatan Bisk
Carnegie Mellon University

Abstract

Action verbs describe not only the physical outcomes of actions, but also how those actions are performed. Yet action representations in vision-language-action models (VLAs) are typically optimized for reconstruction under L1/L2 losses in raw action space, where numerical proximity need not reflect linguistically meaningful distinctions. On BridgeV2, we show that action trajectories contain verb-grounding information beyond visual state changes, and that reconstruction-only discrete tokenization systematically erodes this information. To address this problem, we introduce **SALT**, a **S**emantically **A**Ligned action **T**okenizer that augments a VQ-VAE-style tokenizer with an auxiliary objective requiring a frozen vision-language model to recover the episode instruction from quantized action latents. Policies trained with SALT achieve 71.9% average success in SimplerEnv, compared with 42.7% for a reconstruction-only VQ-VAE tokenizer and 31.2% for FAST. SALT also develops verb-specialized codes while maintaining reconstruction fidelity. These results show that robot action trajectories provide a source of language grounding and that preserving this structure in action representations can substantially improve language-conditioned control.

1 Introduction

How is language grounded in embodied experience in the world? How much of it can be explained by visual input alone? In particular, do the action modality provides extra grounding signals for verbs? Verb meanings reflect regularities in how agents act in the world, such as patterns of motion, contact, and resulting changes in physical state. In turn, language shapes how actions are represented: which variations are grouped as instances of the same action class and which distinctions are treated

as consequential. An action representation should not only be expressive enough to support execution, but also preserve linguistically meaningful distinctions.

However, action representations in existing vision-language-action models (VLAs) (Brohan et al., 2023b,a; Kim et al., 2024; Black et al., 2024) are not explicitly designed to be aligned with language abstractions. VLAs are trained with objectives and design choices defined primarily over the Euclidean physical control space, while their relationship to language is left for downstream policy training to infer. The implicit assumption is that having low action-space loss in predicting a trajectory also preserves the language grounding signals that connect the trajectory to the instruction that describes it.

This omission reflects how VLAs are constructed. VLAs adapt foundation vision-language models (VLMs) (Vinyals et al., 2015; Karpathy and Fei-Fei, 2015; Alayrac et al., 2022) for robot control by conditioning action prediction on visual observations and natural-language instructions. This paradigm allows robot policies to inherit representations already aligned across vision and language. In many existing VLAs and language-conditioned robotics literature, language serves primarily as goal conditioning. In particular, through attributes such as color, shape, or location, it usually identifies the object to be manipulated and specifies its desired location or physical state (Shridhar et al., 2021; Jang et al., 2022; Mees et al., 2022; Brohan et al., 2023a; Liu et al., 2023). Because these aspects of an instruction can often be grounded in visual observations, alignment research in VLAs has concentrated largely on connecting referring expressions to objects and scenes (Shridhar et al., 2021; Brohan et al., 2023a; Kim et al., 2024) and shaping visual representations through language supervision (Nair et al., 2022; Karamcheti et al., 2023; Ma et al., 2023). Action representations,

* Correspondence to wenjie12@andrew.cmu.edu.

General properties		Example episode		
Verb	Motion profile	Instruction	Action trajectory	Visual state change
reach 	- Direct, near-straight-line trajectory - Net descent toward the target - One-way trajectory (no return)	"end effector reaching salmon"		 → 
pick up 	- Motion concentrated along a single axis - Narrow horizontal range - Mostly vertical (lift or drop)	"pick up violet Allen key"		 → 
push 	- Gripper stays open most of the episode - Gripper closes late (or not at all) - Brief, motionless closed phases	"push the can to the lower right side of the table"		 → 
turn 	- Stays planar (almost no vertical motion) - Direct, near-straight-line trajectory - Gripper closes late (or not at all)	"turn the spoon parallel to the table edge"		 → 
sweep 	- Long sustained motion while gripping - Travels much farther than other verbs - Long episode duration	"sweep into pile"		 → 
close 	- Long travel with gripper open - Gripper stays open most of the episode - Gripper closes late (or not at all)	"close the drawer"		 → 

Figure 1: **Verbs describe both motion dynamics and action goal.** Six BridgeV2 verbs with their characteristic motion profile (trajectory shape, axis, gripper timing) and one example episode each (instruction, 3D end-effector trajectory, first/last frames). Profiles are derived from action trajectories of the verb, using pre-defined formulas. Recipe in (Appendix B).

which turn a VLM into a VLA, and their alignment with linguistic abstractions have received comparatively little attention.

We show that this mismatch creates a consequential bottleneck at the action interface of a VLA. Robot trajectories can be represented continuously (Black et al., 2024), or discretely as action tokens (Brohan et al., 2023b). Discrete tokenization is especially common because it allows VLAs to reuse the autoregressive next-token prediction interface of pretrained VLMs. Moreover, it is sometimes used during pretraining before a model transitions to a continuous action head (Black et al., 2024). Representative discrete action tokenization approaches include per-dimension **Bin** tokenization, used by RT-1, RT-2, and OpenVLA (Brohan et al., 2023b,a; Kim et al., 2024); learned **VQ-VAE**-style tokenizers, optimized primarily for trajectory reconstruction (van den Oord et al., 2017; Wang et al., 2025; Mete et al., 2024; Liu et al., 2026); and **FAST**, which transforms trajectories into frequency coefficients and compresses them through byte-pair encoding (Pertsch et al., 2025). In common practice, the tokenizer and the VLA are trained separately: tokenizer training determines how continuous actions are represented as discrete tokens, while VLA training determines, given vision and language, which token to predict. The action vocabulary is fixed before language enters the pipeline,

and none of these approaches explicitly requires it to preserve linguistically meaningful distinctions.

Studying alignment between language and action representations requires realistic actions paired with naturalistic human language. We use BridgeV2 (Walke et al., 2023), a dataset of tele-operated real-robot manipulation trajectories annotated with free-form natural-language instructions. While bounded by the affordances of a WidowX robot arm and its 7-Degrees-of-Freedom (DoF) control space, BridgeV2 provides human-like action and language use, which most existing scripted pick-and-place robotics demonstrations do not provide. On this dataset, we establish three findings:

(i) **Action trajectories contain verb-grounding information that visual outcomes alone do not capture.** We decompose verb information into *action goals* (the visual change between episode endpoints) and *motion dynamics* (the 7-DoF action trajectory). Each carries verb information not captured by the other as verbs are grounded not only in resulting state changes, but also in motion profiles, such as contact patterns and gripper timing (Section 2). This demonstrates that grounding language for control requires alignment with both vision and action.

(ii) **Reconstruction-only action tokenization systematically erodes verb-grounding signals.** Across Bin, VQ-VAE, and FAST tokenizers, mutual information between verbs and tokenized ac-

tion representations decreases, with larger losses under stronger compression (Section 3). Downstream language-conditioned policy training does not fully recover the missing structure. These results identify the discrete action interface as a bottleneck between language and control.

(iii) Semantically aligned tokenization improves task performance and produces more linguistically meaningful action representations. We introduce **SALT**, a Semantically **AL**igned action **T**okenizer that augments VQ-VAE training with an auxiliary instruction-generation objective (Section 4). During tokenizer training, quantized action latents are fed to a frozen pretrained VLM, which must generate the corresponding episode instruction. This supervision encourages actions described by similar language to receive similar representations while still preserving reconstruction fidelity in continuous control space.

In **SimplerEnv** (Li et al., 2024), we empirically demonstrate that SALT substantially improves both semantic alignment and closed-loop task success (Section 5). Policies trained with SALT achieve a 71.9% average task success rate, compared with 42.7% for a reconstruction-only VQ-VAE tokenizer and 31.2% for FAST. Although the alignment objective does not explicitly identify verbs, the learned vocabulary becomes organized around verb-relevant distinctions: individual codes become highly selective for action classes such as *flip*, which reconstruction-only tokenizations distribute across generic, semantically mixed codes.

Together, these results support a general design principle for language-conditioned control: **action representations should preserve not only executable trajectories but also the abstractions through which language represents action.**

2 Diagnostic 1: Verbs Describe Both Action Goals and Motion Dynamics

Language instructions can specify both an *action goal*, which is the desired change in the physical environment, and the *motion dynamics* through which that change is produced (Hwang et al., 2025; Zhang et al., 2026). This distinction parallels a long-standing observation in lexical semantics: verb meanings lexicalize *manner* — how an action is performed — and/or *result* — the change it brings about (Talmy, 1985; Levin and Rappaport Hovav, 1991; Rappaport Hovav and Levin, 2010). We operationalize the action goal as the visual change

Table 1: **Action goals and motion dynamics provide complementary verb-grounding information on BridgeV2.** Each entry reports the estimated contribution of a verb to $I(Y; X)$ under the motion-only, goal-only, and combined conditions. $\Delta_{\text{motion}} = \text{Both} - \text{Goal}$ denotes information contributed uniquely by motion. A verb can carry unique information in both dimensions. The complete table appears in Appendix H.

$Y = \text{Verb}$	Motion	Goal	Both	Δ_{motion}	Δ_{goal}
move	0.157	0.189	0.213	+0.023	+0.056
put	0.134	0.143	0.161	+0.018	+0.027
fold	0.143	0.189	0.187	−0.001	+0.044
push	0.011	0.008	0.014	+0.005	+0.002

between an episode’s first and last frames, and motion dynamics as its 7-DoF action trajectory. This distinction maps onto different VLA modalities, as the goal information can be conveyed through vision, whereas motion-specific information must enter through the action representation. Figure 1 illustrates how verbs such as *push*, *flip*, and *fold* are associated with characteristic patterns of translation, rotation, contact, and gripper timing.

We study this distinction on BridgeV2 (Walke et al., 2023), which pairs teleoperated real-robot manipulation trajectories with free-form natural-language instructions. After extracting and lemmatizing instruction verbs, we retain 17 verb classes spanning 27,271 episodes. Dataset processing and the motion-profile features shown in Figure 1 are detailed in Appendices A and B.

Estimating verb information. Let Y denote the verb and X denote motion dynamics, action goal, or both. We estimate

$$I(Y; X) = H(Y) - H(Y | X) \quad (1)$$

from the held-out cross-entropy of matched Transformer classifiers under 5-fold stratified cross-validation. The motion condition receives the 7-DoF action sequence, the goal condition receives frozen DINOv2-S (Oquab et al., 2024) representations of the first and last frames, and the combined condition receives both. The architecture and optimization details are provided in Appendix F, with a corroborating R^2 commonality analysis in Appendix G.

Motion dynamics provide unique verb-grounding information. Table 1 shows that action goals and motion dynamics provide complementary information about verbs. In aggregate, the combined representation contains more verb

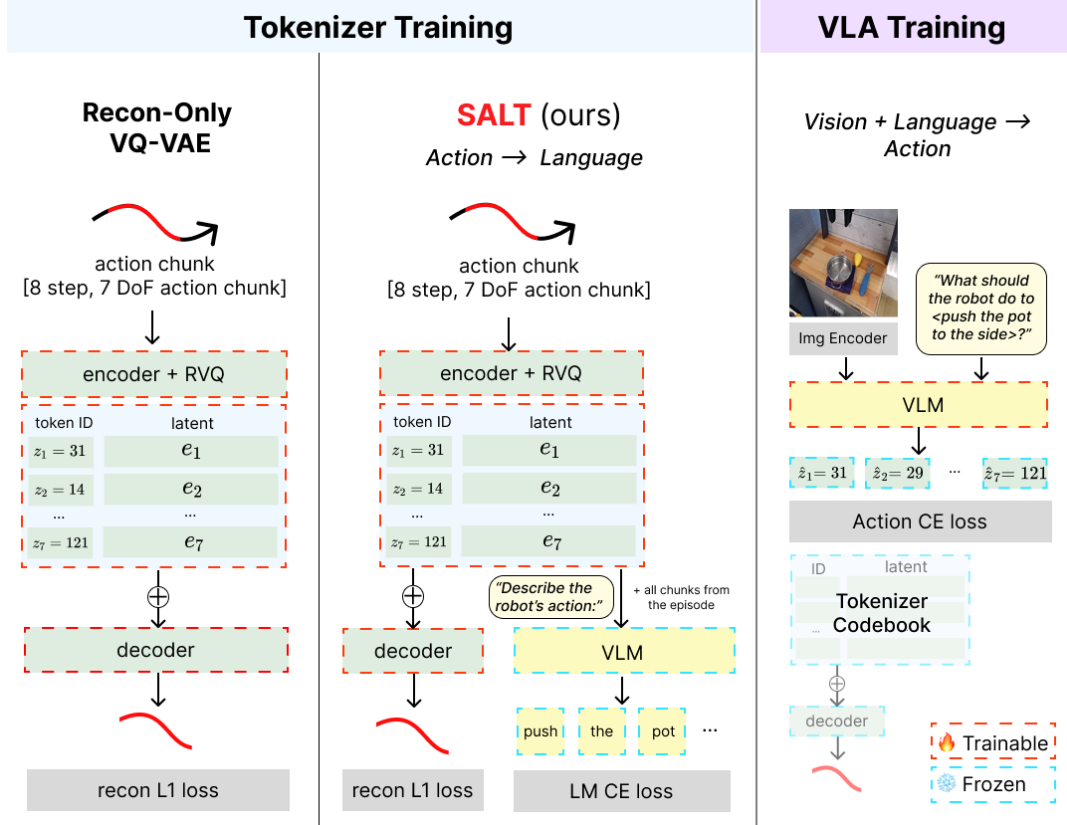


Figure 2: **SALT: semantically aligned action tokenization for VLAs.** **Left:** a reconstruction-only VQ-VAE tokenizer encodes an 8-step, 7-DoF action chunk with a residual-VQ encoder and is trained on reconstruction alone. **Middle (SALT):** the same architecture adds a generative alignment pressure: the quantized latents of all chunks in an episode are fed, with a describe prompt, to a frozen VLM that must generate the episode’s instruction, and the LM cross-entropy shapes the encoder and codebook through the straight-through quantizer. **Right:** during VLA training, the VLM backbone predicts action-token IDs from vision and language (action cross-entropy), and the frozen tokenizer codebook and decoder turn predicted IDs into executable actions.

information than either modality alone: motion contributes an estimated 0.059 bits beyond the visual goal representation, while action goals contribute 0.282 bits beyond motion. Visual state change explains more unique verb information overall, but does not exhaust the information available in the trajectory.

The motion-specific contribution is concentrated among verbs whose endpoint states can appear similar despite systematic differences in how the action is executed. For example, move and put together account for approximately two-thirds of the estimated motion-unique signal, while the less frequent verb push also contributes positively. Conversely, verbs whose meanings are strongly reflected in the resulting physical state are more strongly associated with the action-goal representation. For fold, for instance, the visual transition from an unfolded object to a folded configuration

is particularly salient, while the estimated motion-unique contribution is negligible.

These results establish that verbs are grounded in both what an action achieves and how it is performed. Because the motion-specific component can reach a VLA only through its action representation, language-conditioned control requires action–language alignment in addition to vision–language alignment. A complete breakdown across all 17 verbs is provided in Appendix H.

3 Diagnostic 2: Semantic Loss in Discrete Tokenization

Many VLAs represent continuous robot trajectories as discrete action tokens that can be predicted through the autoregressive interface of a pretrained VLM (Brohan et al., 2023b,a; Pertsch et al., 2025; Wang et al., 2025). An action tokenizer τ maps each continuous action chunk $a_{1:H} \in \mathbb{R}^{H \times d_a}$ to a sequence of K discrete symbols ($z_{1:K} \in \mathbb{V}^K$,

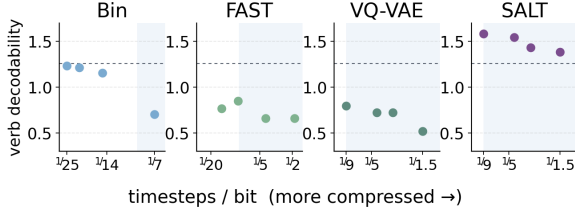


Figure 3: **A rate-distortion view of verb decodability erosion, across four tokenizer families on BridgeV2.** Verb decodability is the mutual information $I(\text{verb}; \text{tokens})$ in bits; the x-axis is timesteps per bit, so rightward is more compressed (a tick $1/N$ denotes N bits per timestep; Appendix C). Dashed line: continuous-trajectory reference (1.26 bits). Every reconstruction-only tokenizer is below the reference and declines as compression grows. SALT closes the tokenization gap across the full compression range.

where $\mathbb{V}$ is the token vocabulary): $\tau : \mathbb{R}^{H \times d_a} \rightarrow \mathbb{V}^K$, where $\tau(a_{1:H}) = z_{1:K}$. We consider three representative families: **Bin**, which independently discretizes each action dimension; **FAST** (Pertsch et al., 2025), which transforms trajectories into frequency coefficients and compresses them with byte-pair encoding; and **VQ-VAE** (van den Oord et al., 2017), which learns an encoder, codebook, and decoder through trajectory reconstruction. Although these approaches partition action space differently, none explicitly optimizes the resulting representation to preserve distinctions expressed by language.

We apply the verb-decoding analysis from Section 2 to token sequences produced by each tokenizer across compression levels. Figure 3 reports mutual information between verbs and tokenized actions as a function of bitrate, with the continuous trajectory as a reference. Across all three families, tokenized representations retain less verb information than the continuous actions, and the gap generally widens as compression increases.

These results show that the reconstruction-only objective does not guarantee preservation of language grounding signals in compressed action representations. A small error in Euclidean space could be consequential for verbs, and larger variations could be incidental and semantically equivalent, bottlenecking downstream VLA performance. Additional metrics and methodology details are provided in Appendices C and D.

4 Method: **S**emantically **A**Ligned **T**okenization

SALT augments a VQ-VAE action tokenizer (Wang et al., 2025) with language supervision, encouraging its latent space to preserve distinctions predictive of natural-language instructions. **SALT** intervenes exclusively in the tokenizer-training stage: it changes how the action vocabulary is learned, while VLA training and the policy’s use of the tokens remain unchanged (Figure 2). We focus on VQ-VAE tokenizers because, unlike fixed **Bin** or **FAST** representations, their encoder and codebooks can be directly shaped during training.

We instantiate **SALT** using a residual VQ-VAE. Given an action chunk $a_{1:H}^{(i)}$, the tokenizer predicts K codebook indices $z_{i,1:K}$. The corresponding codebook vectors are summed to form the quantized latent

$$\mathbf{q}_i = \sum_{k=1}^K \mathbf{e}_{z_{i,k}}^{(k)}. \quad (2)$$

SALT augments the standard VQ-VAE objective with a language-alignment loss,

$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{VQ}} + \lambda \mathcal{L}_{\text{align}}, \quad (3)$$

where $\mathcal{L}_{\text{recon}}$ is the action reconstruction loss, $\mathcal{L}_{\text{VQ}}$ comprises the codebook and commitment losses (van den Oord et al., 2017), and $\mathcal{L}_{\text{align}}$ encourages the tokenizer to preserve information predictive of language.

For alignment, an episode is divided into M action chunks, yielding quantized latents $\mathbf{q}_{1:M} = (\mathbf{q}_1, \dots, \mathbf{q}_M)$. We match the tokenizer latent dimension to the language-model embedding dimension and convert each latent into a soft prefix embedding,

$$\mathbf{p}_i = g \mathbf{q}_i + \text{PE}(i), \quad (4)$$

where g is a learned scalar gain and $\text{PE}(i)$ is a positional encoding. The resulting prefix sequence $\mathbf{P} = [\mathbf{p}_1, \dots, \mathbf{p}_M]$ is provided, together with a short textual prompt s , to a frozen pretrained language model (p_{LM}), which predicts the episode instruction $w_{1:L}$ where L is the instruction length. The alignment objective is

$$\mathcal{L}_{\text{align}} = -\frac{1}{L} \sum_{t=1}^L \log p_{\text{LM}}(w_t \mid w_{<t}, \mathbf{P}, s). \quad (5)$$

The language model remains frozen, while gradients propagate through the input embeddings and the straight-through quantizer into the tokenizer encoder and codebooks. Consequently, the alignment objective changes how the tokenizer partitions action space without modifying the downstream VLA architecture. Because it operates directly on free-form instructions, it requires neither a predefined verb inventory nor a separate text encoder or contrastive negative pairs.

After tokenizer training, the language model is discarded and the tokenizer is frozen. Downstream VLA training proceeds normally, that is the policy predicts discrete codebook indices, which are decoded by the tokenizer into continuous actions.

5 Experiments

5.1 Setup

We evaluate SALT as a drop-in action tokenizer for miniVLA (Belkhale and Sadigh, 2024), a Prismatic-style VLA with a Qwen2.5-0.5B (Yang et al., 2025) backbone. All policies are trained on BridgeV2 from the base vision-language checkpoint (no prior robot-action pretraining) for 15k gradient steps at a global batch size of 128. The action tokenizer is the only component that differs across conditions.

We compare three tokenizers. **SALT** and **VQ-VAE** share the same residual-VQ architecture (8-timestep chunks; 7 residual groups of 256 codes each, so every chunk becomes 7 token IDs) and the same training data, and differ only in the language-alignment loss of Section 4 (VQ-VAE is trained with reconstruction and commitment losses alone). **FAST** (Pertsch et al., 2025) is fitted on the same BridgeV2 chunks with vocabulary 1,024 and ≈ 7 tokens per chunk. These hyperparameters are selected so that all three tokenizers operate at a comparable compute and compression budget: ≈ 7 tokens per 8-timestep chunk, i.e. 7.0–8.6 bits per timestep (Appendix C).

Rollout success is measured in SimplerEnv’s visual-matching WidowX suite (Li et al., 2024): four tabletop tasks (put spoon on towel, put carrot on plate, stack green block on yellow, put eggplant in basket), 24 episodes each (96 rollouts per policy), with open-loop execution of 8-step action chunks.

5.2 Results

SALT improves deployment success. Table 2 reports rollout success. SALT reaches 71.9%, against 42.7% for VQ-VAE and 31.2% for FAST, and leads

Table 2: SimplerEnv WidowX rollout success rate (%) for BridgeV2-trained miniVLA, per task (24 episodes each) and averaged ($n=96$). Tasks: put spoon on towel, put carrot on plate, stack green block on yellow, put eggplant in basket. All policies are trained identically for 15k steps; the action tokenizer is the only changed component. Best in **bold**.

Tokenizer	Spoon	Carrot	Stack	Eggplant	Mean $\uparrow$
FAST	54.2	29.2	20.8	20.8	31.2
VQ-VAE	58.3	45.8	33.3	33.3	42.7
SALT	75.0	62.5	70.8	79.2	71.9

Table 3: Verb decodability and reconstruction fidelity of the policy tokenizers (probe and protocol of Section 3, 5-fold stratified CV; accuracy companions in Appendix D). *TokID* probes the discrete token IDs; E_{in} probes the trained policy’s frozen action-token input embeddings (fold 0). *Recon* is held-out reconstruction L1 in normalized action units. Best in **bold**.

Tokenizer	Recon $\downarrow$	TokID MF1 $\uparrow$	E_{in} MF1 $\uparrow$
FAST ($V=1024$)	0.113	30.3	36.3
VQ-VAE	0.080	37.3	38.3
SALT	0.088	39.1	43.7
Native (continuous, ref.)	—	53.0	—

on every individual task, with the largest margins on the two most difficult (stack: 70.8 vs. 33.3; eggplant: 79.2 vs. 33.3). The SALT-vs-VQ-VAE comparison isolates the alignment loss: the two tokenizers share architecture, capacity, data, and the entire VLA training recipe, so the 29.2-point gap is attributable to how the action vocabulary is partitioned, not to how much it compresses.

Language alignment transfers beyond the tokenizer. Although the alignment objective supervises only the tokenizer’s latent representations, its effects propagate to both the discrete token IDs produced by the tokenizer and the action-token embeddings learned by the downstream VLA (Table 3). This transfer is not guaranteed: the alignment loss is applied to the quantized latents inside the tokenizer, and it never directly supervises (i) the discrete symbols or (ii) the action-token embeddings re-initialized during VLA training. Nevertheless, SALT achieves the highest verb decodability at both stages, reaching 39.1 macro-F1 on token IDs (vs. 37.3 for VQ-VAE and 30.3 for FAST) and 43.7 on learned action embeddings (vs. 38.3 and 36.3). Moreover, its token-ID accuracy (58.7%) matches the continuous-action reference (58.0%; Appendix D). These results suggest that semantic structure introduced during tokenizer training sur-

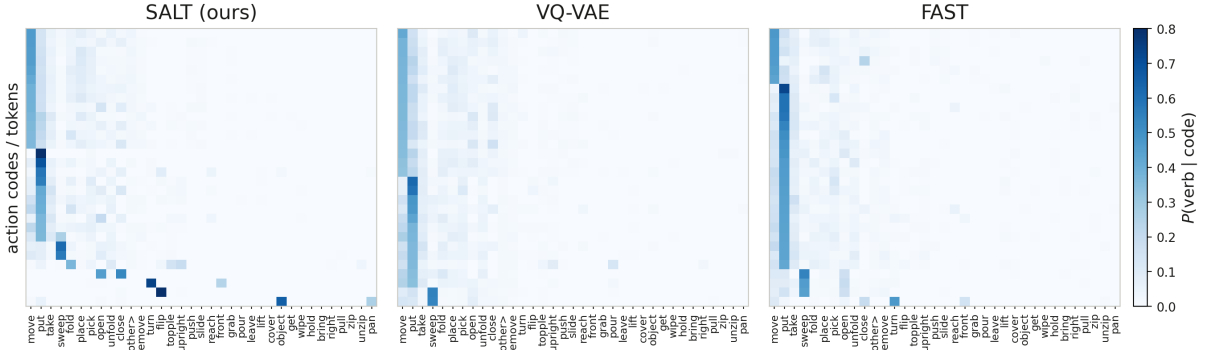


Figure 4: **Per-code verb distributions, first residual group.** For each tokenizer we show its 30 most verb-selective units — highest $\max_v P(\text{verb} | \text{code})$ among codes/tokens used in ≥ 100 training chunks — as rows sorted by dominant verb; each cell is $P(\text{verb} | \text{code})$ by counting. Sharp verb-selective rows emerge for SALT across many verbs, including rare ones (*flip* 98%, *turn* 74%, *pour*, *topple*); the selective units of VQ-VAE (middle) and FAST (right) are confined to the most frequent verbs (*put*, *sweep*), with the remainder diffuse.

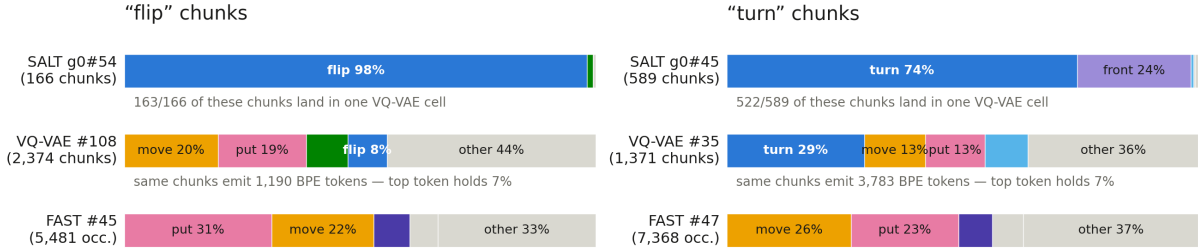


Figure 5: **The same action chunks under three tokenizations.** For SALT’s *flip*- and *turn*-dedicated codes, the verb composition of (top bar) the SALT code, (middle) the single VQ-VAE cell that receives almost all of the same chunks, and (bottom) the FAST BPE token receiving the largest share of their token occurrences. SALT dedicates a symbol; VQ-VAE lumps the chunks into a generic mixed cell; FAST distributes them across generic tokens with no dominant home. The *front* share of SALT’s *turn* code comes entirely from the instruction “*lever vertical to front*” — the same lever-turning action worded without “*turn*” — so the code unites this paraphrase with *turn*: the alignment tracks meaning, not surface wording.

vives discretization and is preserved throughout downstream policy learning.

Semantic alignment preserves reconstruction fidelity. The improved semantic structure does not come at the expense of reconstruction quality. At the matched compression rate of seven tokens per eight-step chunk, SALT’s held-out reconstruction error remains close to that of VQ-VAE (Table 3). Likewise, the characteristic motion signatures identified in Section 2 are largely preserved after reconstruction: the rank correlation of one-vs-rest effect sizes across 63 interpretable trajectory features remains at least 0.92 (vs. 0.96 for VQ-VAE), and distinctive patterns such as *flip*’s dominant rotational motion are retained. Thus, semantic alignment primarily changes how action trajectories are partitioned into discrete symbols rather than how faithfully those symbols reconstruct continuous actions.

Verb-specialized codes emerge under semantic alignment. Probe accuracy shows that SALT preserves more verb information, but does not reveal how its vocabulary is organized. We therefore examine code–verb co-occurrence directly, without training an additional classifier. If codes specialize by action semantics, each code’s distribution should concentrate on one or a small number of verbs, producing sharp verb-selective bands; semantically mixed codes instead produce diffuse distributions. Figure 4 shows that SALT develops highly selective codes associated with particular verbs, whereas VQ-VAE distributes the same trajectories across semantically mixed codes and FAST largely reflects the corpus verb distribution. Figure 5 illustrates this effect for representative *flip* and *turn* codes. Interestingly, the *turn* code also captures instructions phrased as “*lever vertical to front*,” grouping semantically equivalent actions

despite different wording. Beyond these examples, the same pattern holds quantitatively. A probe-free majority-vote lookup based solely on code–verb co-occurrence achieves higher held-out accuracy for SALT than VQ-VAE for every individual residual group and every cumulative group prefix (Appendix E); for example, using only the first two residual groups yields 46.3% episode-level accuracy for SALT versus 43.6% for VQ-VAE (McNemar $p = .011$), with FAST reaching 35.0%. Together, these results indicate that semantic alignment reorganizes the action vocabulary around linguistically meaningful action categories rather than reconstruction alone.

6 Related Work

Embodied language grounding and action semantics. Embodied language grounding studies how words connect to perception, affordances, skills, and action. Most robotics work grounds language in objects, spatial relations, task goals, or feasible plans (Shridhar et al., 2021; Mees et al., 2022; Ahn et al., 2022). We focus on verbs because they provide a compact probe of action-level structure: they group trajectories by motion pattern, contact mode, gripper timing, force, path/result structure, and object interaction. Unlike work aimed at learning better language representations, we use language as supervision and diagnosis for learning better action representations. Our representational diagnostics adapt the probing methodology developed for NLP (Hewitt and Liang, 2019; Belinkov, 2022) to the discrete-token interface of a VLA.

Cross-modal alignment in robot learning. Most language-conditioned robot learning aligns language with perception. Methods such as CLIPort (Shridhar et al., 2021), BC-Z (Jang et al., 2022), CALVIN (Mees et al., 2022), and Say-Can (Ahn et al., 2022) use language primarily to specify objects, spatial relations, or task goals. Our work instead studies alignment at the action interface, asking whether action representations themselves preserve the distinctions expressed by language.

Vision-language-action models. Vision-language-action (VLA) models adapt pretrained vision-language or language-model backbones for robot control by introducing an action prediction interface (Brohan et al., 2023b,a; Driess et al., 2023; Kim et al., 2024). While this paradigm

benefits from pretrained visual and linguistic representations, it raises a representational question that has received comparatively little attention: how continuous robot actions should be represented before they are consumed by the backbone. Our work focuses specifically on this action interface rather than scaling VLA architectures themselves.

Action representations in VLAs. Existing VLAs represent actions either as discrete tokens or continuous action chunks. Discrete approaches, including RT-1/RT-2-style binning (Brohan et al., 2023b,a), VQ-based tokenizers (Lee et al., 2024; Wang et al., 2025), FAST (Pertsch et al., 2025), BeT (Shafiullah et al., 2022), QueST (Mete et al., 2024), LAPA (Ye et al., 2025), and OAT (Liu et al., 2026), are typically optimized for reconstruction or self-supervised prediction. Continuous approaches such as Diffusion Policy (Chi et al., 2024), Octo (Octo Model Team et al., 2024), and π_0 (Black et al., 2024) avoid discretization by predicting continuous action chunks. Our work addresses the discrete setting, showing that action tokenization should preserve language-relevant structure in addition to executable trajectories.

Language-aligned representations for robot control. A separate line of work uses natural language as supervision for *visual* representations in robotics. R3M (Nair et al., 2022) aligns video frames with time-shifted language captions, Voltron (Karamcheti et al., 2023) jointly reconstructs and grounds language during manipulation pretraining, and LIV (Ma et al., 2023) unifies value learning with vision-language alignment via a CLIP-style (Radford et al., 2021) contrastive objective. These methods inject language structure into the perception side of a robot policy. SALT applies the analogous idea on the *action* side: instead of aligning a visual encoder with instructions, we align the latent space of an action tokenizer with instructions, so that the discrete action vocabulary itself carries language-relevant geometry. It also differs in mechanism: where these methods use contrastive objectives against a text encoder, SALT’s pressure is generative because a frozen LM must produce the instruction from the quantized action latents, which aligns the representation to the same soft-token interface a VLA backbone consumes.

Limitations

Our conclusions are subject to several limitations. First, current language-conditioned robotics datasets exhibit limited linguistic diversity: after filtering, BridgeV2 contains only 17 verb classes. Although sufficient to study action-language alignment, richer datasets with broader verb inventories would better test whether the benefits of semantic tokenization grow with linguistic diversity.

Second, SALT currently applies only to tokenizers with a learnable latent representation. Extending similar semantic supervision to fixed discretization schemes such as Bin or signal-processing approaches such as FAST remains an open question.

Third, our experiments use a relatively small VLA (0.5B parameters), a single training dataset, and simulation-based evaluation in SimplerEnv. Future work should determine whether the observed gains persist under large-scale multi-embodiment pretraining and real-robot deployment.

Finally, while we show that semantic alignment produces more interpretable action vocabularies and improves downstream policy performance, we do not establish a causal mechanism relating these two effects. Understanding how semantically organized action codes influence policy learning remains an important direction for future work.

References

- Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, and 25 others. 2022. *Do as i can, not as i say: Grounding language in robotic affordances*. *Preprint*, arXiv:2204.01691.
- Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, and 8 others. 2022. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 23716–23736.
- Yonatan Belinkov. 2022. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219.
- Suneel Belkale and Dorsa Sadigh. 2024. miniVLA: A better OpenVLA with a smaller footprint. <https://github.com/Stanford-ILIAD/openvla-mini>.
- Kevin Black, Noah Brown, Danny Driess, and 1 others. 2024. π_0 : A vision-language-action flow model for general robot control.
- Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, and 31 others. 2023a. *RT-2: Vision-language-action models transfer web knowledge to robotic control*. *Preprint*, arXiv:2307.15818.
- Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, and 31 others. 2023b. *RT-1: Robotics transformer for real-world control at scale*. *Preprint*, arXiv:2212.06817.
- Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. 2024. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*.
- Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, and 3 others. 2023. PaLM-E: An embodied multimodal language model. In *International Conference on Machine Learning*.
- John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Conference on Empirical Methods in Natural Language Processing*.
- Minyoung Hwang, Joey Hejna, Dorsa Sadigh, and Yonatan Bisk. 2025. MotIF: Motion instruction fine-tuning. *IEEE Robotics and Automation Letters*.
- Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. 2022. BC-Z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*.
- Siddharth Karamcheti, Suraj Nair, Annie S. Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. 2023. Language-driven representation learning for robotics. In *Robotics: Science and Systems*.
- Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3128–3137.

- Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. 2024. [OpenVLA: An open-source vision-language-action model](#). *Preprint*, arXiv:2406.09246.
- Seungjae Lee, Yecheng Jason Wang, Haritheja Etukuru, H. Jin Kim, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. 2024. Behavior generation with latent actions.
- Beth Levin and Malka Rappaport Hovav. 1991. Wiping the slate clean: A lexical semantic exploration. *Cognition*, 41(1–3):123–151.
- Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. 2024. Evaluating real-world robot manipulation policies in simulation. In *Conference on Robot Learning (CoRL)*.
- Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. 2023. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Chaoqi Liu, Xiaoshen Han, Jiawei Gao, Yue Zhao, Haonan Chen, and Yilun Du. 2026. Oat: Ordered action tokenization. *arXiv preprint arXiv:2602.04215*.
- Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. 2023. LIV: Language-image representations and rewards for robotic control. In *International Conference on Machine Learning*.
- Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. 2022. CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*.
- Atharva Mete, Haotian Xue, Albert Wilcox, Yongxin Chen, and Animesh Garg. 2024. QueST: Self-supervised skill abstractions for continuous control. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. 2022. R3M: A universal visual representation for robot manipulation. In *Conference on Robot Learning*.
- Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Charles Xu, Jianlan Luo, Tairan Kreiman, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. 2024. [Octo: An open-source generalist robot policy](#). *Preprint*, arXiv:2405.12213.
- Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, and 7 others. 2024. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*.
- Karl Pertsch, Kyle Stachowicz, Brian Ichter, and 1 others. 2025. FAST: Efficient action tokenization for vision-language-action models.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*.
- Malka Rappaport Hovav and Beth Levin. 2010. Reflections on manner/result complementarity. In Malka Rappaport Hovav, Edit Doron, and Ivy Sichel, editors, *Syntax, Lexical Semantics, and Event Structure*, pages 21–38. Oxford University Press.
- Nur Muhammad Mahi Shafiullah, Zichen Jeff Cui, Ariuntuya Altanzaya, and Lerrel Pinto. 2022. Behavior transformers: Cloning k modes with one stone. In *Advances in Neural Information Processing Systems*.
- Mohit Shridhar, Lucas Manuelli, and Dieter Fox. 2021. CLIPort: What and where pathways for robotic manipulation. In *Conference on Robot Learning*.
- Leonard Talmy. 1985. Lexicalization patterns: Semantic structure in lexical forms. In Timothy Shopen, editor, *Language Typology and Syntactic Description, Vol. III: Grammatical Categories and the Lexicon*, pages 57–149. Cambridge University Press.
- Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*.
- Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3156–3164.
- Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. 2023. [BridgeData V2: A dataset for robot learning at scale](#). *Preprint*, arXiv:2308.12952.
- Yating Wang, Haoyi Zhu, Mingyu Liu, Jiange Yang, Hao-Shu Fang, and Tong He. 2025. Vq-vla: Improving vision-language-action models via scaling vector-quantized action tokenizers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

An Yang, Baosong Yang, Beichen Zhang, and 1 others. 2025. [Qwen2.5 technical report](#). *Preprint*, arXiv:2412.15115.

Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Sejeun Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, and 1 others. 2025. Latent action pretraining from videos. In *International Conference on Learning Representations*.

Jianing Zhang, Chenhao Zheng, Yajun Yang, Max Argus, Rustin Soraki, Winson Han, Taira Anderson, Chun-Liang Li, Shuo Liu, Jiafei Duan, Zhongzheng Ren, Jieyu Zhang, and Ranjay Krishna. 2026. Molmomoion: Forecasting point trajectories in 3d with language instruction. *arXiv preprint arXiv:2606.18558*.

A Dataset: BridgeV2

We conduct all experiments on BridgeV2 (Walke et al., 2023), a large-scale dataset of real-robot tabletop manipulation collected across multiple laboratory environments.

Setting. Episodes are collected via human teleoperation of a WidowX 250 6-DoF robot arm equipped with a parallel-jaw gripper. The workspace is a tabletop with diverse household objects (toys, kitchenware, cloths, blocks, etc.). Each episode captures a single short-horizon manipulation task—typically lasting ~ 37 timesteps—from a third-person camera mounted above the table.

Observations and actions. At each timestep, the dataset records an RGB image (image_0, 256×256) and a 7-dimensional action vector:

$$a_t = [\Delta x, \Delta y, \Delta z, \Delta \phi, \Delta \theta, \Delta \psi, g] \in \mathbb{R}^7, \quad (6)$$

where $\Delta x: \Delta \psi$ are end-effector velocity commands (3 translational + 3 rotational) and $g \in \{0, 1\}$ is the gripper open/close command. Each episode is thus a variable-length sequence of (image, action) pairs.

Instructions. Every episode is paired with a natural-language task instruction (e.g., “put the corn into the pot,” “fold the cloth in half”). Instructions were provided by the teleoperators at collection time, describing the intended task for each demonstration. BridgeV2 contains 27,575 episodes (after basic quality filtering) spanning 519 unique manipulated objects.

Verb labels. We extract a primary verb from each instruction using SpaCy dependency parsing. Verb forms are lemmatized (“moved” $\rightarrow$ “move”), directional variants are merged (“put down/on/up/in”

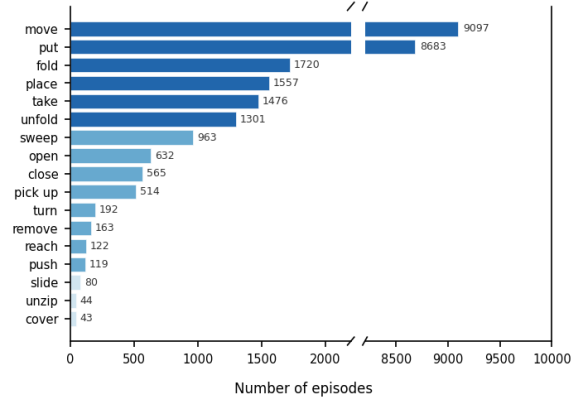


Figure 6: Distribution of the 17 verb classes in BridgeV2 (episodes with ≥ 30 training examples). The two dominant verbs (*move*, *put*) comprise 65% of episodes; the long tail spans 15 additional verbs.

$\rightarrow$ “put”), and non-English or non-verb tokens are dropped. Applying a minimum-count filter (≥ 30 training examples) yields 17 verb classes covering 27,271 episodes. For VLA training we split 90/10 by episode into 24,544 training and 2,727 validation samples (seed = 42); for the probes of Sections 2 and 3 we use 5-fold stratified CV with the same seed.

Figure 6 shows the resulting distribution. The class frequencies are heavily imbalanced: the top two verbs (*move*, *put*) account for 65% of all episodes, while the five rarest classes each have fewer than 200 examples. All probes are nonetheless trained with vanilla cross-entropy on the natural class distribution (Appendix F), which is required for the calibrated posteriors the MI estimates rely on; class imbalance is reported through macro-averaged metrics instead.

B Verb Motion-Profile Features

This appendix details how the per-verb “characteristic motion profiles” of Figure 1 are derived. The procedure is fully automatic given the verb labels; no profile is hand-written.

Feature bank. Each episode’s 7-DoF action trajectory (no images) is summarized by ~ 63 scalar features chosen to be interpretable and scene-invariant — they describe *how* the arm moved, not *where* in the workspace. The features fall into eight families: **(i) path geometry**: path length, net displacement, path efficiency (net/total), mean curvature, direction changes, per-axis totals $|\Delta x|, |\Delta y|, |\Delta z|$, dominant-axis fraction, lateral-to-vertical ratio; **(ii) rotation**: per-

axis $|\Delta\text{roll/pitch/yaw}|$ and signed totals, rotation-to-translation ratio, rotation dominant-axis fraction; **(iii) velocity-profile shape**: mean and peak speed, time of peak speed, early-vs-late energy ratio, monotonicity, number of speed peaks, kurtosis; **(iv) acceleration and smoothness**: peak acceleration, acceleration variance, mean jerk, mean cosine between consecutive displacement vectors; **(v) gripper state and timing**: open fraction, close→open transitions, time to first close/open, closed- and open-phase durations, gripper state at peak speed; **(vi) contact-conditioned motion**: path length and mean speed while the gripper is closed vs. open, closed-to-open speed ratio, vertical motion while closed; **(vii) spatial extent**: per-axis bounding-box ranges, bounding-box volume, radius of gyration; **(viii) symmetry and oscillation**: start–end distance, out-and-back overlap, dominant speed frequency, per-axis zero crossings.

Scoring and selection. For each verb we sample up to 100 episodes (seed 42, minimum 5 timesteps) from the same 17-class corpus as Section 2, compute all features, and score every (verb, feature) pair with a one-vs-rest Cohen’s d :

$$d_{v,f} = \frac{\bar{x}_{v,f} - \bar{x}_{\neg v,f}}{s_{\text{pooled}}}, \quad (7)$$

where $\bar{x}_{v,f}$ is the focal verb’s mean on feature f , $\bar{x}_{\neg v,f}$ the pooled mean of all other verbs, and s_{pooled} the pooled standard deviation. A verb’s characteristic profile is its top-3 features by $|d|$, subject to $|d| \geq 0.6$ (a medium-or-larger effect); each selected feature is rendered in Figure 1 with a plain-English label and the sign of its deviation. The six verbs shown in the figure were chosen to span distinct motion archetypes (approach, lift, push, planar rotation, sustained wiping while gripping, fixture closing); the same procedure applies to all 17 classes.

C Bits per Timestep

We use *bits per timestep* (bpts) as the underlying rate measure for the tokenizer sweeps. For a tokenizer that emits K discrete tokens drawn from a vocabulary of size V per chunk of T continuous timesteps, the per-timestep rate is

$$\text{bpts} = \frac{K \log_2 V}{T}. \quad (8)$$

Figure 3 plots its inverse, *timesteps per bit* (1/bpts), so that the axis grows with compression: an episode

segment of N timesteps that tokenizes into N bits sits at 1, and more aggressive tokenizers sit further right. Axis ticks are labeled $1/N$, i.e. N bits per timestep. This is the average number of bits the tokenized stream carries per original action timestep. For variable-length tokenizers (e.g. FAST) K is replaced by its empirical mean over the training set. Bpts is meaningful for comparing configurations *within* a tokenizer family (Bin sweep with varying bin counts, FAST sweep with varying vocabulary or scale parameter, VQ-VAE sweep with varying codebook size); cross-family absolute positions reflect architectural rate ranges (e.g. Bin emits 7 tokens per timestep regardless of bin count; VQ-VAE emits 1 token per chunk) and should not be over-interpreted as a global compression ordering.

D Macro-F1 Version of Tokenizer Probes

Table 4 tabulates the Section 3 token-ID verb probes at representative configurations, with overall accuracy (Acc) and per-class accuracy (PCA) reported alongside macro-F1.

Table 4: Token-ID verb probes on BridgeV2 (%; 5-fold stratified CV; standard errors ≤ 1.5 MF1). Accompanies Section 3 and Figure 3.

Tokenizer	Acc	PCA	MF1
Bin (256/dim)	50.7	42.0	43.1
FAST (vocab = 256)	51.1	33.6	34.8
VQ-VAE (ng9/nemb256)	52.7	30.3	34.8
Native (continuous, ref.)	58.0	52.3	53.0

Table 5 is the same companion for the token-ID probes of the three Section 5 tokenizers (identical protocol; the VQ tokenizers emit 7 residual-group IDs per 8-step chunk, and FAST is the policy’s fitted vocabulary-1,024 tokenizer applied per chunk).

Table 5: Accuracy companion for the Section 5 token-ID probes (%; 5-fold stratified CV; per-fold sd ≤ 1.5). Accompanies Table 3.

Tokenizer	Acc	PCA	MF1
FAST (chunked, vocab = 1024)	50.7	28.8	30.3
VQ-VAE	54.5	36.7	37.3
SALT	58.7	37.9	39.1
FAST E_{in} (fold 0)	55.7	35.4	36.3
VQ-VAE E_{in} (fold 0)	54.1	39.0	38.3
SALT E_{in} (fold 0)	62.4	43.3	43.7
Native (continuous, ref.)	58.0	52.3	53.0

E Held-Out Code–Verb Lookup Accuracy

Figure 7 details the probe-free lookup statistic referenced in Section 5.2. On training chunks we record, for each code (or combination of codes), the majority verb of the episodes it fires in; on held-out episodes we predict by soft voting over each episode’s chunks and score accuracy. The evaluation is restricted to code combinations seen in training. The top row reports top-1 accuracy; the bottom row macro-averages over verb classes with at least 10 validation episodes. SALT exceeds VQ-VAE for every individual residual group and every cumulative group prefix.

F Probe Architecture and Training Details

This appendix details the verb classifier used in Section 2 to estimate $H(Y \mid X^{(m)})$ for $m \in \{a, v, (a, v)\}$.

Architecture. A single Transformer encoder consumes a sequence

$$[\text{CLS}, v_1, \dots, v_V, a_1, \dots, a_T],$$

where v_i are vision patch tokens (frozen DINOv2-S patch features projected to d_{model} , two endpoint frames $\Rightarrow V = 2 \cdot 49 = 98$ tokens) and a_t are action tokens (linear projection of the 7-D action vector at timestep t , padded to a fixed batch length). Sinusoidal temporal positional encoding on action tokens, learned 2-D positional encoding on patch tokens, and learned modality-type embeddings added to vision and action token streams. The encoder is pre-norm with 4 layers, $d_{\text{model}} = 128$, 8 attention heads, dim-feedforward $4d_{\text{model}}$, and dropout 0.1. The classifier head reads the CLS token and applies LayerNorm $\rightarrow$ ReLU $\rightarrow$ Dropout $\rightarrow$ Linear($d_{\text{model}}, |\mathcal{Y}|$), with $|\mathcal{Y}| = 17$.

Three input variants. The same architecture supports three masking conditions selected at forward time: motion-only masks the V vision tokens out of attention, goal-only masks the T action tokens, and both leaves the full sequence unmasked. The CLS token always attends to whatever is unmasked. Since masked tokens contribute no gradients, this is equivalent to training three separate probes whose only architectural difference is which tokens they receive — the parameter count is identical across the three runs.

Training. Vanilla cross-entropy on the natural verb distribution (no class weighting, no label smoothing); this is the right loss for a calibrated $p(y \mid x)$ estimate, which is what the variational MI bound requires. AdamW with learning rate 10^{-4} , batch size 32, weight decay 10^{-2} , OneCycleLR scheduler, gradient clipping at 1.0. Up to 100 epochs with early stopping on val cross-entropy, patience 15. The model checkpoint with the lowest val cross-entropy is used for held-out logit extraction.

Cross-validation. Stratified 5-fold CV on the 27,271 verb-labeled BridgeV2 episodes (StratifiedKFold, seed 42). The same fold assignment is used across the three modality conditions, so the per-fold differences $H_k(Y \mid X_v) - H_k(Y \mid X_a, X_v)$ are paired samples. Across the 5 folds, every episode appears in val exactly once.

Estimating MI. For each (m, k) we extract logits on fold k ’s held-out episodes, apply log-softmax, and compute

$$\hat{H}_k(Y \mid X^{(m)}) = -\frac{1}{N_k} \sum_{i \in \text{val}_k} \log_2 p_\theta(y_i \mid x_i^{(m)}),$$

in bits. Then $\hat{I}_k(Y; X^{(m)}) = \hat{H}(Y) - \hat{H}_k(Y \mid X^{(m)})$ where $\hat{H}(Y)$ is the empirical entropy of the verb prior over the full verb-labeled corpus. Conditional MI is the per-fold difference of two cross-entropies; its 5-fold mean and standard error are reported in the main text. Wilcoxon signed-rank tests are computed on the 5 paired samples.

G R^2 Commonality Decomposition

As a corroborating analysis, we apply R^2 commonality (variance partitioning) on top of the same probe representations used in Section 2. We extract frozen [CLS] embeddings from the motion-only and goal-only probes, then regress one-hot verb labels on three feature sets — motion only, goal only, and their concatenation — using ridge regression. The decomposition separates variance uniquely explained by action trajectories, variance uniquely explained by endpoint state changes, and variance shared by both:

$$\begin{aligned} R_{\text{unique motion}}^2 &= R_{\text{cat}}^2 - R_{\text{vision}}^2, \\ R_{\text{unique goal}}^2 &= R_{\text{cat}}^2 - R_{\text{action}}^2, \\ R_{\text{shared}}^2 &= R_{\text{action}}^2 + R_{\text{vision}}^2 - R_{\text{cat}}^2. \end{aligned}$$

$P(\text{verb} \mid \text{code})$ by counting — episode level, SALT vs Recon-VQ codes (bridge, teleop only) — seen cells only
(right panels evaluated only where the code combination was seen in training — the population shrinks with k)

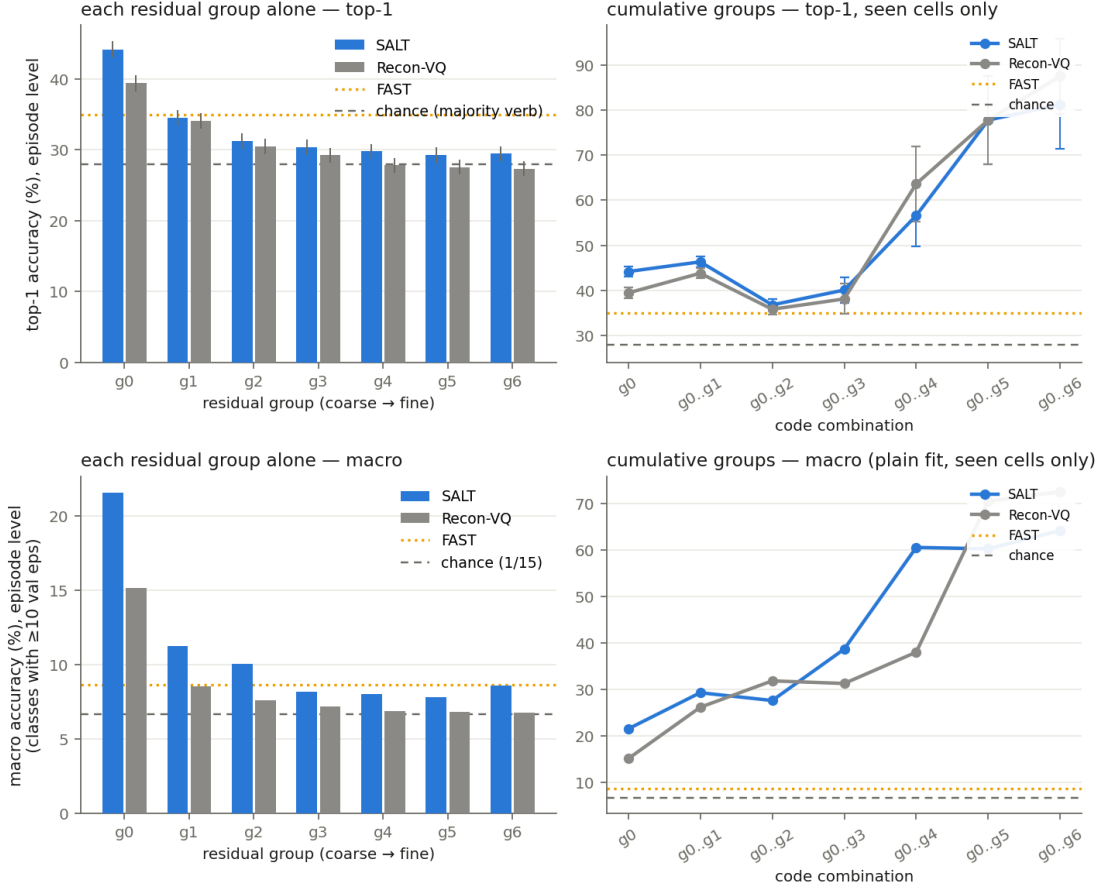


Figure 7: $P(\text{verb} \mid \text{code})$ by counting: majority-vote lookup accuracy on held-out episodes (seen code combinations). Top row: top-1 accuracy; bottom row: macro accuracy over classes with ≥ 10 validation episodes. Left: each residual group alone; right: cumulative group prefixes. SALT exceeds VQ-VAE at every group and prefix; FAST references dotted, chance dashed.

The concatenated representation explains $R_{\text{cat}}^2 = 0.551$ of verb variance. Of this, $R_{\text{unique motion}}^2 = 0.046$ is unique to action trajectories, $R_{\text{unique goal}}^2 = 0.122$ is unique to endpoint state, and the remaining 69.5% of explained variance is shared. The qualitative pattern matches the mutual-information decomposition in Section 2: a small but reliable unique-motion component sits on top of a substantial shared component, with goal carrying somewhat more unique information than motion (here, ratio $\approx 2.6\times$; in the MI estimate, $\approx 4.8\times$).

H Per-Verb MI Contribution

Table 6 reports the per-verb contribution to total mutual information for each modality condition, pooled over the five held-out folds. The contribu-

tion of class c to total $I(Y; X^{(m)})$ is

$$\text{contrib}_c(m) = \pi_c (-\log_2 \pi_c - \overline{\text{CE}}_c(m)),$$

where π_c is the class prior and $\overline{\text{CE}}_c(m)$ is the average cross-entropy on val episodes whose true verb is c . Per-modality $\Delta_{\text{motion}} = \text{contrib}_c(\text{Both}) - \text{contrib}_c(\text{Goal})$ and $\Delta_{\text{goal}} = \text{contrib}_c(\text{Both}) - \text{contrib}_c(\text{Motion})$ isolate where each modality’s unique contribution lives; their column totals match the global conditional MIs reported in Section 2 up to fold-pooling rounding.

I Artifact Licenses and Intended Use

All datasets and pretrained models used in this work are publicly released for academic research, and we use them in a manner consistent with their stated intended use.

Table 6: Per-verb MI contribution (bits, pooled across 5 CV folds). $\pi \log_2 \pi$ is the entropy contribution from the class prior.

Verb	count	$\pi \log_2 \pi$	Motion	Goal	Both	Δ_{motion}	Δ_{goal}
move	27,291	0.528	0.157	0.189	0.213	+0.023	+0.056
put	26,049	0.526	0.134	0.143	0.161	+0.018	+0.027
fold	5,160	0.252	0.143	0.189	0.187	-0.001	+0.044
place	4,671	0.236	0.056	0.072	0.079	+0.007	+0.023
take	4,428	0.228	0.121	0.162	0.162	+0.000	+0.041
unfold	3,903	0.209	0.100	0.145	0.156	+0.011	+0.056
sweep	2,889	0.170	0.162	0.166	0.165	-0.001	+0.003
open	1,896	0.126	0.097	0.105	0.099	-0.006	+0.002
close	1,695	0.116	0.079	0.099	0.097	-0.001	+0.018
pick up	1,542	0.108	0.087	0.089	0.089	+0.000	+0.002
turn	576	0.050	0.044	0.045	0.046	+0.001	+0.002
remove	489	0.044	0.009	0.012	0.012	+0.001	+0.003
reach	366	0.035	0.034	0.034	0.035	+0.000	+0.000
push	357	0.034	0.011	0.008	0.014	+0.005	+0.002
slide	240	0.025	0.008	0.007	0.010	+0.003	+0.002
unzip	132	0.015	0.014	0.014	0.014	-0.000	+0.000
cover	129	0.015	0.004	0.004	0.004	+0.000	+0.000
Total	81,813	2.717	1.260	1.483	1.542	+0.059	+0.282

Datasets. BridgeV2 (Walke et al., 2023) is released under the Creative Commons Attribution 4.0 (CC-BY-4.0) license. The dataset contains no personally identifying information or offensive content: it consists of third-person camera frames of a robot arm manipulating household objects, together with short manipulation-instruction strings (e.g. “put the pot on the stove”).

Pretrained models. The miniVLA base VLM checkpoint (Belkale and Sadigh, 2024) and the Stanford VQ-VAE bridge tokenizer are released under the MIT license by the Stanford ILIAD group via HuggingFace. Qwen2.5-0.5B (Yang et al., 2025) is released under the Apache 2.0 license. DINOv2 (Oquab et al., 2024) is released under the Apache 2.0 license. The FAST action tokenizer (Pertsch et al., 2025) is released under the Apache 2.0 license. SimplerEnv (Li et al., 2024) is released under the MIT license.

Software. SpaCy is released under the MIT license. PyTorch is released under a BSD-style license.

Derived artifacts. Our SALT tokenizer checkpoints and probe code are intended for academic research only and will be released under a research-only license.

J Compute Budget

Hardware. All training runs were performed on NVIDIA L40S GPUs (48 GB) on a shared academic cluster.

Policy training. Each headline policy (SALT, VQ-VAE, and FAST miniVLA) was trained for

15k gradient steps at global batch size 128 on 2 L40S GPUs for roughly 1–2 days of wall-clock time per run.

Action tokenizer training. SALT tokenizers (14 configurations across the $n_g \times n_{\text{emb}}$ sweep) and the ablation tokenizers were each trained on 1 L40S GPU for 4–8 hours (depending on codebook size), for a total of roughly 150 GPU-hours across all tokenizer runs.

Probes. Each 5-fold verb-probe (Transformer-based) takes roughly 20 minutes on a single L40S; the full set of probes reported in Tables 1, 2, 4 and Figure 3 totals ≈ 80 GPU-hours.

Total. Aggregate compute, including failed runs and ablation runs not reported in the main text, is approximately 1,500 L40S GPU-hours.

K Use of AI Assistants

AI coding assistants were used to accelerate routine engineering tasks: writing plotting scripts, SLURM submit-script scaffolding, LaTeX formatting, and proofreading the manuscript. All scientific contributions — research questions, experimental designs, claims, analyses, and result interpretations — are the authors’.