MONAD: ONE SHARED DIFFUSION VISION-LANGUAGE MODEL FOR CONTINUOUS CONTROL

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language-action policies begin with a vision-language backbone that emits discrete tokens and must end in continuous control. Every current answer pays a different price. A second module holds action-specific parameters, and its gradients degrade the pretrained backbone; the common remedy blocks them at the cost of fixing the separation. A symbol vocabulary commits the trajectory to bins whose errors carry no fixed physical size. One that keeps the backbone and swaps its objective writes a second target into the representation that language modeling trained. We observe that none of these prices is necessary. A diffusion vision-language model already reconstructs masked spans inside its own hidden space, and a complete action vector is a span like any other. We introduce MONAD, which represents each control timestep as one continuous latent token and reconstructs temporally ordered action blocks with the same masked-token procedure the backbone applies to language. Two position-wise projections carry the coordinate mapping and hold 0.45% of model parameters. Action experts in π_0 and $\pi_{0.5}$ instead allocate 16.5% and 19.2%. Layer probes place the cross-time response inside the shared backbone. MONAD adds no action expert, tokenizer, or codec, and replaces no pretrained objective. On standard benchmarks, MONAD reaches 98.15% on regular LIBERO, 85.9% on the 10,030-task LIBERO-Plus aggregate, and 4.142 average completed chain length on CALVIN ABC–D.

1 INTRODUCTION

Vision-language-action policies inherit perception and language grounding from a pretrained vision-language backbone, and that backbone emits discrete tokens. Robot control needs continuous commands. Bridging the two is the field's central design decision, and every current answer pays for it somewhere. One route adds a module. One reuses the output head over a symbol vocabulary. One keeps the backbone and replaces its pretraining objective.

The dominant route adds a module. In π_0 and $\pi_{0.5}$ it is a flow-matching action expert attached to the backbone (Black et al., 2024; Physical Intelligence et al., 2025), and X-VLA fills the same role with a 24-layer action transformer (Zheng et al., 2025). The expert carries the temporal computation, holds action-specific parameters, and its gradients damage the pretrained backbone; Knowledge Insulation stops them, which removes the conflict while fixing the separation (Driess et al., 2025). A second route reuses the output head over a symbol vocabulary. OpenVLA quantizes each action dimension into bins and lets the language head emit them (Kim et al., 2024), and FAST compresses a trajectory with a discrete cosine transform and byte-pair encoding (Pertsch et al., 2025). Both commit the trajectory to symbols whose entries carry no fixed physical displacement. A third route, taken by Dream-VLA, keeps the backbone and swaps its objective, training the trajectory with flow matching in place of the token objective that produced the backbone (Ye et al., 2026). That writes a second target into the representation language modeling trained.

We take the opposite position. Neither a second module nor a second objective is necessary once the action variable lives in the backbone hidden space. We introduce MONAD, a blockwise reconstruction policy in which one complete action vector becomes one continuous latent token. Two position-wise multilayer perceptrons map between control coordinates and the backbone width, and the backbone performs every cross-time computation. The interfaces hold 0.45% of model parameters, against 16.5% and 19.2% of action-specific parameters in π_0 and $\pi_{0.5}$ (Appendix B.9). A layer-wise finite-

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

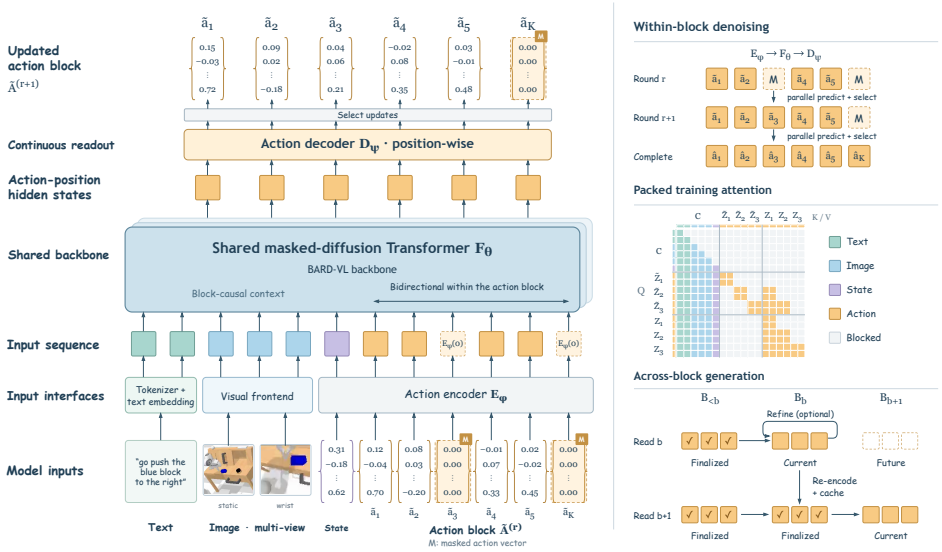


Figure 1: **MONAD reconstructs continuous action blocks inside a shared diffusion vision-language model.** The figure connects multimodal inputs to the shared backbone, position-wise continuous action readout, within-block selection, packed training attention, and across-block generation. Amber tiles denote complete action vectors; dashed "M" tiles denote masked vectors.

difference probe confirms the division of labor: the interface responds only at the perturbed timestep, and the cross-time response appears in the shared backbone (Section 4.5).

The routes also differ in how many positions a trajectory occupies. Writing a length-H, dimension-D trajectory with one token per scalar consumes HD positions, and compressed symbol routes spend a count their encoding fixes. The published continuous routes and MONAD both spend one position per control timestep, so a trajectory of H steps costs H positions. A block then covers a contiguous run of control steps, and the corruption mask decides which of those vectors the model reconstructs.

Latent action pretraining shares the goal of a unified action space but separates the learner of that space from its user: LAPA, IGOR, UniVLA, Moto, villa-X, and CAT each train a tokenizer first and consume it second (Ye et al., 2025; Chen et al., 2024; Wang et al., 2025; Chen et al., 2025b;a; Yang et al., 2026). MONAD co-trains its encoder and decoder with the backbone under supervision in normalized action coordinates.

Generation reuses the procedure a diffusion backbone already applies to language. We divide the horizon into temporal blocks that advance in order, and the backbone reconstructs positions inside a block together, which lets one control unit form as a whole. Training packs the multimodal prefix, a corrupted action copy, and a clean copy into a single forward pass under a structured attention mask, so every block receives supervision at once. At inference, the policy reconstructs one block at a time and conditions later blocks on committed earlier ones.

MONAD reaches competitive accuracy on standard benchmarks. It averages 98.15% on regular LIBERO across 2,000 episodes and 85.9% on the 10,030-task LIBERO-Plus aggregate (Table 1), and reaches 4.142 average completed chain length on all 1,000 CALVIN ABC–D sequences. Two controlled ablations locate what that capability depends on (Section 4.3). Enlarging the action block at a fixed chunk raises regular-LIBERO success from 38.5% to 73.0%. A second coordinate objective conflicts with the first at hidden-gradient cosine -0.891 and drives CALVIN average chain length from 2.970 to 0.237. Both ablations use one training seed and small evaluation budgets. Appendix B reports further fixed-checkpoint screens on iterative refinement, action-token context, and wrist-camera blackout.

The contributions are threefold.

- **One backbone, one objective.** We represent each control timestep as one continuous latent token and co-train the action interfaces with a shared diffusion vision-language backbone under its own masked-token objective. We add no action tokenizer, codebook, or external

action expert, and we replace no pretrained objective. The interfaces carry 0.45% of model parameters.

- **Locating the action computation.** We provide interface-level diagnostics that separate a temporally local projection path from backbone-driven temporal computation, and we quantify action-specific parameter share across representative policies.
- **Continuous block reconstruction.** We define corruption, visibility, and loss for continuous action blocks and supervise every block in one packed forward pass, with an iterative refinement procedure at inference.

Figure 1 summarizes the interface, the packed attention structure, and blockwise generation; Figure 2 contrasts the action variables and generation paths of representative policies.

2 Related Work

Latent action representations. Latent action methods learn a compact action variable from video and hand it to a separate control stage. LAPA and IGOR derive the variable from action-free video and from image-goal pairs respectively, and both transfer it to real actions through a second stage (Ye et al., 2025; Chen et al., 2024); UniVLA, Moto, villa-X, and CAT add codebooks, motion tokens, proprioceptive grounding, and a frozen trajectory codec (Wang et al., 2025; Chen et al., 2025b;a; Yang et al., 2026). All of them fix the learned variable before policy learning, so the encoder that defines the control representation is never optimized against the control objective. MONAD co-trains its encoder and decoder with the backbone, so the representation used for control is the one the policy optimizes.

Continuous actions through a second module. This family keeps continuous actions and gives the second module the trajectory objective. Generation by denoising predates vision-language policies (Chi et al., 2023). π_0 and $\pi_{0.5}$ pair a vision-language model with a flow-matching action expert (Black et al., 2024; Physical Intelligence et al., 2025); SmolVLA follows the same division at smaller scale (Shukor et al., 2025), and X-VLA applies a 24-layer action transformer with embodiment-specific soft prompts (Zheng et al., 2025). OpenVLA-OFT reads continuous outputs through a regression head on the shared model (Kim et al., 2025), and HybridVLA optimizes noise prediction alongside discrete action prediction (Liu et al., 2025). Knowledge Insulation reports that action-expert gradients degrade the pretrained backbone and blocks them, which preserves the representation by fixing the architectural separation (Driess et al., 2025). Across this family, parameters outside the backbone generate the trajectory; MONAD places that computation inside it.

Discrete action interfaces. The third family expresses control as symbols the language model already emits. OpenVLA quantizes each action dimension into bins and reuses the language head (Kim et al., 2024); FAST compresses a trajectory with a discrete cosine transform and byte-pair encoding (Pertsch et al., 2025); OAT learns ordered tokens and separates decodability from compression (Liu et al., 2026); Discrete Diffusion VLA and LLaDA-VLA generate action tokens with masked discrete diffusion (Liang et al., 2026; Wen et al., 2025). These interfaces define the training target in symbol space, where one wrong symbol need not match any fixed physical displacement. MONAD supervises directly in normalized action coordinates.

Blockwise generation and packed training. Blockwise diffusion combines ordered generation across blocks with parallel reconstruction inside one. BARD converts an autoregressive vision-language model into a diffusion model through progressive block merging and distillation, with mixed clean, corrupted, and masked states under a packed sequence (Chen et al., 2026). BlockVLA applies block diffusion to discretized actions (Wang et al., 2026), and TBD-VLA adds temporal inpainting for asynchronous execution (Lee et al., 2026). Block causality, within-block parallelism, and packed teacher forcing are established components; MONAD changes what occupies each block position to one complete continuous action vector.

Position. Prior work adds a second module, reuses the output head over a symbol vocabulary, or keeps the backbone and trains the trajectory under a new objective. Dream-VLA comes closest, because it drops the action expert and leaves the architecture untouched, but it replaces the backbone's

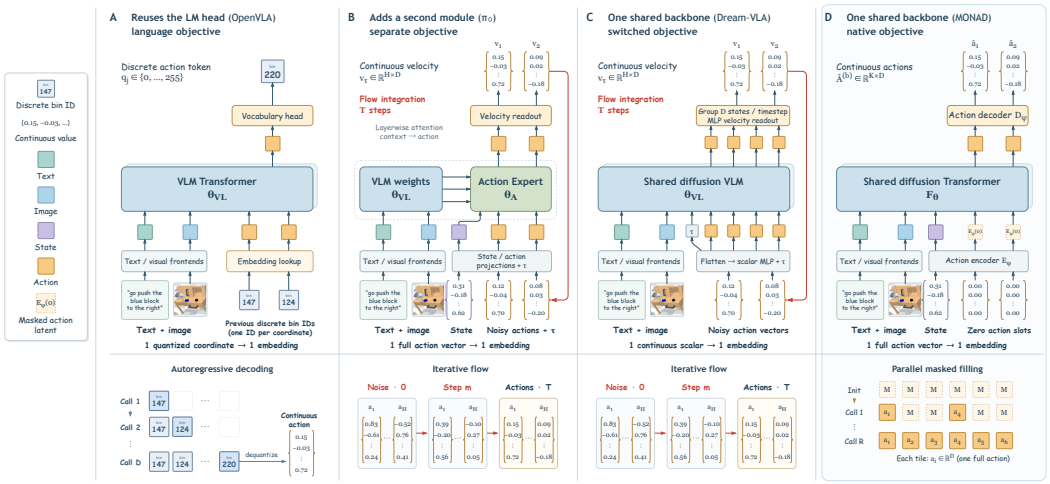


Figure 2: **MONAD keeps both the backbone and its pretraining objective.** The panels compare four routes to continuous control. OpenVLA reuses the language head over a symbol vocabulary. π_0 adds a second module that owns the trajectory. Dream-VLA keeps one backbone but switches its objective. MONAD keeps one backbone and its own objective. The panels summarize architectural contracts; they do not imply matched training or performance.

own objective with flow matching (Ye et al., 2026). MONAD keeps both the backbone and its objective, so the operator that models language reconstructs the trajectory under the same masked procedure. We evaluate on standard benchmarks and report fixed-checkpoint screens; we do not claim matched training or compute against methods that add a second module.

3 Method

MONAD generates continuous control with the shared diffusion vision–language backbone alone. Its computational unit is one complete action vector at one control timestep, which we call a latent token: a position-wise encoder maps it into the backbone hidden space, and a position-wise decoder reads it back out. The backbone computes every interaction across action positions and multimodal conditions, and no separate action expert or action tokenizer exists. We adopt the blockwise topology and corruption schedule of the BARD family Chen et al. (2026) and place one continuous action vector at each block position. The remaining subsections specify the representation, the training visibility, the corruption objective, and the decoding procedure.

3.1 Preliminaries and Continuous Action Representation

Given images $\mathcal{I}$, a language instruction l, and proprioceptive state s, the policy predicts an action chunk $A = (a_1, \ldots, a_H) \in \mathbb{R}^{H \times D}$. A multimodal prefix $C = (c_1, \ldots, c_P)$ contains the visual, language, and state representations, and embodiment information may enter this prefix when enabled. The training horizon H specifies the prediction contract; the number of actions executed before observing the environment again is a separate control setting, and we do not identify a long predicted chunk with a long open-loop execution interval.

Masked reconstruction learns to recover selected entries from a partially observed or corrupted sequence. Here, the selected entry is an entire real-valued vector, rather than an action coordinate represented by a vocabulary symbol, and we use the term *continuous blockwise reconstruction* for this procedure. The continuous representation and use of a diffusion-pretrained backbone do not, by themselves, define a Gaussian diffusion process or a normalized likelihood over trajectories.

An action encoder E_ϕ and decoder D_ψ connect the control coordinates to the backbone hidden dimension d:

$$z_i = E_\phi(a_i) \in \mathbb{R}^d, \qquad \widehat{a}_i = D_\psi(h_i) \in \mathbb{R}^D. \tag{1}$$

Both are multilayer perceptrons applied independently at each position. Consequently, a chunk occupies H latent tokens, one per control timestep, and temporal mixing takes place in the shared Transformer F_θ. The interface uses no action vocabulary, vector-quantization codebook, or separately pretrained action autoencoder. Encoding all coordinates together permits the encoder and decoder to mix coordinates within a timestep; whether this representation improves control is an empirical question.

For heterogeneous embodiments, actions are mapped into a common maximum-width coordinate layout. A dimension-validity indicator $q_{n,d}$ excludes absent coordinates and a timestep-validity indicator $v_{n,i}$ excludes padded trajectory positions, so these masks describe which coordinates and positions are supervised rather than implying identical physical action semantics across embodiments. Normalization and coordinate conversion are part of each dataset's action contract.

3.2 Blockwise Modeling and Packed Teacher Forcing

We divide the horizon into $M = \lceil H/K \rceil$ temporal blocks, with block index $b(i) = \lfloor (i-1)/K \rfloor$. Actions within a block interact bidirectionally, while generation proceeds from earlier to later blocks. The reconstruction of block b is conditioned on the multimodal prefix, completed earlier blocks, and the current corrupted block. This gives a controllable unit of parallel prediction between one-timestep generation and whole-chunk reconstruction.

Training packs a corrupted and a clean action copy after a single prefix, $X = [C, \widetilde{Z}, Z]$ with $\widetilde{Z} = E_\phi(\widetilde{A})$ and $Z = E_\phi(A)$. The copies share action position identifiers, but their attention permissions differ. A corrupted query at position i reads the prefix, its own block $\{\widetilde{z}_j : b(j) = b(i)\}$, and strictly earlier clean blocks $\{z_j : b(j) < b(i)\}$; a clean query reads the prefix and clean positions up to its own block, $\{z_j : b(j) \le b(i)\}$. Predictions are read only from the corrupted branch, so the clean copy supplies teacher-forced representations to later blocks while its current-block target stays hidden. Invalid keys are excluded from attention, which prevents a prediction from recovering its target through the duplicated branch.

The prefix uses causal attention within its valid positions and cannot attend to either action copy, so action queries read multimodal representations at every Transformer layer while action values leave prefix representations unchanged within that forward pass. Shared parameters still let the action objective update the functions that process the prefix in later training steps, which separates the forward conditioning structure from the backward parameter-update path.

The packed sequence evaluates all block reconstruction tasks in one backbone forward pass over $P + 2H$ positions, removing the need for a separate training forward per block. Packing reorganizes the sequence without reducing attention cost, because dense attention storage and computation both scale with the expanded length.

3.3 Corruption and Coordinate Reconstruction

Each training example receives one corruption level per block. Following block-stratified sampling, draw $u_{n,j} \sim \mathcal{U}(0,1)$ for $j = 0, \ldots, M-1$, construct $\bar{t}_{n,j} = (j + u_{n,j})/M$, and randomly permute these levels across temporal blocks. The resulting $t_{n,b}$ is clipped below by ϵ. Every block therefore receives a different stratum within an example, while the permutation avoids assigning a particular corruption range permanently to an early or late block. The level controls corruption and loss weighting; the native action-block path does not add a timestep embedding to action inputs.

We distinguish the two corruption contracts used in the experiments. In *mask-only reconstruction*, each position is selected with probability $t_{n,b(i)}$ and replaced by the all-zero action vector; unselected positions retain their clean values. The selected positions form the supervision mask $m_{n,i}$. The encoder processes the zero vector through the same E_ϕ as other inputs: it is not a separate learned mask token, so a zero-valued action and an unknown action share one encoded value in this path.

In *mixed reconstruction*, each position is assigned a clean, visible-substitution, or masked state. Before numerical flooring and renormalization, their probabilities are

$$\begin{aligned} p_{\text{clean}}(t) &= 1 - \sin(\pi t/2), \\ p_{\text{sub}}(t) &= \sin(\pi t/2) + \cos(\pi t/2) - 1, \\ p_{\text{mask}}(t) &= 1 - \cos(\pi t/2). \end{aligned} \tag{2}$$

Clean positions retain a_i, masked positions receive zero, and substitution positions receive a configured replacement vector. Both substitution and masked positions are supervised, while clean positions supply context. The reference reconstruction runs use mixed corruption, and the later continuation uses mask-only reconstruction. Appendix A records the variant details.

For the regression configuration, the per-position error is a weighted average over valid coordinates:

$$\ell_{n,i} = \frac{\sum_d q_{n,d}\omega_d \, \rho(\widehat{a}_{n,i,d} - a_{n,i,d})}{\max(\sum_d q_{n,d}\omega_d, \delta)}, \tag{3}$$

where ρ is Smooth-L_1, ω_d specifies coordinate weights, and δ protects an empty denominator. The action objective is

$$\mathcal{L}_{\text{act}} = \frac{1}{N} \sum_{n=1}^{N} \frac{\sum_i v_{n,i} m_{n,i} w(t_{n,b(i)}) \ell_{n,i}}{\max(\sum_i v_{n,i}, 1)}, \tag{4}$$

with $w(t) = 1/t$ under mask-only corruption and either $w(t) = 1$ or $1/t$ under mixed corruption, according to the experimental arm. The denominator counts all valid positions, not only supervised ones, and Equation 4 defines the training objective without claiming an unbiased estimate of an uncorrupted reconstruction risk.

The action loss propagates through the output interface, shared backbone, and input interface wherever parameters are trainable, and frozen layers remain part of the computation without being updated. The native training branch uses action supervision without a language-generation loss, so sharing a pretrained backbone is not evidence that its language or reasoning capabilities are preserved. Appendix A documents the gripper-objective variants separately from the all-coordinate regression configurations.

3.4 Blockwise Decoding and Closed-Loop Execution

At inference, the policy prefills the multimodal prefix into a key–value cache and initializes the requested action positions to zero, then reconstructs blocks in temporal order. On each proposal round, the shared backbone reads the current block plus the cached prefix and earlier committed blocks, and the decoder predicts every position in the current block in parallel. A commitment rule selects pending positions and inserts their predicted vectors; any remaining pending positions are committed on the last round.

The evaluated BARD configurations rank proposals with a small token-risk head that predicts the logarithm of the per-position reconstruction error under a detached Smooth-L_1 objective, so its auxiliary loss trains the estimator without supplying an additional gradient to the backbone. The negative predicted log-risk scores each proposal; it is not a calibrated probability of action correctness.

Optional refinement remasks low-scoring positions after the initial proposals and reconstructs them with the same backbone, and the acceptance, cooldown, and stopping rules are decoding settings rather than an additional trajectory model. Once a block is committed, its clean vectors are re-encoded into the persistent cache if a later block requires them, while intermediate proposals use a read-only cache so that temporary candidates never become conditions for later blocks. After decoding, the controller executes the configured action prefix and replans from a new observation. Figure 7 summarizes the risk head and the refinement rule, and Appendix A reports the proposal budget, refinement settings, prediction horizon, and execution horizon separately because each affects the realized control and inference cost.

4 Experiments

The experiments measure how the policy performs on standard benchmarks, which training variables govern closed-loop control, and where the action computation lives. Every experiment uses a fixed

checkpoint and retains its task, window, and intervention contract. Appendix B reports further fixed-checkpoint screens on iterative refinement, wrist-camera blackout, action-token context, and interface representation.

4.1 Evaluation protocol and scope

The main benchmark screen covers regular LIBERO, LIBERO-Plus, and CALVIN ABC–D (Liu et al., 1939; Fei et al., 2025; Mees et al., 2022). Regular LIBERO uses 50 episodes per task across its four suites; LIBERO-Plus covers all 10,030 catalog tasks; CALVIN covers the 1,000 recorded five-task sequences. We report success rate, average completed chain length (AvgLen), and sequence success rates (SR@k) where the benchmark provides them. The focused inference screen uses one fixed LIBERO checkpoint and five episodes per task on tasks 0 and 1 of LIBERO-Spatial and LIBERO-10. The wrist-camera screen uses paired LIBERO-Spatial and LIBERO-10 episodes with the same task and initial state in the clean and blackout conditions.

The selected checkpoints use a recorded global batch size of 1,536, obtained from eight GPUs with batch 192 per GPU and no gradient accumulation. The LIBERO screens use ten-action chunks and execute ten actions before replanning. We train the selected checkpoints from a cleaned multi-source mixture, and the selected LIBERO and CALVIN branches add a 30,000-update continuation before their 5,000-update target finetuning. Appendix B.1 reports the source mixture, update stages, and shared optimization settings.

4.2 Main benchmark results

We select one complete result per benchmark family using the highest-scoring checkpoint whose evaluation identity and coverage are available in the audit records. The regular-LIBERO checkpoint is the later BrAD branch with drop-nine-frame preprocessing and no state-Z masking; it evaluates 2,000 episodes, 50 per task, under relative control with agent-view and wrist cameras. The CALVIN checkpoint is the later cleaned-parent continuation with closure BCE and frozen action interfaces and backbone layers 0–1; it evaluates all 1,000 official ABC–D sequences with absolute actions. These are benchmark selections, while the historical ablations below retain their own recipe identities.

Table 1 puts the three benchmark families in one comparison. The external rows reproduce published evaluations and use their reported checkpoints and protocols; their data preparation, action interfaces, and trial contracts differ from MONAD, so the table gives provenance-matched reference points. The caption records the coverage of each row. The selected regular-LIBERO checkpoint evaluates 2,000 episodes under relative control with agent-view and wrist cameras, and the selected CALVIN checkpoint evaluates all 1,000 official ABC–D sequences with absolute actions under H10 execution and D4/R5 decoding. The LIBERO-Plus mixed arm reports 2,106/2,402 Spatial, 2,216/2,518 Object, 2,150/2,591 Goal, and 2,143/2,519 LIBERO-10 successes, for 8,615/10,030 overall.

4.3 Block topology governs closed-loop control

Both ablations in this section come from an earlier BARD-B4 training family with one training seed each and smaller evaluation budgets, so we read them as controlled comparisons inside one family and keep them out of the benchmark table.

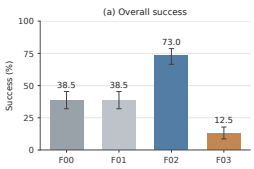
This ablation fixes the horizon at $H = 50$, the training contract, and the downstream evaluation, and changes only the block layout from ten blocks of five to five blocks of ten. Both arms finetune for 2,600 updates and then evaluate on 200 regular-LIBERO episodes across 40 tasks and on 1,000 CALVIN ABC–D sequences with a 25-step execution horizon.

Larger blocks raise closed-loop success (Figure 3). On regular LIBERO the ten-by-five layout reaches 73.0% (146/200) against 38.5% (77/200), a gain of 34.5 points; 32 of 40 task entries improve, seven tie, and one declines. CALVIN moves the same way: average completed chain length rises from 0.055 to 1.945, SR@1 from 5.1% to 78.2%, and SR@5 from 0 to 11.7%.

Two controls bound this reading. An arm that changes only the visible corruption distribution keeps regular-LIBERO success at 38.5% and reaches 0.147 average chain length, so noise form alone does not reproduce the gain. A third arm uses twenty blocks of five with a 100-step action chunk and drops to 12.5% and 0.115, but it changes the horizon together with the block count, so we attribute its

Method / checkpoint	Regular LIBERO (%)					LIBERO-Plus (%)					CALVIN ABC–D					
	Spatial	Object	Goal	LIBERO-10	Avg.	Spatial	Object	Goal	LIBERO-10	Overall	AvgLen	SR@1	SR@2	SR@3	SR@4	SR@5
Octo (Team et al., 2024)	78.9	85.7	84.6	51.1	75.1	–	–	–	–	–	–	–	–	–	–	–
OpenVLA (Kim et al., 2024)	84.7	88.4	79.2	53.7	76.5	19.4	14.0	15.1	14.3	15.6	–	–	–	–	–	–
π_0 (Black et al., 2024)	96.8	98.8	95.8	85.2	94.2	60.7	61.4	44.9	48.4	53.6	–	–	–	–	–	–
$\pi_{0.5}$ (Physical Intelligence et al., 2025; Hosseinkhani et al., 2026)	97.2	99.0	97.8	96.0	97.5	–	–	–	–	–	–	–	–	–	–	–
π_0-Fast (Pertsch et al., 2025)	96.4	96.8	88.6	60.2	85.5	74.4	72.7	57.5	43.4	61.6	–	–	–	–	–	–
OpenVLA-OFT (Kim et al., 2025)	97.6	98.4	97.9	94.5	97.1	84.0	66.5	63.0	66.4	69.6	–	–	–	–	–	–
OpenVLA-OFT$_w$ (Fei et al., 2025)	–	–	–	–	–	62.5	56.0	53.3	52.2	55.8	–	–	–	–	–	–
OpenVLA-OFT$_m$ (Fei et al., 2025)	–	–	–	–	–	75.4	77.1	56.2	63.9	67.9	–	–	–	–	–	–
NORA (Hung et al., 2025)	–	–	–	–	–	47.6	34.4	38.8	36.3	39.0	–	–	–	–	–	–
RIPT-VLA (Tan et al., 2025)	98.6	98.6	**99.0**	93.8	97.5	85.8	64.3	58.0	67.5	68.4	–	–	–	–	–	–
SmolVLA 2.25B (Shukor et al., 2025)	93.0	94.0	91.0	77.0	88.75	–	–	–	–	–	–	–	–	–	–	–
UniVLA (Wang et al., 2025)	95.4	98.8	93.6	94.0	95.5	55.5	36.7	40.7	39.9	42.9	4.41	**98.9**	**94.8**	**89.0**	82.8	75.1
WorldVLA (512^2) (Cen et al., 2025)	87.6	96.2	83.4	60.0	81.8	32.5	28.6	31.8	8.2	25.0	–	–	–	–	–	–
LLaDA-VLA (Wen et al., 2025)	–	–	–	–	–	–	–	–	–	–	4.01	95.6	87.8	79.5	73.9	64.5
X-VLA (Zheng et al., 2025)	98.2	98.6	97.8	**97.6**	98.1	–	–	–	–	–	**4.43**	97.1	92.6	88.5	**84.4**	**78.8**
Discrete Diffusion VLA (Liang et al., 2026)	97.2	99.4	96.8	92.2	96.4	–	–	–	–	–	–	–	–	–	–	–
Dream-VLA (Ye et al., 2026)	97.6	98.8	97.2	95.0	97.2	–	–	–	–	–	–	–	–	–	–	–
IMLE-VLA (Hosseinkhani et al., 2026)	98.0	99.8	98.2	96.0	98.0	–	–	–	–	–	–	–	–	–	–	–
EFFVLA (Sun et al., 2026)	–	–	–	–	–	84.4	84.5	75.5	74.7	79.8	–	–	–	–	–	–
MONAD, selected	**99.0**	**100.0**	98.8	94.8	**98.15**	**87.7**	**88.0**	**83.0**	**85.1**	**85.9**	4.142	98.6	93.8	83.2	73.7	64.9

Table 1: Main results across regular LIBERO, LIBERO-Plus, and CALVIN ABC–D. Regular-LIBERO Avg. is the arithmetic mean of its four suite scores. Rows run approximately from early to late release year, with variants of one family adjacent and the selected MONAD result last. Bold marks the maximum among displayed methods per column; external rows retain their published protocols and serve as reference points. The $\pi_{0.5}$ and IMLE-VLA rows use the $H = 10$ comparison of Hosseinkhani et al. (Hosseinkhani et al., 2026). Discrete Diffusion VLA and Dream-VLA report regular-LIBERO suites and LLaDA-VLA reports CALVIN, each retaining its source coverage. EFFVLA uses its own suite-level LIBERO-Plus evaluation, while the MONAD and LIBERO-Plus rows report the official 10,030-task aggregate with Long labeled LIBERO-10. CALVIN reports average completed chain length and sequence success rates for ABC→D.



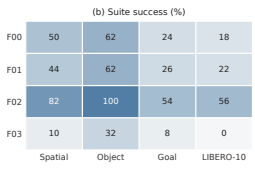


Figure 3: **Enlarging the action block raises regular-LIBERO success at fixed 50-step chunk length.** Overall and per-suite success over 40 tasks with five episodes each, with archived episode-level Wilson 95% intervals. F00 uses uniform corruption with ten blocks of five, F01 changes only the corruption to Gaussian, F02 uses five blocks of ten, and F03 changes both topology and the 100-step chunk. One training seed per arm; intervals exclude seed variation and task clustering.

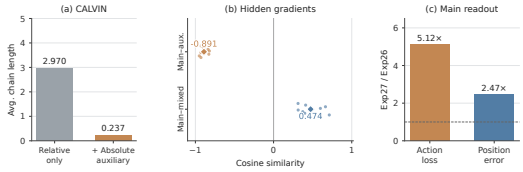


Figure 4: **An auxiliary absolute-action objective opposes the relative-action objective in this matched intervention.** (a) Average completed CALVIN chain length over 1,000 sequences per arm. (b) Main-versus-auxiliary and main-versus-combined hidden-gradient cosine on eight fixed windows; diamonds pool all hidden elements. (c) Main action-loss and one-pass position-error ratios against the Exp26 baseline. The main interface stays frozen and bitwise equal to pretraining in both arms.

decline to neither variable alone. Block partition therefore acts as a training variable that governs closed-loop behavior in this family. These single-seed results do not show that ten-by-five is optimal for blockwise models in general, and they do not compare against direct regression or flow-matching policies.

4.4 A SECOND COORDINATE OBJECTIVE CONFLICTS WITH THE FIRST

This intervention matches the two runs on every policy setting except one. Exp26 supervises relative actions alone; Exp27 adds a frozen auxiliary decoder with an absolute-action target at weight 0.5.

The main action interface stays bitwise equal to the pretraining interface in both runs. We measure hidden gradients on eight fixed CALVIN windows and task performance on 1,000 CALVIN ABC–D sequences.

The added target produces opposing gradients at the shared hidden state (Figure 4). The pooled cosine between the main and auxiliary hidden gradients is -0.891, 99% of action-slot cosines are negative, and the weighted auxiliary-to-main gradient-norm ratio is 0.881. The combined gradient still aligns with the main objective at cosine 0.474, yet the two runs move the backbone along directions whose cosine is 0.225. Task performance follows the conflict: average completed chain length falls from 2.970 to 0.237, teacher-forced main loss rises 5.12-fold and one-pass position error rises 2.47-fold on the same windows. An auxiliary coordinate target therefore reshapes the shared optimization direction and weakens the main readout.

4.5 Temporal-response and non-local response diagnostic

For temporal attribution, we perturb one input action coordinate and measure the output response R. The non-local response ratio is

$$\mathrm{NLR} = \frac{\sum_{|t-t_0|\geq 6} \|R_{t,:}\|_2^2}{\|R\|_F^2}. \tag{5}$$

We compare a temporally local encoder–decoder path with the same interface plus one backbone forward pass. The legacy non-BARD MONAD checkpoints yield interface NLR of zero and full-path NLR of 1.442%, 1.467%, and 7.135% for the pretraining checkpoint, the absolute-action downstream configuration, and the relative-action downstream configuration, respectively. Repeated forwards reproduce these values with zero jitter, and an epsilon sweep from 0.01 to 0.04 moves each mean by less than 0.006 percentage points (Appendix B.9). Layer probes locate larger far-time fractions at middle and later probe locations. Within-model probes for π_0 and $\pi_{0.5}$ (Black et al., 2024; Physical Intelligence et al., 2025) likewise yield zero interface NLR and nonzero full-expert NLR, 8.14% and 7.14%. These within-model contrasts place temporal mixing in the action expert for the π models and in the shared backbone for MONAD. Figure 11(c,d) shows the MONAD response by layer and probe.

5 Conclusion

MONAD produces continuous control without a second module or a second objective: one complete action vec tor becomes one continuous latent token in a shared diffusion vision-language backbone, whose action int erfaces carry 0.45% of model parameters. It reaches 98.15% on regular LIBERO, 85.9% on the 10,030-t ask LIBERO-Plus aggregate, and 4.142 average completed chain length on CALVIN ABC–D. Two controlled wit hin-family ablations support the same reading of where that computation belongs: enlarging the action bl ock at a fixed 50-step chunk raises regular-LIBERO success from 38.5% to 73.0% and lifts CALVIN averag e chain length from 0.055 to 1.945, while an auxiliary absolute-action objective produces opposing hidde n gradients at cosine -0.891 and drives CALVIN average chain length from 2.970 to 0.237. Both ablation s use one training seed and small evaluation budgets, so we keep them out of the benchmark table. The op en questions are efficiency and scale: we measure neither latency nor a common-budget comparison against policies that add a second module.

AI Use Statement

AI assistants were used for LaTeX editing, prose revision, and consistency checks on the manuscript text and its figures. They were not used to generate experimental results, to produce or modify the reported measurements, or to write code used to run the experiments. All numerical claims, experimental designs, and cited sources were verified by the authors, and the authors take full responsibility for the content of this work.

REFERENCES

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024. URL https://arxiv.org/abs/2410.24164.

Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, Deli Zhao, and Hao Chen. WorldVLA: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025. URL https://arxiv.org/abs/2506.21539.

Baoyou Chen, Hanchen Xia, Peng Tu, Haojun Shi, Liwei Zhang, Yuxuan Yao, Weihao Yuan, and Siyu Zhu. BARD: Bridging AutoRegressive and diffusion vision-language models via highly efficient progressive block merging and stage-wise distillation. *arXiv preprint arXiv:2604.16514*, 2026. URL https://arxiv.org/abs/2604.16514.

Xiaoyu Chen, Junliang Guo, Tianyu He, Chuheng Zhang, Pushi Zhang, Derek Cathera Yang, Li Zhao, and Jiang Bian. IGOR: Image-GOal representations are the atomic control units for foundation models in embodied AI. *arXiv preprint arXiv:2411.00785*, 2024. URL https://arxiv.org/abs/2411.00785.

Xiaoyu Chen, Hangxing Wei, Pushi Zhang, Chuheng Zhang, Kaixin Wang, Yanjiang Guo, Rushuai Yang, Yucen Wang, Xinquan Xiao, Li Zhao, Jianyu Chen, and Jiang Bian. villa-X: Enhancing latent action modeling in vision-language-action models. *arXiv preprint arXiv:2507.23682*, 2025a. URL https://arxiv.org/abs/2507.23682.

Yi Chen, Yuying Ge, Weiliang Tang, Yizhuo Li, Yixiao Ge, Mingyu Ding, Ying Shan, and Xihui Liu. Moto: Latent motion token as the bridging language for robot manipulation. *arXiv preprint arXiv:2412.04445*, 2025b. URL https://arxiv.org/abs/2412.04445.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023. URL https://arxiv.org/abs/2303.04137.

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

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, Jinlan Fu, and Jingjing Gong. LIBERO-Plus: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025. URL https://arxiv.org/abs/2510.13626.

Kian Hosseinkhani, Qinhe Peng, George Shramko, Mehran Aghabozorgi, Jianing Qian, Tristan Engst, Alireza Moazeni, Dinesh Jayaraman, and Ke Li. IMLE-VLA: Fast single-step action generation for vision-language-action policies. *arXiv preprint arXiv:2609.10915*, 2026. URL https://arxiv.org/abs/2609.10915.

Chia-Yu Hung, Qi Sun, Pengfei Hong, Amir Zadeh, Chuan Li, U-Xuan Tan, Navonil Majumder, and Soujanya Poria. NORA: A small open-sourced generalist vision-language-action model for embodied tasks. *arXiv preprint arXiv:2504.19854*, 2025. URL https://arxiv.org/abs/2504.19854.

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

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

Sung-Wook Lee, Xuhui Kang, and Yen-Ling Kuo. TBD-VLA: Temporal block diffusion vision language action model. *arXiv preprint arXiv:2606.07895*, 2026. URL https://arxiv.org/abs/2606.07895.

Zhixuan Liang, Yizhuo Li, Tianshuo Yang, Chengyue Wu, Sitong Mao, Liuao Pei, Tian Nian, Shunbo Zhou, Xiaokang Yang, Jiangmiao Pang, Yao Mu, and Ping Luo. Discrete diffusion VLA: Bringing discrete diffusion to action decoding in vision-language-action policies. *arXiv preprint arXiv:2508.20072*, 2026. URL https://arxiv.org/abs/2508.20072.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. *arXiv preprint arXiv:2306.03310*, 1939. URL https://arxiv.org/abs/2306.03310.

Chaoqi Liu, Xiaoshen Han, Jiawei Gao, Yue Zhao, Haonan Chen, and Yilun Du. OAT: Ordered action tokenization. *arXiv preprint arXiv:2602.04215*, 2026. URL https://arxiv.org/abs/2602.04215.

Jiaming Liu, Hao Chen, Pengju An, Zhuoyang Liu, Renrui Zhang, Chenyang Gu, Xiaoqi Li, Ziyu Guo, Sixiang Chen, Mengzhen Liu, Chengkai Hou, Mengdi Zhao, KC alex Zhou, Pheng-Ann Heng, and Shanghang Zhang. HybridVLA: Collaborative diffusion and autoregression in a unified vision-language-action model. *arXiv preprint arXiv:2503.10631*, 2025. URL https://arxiv.org/abs/2503.10631.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *arXiv preprint arXiv:2112.03227*, 2022. URL https://arxiv.org/abs/2112.03227.

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

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

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. SmolVLA: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025. URL https://arxiv.org/abs/2506.01844.

Luoyang Sun, Guoyang Xia, Fengfa Li, Lei Ren, Xinyu Cui, Haifeng Zhang, Fangxiang Feng, Kaike Zhang, Kun Zhan, Yan Xie, Jun Wang, and Cheng Deng. What makes an efficient vla? navigating action-head design, scaling, and latency. *arXiv preprint arXiv:2609.13984*, 2026. URL https://arxiv.org/abs/2609.13984.

Shuhan Tan, Kairan Dou, Yue Zhao, and Philipp Krahenbuehl. Interactive post-training for vision-language-action models. *arXiv preprint arXiv:2505.17016*, 2025. URL https://arxiv.org/abs/2505.17016.

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

Ruiheng Wang, Shuanghao Bai, Haoran Zhang, Badong Chen, and Xiangyu Xu. BlockVLA: Accelerating autoregressive VLA via block diffusion finetuning. *arXiv preprint arXiv:2605.13382*, 2026. URL https://arxiv.org/abs/2605.13382.

Yuqi Wang, Xinghang Li, Wenxuan Wang, Junbo Zhang, Yingyan Li, Yuntao Chen, Xinlong Wang, and Zhaoxiang Zhang. Unified vision-language-action model. *arXiv preprint arXiv:2506.19850*, 2025. URL https://arxiv.org/abs/2506.19850.

Yuqing Wen, Hebei Li, Kefan Gu, Yucheng Zhao, Tiancai Wang, and Xiaoyan Sun. LLaDA-VLA: Vision language diffusion action models. *arXiv preprint arXiv:2509.06932*, 2025. URL https://arxiv.org/abs/2509.06932.

Tong Yang, Jingkai Jia, Yuecheng Xu, Xueyao Chen, Chi Zhang, and Wenqiang Zhang. Trajectory-level continuous action representation for robotic manipulation. *arXiv preprint arXiv:2608.24111*, 2026. URL https://arxiv.org/abs/2608.24111.

Jiacheng Ye, Shansan Gong, Jiahui Gao, Junming Fan, Shuang Wu, Wei Bi, Haoli Bai, Lifeng Shang, and Lingpeng Kong. Dream-VL & Dream-VLA: Open vision-language and vision-language-action models with diffusion language model backbone. *arXiv preprint arXiv:2512.22615*, 2026. URL https://arxiv.org/abs/2512.22615.

Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Sejune Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, Lars Liden, Kimin Lee, Jianfeng Gao, Luke Zettlemoyer, Dieter Fox, and Minjoon Seo. LAPA: Latent action pretraining from videos. *arXiv preprint arXiv:2410.11758*, 2025. URL https://arxiv.org/abs/2410.11758.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Ya-Qin Zhang, Jiangmiao Pang, Jingjing Liu, Tai Wang, and Xianyuan Zhan. X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025. URL https://arxiv.org/abs/2510.10274.

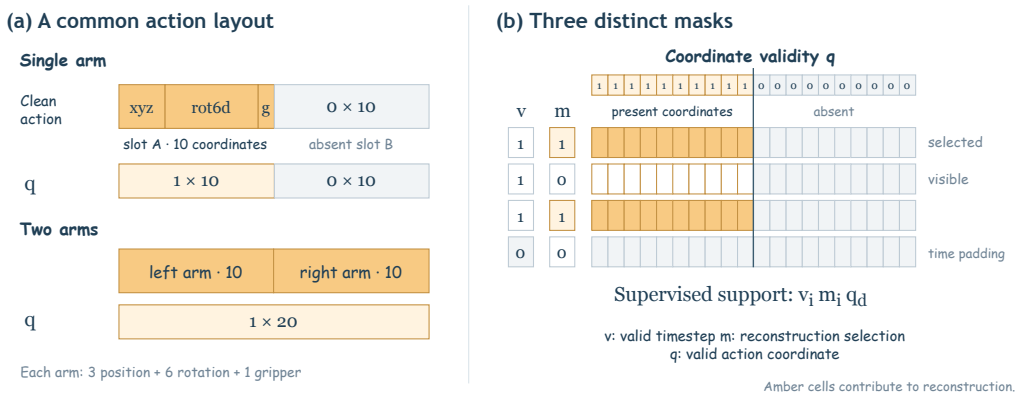


Figure 5: **Validity masks separate embodiment coordinates, reconstruction selection, and time padding.** The canonical layout provides fixed slots for single-arm and two-arm inputs; q, m, and v retain distinct meanings in the supervised objective.

A Implementation and Objective Details

A.1 Reference Configuration and Variant Identity

The reference configuration instantiates the native conditional action-block path with a BARD-VL-B4-Mask-4B-Instruct checkpoint and a Qwen3-VL-compatible multimodal layout. It uses $D = 20$, $H = 50$, $K = 10$, and $M = 5$. Each arm occupies ten canonical coordinates: three translations, a continuous six-dimensional rotation representation, and one gripper coordinate. Missing embodiment coordinates are masked. This coordinate layout is an interface convention, not a claim that all source datasets measure identical quantities before conversion.

Both the action encoder and action decoder have four linear layers with hidden width 1,024 and SiLU activations. They use no dropout. Proprioceptive state shares the action input projection because its width matches the action width; the separately declared state-layer count does not instantiate another state MLP in this shared-encoder setting. There are no action-boundary special tokens, embodiment-specific adapters, or embodiment soft prompts in the reference configuration. The first two language Transformer layers are frozen; the remaining enabled backbone parameters and action interfaces are trainable. No text-output loss is enabled.

The coordinate-loss weights are $(500, 500, 500, 10, 10, 10, 10, 10, 10, 1)$, repeated for the second arm. All valid coordinates, including the gripper, use the base Smooth-L_1 objective. These coefficients enter both the loss numerator and its valid-coordinate normalization. Therefore, they specify relative contributions.

A later H20 continuation configuration uses $H = 20$, $K = 10$, and $M = 2$, mask-only corruption with inverse-t weighting, a fully trainable backbone, and probability-valued gripper supervision. Checkpoint-specific execution and refinement overrides remain distinct from a training YAML's stored inference defaults. A configuration file documents a requested recipe; the corresponding resolved configuration and evaluation receipt identify an executed run.

A.2 Masking, Padding, and Attention

The mask-only branch uses $t_{n,b} \geq \epsilon = 10^{-3}$ as both its masking probability and the inverse-weight denominator. The implementation draws one categorical decision per complete action vector. In the mixed branch, the three probabilities in Equation 2 are individually floored at 10^{-3} and then normalized to sum to one. Visible substitution uses the replacement vector directly; it is neither token insertion/deletion nor a convex interpolation between a sampled replacement and the clean action. Mask-only weighting is always inverse-t in this native path, even if the mixed-loss-weighting field uses `none`.

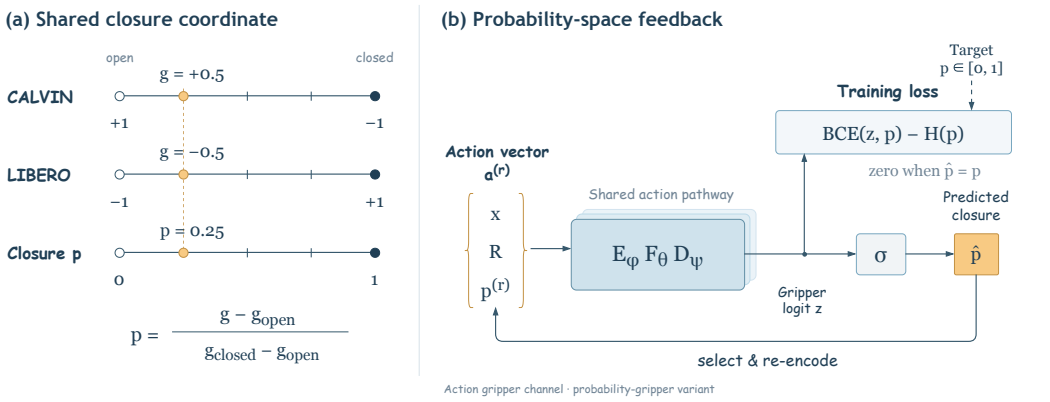


Figure 6: **The probability-gripper variant maps heterogeneous closure conventions to a shared coordinate.** The decoder predicts a logit, applies the sigmoid, and re-encodes the probability-valued closure coordinate for the next reconstruction round.

Clean inputs in invalid action dimensions are zeroed before encoding, and invalid dimensions are excluded from the loss denominator. Corruption is applied after the clean-input masking; thus the mixed-substitution branch can place replacement values in invalid dimensions of corrupted vectors. These values receive no direct coordinate supervision. They should not be described as universally zero throughout the corrupted forward pass. Invalid timesteps are hidden as attention keys and removed from supervision. To avoid a completely masked attention row, an invalid query can attend to itself; this numerical convention does not restore its validity.

The ordinary multimodal prefix is causal and isolated from both action copies. The corrupted branch reads its own block and strictly earlier clean blocks, while the clean branch reads its current and earlier clean blocks. Position identifiers for the two action copies are identical. Predictions are sliced from the corrupted copy only. This makes the clean copy a source of earlier-block context and prevents a same-block clean-target shortcut. Optional action-history modes have additional prefix rules and are outside the no-history reference configuration.

The current native input function applies the action projection directly, followed by the configured action-embedding processing. It does not inject an explicit corruption-time embedding, a pending/committed state embedding, or a separate learned action-mask vector. The all-zero sentinel therefore shares its encoded value with a legitimate zero action. Assessing a dedicated mask indicator is a useful future representation ablation; its effectiveness is not assumed here.

A.3 Probability-Valued Gripper Supervision

In the later probability-gripper variant, the head produces a logit r for a target closure coordinate $p \in [0, 1]$. Binary cross-entropy with logits (BCEWithLogits) gives the coordinate-level objective

$$\rho_{\text{grip}}(r, p) = \text{BCEWithLogits}(r, p) - \mathcal{H}(p) = \text{KL}(\text{Ber}(p) \,\|\, \text{Ber}(\sigma(r))) \,, \tag{6}$$

where $\mathcal{H}$ denotes Bernoulli entropy. Subtracting target entropy leaves the BCE gradient unchanged and sets the optimal soft-target error to zero. The implementation preserves this gradient while clamping tiny negative forward values caused by numerical cancellation. With hard targets, the entropy term is zero. At inference, logits are converted to closure probabilities before they are committed, re-encoded, or scored, so an iterative action state does not mix probability coordinates with raw logits. This loss replaces the selected gripper dimensions within the same weighted valid-coordinate average as the regression coordinates.

A.4 Risk Estimation and Decoder Accounting

The token-risk head applies LayerNorm followed by a two-layer MLP with hidden width 256, SiLU activation, and a scalar output. Its dropout rate is zero in the reference recipes. For supervised valid positions, its target is $\log(\ell_{n,i} + 10^{-6})$ and its Smooth-L_1 auxiliary objective is weighted by 0.05.

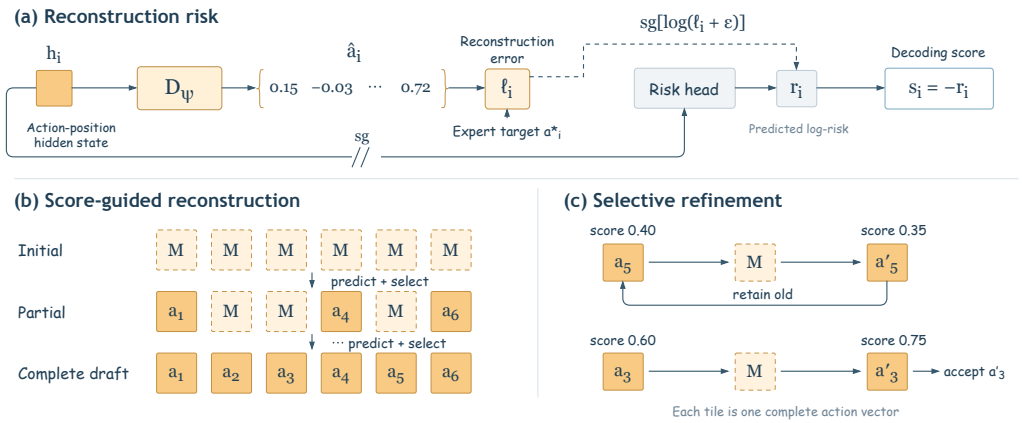


Figure 7: **Risk-guided decoding scores candidate action vectors and supports selective refinement.** The risk head predicts log reconstruction error from detached hidden states; the decoder selects pending positions, and refinement accepts a replacement only when its score improves. Values in the diagram are schematic.

Both target and hidden input are detached in the reference and H20 continuation recipes. Its reduction averages selected positions across the batch, unlike the per-example, all-valid-position denominator of the main reconstruction loss. This auxiliary prediction is a decoding score, not a likelihood term or an additional action sequence model.

The generic sampler dataclass has a `combined` confidence default based on hidden-state norm and prediction stability, while the evaluated recipes explicitly select `token_risk`. Similarly, a native conditional configuration with an omitted noise mode resolves to mixed corruption, while the H20 continuation explicitly selects mask-only corruption. Method descriptions use resolved experiment settings rather than generic defaults.

Within each block, the decoder begins with a per-round transfer budget that partitions the pending count as evenly as possible. The dynamic rule may increase that budget up to its configured cap using score thresholding. Thus a stored maximum proposal-round count does not necessarily equal the number of executed backbone reads. Visible positions remain revisable until block commit. Refinement then selects low-scoring eligible positions, substitutes the zero sentinel, and optionally accepts a new value only when the score improves. Cooldown, a decaying remask budget, patience, and best-state preservation can further affect the number of actual reads.

For M decoded blocks, at most R proposal rounds per block, and at most Q refinement reads per block, inference uses one prefix prefill, at most $M(R+Q)$ block reads, and $M-1$ clean-block commits in evaluation mode. The last block needs no cache commit because no subsequent block consumes it. This bound counts model calls, not equal-cost operations: the cache and block attention size change with position, and some configured rounds can perform no read. Actual latency and read counts must therefore be measured rather than inferred from the proposal budget alone.

A.5 What the Reconstruction Objective Estimates

For mask-only corruption, fix an example and the block corruption levels. Because the prediction depends on the mask, the identity relevant to one position is

$$\mathbb{E}_m\left[\frac{m_i}{t_i}\ell_i(m)\,\middle|\,t\right] = \mathbb{E}_m[\ell_i(m) \mid m_i = 1, t]. \tag{7}$$

It is the expected error conditional on that position being masked, averaged over the remaining corruption, rather than the error of an uncorrupted input. In mixed reconstruction, inverse-t is a chosen time weighting and is not the inverse of the probability of selecting either supervised corruption state. We therefore use Equation 4 as the explicit objective and do not interpret either variant as an unbiased estimator of an unspecified full-action risk.

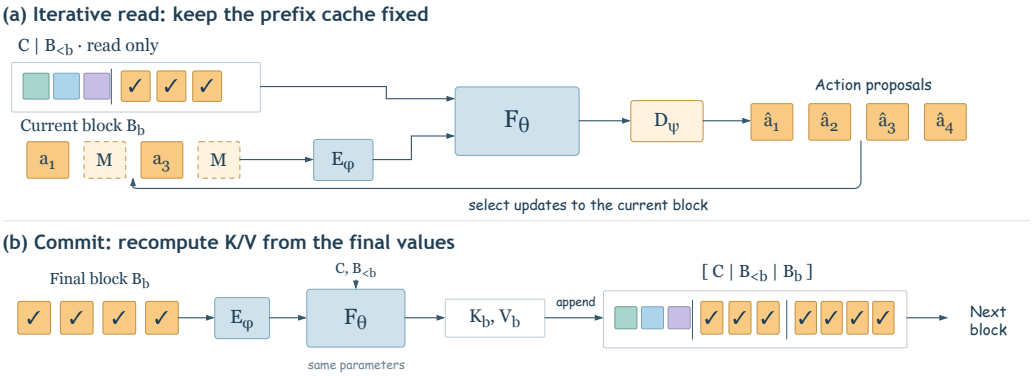


Figure 8: **MONAD keeps the prefix cache read-only during proposals and commits final values between blocks.** The current block reads the fixed prefix and earlier committed blocks; after commitment, MONAD re-encodes final actions and appends their keys and values for the next block.

B Additional Experimental Details

B.1 Training Recipe

The cleaned multi-source recipe, update stages, and shared optimization settings are summarized below. These tables document checkpoint identity and training contracts; they are not additional benchmark results.

Source group	Phys./log. entries/slots	Episodes	Frames	Configured weight	Normalized weight
DROID	1/2	97,685	27,348,488	0.30	0.3333
AgiBotWorld–Alpha	34/34	32,735	46,156,708	0.40	0.4444
RoboMIND AgileX	1/1	8,906	5,387,380	0.07	0.0778
RoboMIND Franka	1/1	5,333	943,842	0.10	0.1111
RoboMIND FR3	1/1	1,774	375,807	0.03	0.0333
Active total	38/39	146,433	80,212,225	0.90	1.0000
RoboMIND–UR (blocked)	0/0	n/a	n/a	0.10	excluded

Table 2: Cleaned multi-embodiment pretraining mixture. Physical entries are materialized dataset roots and logical slots are sampler views. The 34 AgiBot entries are the retained task subsets; `agibot-task_392` is blocked and therefore absent from the table. Episodes and frames are materialized counts, and the DROID counts are not duplicated across its two views.

B.2 Evidence and Checkpoint Identity

The focused LIBERO screen uses one fixed `pretrained_model/` checkpoint and 12 validated cells. The context-replay screen uses the current CALVIN checkpoint, 12 fixed windows, and two validated raw-record groups. The wrist-camera screens use the archived MONAD checkpoint and the public $\pi_{0.5}$ release, with paired task and initial-state contracts. The accompanying machine-readable summaries retain cell identities, episode counts, validation status, and raw-output paths for each result.

B.3 Pretraining Data Cleaning and Materialization

Before the multi-embodiment pretraining run, we built a source-preserving, plan-first release through a multi-stage audit of the raw roots, meaning the original dataset directories. The audit recorded source manifests and source-specific contracts. It checked metadata, episode catalogs, Parquet row counts, timestamps, finite action and state values, trajectory structure, and video inventories. Separate checks examined canonical actions for static intervals, auxiliary channels for motion, video evidence for visible motion, and semantic rules for protected phases. Automated checks emitted candidate

Stage	Data	Updates	Contract
Pretraining	Mixture in Table 2	60,000	Canonical 20D actions; valid-dimension masks for single-arm sources; DROID views aligned to 30 Hz
Continued pretraining (selected branch)	Same cleaned mixture	30,000	T20/block10×2 continuation; fresh optimizer; resolved branch recipe
Finetuning	CALVIN ABC–D	5,000	Absolute actions; $2 \times 2 \times 2$ objective/freezing sweep
Finetuning	Regular LIBERO	5,000	Relative actions; one target dataset per run

Table 3: Training stages and recipe identity. The base cleaned recipe has one 60,000-update pretraining stage followed by target-specific finetuning. The selected benchmark checkpoints add the labelled 30,000-update continuation before finetuning.

Setting	Value
Devices and batch	8 GPUs; 192 examples/GPU; global batch 1,536; no gradient accumulation
Randomness	Seed 42
Augmentation	Dataset-level augmentation disabled
Optimization	80-step warmup
Checkpointing	Save every 1,250 steps

Table 4: Shared settings for the cleaned pretraining and downstream training bundle.

findings. Each finding remained a candidate until review; incomplete evidence led to retention or review.

We compiled reviewed episode-, frame-, and field-level decisions into a content-addressed cleaning plan. The plan stores each decision with content-addressed identifiers so that the same inputs yield the same release. The materializer, the tool that applies this frozen plan, left the raw roots read-only. It emitted a LeRobot release, the directory format consumed by the training loader, with manifests, source identifiers, coverage records, and a READY marker.

We treated static intervals as candidates only. The policy excludes an interval only when canonical-action evidence marks it static, auxiliary channels show no motion, visual evidence supports the pause, and no semantic veto applies. Long gripper holds and other pauses can carry valid control meaning. The policy protects the pour phase in AgiBot task 410. We blocked AgiBot task 392 after finding a duplicate episode-metadata row. We blocked RoboMIND–UR because its canonical end-effector action required an authoritative reconstruction.

The resulting cleaned-v4r6 release is summarized in Table 5. A physical entry is one materialized dataset root; a logical slot is one sampler view over that root. DROID contributes two logical slots from one physical root. The materializer reuses media through symlinks. A machine-readable receipt records the aggregate definition, source identifiers, IDs, and blocked entries. The checkpoint has no runtime quality-sidecar sampler, an online quality filter; filtering happened during materialization.

Release field	Value
Materialized physical entries	38
Logical training slots	39
Episodes	146,433
Frames	80,212,225
Referenced media files	11,128
Blocked entries	2
Media mode	Symlink reuse

Table 5: Materialization summary for the cleaned-v4r6 pretraining release.

B.4 Focused LIBERO refinement screen

The focused screen evaluates tasks 0 and 1 in LIBERO-Spatial and LIBERO-10. Each strategy-task cell contains five episodes, and every cell produces five rollout videos and five action traces. The selected LIBERO checkpoint identity remains fixed across $d1_r0$, $d4_r0$, and $d4_r5$.

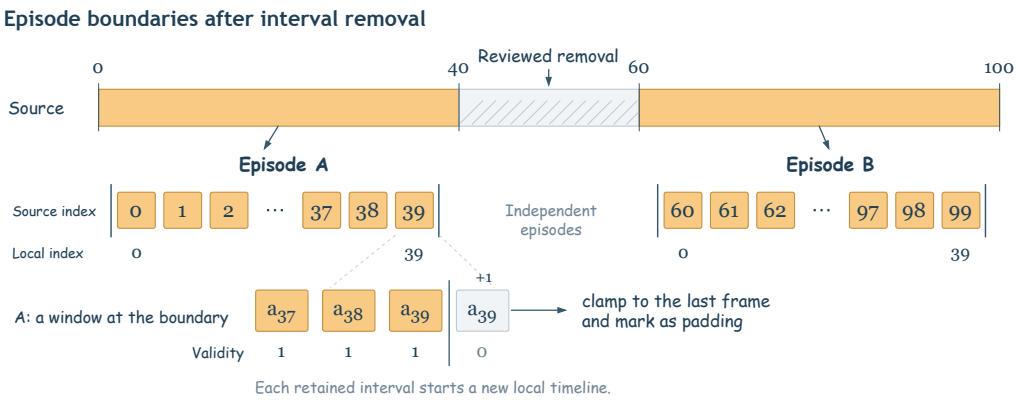


Figure 9: **Interval removal resets local episode boundaries.** Retained source intervals become independent episodes, and a window that crosses a boundary receives padding with an explicit validity mask.

Strategy	LIBERO-Spatial 10 episodes	LIBERO-10 10 episodes	Overall 20 episodes
$d1_r0$	100.0	90.0	95.0
$d4_r0$	100.0	100.0	100.0
$d4_r5$	100.0	100.0	100.0

Table 6: Focused matched-subset success by refinement strategy, reported in percent. The same fixed checkpoint supplies all 12 cells; each cell contains five episodes.

Both four-round strategies reach 100% on all four task cells.

B.5 Frozen-checkpoint wrist-camera screen

The supplementary screen uses the archived MONAD checkpoint and the public $\pi_{0.5}$ release (Physical Intelligence et al., 2025). MONAD uses a ten-action chunk with four proposal rounds and five remasking steps; $\pi_{0.5}$ retains its 50-action chunk and ten flow steps. Both methods execute ten actions before replanning. The LIBERO-Spatial block covers ten tasks and states 5–9 with a blackout at steps 40–59. The LIBERO-10 block covers ten tasks and states 0–2 with a blackout at steps 100–119. The machine-readable archive records checkpoint revisions and processor files.

The paired screen contains 50 LIBERO-Spatial episodes and 30 LIBERO-10 episodes per method. Each blackout episode reuses the task and initial state from its clean counterpart.

Suite	MONAD blackout	$\pi_{0.5}$ blackout	DID (pp)
LIBERO-Spatial	46/50 (92%)	43/50 (86%)	+8
LIBERO-10	28/30 (93.33%)	27/30 (90%)	+6.67

Table 7: Frozen-checkpoint wrist-camera blackout results. The table reports blackout success and the paired clean-to-blackout DID point estimate; clean-pair episode records are retained in the machine-readable archive.

The blackout-success point estimates are higher for MONAD on both suites, with an eight-point DID on LIBERO-Spatial and a 6.67-point DID on LIBERO-10.

B.6 Single-task wrist-camera blackout

We also evaluate a fixed-grid intervention on LIBERO-Spatial task 4. The archived MONAD checkpoint uses five initial states with seeds 1000–1004 and replaces the wrist red-green-blue (RGB)

Method	Wrist blackout
MONAD	5/5 (100%)
$\pi_{0.5}$	1/5 (20%)

Table 8: Fixed-grid wrist-camera blackout on LIBERO-Spatial task 4. The archived MONAD checkpoint and the public $\pi_{0.5}$ release use the same five initial states.

Round	Self context	Matched expert context
0	0.0643	0.0643
1	0.0700	0.0418
2	0.0843	0.0286
3	0.0924	0.0134

Table 9: Pending non-gripper action-coordinate MAE under context replay on 12 fixed CALVIN windows. Both conditions use the same inputs and selection masks.

image with zeros at steps 40–59 while keeping the main camera and proprioceptive state current. Both methods use the same five initial states and five episodes per condition.

MONAD completes all five initial states during the blackout, while $\pi_{0.5}$ completes one. The fixed-grid comparison gives an 80-point intervention-condition gap in MONAD's favor under this archived-checkpoint diagnostic.

B.7 Action-token context replay reduces pending error

The context-replay probe measures how committed action tokens affect later proposals. We use 12 fixed windows from the CALVIN training corpus, keep the inputs and selection masks fixed, and run four refinement rounds. Self context inserts MONAD's committed action tokens. Matched expert context inserts the expert tokens from the same window. We report pending non-gripper action-coordinate mean absolute error (MAE) after each round.

The conditions share the same round-0 proposal. At round 3, matched expert context lowers pending MAE from 0.0924 to 0.0134, an 85.4% reduction computed from unrounded values. Matched expert context improves all 12 windows.

Takeaway. Matched expert tokens give later MONAD proposals lower pending non-gripper action-coordinate MAE under the tested replay intervention.

B.8 Representation stress-test protocol

The canonical representation audit samples 512 real, contiguous CALVIN relative-action windows, with $H = 30$, $D = 7$, and normalized range $[-1, 1]$, using seed 20260901. A follow-up samples 8,192 windows under the same contract and adds four separately seeded 2,048-window samples. Errors below use normalized coordinates; translation, rotation, and gripper dimensions correspond to distinct physical quantities. The uniform quantizer has 256 boundaries, 255 reconstruction centers, and spacing $2/255$. Its random-substitution quantiles use 10,000 sampled mutations. FAST-based results use one single-token substitution per window.

Representation	MAE	Mean tokens	Invalid shape (%)	Valid RMSE median
Uniform quantizer	0.002256	210	0	n/a
FAST+	0.019959	57.68	82.23	0.0526
MolmoAct2-FAST, repaired	0.019959	90.15	77.93	0.0690

Table 10: Representation diagnostics. MAE measures unperturbed reconstruction; invalid shape and valid RMSE refer to single-token substitution. RMSE is conditional on shape-valid output: 91 FAST+ and 113 repaired MolmoAct2-FAST cases. Token counts do not imply matched model compute.

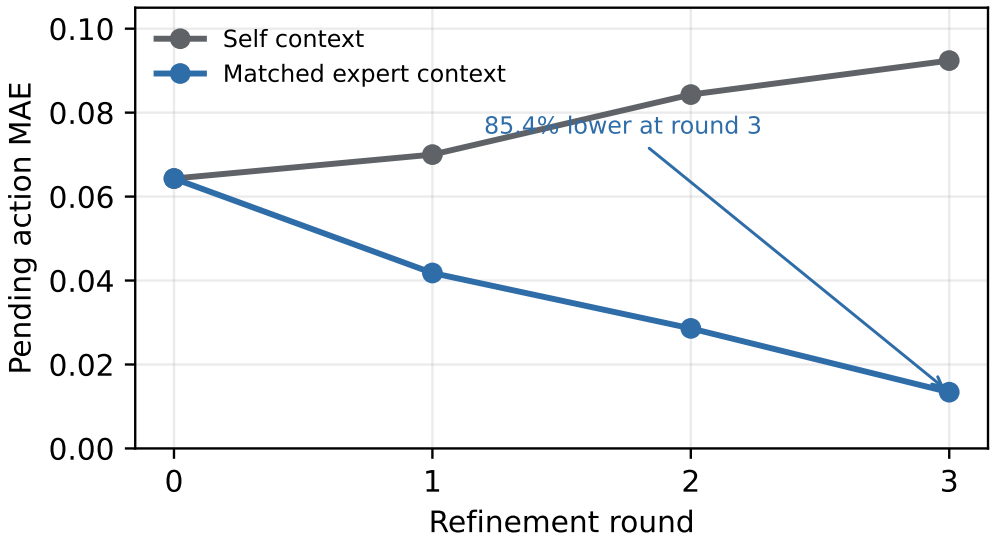


Figure 10: **Matched expert context lowers pending non-gripper action-coordinate MAE across refinement rounds.** The lines report the values in Table 9.

Run	Windows	FAST+ invalid (%)	FAST+ RMSE median	Molmo invalid (%)	Molmo RMSE median	Time support
Canonical	512	82.23	0.0526	77.93	0.0690	30/30
Follow-up	8,192	84.52	0.0590	79.24	0.1173	30/30
Four-seed range	$4 \times 2,048$	81.30–85.11	0.0552–0.0631	76.22–79.00	0.0966–0.1171	30/30

Table 11: Expanded representation stress test. Invalid shape is the fraction of one-token mutations that fail to recover 30×7 coefficients. RMSE reports the median over shape-valid mutations; the figure shows the corresponding fifth-to-ninety-fifth percentile range. Time support counts affected steps with a coordinate change above 0.01 normalized units.

For the uniform quantizer, the canonical substitution-induced absolute coordinate-error quantiles are 0.054902 and 1.380392 at the fifth and ninety-fifth percentiles. Their ratio is 25.1429. The 8,192-window follow-up gives 0.0549 and 1.4039, with a ratio of 25.5714. For the compressed representations, temporal support counts steps containing at least one coordinate change exceeding 0.01; the median is 30 of 30 steps for each encoding among shape-valid substitutions. The released Molmo tokenizer initially omitted 44 symbols across 42 windows. A fallback to existing raw-symbol vocabulary IDs repairs those omissions without adding vocabulary entries; all 512 unperturbed encodings then recover the expected 210 coefficients. "Lossless repair" denotes symbol and shape recovery with the tokenizer's original coefficient quantization. Across the expanded samples, compressed single-token edits preserve the high invalid-shape fractions and full-horizon median temporal support seen in the canonical run. These artificial mutations are not sampled from a policy's predictive distribution; they do not estimate natural invalid-output rates or task success.

B.9 Temporal-response and parameter diagnostics

The MONAD probe compares a per-time-step action encoder–decoder composition with one complete shared-backbone forward pass using the same interface. It performs central finite differences at 11 selected time/dimension coordinates and averages the per-probe non-local response ratios. The full path excludes iterative sampling, commit scheduling, and remasking. Computation uses float32, disabled TF32, and math-only SDPA. Repeated-forward L2 and maximum-absolute jitter are zero for every rerun. Table 13 reports the epsilon sweeps for all three MONAD checkpoints, and Figure 11 plots the expanded-sample spread and layer/probe response.

The pretraining checkpoint uses fixed language IDs without images or state; the two downstream configurations use a short language instruction with zero normalized state and no images. The π probes use forward differences and eager attention. These fixed contracts isolate the forward

Checkpoint	Action coords.	Action interface	Frozen BB layers	H	Valid D	Interface NLR	Full NLR	Temporal mixing
π_0 base	expert	separate expert	n/a	50	32	0	8.141%	Action expert
$\pi_{0.5}$ base	expert	separate expert	n/a	50	32	0	7.141%	Action expert
MONAD pretraining	canonical	trainable	none	50	20	0	1.442%	Shared backbone
MONAD downstream	absolute	frozen	0–1	25	10	0	1.467%	Shared backbone
MONAD downstream	relative	trainable	none	25	10	0	7.135%	Shared backbone

Table 12: Checkpoint-level mechanism diagnostics. The action-interface column records whether the MONAD action encoder and decoder remain trainable; the frozen-layer column lists backbone layer indices held fixed during downstream training. The two MONAD downstream rows jointly vary action-coordinate convention, action-interface freezing, and backbone-layer freezing, so the table presents them as configuration comparisons. MONAD rows use the legacy non-BARD implementation.

Checkpoint	$\epsilon = 0.01$	$\epsilon = 0.02$	$\epsilon = 0.04$	Interface NLR
MONAD pretraining	1.441995%	1.442460%	1.442852%	0
MONAD downstream (absolute)	1.464357%	1.467302%	1.469811%	0
MONAD downstream (relative)	7.132235%	7.135435%	7.134574%	0

Table 13: Full-path NLR remains stable across finite-difference steps. Each cell averages 11 probes under the checkpoint-specific input contract; interface-only NLR remains zero.

computation path, while the layer and coordinate probes show where temporal response enters each model.

The legacy MONAD checkpoint contains 37,282,324 action encoder/decoder parameters out of 8,354,355,220 total parameters (0.446262%), with no separate action Transformer. The audited π_0 and $\pi_{0.5}$ expert-plus-bridge proportions are 16.51% and 19.17%. Module accounting must remain explicit: the MONAD number counts interface parameters, whereas the expert numbers include both interfaces and generation networks. These fractions describe parameter allocation; they are not latency, memory, or task-quality measures and are not assigned to the BARD checkpoints.

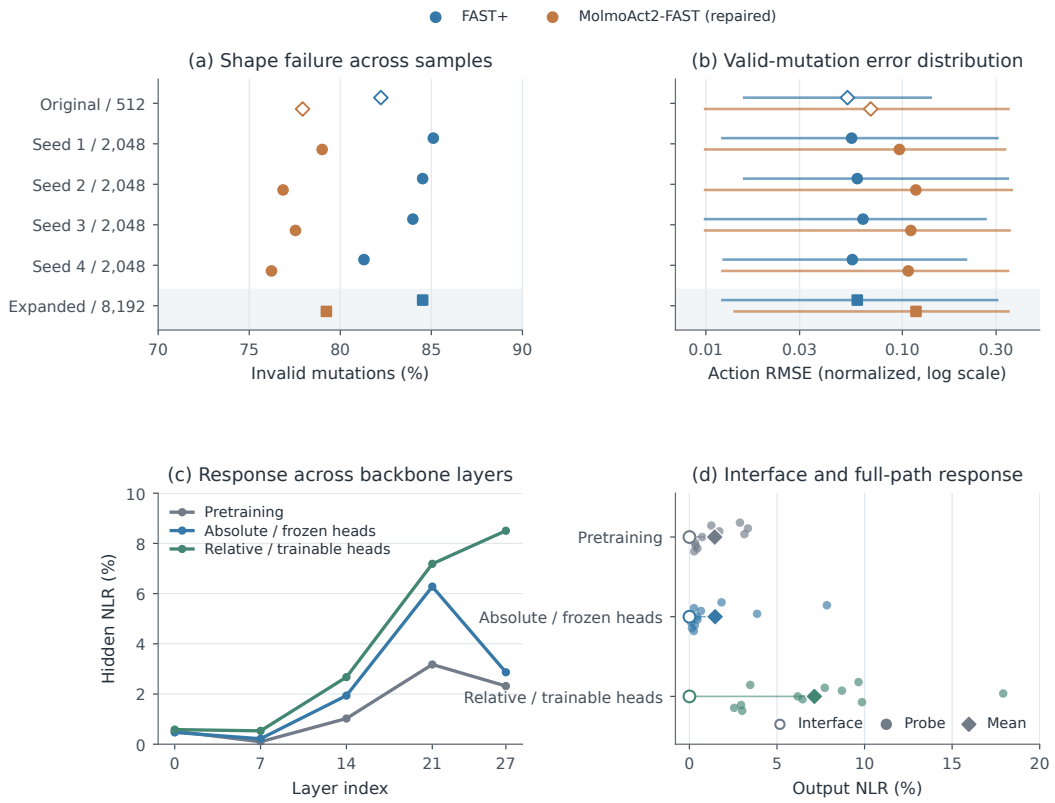


Figure 11: **Expanded probes preserve the diagnostic structure.** Panel (a) shows invalid-shape fractions across the canonical and expanded samples. Panel (b) shows median valid-mutation RMSE with fifth-to-ninety-fifth percentile ranges. Filled points represent four separately seeded 2,048-window samples and one 8,192-window sample; diamonds mark the canonical 512-window values. Panel (c) shows hidden-state far-time response at five sampled backbone layers, and panel (d) shows decoded-action far-time response for 11 fixed coordinate probes; open circles mark the zero interface NLR and diamonds mark means. All expanded compressed-token runs have median temporal support 30/30. These plots summarize artificial interface edits and checkpoint-level mechanism probes; they do not rank task performance.