PHYSIS-EMBED: UNIFIED VLA EMBEDDINGS FOR ROBOT IN-CONTEXT LEARNING AND EVOLUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Retrieving context for a robot requires matching both its task and its stage of execution. We introduce *Physis-Embed*, to our knowledge the first vision–language–action (VLA) embedding model to jointly encode all three modalities in both current-state queries and recorded future continuations. Predictive training aligns these endpoints for physical-experience retrieval; at inference, the model retrieves historical continuations as context. We construct multi-source training data and introduce *Physis-Embed-Bench*, on which our model achieves 47.33% clip-level and 81.77% episode-level R@1, surpassing action-free Gemini Embedding 2 by 22.09 and 33.47 percentage points. In our retrieval studies, removing native action degrades retrieval more than removing vision. This retrieval space supports two applications. *Embed2ICL* uses retrieved experience for training-free and training-based policy conditioning, increasing success from 47.0% to 56.3% in DROID-Sim and from 86.4% to 92.8% on the LIBERO difficult subset. Both interfaces also improve task progress in real-robot evaluations. *Embed2Evolve* uses execution feedback to grow task-local memory and refine its use around a frozen GPT-6 Astra policy. Across five ManiSkill tasks, it achieves 68% success, compared with 24% for GPT-6 Astra Direct and 33% for a harness independently evolved with Omni-Embed-Nemotron-3B under the same protocol. Relative to the latter, it uses approximately 79.5% fewer policy tokens per test rollout.

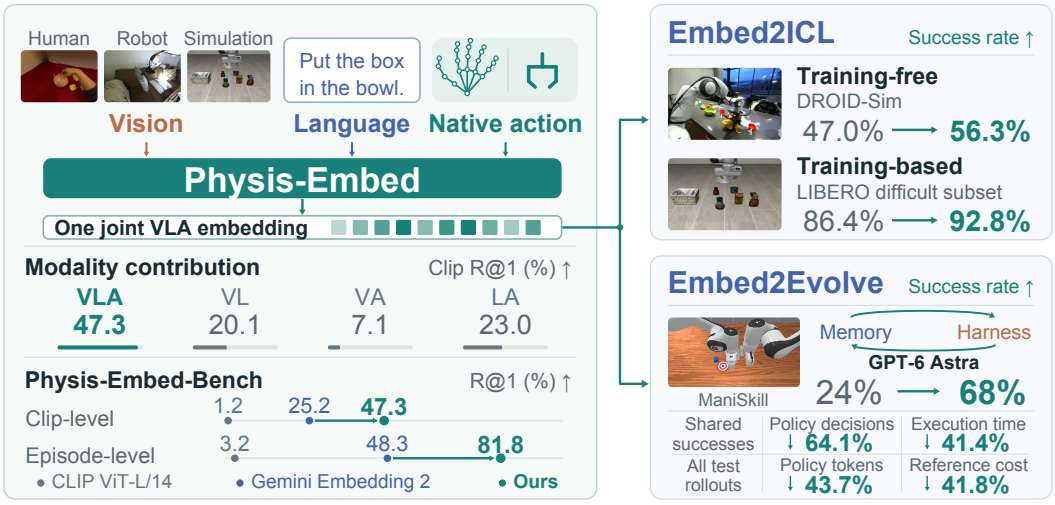

Figure 1: **One joint VLA embedding, two context applications.** Physis-Embed jointly encodes vision, language, and native action from human, robot, and simulated experience to retrieve recorded continuations. Action improves retrieval beyond vision and language alone. Embed2ICL uses retrieved demonstrations to condition robot policies. Embed2Evolve grows task-local memory and refines how a fixed GPT-6 Astra policy uses it, improving success and efficiency relative to Direct.

1 INTRODUCTION

Robot policies increasingly use experience supplied in context. GEN-1.5 (Team, 2026) and S1 (AI, 2026) demonstrate Robot In-Context Learning from sensorimotor or video examples. In our ex-

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.

periments, GPT-6 Astra (OpenAI, 2026) generates robot actions from observations and execution history. These capabilities make experience selection part of the control problem. Demonstration-conditioned world action models (WAMs) need useful existing examples; policies acting repeatedly must organize accumulated experience. Both raise a representation question: *how should physical experience be represented for task- and stage-sensitive retrieval?*

Task similarity alone is insufficient. Reaching toward a bowl and withdrawing after placement can share objects and an instruction yet require different continuations. Language specifies the goal, vision the interaction state, and native action the ongoing motion. The retrieval endpoints have different temporal roles: the query describes the present, whereas a candidate records what followed an earlier interaction. *Predictive retrieval* therefore requires sensitivity to task intent and execution stage: the goal is an appropriate recorded continuation, not simply a similar scene.

General-purpose embeddings align vision and language (Radford et al., 2021; Girdhar et al., 2023; Xu et al., 2025), while action-aware metric learning connects perception or language with behavior (Sung et al., 2017; Rana et al., 2023). We introduce ***Physis-Embed***, a unified vision–language–action (VLA) embedding model for physical-experience retrieval. To our knowledge, it is the first to jointly encode all three modalities in both current-state queries and recorded continuations. Each endpoint yields one vector, so native action informs retrieval itself. Source-specific projectors handle heterogeneous action formats.

We construct multi-source training data and introduce *Physis-Embed-Bench* for human, simulated, and real-robot experience. Nearby same-episode clips test whether retrieval locates the recorded continuation or merely its episode. Across the four sources, *Physis-Embed* achieves 47.33% clip-level and 81.77% episode-level R@1. Native action contributes beyond text and images: with backbone, data, and training budget matched on DROID, adding action raises clip-level R@1 from 59.8% to 78.7%. Removing action reduces retrieval accuracy more than removing vision in both matched retraining and inference-time query masking.

As shown in Figure 1, we use *Physis-Embed* in two applications: learning from existing demonstrations and evolving context through repeated interaction. **Embed2ICL** provides *embedding-guided robot in-context learning*. For DreamZero (Ye et al., 2026), we initialize the visual cache with reference videos without policy updates; for Fast-WAM (Yuan et al., 2026), we fine-tune the policy to condition on retrieved Context Tokens. Both interfaces improve simulation task success and real-robot task progress. Random references yield lower simulation success.

Embed2Evolve performs *embedding-guided context evolution*: repeated attempts grow task-local memory, and execution feedback refines how a frozen GPT-6 Astra policy accesses it. Across five ManiSkill tasks, the selected *context harness* raises success from 24% with GPT-6 Astra Direct to 68%. To test the retrieval model's role, we replace it with Omni-Embed-Nemotron-3B and evolve both conditions independently from the same initial memory under the same search protocol. Compared with this baseline, which reaches 33% success, Embed2Evolve uses approximately 79.5% fewer policy tokens per test rollout, including failures. The retrieval representation therefore matters for both memory selection and the final harness's success and token use.

We make four contributions. **(1) Physis-Embed**, a unified VLA embedding model for predictive physical-experience retrieval. **(2) Data and evaluation**: a multi-source training corpus, *Physis-Embed-Bench*, and controlled studies of the representation and its inputs. **(3) Embed2ICL**, connecting retrieved experience to training-free and training-based Robot In-Context Learning in simulation and on real robots. **(4) Embed2Evolve**, developing task-local memory and its use around a frozen policy, with a matched comparison of retrieval embedding models.

2 Related Work

Multimodal and action-aware embeddings. CLIP aligns images with language (Radford et al., 2021); Perception Encoder and InternVideo2 extend vision–language representation learning to video (Bolya et al., 2026; Wang et al., 2024). Unified vision–language embeddings support image and video retrieval in one space (Meng et al., 2025; Xu et al., 2025; Zhou et al., 2026b; Shanbhogue et al., 2026; Li et al., 2026b); ImageBind (Girdhar et al., 2023) includes additional sensory modalities. Deep Multimodal Embedding (Sung et al., 2017) and CLASP (Rana et al., 2023) align language or perception with action. VLA-R (Seong et al., 2025) embeds visual–language queries and action-

token candidates in a shared space, while RADAR (Gao et al., 2026) reuses VLA-policy features for retrieval. *Physis-Embed* jointly encodes vision, language, and native action at both retrieval endpoints: the current state and the recorded future clip.

Physical-experience retrieval. Action can enter the retrieval key or the content returned to the policy. ROSER (Rahman et al., 2026) searches proprioceptive sequences; RAEA (Zhu et al., 2024), REGENT (Sridhar et al., 2025a), and RICL (Sridhar et al., 2025b) return actions selected through action-free keys. Zeva (Chen et al., 2026a) indexes action–effect evidence, whereas RADAR (Gao et al., 2026) uses policy features. MemER (Sridhar et al., 2026) selects rollout keyframes; Retrieve-Reason-Act (Temiraliev et al., 2026) indexes visual manuals. *Physis-Embed* includes native action in both query and candidate keys, so motion informs the similarity comparison itself (Table 40).

Robot In-Context Learning and Evolution. GEN-1.5 (Team, 2026) conditions on a sensorimotor example; S1 (AI, 2026), Zero-WAM (Zhou et al., 2026a), and HOST (Chen et al., 2026b) use supplied video. R+X (Papagiannis et al., 2025) retrieves human-video clips and converts them into keypoint and hand-trajectory context. Embed2ICL uses *Physis-Embed* to select experience for training-free and training-based WAM conditioning. Reflexion (Shinn et al., 2023) stores verbal reflections in episodic memory, using feedback to improve later trials without updating model weights. In robotics, Zeva (Chen et al., 2026a) retrieves action–effect evidence across retries within an episode, resetting memory for each new episode. Embed2Evolve performs *in-context evolution*: it grows task-local memory across development episodes and refines retrieval, refresh, and history settings around a frozen policy. Evaluation uses the selected memory and context settings without further changes on fresh task initializations.

3 *Physis-Embed*

3.1 Problem Formulation

Physical AI experience combines language, vision, and source-native motion or control from human, simulated, or robotic interaction. Given a current VLA state, *Physis-Embed* retrieves the video and actions that followed an anchor in a recorded trajectory.

For sample i from source d_i, the query combines instruction ℓ_i, an interaction-region image $z_{i,t}$, and synchronized native action $a_{i,t}$. The paired target combines the same instruction with a four-second video clip V_i^+ and aligned actions A_i^+ recorded strictly after t:

$$x_i^q = (\ell_i, z_{i,t}, a_{i,t}, d_i), \qquad x_i^k = (\ell_i, V_i^+, A_i^+, d_i). \tag{1}$$

3.2 Action-Native Predictive VLA Embeddings

We train *Physis-Embed* for current-to-future retrieval using Cosmos3-Edge 4B (Agarwal et al., 2026) as the base model (Figure 2). Query and target share the Transformer, video tokenizer, and head, with separate adapters A_q, A_k. Source-specific projectors P_{d_i} map native actions into the backbone. Each endpoint jointly encodes language, vision, and action into one normalized vector. For retrieval, we rank candidates by the cosine similarity between the current-state query $q_i = f_\theta^q(x_i^q; P_{d_i})$ and the future-clip target $k_i = f_\theta^k(x_i^k; P_{d_i})$.

Retrieval-oriented training. We train with predictive alignment, multimodal grounding, and representation regularization. Structured parsing and deterministic rules supply weak labels.

Predictive retrieval alignment ($\mathcal{L}_{\text{align}}$) uses bidirectional InfoNCE with exact future-clip positives and task, object, action, or joint conflicts as negatives. Compatible, uncertain, or potentially duplicate non-positives are ignored. Figure 2 shows the validity mask. We optimize:

$$\mathcal{L} = \mathcal{L}_{\text{align}} + \mathcal{L}_{\text{ground}} + \mathcal{L}_{\text{reg}}. \tag{2}$$

Multimodal grounding and discrimination ($\mathcal{L}_{\text{ground}}$) combines task/object auxiliaries with margins against modality swaps and hard semantic conflicts, so that similarity responds to changes in the inputs. Representation regularization ($\mathcal{L}_{\text{reg}}$) centers query and target batches and discourages low-variance, low-rank embeddings. Appendix A.1–A.2 detail the architecture, native-action preprocessing, optimization schedule, and individual training losses.

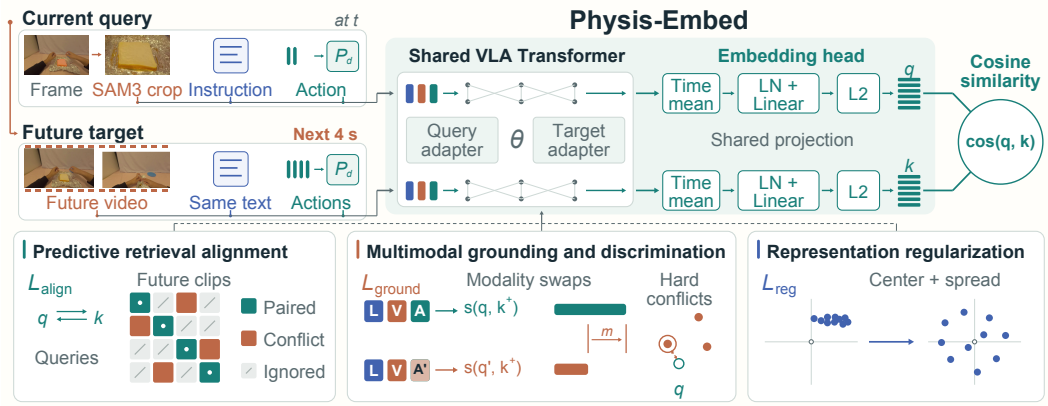

Figure 2: ***Physis-Embed* architecture and objectives.** Each query is paired with the next four seconds of recorded video and actions. Three objective groups train a joint VLA space.

3.3 Training Data and Retrieval Benchmark

Our training corpus contains 1,695,684 filtered current–future pairs from EgoDex (Hoque et al., 2026), DROID (Khazatsky et al., 2024), LIBERO (Liu et al., 2023), and RoboMIND–Franka-dual (Wu et al., 2024). EgoDex records tracked human motion; the other sources record robot controls. We sample sources equally and form within-source pairs (Figure 3; Appendix A.3).



Figure 3: **Multi-source training-data curation.** We split episodes before sampling anchors, localize and verify interaction crops, and pair accepted anchors with video and actions from the next four seconds. Query filtering uses only the instruction and current observations.

***Physis-Embed-Bench*.** All 4,000 held-out queries search one shared 8,000-clip gallery from the four source datasets. Each source contributes 1,000 queries from 800 episodes, 1,000 paired future clips, and 1,000 nearby same-episode clips. Queries follow Equation 1. *Clip-level retrieval* treats only the paired clip as positive; *episode-level retrieval* accepts all clips from the same source episode. Downstream experiments test whether retrieved context helps control.

We split episodes, sessions, and content identities before sampling anchors; task, object, and embodiment categories may recur. Clip-level metrics weight sources equally. Episode-level metrics aggregate within episodes, then equally over each source's 800 episodes and the four sources. We use cosine similarity without reranking (Appendix B.1).

4 Embedding Evaluation

4.1 Unified VLA Retrieval on *Physis-Embed-Bench*

Protocol and models. Eight action-free baselines search the shared gallery (Section 3.3). CLIP-style models retrieve video from text; unified VL models also use visual queries (Table 1). We use

Table 1: **Retrieval on *Physis-Embed-Bench*.** Four-source average recall and MRR (%) on the shared 8,000-clip gallery. Bold marks column-wise best results.

	Clip-level				Episode-level			
Model	R@1	R@5	R@10	MRR	R@1	R@5	R@10	MRR
CLIP-style text–video embeddings								
CLIP ViT-L/14	1.2	5.8	9.8	4.06	3.2	7.7	11.6	5.98
Perception Encoder B16	2.0	8.1	12.8	5.52	5.0	10.1	15.2	8.01
InternVideo2	1.6	7.3	11.4	4.82	4.1	9.1	13.2	6.94
Unified VL embeddings								
Qwen3-VL-Embedding-2B	23.3	51.6	57.3	36.20	45.1	55.9	61.0	50.71
VLM2Vec-V2	24.1	51.9	57.6	36.67	47.3	58.1	63.3	52.88
WeMM-Embedding-2B	24.1	52.2	58.2	36.78	45.1	57.3	63.2	51.08
Omni-Embed-Nemotron-3B	24.5	54.5	60.1	38.10	48.6	59.1	64.4	53.84
Gemini Embedding 2	25.2	52.3	57.5	37.68	48.3	58.2	64.7	53.69
Unified VLA embedding								
***Physis-Embed* (ours)**	**47.3**	**91.9**	**97.0**	**67.05**	**81.8**	**94.9**	**98.1**	**86.96**

each model's supported interface. The retraining studies in Section 4.2 use the same DROID data splits and training budgets for the compared model variants.

Main results. *Physis-Embed* achieves 47.33% clip-level and 81.77% episode-level R@1. The strongest baselines, Gemini Embedding 2 and Omni-Embed-Nemotron-3B, reach 25.24% and 48.59%, respectively. *Physis-Embed* improves on them by 22.09 and 33.18 percentage points.

Gains over unified VL baselines are largest on LIBERO and RoboMIND-FD (Appendix B.3). Episode retrieval does not resolve temporal localization: on EgoDex, episode/clip R@1 is 87.06/36.70%, versus 99.88/74.10% on DROID. Finding a relevant experience is easier than locating its paired continuation, the target of predictive retrieval.

Benchmark validity. Masking candidate text before fusion and recomputing embeddings lowers clip-level R@1 from 47.33% to 12.78%, confirming that the model uses language. Restricting candidates to the same instruction still gives 47.62% R@1, so the model also distinguishes continuations that share an instruction. Within same-instruction galleries, *Physis-Embed* achieves 38.60% and 40.10% R@1 on LIBERO and RoboMIND-FD, compared with random-ranking rates of 1.90% and 0.24%, respectively. High-k recall approaches saturation in the smaller EgoDex and DROID pools. Appendix B.4 gives per-source results, candidate counts, and random baselines.

Index-size scaling. Expanding the precomputed gallery from 1M to 15M entries increases query encoding plus retrieval latency from 160.36 to 195.17 ms on one NVIDIA A100 (Table 26). At 15M, Top-10 search alone takes 37.47 ms. These timings exclude time spent on SAM3 preprocessing, reference reranking, and downstream policy execution (Appendix B.6).

4.2 Controlled *Physis-Embed* Ablations

Retraining tests how well each input set supports retrieval; query masking tests a fixed model's reliance on its inputs. Retraining uses matched DROID splits and budgets, with clip-level metrics (%). Tables 2 and 7 report four-source averages for frozen-feature adaptation and final-model masking.

Table 2: **Retrieval adaptation.**

Setting	Clip R@1	Episode R@1
Frozen readout	0.65	1.17
Physis-Embed	**47.33**	**81.77**

Table 3: **Trainable scope.**

Trainable scope	R@1	MRR
Head only	13.8	24.00
Backbone + head	**78.7**	**88.00**

Retrieval adaptation and trainable scope. Across four sources, training raises clip R@1 from 0.65% with frozen features to 47.33% (Table 2). On matched DROID, head-only training reaches 13.8%, versus 78.7% with backbone adaptation (Table 3). Current-to-future matching benefits from adapting the backbone itself, rather than learning only a readout from its pretrained features.

Table 4: **Objective groups.**

Objective (Eq. 2)	**R@1**	**MRR**
$\mathcal{L}_{\text{align}}$	75.6	86.20
$\mathcal{L}_{\text{align}} + \mathcal{L}_{\text{reg}}$	77.0	87.10
$\mathcal{L}_{\text{align}} + \mathcal{L}_{\text{ground}} + \mathcal{L}_{\text{reg}}$	**78.7**	**88.00**

Table 5: **Visual input.**

Visual input	**R@1**	**MRR**
w/o vision (retrained)	76.8	86.69
Full frame	77.9	87.40
SAM3 object zoom	**78.7**	**88.00**

Objective groups. Predictive alignment reaches 75.6% R@1; adding $\mathcal{L}_{\text{reg}}$ gives 77.0%, and $\mathcal{L}_{\text{ground}}$ brings it to 78.7% (Table 4). R@10 remains 100.0% while top-rank ordering improves.

Visual input. SAM3 interaction crops reach 78.7% R@1, versus 77.9% for full frames and 76.8% without vision, and yield the highest MRR (Table 5).

Table 6: **Training-time modalities.**

Setting	**Input**	**Clip R@1**	Δ clip R@1
Full VLA	VLA	**78.7**	—
No action	VL	59.8	−18.9
No vision	LA	76.8	−1.9
No language	VA	20.2	−58.5

Table 7: **Inference-time modalities.**

Input	**Clip R@1**	**Episode R@1**	Δ clip R@1
VLA	**47.33**	**81.77**	—
VL	20.08	37.36	−27.25
LA	23.00	39.50	−24.33
VA	7.13	14.73	−40.20

Left: matched DROID retraining; right: four-source query masking with the checkpoint and gallery fixed. V/L/A denote vision/language/action; Δ is the percentage-point change from Full VLA.

Training-time modalities. Removing action during retraining lowers R@1 by 18.9 percentage points (Table 6). Native action contributes retrieval information beyond vision and language. Removing language has the largest effect; removing vision lowers R@1 by 1.9 points.

Inference-time modalities. With the final checkpoint and gallery fixed, withholding query action, image, or text lowers clip R@1 by 27.25, 24.33, or 40.20 percentage points (Table 7). Full VLA leads at both clip and episode levels; the trained model uses all three query modalities.

Removing action hurts retrieval more than removing vision in both studies, whether the model is retrained or its query inputs are masked (Appendix B.5). The DROID no-vision model reaches 76.8% R@1 after retraining, but masking vision in the full model gives 60.9%. Retraining permits adaptation to the remaining inputs; masking tests the learned fusion under an input shift. The small retraining gap therefore does not mean the full model ignores vision.

5 From Embeddings to Robot Context

5.1 Embed2ICL: Embedding-Guided Robot In-Context Learning

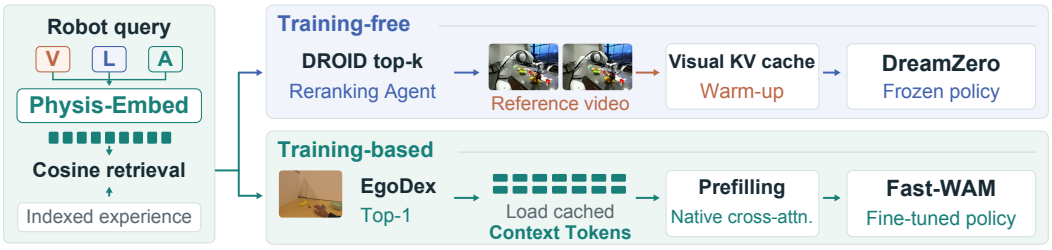

Figure 4: **Embed2ICL: one retriever, two policy interfaces.** Reranked DROID videos warm up DreamZero's visual cache; top-1 EgoDex matches supply Context Tokens for Fast-WAM's native cross-attention. Retrieval vectors select experience; videos or tokens condition the policy.

Training-free ICL with DreamZero. Embed2ICL initializes the visual cache of frozen DreamZero-DROID (Ye et al., 2026) with a retrieved video. Table 8 compares top-1 and reranked retrieval on three DROID-Sim tasks. Both retrievers use the same GPT-5.6 sol Reranking Agent to select task-relevant four-second references with completion evidence. All six conditions use 100

trials per task, including the random-reference control. Embed2ICL uses one fixed reference video per task across all evaluation trials (Appendix C.1).

Table 8: **Training-free DreamZero-DROID in DROID-Sim.** Success rates (%); parentheses show percentage-point changes from no warm-up. Reranked conditions share the same agent.

Reference condition	Cube → Bowl	Can → Mug	Banana → Bin	Average
No warm-up	61	13	67	47.0
Random reference	50 (−11)	7 (−6)	51 (−16)	36.0 (−11.0)
Omni-Embed-Nemotron-3B				
Top-1	57 (−4)	5 (−8)	68 (+1)	43.3 (−3.7)
+ reranking	62 (+1)	19 (+6)	72 (+5)	51.0 (+4.0)
Physis-Embed Top-1	63 (+2)	16 (+3)	72 (+5)	50.3 (+3.3)
Embed2ICL	**65** (+4)	**24** (+11)	**80** (+13)	**56.3** (+9.3)

Embed2ICL performs best on all three tasks, averaging 56.3% success versus 47.0% without warm-up and 36.0% with random references. Reranking improves both retrievers. *Physis-Embed* exceeds Omni-Embed-Nemotron-3B before reranking (50.3% versus 43.3%) and with the same reranker (56.3% versus 51.0%; Appendix C.2). Random references underperform no warm-up: adding a demonstration is not sufficient. Selected references preserve placement relations despite changes in objects or receptacles, suggesting that useful context need not reproduce the target scene exactly.

On YAM, Embed2ICL raises Banana-to-Bowl mean task progress from 21.67 to 40.00 (Table 11). Both conditions use the same DreamZero-AgiBot policy adapted to YAM, with no further policy updates for reference warm-up (Appendix C.3).

Training-based ICL with Fast-WAM. For Fast-WAM (Yuan et al., 2026), *Physis-Embed* retrieves the top-ranked EgoDex demonstration. Cached Context Tokens from its multimodal temporal states condition the policy (Figure 4). We train the policy to use this context while keeping the offline embedding encoder fixed (Appendix C.4).

LIBERO difficult subset. Before ICL evaluation, an independent pilot selected five tasks on which the official release succeeded in at most 46 of 50 trials. In 50 trials per task and condition, Embed2ICL improves every task and raises average success from 86.4% to 92.8% (Table 9).

Table 9: **Training-based ICL on the LIBERO difficult subset.** Successes out of 50 per task; last row: average success (%). Parentheses give percentage-point gains over the official release.

Suite / ID	Abbreviated task	Official release	Embed2ICL
Spatial / S4	Black bowl → plate	46	**49** (+6)
Goal / G3	Bowl → top drawer	42	**50** (+16)
Goal / G9	Wine bottle → rack	45	**47** (+4)
LIBERO-10 / L4	Two mugs → plates	43	**44** (+2)
LIBERO-10 / L6	Mug–pudding arrangement	40	**42** (+4)
Five-task average (%)		86.4	**92.8** (+6.4)

Table 10: **LIBERO controls.** Five-task average success (%); 50 trials per task and condition.

Official release (no ICL)	Random Context Tokens	**Embed2ICL (retrieved)**
86.4	88.0	**92.8**

With the same policy checkpoint, retrieved Context Tokens outperform tokens from random references by 4.8 percentage points (Table 10). After the same 21,700 training updates as Embed2ICL, the no-ICL continuation still achieves 86.4% (Appendix C.5), versus 92.8% with retrieved context. Figure 13 illustrates a retrieved human reference and robot execution.

Real-robot evaluation. Both interfaces improve task progress on real robots (Table 11). With partial-credit scoring inspired by T-Rex (Niu et al., 2026), Fast-WAM gains more on Split Cup than

on Pick Fruit. Split Cup requires one hand to stabilize a stack while the other separates a cup; Pick Fruit scores each arm's grasp and placement into the center basket.

Table 11: **Embed2ICL on real robots.** Mean task progress on task-specific 0–100 scales over 12 rollouts per task and condition. Parentheses show gains computed before rounding.

Task	No ICL	Embed2ICL
Training-free: DreamZero		
Banana → Bowl	21.67	**40.00** (+18.33)
Training-based: Fast-WAM		
Pick Fruit	75.00	**79.17** (+4.17)
Split Cup	66.67	**87.50** (+20.83)

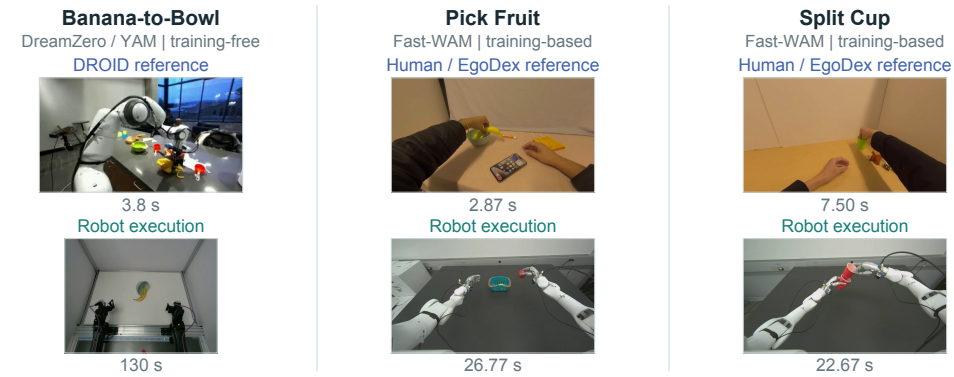

Figure 5: **Retrieved references and real-robot execution.** Each task pairs a retrieved reference (top) with real-robot execution (bottom). References come from DROID for DreamZero/YAM and EgoDex for Fast-WAM. Timestamps are relative to each clip's start.

5.2 Embed2Evolve: Embedding-Guided Context Evolution

Embed2Evolve builds task-local memory and refines how it is used through repeated attempts. This memory and context recipe form a *harness*; GPT-6 Astra and *Physis-Embed* remain fixed (Figure 6).

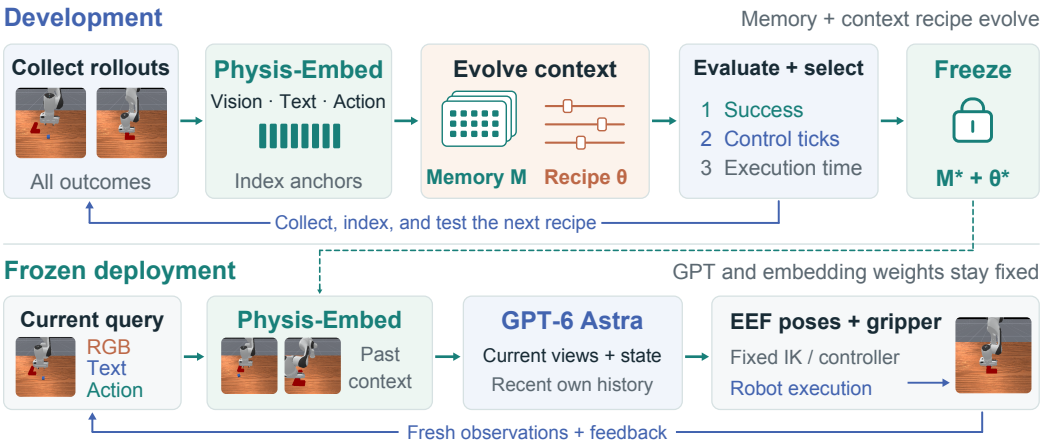

Figure 6: **Embed2Evolve.** Development grows memory and optimizes its use; testing fixes both. GPT-6 Astra proposes actions for a fixed controller. The model comparison replaces Physis-Embed in both phases. Images show the PullCubeTool task.

Memory and context search. Completed attempts, successful or failed, add indexed continuations of up to four seconds to memory. Deterministic search varies memory ranking and count, refresh cadence, and retained history. Development prioritizes success, using successful-execution efficiency to break ties. The selected memory and recipe are then fixed for testing on fresh task initializations (Figure 14; Appendix D.2). Along PullCubeTool's R4→R5→R8 development path,

memory grows from 938 to 1,667 anchors while retained history shrinks from eight to four pairs. A larger store of experience need not produce a longer context for the policy.

Context-guided control. The current VLA query retrieves historical RGB, actions, and feedback. Context guides GPT-6 Astra's commands within an unchanged action interface (Appendix D.1).

Evaluation. Five development-selected ManiSkill tasks (Tao et al., 2025) use 20 fresh matched initializations per condition. GPT-6 Astra Direct (Direct) uses within-episode history; GPT-6 Astra + random history adds a fixed random same-task RGB clip. These conditions use the same policy, current observations, controller, and per-rollout control budgets.

Embedding-model comparison. We replace only *Physis-Embed* with Omni-Embed-Nemotron-3B (Xu et al., 2025) and independently repeat evolution. Both conditions share the policy, prompt, initial memory, seeds, proposal and stopping rules, and eight-round cap per task. Both retrieve actions and feedback, but their resulting memory banks and context recipes may differ.

Table 12: **Embed2Evolve in ManiSkill simulation.** Success (%) over 20 trials per task and condition; mean policy tokens per test rollout, including failures.

Task	GPT-6 Astra Direct	GPT-6 Astra + random history	Evolve + Omni-Embed-Nemotron-3B	Embed2Evolve
Poke Cube	45	15	45	**80**
Stack Cube	10	20	20	**60**
Lift Peg Upright	30	40	55	**65**
Pull Cube with Tool	10	10	20	**50**
Two-Robot Stack Cube	25	15	25	**85**
Average (%)	24.0	20.0	33.0	**68.0**
Tokens/rollout (M)	0.921	–	≈2.531	**0.518**

The Omni-Embed-Nemotron-3B harness raises success from Direct's 24% to 33%, while mean policy tokens rise from 0.921M to approximately 2.531M per test rollout (Table 12). Embed2Evolve reaches 68% at 0.518M tokens, improving success on every task while using approximately 79.5% fewer tokens than this independently evolved baseline. Under matched evolution settings, the embedding model affects the resulting harness's success and token use.

Tool use and dual-arm coordination. Within the same action interface, retrieved continuations may narrow GPT-6 Astra's effective decision space through task- and stage-specific examples of tool contact, motion direction, and arm coordination. Against Omni-Embed-Nemotron-3B, success rises from 20% to 50% on Pull Cube with Tool and from 25% to 85% on Two-Robot Stack Cube.

Test-time efficiency. Compared with Direct, Embed2Evolve uses 43.7% fewer tokens across all test rollouts and reduces reference API cost by 41.8%, from \$9.9641 to \$5.7980 (Appendix D.6). Most savings come from inputs. On the 19 shared-success pairs, policy decisions fall by 64.1% and client execution time, including API waits, falls by 41.4%. Both reductions average task-wise percentages equally across the five tasks. Averaging the task-specific Direct-to-Embed2Evolve time ratios gives a 1.90× mean speedup (Appendix D.5). These results are consistent with context focusing the policy on relevant next actions. Fewer policy calls do not necessarily reduce the number of physical control ticks (Table 38).

6 Conclusion

Physis-Embed retrieves recorded continuations by jointly encoding task intent, visual state, and native action. Action improves retrieval beyond language and vision, supporting retrieved-context conditioning in Embed2ICL and memory-based context evolution in Embed2Evolve. Under matched evolution settings, Embed2Evolve achieves higher success with fewer policy tokens than its Omni-Embed-Nemotron-3B counterpart. Experience representation affects both the context retrieved for a decision and the harness developed around a frozen policy.

ETHICS STATEMENT

Human and robot recordings may contain sensitive information; their use and redistribution must respect dataset licenses, access conditions, and privacy requirements. Retrieved experience can be irrelevant or unsuccessful and may lead to unsafe robot actions. Deployment therefore requires task-specific safety validation, operating limits, and human oversight.

REPRODUCIBILITY STATEMENT

Appendix A.1 specifies the embedding readout, native action normalization, and optimization schedule; Appendix A.2 gives the training objectives and semantic-conflict weak-label rules, and Appendix A.3 describes data curation and split isolation. Appendix B.1 defines benchmark composition, relevance, and metric aggregation. The Embed2ICL protocols cover DreamZero reference warm-up (Appendix C.1), Context Token generation and Fast-WAM training (Appendix C.4), and real-robot evaluation and progress scoring (Appendix C.3; Appendix C.4). Embed2Evolve's memory scoring and failure handling appear in Appendix D.1; Appendix D.2 specifies candidate generation, configuration selection, and stopping. Appendix D.4 describes matched initial conditions and independent context evolution with the two embedding models. The held-out test protocol and paired-success efficiency measures are in Appendix D.5, while Appendix D.6 reports all-test-rollout token accounting and the fixed reference-cost calculation.

AI USE STATEMENT

We used generative AI in the following ways. First, Qwen3-VL-8B-Instruct screens candidate interaction crops during dataset construction, retaining crops that satisfy the visual and semantic quality criteria in Appendix A.3. The same model also parses task instructions into operations and object roles using non-sampling decoding. Deterministic rules use the resulting task and object signatures to construct semantic-conflict weak supervision (Appendix A.2); these are not human-annotated labels.

Second, the training-free Robot In-Context Learning system uses a GPT-5.6 sol Reranking Agent to select an existing segment from retrieved candidates in Embed2ICL and its reranking comparisons. The agent selects reference content; it does not generate demonstrations or robot actions.

Third, Embed2Evolve uses GPT-6 Astra as an action-generating policy, proposing end-effector and gripper commands from current observations and retrieved historical context. A fixed controller executes them. Development changes external memory and context settings.

Fourth, LLMs assisted with manuscript editing, including sentence rephrasing, grammar correction, and readability improvements. All research ideas, methods, experiments, and analyses are the authors' own, and the authors take full responsibility for the content of this manuscript.

REFERENCES

Niket Agarwal, Arslan Ali, Jon Allen, Martin Antolini, Adeline Aubame, Alisson Azzolini, Junjie Bai, Maciej Bala, Yogesh Balaji, Josh Bapst, et al. Cosmos 3: Omnimodal world models for physical ai. *arXiv preprint arXiv:2606.02800*, 2026.

Skild AI. Introducing s1: In-context learning for robotics. August 2026. URL https://skild.ai/blogs/s1.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Daniel Bolya, Po-Yao Huang, Peize Sun, Jang Hyun Cho, Andrea Madotto, Chen Wei, Tengyu Ma, Jiale Zhi, Jathushan Rajasegaran, Hanoona Bangalath, et al. Perception encoder: The best visual embeddings are not at the output of the network. *Advances in Neural Information Processing Systems*, 38:60884–60937, 2026.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris Coll-Vinent, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. Sam 3: Segment anything with concepts. In *International conference on learning representations*, volume 2026, pp. 138846–138923, 2026.

Fu Chen, Xin Ding, Bingjia Huang, Xiangyu Li, Mingju Wang, Jiawei He, Kun Li, Wei Sun, Yunxin Liu, Hao Wu, et al. Zeva: In-context causal learning for generalizable embodied manipulation. *arXiv preprint arXiv:2608.30880*, 2026a.

Guangyan Chen, Meiling Wang, Te Cui, Zichen Zhou, Qi Shao, Shalfun Li, Hang Su, Roy Gan, Hao Wang, Mengyin Fu, et al. Robots acquire manipulation skills in seconds from a single human video. *arXiv preprint arXiv:2607.20033*, 2026b.

Jensen Gao, Dorsa Sadigh, Sandy Huang, and Dhruv Shah. Grounding robot generalization in training data via retrieval-augmented vlms. *arXiv preprint arXiv:2603.11426*, 2026.

Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind one embedding space to bind them all. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15180–15190. IEEE, 2023.

Ryan Hoque, Peide Huang, David Yoon, Jian Zhang, et al. Egodex: Learning dexterous manipulation from large-scale egocentric video. In *International Conference on Learning Representations*, volume 2026, pp. 4218–4237, 2026.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, et al. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026a.

Mingxin Li, Yanzhao Zhang, Dingkun Long, Keqin Chen, Sibo Song, Shuai Bai, Zhibo Yang, Pengjun Xie, An Yang, Dayiheng Liu, et al. Qwen3-vl-embedding and qwen3-vl-reranker: A unified framework for state-of-the-art multimodal retrieval and ranking. *arXiv preprint arXiv:2601.04720*, 2026b.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Rui Meng, Ziyan Jiang, Ye Liu, Mingyi Su, Xinyi Yang, Yuepeng Fu, Can Qin, Zeyuan Chen, Ran Xu, Caiming Xiong, et al. Vlm2vec-v2: Advancing multimodal embedding for videos, images, and visual documents. *arXiv preprint arXiv:2507.04590*, 2025.

Dantong Niu, Zhuoyang Liu, Zekai Wang, Boning Shao, Zhao-Heng Yin, Anirudh Pai, Yuvan Sharma, Stefano Saravalle, Ruijie Zheng, Jing Wang, et al. T-rex: Tactile-reactive dexterous manipulation. *arXiv preprint arXiv:2606.17055*, 2026.

OpenAI. Gpt-6 astra: A new generation of intelligence. https://openai.com/index/gpt-6-astra/, 2026.

Georgios Papagiannis, Norman Di Palo, Pietro Vitiello, and Edward Johns. R+ x: Retrieval and execution from everyday human videos. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8284–8290. IEEE, 2025.

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

Zillur Rahman, Eddison Pham, Alejandro Daniel Noel, and Cristian Meo. Roser: Few-shot robotic sequence retrieval for scalable robot learning. 2026.

Krishan Rana, Andrew Melnik, and Niko Sünderhauf. Contrastive language, action, and state pre-training for robot learning. *arXiv preprint arXiv:2304.10782*, 2023.

Hyunki Seong, Seongwoo Moon, Hojin Ahn, Jehun Kang, and David Hyunchul Shim. Vla-r: Vision-language action retrieval toward open-world end-to-end autonomous driving. *arXiv preprint arXiv:2511.12405*, 2025.

Madhuri Shanbhogue, Zhe Li, Shanfeng Zhang, Gustavo Hernández Ábrego, Shih-Cheng Huang, Aashi Jain, Daniel Salz, Sonam Goenka, Chaitra Hegde, Ji Ma, et al. Gemini embedding 2: A native multimodal embedding model from gemini. *arXiv preprint arXiv:2605.27295*, 2026.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

Ajay Sridhar, Jennifer Pan, Satvik Sharma, and Chelsea Finn. Scaling up memory for robotic control via experience retrieval. In *International Conference on Learning Representations*, volume 2026, pp. 97142–97166, 2026.

Kaustubh Sridhar, Souradeep Dutta, Dinesh Jayaraman, and Insup Lee. Regent: A retrieval-augmented generalist agent that can act in-context in new environments. In *International Conference on Learning Representations*, volume 2025, pp. 39385–39414, 2025a.

Kaustubh Sridhar, Souradeep Dutta, Dinesh Jayaraman, and Insup Lee. Ricl: Adding in-context adaptability to pre-trained vision-language-action models. *arXiv preprint arXiv:2508.02062*, 2025b.

Jaeyong Sung, Ian Lenz, and Ashutosh Saxena. Deep multimodal embedding: Manipulating novel objects with point-clouds, language and trajectories. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2794–2801. IEEE, 2017.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-Kai Chan, et al. Maniskill3: Gpu parallelized robot simulation and rendering for generalizable embodied ai. In *7th Robot Learning Workshop: Towards Robots with Human-Level Abilities*, 2025.

Generalist Team. Gen-1.5: Embodied foundation models are one-shot learners. *Generalist AI Blog*, 2026. https://generalistai.com/blog/gen-1.5.

Izat Temiraliev, Diji Yang, and Yi Zhang. Retrieval-augmented robots via retrieve-reason-act. *arXiv preprint arXiv:2603.02688*, 2026.

Yi Wang, Kunchang Li, Xinhao Li, Jiashuo Yu, Yinan He, Guo Chen, Baoqi Pei, Rongkun Zheng, Zun Wang, Yansong Shi, et al. Internvideo2: Scaling foundation models for multimodal video understanding. In *European conference on computer vision*, pp. 396–416. Springer, 2024.

Kun Wu, Chengkai Hou, Jiaming Liu, Zhengping Che, Xiaozhu Ju, Zhuqin Yang, Meng Li, Yinuo Zhao, Zhiyuan Xu, Guang Yang, et al. Robomind: Benchmark on multi-embodiment intelligence normative data for robot manipulation. *arXiv preprint arXiv:2412.13877*, 2024.

Mengyao Xu, Wenfei Zhou, Yauhen Babakhin, Gabriel Moreira, Ronay Ak, Radek Osmulski, Bo Liu, Even Oldridge, and Benedikt Schifferer. Omni-embed-nemotron: A unified multimodal retrieval model for text, image, audio, and video. *arXiv preprint arXiv:2510.03458*, 2025.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-wam: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026.

Jiaming Zhou, Qihang Zhang, Gangwei Xu, Cunxin Fan, Yujie Zhao, Ruilin Wang, Yiming Luo, Shuai Yang, Xing Zhu, Yujun Shen, et al. Zero-wam: In-context world-action modeling from human videos for open-ended task generalization. *arXiv preprint arXiv:2608.26103*, 2026a.

Junjie Zhou, Ke Mei, Lei Li, Tianyi Wang, Fengyun Rao, and Jing Lyu. Wemm-embedding: Wechat multi-modal embedding technical report. *arXiv preprint arXiv:2608.24053*, 2026b.

Yichen Zhu, Zhicai Ou, Xiaofeng Mou, and Jian Tang. Retrieval-augmented embodied agents. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17985–17995. IEEE, 2024.

APPENDIX

A *Physis-Embed* METHOD DETAILS

A.1 ARCHITECTURE AND OPTIMIZATION

Paired inputs. Queries use instructions, synchronized current actions, and SAM3 (Carion et al., 2026) interaction crops. Targets pair instructions with 16 frames and native-rate actions from strictly future four-second windows. We favor less-occluded third-person views; EgoDex is egocentric.

Shared backbone and endpoint adapters. *Physis-Embed* uses Cosmos3-Edge 4B, with 28 Transformer layers and hidden width 2,048. The Transformer, shared retrieval head, endpoint adapters, and temperature are initialized from its pretrained checkpoint. Query and target share the Transformer, video tokenizer, and embedding projection head. Separate bias-free residual linear adapters specialize the current-state and future-clip endpoints. Each source retains its native action projector, trained jointly with the Transformer. For endpoint $r \in \{q, k\}$, let $H^r \in \mathbb{R}^{T_r \times 2048}$ denote the multimodally fused hidden states at native action-token positions, after the generation RMSNorm following layer 28. These states jointly encode the endpoint's language, vision, and action. Let g_r denote the endpoint readout, comprising the shared LayerNorm and projection $W \in \mathbb{R}^{1024\times 2048}$ together with the endpoint-specific residual adapter. Temporal pooling and normalization give

$$\bar{h}^r = \frac{1}{T_r}\sum_{t=1}^{T_r} H_t^r, \qquad z^r = \frac{g_r(\bar{h}^r)}{\|g_r(\bar{h}^r)\|_2}. \tag{3}$$

The shared projection is bias-free. Benchmark retrieval uses cosine similarity; only training-free Embed2ICL reranks. Temporal pooling of the target states, shared projection, and per-token normalization yield Embed2ICL's cached Context Tokens (Appendix C.4).

Native actions. Table 13 specifies the native action interfaces. For each source, we compute per-coordinate quantiles q_{01} and q_{99} using its training partition and normalize actions as

$$\hat{a} = 2\frac{a - q_{01}}{\max(q_{99} - q_{01}, 10^{-8})} - 1. \tag{4}$$

Values remain unclipped, and zero-padding to 64 dimensions standardizes tensor shape. The source identifier selects the native projector; future actions retain their native sampling rate.

Table 13: **Native action interfaces.** Dimensions and rates describe native actions before normalization and padding. Domain IDs route actions to their Cosmos3-Edge projectors.

Source	Domain ID	Action dimensions	Rate (Hz)
EgoDex	3	57	30
DROID	8	10	15
LIBERO	5	10	10
RoboMIND–Franka-dual	12	20	10

Optimization. We use AdamW with $\beta = (0.9, 0.95)$ and gradient clipping at 1.0. The Transformer learning rate is 3×10^{-7}, and the projection head and endpoint adapters use 10^{-4}. Training runs for 26,275 optimizer steps, with 1,314 steps of linear warm-up followed by cosine decay, global batch size 512, and equal sampling of EgoDex, DROID, LIBERO, and RoboMIND–Franka-dual (1:1:1:1). The optimizer and scheduler are reinitialized for retrieval training. Both parameter groups use weight decay 0.05; the video VAE is frozen. We use the final-step checkpoint (26,275).

A.2 *Physis-Embed* OBJECTIVE

The objective (Eq. 2) combines predictive retrieval alignment, multimodal grounding and discrimination, and representation regularization. Table 14 lists term coefficients and ranking margins.

Table 14: ***Physis-Embed* training objective.** Loss coefficients and ranking margins for each term, organized into the three objective groups in Eq. 2.

Objective	Constraint or positive relation	Weight	Margin
Predictive retrieval alignment			
Clip-level retrieval	Exact future-clip identity; bidirectional InfoNCE	1.00	–
Multimodal grounding and discrimination			
Text auxiliary	Text-only query; matched task	0.15	–
Image auxiliary	Image-only query; matched interaction object	0.20	–
Action auxiliary	Disabled; future identity uses multimodal context	0.00	–
Text swap	Replacing query text must reduce similarity	0.25	0.08
Image swap	Replacing the query zoom must reduce similarity	0.35	0.10
Action swap	Replacing current action must reduce similarity	0.20	0.06
Hard task	Same object roles; operation/relation/direction conflict	0.25	0.08
Hard object	Same task relation; interaction-object category conflict	0.35	0.10
Hard action	Same task/objects; action-primitive/phase conflict	0.20	0.06
Fused dominance	Fused query must outperform every unimodal query	0.20	0.02
Representation regularization			
Isotropy	Variance, participation-ratio rank, and centering	0.05	–
Query center	Direct query-centering penalty	0.10	–
Target center	Direct target-centering penalty	0.10	–

Positive, negative, and ignored pairs. Positive pairs share the exact future-clip ID. For query q_i, k_i is the paired positive (k^+ in Figure 2). Each pair links a current state to its recorded future clip within the same source. Let $s(q, k)$ be cosine similarity and $\mathcal{N}(i)$ contain only candidates with a different clip ID and an explicit semantic conflict. The query-to-target term is

$$\mathcal{L}_{q\to k} = -\frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} \log \frac{\exp(s(q_i, k_i)/\tau)}{\exp(s(q_i, k_i)/\tau) + \sum_{j\in\mathcal{N}(i)} \exp(s(q_i, k_j)/\tau)}. \tag{5}$$

We independently build the reverse validity mask and average the directions: $\mathcal{L}_{\text{align}} = \frac{1}{2}(\mathcal{L}_{q\to k} + \mathcal{L}_{k\to q})$. The global pool contains 512 entries before masking; each denominator retains only the positive and valid negatives. We optimize the inherited Cosmos3-Edge temperature τ within $[0.01, 0.2]$.

Semantic conflict rules. We generate weak labels through structured parsing and deterministic rules. Qwen3-VL-8B-Instruct (Bai et al., 2025) parses task instructions into operation names and object roles using non-sampling decoding. From this output, we construct signatures for task relations, object identities, and object categories. The resulting labels are rule-driven weak supervision, not human annotations. Before similarity-based mining, we determine eligibility for $\mathcal{N}(i)$ using four pairwise conflicts: *Task conflict* preserves object roles but changes the operation, relation, or direction. *Object conflict* preserves the task relation but changes a valid interaction-object category. *Action conflict* shares task and objects but differs in the current-action primitive or future-action phase. *Joint conflict* changes both task and objects.

For same-episode action negatives, future windows with temporal intersection-over-union greater than 0.25 are treated as near-duplicates and excluded. Cross-episode action negatives must additionally differ in future gripper phase. These checks filter the action-conflict class; semantically compatible, uncertain, or potentially duplicate candidates are masked out of the negative set. A different clip ID alone is therefore insufficient to make a pair negative.

Current-action primitives are obtained by scale-standardizing the domain-normalized action and discretizing each coordinate at thresholds -0.5 and $+0.5$. Future-action phase combines the dominant motion direction, the timing of peak motion, and changes in gripper state. To compare future actions, we sample 16 time points and concatenate the per-coordinate means, standard deviations, and first-to-last differences. We compare these descriptors using scale-normalized RMS distances.

Auxiliary alignment. Text-only and image-only auxiliary queries align with targets sharing task and interaction-object identity, respectively. We disable standalone action-only alignment to exact future clips: a single current action does not uniquely determine a future trajectory. Action remains an input to fused retrieval alignment, with action swaps discouraging the model from ignoring it.

Modality comparisons. Each swapped query replaces exactly one modality while keeping its paired target fixed. Text donors share object identity but differ in task relation; image donors share the task relation but differ in valid object category. Action donors share the action domain, task, and object identity, but differ in both current-action primitive and future-action phase. The standardized RMS distances between their current actions and between their future descriptors must each be at least 0.75. These donor conditions are stricter than the action-conflict eligibility rules.

Let q' encode a valid swapped query. We use the cosine hinge loss

$$\ell_{\text{swap}}(q, q', k^+) = \max\{0,\, m - s(q, k^+) + s(q', k^+)\}\,, \tag{6}$$

with $m = 0.08$ (text), 0.10 (image), and 0.06 (action). We skip only the corresponding swap loss when no valid donor exists. A 0.02 margin favors the fused query over each unimodal query.

Hard-negative constraints. The contrastive denominator uses all valid negatives. For the hard-task, hard-object, and hard-action terms, we select the highest-similarity candidate from the corresponding nonempty conflict class c after validity masking: $j_c^*(i) = \arg\max_{j \in \mathcal{N}_c(i)} s(q_i, k_j)$. Their margins are 0.08, 0.10, and 0.06, respectively. Conditional hinge losses are averaged over valid samples within each rank, then averaged across ranks.

Representation regularization. We encourage near-zero batch centers, a per-dimension standard-deviation floor near 0.02, and participation-ratio effective ranks of at least 16 for query batches and 18 for target batches. Center losses weighted at 0.10 per side reinforce isotropy-based centering.

A.3 Training Data and Curation

Our training dataset contains 1,695,684 filtered anchor-level examples from EgoDex (Hoque et al., 2026), DROID (Khazatsky et al., 2024), LIBERO (Liu et al., 2023), and RoboMIND–Franka-dual (Wu et al., 2024). Table 15 gives the source composition. EgoDex contributes tracked human hand and body pose trajectories, encoded by its source-specific native-action projector. Before constructing dense anchors, we assign complete episodes, sessions, and content identities to disjoint training, validation, and original test partitions.

Table 15: **Training-data composition.** Filtered anchor-level training examples by source.

Source domain	Training rows
EgoDex (Hoque et al., 2026)	1,345,160
DROID (Khazatsky et al., 2024)	26,563
LIBERO (Liu et al., 2023)	136,671
RoboMIND–Franka-dual (Wu et al., 2024)	187,290
Total	**1,695,684**

Anchor enumeration and object localization. For a source sequence with N frames, let a be a frame index and h the future horizon in frames. We enumerate dense anchors satisfying $a + h + 1 < N$. We prompt SAM3 with phrases naming the instruction's physical interaction objects, using a proposal threshold of 0.35. We retain detections with a final score of at least 0.40 that cover at least 0.08% of the source image, and permit a zoom ratio of at least 1.10. We independently recompute each accepted crop at 320×192.

Visual quality gate. Qwen3-VL-8B-Instruct (Bai et al., 2025) verifies each crop deterministically. A crop passes only if the model confirms with high confidence that the interaction object is visible, matches the instruction, is separable from other instances, and is unoccluded. If verification fails, we resample an earlier neighboring timestamp, updating the image, synchronized action, and temporal anchor together. Selection uses only the instruction and current observations.

Future target construction. We pair each accepted anchor with the strictly future four-second action sequence and 16 video frames from that interval. The window starts after the accepted timestamp, including when resampling selects an earlier one. Figure 3 shows the pipeline.

B *Physis-Embed-Bench* Evaluation and Diagnostics

B.1 *Physis-Embed-Bench*: Composition and Evaluation

EgoDex, DROID, LIBERO, and RoboMIND–Franka-dual each contribute 1,000 held-out queries from 800 unique episodes (Table 16). The shared 8,000-item gallery contains 2,000 candidates per source: 1,000 paired future clips and 1,000 nearby same-episode clips. Nearby clips often share the positive's instruction, objects, and visual content, requiring precise temporal localization.

Table 16: **Composition of *Physis-Embed-Bench*.** All queries search the shared 8,000-clip gallery; task labels are counted within each source.

Source domain	Queries	Query episodes	Gallery	Tasks
EgoDex	1,000	800	2,000	591
DROID	1,000	800	2,000	800
LIBERO	1,000	800	2,000	39
RoboMIND–Franka-dual	1,000	800	2,000	5
Total	**4,000**	**3,200**	**8,000**	–

The four sources contain 591, 800, 39, and 5 distinct task-instruction strings, respectively, covering human experience, simulation, and single- and dual-arm robots. DROID and LIBERO also support the downstream ICL studies, evaluated separately from retrieval.

Clip-level retrieval assigns one positive to each query: its uniquely paired future four-second window. Episode-level retrieval treats any indexed window from the same source-scoped episode as relevant. Episode-level metrics average query outcomes within each episode, then equally over the 800 episodes per source and the four sources, giving each episode equal weight regardless of its query count. In total, the benchmark contains 4,000 queries and 3,200 episodes. Training, validation, and original test partitions are separated by complete episode, session, and content identity before dense anchoring. Benchmark episodes are excluded from training; task, object, and embodiment categories may overlap across partitions. Every target is strictly later than its paired query anchor. Ranking uses cosine similarity on L2-normalized embeddings, without reranking. We evaluate the final *Physis-Embed* checkpoint at training step 26,275, the last step of retrieval training.

Baseline input interfaces. *Physis-Embed* queries combine instruction, a SAM3 object zoom, and synchronized action; gallery entries combine instruction, future video, and future action. Omni-Embed-Nemotron-3B and Qwen3-VL-Embedding-2B use instruction and the uncropped current frame as the query, and instruction and the 16-frame future clip as the candidate. CLIP, Perception Encoder, and InternVideo2 use the instruction to retrieve the future video clip. Gemini Embedding 2 and VLM2Vec-V2 likewise use instruction plus the original, uncropped current image as the query, without action input, and instruction plus the future four-second video as the gallery entry. For Gemini Embedding 2, we upload a video assembled from 16 frames and obtain normalized 768-dimensional vectors. VLM2Vec-V2 receives the 16 frames directly and uses last-token pooling followed by L2 normalization, producing 1,536-dimensional vectors. Both are evaluated on all 4,000 queries and 8,000 gallery entries. The released WeMM-Embedding-2B model is evaluated zero-shot with instruction and the uncropped current frame as the query, instruction and the 16-frame future clip as the candidate, and L2-normalized 2,048-dimensional outputs. Each external baseline is evaluated through its supported input interface. The controlled DROID study isolates modality choices with matched training data, backbone architecture, and optimization budget.

Frozen Cosmos3-Edge feature baseline. The frozen Cosmos3-Edge control uses the same query and target fields and native action projectors as *Physis-Embed*. We extract dense action-token features after the final RMSNorm of layer 28, before the action-output projection. Temporal averaging and L2 normalization produce a 2,048-dimensional VLA-conditioned backbone readout.

B.2 Benchmark Retrieval Examples

Figure 7 contrasts a paired-clip hit with a temporal near-miss. The latter retrieves a nearby window from the correct episode rather than the paired continuation, illustrating why episode matching and clip localization require separate metrics.

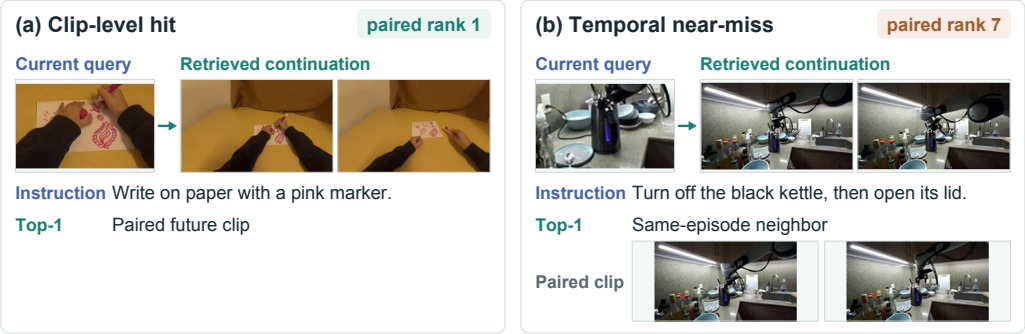

Figure 7: **Selected retrieval diagnostics.** (a) The paired future clip ranks first. (b) A nearby clip from the same episode ranks first, while the paired clip ranks seventh. Both top-1 returns are episode-level matches; only (a) is also a clip-level match.

B.3 Per-Domain Results

Tables 17–19 report retrieval on the shared 8,000-item gallery. Recall and MRR are percentages; lower ranks are better. *Average* weights sources equally. Episode-level metrics first average within episodes, then over 800 episodes per source.

For *Physis-Embed*, EgoDex episode-level R@1 reaches 87.06%, while clip-level R@1 is 36.70%; the corresponding DROID scores are 99.88% and 74.10%. The larger gap on EgoDex shows that identifying the episode often leaves the paired window unresolved, as in the near-miss in Figure 7. VLM2Vec-V2 and WeMM-Embedding-2B both perform best on DROID and EgoDex, with markedly lower recall on LIBERO and RoboMIND-FD.

Table 17: ***Physis-Embed* retrieval by source.** Overall median ranks: clip 2, episode 1.

Domain	R@1	R@5	R@10	MRR	Median rank	Mean rank
Clip-level retrieval						
EgoDex	36.70	96.60	99.30	62.22	2	2.68
DROID	74.10	99.00	100.00	85.27	1	1.41
LIBERO	38.50	87.00	96.60	60.33	2	2.92
RoboMIND-FD	40.00	85.00	91.90	60.39	2	3.87
Average	**47.33**	**91.90**	**96.95**	**67.05**	–	**2.72**
Episode-level retrieval						
EgoDex	87.06	98.94	99.44	90.98	1	1.54
DROID	99.88	100.00	100.00	99.94	1	1.00
LIBERO	67.62	91.81	98.19	77.42	1	2.17
RoboMIND-FD	72.50	88.88	94.62	79.50	1	2.80
Average	**81.77**	**94.91**	**98.06**	**86.96**	–	**1.88**

Table 18: **VLM2Vec-V2 retrieval by source.** Overall median ranks: clip 5, episode 2.

Level	Domain	R@1	R@5	R@10	MRR	Median rank	Mean rank
Clip	EgoDex	32.40	92.00	99.20	56.90	2	2.62
	DROID	58.70	97.80	99.90	75.15	1	1.77
	LIBERO	4.70	16.20	28.10	12.68	22	34.56
	RoboMIND-FD	0.40	1.40	3.00	1.96	185	222.92
	Average	**24.05**	**51.85**	**57.55**	**36.67**	–	**65.47**
Episode	EgoDex	78.13	97.81	99.88	86.04	1	1.50
	DROID	99.19	100.00	100.00	99.51	1	1.01
	LIBERO	10.75	30.44	46.25	22.01	12	17.70
	RoboMIND-FD	1.19	3.94	6.94	3.95	107.5	138.86
	Average	**47.31**	**58.05**	**63.27**	**52.88**	–	**39.77**

Table 19: **WeMM-Embedding-2B retrieval by source.** Overall median ranks: clip 5, episode 3.

Level	Domain	R@1	R@5	R@10	MRR	Median rank	Mean rank
Clip	EgoDex	30.80	90.20	98.50	54.54	2	2.86
	DROID	61.10	98.40	100.00	77.18	1	1.67
	LIBERO	4.30	18.30	30.30	13.31	19	23.18
	RoboMIND-FD	0.30	1.70	3.80	2.09	159.5	204.10
	Average	**24.13**	**52.15**	**58.15**	**36.78**	–	**57.95**
Episode	EgoDex	70.31	96.25	99.63	80.22	1	1.76
	DROID	99.69	100.00	100.00	99.84	1	1.00
	LIBERO	9.00	28.56	44.31	20.05	13	16.70
	RoboMIND-FD	1.25	4.44	8.75	4.20	112.5	140.60
	Average	**45.06**	**57.31**	**63.17**	**51.08**	–	**40.02**

B.4 Candidate Language and Gallery Controls

Two evaluation-time controls on *Physis-Embed-Bench* test reliance on candidate language and discrimination among clips with the same instruction. Queries and targets come from held-out episodes excluded from training, but retain within-episode visual continuity.

Both controls use the same fixed *Physis-Embed* model. *Candidate text masking* replaces each gallery instruction with an empty prompt before multimodal fusion, then recomputes embeddings while preserving candidate video and action. *Same-instruction gallery* restricts each query to candidates with the same instruction, testing within-instruction ranking. The original protocol and candidate-text masking both retain the full shared 8,000-item cross-source gallery.

Table 20: **Aggregate benchmark-validity controls.** Candidate-text masking and same-instruction retrieval use a fixed model. Recall and MRR (%) are averaged equally across four sources.

	Clip-level				Episode-level			
Setting	R@1	R@5	R@10	MRR	R@1	R@5	R@10	MRR
Original	47.33	91.90	96.95	67.05	81.77	94.91	98.06	86.96
Candidate text masked	12.78	33.65	43.10	23.06	23.17	36.83	46.69	30.32
Same-instruction gallery	47.62	92.45	97.25	67.46	82.53	95.23	98.20	87.55

Candidate text masking lowers clip-level R@1 from 47.33% to 12.78% and episode-level R@1 from 81.77% to 23.17%. The model relies on candidate language for current-to-future alignment: masked and original candidate embeddings have a mean cosine similarity of 0.506.

Restricting the gallery to the same instruction yields 47.62% clip-level R@1, approximately 0.3 percentage points above the original protocol after removing different-instruction distractors. The retained candidates share the instruction, so ranking them depends on their visual and action content. Nearby same-episode distractors further require precise localization of the paired future clip.

Table 21: **Same-instruction gallery difficulty.** Candidate counts and clip-level recall (%) within query-specific galleries. Random recall uses the same restricted candidates. The final row averages source means and recalls equally; its candidate median is the cross-source aggregate. RoboMIND-FD denotes RoboMIND–Franka-dual.

	Candidates		R@1		R@5	
Domain	Mean	Median	Model	Random	Model	Random
EgoDex	4.06	4	**37.60**	29.76	**98.40**	95.09
DROID	2.71	2	**74.20**	54.64	**99.20**	97.46
LIBERO	54.51	50	**38.60**	1.90	**87.00**	9.50
RoboMIND-FD	473.59	342	**40.10**	0.24	**85.20**	1.19
All domains	133.72	25	**47.62**	21.63	**92.45**	50.81

Same-instruction pools are small in EgoDex and DROID, so R@5 approaches saturation. LIBERO and RoboMIND-FD have larger pools, averaging 54.51 and 473.59 candidates. Their R@1 scores of 38.60% and 40.10% exceed the corresponding random rates of 1.90% and 0.24%.

Table 22: **Clip-level validity controls by source.** Candidate text masking leaves video and action unchanged; same-instruction galleries contain candidates sharing the query instruction. Recall and MRR are percentages; lower ranks are better.

Setting	**Domain**	**R@1**	**R@5**	**R@10**	**MRR**	**Median rank**	**Mean rank**
Original	EgoDex	36.70	96.60	99.30	62.22	2	2.68
	DROID	74.10	99.00	100.00	85.27	1	1.41
	LIBERO	38.50	87.00	96.60	60.33	2	2.92
	RoboMIND-FD	40.00	85.00	91.90	60.39	2	3.87
Candidate text masked	EgoDex	4.00	14.30	21.40	9.73	97	434.41
	DROID	4.40	13.30	21.20	9.69	71	351.95
	LIBERO	13.40	36.40	49.30	25.03	11	98.95
	RoboMIND-FD	29.30	70.60	80.50	47.77	2	27.95
Same-instruction gallery	EgoDex	37.60	98.40	100.00	63.59	2	2.04
	DROID	74.20	99.20	100.00	85.34	1	1.40
	LIBERO	38.60	87.00	96.80	60.41	2	2.90
	RoboMIND-FD	40.10	85.20	92.20	60.51	2	3.84

Table 23: **Episode-level validity controls by source.** Recall and MRR (%) are averaged over queries within each episode, then equally over the 800 episodes per source (Section 3.3).

Setting	Domain	R@1	R@5	R@10	MRR
Original	EgoDex	87.06	98.94	99.44	90.98
	DROID	99.88	100.00	100.00	99.94
	LIBERO	67.62	91.81	98.19	77.42
	RoboMIND-FD	72.50	88.88	94.62	79.50
Candidate text masked	EgoDex	9.00	16.81	24.75	13.62
	DROID	6.50	16.81	25.25	12.57
	LIBERO	23.19	39.56	53.87	32.09
	RoboMIND-FD	54.00	74.12	82.88	63.01
Same-instruction gallery	EgoDex	89.69	100.00	100.00	92.99
	DROID	100.00	100.00	100.00	100.00
	LIBERO	67.69	91.81	98.19	77.47
	RoboMIND-FD	72.75	89.12	94.62	79.73

B.5 Additional Query Masking

Single-modality queries. Table 24 extends final-model masking (Table 7) to single-modality queries, keeping model weights and gallery embeddings fixed. Text alone reaches 16.23% clip-level R@1; vision and action alone yield lower recall, and all three fall below the fused VLA query.

Table 24: **Single-modality queries.** Four-source average recall (%) with the model and gallery embeddings fixed. Δ gives the percentage-point change in clip-level R@1 from the full VLA query.

Query input	Clip R@1	Episode R@1	Δ clip R@1
Full VLA	**47.33**	**81.77**	—
Text only	16.23	31.09	−31.10
Image only	1.68	3.66	−45.65
Action only	0.53	1.58	−46.80

Controlled DROID masking. We mask inputs to the fixed DROID Full VLA model and compare its response with the retrained variants in Table 6. This test uses the controlled DROID checkpoint and split, whereas Table 7 uses the final checkpoint across four sources.

Table 25: **DROID query-only masking.** Clip-level recall and MRR (%) with the model and gallery embeddings fixed. Δ R@1 gives the percentage-point change from the full VLA query.

Query setting	Input	R@1	R@5	R@10	MRR	Δ R@1
Full query	VLA	**78.7**	**99.4**	**100.0**	**88.00**	**0.0**
Action masked	VL	50.7	90.3	94.9	67.60	−28.0
Image masked	LA	60.9	91.8	96.1	74.17	−17.8
Text masked	VA	1.9	7.2	12.0	5.41	−76.8

Masking action, image, or text lowers clip-level R@1 from 78.7% to 50.7%, 60.9%, or 1.9%, respectively. Removing text has the largest effect, followed by action and image. No-vision retraining reaches 76.8% R@1, whereas masking vision in the full model yields 60.9%. Retraining measures performance after adapting to the available modalities; masking measures the fixed model's reliance on its inputs, including sensitivity to distribution shift.

B.6 Index-Size Scaling and Retrieval Latency

Table 26 measures query encoding and Top-10 search over precomputed galleries on one NVIDIA A100. Timings exclude SAM3 preprocessing, reranking, policy inference, and robot execution. Throughput is the reciprocal of total latency; GPU memory is estimated. Encoding takes 157.70 ms

at every scale, while expanding the gallery from one million to 15 million entries adds 34.81 ms to search latency. Retrieval quality is evaluated separately on the 8,000-entry benchmark.

Table 26: **Retrieval scaling on one NVIDIA A100.** Measured latencies cover Top-10 search alone and query encoding plus search. Throughput is derived from the latter; GPU memory is estimated.

Gallery	Top-10 search (ms)	Encode + retrieve latency (ms)	Single-stream QPS	Est. GPU memory (GB)
1M	2.66	160.36	6.24	11.29
2M	5.14	162.84	6.14	15.39
5M	12.60	170.30	5.87	27.69
10M	25.04	182.74	5.47	48.19
15M	37.47	195.17	5.12	68.69

B.7 Qualitative Cross-Embodiment Analysis

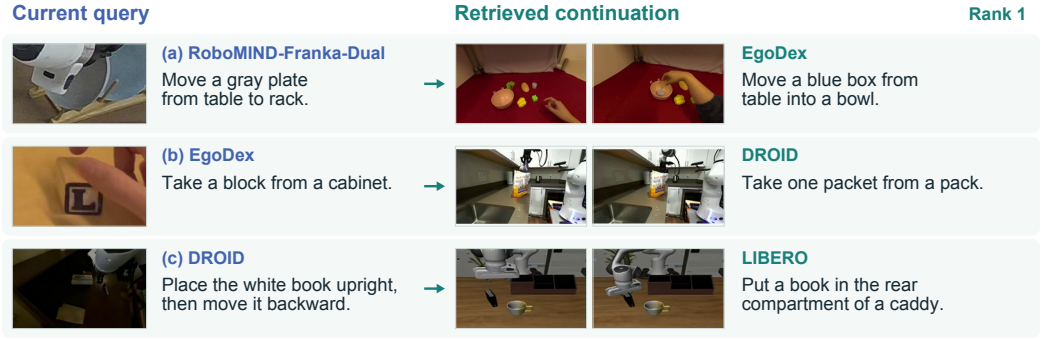

Figure 8: **Selected qualitative evidence across embodiments.** Each query searches only the illustrated target source. The top-ranked clips share placement or removal roles with their queries.

Figure 8 presents three selected cases from a diagnostic checkpoint. Each query searches only the illustrated target source, excluding its own. The top-ranked clips share placement or removal roles with their queries across human, robot, and simulated experience.

C Embed2ICL Implementation and Supplementary Results

This section covers DreamZero's training-free video-cache warm-up and Fast-WAM's training-based Context Tokens, followed by integration controls.

C.1 DreamZero-DROID Reference Warm-Up

Embed2ICL initializes frozen DreamZero-DROID's cache with a retrieved and reranked video (Figure 9). Both retrievers use the same GPT-5.6 sol Reranking Agent. All reference conditions use four-second videos. Each Embed2ICL reference contains three synchronized streams at 15 Hz for four seconds (60 frames per view): 45 frames before the detected final gripper release, the release frame, and 14 frames after it. All three streams serve as video-only policy input. Embed2ICL performs one warm-up per applicable task phase and issues eight reference-conditioned policy requests. Live observations then update the rolling cache. The source episode and temporal window remain fixed across all trials for a task, so the experiment evaluates conditioning on a selected task reference. We evaluate the three tasks in Table 8. No warm-up, random reference, both retrievers' top-1, and both reranked conditions each use 100 trials per task, totaling 300 per condition. All share the released DreamZero-DROID policy; only the test-time video reference changes. The top-1 conditions select an episode and recut a four-second reference from its successful stage.

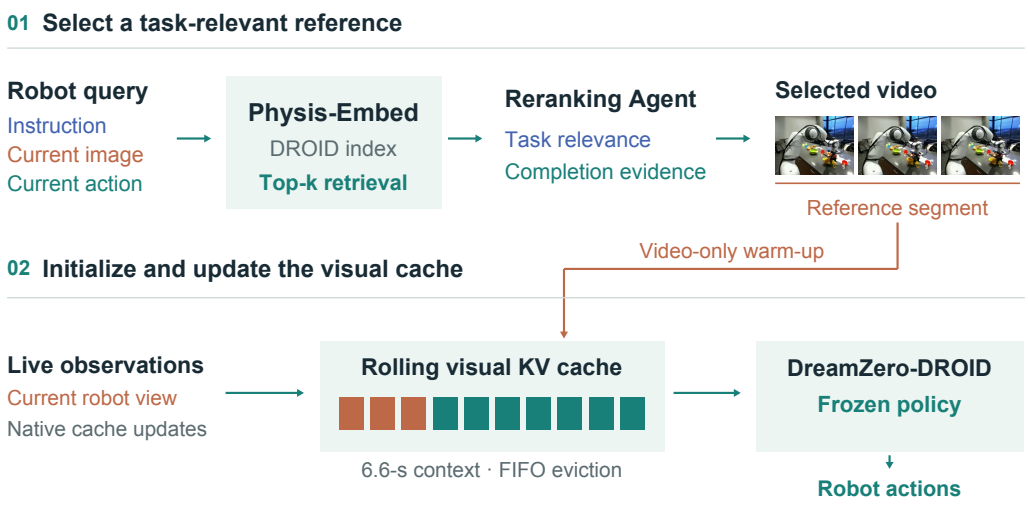

Figure 9: **Training-free Embed2ICL.** *Physis-Embed* retrieves candidate VLA clips, and the Reranking Agent selects a task-relevant reference containing completion evidence. Its video initializes DreamZero-DROID's visual key–value cache. The policy remains frozen while live observations update the cache during execution.

C.2 DreamZero Reference and Interface Analysis

Figure 10 shows the task-specific DROID warm-up references for full *Embed2ICL* (Table 8). All three preserve the placement relation while changing the object or receptacle: Cube→Bowl uses a block-to-bowl example; Can→Mug uses a cap-to-container analogue; and Banana→Bin preserves the manipulated object but changes the receptacle. *Physis-Embed* outperforms Omni-Embed-Nemotron-3B with both raw top-1 selection and reranking (Table 8). Reranking improves the reported rates for both retrievers. It selects references using completion evidence among other criteria; these comparisons measure their usefulness as policy context.

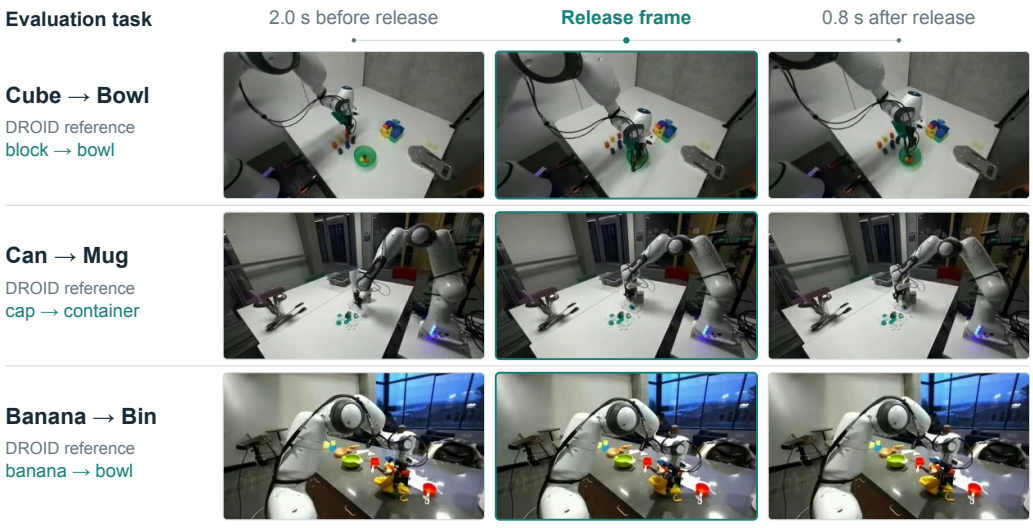

Figure 10: **DROID references for training-free Embed2ICL.** Each row shows one task's fixed reference from a single camera, before, at, and after the detected final gripper release. DreamZero-DROID receives all three synchronized four-second video streams.

Table 27: **Reference-cache warm-up compatibility.** Native test-time state and compatibility with initialization from a retrieved video.

Policy	Test-time video/state interface	Direct video warm-up
DreamZero-DROID (Ye et al., 2026)	Autoregressive video/action; rolling 6.6-s visual KV cache.	**Compatible.** The selected video initializes the state read by later chunks.
LingBot-VA (Li et al., 2026a)	Causal video-action rollout with coupled RoPE-indexed history and a streaming VAE.	**Synchronized initialization required.** Initialization must align video, action, and encoder history.
Fast-WAM (Yuan et al., 2026)	Action replanning with visual KV state reset at each policy call.	**Not natively supported.** Embed2ICL instead uses trained Context Tokens.

Reference warm-up depends on the policy's test-time state. DreamZero-DROID combines a rolling visual cache with a stateless VAE. LingBot-VA (CauVA in the proceedings) requires synchronized initialization of video and action positions and streaming VAE state. Fast-WAM instead uses Embed2ICL's trained Context-Token interface (Appendix C.4).

C.3 Training-Free Embed2ICL on the YAM Robot

Policy and reference. We adapt DreamZero-AgiBot (Ye et al., 2026) to YAM for 20 epochs on 55 play trajectories, comprising approximately 30 minutes of interaction. Both conditions share this adapted policy; Embed2ICL adds retrieved video through visual KV-cache warm-up without further policy updates. It reuses the banana-placement reference selected by *Physis-Embed* and the Reranking Agent for the DROID-Sim Banana-to-Bin task. This DROID reference depicts placement into a bowl; the real-robot task is Banana-to-Bowl.

Evaluation. We evaluate each condition over 12 real-robot rollouts. Success requires the banana to remain inside the bowl while the bowl stays upright. We also report mean cumulative task progress on a 0–100 scale, awarding 20 points for each completed stage: (1) gripper contact with the banana; (2) grasping and lifting it from the table; (3) transporting it above the bowl; (4) releasing it into the bowl; and (5) stable placement with the bowl upright. Embed2ICL raises mean progress from 21.67 to 40.00 (Table 11); success increases from 0.0% to 16.7%.

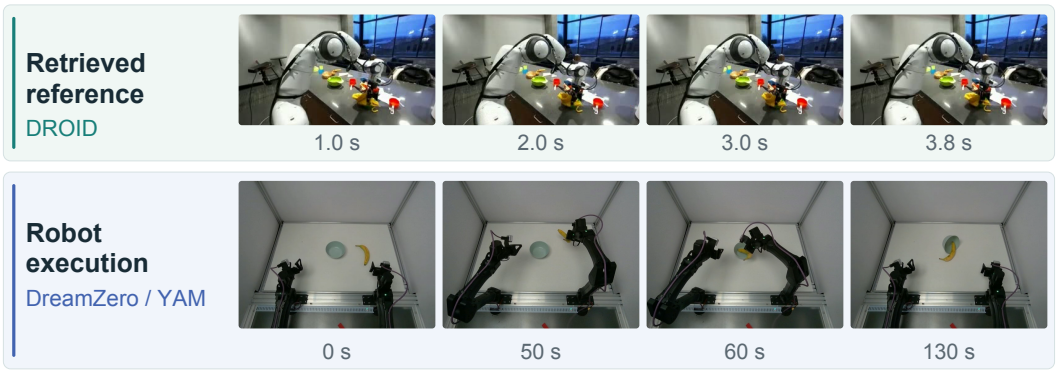

Figure 11: **Retrieved reference and DreamZero execution on YAM.** Simulation and YAM share the DROID banana-placement reference. Frames show a selected real-robot rollout's initial state, transport, bowl interaction, and final visible state. Times are relative to each video's start.

C.4 Fast-WAM Integration Details

Context Tokens from target states. The target encoder jointly encodes task text, future video, and native actions. From valid action slots after layer 28's generation RMSNorm, we extract fused states $H_i^k \in \mathbb{R}^{T_i \times 2048}$ used by the global retrieval branch (Eq. 3). Adaptive temporal average pooling retains at most 32 positions; standard windows from all four sources yield 32. Pooled states pass

through shared LayerNorm, $2048 \to 1024$ projection W, and target residual adapter A_k:

$$\widetilde{H}_i = \mathrm{Pool}_{32}(H_i^k), \qquad u_{i,j} = A_k\Big(W\,\mathrm{LN}(\widetilde{H}_{i,j})\Big), \\ C_{i,j} = u_{i,j}/\|u_{i,j}\|_2, \qquad C_i \in \mathbb{R}^{32\times 1024}. \tag{7}$$

Standard windows yield 32 independently L2-normalized tokens from temporally pooled states, not copies of the global retrieval vector. Cosine search uses k_i; Fast-WAM consumes C_i.

Retrieval-trained parameters. Context Tokens reuse the Transformer, shared projection, and endpoint adapters learned during retrieval training. The video VAE remains frozen. Token extraction adds no separate head or token-level alignment or reconstruction loss.

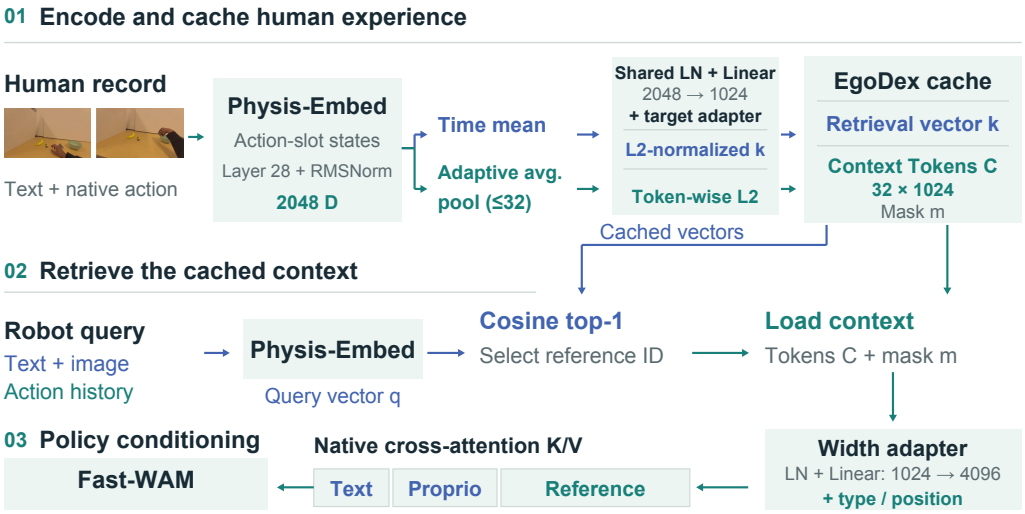

Figure 12: **Training-based Embed2ICL.** Offline encoding caches each candidate's retrieval vector, Context Tokens, and context mask. Online cosine top-1 search selects a reference and loads its tokens and mask. The prefilling adapter projects the tokens to Fast-WAM's context width and appends them to the native text and proprioceptive conditioning for cross-attention.

Offline cache and online lookup. Offline, we cache each candidate's retrieval vector, Context Tokens, and context mask m_i. Online, we encode the robot query as q, find $i^* = \arg\max_i q^\top k_i$, and load (C_{i^*}, m_{i^*}) without re-encoding the candidate. Each rollout starts without reference context. After the first successful retrieval, that reference stays active until the rollout ends. Task-progress metadata is excluded from conditioning.

Prefilling. Embed2ICL projects $C = C_{i^*}$ to Fast-WAM's conditioning width:

$$R = \mathrm{Linear}_{1024\to 4096}(\mathrm{LN}(C)) + E_{\mathrm{type}} + E_{\mathrm{pos}}, \quad R \in \mathbb{R}^{32\times 4096}. \tag{8}$$

The projected sequence extends native conditioning $[H_{\mathrm{text}}; h_{\mathrm{proprio}}]$; E_{type} and E_{pos} are learned type and position embeddings. Native cross-attention reads it in all 30 video-expert and 30 action-expert layers, at every action-diffusion denoising step, and during video KV-cache prefilling.

Policy-adaptation scope. Video and action flow-matching losses update approximately 5.00B video-expert parameters, 1.02B action-expert parameters, the proprioception encoder, and a 4.34M-parameter projection adapter: approximately 6.02B trainable parameters in total. The text encoder and VAE remain frozen. The context adapter inserts cached tokens into native conditioning, with no gradient propagation to the offline *Physis-Embed* encoder.

LIBERO training schedule. The retrieved-context condition starts from the official Fast-WAM LIBERO checkpoint and trains on 1,712 trajectories comprising 277,713 samples across the four LIBERO suites. Five epochs yield 21,700 parameter updates with an effective batch size of 64. We use AdamW with a learning rate of 10^{-4}, a 5% warm-up, and cosine decay to 10^{-6}. The random-reference condition evaluates this same policy checkpoint. A separate no-ICL control uses the same 21,700 training updates (Appendix C.5).

Real-robot evaluation. Pick Fruit and Split Cup each use 12 rollouts per task and condition. Following T-Rex's partial-credit approach (Niu et al., 2026), we report $100 \frac{1}{12} \sum_{i=1}^{12} s_i$, where $s_i \in [0, 1]$ is a rollout's normalized task-progress score.

Additive scoring. Pick Fruit and Split Cup award 0.25 independently for each completed sub-step, up to a total of 1. For *Pick Fruit*, the four sub-steps are: the left hand grasps a fruit from the left side; the left hand places that fruit in the center basket; the right hand grasps a fruit from the right side; and the right hand places that fruit in the center basket. For *Split Cup*, they are: the left hand grasps and stabilizes the cup stack; the right hand separates exactly one cup; the right hand grasps the separated cup; and the separated cup is placed upright on the table.

Table 11 reports mean task progress and gains computed before rounding. Figure 5 pairs retrieved references with illustrative executions for these two tasks and DreamZero/YAM.

C.5 LIBERO Supplementary Controls

Table 10 compares retrieved and random Context Tokens with the official release without ICL. The retrieved- and random-reference conditions evaluate the same policy checkpoint, changing only the source of Context Tokens at inference. For the random-context condition, a fixed seed deterministically selects a pseudorandom real reference window from the EgoDex gallery. Its precomputed 32×1024 Context Tokens replace the retrieved window's tokens; the control neither injects noise nor shuffles token order. Each condition uses 50 binary-success trials per task across the five-task LIBERO difficult subset. Retrieved tokens reach 92.8% success, compared with 88.0% for tokens from random references. This isolates reference selection at fixed policy weights.

Separately, continuing the official release without ICL for 21,700 updates, matching Embed2ICL's update count, yields 216/250 successes (86.4%). Success remains unchanged from the official release and 6.4 percentage points below Embed2ICL. Thus, continuation for the same number of updates does not reproduce the gain from retrieved-context conditioning. An earlier 2,000-step partial report also shows no gain on its two tasks: S4 falls from 46/50 to 45/50 and G9 from 45/50 to 43/50; a five-task aggregate was not reported for that evaluation.

Retrieved human context. Figure 13 pairs the G3 bowl-to-drawer task with a top-1 EgoDex clip of a human moving an object toward a bowl. Both depict placement into a receptacle. The selected Embed2ICL rollout completes G3, whereas the rollout from the official release does not.

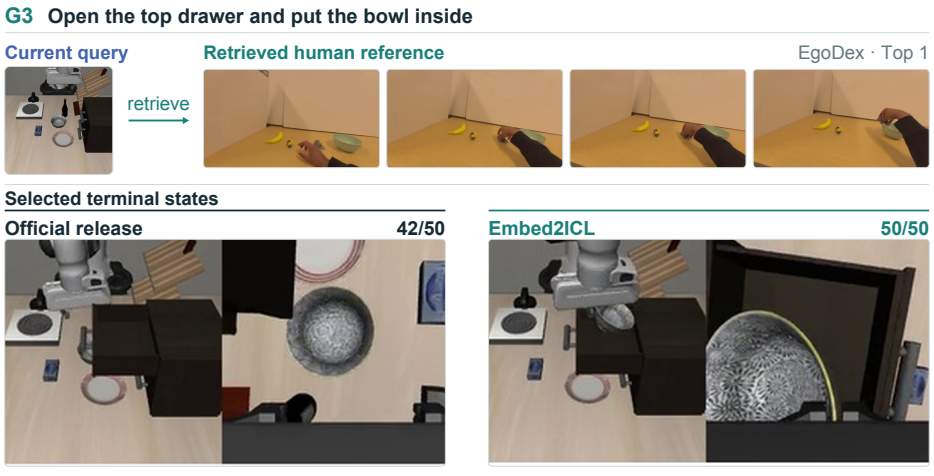

Figure 13: **LIBERO G3: reference and final states.** Successes out of 50 per condition.

C.6 Training-Based Integration Ablation

We compare two complete integration settings: Embed2ICL's native-context prefilling with policy fine-tuning, and a frozen-backbone reference side branch. Both use the *Physis-Embed* retriever, EgoDex index, cosine top-1 selection, and complete 32×1024 Context Tokens. They differ in both the conditioning path and trainable scope.

Side-branch setting. A shared reference encoder projects the cached tokens to width 512. Each of the 30 video-expert and 30 action-expert layers receives an eight-head reference cross-attention block, inserted after self-attention and before native text/proprioception cross-attention. Channel-wise residual gates are initialized to zero, preserving the original policy at initialization. Training updates the reference encoder, added attention blocks, and gates (approximately 158.74M parameters), while freezing the original Fast-WAM backbone and proprioception encoder. Standard Embed2ICL instead fine-tunes approximately 6.02B parameters with its prefilling interface (Appendix C.4).

Table 28: **LIBERO integration ablation.** Successes from 50 trials per task and condition; the final row gives the five-task average (%). Parentheses show percentage-point changes from the official release; bold marks the better ICL setting.

Suite / ID	Official release	Side branch	Embed2ICL
Spatial / S4	46	46 (0)	**49** (+6)
Goal / G3	42	44 (+4)	**50** (+16)
Goal / G9	45	44 (−2)	**47** (+4)
LIBERO-10 / L4	43	43 (0)	**44** (+2)
LIBERO-10 / L6	40	**43** (+6)	42 (+4)
Five-task average (%)	86.4	88.0 (+1.6)	**92.8** (+6.4)

LIBERO. The side branch reaches 88.0% average success, compared with 86.4% for the official release and 92.8% for Embed2ICL (Table 28). Embed2ICL performs better on four tasks; the side branch performs better on L6 (43 versus 42 successes). The side branch uses fewer trainable parameters, but Embed2ICL reaches higher average success.

Table 29: **Real-robot integration ablation.** Mean normalized task progress (%) from 12 rollouts per task and condition. Parentheses show percentage-point changes from Fast-WAM without ICL.

Task	No ICL	Side branch	Embed2ICL
Pick Fruit	75.00	77.08 (+2.08)	**79.17** (+4.17)
Split Cup	66.67	75.00 (+8.33)	**87.50** (+20.83)

Real robots. Both settings improve mean normalized task progress over no ICL (Table 29). Embed2ICL reaches 79.17% on Pick Fruit and 87.50% on Split Cup, compared with 77.08% and 75.00% for the side branch. The larger gap occurs on Split Cup.

Table 30: **Retrieved versus random Context Tokens.** Each integration is evaluated on the LIBERO difficult subset with 50 trials per task and condition. Results are five-task average success rates (%).

Integration	Random tokens	Retrieved tokens
Side branch	85.6	**88.0**
Embed2ICL	88.0	**92.8**

Retrieved versus random context. Retrieved tokens yield higher success than random tokens within each setting (Table 30): 88.0% versus 85.6% for the side branch and 92.8% versus 88.0% for Embed2ICL. In the Embed2ICL comparison, both reference conditions use the same policy checkpoint. Retrieved context helps in both the frozen-backbone and fine-tuned settings.

D Embed2Evolve: Harnesses, Development, and Evaluation

Embed2Evolve accumulates completed attempts in task-local memory and searches for a recipe for using that memory during control. Online retrieval ranks individual memory anchors; development compares entire context configurations. The two decisions use different criteria. The policy and embedding models remain fixed throughout.

D.1 Policy and Memory Interfaces

Matched embedding-model design. We compare *Physis-Embed* with Omni-Embed-Nemotron-3B (Xu et al., 2025) by replacing only the retrieval embedding model and developing each condition independently. Both use the same GPT-6 Astra policy, prompt, initial memory, development and test seeds, candidate-generation rules, and stopping criteria. Retrieved memories supply the same RGB, action, and feedback fields to the policy. Both evolved systems use the full memory interface; the random-history baseline supplies only RGB.

At each decision, GPT-6 Astra receives all current RGB views, robot state, execution feedback, recent action/feedback history, and external context when available. It proposes end-effector target poses and gripper commands, which are validated and executed by a fixed IK/controller. Table 31 gives the runtime and budgets. The four single-arm tasks use `base_camera`, `vision_side`, `vision_top`, and `vision_wrist`; TwoRobotStackCube uses base, side, top, and separate left/right wrist cameras. The policy observes all four or five views, respectively, at every decision.

Table 31: **Embed2Evolve runtime and observation interface.** Simulation time is distinct from client execution time, which includes API waiting.

Component	Configuration
Policy	GPT-6 Astra, reasoning effort `low`
Simulator	ManiSkill 3.0.1; SAPIEN 3.0.3
Current RGB	One 256×256 image per camera per decision
Cameras	Four for single-arm tasks; five for TwoRobotStackCube
Control rate	20 Hz; two ticks per primitive (0.1 s simulation time)
Decision budget	At most eight primitives per GPT decision
Rollout budget	100 decisions, 300 primitives, and 600 policy control ticks
Target-change limits	Per tick: 0.026 m translation and 0.1 rad rotation

Action execution and observation constraints. Each primitive specifies one arm's world-frame end-effector position in metres, unit XYZW quaternion, and gripper open/close command. Targets are transformed to that robot's base frame and executed by the native IK/PD controller. In the two-arm task, the other arm holds its measured pose at the start of the primitive. The shared prompt requires explicit targets, without hidden tool calls or automatic execution until a target is reached. It restricts observations to RGB, the robot's own state, the instruction, and public actions and feedback; object poses, private simulator state, and task code are excluded. A gripper command does not establish grasp success, nor does an IK target establish arrival. The `joint_eff` field estimates the robot's own PD drive effort, not contact ground truth or measured torque. Native environment feedback determines task success.

Historical-context prompt. Both conditions retain the Direct system prompt except that "This condition is Direct: no memory or retrieval." is replaced by "A historical RGB video excerpt may accompany the current observations." Both use this fixed preamble for historical video:

> This reference is a uniformly random four-second excerpt from a retrieved historical video. The images are historical, not current observations.

Here, "uniformly random" is inherited prompt wording: both conditions use the clip associated with the retrieved anchor, not a randomly drawn excerpt from the selected video.

Anchor index and query. For *Physis-Embed*, the query combines the instruction, current main-camera RGB, and an endpoint-error action proxy derived from the most recent acknowledged active-arm command, rather than achieved displacement. Applying the LIBERO action codec to ManiSkill is an out-of-distribution adaptation.

Omni-Embed-Nemotron-3B runs in BF16 with eager attention. Its query uses the `query:` prefix, task text, and current RGB, without native actions; its target uses `passage:`, task text, and historical video. Retrieved context supplied to GPT includes the clip's public actions, execution feedback,

relative timestamps, and robot state after command acknowledgement. These policy-context fields are distinct from the embedding model's inputs.

Memory indexes completed development episodes, including successes and failures. Each key represents a current-state anchor linked to at most four seconds of recorded RGB, commands, robot state, feedback, and timestamps. Retrieval matches the task and active arm, excludes episodes with the current seed, and selects anchors from distinct source episodes.

Ranking and refresh. For candidate anchor j, let $S_j \in \{0, 1\}$ be its source episode's native success label and L_j its total control ticks, with budget 600. For normalized query and anchor vectors q, z_j from the chosen embedding model, the score uses success/efficiency bonuses α, β:

$$R(q, j) = \cos(q, z_j) + \alpha S_j + \beta S_j \left(1 - \frac{L_j}{600}\right). \tag{9}$$

Candidates are ranked in descending order, with at most one anchor retained per source episode. The efficiency bonus rewards successful episodes that use fewer control ticks; tokens, cost, and policy decision counts do not enter the score. For example, 300 ticks with $\alpha = 0.10, \beta = 0.03$ adds 0.115 to cosine similarity. The episode-level label records the outcome of the whole attempt, so it does not establish that every local action was correct. Decision zero uses no retrieved context. Retrieval starts at decision one, refreshes at the selected cadence and upon an active-arm change, and caches context between refreshes.

Failed attempts. Ordinary task failures and budget-exhausted episodes count as unsuccessful trials and can enter memory. With $S_j = 0$, their anchors receive neither a bonus nor a penalty and remain retrievable by similarity. Exhausted repairs of malformed policy output also count as valid failures, but the memory builder does not admit these episodes. Audited infrastructure interruptions are excluded from the valid-trial denominator and memory; replacement runs complete the trial budget. The policy receives failed attempts as execution records under the fixed prompt; these records are not rewritten as prompt advice.

D.2 Development and Selected Harnesses

Search varies retrieval count k, refresh cadence δ, history pairs H, and success/efficiency bonuses α, β. Models, prompt prose, controller, and episode budgets stay fixed. Every candidate uses development seeds 0–9 and retains historical images. The history window keeps the most recent $2H$ messages, corresponding to H action/feedback pairs; δ counts policy decisions. Each task begins with the three presets in Table 32.

Table 32: **Initial context configurations.** All presets enable historical images and are evaluated on the same ten development seeds.

Preset	k	**Cadence** δ	**History** H	α	β
A	2	2	12	0.10	0.03
B	1	4	6	0.10	0.10
C	2	1	8	0.15	0.05

Success-first selection. Let $\mathcal{E}$ contain the eligible numbered development rounds. For round ℓ, write N_ℓ for its number of successful trials, and $\overline{L}_\ell, \overline{T}_\ell$ for mean control ticks and client execution time over its successful trials. We select lexicographically:

$$\ell^\star = \arg\min_{\ell \in \mathcal{E}} \left(-N_\ell, \overline{L}_\ell, \overline{T}_\ell, \ell\right). \tag{10}$$

We first maximize successes, then break ties by fewer control ticks, shorter time, and the earlier round. Every round contains ten valid trials. If none succeeds, both successful-trial means are $+\infty$,

so rapid failure confers no efficiency advantage. Configuration selection thus ranks success and efficiency in order of priority; online retrieval combines them in a weighted score.

Local proposals. After the presets, search round r copies the current winner and makes one deterministic local proposal; all unlisted settings are inherited:

- If $r \bmod 3 = 1$, set $\delta \leftarrow \min(4, \delta + 1)$ and $H \leftarrow 6$ if $H > 6$, or $H \leftarrow 4$ otherwise.
- If $r \bmod 3 = 2$, set $\delta \leftarrow \max(1, \delta - 1)$, $k \leftarrow 2$, and $\alpha \leftarrow \min(0.25, \alpha + 0.05)$.
- If $r \bmod 3 = 0$, set $k \leftarrow 1$ and $\beta \leftarrow \min(0.20, \beta + 0.05)$.

Memory expands between rounds, so development changes both available experience and its retrieval recipe. Search round r need not equal the numbered report round ℓ: earlier transport-version rounds can contribute memory but are excluded from current-version configuration selection.

Stopping and freezing. Stopping tracks the Pareto frontier of $(N_\ell, -\overline{L}_\ell, -\overline{T}_\ell)$. A point dominates another when it is no worse in all three coordinates and better in at least one. A candidate adds a frontier point only if no existing point dominates it and its coordinates are not already present.

After at least three eligible rounds under the current transport version, two consecutive rounds without a new frontier point trigger stopping. A separate cap allows at most eight numbered development rounds per task, including earlier transport versions; baseline collection and formal testing do not count toward it. Any started round must complete ten valid trials.

For each embedding model, formal testing freezes the winning configuration and the memory bank from that round. Test outcomes do not enter selection. The two conditions develop independently and can produce different banks and recipes. The comparison measures the embedding model's contribution through evolution, not a replacement within one final harness.

With *Physis-Embed*, StackCube stops by the frontier rule after four rounds and TwoRobotStackCube after eight. PokeCube, LiftPegUpright, and PullCubeTool each reach the eight-round cap. The 36 recorded configurations and 360 trials include historical first rounds that are ineligible for selection. Table 33 lists the selected harnesses, and the trajectories below show eligible candidates.

Table 33: **Selected Embed2Evolve harnesses.** Bank sizes count development episodes and indexed anchors. Cadence counts GPT decisions; H counts retained action/feedback pairs.

Task	Round	Episodes	Anchors	k	Cadence	H	α/β
Poke Cube	8	80	1,610	2	2	6	.15/.05
Stack Cube	2	20	454	2	2	12	.10/.03
Lift Peg Upright	6	60	1,116	2	1	6	.20/.05
Pull Cube with Tool	8	80	1,667	2	3	4	.15/.05
Two-Robot Stack Cube	5	50	1,126	2	3	6	.10/.03

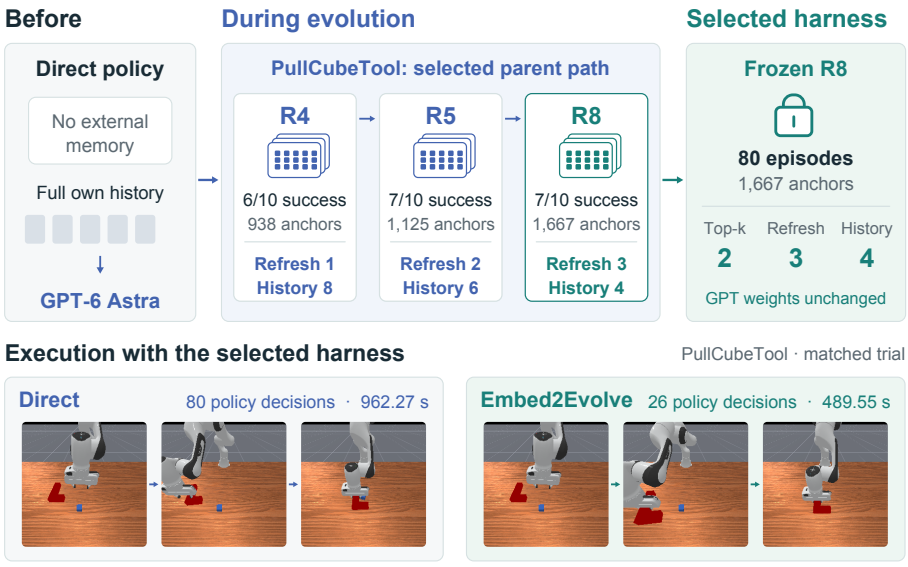

Figure 14: **From Direct control to a selected harness.** PullCubeTool's R4 → R5 → R8 parent–child sequence grows memory while reducing retained history and retrieval frequency. The bottom filmstrips show the same fresh-test initialization completed successfully by both systems. Task execution time includes API waits.

D.3 Development Trajectories

Figures 15 and 16 trace Physis-Embed's candidate performance and context recipes. Decision counts average each round's successful trials; test-time efficiency uses matched initializations (Appendix D.5). Hollow diamonds mark selected harnesses, which need not come from the final round.

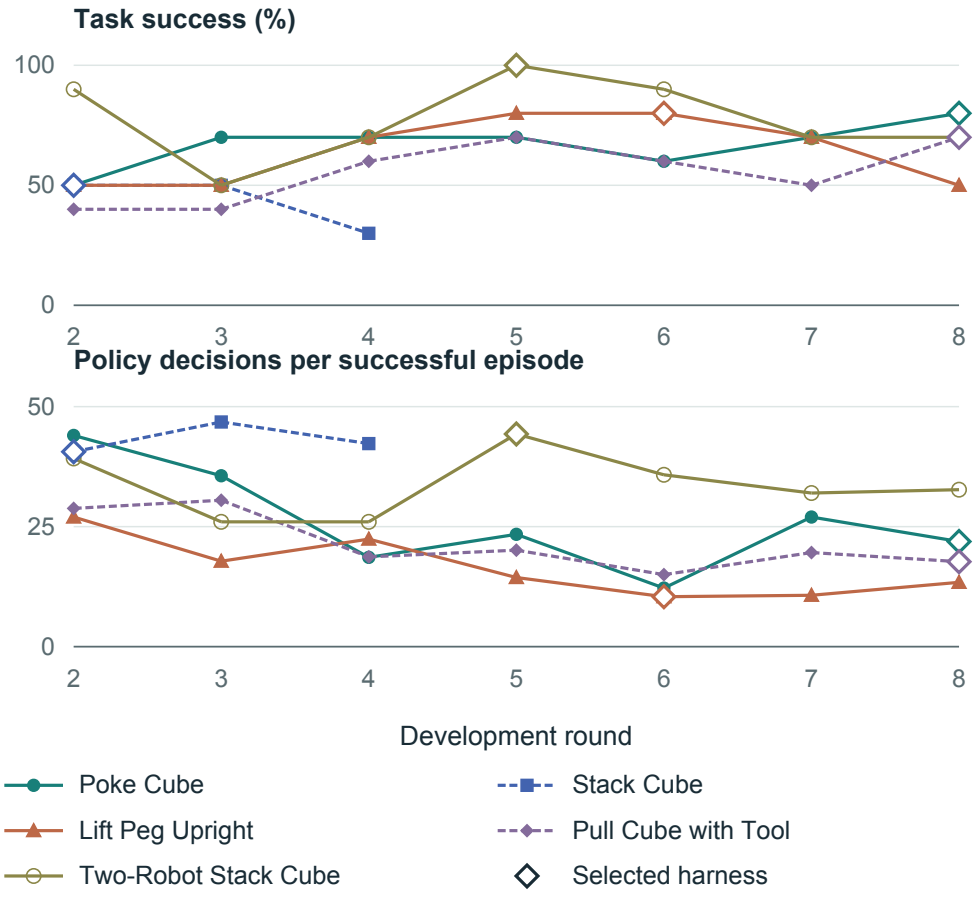

Figure 15: **Task-wise development trajectories.** Success uses ten trials per round; decisions average that round's successful trials. Hollow diamonds mark selected harnesses. Lines connect evaluated candidates, not necessarily parent–child configurations.

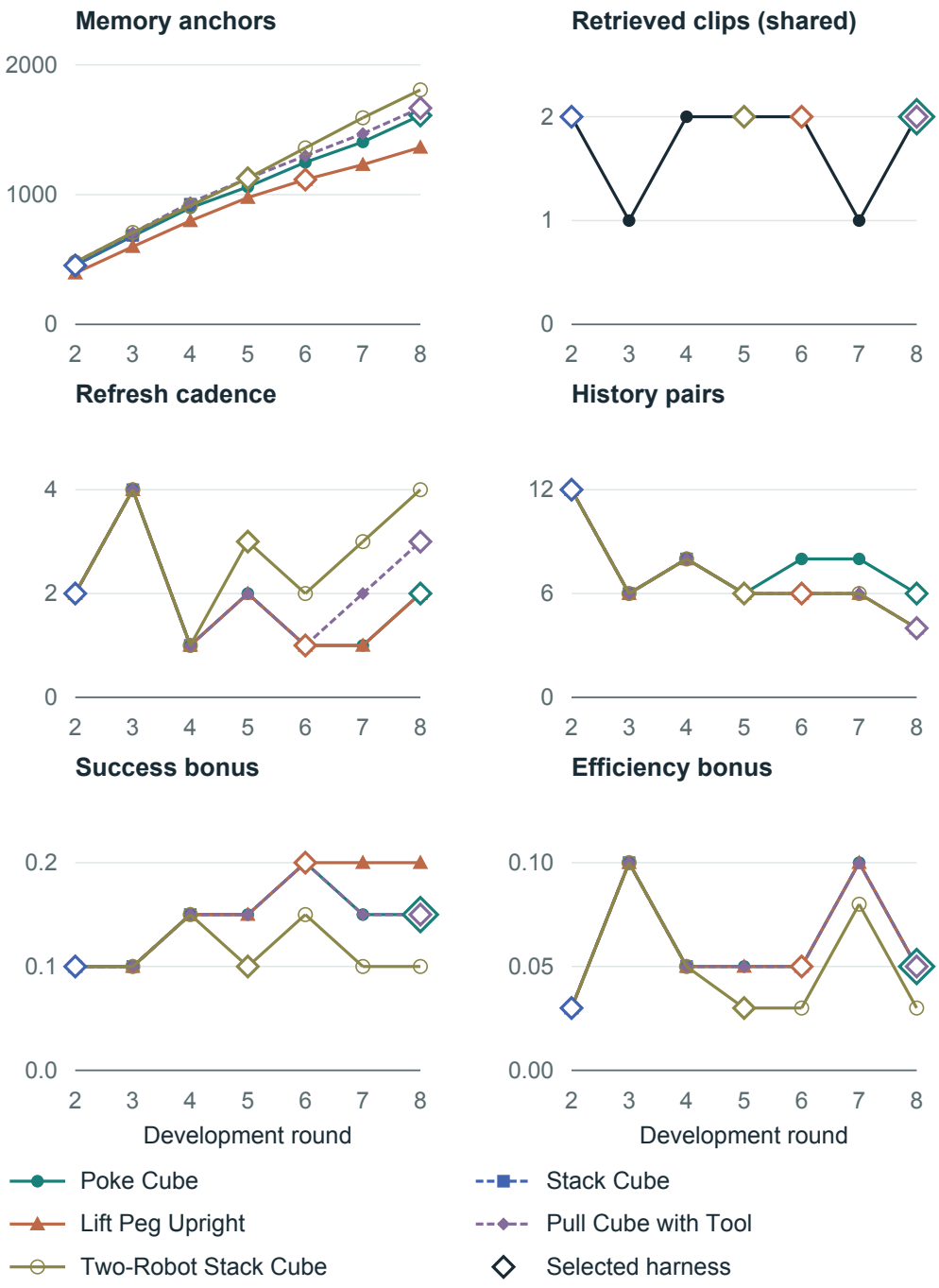

Figure 16: **Two evolving components: memory and context recipe.** Per-task trajectories show memory size, retrieval count, refresh cadence, history, and ranking bonuses with fixed model weights. Hollow diamonds mark selected rounds; nested diamonds show coincident selections.

D.4 Embedding-Model Comparison

Table 34 lists the completed and selected rounds for each embedding model. Both stay within the eight-round cap and produce their own memory banks and context configurations.

Table 34: **Independent development under matched search rules.** Completed rounds count all numbered rounds; selected rounds identify the configuration and memory frozen for testing.

	Omni-Embed-Nemotron-3B		Physis-Embed	
Task	**Completed**	**Selected**	**Completed**	**Selected**
Poke Cube	8	6	8	8
Stack Cube	6	3	4	2
Lift Peg Upright	8	4	8	6
Pull Cube with Tool	3	1	8	8
Two-Robot Stack Cube	7	5	8	5

Table 35 records the rollout videos in each selected Omni-Embed-Nemotron-3B release's frozen bank. These snapshots contain 182 videos: 49 from the original Direct collection and 133 from that condition's own development, with 30 successful and 152 failed rollouts. They are not reconstructed by merging all trajectories after development ends. Video counts differ from indexed-anchor counts; a failed rollout remains retrievable by similarity under the scoring rule in Appendix D.1.

Table 35: **Frozen memory banks for Omni-Embed-Nemotron-3B.** Counts refer to rollout videos bound to the selected releases. Source columns and outcome columns each sum to the video total.

Task	**Videos**	**Direct source**	**Own development**	**Success**	**Failure**
Poke Cube	54	9	45	5	49
Stack Cube	30	10	20	5	25
Lift Peg Upright	38	10	28	16	22
Pull Cube with Tool	10	10	0	1	9
Two-Robot Stack Cube	50	10	40	3	47
Total	182	49	133	30	152

The harness developed with *Physis-Embed* succeeds on 68 of 100 tests, compared with 33 for Omni-Embed-Nemotron-3B, and improves success on every task (Table 12). Both systems have access to memory, but the retrieval representation affects the quality of the resulting harness. This comparison changes the entire embedding model, including its input modalities and retrieval training. The benchmark's clip-localization and modality results suggest an explanation: action-conditioned retrieval may help distinguish execution stage within the same task.

Context as an action prior. Retrieved continuations can provide a soft prior over GPT-6 Astra's next actions. For tool use, an action–outcome example links tool contact and motion to their effect on an object; for dual-arm manipulation, it provides a reference for arm roles and coordination. Stage-relevant context may thus focus the policy on plausible continuations within the unchanged action interface. Such guidance offers a possible explanation for higher success with fewer policy decisions. Fewer high-level decisions can also coexist with more physical control ticks (Table 38); the two measure different aspects of execution.

D.5 Fresh-Test Protocol and Efficiency

The five tasks—PokeCube, StackCube, LiftPegUpright, PullCubeTool, and TwoRobotStackCube—were selected from earlier development results. All four conditions use the same 20 test seeds (40–59) per task, disjoint from development seeds 0–9, yielding 400 valid test episodes. We measure native task success; budget exhaustion and policy failures count as failures. Audited infrastructure interruptions are excluded.

System comparisons. GPT-6 Astra Direct (Direct) uses its original prompt and full within-episode action/feedback history without external memory. GPT-6 Astra + random history adds one randomly sampled same-task development clip with up to eight ordered RGB frames. This RGB-only reference remains fixed within the episode. The harnesses evolved with Omni-Embed-Nemotron-3B and *Physis-Embed* use their selected memories, bounded histories, ranking, and context refresh. Table 12 compares these four context systems.

Success rates and token-use means include all test trials; decision-count and execution-time comparisons use shared successes. Table 36 specifies each metric's population and aggregation rule.

Table 36: **Embed2Evolve test-metric accounting.** All five tasks receive equal weight. Paired comparisons and reference API costs use Direct and Embed2Evolve.

Metric	Test population	Five-task aggregation
Task success	All 20 trials per task in all four conditions	Mean of task success rates.
Policy decisions and execution time	Matched initializations successful under Direct and Embed2Evolve	Mean of task-wise relative reductions; speedup averages Direct/Embed2Evolve time ratios.
Policy tokens	All 20 trials per task for Direct, evolution with Omni-Embed-Nemotron-3B, and Embed2Evolve, including failures	Relative reduction between equal-task per-rollout means.
Reference API cost	All 20 trials per task for Direct and Embed2Evolve, including failures	Relative reduction between equal-task per-rollout means at fixed input/output rates.

Efficiency accounting. Direct and Embed2Evolve both succeed on 7, 2, 5, 1, and 4 matched initializations in task order. Here, Evolve denotes Embed2Evolve. Per task, we compute the reduction $100(1 - \overline{x}_{\text{Evolve}}/\overline{x}_{\text{Direct}})$ on this paired set. Averaging the five task-wise reductions equally gives 64.1% fewer policy decisions and 41.4% less task execution time. Mean speedup is $\frac{1}{5}\sum_{j=1}^{5} \overline{T}_{\text{Direct},j}/\overline{T}_{\text{Evolve},j} \approx 1.90$, where each mean uses that task's shared successful trials. The $1.90\times$ speedup and 41.4% reduction average task-wise time ratios and relative savings, respectively.

Task execution time includes context preparation, API waiting, and execution overhead, but excludes embedding-model cold-start time. It varies with API load, whereas replay videos show simulation time without API waits. Table 38 reports physical control ticks separately from GPT decisions.

Table 37: **Efficiency on shared successes.** Direct $\rightarrow$ Embed2Evolve means over n matched successful initializations. Execution time includes API waiting.

Task	Paired n	Policy decisions	Execution time (s)
Poke Cube	7	34.86 $\rightarrow$ **22.71**	419.30 $\rightarrow$ **382.16**
Stack Cube	2	45.00 $\rightarrow$ **10.00**	499.03 $\rightarrow$ **186.35**
Lift Peg Upright	5	57.20 $\rightarrow$ **12.40**	556.06 $\rightarrow$ **234.46**
Pull Cube with Tool	1	80.00 $\rightarrow$ **26.00**	962.27 $\rightarrow$ **489.55**
Two-Robot Stack Cube	4	87.25 $\rightarrow$ **33.00**	1054.67 $\rightarrow$ **751.59**

Table 38: **Physical control on shared successes.** Mean executed ticks on the same paired trials as Table 37. Change is $100(\text{Evolve}/\text{Direct} - 1)$; positive values mean more physical control ticks despite fewer GPT decisions.

Task	Paired n	GPT-6 Astra Direct	Embed2Evolve	Change (%)
Poke Cube	7	133.0	181.4	+36.4
Stack Cube	2	90.0	65.0	−27.8
Lift Peg Upright	5	152.0	149.6	−1.6
Pull Cube with Tool	1	347.0	337.0	−2.9
Two-Robot Stack Cube	4	206.0	312.8	+51.8

D.6 Token Usage and Reference Cost

Table 39 reports mean policy input, output, and total tokens across all 20 test rollouts per task and condition, including successes and failures, for GPT-6 Astra Direct and Embed2Evolve. We average the five task means equally before comparing methods: total tokens fall from 920,979.76 to 518,216.15 per rollout, a 43.7% reduction. Reference API cost falls from $9.9641 to $5.7980, or 41.8%. Both reductions use $100(1 - \overline{x}_{\text{Evolve}}/\overline{x}_{\text{Direct}})$ on these five-task means. They measure resource use per attempt, including failures; timing in Appendix D.5 uses initializations on which both Direct and Embed2Evolve succeed.

Reference costs use fixed rates of $10 per million input tokens and $50 per million output tokens, giving $(10I + 50O)/10^6$ USD for input count I and output count O. This fixed-rate estimate of test-time policy cost excludes cache discounts, retrieval, simulation, and memory/harness development.

Total token use decreases on every task, most on Pull Cube with Tool (57.5%). Input tokens account for most of the savings. Even on Stack Cube, where output usage increases, the reduction in input tokens lowers total consumption.

Token use after matched context evolution. The harness independently evolved with Omni-Embed-Nemotron-3B averages approximately 2,531,000 policy tokens per test rollout, including both successes and failures. Embed2Evolve averages 518,216.15 under the same test protocol: approximately 79.5% fewer tokens alongside 35 percentage points higher success. These means describe the selected harnesses at test time and exclude development consumption. The timing results above compare Direct and Embed2Evolve on their shared successful trials.

Table 39: **Policy token usage and reference cost per rollout.** Means include all 20 test rollouts per task and condition, regardless of outcome. Direct denotes GPT-6 Astra Direct; the final group averages task means equally. USD estimates use the fixed input/output rates above.

Method	Input tokens	Output tokens	Total tokens	Ref. cost (USD)
Poke Cube				
Direct	693,733.05	16,052.50	709,785.55	7.7400
Embed2Evolve	375,759.60	12,903.40	388,663.00	4.4028
Stack Cube				
Direct	935,086.30	18,401.05	953,487.35	10.2709
Embed2Evolve	656,130.60	19,000.10	675,130.70	7.5113
Lift Peg Upright				
Direct	839,705.25	16,225.95	855,931.20	9.2083
Embed2Evolve	380,229.95	13,087.65	393,317.60	4.4567
Pull Cube with Tool				
Direct	955,384.05	20,302.90	975,686.95	10.5690
Embed2Evolve	399,519.65	15,041.75	414,561.40	4.7473
Two-Robot Stack Cube				
Direct	1,086,699.35	23,308.40	1,110,007.75	12.0324
Embed2Evolve	702,461.85	16,946.20	719,408.05	7.8719
Five-task average				
Direct	902,121.60	18,858.16	920,979.76	9.9641
Embed2Evolve	502,820.33	15,395.82	518,216.15	5.7980

E Related Retrieval Protocols

Table 40 compares retrieval inputs, the role of action, gallery scope, and policy dependence. Action can enter the retrieval key, a retrieved record, or the consuming policy's output. *Policy coupling* specifies how the retrieval interface depends on that policy; *lifetime* specifies the scope over which retrieved experience is stored or reused. In Zeva, both phase keys encode recurrent interaction history, including executed actions and observed effects. RADAR's row follows its observation–language retrieval formulation. Embed2ICL and Embed2Evolve share *Physis-Embed*'s retrieval space.

Table 40: **Representative physical-experience retrieval interfaces.** V/L/A/S/P denote vision, language, native action, state, and proprioception; 3D denotes point clouds. Training-only action use is listed under action role; cross-source galleries span datasets or interfaces.

Method	Modalities	Query → candidate	Action role	Cross-source	Policy coupling	Lifetime
Deep Multimodal	3D/L/A	3D+L → trajectory	Candidate key	No	Independent	Offline corpus
CLASP	L/S/A	L → state+action	Candidate key	No	Independent	Offline corpus
VLA-R	V/L/A	scene V/L → action token	Candidate key	No	Direct control	Action vocab.
ROSER	P	prototype → window	None	No	Independent	Offline corpus
RAEA	V/L	V/L → V/L record	Returned only	Yes	Coupled	Persistent bank
REGENT	S	state → demo state	Returned+blend	No	Coupled	Task buffer
RICL	V	image → demo frame	Returned+blend	No	Coupled	Task buffer
Zeva	V/A	task phase → stored phase	Recurrent keys	No	Coupled	Within episode
RADAR	V/L	eval V/L → train V/L	Policy prior	No	Analysis only	Offline corpus
Physis-Embed	**V/L/A**	**current VLA → future VLA**	**Query + candidate**	**Yes**	**Independent**	**Persistent index**