POST-TRAINING MULTIMODAL EMBEDDING MODELS TOWARDS PHYSICAL DYNAMICS AWARENESS

Anonymous authors
Paper under double-blind review

ABSTRACT

Physical AI needs to connect what is observed with what can happen next, but pixel-level prediction couples learning physical dynamics with reproducing potentially irrelevant appearance details. Predicting learned representations can ease this burden while retaining the state information needed to anticipate change. Embedding models built on multimodal large language models (MLLMs) offer a promising starting point, with our experiments indicating greater utility as physical state encoders than same-size MLLMs. These embedding models combine vision–language processing with reusable representations, yet their relevance-oriented training does not directly target physical transitions. We use soft tokens and learned-query pooling to learn physical state representations alongside a frozen backbone's native retrieval embeddings, then look beyond task-specific adaptation to explore cross-task predictive post-training of a shared, dynamics-aware encoder. Frozen-backbone adaptation delivers stronger overall planning performance than LeWorldModel with fewer trainable parameters, while cross-task post-training improves the shared encoder's utility for world modeling, supporting the potential of a dynamics-aware representation foundation.

1 INTRODUCTION

World models support applications ranging from interactive simulation (Bruce et al., 2024) to behavior learning through imagined trajectories (Hafner et al., 2023). When observations are images, a central question is which aspects of the scene the prediction objective should emphasize. Pixel-level targets require a model to account for detailed appearance, including variation that may contribute little to the physical changes of interest (Bardes et al., 2024; Baldassarre et al., 2025). Visual fidelity is therefore an imperfect proxy: faithful observation prediction does not necessarily produce states useful for downstream control (Hansen et al., 2023; Zhang et al., 2020).

Feature prediction can omit incidental appearance from the learning target, so a predictor need not reproduce those details (Bardes et al., 2024; Assran et al., 2025; Baldassarre et al., 2025). This benefit depends on what the encoder retains: compression alone does not ensure that its outputs preserve the state information needed to predict physical change. Pretrained visual features support control (Nair et al., 2022; Ma et al., 2022) and dynamics learning (Zhou et al., 2024; Baldassarre et al., 2025), making encoder choice and adaptation important parts of the physical prediction problem.

Embedding models built on multimodal large language models (MLLMs) combine joint image–language processing with reusable representations learned through relevance-oriented training (Jiang et al., 2024b; Meng et al., 2025; Li et al., 2026). Their image–language interface allows task-relevant input context to shape which visual information the encoder emphasizes. Our experiments support MLLM embedding models as more useful physical state encoders than same-size MLLMs (Section 3). Yet training for retrieval, question answering, and other matching tasks does not explicitly require these representations to retain the distinctions needed to predict the effects of actions. This motivates two questions: can we improve physical prediction for individual tasks by learning how to elicit and read out features from a frozen backbone, and can learning across tasks develop a shared encoder whose representations are more useful for modeling physical dynamics?

We begin by training soft tokens and learned-query pooling jointly with an action-conditioned predictor over a frozen backbone, learning task-specific physical representations while retaining its native retrieval embeddings. This extends task-specific learning from the use of a fixed encoder output

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

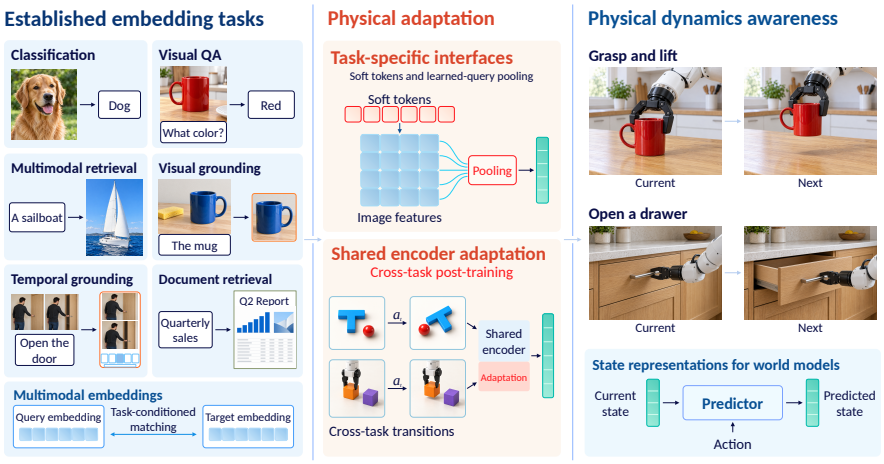


Figure 1: From multimodal matching to physical state representation. Left: representative multimodal embedding tasks formulated as query–target matching. Center: physical adaptation targets how pretrained features are used within a task and what a shared encoder learns across tasks. Right: embodied interactions motivate preserving information for action-conditioned prediction in representation space.

to feature extraction and aggregation. Beyond this specialization, we ask whether physical experience can improve the shared encoder's image representations themselves. We address this question by post-training an encoder on transitions from multiple tasks, aligning current-image embeddings conditioned on state/action descriptions with future-image embeddings. Language provides a common format for heterogeneous controls during post-training. To limit encoder activation memory, we adapt gradient caching (Gao et al., 2021) while retaining full-batch statistics for normalization and regularization. The adapted encoder subsequently supplies image-only representations to task-specific world models. Figure 1 illustrates this shift from established multimodal matching tasks toward physical state representation.

We use physical dynamics awareness to mean the retention of state information useful for predicting action-conditioned change. We assess this through closed-loop visual-goal planning on PushT (Chi et al., 2023), OGBench-Cube (Park et al., 2024), and TwoRoom (Sobal et al., 2025), testing whether representations support both dynamics prediction and goal comparison. Complementary linear probes examine which physical attributes can be predicted directly from the representations. Task-specific adaptation yields stronger overall planning performance with fewer trainable parameters than LeWorldModel (LeWM) (Maes et al., 2026), a strong end-to-end latent world model that rivals foundation-model-based approaches in planning. Beyond these task-specific gains, cross-task post-training improves the shared encoder's utility for downstream world modeling. These findings point toward a representation foundation for physical dynamics awareness.

Our contributions are: (1) We show that the value of MLLM embedding models extends beyond relevance matching to physical state representation. Their advantages over MLLMs suggest that these embeddings offer a promising basis for learning physical dynamics. (2) We develop parameter-efficient adaptation that learns task-specific feature extraction and aggregation over a frozen backbone. This extends physical adaptation beyond a fixed native embedding while retaining the original retrieval function. (3) We extend physical adaptation to shared representation learning through cross-task predictive post-training, supported by gradient caching that limits encoder activation memory while retaining full-batch statistics. The gains in representation utility support the prospect of a representation foundation for physical dynamics awareness.

2 Related Work

Multimodal embedding models. Image–text alignment learns a shared comparison space from large-scale paired data, with contrastive and sigmoid objectives supporting transferable visual fea-

tures and cross-modal retrieval (Radford et al., 2021; Jia et al., 2021; Zhai et al., 2023; Tschannen et al., 2025). Combining alignment with captioning connects representation learning to both vision–language understanding and generation (Li et al., 2022; Yu et al., 2022). Universal and instruction-guided retrieval broaden the matching interface to heterogeneous query–candidate modalities (Wei et al., 2023) and image–language queries that express relations beyond visual similarity (Zhang et al., 2024a). MLLM embedding models bring joint image–language processing to this interface by adapting generative backbones through representation training on text alone (Jiang et al., 2024a) or multimodal inputs (Jiang et al., 2024b; Zhang et al., 2024b; Lin et al., 2024). More recent systems extend instruction-conditioned embeddings across images, videos, and visual documents, unifying classification, question answering, retrieval, and visual or temporal grounding as relevance-matching tasks (Meng et al., 2025; Li et al., 2026).

State representations for world models. World-model states are shaped by reconstructive, decision-oriented, and predictive objectives. Observation reconstruction can accompany latent transition learning for planning (Hafner et al., 2018) and behavior learning through imagined trajectories (Hafner et al., 2023). Reward-, value-, and policy-based objectives instead retain distinctions relevant to decision making without requiring pixel reconstruction (Schrittwieser et al., 2019; Zhang et al., 2020; Hansen et al., 2022; 2023). A complementary line obtains transferable visual encoders before control learning through masked reconstruction (Xiao et al., 2022; Majumdar et al., 2023), temporal supervision (Nair et al., 2022) and value-based supervision (Ma et al., 2022), or language-conditioned reconstruction (Karamcheti et al., 2023) and multimodal value learning (Ma et al., 2023). Feature prediction replaces pixel targets with learned representations, using masked image and video objectives to acquire visual structure without reconstructing the observations (Assran et al., 2023; Bardes et al., 2024; Assran et al., 2025). For action-conditioned dynamics, pretrained spatial features can serve directly as prediction targets (Zhou et al., 2024; Baldassarre et al., 2025). The prediction space can also be adapted through compact states aligned with proprioception and visual features (Fu et al., 2026), or learned jointly with the predictor through future-embedding prediction and distributional regularization (Maes et al., 2026).

Foundation models for physical interaction. Embodied multimodal models connect sensory observations with language knowledge to produce perception-grounded high-level plans (Driess et al., 2023). Generalist robot policies learn from heterogeneous robot experience and support adaptation across tasks and embodiments (Open X-Embodiment Collaboration et al., 2023; Octo Model Team et al., 2024). Vision–language–action models integrate pretrained vision–language backbones with tokenized action prediction (Brohan et al., 2023; Kim et al., 2024) or continuous action prediction (Black et al., 2024). Mixed-source training further combines robot trajectories with web-derived supervision or human video to connect broad visual knowledge with motor control (Physical Intelligence et al., 2025; NVIDIA, 2025b). Video prediction provides another route to action: generated visual plans can be translated through inverse dynamics (Du et al., 2023), while video pretraining followed by joint visual and action prediction develops policies that also anticipate future observations (Wu et al., 2023; Cheang et al., 2024; Ye et al., 2026). Generative world models learn controllable future observations using language instructions (Zhang et al., 2026), inferred latent actions (Bruce et al., 2024), or structured physical conditions (NVIDIA, 2025a; Russell et al., 2025), spanning interactive environments, robotics, and driving. Shared-latent approaches combine visual prediction with language supervision to support multiple downstream modalities through a frozen backbone (Wang et al., 2026).

3 Multimodal Embeddings as a Starting Point for World Modeling

An embedding useful for relevance matching is not necessarily a suitable state representation for physical prediction. For example, two images can both match a description of an object while differing in its position or contact with another object, so the same control can produce different outcomes. A state representation for physical prediction must therefore preserve these distinctions, even when they do not affect relevance to the description. We compare MLLM embedding models (Jiang et al., 2024b; Li et al., 2026) with the MLLMs on which they are built to examine whether representations learned for matching provide a stronger starting point for physical dynamics modeling.

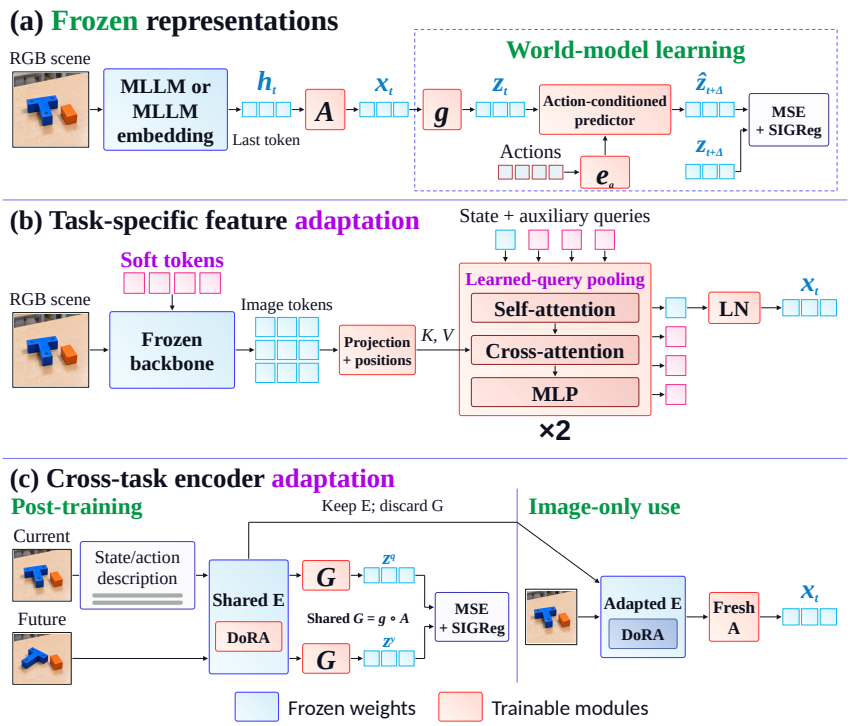


Figure 2: From frozen representations to feature and encoder adaptation. (a) Frozen last-token embeddings with an action-conditioned predictor. (b) Task-specific feature adaptation with soft tokens and learned-query pooling. (c) Cross-task predictive post-training and subsequent use of the adapted encoder. The downstream outputs x_t in (b) and (c) continue through g as in (a). Teal identifies the trainable state query and its selected output.

3.1 Method

We compare pretrained representations using a common action-conditioned predictor and training objective (Maes et al., 2026), as illustrated in Figure 2(a). For an RGB observation o_t, the frozen encoder produces a last-token representation $h_t = E_\theta(o_t) \in \mathbb{R}^D$. A learned affine map $A : \mathbb{R}^D \to \mathbb{R}^d$ connects this output to the downstream model, followed by its nonlinear projector g:

$$x_t = A(h_t), \qquad z_t = g(x_t). \tag{1}$$

This interface decouples the world model's state dimension from the encoder's output width, allowing different backbones to be paired with a common downstream architecture. The state z_t is used for future-state prediction and for comparing predicted outcomes with visual goals. The predictor receives visual states and numeric controls, without proprioceptive or simulator-state input.

The backbone remains frozen while A, g, the action encoder, and the predictor are trained jointly. Training combines a future-state prediction loss with SIGReg (Balestriero & LeCun, 2025), which regularizes the learned states to prevent collapse. Appendix B.1 specifies the predictor and objective.

3.2 Main Results

We compare Qwen3-VL-Instruct (Bai et al., 2025) and Qwen3-VL-Embedding (Li et al., 2026) at matching 2B and 8B scales on PushT (Chi et al., 2023), OGBench-Cube (Cube) (Park et al., 2024), and TwoRoom (Sobal et al., 2025). Within each task, downstream modules for both model families are trained from a fresh initialization using the same architecture and training budget, then evaluated under the same planning protocol. We also re-evaluate LeWM's public models (Maes et al., 2026) under the same planning seeds and budgets. Their original three-frame context is retained, whereas our TwoRoom models use one frame. Appendices A.1–A.3 provide the dataset, training, and planning details.

We pair closed-loop visual-goal planning with linear probes to assess both downstream control utility and the physical attributes accessible in the representations. Probes use native representations before world-model projection. We report overall standardized-target MSE and Pearson r, averaging each metric across a task's physical state dimensions. Appendix C.1 gives the extraction points and full probe protocol.

Table 1: Planning success with frozen last-token representations and re-evaluated LeWM (%, mean $\pm$ std). LeWM is re-evaluated to match our planning settings.

Model	PushT (↑)	TwoRoom (↑)	Cube (↑)	Average (↑)
Qwen3-VL-2B-Instruct	61.33 ± 3.06	$\mathbf{96.00 \pm 3.46}$	70.67 ± 6.43	76.00
Qwen3-VL-Embedding-2B	$\mathbf{75.33 \pm 1.15}$	92.67 ± 4.16	$\mathbf{74.00 \pm 8.00}$	**80.67**
Qwen3-VL-8B-Instruct	76.67 ± 2.31	$\mathbf{96.67 \pm 3.06}$	77.33 ± 5.03	83.56
Qwen3-VL-Embedding-8B	$\mathbf{79.33 \pm 2.31}$	$\mathbf{96.67 \pm 3.06}$	$\mathbf{78.00 \pm 4.00}$	**84.67**
LeWM (re-evaluated)	91.33 ± 3.06	84.00 ± 4.00	78.67 ± 6.43	84.67

Table 2: Overall physical-state probes of pretrained representations (MSE and Pearson r). We fit new probes on LeWM's public encoder features to match the data splits and regression protocol across models.

Model	Dim.	PushT		TwoRoom		Cube	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-2B-Instruct	2048	0.148	0.920	0.006	0.997	**0.548**	**0.482**
Qwen3-VL-Embedding-2B	2048	**0.120**	**0.937**	**0.003**	**0.999**	0.555	0.466
Qwen3-VL-8B-Instruct	4096	0.114	0.940	0.003	**0.999**	0.539	0.491
Qwen3-VL-Embedding-8B	4096	**0.090**	**0.953**	**0.002**	**0.999**	**0.518**	**0.514**
LeWM	192	0.047	0.976	0.031	0.985	0.487	0.523

Finding I: MLLM embedding models offer a stronger starting point for world modeling than MLLMs. Qwen3-VL-Embedding achieves stronger overall planning performance than Qwen3-VL-Instruct at both model sizes, as shown in Table 1. On PushT and Cube, where the Instruct models fall short of LeWM, the embedding models narrow the performance gap at both sizes. On TwoRoom, the embedding models retain the Instruct models' advantage over the LeWM baseline, despite a modest decline for the 2B embedding model relative to the same-size Instruct model. Linear probes in Table 2 also favor embeddings on PushT and TwoRoom for both model sizes, and on Cube for the 8B models. On Cube, Qwen3-VL-Embedding-2B's planning advantage over Qwen3-VL-2B-Instruct in Table 1 is accompanied by more accurate linear prediction of gripper opening, a key state variable for grasping and releasing objects, as shown in Table 15 in Appendix C.2, despite weaker overall probe scores. These results support using relevance-trained embeddings as a foundation for world modeling, with useful physical information already present before task-specific feature adaptation. This motivates learning how to extract and aggregate their features to emphasize the state distinctions needed to model a task's dynamics. Tables 13–15 in Appendix C.2 give the complete property-level probes.

4 Adapting a Frozen Backbone with Soft Tokens and Pooling

The native embedding provides a useful starting point for physical modeling, but need not capture the full potential of the pretrained backbone. Even with fixed weights, its image representations depend on the input context. Learning this context from a physical task can therefore adapt feature extraction, rather than only how a downstream model uses a fixed encoder output. We combine this

contextual adaptation with trainable feature aggregation for parameter-efficient physical specialization while retaining the backbone's original retrieval function.

4.1 METHOD

Figure 2(b) illustrates the architecture and trainable components of our task-specific adaptation. For each task, we condition the frozen backbone with $S + 1$ soft tokens (Jia et al., 2022) inserted immediately before the first image token. Both the vision tower and language-model weights remain frozen, but gradients pass through the language model to the soft tokens. The resulting contextualized image-token representations are $P_t \in \mathbb{R}^{N \times D}$, where N is the number of image tokens.

Learned-query pooling replaces the last-token projection A in Eq. (1), using Q-Former-style attention blocks (Li et al., 2023) to aggregate P_t into x_t. The module first projects the image tokens to width d and adds learned positional embeddings, yielding $\widetilde{P}_t \in \mathbb{R}^{N \times d}$. Its query sequence contains one state query and Q auxiliary queries, all learned vectors in $\mathbb{R}^d$:

$$\begin{cases} U_t^{(0)} = [q_{\text{state}}; q_1; \ldots; q_Q], \\ U_t^{(\ell+1)} = \mathcal{B}_\ell(U_t^{(\ell)}, \widetilde{P}_t), \qquad \ell = 0, \ldots, L-1, \\ x_t = \text{LN}(U_{t,0}^{(L)}). \end{cases} \tag{2}$$

Here L is the number of attention blocks $\mathcal{B}_\ell$, and LN denotes layer normalization. The output at index zero corresponds to the state query. The auxiliary queries participate in attention but are not returned as separate states. The pooled representation x_t passes through the projector g to form the state $z_t = g(x_t)$ used by the action-conditioned predictor. Both the soft tokens and pooling module are trained jointly with the downstream modules under the world-model objective. Appendix B.3 specifies the architecture and token-count conventions. Full-batch normalization and SIGReg statistics are maintained across the training implementations described in Appendix B.6.

The backbone can also be used in its original retrieval configuration, with its pretrained input and output conventions and without the task-specific soft tokens or pooling. This produces the native retrieval embedding in a separate encoding pass, while sharing the same backbone weights with physical-state encoding.

4.2 MAIN RESULTS

Table 3: Native last-token references and frozen-backbone adaptation: planning success (%, mean ± std) and trainable parameters (M). LeWM is re-evaluated to match our planning settings. Parameter counts use PushT and exclude frozen weights. LeWM is counted from the same public checkpoint used for evaluation.

Configuration	Trainable (M)	PushT (↑)	TwoRoom (↑)	Cube (↑)	Average (↑)
Qwen3-VL-Embedding-2B					
Last-token baseline	–	75.33 ± 1.15	92.67 ± 4.16	74.00 ± 8.00	80.67
$S = Q = 2$	14.131	92.67 ± 1.15	92.00 ± 4.00	$\mathbf{82.67 \pm 3.06}$	89.11
$S = Q = 4$	14.135	$\mathbf{95.33 \pm 1.15}$	94.67 ± 2.31	$\mathbf{82.67 \pm 5.03}$	**90.89**
$S = Q = 8$	14.144	94.00 ± 3.46	93.33 ± 3.06	78.00 ± 7.21	88.44
$S = Q = 32$	14.198	94.67 ± 1.15	$\mathbf{95.33 \pm 2.31}$	$\mathbf{82.67 \pm 5.03}$	**90.89**
Qwen3-VL-Embedding-8B					
Last-token baseline	–	79.33 ± 2.31	$\mathbf{96.67 \pm 3.06}$	78.00 ± 4.00	84.67
$S = Q = 2$	14.530	90.67 ± 3.06	92.67 ± 3.06	75.33 ± 8.33	86.22
$S = Q = 4$	14.539	90.00 ± 3.46	94.00 ± 4.00	$\mathbf{80.67 \pm 8.08}$	88.22
$S = Q = 8$	14.556	$\mathbf{95.33 \pm 4.16}$	92.67 ± 3.06	77.33 ± 6.43	88.44
$S = Q = 32$	14.659	92.67 ± 3.06	93.33 ± 3.06	80.00 ± 2.00	**88.67**
LeWM (re-evaluated)	18.034	91.33 ± 3.06	84.00 ± 4.00	78.67 ± 6.43	84.67

We evaluate the 2B and 8B Qwen3-VL-Embedding backbones with $S = Q \in \{2, 4, 8, 32\}$, jointly varying the soft-token and query counts. Table 3 reports the native last-token references alongside the adapted configurations, with planning success and trainable parameters shown against LeWM. Linear probes assess all these configurations using the protocol in Appendix C.1. Appendix A.4 specifies the released-model evaluation.

Finding II: Soft tokens and learned-query pooling enable parameter-efficient physical adaptation of frozen backbones. Table 3 shows that, across both backbone scales and all token budgets, jointly trained soft tokens and learned-query pooling improve aggregate planning relative to the native last-token references, demonstrating more effective use of the same frozen backbone for physical modeling. Relative to the end-to-end LeWM baseline, they achieve stronger aggregate performance while using 18.7–21.6% fewer trainable parameters. Table 16 in Appendix C.3 shows that complementary linear probes achieve lower MSE and higher Pearson r than LeWM on PushT and TwoRoom for every adapted configuration at the common output width of 192. On Cube, despite weaker aggregate probe scores than LeWM's, the adapted representations retain competitive planning performance with fewer trainable parameters. Tables 17–19 in Appendix C.3 provide the complete property-level results and adaptation configurations. Together, these results show that soft tokens and learned-query pooling can equip a frozen multimodal embedding backbone with task-specific physical representations while retaining its native retrieval function.

4.3 Ablation Study

To assess whether combining soft tokens with learned-query pooling improves planning beyond either component alone, we compare the full configuration with pooling-only and soft-token-only variants on Qwen3-VL-Embedding-2B. The pooling-only variant removes all soft tokens, while the soft-token-only variant retains the last-token representation and projection from Section 3.

Table 4: Soft-token and learned-query pooling ablations on Qwen3-VL-Embedding-2B: planning success (%, mean ± std). Soft tokens and pooling use $S = 32$ and $Q = 32$ when present.

Configuration	PushT (↑)	TwoRoom (↑)	Cube (↑)	Average (↑)
Qwen3-VL-Embedding-2B				
Last-token baseline	75.33 ± 1.15	92.67 ± 4.16	74.00 ± 8.00	80.67
Learned-query pooling only	91.33 ± 3.06	$\mathbf{95.33 \pm 1.15}$	78.67 ± 3.06	88.44
Soft tokens only	86.67 ± 1.15	92.00 ± 3.46	74.00 ± 9.17	84.22
Soft tokens + learned-query pooling	$\mathbf{94.67 \pm 1.15}$	$\mathbf{95.33 \pm 2.31}$	$\mathbf{82.67 \pm 5.03}$	**90.89**

Table 4 shows that soft-token conditioning and learned-query pooling deliver complementary gains, with their combination achieving the strongest overall planning performance. The full configuration improves on soft-token-only adaptation across all three tasks, highlighting the value of learned queries for constructing physical states from task-conditioned image features. Relative to pooling alone, it improves planning on PushT and Cube while maintaining TwoRoom performance, supporting soft-token conditioning of the features supplied to learned-query pooling.

5 Cross-Task Predictive Post-Training towards a Dynamics-Aware Representation Foundation

Task-specific adaptation turns pretrained features into representations suited to individual physical tasks. A dynamics-aware representation foundation would extend this benefit by allowing experience from multiple tasks to improve the encoder they share. Although scenes and controls vary, interactions have a common predictive structure: a current state and an action are linked to an observed outcome. These relationships provide supervision beyond relevance matching, encouraging the encoder to retain information about how actions change the world. We therefore pursue cross-task predictive post-training, using heterogeneous physical experience to develop a shared representational basis for downstream world models.

5.1 METHOD

We use the encoder's image–language capabilities to learn from interaction descriptions while retaining image-only state encoding for downstream use. As illustrated in Figure 2(c), recorded actions and their current-state context condition the current-image branch, whereas the future-image branch receives no such descriptions. The branches share the encoder, allowing the predictive objective to shape the image representations subsequently used by world models.

Each training item consists of a current image o_t, a deterministic description c_t, and the observed image $o_{t+\Delta}$ from the same episode, where Δ is the interval between observations. The description summarizes the current simulator state and compresses the intervening controls into qualitative attributes. Future states, rewards, success labels, and planning goals are excluded from the description. Figures 3, 4, and 5 in Appendix B.5 show real image pairs and their stored descriptions.

We initialize the shared encoder E from the original pretrained embedding model used in Section 3. A shared temporary projection G maps its last-token outputs into the space where the two branches are aligned:

$$z^q = G(E(o_t, c_t)), \qquad z^y = G(E(o_{t+\Delta})), \tag{3}$$

where superscripts q and y denote the conditioned current-image and future-image branches, respectively. The encoder performs the conditioned mapping directly, without a separate predictor in this stage. We fine-tune the encoder's language-model component with DoRA (Liu et al., 2024) and jointly train G, using gradients from both branches. The objective combines mean-squared error between the branch outputs with SIGReg (Balestriero & LeCun, 2025), which regularizes each branch to prevent collapse. Appendix B.4 gives the complete objective and implementation details.

Memory-efficient training. Retaining encoder activations for an entire batch is memory-intensive. However, computing the projection and loss independently within smaller chunks changes the batch statistics used by BatchNorm and SIGReg, altering the training objective. We therefore adapt gradient caching (Gao et al., 2021) to separate chunked encoder execution from full-batch projection and loss computation. We first encode both branches in chunks without retaining computation graphs, then evaluate G and the loss on the gathered representations. Gradients with respect to the encoder outputs are cached and propagated through both branches by recomputing one encoder chunk at a time. Thus, chunk size governs encoder activation storage, while normalization and regularization retain their full statistical batch. Appendix B.6 specifies the distributed implementation and its use across our experiments.

5.2 MAIN RESULTS

We post-train on the full transition datasets from PushT, TwoRoom, and Cube with equal task weighting. After post-training, we discard the temporary projection G and freeze the adapted encoder. We train each task's downstream modules from scratch on the encoder's last-token embeddings, retaining the architecture, image-encoding interface, and training budget of the unadapted baseline in Section 3. During downstream training and planning, the encoder receives images alone, while numeric controls condition the predictor. Table 10 in Appendix B.4 and Table 8 in Appendix A.2 provide the dataset and training details, respectively. We also evaluate out-of-distribution (OOD) transfer to PointMaze (Zhou et al., 2024), an environment excluded from encoder post-training (Appendix A.5). Tables 20–22 in Appendix C.4 provide the complete property-level probes.

Finding III: Language-conditioned predictive post-training integrates heterogeneous physical experience into a stronger shared encoder. Table 5 shows that jointly learning from different tasks' visual transitions and state/action descriptions improves overall planning performance for both Qwen3-VL-Embedding models. Cross-task post-training strengthens manipulation performance on PushT and Cube, where the original encoders trail LeWM, while preserving their established navigation advantage on TwoRoom even after the 8B model's modest decline. Both post-trained encoders also improve planning on PointMaze, indicating that the representation gains extend beyond the environments used for post-training. Complementary probes in Table 6 show lower overall MSE and higher Pearson r on PushT and TwoRoom for both post-trained models relative to their original encoders, indicating better access to physical-state information. On Cube, both models' planning gains

Table 5: Planning success before and after cross-task post-training (%, mean ± std). †: post-trained encoder. LeWM is re-evaluated to match our planning settings.

Model	Post-training environments				OOD transfer
	PushT (↑)	TwoRoom (↑)	Cube (↑)	Average (↑)	PointMaze (↑)
Qwen3-VL-Embedding-2B	75.33 ± 1.15	92.67 ± 4.16	74.00 ± 8.00	80.67	90.67 ± 1.15
Qwen3-VL-Embedding-2B†	$\mathbf{82.67 \pm 3.06}$	$\mathbf{96.00 \pm 0.00}$	$\mathbf{78.00 \pm 7.21}$	**85.56**	$\mathbf{93.33 \pm 6.11}$
Qwen3-VL-Embedding-8B	79.33 ± 2.31	$\mathbf{96.67 \pm 3.06}$	78.00 ± 4.00	84.67	92.00 ± 0.00
Qwen3-VL-Embedding-8B†	$\mathbf{87.33 \pm 3.06}$	94.67 ± 2.31	$\mathbf{84.00 \pm 6.00}$	**88.67**	$\mathbf{94.00 \pm 6.00}$
LeWM (re-evaluated)	91.33 ± 3.06	84.00 ± 4.00	78.67 ± 6.43	84.67	–

Table 6: Overall physical-state probes before and after cross-task post-training (MSE and Pearson r). †: post-trained encoder. We fit new probes on LeWM's public encoder features to match the data splits and regression protocol across models.

Model	Dim.	PushT		TwoRoom		Cube	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B	2048	0.120	0.937	0.003	0.999	0.555	**0.466**
Qwen3-VL-Embedding-2B†	2048	**0.074**	**0.961**	**0.000**	**1.000**	**0.523**	0.463
Qwen3-VL-Embedding-8B	4096	0.090	0.953	0.002	0.999	**0.518**	**0.514**
Qwen3-VL-Embedding-8B†	4096	**0.051**	**0.974**	**0.000**	**1.000**	0.520	0.468
LeWM	192	0.047	0.976	0.031	0.985	0.487	0.523

accompany more accurate linear prediction of end-effector position, gripper opening, and block position, key state variables for locating, grasping, and placing objects, despite mixed aggregate probe results, as shown in Tables 21 and 22 in Appendix C.4. Together, the planning and probe results show that experience from different state and control spaces can jointly improve an encoder under a common language-conditioned predictive objective. This supports the prospect of a dynamics-aware representation foundation learned from heterogeneous physical interactions.

5.3 Ablation Study

We ablate the post-training descriptions on Qwen3-VL-Embedding-2B to distinguish the roles of state and action information. Table 23 in Appendix D shows that full state and action descriptions yield stronger overall planning performance than state descriptions or image pairs alone, supporting the use of interaction descriptions to guide representation adaptation. The appendix provides the settings and analysis, including overall probes in Table 24.

6 Conclusion

MLLM embedding models offer more for physical modeling than their native retrieval-oriented outputs alone reveal. With soft tokens and learned-query pooling, a frozen backbone can support parameter-efficient physical specialization alongside its original retrieval function. Beyond this task-specific specialization, language-conditioned predictive post-training allows experience from distinct state and control spaces to improve the shared encoder itself. These findings motivate developing multimodal embeddings that connect relevance matching with the anticipation of physical change, toward a dynamics-aware representation foundation.

AI USE STATEMENT

We used generative AI tools to assist with manuscript refinement, literature discovery, and code development. For writing, the tools suggested revisions to improve clarity and readability, which the human authors reviewed and decided whether to adopt. AI-assisted searches helped broaden the coverage of related work and reduce the risk of overlooking relevant studies. The authors evaluated the retrieved literature, selected the references, and synthesized the related-work discussion. For implementation, AI tools assisted with code scaffolding and debugging, with the resulting code reviewed by the authors. The authors take full responsibility for the final content of this paper, including all AI-assisted material.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-Supervised Learning from Images with a Joint-Embedding Predictive Architecture. *arXiv preprint arXiv:2301.08243*, 2023. URL https://arxiv.org/abs/2301.08243.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-JEPA 2: Self-Supervised Video Models Enable Understanding, Prediction and Planning. *arXiv preprint arXiv:2506.09985*, 2025. URL https://arxiv.org/abs/2506.09985.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-VL Technical Report. *arXiv preprint arXiv:2511.21631*, 2025. URL https://arxiv.org/abs/2511.21631.

Federico Baldassarre, Marc Szafraniec, Basile Terver, Vasil Khalidov, Francisco Massa, Yann LeCun, Patrick Labatut, Maximilian Seitzer, and Piotr Bojanowski. Back to the Features: DINO as a Foundation for Video World Models. *arXiv preprint arXiv:2507.19468*, 2025. URL https://arxiv.org/abs/2507.19468.

Randall Balestriero and Yann LeCun. LeJEPA: Provable and Scalable Self-Supervised Learning Without the Heuristics. *arXiv preprint arXiv:2511.08544*, 2025. URL https://arxiv.org/abs/2511.08544.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting Feature Prediction for Learning Visual Representations from Video. *arXiv preprint arXiv:2404.08471*, 2024. URL https://arxiv.org/abs/2404.08471.

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

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu,

Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. *arXiv preprint arXiv:2307.15818*, 2023. URL https://arxiv.org/abs/2307.15818.

Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal Behbahani, Stephanie Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative Interactive Environments. *arXiv preprint arXiv:2402.15391*, 2024. URL https://arxiv.org/abs/2402.15391.

Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, Hanbo Zhang, and Minzhao Zhu. GR-2: A Generative Video-Language-Action Model with Web-Scale Knowledge for Robot Manipulation. *arXiv preprint arXiv:2410.06158*, 2024. URL https://arxiv.org/abs/2410.06158.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. *arXiv preprint arXiv:2303.04137*, 2023. URL https://arxiv.org/abs/2303.04137.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. PaLM-E: An Embodied Multimodal Language Model. *arXiv preprint arXiv:2303.03378*, 2023. URL https://arxiv.org/abs/2303.03378.

Yilun Du, Mengjiao Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Joshua B. Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning Universal Policies via Text-Guided Video Generation. *arXiv preprint arXiv:2302.00111*, 2023. URL https://arxiv.org/abs/2302.00111.

Minghao Fu, Fan Feng, Nicklas Hansen, and Biwei Huang. Back to Parsimonious Latents: Learning Task-Centric World Models from Visual Foundations. *arXiv preprint arXiv:2605.25620*, 2026. URL https://arxiv.org/abs/2605.25620.

Luyu Gao, Yunyi Zhang, Jiawei Han, and Jamie Callan. Scaling Deep Contrastive Learning Batch Size under Memory Limited Setup. *arXiv preprint arXiv:2101.06983*, 2021. URL https://arxiv.org/abs/2101.06983.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning Latent Dynamics for Planning from Pixels. *arXiv preprint arXiv:1811.04551*, 2018. URL https://arxiv.org/abs/1811.04551.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering Diverse Domains through World Models. *arXiv preprint arXiv:2301.04104*, 2023. URL https://arxiv.org/abs/2301.04104.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal Difference Learning for Model Predictive Control. *arXiv preprint arXiv:2203.04955*, 2022. URL https://arxiv.org/abs/2203.04955.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, Robust World Models for Continuous Control. *arXiv preprint arXiv:2310.16828*, 2023. URL https://arxiv.org/abs/2310.16828.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. *arXiv preprint arXiv:2106.09685*, 2021. URL https://arxiv.org/abs/2106.09685.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yunhsuan Sung, Zhen Li, and Tom Duerig. Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision. *arXiv preprint arXiv:2102.05918*, 2021. URL https://arxiv.org/abs/2102.05918.

Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual Prompt Tuning. *arXiv preprint arXiv:2203.12119*, 2022. URL https://arxiv.org/abs/2203.12119.

Ting Jiang, Minghui Song, Zihan Zhang, Haizhen Huang, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, and Fuzhen Zhuang. E5-V: Universal Embeddings with Multimodal Large Language Models. *arXiv preprint arXiv:2407.12580*, 2024a. URL https://arxiv.org/abs/2407.12580.

Ziyan Jiang, Rui Meng, Xinyi Yang, Semih Yavuz, Yingbo Zhou, and Wenhu Chen. VLM2Vec: Training Vision-Language Models for Massive Multimodal Embedding Tasks. *arXiv preprint arXiv:2410.05160*, 2024b. URL https://arxiv.org/abs/2410.05160.

Siddharth Karamcheti, Suraj Nair, Annie S. Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-Driven Representation Learning for Robotics. *arXiv preprint arXiv:2302.12766*, 2023. URL https://arxiv.org/abs/2302.12766.

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

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation. *arXiv preprint arXiv:2201.12086*, 2022. URL https://arxiv.org/abs/2201.12086.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models. In *Proceedings of the 40th International Conference on Machine Learning*, 2023. URL https://proceedings.mlr.press/v202/li23q.html.

Mingxin Li, Yanzhao Zhang, Dingkun Long, Keqin Chen, Sibo Song, Shuai Bai, Zhibo Yang, Pengjun Xie, An Yang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Qwen3-VL-Embedding and Qwen3-VL-Reranker: A Unified Framework for State-of-the-Art Multimodal Retrieval and Ranking. *arXiv preprint arXiv:2601.04720*, 2026. URL https://arxiv.org/abs/2601.04720.

Sheng-Chieh Lin, Chankyu Lee, Mohammad Shoeybi, Jimmy Lin, Bryan Catanzaro, and Wei Ping. MM-Embed: Universal Multimodal Retrieval with Multimodal LLMs. *arXiv preprint arXiv:2411.02571*, 2024. URL https://arxiv.org/abs/2411.02571.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. DoRA: Weight-Decomposed Low-Rank Adaptation. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. URL https://proceedings.mlr.press/v235/liu24bn.html.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. *arXiv preprint arXiv:1711.05101*, 2017. URL https://arxiv.org/abs/1711.05101.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. VIP: Towards Universal Visual Reward and Representation via Value-Implicit Pre-Training. *arXiv preprint arXiv:2210.00030*, 2022. URL https://arxiv.org/abs/2210.00030.

Yecheng Jason Ma, William Liang, Vaidehi Som, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. LIV: Language-Image Representations and Rewards for Robotic Control. *arXiv preprint arXiv:2306.00958*, 2023. URL https://arxiv.org/abs/2306.00958.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. LeWorld-Model: Stable End-to-End Joint-Embedding Predictive Architecture from Pixels. *arXiv preprint arXiv:2603.19312*, 2026. URL https://arxiv.org/abs/2603.19312v3. Version 3.

Arjun Majumdar, Karmesh Yadav, Sergio Arnaud, Yecheng Jason Ma, Claire Chen, Sneha Silwal, Aryan Jain, Vincent-Pierre Berges, Pieter Abbeel, Jitendra Malik, Dhruv Batra, Yixin Lin, Oleksandr Maksymets, Aravind Rajeswaran, and Franziska Meier. Where are we in the search for an Artificial Visual Cortex for Embodied Intelligence? *arXiv preprint arXiv:2303.18240*, 2023. URL https://arxiv.org/abs/2303.18240.

Rui Meng, Ziyan Jiang, Ye Liu, Mingyi Su, Xinyi Yang, Yuepeng Fu, Can Qin, Zeyuan Chen, Ran Xu, Caiming Xiong, Yingbo Zhou, Wenhu Chen, and Semih Yavuz. VLM2Vec-V2: Advancing Multimodal Embedding for Videos, Images, and Visual Documents. *arXiv preprint arXiv:2507.04590*, 2025. URL https://arxiv.org/abs/2507.04590.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3M: A Universal Visual Representation for Robot Manipulation. *arXiv preprint arXiv:2203.12601*, 2022. URL https://arxiv.org/abs/2203.12601.

NVIDIA. Cosmos World Foundation Model Platform for Physical AI. *arXiv preprint arXiv:2501.03575*, 2025a. URL https://arxiv.org/abs/2501.03575.

NVIDIA. GR00T N1: An Open Foundation Model for Generalist Humanoid Robots. *arXiv preprint arXiv:2503.14734*, 2025b. URL https://arxiv.org/abs/2503.14734.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An Open-Source Generalist Robot Policy. *arXiv preprint arXiv:2405.12213*, 2024. URL https://arxiv.org/abs/2405.12213.

Open X-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, Albert Tung, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anchit Gupta, Andrew Wang, Andrey Kolobov, Anikait Singh, Animesh Garg, Aniruddha Kembhavi, Annie Xie, Anthony Brohan, Antonin Raffin, Archit Sharma, Arefeh Yavary, Arhan Jain, Ashwin Balakrishna, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Blake Wulfe, Brian Ichter, Cewu Lu, Charles Xu, Charlotte Le, Chelsea Finn, Chen Wang, Chenfeng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Christopher Agia, Chuer Pan, Chuyuan Fu, Coline Devin, Danfei Xu, Daniel Morton, Danny Driess, Daphne Chen, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dinesh Jayaraman, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Ethan Foster, Fangchen Liu, Federico Ceola, Fei Xia, Feiyu Zhao, Felipe Vieira Frujeri, Freek Stulp, Gaoyue Zhou, Gaurav S. Sukhatme, Gautam Salhotra, Ge Yan, Gilbert Feng, Giulio Schiavi, Glen Berseth, Gregory Kahn, Guangwen Yang, Guanzhi Wang, Hao Su, Hao-Shu Fang, Haochen Shi, Henghui Bao, Heni Ben Amor, Henrik I Christensen, Hiroki Furuta, Homanga Bharadhwaj, Homer Walke, Hongjie Fang, Huy Ha, Igor Mordatch, Ilija Radosavovic, Isabel Leal, Jacky Liang, Jad Abou-Chakra, Jaehyung Kim, Jaimyn Drake, Jan Peters, Jan Schneider, Jasmine Hsu, Jay Vakil, Jeannette Bohg, Jeffrey Bingham, Jeffrey Wu, Jensen Gao, Jiaheng Hu, Jiajun Wu, Jialin Wu, Jiankai Sun, Jianlan Luo, Jiayuan Gu, Jie Tan, Jihoon Oh, Jimmy Wu, Jingpei Lu, Jingyun Yang, Jitendra Malik, João Silvério, Joey Hejna, Jonathan Booher, Jonathan Tompson, Jonathan Yang, Jordi Salvador, Joseph J. Lim, Junhyek Han, Kaiyuan Wang, Kanishka Rao, Karl Pertsch, Karol Hausman, Keegan Go, Keerthana Gopalakrishnan, Ken Goldberg, Kendra Byrne, Kenneth Oslund, Kento Kawaharazuka, Kevin Black, Kevin Lin, Kevin Zhang, Kiana Ehsani, Kiran Lekkala, Kirsty Ellis, Krishan Rana, Krishnan Srinivasan, Kuan Fang, Kunal Pratap Singh, Kuo-Hao Zeng, Kyle Hatch, Kyle Hsu, Laurent Itti, Lawrence Yunliang Chen, Lerrel Pinto, Li Fei-Fei, Liam Tan, Linxi "Jim" Fan, Lionel Ott, Lisa Lee, Luca Weihs, Magnum Chen, Marion Lepert, Marius Memmel, Masayoshi Tomizuka, Masha Itkina,

Mateo Guaman Castro, Max Spero, Maximilian Du, Michael Ahn, Michael C. Yip, Mingtong Zhang, Mingyu Ding, Minho Heo, Mohan Kumar Srirama, Mohit Sharma, Moo Jin Kim, Muhammad Zubair Irshad, Naoaki Kanazawa, Nicklas Hansen, Nicolas Heess, Nikhil J Joshi, Niko Suenderhauf, Ning Liu, Norman Di Palo, Nur Muhammad Mahi Shafiullah, Oier Mees, Oliver Kroemer, Osbert Bastani, Pannag R Sanketi, Patrick "Tree" Miller, Patrick Yin, Paul Wohlhart, Peng Xu, Peter David Fagan, Peter Mitrano, Pierre Sermanet, Pieter Abbeel, Priya Sundaresan, Qiuyu Chen, Quan Vuong, Rafael Rafailov, Ran Tian, Ria Doshi, Roberto Martín-Martín, Rohan Baijal, Rosario Scalise, Rose Hendrix, Roy Lin, Runjia Qian, Ruohan Zhang, Russell Mendonca, Rutav Shah, Ryan Hoque, Ryan Julian, Samuel Bustamante, Sean Kirmani, Sergey Levine, Shan Lin, Sherry Moore, Shikhar Bahl, Shivin Dass, Shubham Sonawani, Shubham Tulsiani, Shuran Song, Sichun Xu, Siddhant Haldar, Siddharth Karamcheti, Simeon Adebola, Simon Guist, Soroush Nasiriany, Stefan Schaal, Stefan Welker, Stephen Tian, Subramanian Ramamoorthy, Sudeep Dasari, Suneel Belkhale, Sungjae Park, Suraj Nair, Suvir Mirchandani, Takayuki Osa, Tanmay Gupta, Tatsuya Harada, Tatsuya Matsushima, Ted Xiao, Thomas Kollar, Tianhe Yu, Tianli Ding, Todor Davchev, Tony Z. Zhao, Travis Armstrong, Trevor Darrell, Trinity Chung, Vidhi Jain, Vikash Kumar, Vincent Vanhoucke, Vitor Guizilini, Wei Zhan, Wenxuan Zhou, Wolfram Burgard, Xi Chen, Xiangyu Chen, Xiaolong Wang, Xinghao Zhu, Xinyang Geng, Xiyuan Liu, Xu Liangwei, Xuanlin Li, Yansong Pang, Yao Lu, Yecheng Jason Ma, Yejin Kim, Yevgen Chebotar, Yifan Zhou, Yifeng Zhu, Yilin Wu, Ying Xu, Yixuan Wang, Yonatan Bisk, Yongqiang Dou, Yoonyoung Cho, Youngwoon Lee, Yuchen Cui, Yue Cao, Yueh-Hua Wu, Yujin Tang, Yuke Zhu, Yunchu Zhang, Yunfan Jiang, Yunshuang Li, Yunzhu Li, Yusuke Iwasawa, Yutaka Matsuo, Zehan Ma, Zhuo Xu, Zichen Jeff Cui, Zichen Zhang, Zipeng Fu, and Zipeng Lin. Open X-Embodiment: Robotic Learning Datasets and RT-X Models. *arXiv preprint arXiv:2310.08864*, 2023. URL https://arxiv.org/abs/2310.08864.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. OGBench: Benchmarking Offline Goal-Conditioned RL. *arXiv preprint arXiv:2410.20092*, 2024. URL https://arxiv.org/abs/2410.20092.

William Peebles and Saining Xie. Scalable Diffusion Models with Transformers. *arXiv preprint arXiv:2212.09748*, 2022. URL https://arxiv.org/abs/2212.09748.

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

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. *arXiv preprint arXiv:2103.00020*, 2021. URL https://arxiv.org/abs/2103.00020.

Lloyd Russell, Anthony Hu, Lorenzo Bertoni, George Fedoseev, Jamie Shotton, Elahe Arani, and Gianluca Corrado. GAIA-2: A Controllable Multi-View Generative World Model for Autonomous Driving. *arXiv preprint arXiv:2503.20523*, 2025. URL https://arxiv.org/abs/2503.20523.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model. *arXiv preprint arXiv:1911.08265*, 2019. URL https://arxiv.org/abs/1911.08265.

Vlad Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, Tim G. J. Rudner, and Yann LeCun. Learning from Reward-Free Offline Data: A Case for Planning with Latent Dynamics Models. *arXiv preprint arXiv:2502.14819*, 2025. URL https://arxiv.org/abs/2502.14819.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, Olivier Hénaff, Jeremiah Harmsen, Andreas Steiner, and Xiaohua Zhai. SigLIP 2: Multilingual Vision-Language Encoders with Improved Semantic Understanding, Localization, and Dense Features. *arXiv preprint arXiv:2502.14786*, 2025. URL https://arxiv.org/abs/2502.14786.

Yihao Wang, Yuheng Ji, Mingyu Cao, Yanqing Shen, Runze Xiao, Huaihai Lyu, Senwei Xie, Euan Liu, Klara Tian, Tianfeng Long, Yichi Zhang, Zhengliang Cai, Ruike Chen, Jifan Zhao, Ruochuan Shi, Zihan Tang, Jing Lyu, Wenxing Tan, Ningbo Zhang, Yangtao Hu, Yuming Gao, Xiansheng Chen, Junkai Zhao, Congsheng Xu, Boan Zhu, Ziqi Wang, Yupu Feng, Qiongqiong Zhang, Yingli Zhao, Yulong Ao, Shaoxuan Xie, You Liu, Guocai Yao, Leiduo Zhang, Xiaodan Liu, Yunyan Zhang, Yance Jiao, Xinyan Yang, Jiaxing Wei, Xu Liu, Tengfei Pan, Shaokai Nie, Chunlei Men, Sen Cui, Xiaojie Jin, Hongyang Li, Jianlan Luo, Yao Mu, Yunchao Wei, Jun Yan, Hang Zhao, Xiaolong Zheng, Jiaming Li, Yonghua Lin, Tiejun Huang, Zhongyuan Wang, and Pengwei Wang. Orca: The World is in Your Mind. *arXiv preprint arXiv:2606.30534*, 2026. URL https://arxiv.org/abs/2606.30534.

Cong Wei, Yang Chen, Haonan Chen, Hexiang Hu, Ge Zhang, Jie Fu, Alan Ritter, and Wenhu Chen. UniIR: Training and Benchmarking Universal Multimodal Information Retrievers. *arXiv preprint arXiv:2311.17136*, 2023. URL https://arxiv.org/abs/2311.17136.

Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing Large-Scale Video Generative Pre-training for Visual Robot Manipulation. *arXiv preprint arXiv:2312.13139*, 2023. URL https://arxiv.org/abs/2312.13139.

Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked Visual Pre-training for Motor Control. *arXiv preprint arXiv:2203.06173*, 2022. URL https://arxiv.org/abs/2203.06173.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, Ayaan Malik, Kyungmin Lee, William Liang, Nadun Ranawaka, Jiasheng Gu, Yinzhen Xu, Guanzhi Wang, Fengyuan Hu, Avnish Narayan, Johan Bjorck, Jing Wang, Gwanghyun Kim, Dantong Niu, Ruijie Zheng, Yuqi Xie, Jimmy Wu, Qi Wang, Ryan Julian, Danfei Xu, Yilun Du, Yevgen Chebotar, Scott Reed, Jan Kautz, Yuke Zhu, Linxi "Jim" Fan, and Joel Jang. World Action Models are Zero-shot Policies. *arXiv preprint arXiv:2602.15922*, 2026. URL https://arxiv.org/abs/2602.15922.

Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. CoCa: Contrastive Captioners are Image-Text Foundation Models. *arXiv preprint arXiv:2205.01917*, 2022. URL https://arxiv.org/abs/2205.01917.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid Loss for Language Image Pre-Training. *arXiv preprint arXiv:2303.15343*, 2023. URL https://arxiv.org/abs/2303.15343.

Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning Invariant Representations for Reinforcement Learning without Reconstruction. *arXiv preprint arXiv:2006.10742*, 2020. URL https://arxiv.org/abs/2006.10742.

Jie Zhang, Xiaoyue Chen, Anzhe Chen, Dayiheng Liu, Deqing Li, Gengze Zhou, Hale Yin, Haoqi Yuan, Haoyang Li, Jiahao Li, Jiazhao Zhang, Jingren Zhou, Kaiyuan Gao, Kun Yan, Lihan Jiang, Ningyuan Tang, Pei Lin, Qihang Peng, Shengming Yin, Tianhe Wu, Tianyi Yan, Xiao Xu, Yan Shu, Yanran Zhang, Ye Wang, Yi Wang, Yilei Chen, Yixian Xu, Yiyang Huang, Yuxiang Chen, Zekai Zhang, Zhendong Wang, Zixing Lei, Zhixuan Liang, Zihao Liu, Zikai Zhou, Chenxu Lv, Xiong-Hui Chen, and Chenfei Wu. Qwen-RobotWorld Technical Report: Unifying Embodied World Modeling through Language-Conditioned Video Generation. *arXiv preprint arXiv:2606.17030*, 2026. URL https://arxiv.org/abs/2606.17030.

Kai Zhang, Yi Luan, Hexiang Hu, Kenton Lee, Siyuan Qiao, Wenhu Chen, Yu Su, and Ming-Wei Chang. MagicLens: Self-Supervised Image Retrieval with Open-Ended Instructions. *arXiv preprint arXiv:2403.19651*, 2024a. URL https://arxiv.org/abs/2403.19651.

Xin Zhang, Yanzhao Zhang, Wen Xie, Mingxin Li, Ziqi Dai, Dingkun Long, Pengjun Xie, Meishan Zhang, Wenjie Li, and Min Zhang. GME: Improving Universal Multimodal Retrieval by Multimodal LLMs. *arXiv preprint arXiv:2412.16855*, 2024b. URL https://arxiv.org/abs/2412.16855.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World Models on Pre-trained Visual Features enable Zero-shot Planning. *arXiv preprint arXiv:2411.04983*, 2024. URL https://arxiv.org/abs/2411.04983.

A Experimental Setup

A.1 Datasets and observations

We use the interaction datasets for PushT, OGBench-Cube (Cube), and TwoRoom provided by Maes et al. (2026). Cube originates from OGBench (Park et al., 2024), and PushT follows the pushing environment used by Diffusion Policy (Chi et al., 2023). PushT requires moving a planar T-shaped block to a target pose. Cube requires controlling a gripper to reach a target cube configuration. TwoRoom (Sobal et al., 2025) tests navigation through a doorway between two rooms. All three environments use RGB observations and dataset-derived visual goals.

For these three environments, downstream training windows are randomly divided using a fixed seed, with 90% assigned to the training pool. Planning starts and goal images are sampled from the task datasets, and performance is measured through closed-loop action execution in the environment (Maes et al., 2026). Appendix A.5 specifies the separate PointMaze transfer protocol.

A.2 Image encoding and optimization

Table 7: Representation and adaptation configurations.

Study	Representation adaptation	Downstream representation
Pretrained representations	None	Native last-token output
Frozen-backbone adaptation	Task-specific soft tokens and learned-query pooling	Pooled image-token features
Cross-task post-training	Shared encoder adapted across tasks	Adapted encoder's last-token output

Table 7 summarizes the representation configurations used throughout the paper.

Table 8 separates shared downstream settings from the cross-task post-training settings. Both stages use AdamW (Loshchilov & Hutter, 2017).

Images are resized to 224×224 pixels. The fixed image-encoding instruction uses the system message *"Represent the user's input."* A user message containing an image and a single space follows. The native chat template includes its generation prefix. Last-token features are taken from the last non-padding position, without appending an extra EOS token. The model-specific template used to generate training features is retained at evaluation. The extracted representations are used at their full backbone width without unit normalization.

The native encoder width is $D = 2048$ for the 2B models and $D = 4096$ for the 8B models. All downstream states have width $d = 192$. The last-token projection A is a learned affine map $D \to 192$. Learned-query pooling instead projects and aggregates the image tokens into a 192-dimensional representation x_t. In both cases, the projector g maps x_t to the state z_t.

A.3 Closed-loop planning

Table 9 follows the task-specific control settings in Appendices D and F of LeWM (Maes et al., 2026), arXiv version 3. All three tasks use a goal offset of 25 and an evaluation budget of 50 raw steps. Each planning seed determines the sampled starts and random planning choices. All comparisons within a study reuse the same seed list. The numeric action dimension is two for PushT and TwoRoom and five for Cube. A five-step block therefore has dimension 10 or 25. Action normalization is fitted to the source dataset's action column before splitting its eligible training windows. The corresponding dataset statistics are used for control as well.

A goal image is encoded into z_g. The world model rolls out candidate control sequences autoregressively, and the cross-entropy method (CEM) planner minimizes the terminal squared distance $\|\widehat{z}_{t+K\Delta} - z_g\|_2^2$ over a $K = 5$-block horizon. New observations close the loop after action execution. Success is the fraction of episodes satisfying the environment's goal-achievement criterion.

Table 8: Downstream world-model training on PushT, TwoRoom, and Cube, and cross-task post-training settings.

Setting	Downstream training	Cross-task post-training
Global batch	128 sequences	128 pairs from one task
Training budget	10 epochs	52,575 steps
Optimizer	AdamW	AdamW
Peak learning rate	5×10^{-5}	5×10^{-6}
Weight decay	10^{-3}	10^{-3}
Learning-rate schedule	1% linear warmup, cosine decay	1% linear warmup, cosine decay
Gradient clipping	Norm 1.0	Norm 1.0
Training-data fraction	90% of eligible windows	Full transition dataset
Latent width	192	192
Projector g	Linear–BN–GELU–Linear, $192 \to 2048 \to 192$	Same widths, shared across branches
Prediction projection g_p	$192 \to 2048 \to 192$, with BN and GELU	None
Temporal Transformer F	Six causal Transformer blocks; 16 heads, head width 64; MLP 2048	None
Transformer dropout	0.1	Not applicable
Action conditioning	Learned control-block embedding, zero-initialized adaptive LayerNorm	Qualitative state and action descriptions
SIGReg weight	0.09	0.09
SIGReg sampling	17 knots, 1024 directions	17 knots, 1024 directions
Backbone adaptation	Frozen weights, task-specific soft tokens where present	DoRA on language model, vision tower frozen
DoRA settings	Not applicable	Rank 16, alpha 64, dropout 0.1
Adapter locations	Not applicable	Attention query, key, value, and output projections; MLP gating, expansion, and output projections

Table 9: Planning settings for our models on PushT, TwoRoom, and Cube. Offsets and budgets count raw environment steps.

Parameter	PushT	TwoRoom	Cube
Context length H	3	1	3
Observation stride Δ	5	5	5
Goal offset	25	25	25
Evaluation budget	50	50	50
CEM iterations	30	10	10
Planning horizon (blocks)	5	5	5
Receding horizon (blocks)	5	5	5
Actions per block	5	5	5
Candidates per CEM iteration	300	300	300
Elites per iteration	30	30	30
Initial CEM variance parameter	1	1	1
CEM evaluation batch size	1	1	1
Episodes per evaluation seed	50	50	50

All models are evaluated with three planning seeds and 50 episodes per seed. We report means and standard deviations across these evaluations, using the same final ten-epoch checkpoint for each of our configurations. The Average column is the equally weighted mean of the unrounded PushT, TwoRoom, and Cube means, using the same model and adaptation configuration across tasks.

A.4 Public LeWM evaluation

We re-evaluate the public LeWM models (Maes et al., 2026) for PushT, TwoRoom, and Cube without retraining or selecting among checkpoints. These are the same models used for the linear probes in Appendix C.1. Evaluation uses the same script, evaluation seeds, 50 episodes per seed, and the planning budgets and CEM settings in Table 9. All three released predictors and their rollouts retain a three-frame context, including TwoRoom, where our models use one frame. Images are encoded online. All models use BF16 inference.

A.5 PointMaze transfer

PointMaze tests whether an encoder adapted on PushT, TwoRoom, and Cube provides useful representations for a new environment. The OOD transfer group in Table 5 refers to an environment excluded from encoder post-training. We use 1,600 trajectories (160,000 frames) from the PointMaze image/action dataset released by Zhou et al. (2024). The target-environment training data also supply the action-normalization statistics. The predictor uses three observations spaced five environment steps apart.

Planning uses the native PointMaze environment with a fixed U-shaped maze and randomly sampled initial and goal states. Every model receives the same cases under different evaluation seeds, with 50 cases per seed. CEM uses 300 candidates, 10 iterations, and 30 elites. The planning and receding horizons are both five blocks of five actions, giving 25 environment steps between replanning updates. An episode succeeds when the Euclidean distance to the goal position is below 0.5, including cases that satisfy this criterion at initialization. Table 5 reports success within 50 environment steps, computed from the first successful step in the saved trajectories. PointMaze results are reported separately from the three-environment Average.

B Implementation Details

B.1 Downstream predictor and training objective

The state interface in Eq. (1) is paired with an action-conditioned predictor (Maes et al., 2026). We sample observations every $\Delta = 5$ raw steps. For raw control a_t, the associated control block $u_t =$

$[a_t; \ldots; a_{t+\Delta-1}]$ is encoded by the action encoder e_a. The action-conditioned predictor consists of a causal temporal Transformer F and a prediction projection g_p. The Transformer receives the H-observation window $\mathbf{z}_t = (z_{t-(H-1)\Delta}, \ldots, z_t)$, with the corresponding control blocks $\mathbf{u}_t = (u_{t-(H-1)\Delta}, \ldots, u_t)$ encoded positionwise by e_a. Controls condition the Transformer through zero-initialized adaptive LayerNorm (Peebles & Xie, 2022). The prediction projection g_p maps the final temporal output back to the state space:

$$\widehat{z}_{t+\Delta} = g_p(F(\mathbf{z}_t, e_a(\mathbf{u}_t))_H)\,. \tag{4}$$

The subscript H selects the final position. Training supervises the next sampled state at every position of the context window. The predictor receives visual states and encoded control blocks, with no additional proprioceptive or simulator-state input. Table 8 gives the architecture.

The state mapping (the last-token projection A or learned-query pooling) and projector g are learned jointly with the predictor, even when the backbone is frozen. To regularize these trainable states, the downstream objective (Maes et al., 2026) combines future-state prediction error with SIGReg (Balestriero & LeCun, 2025):

$$\mathcal{L}_{\mathrm{WM}} = \frac{1}{BHd}\sum_{b=1}^{B}\sum_{i=0}^{H-1}\|\widehat{z}_{b,i+1} - z_{b,i+1}\|_2^2 + \frac{\lambda}{H+1}\sum_{i=0}^{H}\mathcal{R}_{\mathrm{SIG}}(Z_i), \tag{5}$$

where b indexes sequences, i indexes sampled times within a sequence, $Z_i \in \mathbb{R}^{B\times d}$ stacks the projected states at position i, and B is the global batch size. SIGReg compares the distributions of random one-dimensional projections with a standard Gaussian, and λ controls its contribution. Gradients flow through both prediction and target projections. BatchNorm in the projector operates over the flattened global sequence batch, whereas SIGReg keeps time positions separate. All downstream settings use $B = 128$ and $\lambda = 0.09$. Appendix B.2 defines the statistic.

B.2 SIGReg implementation

For a matrix $Z \in \mathbb{R}^{B\times d}$, sample independent directions $v_j \sim \mathcal{N}(0, I_d)$ and normalize each to unit length, for $j = 1, \ldots, J$. Define projected samples $u_{bj} = z_b^\top v_j$ and the empirical characteristic function

$$\widehat{\varphi}_j(s) = \frac{1}{B}\sum_{b=1}^{B}\exp(\mathrm{i}su_{bj}), \qquad \varphi_0(s) = \exp(-s^2/2). \tag{6}$$

Here s is the real-valued Fourier argument and i is the imaginary unit. The 17 fixed integration knots are $s_k = 3k/16$ for $k = 0, \ldots, 16$. With $\delta = 3/16$, the implementation uses

$$w_k = \varphi_0(s_k)\begin{cases}\delta, & k \in \{0, 16\},\\ 2\delta, & \text{otherwise},\end{cases} \tag{7}$$

$$\mathcal{R}_{\mathrm{SIG}}(Z) = \frac{B}{J}\sum_{j=1}^{J}\sum_{k=0}^{16} w_k\left[(\mathrm{Re}\,\widehat{\varphi}_j(s_k) - \varphi_0(s_k))^2 + (\mathrm{Im}\,\widehat{\varphi}_j(s_k))^2\right], \quad J = 1024. \tag{8}$$

This is the discretized SIGReg statistic (Balestriero & LeCun, 2025) used in the experiments, including the quadrature scaling of the reference implementation (Maes et al., 2026). Random directions are shared across distributed ranks. The statistic uses all 128 examples for each time position or branch; averaging independently computed per-device statistics is not equivalent.

B.3 Soft tokens and learned-query pooling

With 224-pixel square images, the frozen-backbone adaptation extracts $N = 49$ contextualized image tokens from the language model. Its learned-query pooling uses $L = 2$ attention blocks, each with eight heads and MLP width 768. Each block applies query self-attention, cross-attention to the image tokens, and an MLP with residual connections. The soft-token prefix contains $S + 1$ learned tokens. The pooling module uses $Q + 1$ learned queries, one designated as the state query. Soft tokens and pooling queries are separate parameter sets with different roles. The state-query output passes through LayerNorm to form x_t and then through g to form z_t. The auxiliary-query outputs are not averaged into x_t.

The pooling-only ablation removes the entire soft-token prefix and uses $Q = 32$. The soft-token-only ablation uses $S = 32$, retaining the $S + 1$ prefix tokens and extracting the hidden state at the last non-padding position. Prefix tokens are never used as the returned representation. Each variant matches the retained component of the full $S = Q = 32$ configuration.

Trainable parameter counts. Table 3 reports the total trainable parameters for frozen-backbone adaptation, excluding the backbone weights. The count includes soft tokens, learned-query pooling, the projector, the action encoder, and the predictor with its prediction projection. For LeWM (Maes et al., 2026), we count the parameters in the public models used for planning and linear probes, including the visual encoder and all world-model modules before evaluation-time freezing. Counts exclude non-parameter buffers. These models contain 18,034,478 trainable parameters on PushT and TwoRoom, and 18,034,628 on Cube, with their original three-frame context retained. Table 3 uses the PushT counts for both methods, expressed in millions ($M = 10^6$). For our models, Cube adds 150 parameters because of its control dimension, and TwoRoom with $H = 1$ removes 384 temporal-position parameters. Both changes are below 0.001M. Soft-token training still requires backbone forward computation and backpropagation through the language model.

B.4 Cross-task predictive post-training

Post-training uses the full transition datasets. Each valid pair contains images from the same episode separated by $\Delta = 5$ raw steps. The description summarizes the current simulator state and the five intervening controls. The future-image branch uses the fixed image-encoding instruction specified in Appendix A.2, without a state or action description. Appendix B.5 details the description construction.

The language-model adapter uses DoRA (Liu et al., 2024), a weight-decomposed extension of LoRA (Hu et al., 2021). Its rank, target modules, and optimization settings are given in Table 8. The vision tower and original backbone weights remain frozen, while the adapter and shared temporary projection are trained. The projection in Eq. (3) is $G = g \circ A$, where the affine map A reduces the native embedding width D to $d = 192$ and the MLP g has widths $192 \to 2048 \to 192$, with BatchNorm and GELU. During training, G processes the stacked current- and future-image embeddings, so its BatchNorm statistics use both branches jointly.

For a batch of B pairs, let z_b^q and z_b^y denote the outputs in Eq. (3) for pair b, and let $Z^q, Z^y \in \mathbb{R}^{B\times d}$ stack the outputs of each branch. We combine mean-squared alignment error with branchwise SIGReg (Balestriero & LeCun, 2025):

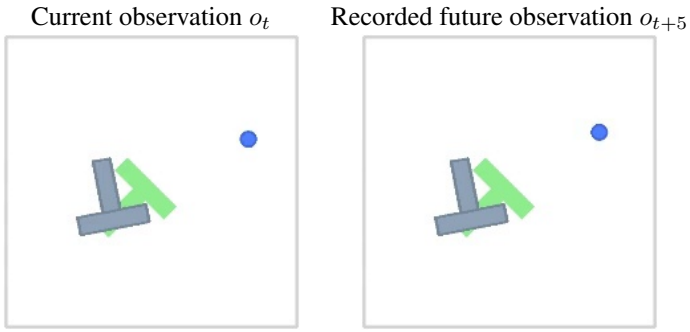
$$\mathcal{L}_{\text{joint}} = \frac{1}{dB}\sum_{b=1}^{B} \|z_b^q - z_b^y\|_2^2 + \frac{\lambda}{2}\left[\mathcal{R}_{\text{SIG}}(Z^q) + \mathcal{R}_{\text{SIG}}(Z^y)\right]. \tag{9}$$

We use $\lambda = 0.09$, matching the downstream regularization weight. Appendix B.2 specifies the SIGReg statistic, and Appendix B.6 describes the distributed execution of this objective.

Table 10: Available transition pairs and sampled pairs in one balanced post-training epoch.

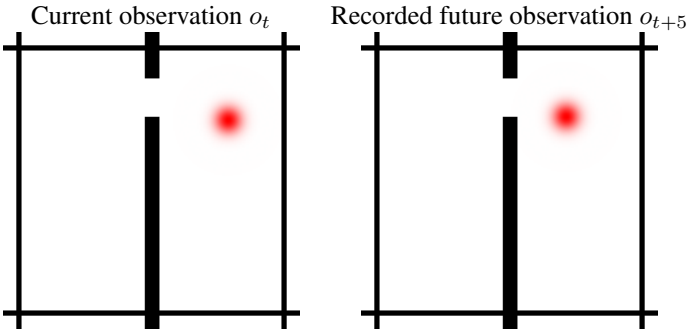
Task	Available pairs	Sampled pairs
PushT	2,243,311	2,243,200
OGBench-Cube	1,960,000	2,243,200
TwoRoom	870,809	2,243,200
Total	5,074,120	6,729,600

Tasks alternate in a fixed round-robin schedule. Each global batch contains 128 pairs from one task, and smaller datasets are resampled to match the largest task's step count. Table 10 reports the available and sampled pairs, and Table 8 specifies the optimization budget.



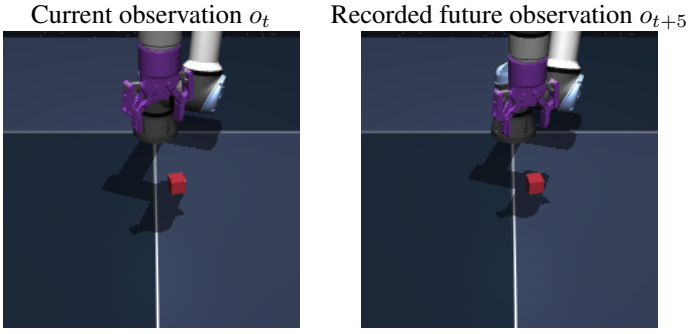
State and action description: From the top-down Push-T view, the circular pusher starts right and above relative to the center of the T-shaped block. The stem of the T points upward from its crossbar. The circular pusher is commanded to move very slightly left and upward on average over this interval. Retrieve the next Push-T observation after this action interval.

Figure 3: PushT: a training example with a current observation, a state and action description, and a recorded future observation. The common system instruction is omitted from the display.



State and action description: From the top-down TwoRoom view, the agent starts in the right room, away from the nearest doorway and below the doorway. The agent is commanded to move moderately left on average over this interval. Retrieve the next TwoRoom observation after this motion.

Figure 4: TwoRoom: a training example with a current observation, a state and action description, and a recorded future observation. The common system instruction is omitted from the display.



State and action description: Using the tabletop axes of the front view, the gripper pinch point starts on the near side and above relative to the cube center. The robot gripper is commanded to move slightly toward the far edge of the table and right and downward on average over this interval. It has a net clockwise wrist-turn command when viewed from above and applies a gripper opening command. Retrieve the next cube observation after this action interval.

Figure 5: Cube: a training example with a current observation, a state and action description, and a recorded future observation. The common system instruction is omitted from the display.

B.5 State and action descriptions

Figures 3, 4, and 5 present training examples. Each shows the current frame o_t, its description, and the recorded future frame o_{t+5}. Descriptions summarize the mean five-step control and current-state geometry. Multiple control sequences can therefore share a description, while each training item uses its recorded future observation as the target. The descriptions express commanded motion rather than measured future displacement, with realized effects depending on contact and obstacles.

Two description lengths alternate deterministically by sample index. The longer form adds context about possible contact or obstacles. It does not inspect the target frame. For PushT and Cube, directional components and magnitude categories are derived from the control block in a fixed coordinate system. For Cube, translation, wrist rotation, and gripper commands are described separately.

B.6 Gradient caching with global-batch statistics

Cross-task post-training uses gradient caching (Gao et al., 2021) to evaluate Eq. (9) over the full batch while processing the encoder in chunks of eight examples. The cache boundary is the native last-token representation, before the trainable projection G. This keeps the batch-dependent projection and loss outside the chunked encoder computation. The frozen vision tower's features are cached before training.

Each optimizer step has three stages:

1. Encode both branches in chunks without retaining computation graphs. Store the native last-token representations and each chunk's random state, using the current DoRA parameters.

2. Gather the representations across devices and apply G jointly to both branches. BatchNorm uses all $2B = 256$ representations, while each SIGReg term uses the $B = 128$ representations in its own branch. Compute the full loss, the gradients of G, and the gradients with respect to both branches' encoder outputs.

3. Restore each chunk's random state and recompute its encoder outputs with autograd. Backpropagate the inner product of these outputs and their cached gradients, treating the cached gradients as constants. After both branches have been replayed, reduce parameter gradients across devices and take one optimizer step.

Each device evaluates the same global projection and loss. SIGReg's random directions are sampled on one device and broadcast to the others. Each device replays only its local examples, so DoRA gradients are summed across devices, whereas the duplicated global projection gradients are averaged. Both branches contribute to the encoder update, without target stop-gradient or a moving-average encoder. Replaying the encoder does not repeat G or update its BatchNorm running statistics a second time.

This procedure retains encoder activations for one chunk at a time, while storing full-batch representations and projection activations. Recomputation exchanges additional encoder work for reduced activation storage. The per-step representation-gradient cache is distinct from the persistent frozen-feature cache and from activation checkpointing within the encoder.

Table 11 distinguishes the execution paths used in the three studies. Full adaptation refers to soft tokens and learned-query pooling together, across all four budgets. Backpropagation through the frozen backbone in these configurations updates the soft tokens, not the backbone weights. Local pooling processes each sample separately, with zero dropout and no BatchNorm, and the frozen backbone remains in evaluation mode. The locally computed pooling gradients are summed across devices because each device processes only its local samples. Subsequent global modules use averaged gradients, as above. All paths compute BatchNorm and SIGReg with the full statistical batch.

Table 11: Feature caching and gradient computation across training configurations. Cached features are produced by frozen modules.

Configuration	Cached features	Gradient computation
Frozen or post-trained encoder, downstream training	Native last-token outputs	Downstream modules only
Full adaptation, 2B PushT and Cube	Vision-tower outputs	Encoder gradient caching, global learned-query pooling
Full adaptation, remaining settings	Vision-tower outputs	Direct encoder backpropagation, local learned-query pooling
Soft-token-only ablation	Vision-tower outputs	Direct encoder backpropagation, last-token output
Pooling-only ablation	Contextualized image tokens	Local learned-query pooling, no encoder backpropagation
Cross-task post-training	Vision-tower outputs	Two-branch encoder gradient caching, global projection G

Table 12: Representations used for physical-state linear probes. Paired widths correspond to the 2B / 8B models, respectively.

Representation	Extraction point	Width
Pretrained MLLM / MLLM embedding	Last token, before A	2048 / 4096
Cross-task post-training	Adapted encoder's last token, before A	2048 / 4096
Soft tokens + pooling / pooling only	Pooled output x_t, before g	192
Soft tokens only (2B)	Last token, before A	2048
LeWM	Visual encoder's CLS output, before its projector	192

C PHYSICAL-STATE LINEAR PROBES

C.1 REPRESENTATIONS AND FITTING PROTOCOL

We probe all pretrained, task-adapted, and cross-task post-trained representations evaluated in the PushT, TwoRoom, and Cube planning experiments. Table 12 identifies the representation used for each method. All representation-producing modules are frozen during extraction and probe fitting. The probes operate before the world-model projector g and never use its predictor. Cross-task post-training is evaluated at its final checkpoint, with the temporary projection discarded and the default image-encoding instruction restored. For LeWM (Maes et al., 2026), we extract the native visual features from the same public models used for planning and fit new probes under the protocol below.

For the linear probes, each task has a fixed set of 8,000 training and 2,000 test frames randomly selected from episode-disjoint partitions. All models use the same frames and labels within a task. Within the training set, 20% of represented episodes are randomly selected to form a validation partition. Feature and target dimensions are standardized using only the fitting partition's means and standard deviations, with a standard-deviation floor of 10^{-6}.

A closed-form ridge regression with an unpenalized intercept is fitted in double precision to predict simulator-recorded physical attributes. Its objective is squared residual error divided by the number of fitting samples, plus the ridge coefficient times the squared weight norm. Every model uses the same 11 coefficients $\{10^{-8}, 10^{-7}, \ldots, 10^{2}\}$. We choose one coefficient per representation and task by the lowest validation MSE averaged across standardized target dimensions, resolving exact ties in favor of the larger coefficient. We then recompute standardization statistics on all 8,000 training frames and refit using the selected coefficient. Test labels do not enter fitting, standardization, or coefficient selection.

The target properties follow Maes et al. (2026). PushT has five scalar dimensions: agent position (two), block position (two), and block angle (one). TwoRoom uses the agent's two-dimensional position. Cube has 25 dimensions: joint position (six), joint velocity (six), end-effector position (three), end-effector yaw (one), gripper opening (one), block position (three), block quaternion (four), and block yaw (one). Scalar angles and quaternion components retain their dataset encodings, without periodic or symmetry-aware transformations. Their probe errors therefore also reflect angle discontinuities, equivalent orientation encodings, and visual observability.

MSE uses targets standardized by the final training-set statistics and is therefore dimensionless. MSE and Pearson r (denoted r in the tables) are computed for each scalar target dimension. Property-level scores average over the dimensions of that property, while overall scores average over all state dimensions within a task. Each scalar dimension receives equal weight. Overall Pearson r averages the per-dimension correlations rather than correlating flattened predictions and targets. TwoRoom's sole property is agent position, so its property and overall scores are identical. The single probe split is distinct from the planning-seed protocol in Appendix A.3.

C.2 PRETRAINED REPRESENTATIONS

Tables 13–15 give all properties for the frozen Qwen3-VL-Instruct and Qwen3-VL-Embedding models.

C.3 FROZEN-BACKBONE ADAPTATION

Table 16 reports overall physical-state probes for the native representations and all soft-token and pooling budgets.

Tables 17–19 include every token budget and both single-component configurations from Section 4. Each model group begins with its native last-token reference. All pooling configurations produce 192-dimensional outputs, whereas soft-token-only adaptation retains the 2048-dimensional last-token output. The single-component variants use $Q = 32$ or $S = 32$, as in Table 4.

Table 13: Pretrained representations: PushT and TwoRoom linear probes. We fit new probes on LeWM's public encoder features to match the data splits and regression protocol across models.

Model	Dim.	PushT Agent pos.		PushT Block pos.	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-2B-Instruct	2048	0.059	0.970	0.084	0.959
Qwen3-VL-Embedding-2B	2048	**0.054**	**0.973**	**0.079**	**0.962**
Qwen3-VL-8B-Instruct	4096	0.035	0.982	0.071	0.965
Qwen3-VL-Embedding-8B	4096	**0.030**	**0.985**	**0.053**	**0.974**
LeWM	192	0.040	0.980	0.016	0.993

Model	Dim.	PushT Block angle		TwoRoom Agent pos.	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-2B-Instruct	2048	0.456	0.741	0.006	0.997
Qwen3-VL-Embedding-2B	2048	**0.333**	**0.818**	**0.003**	**0.999**
Qwen3-VL-8B-Instruct	4096	0.359	0.804	0.003	**0.999**
Qwen3-VL-Embedding-8B	4096	**0.284**	**0.848**	**0.002**	**0.999**
LeWM	192	0.125	0.936	0.031	0.985

Table 14: Pretrained representations: Cube robot-state linear probes. We fit new probes on LeWM's public encoder features to match the data splits and regression protocol across models.

Model	Dim.	Joint position		Joint velocity	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-2B-Instruct	2048	**0.376**	**0.687**	**0.949**	**0.175**
Qwen3-VL-Embedding-2B	2048	0.390	0.677	0.950	0.144
Qwen3-VL-8B-Instruct	4096	0.371	0.696	0.929	0.196
Qwen3-VL-Embedding-8B	4096	**0.347**	**0.713**	**0.898**	**0.247**
LeWM	192	0.330	0.703	0.873	0.265

Model	Dim.	End-effector position		End-effector yaw	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-2B-Instruct	2048	**0.082**	**0.959**	1.034	**0.155**
Qwen3-VL-Embedding-2B	2048	0.109	0.948	**1.030**	0.141
Qwen3-VL-8B-Instruct	4096	0.089	0.958	1.003	0.216
Qwen3-VL-Embedding-8B	4096	**0.055**	**0.974**	**0.975**	**0.270**
LeWM	192	0.011	0.995	1.058	0.069

Table 15: Pretrained representations: Cube gripper and block-state linear probes. We fit new probes on LeWM's public encoder features to match the data splits and regression protocol across models.

Model	Dim.	Gripper opening		Block position	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-2B-Instruct	2048	0.267	0.857	**0.105**	**0.947**
Qwen3-VL-Embedding-2B	2048	**0.245**	**0.871**	0.142	0.930
Qwen3-VL-8B-Instruct	4096	0.272	0.855	0.111	0.946
Qwen3-VL-Embedding-8B	4096	**0.224**	**0.883**	**0.092**	**0.955**
LeWM	192	0.047	0.976	0.006	0.997

Model	Dim.	Block quaternion		Block yaw	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-2B-Instruct	2048	0.718	**0.031**	1.026	0.017
Qwen3-VL-Embedding-2B	2048	**0.699**	0.012	**1.002**	**0.023**
Qwen3-VL-8B-Instruct	4096	**0.699**	**0.029**	**1.006**	0.029
Qwen3-VL-Embedding-8B	4096	0.705	0.028	1.012	**0.033**
LeWM	192	0.693	0.068	1.017	-0.015

Table 16: Native and adapted representations: overall physical-state probes (MSE and Pearson r). Native rows retain their encoder width in the Dim. column; adapted rows use the 192-dimensional pooled representation x_t. We fit new probes on LeWM's public encoder features to match the data splits and regression protocol across models.

Configuration	Dim.	PushT		TwoRoom		Cube	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B							
Last-token baseline	2048	0.120	0.937	0.003	0.999	0.555	0.466
$S = Q = 2$	192	**0.043**	**0.978**	0.006	0.997	0.514	0.480
$S = Q = 4$	192	0.046	0.977	0.006	0.997	**0.512**	0.483
$S = Q = 8$	192	**0.043**	**0.978**	**0.005**	**0.998**	**0.512**	0.483
$S = Q = 32$	192	**0.043**	**0.978**	**0.005**	**0.998**	**0.512**	**0.487**
Qwen3-VL-Embedding-8B							
Last-token baseline	4096	0.090	0.953	0.002	0.999	0.518	0.514
$S = Q = 2$	192	0.039	0.980	0.006	0.997	0.511	0.485
$S = Q = 4$	192	0.041	0.979	0.006	0.997	0.514	0.481
$S = Q = 8$	192	0.040	0.980	0.006	0.997	**0.507**	**0.497**
$S = Q = 32$	192	**0.038**	**0.981**	**0.005**	**0.998**	0.512	0.481
LeWM	192	0.047	0.976	0.031	0.985	0.487	0.523

Table 17: Frozen-backbone adaptation: PushT and TwoRoom linear probes. We fit new probes on LeWM's public encoder features to match the data splits and regression protocol across models.

Configuration	Dim.	PushT Agent pos.		PushT Block pos.	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B					
Last-token baseline	2048	0.054	0.973	0.079	0.962
$S = Q = 2$	192	0.026	0.987	0.019	0.991
$S = Q = 4$	192	0.027	0.986	0.018	**0.992**
$S = Q = 8$	192	0.025	0.987	**0.017**	**0.992**
$S = Q = 32$	192	**0.024**	**0.988**	**0.017**	**0.992**
Learned-query pooling only	192	0.023	0.988	0.022	0.990
Soft tokens only	2048	0.015	0.992	0.020	0.991
Qwen3-VL-Embedding-8B					
Last-token baseline	4096	0.030	0.985	0.053	0.974
$S = Q = 2$	192	**0.023**	0.988	**0.014**	**0.993**
$S = Q = 4$	192	0.025	0.988	**0.014**	**0.993**
$S = Q = 8$	192	0.025	0.988	0.017	0.992
$S = Q = 32$	192	**0.023**	**0.989**	0.015	**0.993**
LeWM	192	0.040	0.980	0.016	0.993

Configuration	Dim.	PushT Block angle		TwoRoom Agent pos.	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B					
Last-token baseline	2048	0.333	0.818	0.003	0.999
$S = Q = 2$	192	**0.124**	**0.936**	0.006	0.997
$S = Q = 4$	192	0.138	0.929	0.006	0.997
$S = Q = 8$	192	0.131	0.932	**0.005**	**0.998**
$S = Q = 32$	192	0.135	0.930	**0.005**	**0.998**
Learned-query pooling only	192	0.153	0.921	0.004	0.998
Soft tokens only	2048	0.194	0.899	0.001	0.999
Qwen3-VL-Embedding-8B					
Last-token baseline	4096	0.284	0.848	0.002	0.999
$S = Q = 2$	192	0.119	0.939	0.006	0.997
$S = Q = 4$	192	0.125	0.936	0.006	0.997
$S = Q = 8$	192	0.120	0.939	0.006	0.997
$S = Q = 32$	192	**0.113**	**0.942**	**0.005**	**0.998**
LeWM	192	0.125	0.936	0.031	0.985

Table 18: Frozen-backbone adaptation: Cube robot-state linear probes. We fit new probes on LeWM's public encoder features to match the data splits and regression protocol across models.

Configuration	Dim.	Joint position		Joint velocity	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B					
Last-token baseline	2048	0.390	0.677	0.950	0.144
$S = Q = 2$	192	**0.331**	0.710	0.970	0.104
$S = Q = 4$	192	**0.331**	**0.711**	0.969	0.109
$S = Q = 8$	192	0.332	0.709	0.968	0.109
$S = Q = 32$	192	**0.331**	0.709	**0.967**	**0.126**
Learned-query pooling only	192	0.330	0.711	0.967	0.113
Soft tokens only	2048	0.332	0.716	0.920	0.208
Qwen3-VL-Embedding-8B					
Last-token baseline	4096	0.347	0.713	0.898	0.247
$S = Q = 2$	192	0.329	0.712	0.965	0.118
$S = Q = 4$	192	0.334	0.709	0.968	0.110
$S = Q = 8$	192	**0.325**	**0.723**	**0.961**	**0.128**
$S = Q = 32$	192	0.331	0.711	0.970	0.105
LeWM	192	0.330	0.703	0.873	0.265

Configuration	Dim.	End-effector position		End-effector yaw	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B					
Last-token baseline	2048	0.109	0.948	1.030	0.141
$S = Q = 2$	192	0.019	0.991	1.055	0.062
$S = Q = 4$	192	0.019	0.991	**1.048**	**0.092**
$S = Q = 8$	192	0.020	0.990	1.054	0.069
$S = Q = 32$	192	**0.014**	**0.993**	1.062	0.075
Learned-query pooling only	192	0.018	0.991	1.050	0.085
Soft tokens only	2048	0.029	0.986	1.002	0.221
Qwen3-VL-Embedding-8B					
Last-token baseline	4096	0.055	0.974	0.975	0.270
$S = Q = 2$	192	**0.016**	**0.992**	1.057	0.058
$S = Q = 4$	192	0.022	0.989	1.058	0.055
$S = Q = 8$	192	0.020	0.990	**1.016**	**0.183**
$S = Q = 32$	192	0.018	0.991	1.058	0.056
LeWM	192	0.011	0.995	1.058	0.069

Table 19: Frozen-backbone adaptation: Cube gripper and block-state linear probes. We fit new probes on LeWM's public encoder features to match the data splits and regression protocol across models.

Configuration	Dim.	Gripper opening		Block position	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B					
Last-token baseline	2048	0.245	0.871	0.142	0.930
$S = Q = 2$	192	0.133	0.932	0.009	0.996
$S = Q = 4$	192	0.123	0.937	0.008	0.996
$S = Q = 8$	192	0.110	0.944	0.008	0.996
$S = Q = 32$	192	**0.098**	**0.950**	**0.007**	**0.997**
Learned-query pooling only	192	0.140	0.928	0.011	0.995
Soft tokens only	2048	0.166	0.914	0.028	0.986
Qwen3-VL-Embedding-8B					
Last-token baseline	4096	0.224	0.883	0.092	0.955
$S = Q = 2$	192	0.116	0.941	0.008	0.996
$S = Q = 4$	192	0.121	0.938	0.008	0.996
$S = Q = 8$	192	**0.101**	**0.949**	0.008	0.996
$S = Q = 32$	192	0.120	0.938	**0.007**	**0.997**
LeWM	192	0.047	0.976	0.006	0.997

Configuration	Dim.	Block quaternion		Block yaw	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B					
Last-token baseline	2048	0.699	0.012	1.002	0.023
$S = Q = 2$	192	0.691	0.032	0.999	0.023
$S = Q = 4$	192	0.690	0.038	0.998	0.024
$S = Q = 8$	192	**0.689**	**0.039**	**0.996**	0.031
$S = Q = 32$	192	0.697	0.037	1.003	**0.034**
Learned-query pooling only	192	0.689	0.038	1.004	0.001
Soft tokens only	2048	0.686	0.058	0.998	0.034
Qwen3-VL-Embedding-8B					
Last-token baseline	4096	0.705	0.028	1.012	0.033
$S = Q = 2$	192	**0.688**	0.044	1.001	0.010
$S = Q = 4$	192	0.690	0.036	1.000	0.011
$S = Q = 8$	192	0.689	**0.045**	**0.995**	**0.038**
$S = Q = 32$	192	0.690	0.035	0.996	0.032
LeWM	192	0.693	0.068	1.017	-0.015

Table 20: Cross-task post-training: PushT and TwoRoom linear probes. We fit new probes on LeWM's public encoder features to match the data splits and regression protocol across models.

Model	Dim.	PushT Agent pos.		PushT Block pos.	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B	2048	0.054	0.973	0.079	0.962
Qwen3-VL-Embedding-2B†	2048	**0.019**	**0.991**	**0.022**	**0.990**
Qwen3-VL-Embedding-8B	4096	0.030	0.985	0.053	0.974
Qwen3-VL-Embedding-8B†	4096	**0.012**	**0.994**	**0.014**	**0.993**
LeWM	192	0.040	0.980	0.016	0.993

Model	Dim.	PushT Block angle		TwoRoom Agent pos.	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B	2048	0.333	0.818	0.003	0.999
Qwen3-VL-Embedding-2B†	2048	**0.289**	**0.846**	**0.000**	**1.000**
Qwen3-VL-Embedding-8B	4096	0.284	0.848	0.002	0.999
Qwen3-VL-Embedding-8B†	4096	**0.203**	**0.893**	**0.000**	**1.000**
LeWM	192	0.125	0.936	0.031	0.985

Table 21: Cross-task post-training: Cube robot-state linear probes. We fit new probes on LeWM's public encoder features to match the data splits and regression protocol across models.

Model	Dim.	Joint position		Joint velocity	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B	2048	0.390	0.677	**0.950**	**0.144**
Qwen3-VL-Embedding-2B†	2048	**0.349**	**0.690**	0.974	0.081
Qwen3-VL-Embedding-8B	4096	0.347	**0.713**	**0.898**	**0.247**
Qwen3-VL-Embedding-8B†	4096	**0.340**	0.698	0.975	0.091
LeWM	192	0.330	0.703	0.873	0.265

Model	Dim.	End-effector position		End-effector yaw	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B	2048	0.109	0.948	**1.030**	**0.141**
Qwen3-VL-Embedding-2B†	2048	**0.042**	**0.979**	1.065	0.017
Qwen3-VL-Embedding-8B	4096	0.055	0.974	**0.975**	**0.270**
Qwen3-VL-Embedding-8B†	4096	**0.027**	**0.987**	1.069	0.021
LeWM	192	0.011	0.995	1.058	0.069

Table 22: Cross-task post-training: Cube gripper and block-state linear probes. We fit new probes on LeWM's public encoder features to match the data splits and regression protocol across models.

Model	Dim.	Gripper opening		Block position	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B	2048	0.245	0.871	0.142	0.930
Qwen3-VL-Embedding-2B†	2048	**0.153**	**0.921**	**0.008**	**0.996**
Qwen3-VL-Embedding-8B	4096	0.224	0.883	0.092	0.955
Qwen3-VL-Embedding-8B†	4096	**0.133**	**0.932**	**0.004**	**0.998**
LeWM	192	0.047	0.976	0.006	0.997

Model	Dim.	Block quaternion		Block yaw	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B	2048	0.699	0.012	**1.002**	**0.023**
Qwen3-VL-Embedding-2B†	2048	**0.691**	**0.022**	1.005	-0.001
Qwen3-VL-Embedding-8B	4096	0.705	**0.028**	**1.012**	**0.033**
Qwen3-VL-Embedding-8B†	4096	**0.699**	0.022	1.015	-0.016
LeWM	192	0.693	0.068	1.017	-0.015

C.4 Cross-task post-training

Tables 20–22 compare the original and post-trained Qwen3-VL-Embedding encoders at each model size. Each pair retains the same native last-token width. The symbol † marks the post-trained encoder.

D Post-Training Text Ablation

D.1 Setup

We compare three post-training configurations initialized from Qwen3-VL-Embedding-2B, with the original encoder as the no-post-training reference. *Image pairs only* removes all physical descriptions from the current-image branch. *State descriptions* retains the current-state description but removes action information. *State and action descriptions* is the full method, conditioning on both the current state and intervening controls. All three configurations retain the same current/future image pairs, objective, task weighting, and full-data training budget specified in Appendix B.4. The future-image branch and downstream encoding use the fixed image-encoding instruction in Appendix A.2. Each encoder supplies native 2048-dimensional last-token representations to fresh task-specific downstream modules under the same training and planning settings. Linear probes follow Appendix C.1.

D.2 Results

Tables 23 and 24 show the value of describing interactions beyond improving state readability alone. The full state and action descriptions yield the strongest overall planning performance, exceeding state descriptions alone and image pairs only by 6.22 and 9.11 percentage points, respectively. On PushT, both ablations improve overall probe scores over the original encoder but reduce planning success, whereas the full method improves both. Better access to current-state attributes therefore does not by itself guarantee that predictive post-training improves control.

On Cube, the full method's leading planning result accompanies the most accurate linear prediction of end-effector position, gripper opening, and block position in Tables 26 and 27, despite lower

Table 23: Post-training text ablation: planning success (%, mean ± std).

Configuration	PushT (↑)	TwoRoom (↑)	Cube (↑)	Average (↑)
Qwen3-VL-Embedding-2B				
No post-training	75.33 ± 1.15	92.67 ± 4.16	74.00 ± 8.00	80.67
Image pairs only	61.33 ± 8.08	$\mathbf{96.00 \pm 2.00}$	72.00 ± 6.93	76.44
State descriptions	68.67 ± 5.03	94.00 ± 2.00	75.33 ± 3.06	79.33
State and action descriptions	$\mathbf{82.67 \pm 3.06}$	$\mathbf{96.00 \pm 0.00}$	$\mathbf{78.00 \pm 7.21}$	**85.56**

Table 24: Post-training text ablation: overall physical-state probes (MSE and Pearson r).

Configuration	Dim.	PushT		TwoRoom		Cube	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B							
No post-training	2048	0.120	0.937	0.003	0.999	0.555	0.466
Image pairs only	2048	0.103	0.945	**0.000**	**1.000**	0.534	0.469
State descriptions	2048	0.081	0.957	**0.000**	**1.000**	0.527	**0.473**
State and action descriptions	2048	**0.074**	**0.961**	**0.000**	**1.000**	**0.523**	0.463

overall probe correlation than the other configurations. On TwoRoom, image pairs alone match the full method's planning performance, indicating that the benefit of additional descriptions varies across tasks. Taken together, the ablations support coupling state descriptions with action context when adapting embeddings for world modeling.

D.3 Property-level probes

Table 25: Post-training text ablation: PushT and TwoRoom linear probes.

Configuration	Dim.	PushT Agent pos.		PushT Block pos.	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B					
No post-training	2048	0.054	0.973	0.079	0.962
Image pairs only	2048	0.045	0.977	0.041	0.980
State descriptions	2048	0.040	0.980	**0.022**	0.989
State and action descriptions	2048	**0.019**	**0.991**	**0.022**	**0.990**

Configuration	Dim.	PushT Block angle		TwoRoom Agent pos.	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B					
No post-training	2048	0.333	0.818	0.003	0.999
Image pairs only	2048	0.345	0.810	**0.000**	**1.000**
State descriptions	2048	**0.281**	**0.849**	**0.000**	**1.000**
State and action descriptions	2048	0.289	0.846	**0.000**	**1.000**

Tables 25–27 provide all property-level results for the four configurations, using the same extraction and fitting protocol as the overall probes above.

Table 26: Post-training text ablation: Cube robot-state linear probes.

Configuration	Dim.	Joint position		Joint velocity	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B					
No post-training	2048	0.390	0.677	**0.950**	**0.144**
Image pairs only	2048	0.356	0.693	0.985	0.095
State descriptions	2048	0.357	**0.695**	0.966	0.104
State and action descriptions	2048	**0.349**	0.690	0.974	0.081

Configuration	Dim.	End-effector position		End-effector yaw	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B					
No post-training	2048	0.109	0.948	**1.030**	**0.141**
Image pairs only	2048	0.051	0.975	1.051	0.103
State descriptions	2048	0.057	0.973	1.047	0.084
State and action descriptions	2048	**0.042**	**0.979**	1.065	0.017

Table 27: Post-training text ablation: Cube gripper and block-state linear probes.

Configuration	Dim.	Gripper opening		Block position	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B					
No post-training	2048	0.245	0.871	0.142	0.930
Image pairs only	2048	0.190	0.901	0.021	0.990
State descriptions	2048	0.190	0.901	0.023	0.989
State and action descriptions	2048	**0.153**	**0.921**	**0.008**	**0.996**

Configuration	Dim.	Block quaternion		Block yaw	
		MSE (↓)	r (↑)	MSE (↓)	r (↑)
Qwen3-VL-Embedding-2B					
No post-training	2048	0.699	0.012	1.002	0.023
Image pairs only	2048	0.710	0.013	1.012	0.038
State descriptions	2048	**0.689**	**0.027**	**0.991**	**0.056**
State and action descriptions	2048	0.691	0.022	1.005	-0.001