WORLDPRIOR: INTERNALIZING MULTIMODAL SEMANTICS INTO LATENT WORLD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Latent world models learn compact predictive representations of visual dynamics, but matching future features alone does not guarantee that their hidden states expose the object identity, affordance, interaction phase, relational structure, and contextual cues that determine which futures matter for embodied prediction and control. Vision–language models (VLMs) contain complementary multimodal priors, yet using them online changes the deployment interface and introduces a large inference-time cost. We introduce *WorldPrior*, a training-only adaptation framework that internalizes multimodal semantics into a latent world model while preserving a VLM-free deployment graph. A frozen VLM processes only the observed clip, and detached observation-side representations are aligned with selected internal predictor states through cosine distillation. Because high-level multimodal supervision can perturb fine spatial and dynamical structure, a frozen reference predictor initialized from the same baseline checkpoint acts as a functional anchor, while the backbone-native predictive objective remains active throughout adaptation. All VLM-specific projections, semantic readouts, reference branches, and training-only adaptation objectives are removed at inference. On EgoDex, WorldPrior reduces ADE/FDE from 0.0705/0.0646 to 0.0597/0.0572 and raises threshold accuracy from 0.436 to 0.564. On EgoPAT3D, it obtains the best or tied-best result on all four 3D metrics and three of four 2D metrics across seen and unseen scenes. With LeWM as the control backbone, average success increases from 85.5% to 90.0%; the same principle transfers to DINO-WM, improving the matched four-task average from 83.0% to 85.5%. WorldPrior retains the inference graph and practical latency of the corresponding VLM-free backbone, whereas our online VLM-guidance reproduction is more than $22\times$ slower in forecasting. These results support a broader view of multimodal foundation models as training-time representational priors that can strengthen perception, relational understanding, and task-relevant inference in latent world models without becoming deployment-time dependencies.

1 INTRODUCTION

Embodied agents require predictive representations that capture not only how observations evolve, but also which entities, relations, and interaction states determine the consequences of an action. World models compress observations into latent states for forecasting, planning, and decision making (Ha & Schmidhuber, 2018; Hafner et al., 2020), while joint-embedding predictive architectures (JEPAs) avoid explicit pixel generation by predicting directly in representation space (Assran et al., 2023; Bardes et al., 2024). V-JEPA2 further shows that large-scale self-supervised video pretraining can support motion understanding and physical planning (Assran et al., 2025).

A central limitation is that *predictive accuracy does not uniquely determine representational structure*. Feature prediction can preserve local visual dynamics while leaving object identity, affordance, interaction phase, or action–object relations weakly organized in the hidden state. In embodied settings, these distinctions are consequential: visually similar motion may imply different futures depending on contact and affordance, and a small error in a decision-critical relation can change task success even when average latent discrepancy remains modest. We therefore treat the gap between latent prediction and downstream competence partly as a representation problem.

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.

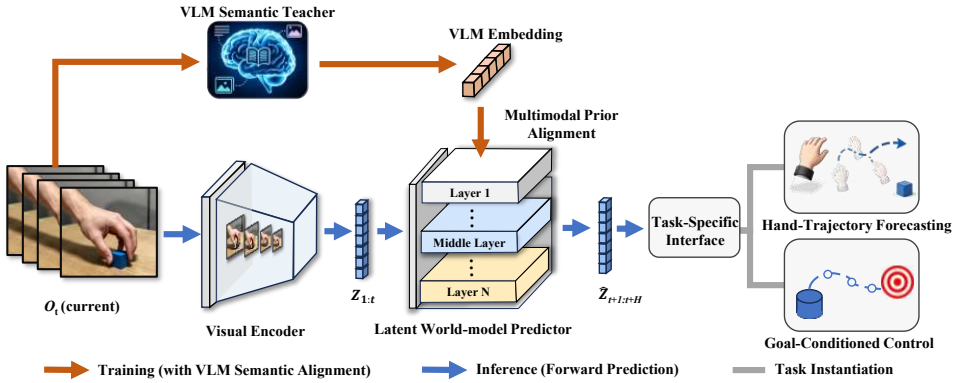


Figure 1: Overview of WorldPrior. A frozen VLM provides multimodal semantic guidance only during training, while deployment uses only the visual latent world-model pathway. The adapted representation is evaluated on hand-trajectory forecasting and goal-conditioned control.

VLMs provide complementary object-level, relational, and contextual priors learned from large-scale multimodal data (Radford et al., 2021; Bai et al., 2025b). Existing VLM-guided world models show that such information can improve prediction when multimodal representations are injected into the predictor (Zhang et al., 2026), but retaining the VLM online changes the deployment interface and requires an additional large-model pass for every prediction. We ask instead whether multimodal knowledge can be *internalized* during training and then removed at inference.

We introduce **WorldPrior** to realize this idea. A frozen VLM receives only the observed clip and provides detached observation-side targets that supervise selected student-predictor states through cosine alignment. Because high-level multimodal supervision can perturb fine spatial or temporal structure, a frozen reference predictor initialized from the same baseline checkpoint anchors the adapted student to the original predictive function. The backbone-native objective remains active throughout adaptation. WorldPrior therefore enriches predictive representations with multimodal semantics while preserving backbone-specific dynamics and keeping deployment teacher-free. Figure 1 summarizes this training-only adaptation and VLM-free deployment design, together with the downstream forecasting and control settings considered in this work.

At inference, all training-only auxiliary branches are removed, restoring the original encoder–predictor–downstream graph. Figure 2 compares error–latency trade-offs within each evaluation space: WorldPrior reduces trajectory error at essentially unchanged V-JEPA2 latency, whereas online VLM guidance is substantially slower.

We evaluate the same principle on EgoDex and EgoPAT3D trajectory forecasting and on Push-T, TwoRoom, Reacher, and OGBench-Cube control. Our primary control instantiation adapts LeWorldModel (LeWM) (Maes et al., 2026), while a matched DINO-WM experiment (Zhou et al., 2025) tests transfer to a backbone built on frozen pretrained spatial features. Our contributions are threefold:

- **Training-time internalization of multimodal semantics.** We use a frozen VLM to reshape latent predictive states during training, while completely removing the VLM at inference, preserving the original world-model deployment interface.
- **Semantic enrichment with dynamics preservation.** We combine observation-side cosine alignment, reference-predictor retention, and the backbone-native objective to inject multimodal structure without overwriting fine-grained predictive dynamics.
- **Cross-task and cross-backbone validation.** Across forecasting and control, the adapted models improve downstream utility on multiple latent world-model backbones, while our analysis shows that better decision utility need not correspond to uniformly lower global latent prediction error.

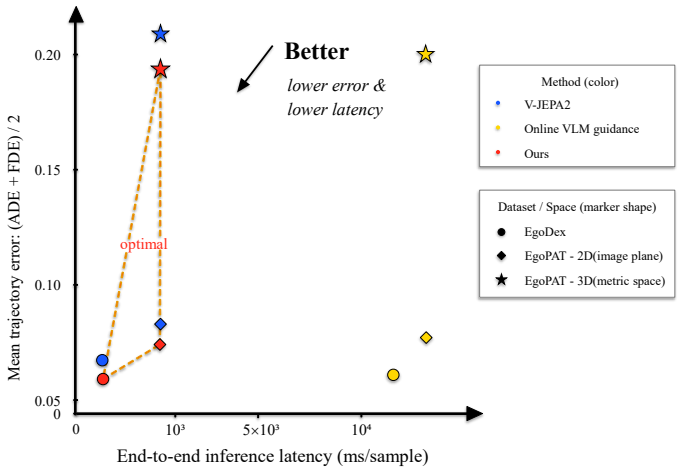


Figure 2: Error–latency comparison on EgoDex and EgoPAT3D. Marker shapes denote different datasets/evaluation spaces; comparisons are intended only within the same marker shape because their error scales are not directly comparable. WorldPrior retains V-JEPA2 latency while reducing trajectory error, whereas online VLM guidance incurs substantially higher latency.

2 Related Work

Predictive latent world models. World models learn compact predictive states for forecasting, planning, and control (Ha & Schmidhuber, 2018; Hafner et al., 2020). Joint-embedding predictive architectures move this learning problem from pixel space to representation space. I-JEPA and V-JEPA establish this principle for images and video (Assran et al., 2023; Bardes et al., 2024), and V-JEPA2 scales self-supervised video pretraining to large-scale visual data before learning action-conditioned latent dynamics for physical planning (Assran et al., 2025). DINO-WM follows a different route by predicting the evolution of frozen DINOv2 spatial patch features, enabling offline dynamics learning and test-time action optimization in a pretrained representation space (Zhou et al., 2025). LeWorldModel instead trains an end-to-end JEPA-style world model directly from pixels with a compact objective designed to stabilize representation learning (Maes et al., 2026). Fast-LeWM further targets rollout efficiency and long-horizon error accumulation by modifying the prediction structure (Gao & Xu, 2026). These works expose complementary design choices in representation construction, optimization stability, and rollout efficiency, but they largely rely on visual predictive supervision to determine how task-relevant structure is organized in the latent state.

Representation alignment, distillation, and privileged supervision. Knowledge distillation transfers predictions or intermediate representations from a teacher to a student (Hinton et al., 2015; Romero et al., 2015; Tian et al., 2020). Learning using privileged information considers a broader setting in which informative supervision is available during training but absent at deployment (Vapnik & Izmailov, 2015). Representation alignment has also become effective in generative modeling: REPA aligns intermediate model states with strong pretrained visual representations and reports substantial improvements in optimization and representation quality (Yu et al., 2025). Together, these studies show that external representations can provide useful inductive bias beyond the native training target. At the same time, successful transfer depends on the compatibility between the structure encoded by the teacher and the geometry required by the receiving model, which becomes particularly important when high-level abstraction and fine-grained spatial information must coexist.

Vision–language guidance for predictive world modeling. Vision–language models provide object-level recognition, relational information, and broader contextual knowledge that can complement local visual prediction. ThinkJEPA studies this complementarity through a dual-temporal design: a densely sampled JEPA branch models fine-grained motion, while a more sparsely sampled VLM branch supplies longer-range semantic guidance (Zhang et al., 2026). Its hierarchical

extraction mechanism aggregates multi-layer VLM representations before injecting them into the predictive pathway. This design addresses the trade-off between dense motion cues and broader semantic context, but it also illustrates a broader challenge in multimodal world modeling: high-level multimodal guidance must remain compatible with the spatial and temporal structure required for precise prediction and control.

Egocentric forecasting and goal-conditioned control. Egocentric trajectory forecasting requires both motion prediction and an understanding of hand–object interaction. EgoPAT3D introduced egocentric 3D action-target prediction (Li et al., 2022), followed by uncertainty-aware state-space models (Bao et al., 2023), diffusion-based 2D forecasting (Ma et al., 2025b), motion-aware diffusion (Ma et al., 2026b), multimodal 3D prediction (Ma et al., 2025a), and unified 2D/3D forecasting (Ma et al., 2026a). EgoDex extends this setting to substantially larger-scale dexterous manipulation data with paired 3D hand annotations (Hoque et al., 2026). Goal-conditioned control evaluates a complementary property: whether predictive representations support action selection toward a visual target. TwoRoom, Push-T, Reacher, and OGBench-Cube span navigation, articulated control, contact-rich planar manipulation, and visually complex 3D manipulation (Sobal et al., 2025; Florence et al., 2022; Tassa et al., 2018; Park et al., 2025). Their diversity is useful because different tasks stress different aspects of a predictive representation, from global spatial relations to fine-grained contact geometry.

3 METHOD

3.1 PROBLEM FORMULATION

Figure 3 details the training-time computation. Let $x_{1:t}$ be an observed clip and $a_{t:t+H-1}$ an optional future action sequence. The deployed visual pathway is

$$z_{1:t} = E_{\text{vis}}(x_{1:t}), \qquad \hat{z}_{t+1:t+H} = P_\theta(z_{1:t}, a_{t:t+H-1}), \tag{1}$$

with the action input omitted for passive forecasting. EgoDex and EgoPAT3D additionally use a trajectory readout H_ϕ; control uses the matched backbone's unchanged planner and introduces no supervised task head. During training, the matched target encoder supplies the future representation required by the native world-model objective,

$$z^+_{t+1:t+H} = \text{sg}(E_{\text{tgt}}(x_{t+1:t+H}))\,, \tag{2}$$

but these future targets are never exposed to the VLM branch.

3.2 OBSERVATION-SIDE MULTIMODAL PRIOR

A frozen VLM F_{VLM} processes only $x_{1:t}$. Teacher caches are constructed with the model's multimodal chat template and autoregressive generation path. We retain valid hidden states from both the multimodal prefill and subsequent decoder steps, while discarding the decoded text itself. The retained observation-side states are then aggregated across selected layers and tokens and detached,

$$r = \text{sg}\Big(\text{Pool}_{\text{tok}}(\text{Pool}_{\text{layer}}(g^{(k)} : k \in \mathcal{K}))\Big)\,, \qquad u = \text{norm}(W_u r), \tag{3}$$

where W_u is a trainable teacher-side projection: a linear map for EgoDex and EgoPAT3D-DT forecasting, and a two-layer MLP for LeWM and DINO-WM control. For each selected predictor layer $q \in \mathcal{Q}$, let $h^{(q)}$ denote its hidden state. A separate trainable student-side projection A_q maps the pooled hidden state to the same alignment space; it is linear for forecasting and a two-layer MLP for control: $s^{(q)} = \text{norm}\big(A_q\,\text{Pool}(h^{(q)})\big)$. We then minimize

$$\mathcal{L}_{\text{sem}} = \frac{1}{|\mathcal{Q}|}\sum_{q\in\mathcal{Q}}\Big(1 - \langle s^{(q)}, u\rangle\Big)\,. \tag{4}$$

All selected predictor readouts contribute equally to the semantic objective (two readouts in every reported instantiation). The objective shapes predictive-state geometry toward multimodal object, relational, and contextual structure rather than asking the student to reproduce language.

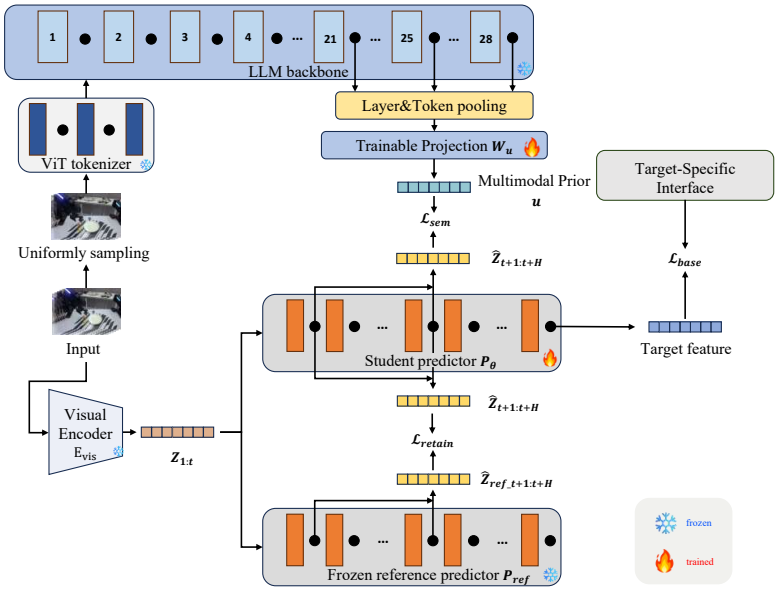


Figure 3: Training-time architecture of WorldPrior. The backbone-native predictive objective remains active, while cosine semantic alignment enriches selected predictive states and reference retention limits functional drift from the matched baseline. The task-specific interface is instantiated as a trajectory readout for egocentric forecasting and as the matched fixed planner for control. All training-only auxiliary branches are removed at inference.

3.3 Reference-Predictor Retention

The student P_θ and frozen reference P_{ref} start from the same matched baseline checkpoint. On identical inputs,

$$z^{\text{ref}}_{t+1:t+H} = P_{\text{ref}}(z_{1:t}, a_{t:t+H-1}), \qquad \mathcal{L}_{\text{retain}} = \left\| P_\theta(z_{1:t}, a_{t:t+H-1}) - \text{sg}(z^{\text{ref}}_{t+1:t+H}) \right\|_1 . \tag{5}$$

This term is functional retention, not future-target supervision: it constrains drift from the pre-adaptation predictive mapping while the native objective continues to supervise future dynamics.

3.4 Backbone-Compatible Adaptation and Teacher-Free Inference

WorldPrior augments, rather than replaces, the receiving objective:

$$\mathcal{L}_{\text{adapt}} = \mathcal{L}_{\text{base}} + \lambda_{\text{sem}}\mathcal{L}_{\text{sem}} + \lambda_{\text{retain}}\mathcal{L}_{\text{retain}}, \tag{6}$$

with $\lambda_{\text{sem}} = 1$ and $\lambda_{\text{retain}} = 0.2$ in the primary experiments. For forecasting, $\mathcal{L}_{\text{base}}$ contains latent prediction and trajectory supervision; for primary control, it is the native LeWM objective; the DINO-WM transfer experiment preserves DINO-WM's native future-patch-feature prediction objective. At inference, the VLM, W_u, semantic readouts, reference predictor, target encoder, and all training losses are absent. The deployed operator graph is therefore unchanged. Appendix A gives the complete backbone-specific objectives and VLM extraction details.

4 Experiments

4.1 Evaluation Setup

We test whether WorldPrior adaptation improves egocentric forecasting, action-conditioned prediction, and closed-loop control; whether it transfers across world-model backbones; and whether the gains are obtained without online VLM cost.

Benchmarks and metrics. EgoDex and EgoPAT3D evaluate future hand trajectories from egocentric video (Hoque et al., 2026; Li et al., 2022); we report ADE/FDE, EgoDex threshold accuracy, and latent FD/SL1/CD. Push-T, TwoRoom, Reacher, and OGBench-Cube use matched LeWM/DINO-WM goal-conditioned protocols and report five-step latent NMSE and closed-loop success. End-to-end latency is measured under matched samples, precision, and hardware. Complete dataset, metric, bootstrap, and timing definitions are in Appendix B.

Implementation. The forecasting encoder is initialized from V-JEPA2 ViT-L/16 and frozen. Student and reference predictors start from the same matched checkpoint. Qwen3-VL-2B-Thinking provides cached observation-side states from blocks 21, 25, and 28; student readouts are attached to V-JEPA2 predictor blocks 1 and 5. On EgoDex, WorldPrior uses 50 epochs in Stage 1 and 50 epochs in Stage 2. On EgoPAT3D-DT, it uses 30 epochs in each stage for both the 2D and 3D settings. The matched V-JEPA2 baselines in Tables 1 and 3 are trained continuously for 100 and 60 epochs, respectively, and report the best checkpoints under the same checkpoint-selection protocol. On both forecasting benchmarks, continuing baseline-only optimization beyond the respective Stage 1 horizon does not improve the earlier best checkpoint. EgoDex uses a 1×10^{-4} predictor/adaptor learning rate with a separate 1×10^{-3} trajectory-head optimizer, whereas EgoPAT3D jointly optimizes its trainable model components at 3.5×10^{-4}. Both forecasting benchmarks use a per-GPU batch size of 4. For goal-conditioned control, the predictor and semantic adaptor share one AdamW optimizer with task-specific learning rates; LeWM-based adaptation uses weight decay 10^{-3} and DINO-WM uses zero weight decay. Control preserves the reproduced LeWM objective and fixed CEM planner, while DINO-WM preserves its native feature-prediction and planning protocol. For all four control tasks, both backbone instantiations use a 50-epoch native-objective Stage 1 followed by 50 epochs of WorldPrior Stage 2 adaptation. Full prompt, decoding, optimizer, batch-size, hardware, and compute details are given in Appendix B.

4.2 EGODEX

Table 1 gives the matched comparison. Relative to V-JEPA2, WorldPrior reduces ADE/FDE from 0.0705/0.0646 to 0.0597/0.0572, raises threshold accuracy from 0.436 to 0.564, and improves FD, SL1, and CD. It also improves all six metrics over our online-guidance reproduction while removing the VLM from inference.

Table 1: Controlled EgoDex comparison under the matched pipeline. All variants share data, backbone, trajectory head, preprocessing, evaluation, and total training budget; matched baselines report the best checkpoint under the same selection protocol.

Method	ADE↓	FDE↓	Acc↑	FD↓	SL1↓	CD↓
V-JEPA2	0.0705	0.0646	0.436	74.47	1.257	0.319
Online VLM guidance (ThinkJEPA repro.)	0.0620	0.0603	0.540	74.06	1.249	0.315
WorldPrior	**0.0597**	**0.0572**	**0.564**	**73.90**	**1.246**	**0.313**

Table 2 provides broader EgoDex ADE/FDE context. WorldPrior obtains the lowest numerical ADE and an FDE close to the reported ThinkJEPA value; because the external rows are cross-source, attribution mainly relies on the matched comparison in Table 1.

4.3 EGOPAT3D

Table 3 reports normalized 2D and metric 3D forecasting. WorldPrior improves the matched V-JEPA2 baseline on every metric, achieving the best or tied-best result on all four 3D metrics and three of four 2D metrics across seen and unseen scenes. It also improves over online VLM guidance on all four 3D metrics.

Table 2: EgoDex ADE/FDE context. External rows follow the ThinkJEPA comparison (Zhang et al., 2026) and are included for cross-source context only.

Group	Method	ADE↓	FDE↓
Task-specific trajectory predictors	Decoder-only + BC	0.0767	0.0818
	Decoder-only + DDPM	0.1148	0.1238
	Decoder-only + Flow Matching	0.1527	0.1574
	Encoder–decoder + BC	0.0774	0.0924
	Encoder–decoder + DDPM	0.1272	0.1245
	Encoder–decoder + Flow Matching	0.1736	0.1557
Reported VLM/JEPA systems	Qwen3-VL Thinking (Bai et al., 2025a)	0.1420	0.1440
	V-JEPA Predictor (Zhang et al., 2026)	0.0710	0.0660
	ThinkJEPA (Zhang et al., 2026)	0.0610	**0.0560**
Ours	WorldPrior	**0.0597**	0.0572

Table 3: EgoPAT3D-DT trajectory forecasting in normalized 2D image-plane and metric 3D camera coordinates (lower is better). External 2D results follow MADiff (Ma et al., 2026b) and Uni-Hand (Ma et al., 2026a); external 3D results follow MMTwin (Ma et al., 2025a). Matched V-JEPA2, online-guidance, and WorldPrior rows use the same evaluation and total training budget.

		ADE↓		FDE↓	
Space	Method	Seen scenes	Unseen scenes	Seen scenes	Unseen scenes
2D image plane	USST (Bao et al., 2023)	0.082	0.060	0.118	0.087
	Diff-IP2D (Ma et al., 2025b)	0.080	0.066	0.130	0.087
	MADiff (Ma et al., 2026b)	0.065	0.054	0.105	0.086
	Uni-Hand (Ma et al., 2026a)	**0.064**	**0.052**	0.109	0.087
	V-JEPA2	0.073	0.062	0.108	0.092
	Online VLM guidance (ThinkJEPA)	0.069	0.056	0.105	**0.082**
	WorldPrior	0.066	**0.052**	**0.103**	**0.082**
3D metric space	CVH	1.100	0.952	1.278	1.018
	OCT	0.370	0.309	0.524	0.397
	USST	0.183	0.120	0.341	0.185
	S-Mamba	0.185	0.138	0.355	0.207
	Diff-IP3D	0.199	0.156	0.377	0.229
	MADiff3D	0.183	0.139	0.363	0.224
	MMTwin (Ma et al., 2025a)	0.170	0.118	**0.336**	0.189
	V-JEPA2	0.173	0.120	0.348	0.199
	Online VLM guidance (ThinkJEPA)	0.170	0.114	0.338	0.182
	WorldPrior	**0.160**	**0.109**	**0.336**	**0.176**

4.4 Action-Conditioned Prediction

Table 4 compares five-step latent prediction for the exact LeWM and WorldPrior checkpoints used in control. Reacher and OGBench-Cube improve by 27.5% and 6.9%, Push-T is statistically unchanged, and TwoRoom NMSE increases despite a success gain from 86% to 94%. This non-monotonic relation motivates reporting both predictive fidelity and closed-loop utility.

Table 4: Five-step latent NMSE on identical validation clips. Baseline and WorldPrior use matched total training budgets, with the baseline reporting its best checkpoint under the same selection protocol. Positive relative change denotes lower error; n.s. means the paired-bootstrap 95% confidence interval of the raw-MSE difference contains zero.

Task	Clips	LeWM NMSE ↓	WorldPrior NMSE ↓	Rel. change ↑	LeWM SR	WorldPrior SR
Push-T	904	0.04223	0.04169	$+1.3\%$ (n.s.)	96%	96%
TwoRoom	2000	0.10527	0.18966	-80.2%	86%	94%
Reacher	389	0.01302	0.00944	$+27.5\%$	86%	90%
OGBench-Cube	976	0.01681	0.01565	$+6.9\%$	74%	80%

4.5 Goal-Conditioned Control and Backbone Transfer

Under the matched LeWM pipeline, WorldPrior improves TwoRoom by 8 pp, Reacher by 4 pp, and OGBench-Cube by 6 pp while preserving Push-T, raising the four-task average from 85.5% to 90.0% (Table 5). Applying the same training-only principle to DINO-WM improves or preserves three of four tasks and raises its matched average from 83.0% to 85.5%, supporting transfer across distinct representation families. For an iso-compute comparison, each reproduced LeWM and DINO-WM baseline is continued for the same Stage 2 epoch budget using only its native objective, and the best checkpoint over the matched total budget is reported. None of these baseline continuations exceeds its earlier best task-success result, indicating that the native control baselines have largely saturated by approximately the Stage 1 horizon; the WorldPrior gains therefore cannot be attributed to additional optimization steps alone.

For literature context, Table 5 also reproduces the unified four-environment baseline results reported in LeWorldModel (Maes et al., 2026); these rows are shown separately from our matched reproductions and are not used for attribution of the WorldPrior gains.

Table 5: Goal-conditioned success rate (%, higher is better). Literature-context rows follow LeWorldModel (Maes et al., 2026); matched reproduced baselines and WorldPrior use the same total training budget, with each baseline reporting its best checkpoint under the same selection protocol. Parentheses show absolute changes from the corresponding reproduced backbone.

Method	TwoRoom	Reacher	Push-T	OGBench-Cube	Avg.
Literature context (LeWorldModel evaluation)					
LeWM (Maes et al., 2026)	87	86	**96**	74	85.75
DINO-WM (Maes et al., 2026; Zhou et al., 2025)	**100**	79	74	**86**	84.75
PLDM (Maes et al., 2026; Sobal et al., 2025)	97	78	78	65	79.50
Matched LeWM adaptation					
LeWM (reproduced)	86	86	**96**	74	85.50
WorldPrior (LeWM)	94 (+8 pp)	**90** (+4 pp)	**96** (+0 pp)	80 (+6 pp)	**90.00** (+4.5 pp)
Matched DINO-WM adaptation					
DINO-WM (reproduced)	**100**	74	74	84	83.00
WorldPrior (DINO-WM)	**100** (+0 pp)	82 (+8 pp)	78 (+4 pp)	82 (−2 pp)	85.50 (+2.5 pp)

4.6 Ablation Study

Table 6 evaluates the contribution of semantic distillation and reference-predictor retention while keeping the native latent-prediction and trajectory objectives active. The full WorldPrior model achieves the best result across all six EgoDex metrics. Removing semantic distillation produces the larger degradation in trajectory ADE/FDE and threshold accuracy, indicating that multimodal supervision provides the primary task-relevant improvement. Removing reference retention also consistently degrades all metrics, supporting its role in preserving useful predictive structure during semantic adaptation. Together, these results show that semantic enrichment and predictive retention provide complementary benefits rather than redundant regularization.

Table 6: Component-removal results on EgoDex. All variants retain $\mathcal{L}_{\text{latent}}$ and $\mathcal{L}_{\text{traj}}$ with unit weight. Lower ADE, FDE, FD, SL1, and CD are better; higher Acc is better. "w/o" denotes without. Bold denotes the best result in each metric.

Setting	ADE↓	FDE↓	Acc↑	FD↓	SL1↓	CD↓
Full module	**0.0597**	**0.0572**	**0.564**	**73.90**	**1.246**	**0.313**
w/o distill	0.0705	0.0677	0.425	74.11	1.250	0.315
w/o retain	0.0678	0.0665	0.461	74.06	1.249	0.315

4.7 Deployment Efficiency

WorldPrior preserves the matched V-JEPA2 baseline's inference-time operator graph. On EgoDex, V-JEPA2 and WorldPrior require 527.9 ms and 525.7 ms per sample, respectively; on EgoPAT3D-2D, the corresponding latencies are 889 ms and 882 ms, and on EgoPAT3D-3D they are 889 ms and 882 ms.

These sub-percent differences are treated as measurement variability rather than computational speedups. In contrast, online VLM guidance requires 14.5 s, 19.7 s, and 20.2 s on EgoDex, EgoPAT3D-2D, and EgoPAT3D-3D, respectively, corresponding to more than a $22\times$ inference-time overhead. On EgoDex, V-JEPA2 and WorldPrior each require approximately 3.22 TFLOPs per sample under the same deployed computation graph, whereas online VLM guidance requires approximately 11.27 TFLOPs ($3.50\times$). The corresponding EgoDex latency ratio, $14.5\,\mathrm{s}/525.7\,\mathrm{ms} = 27.6\times$, is substantially larger than the FLOP ratio, consistent with the sequential token-by-token decoding required by online autoregressive VLM inference, which is not captured by aggregate operation counts. Detailed model-size and FLOP accounting is provided in Appendix B.4.

5 Analysis and Discussion

These results support a representation-centric interpretation: EgoDex gains more in ADE/FDE and threshold accuracy than in latent feature discrepancies. In control, success and NMSE need not vary monotonically: TwoRoom improves success despite higher NMSE; Push-T shows no success gain despite a small, nonsignificant NMSE decrease. Thus, lower global latent error is not necessary for higher success in our evaluation. The mechanism remains unclear; planner sensitivity to latent geometry, feature scaling, or other task-specific factors may contribute.

The comparison with online VLM guidance further suggests that the VLM is most useful here as a source of *representational supervision*, rather than as a permanent inference-time dependency. WorldPrior improves the matched forecasting results while removing the VLM from the deployed graph, indicating that task-relevant multimodal structure can be internalized by the predictive model during training. We do not claim that the adapted student reproduces the full reasoning capability of the VLM; rather, the results support the more specific conclusion that observation-side multimodal priors provide an effective training signal for organizing predictive latent states.

Cross-backbone results indicate that the effect of WorldPrior is representation- and task-dependent rather than uniformly stronger or weaker on DINO-WM. Although the four-task gain is smaller for DINO-WM ($+2.5$ pp vs. $+4.5$ pp), its TwoRoom baseline is already saturated at 100%; excluding this task, both backbones obtain a mean net gain of 3.3 pp over the remaining three tasks. LeWM learns a compact representation end-to-end, whereas DINO-WM predicts frozen DINOv2 spatial patch features, so the two backbones expose substantially different representation geometries to the same adaptation principle. One possible interpretation is that globally pooled multimodal supervision interacts differently with these geometries, potentially benefiting higher-level relational structure while affecting geometry-sensitive information differently across tasks. However, the present experiments do not isolate this mechanism, and the 2-pp change on OGBench-Cube is too small to establish a semantic–geometric trade-off on its own. We therefore treat this explanation as a hypothesis for future investigation rather than a causal conclusion. Extended discussion is provided in Appendix E.

6 Conclusion

WorldPrior uses VLM-derived multimodal representations as training-only priors for latent world-model adaptation. Cosine hidden-state alignment enriches task-relevant structure, reference-predictor retention limits destructive drift, and the native objective preserves the receiving backbone's dynamics. Across egocentric forecasting and goal-conditioned control, the adapted student improves prediction and planning while remaining VLM-free at deployment. The results suggest that multimodal foundation models can serve as reusable representational priors rather than permanent components of the online world-model graph.

Reproducibility Statement

We provide the core architecture, objectives, benchmark definitions, optimization settings, and matched evaluation design in the main paper. Appendix A gives the backbone-specific method and training-objective details, while Appendix B documents the datasets, preprocessing, evaluation metrics, optimization settings, model and inference-compute accounting, hardware, training costs, and latency measurement protocol. An anonymized implementation, together with configuration files

and evaluation scripts for reproducing the principal experiments, is available in our anonymous code repository. An anonymized code snapshot is also provided in the supplementary material.

AI USE STATEMENT

Generative AI tools were used to assist with portions of software development, including drafting and refining implementation code, as well as for language editing to improve the clarity and readability of the manuscript. They were also used to brainstorm high-level presentation and layout ideas for selected scientific figures. Generative AI tools were not used to formulate the core research methodology, generate experimental results, or independently interpret the experimental findings. All AI-assisted code and manuscript content were carefully reviewed, verified, and, where applicable, tested by the authors. The authors take full responsibility for the final content of the paper and all associated artifacts.

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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15619–15629, 2023.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, et al. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, et al. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Wentao Bao, Lele Chen, Libing Zeng, Zhong Li, Yi Xu, Junsong Yuan, and Yu Kong. Uncertainty-aware state space transformer for egocentric 3D hand trajectory forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 13702–13711, 2023. doi: 10.1109/ICCV51070.2023.01260.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *Transactions on Machine Learning Research*, 2024.

Pete Florence, Corey Lynch, Andy Zeng, Oscar A. Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 158–168. PMLR, 2022.

Yuntian Gao and Xiangyu Xu. Fast LeWorldModel. *arXiv preprint arXiv:2606.26217*, 2026.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems 31*, pp. 2451–2463. Curran Associates, Inc., 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Ryan Hoque, Peide Huang, David J. Yoon, Mouli Sivapurapu, and Jian Zhang. EgoDex: Learning dexterous manipulation from large-scale egocentric video. In *International Conference on Learning Representations*, 2026.

Yiming Li, Ziang Cao, Andrew Liang, Benjamin Liang, Luoyao Chen, Hang Zhao, and Chen Feng. EgoPAT3D: Egocentric prediction of action target in 3D. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Junyi Ma, Wentao Bao, Jingyi Xu, Guanzhong Sun, Xieyuanli Chen, and Hesheng Wang. Novel diffusion models for multimodal 3D hand trajectory prediction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2025a. doi: 10.1109/IROS60139.2025.11246803.

Junyi Ma, Jingyi Xu, Xieyuanli Chen, and Hesheng Wang. Diff-IP2D: Diffusion-based hand-object interaction prediction on egocentric videos. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2025b. doi: 10.1109/IROS60139.2025.11247704.

Junyi Ma, Wentao Bao, Jingyi Xu, Guanzhong Sun, Yu Zheng, Erhang Zhang, Xieyuanli Chen, and Hesheng Wang. Uni-Hand: Universal hand motion forecasting in egocentric views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026a. doi: 10.1109/TPAMI.2026.3692504. Early Access.

Junyi Ma, Xieyuanli Chen, Wentao Bao, Jingyi Xu, and Hesheng Wang. MADiff: Motion-aware mamba diffusion models for hand trajectory prediction on egocentric videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(3):3250–3267, 2026b. doi: 10.1109/TPAMI.2025.3633453.

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.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. OGBench: Benchmarking offline goal-conditioned RL. In *International Conference on Learning Representations*, 2025.

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

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. FitNets: Hints for thin deep nets. In *International Conference on Learning Representations*, 2015.

Uladzislau 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. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-1465.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *International Conference on Learning Representations*, 2020.

Vladimir Vapnik and Rauf Izmailov. Learning using privileged information: Similarity control and knowledge transfer. *Journal of Machine Learning Research*, 16(61):2023–2049, 2015.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. In *International Conference on Learning Representations*, 2025.

Haichao Zhang, Yijiang Li, Shwai He, Tushar Nagarajan, Mingfei Chen, Jianglin Lu, Ang Li, and Yun Fu. ThinkJEPA: Empowering latent world models with large vision-language reasoning model. *arXiv preprint arXiv:2603.22281*, 2026.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. In *International Conference on Machine Learning*, 2025.

A Detailed Method and Backbone-Specific Objectives

A.1 Native Future-Target Construction

During training, the matched target encoder E_{tgt} maps ground-truth future observations to the representation expected by the receiving world model,

$$z^{+}_{t+1:t+H} = \mathrm{sg}(E_{\mathrm{tgt}}(x_{t+1:t+H})) \,. \tag{7}$$

The construction of E_{tgt}, target pooling or masking, and the predictive discrepancy follow the corresponding backbone protocol. For V-JEPA2 forecasting, E_{vis} is the frozen V-JEPA2 encoder (Assran et al., 2025). In the primary control experiments, we preserve the reproduced LeWM representation, target construction, native objective, and CEM planning protocol (Maes et al., 2026). The backbone-transfer experiment instead preserves DINO-WM's frozen DINOv2 spatial patch representation and native future-feature prediction objective (Zhou et al., 2025). Future targets are used only by the backbone-native objective and are never provided to the VLM branch.

This distinction is important for the interpretation of training-only multimodal supervision. The VLM observes the same history available at prediction time, whereas the native world-model loss may still use future targets according to the receiving backbone's original training protocol. WorldPrior therefore does not introduce an additional source of future information; it adds an observation-side representational prior to the existing predictive learning problem.

A.2 Observation-Side VLM Feature Extraction

We instantiate the frozen teacher with Qwen3-VL-2B-Thinking (Bai et al., 2025a). Its LLM backbone contains 28 Transformer blocks. Using one-based indices, we extract hidden states from blocks 21, 25, and 28, chosen from the upper portion of the backbone to capture high-level multimodal context without assuming that any single layer isolates a unique semantic or reasoning factor.

The teacher receives only frames sampled from the observed clip $x_{1:t}$ together with a task-oriented multimodal prompt. Cache construction uses the Qwen chat template and autoregressive generation to expose decoder hidden states; the decoded text itself is discarded and is never used as a supervision target or student input. For each selected block k, $g^{(k)}$ contains the valid hidden states from the multimodal prefill and all decoder steps produced before EOS or the configured generation-length limit. No additional fixed-K truncation is applied to generated-token states. No future observation, future target latent, trajectory label, success label, or target-frame metadata is exposed to this branch. We first average the selected layers and then apply masked mean pooling over the retained valid tokens before detaching the source representation. In the notation of the main paper,

$$r = \mathrm{sg}\Big(\mathrm{Pool}_{\mathrm{tok}}\Big(\mathrm{Pool}_{\mathrm{layer}}\left(g^{(21)}, g^{(25)}, g^{(28)}\right)\Big)\Big), \qquad u = \mathrm{norm}(W_u r). \tag{8}$$

The cached source r remains fixed throughout WorldPrior adaptation. The teacher-side projection W_u is trainable, so the projected target u may evolve as the student alignment space changes. Gradients are never propagated into the VLM.

The source representations are computed offline and stored as detached training targets. Once the cache has been constructed, the VLM is not executed in the student-training data path and is never required during validation or test inference. Besides reducing repeated teacher computation, this design makes the training/deployment asymmetry explicit: the large multimodal model defines a representational prior rather than an online input to the world model.

A.3 Student Readouts and Cosine Semantic Alignment

Given the detached teacher representation, we attach a separate student-side projection to each selected predictor state rather than to the final task output. The selected predictor blocks are 1 and 5 for V-JEPA2 forecasting on EgoDex and EgoPAT3D-DT, 3 and 6 for LeWM control, and 4 and 6 for DINO-WM control. All block indices in this subsection are one-based. In every reported instantiation, the two selected states contribute equally to $\mathcal{L}_{\mathrm{sem}}$.

Teacher-side and student-side projections map the pooled representations into a shared 512-dimensional alignment space. For EgoDex and EgoPAT3D-DT, the teacher projection is a trainable

$2048 \to 512$ linear map, and each selected 384-dimensional predictor state has a separate trainable $384 \to 512$ linear projection. For LeWM and DINO-WM control, the teacher projection is a trainable two-layer MLP with dimensions $2048 \to 1024 \to 512$, while each selected predictor state has a separate two-layer MLP projection with dimensions $d_b \to 1024 \to 512$, where d_b is the predictor-state width of the corresponding backbone. The control projections apply layer normalization before the first linear layer and GELU between the two linear layers. All projected representations are ℓ_2-normalized before cosine alignment. The method does not depend on these particular block indices.

The semantic objective acts on representation geometry rather than raw feature magnitude. Its goal is not to reproduce natural language or make the student emulate the full VLM. Instead, it biases predictive states toward information correlated with the VLM's multimodal abstraction, including object identity, interaction context, affordance-relevant cues, and relational structure, while the native objective continues to define the temporal prediction problem.

A.4 Reference-Predictor Retention

The trainable student P_θ and frozen reference P_{ref} are initialized from the same matched baseline checkpoint. They receive identical observed latents and, for control, identical future action sequences. The reference predictor remains frozen throughout WorldPrior adaptation.

Reference retention is deliberately distinct from future-target supervision. Its target is the *pre-adaptation predictive function*, evaluated on the same inputs, rather than the ground-truth future representation. In all reported instantiations it is implemented as element-wise L1 error with mean reduction over the compared prediction tensor, rather than an unnormalized sum. Consequently, the native objective supplies future-prediction supervision, $\mathcal{L}_{\text{sem}}$ supplies multimodal structure, and $\mathcal{L}_{\text{retain}}$ limits destructive drift from spatial and dynamical structure already encoded by the receiving backbone. This separation is especially relevant when high-level semantic alignment and fine-grained geometric prediction induce partially different biases.

A.5 Backbone-Specific Training Objectives

For egocentric forecasting,

$$\mathcal{L}_{\text{latent}} = d_{\text{latent}}\left(\hat{z}_{t+1:t+H}, z^{+}_{t+1:t+H}\right), \tag{9}$$

$$\mathcal{L}_{\text{traj}} = d_{\text{traj}}\left(H_\phi(\hat{z}_{t+1:t+H}), y\right), \tag{10}$$

$$\mathcal{L}_{\text{forecast}} = \mathcal{L}_{\text{latent}} + \mathcal{L}_{\text{traj}} + \lambda_{\text{sem}}\mathcal{L}_{\text{sem}} + \lambda_{\text{retain}}\mathcal{L}_{\text{retain}}. \tag{11}$$

The latent-prediction and trajectory terms retain unit weight in our implementation.

For the primary Push-T, TwoRoom, Reacher, and OGBench-Cube control experiments, the receiving objective is the reproduced LeWM objective,

$$\mathcal{L}_{\text{LeWM}} = \mathcal{L}_{\text{latent}} + \lambda_{\text{sig}}\mathcal{L}_{\text{SIGReg}}, \tag{12}$$

$$\mathcal{L}_{\text{control}} = \mathcal{L}_{\text{LeWM}} + \lambda_{\text{sem}}\mathcal{L}_{\text{sem}} + \lambda_{\text{retain}}\mathcal{L}_{\text{retain}}. \tag{13}$$

λ_{sig} and the SIGReg definition are inherited unchanged from the reproduced LeWM baseline (Maes et al., 2026). No supervised control task loss is introduced, and the CEM planner remains outside the optimization graph.

For DINO-WM, the frozen DINOv2 encoder E_{DINO} maps each observation to a set of spatial patch features,

$$z_t^{\text{DINO}} = E_{\text{DINO}}(o_t), \qquad z_t^{\text{DINO}} \in \mathbb{R}^{N \times D}, \tag{14}$$

where N denotes the number of spatial patches and D the feature dimension. Following the original DINO-WM formulation (Zhou et al., 2025), the action-conditioned transition model predicts future DINOv2 patch features and is optimized with a latent consistency loss. Under our H-step notation, we write this objective as

$$\mathcal{L}_{\text{DINO}} = \frac{1}{H}\sum_{h=1}^{H} \left\| \hat{z}_{t+h}^{\text{DINO}} - \text{sg}\left(E_{\text{DINO}}(o_{t+h})\right) \right\|_F^2, \tag{15}$$

where $\hat{z}_{t+h}^{\mathrm{DINO}}$ is produced autoregressively by the DINO-WM transition model conditioned on the observation history and action sequence. In implementation, the loss is averaged over the predicted frames, spatial patches, and feature dimensions.

WorldPrior adaptation preserves this native DINO-WM objective:

$$\mathcal{L}_{\text{DINO-adapt}} = \mathcal{L}_{\text{DINO}} + \lambda_{\text{sem}}\mathcal{L}_{\text{sem}} + \lambda_{\text{retain}}\mathcal{L}_{\text{retain}}. \tag{16}$$

The optional DINO-WM image decoder and its reconstruction objective are not part of the predictive adaptation objective; in the original DINO-WM formulation, decoder training is independent of the transition model and is used only for visualization and interpretability (Zhou et al., 2025).

Across the primary forecasting and LeWM-control experiments, we use $\lambda_{\text{sem}} = 1$ and $\lambda_{\text{retain}} = 0.2$.

A.6 Teacher-Free Deployment

At validation and test time, the VLM, W_u, semantic readout heads, reference predictor, target encoder, and all training losses are absent. Egocentric forecasting uses only the adapted visual encoder/predictor pathway and trajectory head. The primary control experiments use the unchanged LeWM CEM planner, while the DINO-WM transfer experiment retains its matched downstream planning protocol. WorldPrior adaptation therefore changes learned parameters but introduces no inference-time multimodal input, operator, or large-model forward pass.

B Detailed Experimental Protocols

B.1 Experimental Questions

The experiments are organized around four questions: (i) whether training-only multimodal-semantic alignment improves egocentric future-trajectory forecasting; (ii) whether it improves task-relevant predictive representations and closed-loop control across diverse dynamics regimes; (iii) whether the same adaptation principle transfers across distinct latent-world-model backbones; and (iv) whether these gains can be obtained without retaining the VLM's inference-time cost.

B.2 Benchmarks

EgoDex. EgoDex contains 829 hours of egocentric dexterous manipulation video spanning 194 tabletop tasks, with paired 3D upper-body and hand-pose annotations (Hoque et al., 2026). We evaluate future hand-trajectory prediction and representation-space forecasting.

EgoPAT3D. EgoPAT3D contains 150 RGB-D recordings, approximately 15,000 hand–object actions, and 15 household scenes (Li et al., 2022). We follow the established EgoPAT3D-DT protocol and report both metric 3D and normalized image-plane 2D hand-trajectory errors under the same seen/unseen split and prediction horizon (Bao et al., 2023; Ma et al., 2025a; 2026b;a).

Goal-conditioned control. We evaluate Push-T, TwoRoom, Reacher, and OGBench-Cube using the offline datasets, observation preprocessing, frame skip, and goal-conditioned evaluation protocols associated with the LeWM/DINO-WM benchmark family (Zhou et al., 2025; Maes et al., 2026). TwoRoom originates from the environment of Sobal et al. (Sobal et al., 2025); Push-T is a contact-rich planar manipulation task (Florence et al., 2022); Reacher is drawn from the DeepMind Control Suite (Tassa et al., 2018); and OGBench-Cube uses the single-cube manipulation variant of OGBench (Park et al., 2025). The suite spans navigation, articulated control, planar contact, and visually complex 3D manipulation. Our primary matched control study adapts LeWM. We additionally reproduce DINO-WM and apply WorldPrior to that backbone to evaluate cross-backbone transferability rather than only cross-task performance.

B.3 Metric Definitions

For egocentric trajectory forecasting, average displacement error (ADE) is the Euclidean error averaged over all predicted future steps and trajectory points. Final displacement error (FDE) measures the

Table 7: Approximate parameter counts of the deployed world-model backbones used in our experiments. The DINO-WM total excludes its optional visualization decoder.

Backbone	Representation encoder	Predictor / dynamics	Approx. total
V-JEPA2	ViT-L/16: ∼304M	∼22M	∼326M
LeWM	ViT-Tiny: ∼5M	∼10M	∼15M
DINO-WM	DINOv2 ViT-S/14: ∼21M	∼19M	∼40M

Euclidean error at the final prediction step. Lower values are better. For EgoPAT3D, 3D ADE/FDE are computed in metric camera-coordinate space and reported in meters. The 2D counterparts are computed on image-plane coordinates normalized by the image dimensions, following the established 2D protocol (Ma et al., 2025a; 2026a); they are therefore dimensionless and must not be compared numerically with the 3D values.

For EgoDex, threshold accuracy (Acc) is the fraction of predicted joint positions whose Euclidean error is below 0.05 m. We additionally report latent feature distance (FD), SmoothL1 distance (SL1), and cosine distance (CD) between predicted and target future latent representations. FD is the mean Euclidean feature distance, SL1 is the mean element-wise SmoothL1 discrepancy, and CD is defined as $1 - \cos(\cdot)$. Each metric is averaged over evaluated samples, future steps, and latent tokens. For the error–latency visualization, mean trajectory error is the arithmetic mean of ADE and FDE on EgoDex; on EgoPAT3D it additionally averages the seen- and unseen-scene results within each coordinate space.

For action-conditioned prediction, each model receives the same initial three-frame history and ground-truth action sequence. After the initial history, each predicted latent is fed back into the predictor for the next rollout step. Let $\hat{z}_i, z_i \in \mathbb{R}^{H \times D}$ denote predicted and target latent trajectories for sample i, with $H = 5$. We compute

$$\mathrm{MSE}_{\mathrm{latent}} = \frac{1}{BHD} \sum_{i=1}^{B} \sum_{h=1}^{H} \|\hat{z}_{i,h} - z_{i,h}\|_2^2 . \tag{17}$$

NMSE divides this value by the mean per-dimension variance of target latents on the same validation clips. Within each benchmark, LeWM and WorldPrior use the same target encoder, validation samples, rollout horizon, and normalization denominator. Because latent statistics differ across environments, NMSE is compared only within a benchmark. Statistical significance is assessed by paired bootstrap resampling of validation clips using the raw per-clip MSE difference.

Complementing open-loop latent prediction, closed-loop control performance is measured by task success, defined as the percentage of evaluation episodes satisfying the environment's goal criterion. We report relative NMSE change and absolute success-rate change in percentage points (pp).

B.4 Model Size and Inference Compute

Backbone parameter counts. Table 7 summarizes the deployed backbone sizes used in our experiments. For forecasting, we instantiate the V-JEPA2 ViT-L/16 encoder together with its 12-block predictor (Assran et al., 2025). The encoder contains approximately 304M parameters in the instantiated model (the official model table rounds ViT-L/16 to 300M), while the predictor contains approximately 22M parameters. LeWM reports approximately 15M parameters for its complete end-to-end encoder–predictor world model (Maes et al., 2026). DINO-WM uses the frozen DINOv2 ViT-S/14 representation encoder (approximately 21M parameters) together with an approximately 19M-parameter transition predictor (Zhou et al., 2025), for roughly 40M deployed parameters excluding the optional visualization decoder. For LeWM, the original paper reports an approximately 5M-parameter ViT-Tiny encoder and an approximately 10M-parameter Transformer predictor, yielding approximately 15M parameters in total. These counts describe the deployed world-model components; WorldPrior's VLM teacher, teacher-side projection, semantic readouts, and reference predictor are training-only and therefore do not increase the deployed parameter count.

Inference FLOPs. We additionally report theoretical inference compute on EgoDex in Table 8. FLOPs are counted from the instantiated architectures and evaluation input shapes using two floating-

point operations per multiply–accumulate. V-JEPA2 and WorldPrior execute the same deployed encoder–predictor–trajectory-head graph and therefore have identical theoretical compute, approximately 3.22 TFLOPs per sample. The online VLM-guidance reproduction additionally executes Qwen3-VL-2B-Thinking and requires approximately 11.27 TFLOPs per sample, or $3.50\times$ the compute of WorldPrior.

For the online branch, the count includes Qwen3-VL visual encoding, prompt prefill, KV-cached autoregressive decoding, and the V-JEPA2/FiLM/downstream pathway. The representative EgoDex setting uses 32 frames at 256×256 resolution. The Qwen3-VL vision tower processes 4096 pre-merge visual patches and produces 1024 post-merge visual tokens for the language model. Under the evaluation prompt/template, we use a representative prefill length of 1166 tokens and the observed mean generation length of 435 new tokens. The Qwen branch contributes approximately 8.03 TFLOPs, with the complete online pipeline totaling 11.27 TFLOPs. These values are theoretical operator counts rather than achieved hardware throughput.

Table 8: Theoretical inference compute on EgoDex. Counts use two FLOPs per multiply–accumulate and the representative online-generation length of 435 tokens.

Method	TFLOPs / sample	Relative compute
V-JEPA2	3.22	$1.00\times$
WorldPrior	3.22	$1.00\times$
Online VLM guidance	11.27	$3.50\times$

The FLOP ratio is substantially smaller than the measured EgoDex latency ratio ($27.6\times$ relative to WorldPrior). This is expected: aggregate FLOP counts do not encode execution parallelism, memory traffic, kernel-launch overhead, or the sequential dependency of autoregressive generation. In particular, the 435 decode steps must be executed token by token, whereas the visual world-model forward pass is dominated by comparatively parallel dense computation.

B.5 Latency Protocol

Latency is measured as end-to-end wall-clock time per sample on EgoDex and EgoPAT3D. Within each benchmark, all methods use matched evaluation samples, batch size, numerical precision, hardware, and timing protocol while retaining their native model-specific preprocessing.

For online VLM guidance, Qwen3-VL-2B-Thinking is configured with a maximum output length of 512 tokens. We first record the generated output lengths over the corresponding evaluation set and obtain mean lengths of 435 tokens on EgoDex, 475 tokens on EgoPAT3D-2D, and 487 tokens on EgoPAT3D-3D. To obtain a representative latency estimate for each benchmark, online-guidance latency is measured using these mean generation lengths. The reported timing includes VLM-specific frame sampling, multimodal preprocessing, autoregressive generation, and feature extraction.

V-JEPA2 and WorldPrior share the same deployed architecture, so their sub-percent timing differences are interpreted as measurement variability rather than evidence of a computational speedup. The efficiency claim is therefore architectural: WorldPrior eliminates the online VLM branch rather than accelerating the visual predictor itself.

B.6 Implementation Details

Egocentric forecasting. For EgoDex and EgoPAT3D, the visual encoder is initialized from V-JEPA2 and remains frozen throughout WorldPrior adaptation. The V-JEPA2 baseline and WorldPrior use the same visual encoder, predictor architecture, data split, input preprocessing, target encoding, and evaluation code. The student predictor is initialized from the matched V-JEPA2 baseline checkpoint, and the reference predictor is a frozen copy of the same checkpoint. EgoPAT3D uses coordinate-specific 3D and 2D trajectory heads and prompts under the same seen/unseen split and prediction horizon; both trajectory losses use the matched masked Smooth-L1 formulation described below.

Online VLM-guidance reproduction. Our ThinkJEPA-style online-guidance reproduction uses the same V-JEPA2 visual encoder, predictor backbone, trajectory head, data split, and evaluation

code as the V-JEPA2 and WorldPrior rows. Unlike WorldPrior, it executes Qwen3-VL-2B-Thinking for every inference sample. Generation is capped at 512 output tokens. The observed mean output lengths are 435, 475, and 487 tokens for EgoDex, EgoPAT3D-2D, and EgoPAT3D-3D, respectively; these mean lengths are used as the representative generation lengths for latency measurement. Timing includes VLM-specific frame sampling, multimodal preprocessing, autoregressive generation, and feature extraction. We use this variant to quantify the deployment cost of retaining VLM guidance at inference.

LeWM-based goal-conditioned control. For Push-T, TwoRoom, Reacher, and OGBench-Cube, WorldPrior is initialized from the matched reproduced LeWM checkpoint. The student and frozen reference receive the same observation and action inputs, while the visual encoder, offline data, preprocessing, frame skip, native LeWM objective, CEM planning budget, planning horizon, and success criterion are held fixed relative to the reproduced baseline. Training therefore preserves $\mathcal{L}_{\mathrm{LeWM}}$ and adds only $\mathcal{L}_{\mathrm{sem}}$ and $\mathcal{L}_{\mathrm{retain}}$; no task head or planner supervision is introduced.

DINO-WM backbone transfer. To test whether WorldPrior adaptation is tied to the LeWM representation space, we apply the same framework to a matched DINO-WM reproduction (Zhou et al., 2025). DINO-WM predicts the future evolution of frozen DINOv2 spatial patch features. We preserve the DINOv2 observation representation, native DINO-WM predictive objective, and matched downstream planning/evaluation protocol. WorldPrior contributes only training-time multimodal-semantic alignment and reference retention. The reproduced DINO-WM checkpoint is the comparison baseline, and the adapted checkpoint is evaluated under the same task-success protocol. This experiment tests transferability of the adaptation principle rather than uniform superiority of one backbone over another.

Inference protocol. Neither the VLM, W_u, semantic readout heads, nor the reference predictor is executed during validation or test inference. Egocentric forecasting uses only the adapted predictor and trajectory head. The primary control experiments use the unchanged LeWM CEM planner, while the DINO-WM transfer experiment retains its matched downstream planning protocol.

B.7 Prompting, Decoding, and Readout Reproducibility

Chat formatting and prompt organization. All semantic caches use Qwen3-VL-2B-Thinking and the checkpoint chat template with `add_generation_prompt=True`. The VLM is executed autoregressively to expose decoder hidden states, but the decoded text is never consumed by the student. For clarity and reproducibility, the exact EgoDex and EgoPAT3D-DT prompts are listed separately in Appendix B.8, and the four control-task prompts are listed in Appendix B.9.

Decoding. All WorldPrior semantic caches use `max_new_tokens=512`, covering EgoDex, EgoPAT3D-DT (2D and 3D), and the four goal-conditioned control tasks for both LeWM and DINO-WM. Unless otherwise specified, generation inherits the Qwen3-VL-2B-Thinking checkpoint configuration (`do_sample=True`, temperature $= 1.0$, top_p $= 0.95$, top_k $= 20$). Autoregressive decoding terminates at EOS or the 512-token cap. All valid decoder hidden states produced before termination are retained and pooled together with the multimodal prefill states; no additional fixed-K truncation is applied to generated-token states. The decoded string itself is discarded and never enters the student model or the semantic loss.

Pooling, alignment, and DINO spatial features. For V-JEPA2 forecasting, Qwen decoder blocks 21, 25, and 28 (one-based) are averaged first, followed by masked mean pooling over the retained valid tokens. Student future/query tokens are mean pooled before their student-side projections. In control, the trainable teacher and student MLP projections map both sides to the same 512-dimensional alignment space. For DINO-WM, spatial patch features are mean pooled over the available history-frame and spatial-token axes only when a global semantic readout is required; the feature-channel dimension is preserved, and the native future-patch prediction objective remains patchwise. Action/state channels concatenated inside a dynamics representation are not treated as DINO visual patch channels.

Forecasting trajectory heads. EgoDex uses a learned-query 8-head attention pool over spatial tokens at each predicted time step, followed by a 1024-dimensional residual MLP and a 52×3 coordinate output. Its trajectory loss is mean MSE. The trajectory head uses AdamW with learning rate 1×10^{-3} and weight decay 10^{-4}, while the predictor/semantic-adaptor optimizer uses 1×10^{-4} and weight decay 10^{-4}. EgoPAT3D uses a 512-dimensional fused state with learned future queries and a $512 \to 512 \to 3$ (3D) or $512 \to 512 \to 2$ (2D) MLP. Its primary trajectory objective is masked Smooth-L1 loss with $\beta = 1$. All trainable EgoPAT3D components are optimized jointly with AdamW at 3.5×10^{-4} and weight decay 10^{-2}.

B.8 Forecasting Prompt Templates

The following lists the exact forecasting-side prompts used to construct the semantic caches. Line wrapping is normalized for typesetting.

EgoDex.

```
Describe this video.
```

EgoPAT3D-DT / 3D.

System message:

```
You describe egocentric hand manipulation clips accurately and concisely.
```

User message:

```
Watch this egocentric clip of a hand interacting with objects (observed
frames only). Describe the current state: hand action, contact or grasp
state, motion direction in 3D relative to the camera and nearby objects,
and the interaction target. Do not predict future motion.
```

EgoPAT3D-DT / 2D.

System message:

```
You describe egocentric hand activity on the camera image plane accurately
and concisely.
```

User message:

```
Watch this egocentric clip of a hand (observed frames only). Describe the
current state on the image plane: hand location in the frame, action,
object contact, and visible motion (left/right, up/down). Do not use depth
or 3D coordinates. Do not predict future motion.
```

B.9 Control Prompt Templates

The following reproduces the control-side user-message wording used to construct the semantic caches; line wrapping is normalized for typesetting. Reacher uses the same dynamic template for LeWM and DINO-WM, and the placeholders are filled from each observed clip's measured joint-progress metadata. Machine-readable prompt strings are also provided with the supplementary configuration.

LeWM / Push-T.

```
Analyze this Push-T manipulation scene for control-relevant geometry
and contact dynamics only.

Focus on:
- the T-shaped block pose: position, orientation, and alignment with
  the goal pose;
- the pusher position relative to the T block;
```

```
- which side or edge of the T block is being contacted or approached;
- whether the current contact would translate, rotate, or destabilize
  the block;
- whether the block is moving closer to the goal pose, rotating
  correctly, overshooting, or getting misaligned;
- whether the pusher should maintain contact, reposition, or correct
  rotation.

Ignore background, colors, lighting, texture, and irrelevant visual
 details. Return a compact description of pose alignment, contact mode,
motion progress, and likely correction. Do not describe the scene
 generally.
```

LeWM / TwoRoom.

```
You are supervising a model-based controller for the TwoRoom navigation
 task. The agent moves in a 2D image-space environment with two rooms
 separated by a wall and a doorway. The goal is to reach the target
 while avoiding the wall.

Your output must be useful for choosing the next action, not for
 describing the image appearance. Use image coordinates: left/right/
 up/down refer to what is visible in the frames.

Analyze the image sequence and infer the best short-horizon control
 guidance:
1. Navigation stage
   - same_room_to_target: agent and target are in the same room; move
     directly toward target.
   - need_doorway: target is separated by the wall; direct target motion
     is blocked, so the useful subgoal is the doorway.
   - align_doorway: agent is near the doorway but not centered; first
     align with the doorway opening.
   - cross_doorway: agent is inside or immediately at the doorway; move
     through the opening without touching the wall.
   - after_doorway_to_target: agent has crossed into the target room;
     turn directly toward target.
   - near_target: agent is close to target; use small corrective action
     or stop.
2. Action-level guidance
   - preferred_action_direction: choose one of [left, right, up, down,
     up-left, up-right, down-left, down-right, stop].
   - forbidden_action_direction: directions that would collide with the
     wall, continue pushing into the wall, or move away from the useful
     subgoal.
   - valid_subgoal: choose one of [doorway_center, doorway_entry,
     doorway_exit, target, stop].
   - wall_avoidance_rule: if the straight direction to the target is
     blocked by the wall, do not choose that direction; move toward and
     align with the doorway first.
3. Correction behavior
   - If the agent is stuck near the wall, choose an action parallel to
     the wall that moves it toward the doorway.
   - If the agent overshoots the doorway, choose the shortest corrective
     action back toward the doorway center.
   - If the agent is crossing the doorway, avoid lateral motion that hits
     either side of the wall opening.
   - After crossing, switch subgoal from doorway to target immediately.

Ignore colors, textures, lighting, rendering style, and object appearance
 unless they change navigation geometry. Focus only on agent position,
 target position, wall, doorway, and recent motion.

Return exactly these compact fields:
- current_stage:
```

```
- direct_target_path_blocked:
- valid_subgoal:
- preferred_action_direction:
- forbidden_action_direction:
- wall_avoidance_rule:
- correction:
```

LeWM and DINO-WM / Reacher.

```
You are analyzing a DeepMind Control Reacher trajectory for a two-joint
 planar arm. The task success condition is strict: shoulder/base qpos and
 elbow qpos must both match the future target qpos within 0.05 radians.
 Use the images and the numeric joint progress below to produce a compact
 control-relevant representation.

Measured joint progress from this clip:
- shoulder_delta_to_goal: {meta['shoulder_delta']:+.4f} rad
- shoulder_delta_sign: {meta['shoulder_sign']}
- shoulder_delta_magnitude: {meta['shoulder_mag']}
- elbow_delta_to_goal: {meta['elbow_delta']:+.4f} rad
- elbow_delta_sign: {meta['elbow_sign']}
- elbow_delta_magnitude: {meta['elbow_mag']}
- initial_joint_error_l2: {meta['initial_error']:.4f}
- after_action_joint_error_l2: {meta['after_error']:.4f}
- final_clip_joint_error_l2: {meta['final_error']:.4f}
- action_progress: {meta['action_progress']}
- endpoint_error_direction: {meta['endpoint_direction']}

Focus only on information that helps rank action candidates for MPC:
- which joint must rotate clockwise or counterclockwise;
- whether the observed action moves qpos toward the target or away from it;
- whether the correction should be large, medium, small, or stabilizing;
- whether the endpoint trend is consistent with reducing the joint error.

Return exactly these fields:
- shoulder_delta_sign:
- shoulder_delta_magnitude:
- elbow_delta_sign:
- elbow_delta_magnitude:
- action_progress:
- endpoint_error_direction:
- next_correction:
```

LeWM / OGBench-Cube.

```
Summarize the scene, object relations, action progress, and likely next
 event.
```

DINO-WM / Push-T.

```
The video is a chronological Push-T trajectory. The gray rigid T is the
 movable block, the translucent green T is the fixed target and never
 moves, and the blue circle is the pusher. The comparison image shows
 the earliest frame on the left and the latest frame on the right.
Compare the earliest and latest gray block poses and report only visible
 evidence. Do not confuse blue-pusher motion with gray-block motion. A
 final orientation mismatch relative to the green target alone does not
 mean that the gray block rotated over time. If uncertain, write uncertain.
Output exactly one line with at most eight words per value:
BLOCK_POSE=<gray block's final position and orientation relative to green target>
| PUSHER_RELATION=<blue pusher's final location relative to gray block and contact status>
| CONTACT_EFFECT=<actual gray block displacement or rotation across the video>
| PROGRESS=<how position and orientation mismatch to green target changed>
| TASK_PHASE=<what manipulation is visibly occurring>
```

```
| CORRECTION=<one geometry-specific next adjustment toward overlap>
Begin immediately with BLOCK_POSE=. Do not add reasoning or a preamble.
```

DINO-WM / TwoRoom.

```
You are analyzing a TwoRoom navigation trajectory for model-based
 control. The scene contains two rooms separated by a wall with a
 doorway. The agent must reach the target. Describe only information
 that helps choose future actions.

For the provided image sequence, identify:
1. Agent-target relation
   - Is the agent in the same room as the target, or separated by the wall?
   - Is the target to the left/right/up/down of the agent in image coordinates?
2. Doorway and wall constraint
   - Where is the doorway relative to the agent?
   - Is a direct path to the target blocked by the wall?
   - Does the agent need to approach the doorway before moving toward the target?
3. Progress stage
   - before doorway: agent should move toward and align with the doorway;
   - crossing doorway: agent should pass through without colliding with the wall;
   - after doorway: agent should turn directly toward the target;
   - near target: agent should make small corrective movements.
4. Motion trend and correction
   - Is the recent motion moving closer to the doorway or target?
   - Is the agent overshooting, stuck near the wall, or moving away from the
     useful path?
   - What is the most useful next correction direction?

Ignore visual style, colors, lighting, textures, and object appearance
 unless they affect room geometry or navigation. Do not describe
 irrelevant background details.

Return a compact control-oriented summary with these fields:
- room_relation:
- doorway_alignment:
- path_blocked:
- progress_stage:
- motion_trend:
- next_correction:
```

DINO-WM / OGBench-Cube.

```
Analyze this robot cube-pushing scene for control-relevant spatial
 details only.

Focus on:
- the robot end-effector / gripper position relative to the cube;
- the cube position relative to the target goal;
- whether the cube is moving closer to or farther from the target;
- whether the cube is aligned with the target or offset in x/y direction;
- whether contact is established, approaching, slipping, or lost;
- whether the recent motion suggests progress, overshoot, or correction
  is needed.
```

B.10 Optimization Details

Egocentric forecasting. EgoDex uses separate AdamW optimizers for the trajectory head and the predictor/semantic adaptor: the head uses learning rate 1×10^{-3} and weight decay 10^{-4}, while the predictor/adaptor uses 1×10^{-4} and weight decay 10^{-4}. EgoPAT3D jointly optimizes its trainable predictor, semantic adaptor, and trajectory head with AdamW at learning rate 3.5×10^{-4} and weight decay 10^{-2}. Both benchmarks use a per-GPU batch size of 4 under four-GPU distributed training,

giving an effective global batch size of 16. EgoDex training is performed on four NVIDIA GeForce RTX 5090 GPUs, whereas EgoPAT3D training is performed on four NVIDIA A800 GPUs.

LeWM-based control. For WorldPrior adaptation of LeWM, the predictor and semantic adaptor modules share a single AdamW optimizer and the same task-specific learning rate. AdamW uses weight decay 1×10^{-3} for all four control tasks. The learning rates are 5×10^{-6} for Push-T, 1×10^{-5} for TwoRoom, 2×10^{-6} for Reacher, and 5×10^{-5} for OGBench-Cube. The training batch size is 128 for all four tasks.

DINO-WM-based control. For DINO-WM adaptation, all trainable predictor/adaptor modules are optimized jointly with a single AdamW learning rate for each task, with weight decay set to zero. The learning rates are 5×10^{-5} for Push-T, 1×10^{-5} for TwoRoom, 2×10^{-5} for Reacher, and 1×10^{-5} for OGBench-Cube. The training batch size is 64 for Push-T and TwoRoom, and 32 for Reacher and OGBench-Cube. All LeWM- and DINO-WM-based control experiments are run on four NVIDIA A800 GPUs.

Control training schedule. For each of Push-T, TwoRoom, Reacher, and OGBench-Cube, both backbones use a 50-epoch native-objective Stage 1 checkpoint followed by 50 epochs of WorldPrior Stage 2 adaptation. For LeWM, Stage 2 starts from the released task-specific Stage 1 checkpoints; we do not retrain Stage 1 in our pipeline. For DINO-WM, we train the matched Stage 1 models for 50 epochs before Stage 2. In both cases, Stage 2 starts from the corresponding Stage 1 checkpoint and retains the backbone's native objective while adding semantic alignment and reference retention. For the matched comparison, each native-objective baseline is continued for the same additional 50 epochs without the WorldPrior losses. Its reported result is the best checkpoint over the matched 100-epoch training budget under the same checkpoint-selection protocol.

Across all four control tasks, the continued baseline-only Stage 2 optimization does not improve the earlier best task-success result, indicating that the native baseline optimization has largely saturated by approximately the Stage 1 horizon. Table 9 summarizes the task-specific optimizer settings and batch sizes used for the two control backbones.

Table 9: Task-specific AdamW settings and training batch sizes for the trainable predictor and semantic adaptor modules in goal-conditioned control. Within each backbone–task pair, the trainable modules share one optimizer and one learning rate.

	LeWM adaptation			DINO-WM adaptation		
Task	Learning rate	Weight decay	Batch size	Learning rate	Weight decay	Batch size
Push-T	5×10^{-6}	10^{-3}	128	5×10^{-5}	0	64
TwoRoom	1×10^{-5}	10^{-3}	128	1×10^{-5}	0	64
Reacher	2×10^{-6}	10^{-3}	128	2×10^{-5}	0	32
OGBench-Cube	5×10^{-5}	10^{-3}	128	1×10^{-5}	0	32

EgoDex and EgoPAT3D-DT use benchmark-specific training schedules, with matched V-JEPA2 baseline and WorldPrior total epoch budgets within each benchmark. On EgoDex, the V-JEPA2 baseline is trained continuously for 100 epochs, while WorldPrior uses Stage 1 in epochs 1–50 and Stage 2 in epochs 51–100. On EgoPAT3D-DT, for both the 2D and 3D settings, the V-JEPA2 baseline is trained continuously for 60 epochs, while WorldPrior uses Stage 1 in epochs 1–30 and Stage 2 in epochs 31–60. The reported baseline result for each benchmark is the best checkpoint within its matched total budget under the same checkpoint-selection protocol. In Stage 1, WorldPrior follows the baseline optimization. Stage 2 starts from the Stage 1 checkpoint and activates multimodal-semantic alignment and reference retention while retaining the latent-prediction and trajectory objectives. On both benchmarks, continuing baseline-only optimization beyond the respective Stage 1 horizon does not improve the earlier best checkpoint.

For EgoDex, the downstream trajectory head retains its original Stage 2 MultiStepLR schedule, with local milestones at epochs 16 and 33 and decay factor $\gamma = 0.1$. This scheduler is applied only to the EgoDex trajectory-head optimizer and does not modify the fixed predictor/adaptor learning rates

reported above. FP16 automatic mixed precision with automatic casting and gradient scaling is enabled for the egocentric forecasting experiments.

Training time and compute. We additionally report approximate wall-clock training time for one complete run of each stage under the hardware configurations specified above. These measurements document computational cost and are not intended as normalized speed comparisons across different GPU types. For EgoDex, Stage 1 requires approximately 3 hours and Stage 2 approximately 4 hours on four NVIDIA GeForce RTX 5090 GPUs. For EgoPAT3D-DT, Stage 1 requires approximately 1.5 hours and Stage 2 approximately 2 hours on four NVIDIA A800 GPUs; the 2D and 3D configurations have similar wall-clock cost.

For LeWM-based control, Stage 1 is not retrained in our pipeline: we initialize from the publicly released task-specific LeWM checkpoints for Push-T, TwoRoom, Reacher, and OGBench-Cube and therefore report only the Stage 2 WorldPrior adaptation time. For DINO-WM, we train the matched Stage 1 baseline and then perform Stage 2 WorldPrior adaptation, so both stage-wise and total wall-clock costs are reported. All control timings are measured on four NVIDIA A800 GPUs. Table 10 summarizes the resulting stage-wise wall-clock costs.

Table 10: Approximate wall-clock training time per run under our experimental pipeline. LeWM control uses released Stage 1 checkpoints and therefore reports only Stage 2 adaptation time; for DINO-WM, Total is the sum of Stage 1 baseline training and Stage 2 WorldPrior adaptation.

Setting	Task / benchmark	Stage 1	Stage 2	Total	Hardware
Forecasting	EgoDex	∼3 h	∼4 h	∼7 h	4× RTX 5090
	EgoPAT3D-DT (2D/3D)	∼1.5 h	∼2 h	∼3.5 h	4× A800
LeWM control	Push-T	Released ckpt.	∼7.3 h	—	4× A800
	TwoRoom	Released ckpt.	∼2.7 h	—	4× A800
	Reacher	Released ckpt.	∼6.7 h	—	4× A800
	OGBench-Cube	Released ckpt.	∼7.7 h	—	4× A800
DINO-WM control	Push-T	∼13.3 h	∼16.0 h	∼29.3 h	4× A800
	TwoRoom	∼3.7 h	∼4.3 h	∼8.0 h	4× A800
	Reacher	∼16.0 h	∼19.3 h	∼35.3 h	4× A800
	OGBench-Cube	∼18.3 h	∼22.0 h	∼40.3 h	4× A800

C Qualitative Forecasting and Control Results

C.1 EgoDex Trajectory Examples

Figure 4 complements the aggregate EgoDex results with qualitative trajectory overlays. The color gradient from earlier (blue) to later (red) encodes the temporal progression of the predicted trajectories from the current observation into the future. In Scene 1 (stool assembly/disassembly), WorldPrior reduces ADE/FDE from 4.8/5.1 cm to 4.0/3.4 cm. In Scene 2 (cabinet insertion/removal), it achieves the same ADE as V-JEPA2 (3.5 cm) while reducing FDE from 3.2 cm to 2.4 cm. Across both examples, V-JEPA2 exhibits noticeable abrupt deviations and temporal discontinuities, whereas WorldPrior produces smoother and more coherent trajectories that are more consistent with the continuous evolution of natural hand motion. These examples qualitatively support the aggregate results rather than serving as standalone evidence.

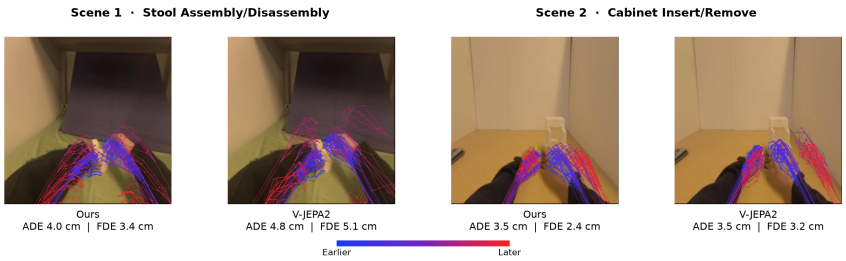
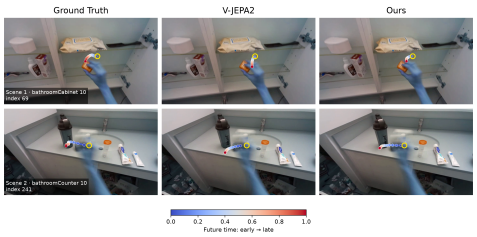


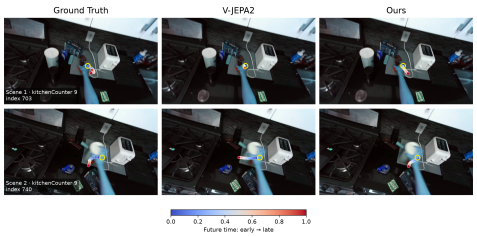
Figure 4: Qualitative EgoDex trajectory forecasts for representative validation clips. WorldPrior and the matched V-JEPA2 baseline receive the same observed clip and use the same prediction horizon. Trajectory color progresses from earlier to later future steps. The examples illustrate case-level behavior and are not used to compute the aggregate results in Tables 1 and 2.

C.2 EgoPAT3D 2D and 3D Examples

Figure 5 shows that WorldPrior more closely preserves the ground-truth motion direction, curvature, and terminal location in both evaluation spaces. In the 2D examples, it reduces the directional and endpoint deviations visible in V-JEPA2 predictions. The projected 3D examples exhibit the same pattern, including cases in which the baseline follows an incorrect lateral direction. This cross-view consistency qualitatively agrees with the lower ADE/FDE in Table 3; the displayed clips remain illustrative rather than a substitute for aggregate evaluation.



(a) Direct prediction in the normalized 2D image plane.

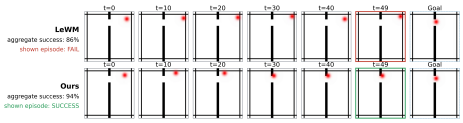


(b) Metric 3D prediction projected onto the image for visualization.

Figure 5: Qualitative EgoPAT3D-DT trajectory comparison in the two evaluation spaces. Each panel shows the ground truth, the matched V-JEPA2 prediction, and the WorldPrior prediction for identical observations. Trajectory color progresses from early to late future steps. Panel (b) visualizes the projected trajectory, while the 3D errors in Table 3 are computed in metric camera coordinates.

C.3 Goal-Conditioned Control Examples

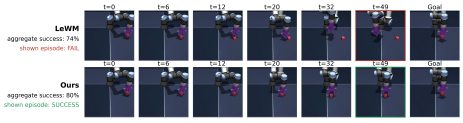
Figure 6 illustrates representative task-dependent outcomes of WorldPrior adaptation. In TwoRoom, WorldPrior produces a trajectory that passes through the doorway and avoids the wall, corresponding to a successful rollout under the adapted model. This qualitative example does not establish why the CEM planner selects a more successful action sequence. In OGBench-Cube and Reacher, the adapted model reaches the target configuration in the displayed cases where LeWM fails. Both methods solve the displayed Push-T episode, consistent with their identical aggregate success rate. These examples visualize representative behavioral differences and provide qualitative context for the aggregate control results, but they do not establish the mechanism responsible for those differences.



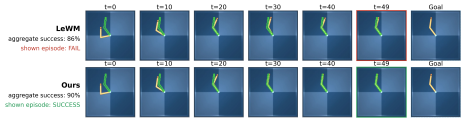
(a) TwoRoom



(b) Push-T



(c) OGBench-Cube



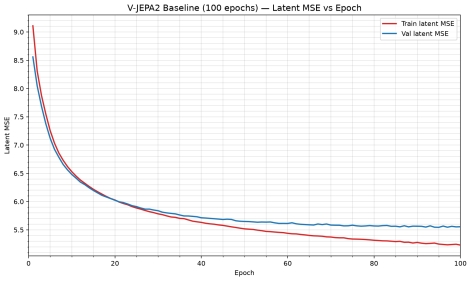
(d) Reacher

Figure 6: Representative goal-conditioned control rollouts under the matched evaluation protocol. In each panel, the top row shows the reproduced LeWM baseline and the bottom row shows WorldPrior; columns progress chronologically toward the goal image at right. Red and green borders mark failure and success, respectively. The displayed episodes provide qualitative context for the aggregate success rates in Table 5 and are not an additional statistical evaluation.

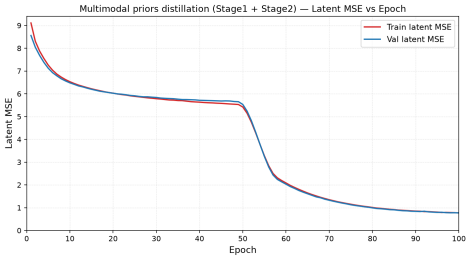
D Optimization Dynamics

Figure 7 compares training and validation latent MSE over the same 100-epoch budget on EgoDex. During the first 50 epochs, WorldPrior follows the matched Stage 1 optimization. Under continued baseline training, V-JEPA2 training MSE continues to decrease while validation MSE approaches a plateau and the train–validation gap widens. We interpret this as diminishing held-out benefit from additional baseline-only optimization, not as conclusive evidence of overfitting. This behavior is consistent with the matched-budget checkpoint analysis used throughout the paper: on EgoDex, EgoPAT3D-DT, and the four control tasks, extending the baseline beyond approximately the Stage 1 horizon does not produce a better selected baseline result.

When WorldPrior adaptation is activated at epoch 51, the training and validation latent MSE undergo a renewed and sustained reduction while remaining closely coupled. Because the original latent-prediction and trajectory objectives remain active, this behavior is consistent with the semantic-alignment and retention objectives adding complementary optimization signal rather than replacing the baseline loss. Taken together, these optimization dynamics indicate that WorldPrior introduces useful multimodal priors that provide complementary learning signals for further refining the latent predictive representation beyond baseline-only optimization. The contribution of the individual components is quantified by the ablation results in Table 6 of the main paper.



(a) V-JEPA2 baseline trained for 100 epochs.



(b) WorldPrior: Stage 1 (1–50) and Stage 2 (51–100).

Figure 7: Latent-MSE dynamics over 100 epochs. The V-JEPA2 validation curve shows diminishing improvement after epoch 50, whereas WorldPrior exhibits a renewed decrease after WorldPrior adaptation is activated at epoch 51.

E Extended Analysis and Limitations

Multimodal priors reshape task-relevant predictive representations. Across EgoDex, the most pronounced gains from WorldPrior appear in trajectory ADE/FDE and threshold accuracy, whereas FD, SL1, and CD improve more moderately. This pattern suggests that the benefit of WorldPrior cannot be explained solely by reducing a particular latent-space discrepancy. Instead, the transferred multimodal prior appears to reorganize the predictive representation so that downstream-relevant structure becomes more accessible. The control results reinforce this interpretation: Reacher and OGBench-Cube exhibit lower five-step NMSE under the LeWM instantiation, Push-T remains statistically unchanged, and TwoRoom achieves higher closed-loop success despite a larger average latent NMSE. Taken together, these results indicate that WorldPrior primarily improves the *task relevance and organization* of predictive representations, rather than uniformly improving every measure of fine-grained dynamics fidelity.

Observation-only supervision transfers priors without future leakage. The VLM receives only the observation window. No future frame, target latent, trajectory label, or task-success signal is exposed to the teacher branch. Consequently, the training-only multimodal supervision cannot directly encode ground-truth future information and must instead provide additional structure extracted from the same observed evidence available to the predictive model. The cached teacher representation is detached during adaptation, and the VLM is completely removed at inference. Any deployment-time improvement must therefore be carried by the adapted visual predictive pathway rather than by future-information leakage or test-time VLM conditioning.

Semantic enrichment and dynamics preservation are complementary. WorldPrior is designed around a natural tension between high-level multimodal abstraction and backbone-specific predictive detail. $\mathcal{L}_{\text{sem}}$ encourages internal states to expose object-level, relational, and contextual structure, whereas the native backbone objective preserves the original predictive learning problem. Reference retention provides an additional functional constraint on how far WorldPrior adaptation may move the predictive mapping from its pre-adaptation behavior. The component-removal results in Table 6 are consistent with this interpretation: removing either semantic alignment or reference retention degrades all six EgoDex metrics while the native latent-prediction and trajectory objectives remain active. We therefore interpret P_{ref} not as a second semantic teacher, but as a dynamics-preserving functional anchor.

Prediction fidelity is not a monotonic proxy for decision utility. Latent NMSE averages prediction error across rollout steps and latent dimensions, whereas closed-loop task success depends on the interaction between the learned representation, the predictive model, and the downstream planner. TwoRoom provides the clearest example: average latent NMSE increases while success improves from 86% to 94%. This establishes a non-monotonic relationship between global latent prediction error and closed-loop success in our evaluation, but does not by itself determine the underlying mechanism. One possible explanation is that adaptation alters aspects of the latent geometry used by CEM to compare candidate action sequences; alternative factors, including latent norm or scale changes and evaluation variability, may also contribute. We do not directly measure candidate-ranking fidelity in the present experiments. Conversely, Push-T exhibits a small, nonsignificant NMSE reduction without a corresponding success gain, further illustrating that small changes in global latent discrepancy need not translate into changes in closed-loop performance. Reporting both predictive and closed-loop metrics is therefore important.

Backbone transferability exhibits task-dependent effects. The smaller four-task average gain for DINO-WM should not be interpreted as uniformly weaker transfer. Its reproduced TwoRoom baseline is already saturated at 100%, leaving no room for further improvement; over the remaining three tasks, both DINO-WM and LeWM obtain a mean net gain of 3.3 pp. The task-wise responses nevertheless differ: WorldPrior improves DINO-WM from 74% to 82% on Reacher and from 74% to 78% on Push-T, preserves TwoRoom at 100%, and changes OGBench-Cube from 84% to 82%, raising the four-task average from 83.0% to 85.5%. The overall improvement across a qualitatively different representation family therefore supports cross-backbone transferability, while the heterogeneous task-level responses indicate that the effect of multimodal adaptation depends on the receiving representation and task.

One possible source of this task dependence is the geometry of the receiving representation. LeWM learns a compact latent representation end-to-end from pixels under its predictive objective and SIGReg (Maes et al., 2026), whereas DINO-WM operates on frozen pretrained DINOv2 spatial patch features and learns dynamics by predicting their future evolution (Zhou et al., 2025). These architectural differences imply that the same globally pooled multimodal supervision is applied to substantially different latent structures. It is therefore plausible that semantic adaptation interacts differently with compact end-to-end representations and spatially structured pretrained features, potentially changing the relative emphasis placed on high-level relational information and fine-grained spatial information.

The small OGBench-Cube decrease motivates, but does not establish, a possible semantic–geometric interaction. Pooled upper-layer VLM representations emphasize object-level, relational, and contextual information, whereas the spatial patch representation used by DINO-WM retains localized visual structure that may be important for precise object-pose and gripper–object reasoning. However, our current experiments do not directly measure whether geometry-sensitive information is preserved or degraded after adaptation, and the 2-pp change on OGBench-Cube is too small to support a causal interpretation on its own. Moreover, WorldPrior improves the contact-rich Push-T task, indicating that any interaction between semantic supervision and spatial geometry is not uniformly detrimental. We therefore regard the semantic–geometric explanation as a mechanistic hypothesis for future investigation rather than an established explanation of the observed task-level differences.

Limitations. Our current cross-backbone evaluation covers LeWM and DINO-WM, and extending the study to additional world-model families would provide stronger evidence of architectural generality. We also use a single observation-side VLM teacher in the present work; broader evaluation across multimodal teachers, prompting strategies, alignment-layer choices, cache construction procedures, and semantic-alignment weights would further characterize the robustness of the approach. While independent repetitions with additional random seeds would provide a stronger characterization of run-to-run variability, exhaustively repeating the full multi-benchmark and cross-backbone evaluation suite would incur substantial additional computational cost. We therefore leave a more comprehensive seed-level variance analysis to future work. Finally, WorldPrior currently focuses on strengthening the predictive representations of latent world models rather than directly learning action-generative embodied policies. Extending the same training-time semantic internalization principle to Vision-Language-Action (VLA) and World-Action Model (WAM) architectures is a promising direction for connecting multimodal representation enhancement more directly to action generation and closed-loop embodied decision making.