CONTROLLABILITY OF PRETRAINED VISUAL REPRESENTATIONS UNDER START-CONTROLLED EVALUATION ON MULTI-OBJECT SCENES

Anonymous authors
Paper under double-blind review

ABSTRACT

Turning pretrained visual representations into behavior requires a procedure for sequential decision making, such as test-time search (CEM) via model-predictive control in the latent space or a goal-conditioned inverse-dynamics model (GC-IDM). We contribute a start-controlled evaluation protocol, an extended evaluation suite with progressively greater object counts, and a long-horizon predictive architecture for learning visual representations. Our start-controlled protocol shows that test-time search methods fail on 3D cube manipulation (0.0% success rate) and that GC-IDM still succeeds (68.9–84.9% success rate). Our extended evaluation suite scales the cube count up to four cubes, revealing that prior work's (LeWM and Sub-JEPA) representations become insufficient for downstream control under GC-IDM. The prior work's random-start evaluation protocol hides both issues, inflating LeWM's four-cube performance from 17.1% to 72.6%. In contrast, on four-cube tasks our proposed architecture learns representations that outperform those of LeWM and Sub-JEPA by up to 20.4 and 35.6 percentage points respectively. Our architecture predicts 50 environment steps of future latents in one pass and, through a structured attention mask, simultaneously learns inverse dynamics from its internal representations of consecutive states; a patch decoder maps each latent back to the encoder's patch representation, retaining object-level detail. Our representations are also more robust to colour distractors in one-cube pick-and-place. Ablations show that our method depends strongly on the training horizon, inverse-dynamics loss, and patch representation reconstruction loss, with the inverse-dynamics term alone worth a third to two fifths of control performance.

1 INTRODUCTION

Pretrained visual representations carry modern robot policies (Brohan et al., 2023; Ghosh et al., 2024). Typically, large vision–language modules process long patch-token sequences from multiple cameras (Zitkovich et al., 2023; Kim et al., 2025; Black et al., 2025a). Perception dominates both parameter count and inference cost, so policy inference cannot match action frequency: π_0 commands a robot at 50 Hz (Black et al., 2025a) but achieves 1–30 Hz on consumer GPUs. It closes that gap by predicting a second of actions and executing them open-loop (Zhao et al., 2023; Black et al., 2025a); real-time chunking shortens that window without removing it (Black et al., 2025b). A perception stack small enough to enable closed-loop reactive control is desirable.

Joint-embedding predictive architectures (JEPAs) spend model capacity differently, predicting representations rather than pixels. They learn to encode the information that makes a scene predictable rather than optimizing for pixel-level reconstruction (Assran et al., 2023; Bardes et al., 2024). Many architectures in this family are large: V-JEPA 2 runs to a billion parameters (Assran et al., 2025). Recent JEPA architectures instead optimize an encoder and an action-conditioned world model end-to-end, predicting the future representation (Maes et al., 2026b; Zhao et al., 2026). With only a 15M-parameter model they learn a compact global representation of the visual observation that can be used for downstream control. The branch we study is this compact one: how well such a latent supports control is what we measure.

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.

LeWM (Maes et al., 2026b) plans up to $48\times$ faster than foundation-model world models such as DINO-WM (Zhou et al., 2025), which trains a world model on the patch embeddings of a pretrained DINOv2 encoder (Oquab et al., 2024). These compact end-to-end JEPA architectures typically achieve control through test-time search (Rubinstein, 1999; Pinneri et al., 2021; Williams et al., 2015). Recent work has shown that planning can be learned from these representations via amortization (GC-IDM), removing test-time search entirely and giving a low-latency, compute-efficient controller (Nguyen et al., 2026). That capability, however, has been shown only at small scale: these methods are evaluated on four simulated environments with one or two objects (Maes et al., 2026b; Zhao et al., 2026; Nguyen et al., 2026), while VLAs train on a million real-robot episodes across cluttered scenes (Kim et al., 2025). Recent JEPA work reaches VLA scale only through a large pretrained V-JEPA 2 encoder (Assran et al., 2025), 300M to 1B parameters (Sun et al., 2026; Miao et al., 2026). Compact end-to-end JEPA variants remain untested in realistic robotic scenes. What we set out to measure, and then improve, is whether a low-dimensional, global state can be not merely predictive but *actionable* as scenes grow in complexity.

Controllability, or the success of leveraging a frozen vision representation for downstream control tasks, is a property of the representation together with the action generation method. Changing only the evaluation protocol, from that of prior works (Maes et al., 2026b; Zhao et al., 2026; Nguyen et al., 2026) to our start-controlled protocol, drastically changes our conclusions. The prior protocol starts evaluation from a random mid-episode state, in which the task is often already solved. Our protocol reveals that every test-time search planner attains a 0.0% success rate on OGB-Cube. Because prior works evaluate only on simplified single-object tasks, how a compact latent copes as objects are added remains unknown. Our work shows that controllability collapses as scenes gain objects. On pick-and-place, LeWM falls from 80.1% to 17.1% as the number of cubes increases from one to four. The standard random-start protocol conceals precisely this performance loss, reporting 72.6% on the same task.

We propose a network architecture that learns representations aimed at closing this gap. It predicts 50 environment steps in a single pass, uses a structured attention mask to simultaneously learn inverse dynamics from its internal representations of consecutive states, and trains a patch decoder that preserves object-level detail in the compressed state (Figure 1). With four objects under the start-controlled protocol, this raises control success over LeWM from 17.1% to 37.5% on pick-and-place, from 3.2% to 17.4% on stacking, and from 18.0% to 24.4% on cyclic rearrangement. The same frozen latent supports a linear probe that recovers the four cube positions with $R^2 = 0.85$, compared with 0.45 for LeWM.

Contributions.

- A start-controlled evaluation protocol for controllers built on frozen representations, and the conclusions it overturns.
- An extended multi-object evaluation suite and analysis measuring the degradation of representation controllability as scene complexity grows.
- A long-horizon predictive architecture that outperforms LeWM by up to 20.4 points and Sub-JEPA by up to 35.6 points on four-object tasks.
- Ablations isolating what features of our architecture make our latent controllable: the training horizon, the inverse-dynamics term and the patch decoder.

2 Related Work

Latent-predictive representations. Joint-embedding predictive architectures train an encoder and a predictor together so that a masked or future view is predicted in latent space rather than in pixels (Assran et al., 2023; Bardes et al., 2024; Assran et al., 2025). The recurring difficulty is representation collapse, answered by asymmetric targets, variance and covariance penalties, Gaussian sketches of the embedding distribution, a frozen pretrained encoder, or object-level structure (Grill et al., 2020; Bardes et al., 2022; Zbontar et al., 2021; Maes et al., 2026b; Balestriero & LeCun, 2025; Zhao et al., 2026; Zhou et al., 2025; Nam et al., 2026). Such objectives can be satisfied without capturing what matters for acting: Sobal et al. (2022) drive a JEPA loss to zero by copying fixed background while discarding the object an action controls. Recent work extends the backbone

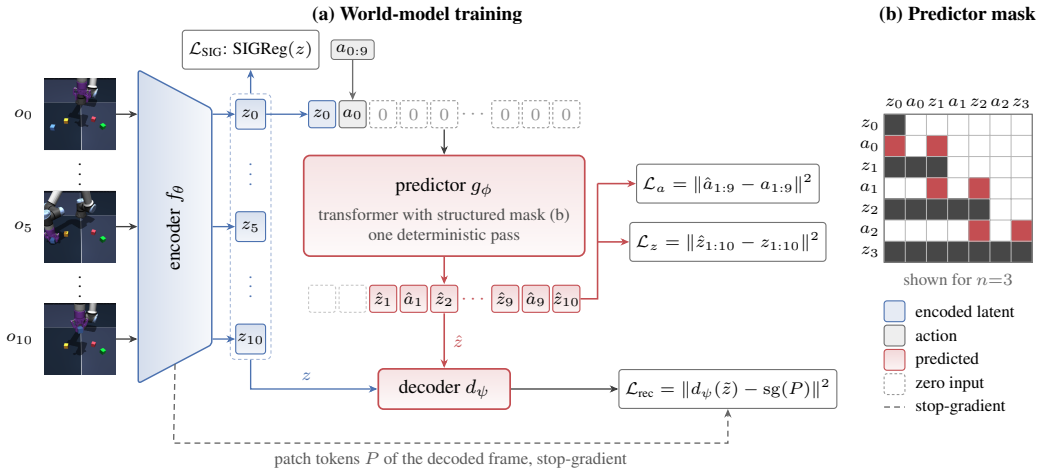


Figure 1: **Our world model. (a)** The training pass: eleven frames are encoded to 192-d latents, the predictor g_ϕ conditions on (z_0, a_0) alone and regresses every remaining latent and action in one pass, and the decoder d_ψ reconstructs patch tokens from both encoded and predicted latents against a stop-gradient target. **(b)** Attention mask of g_ϕ: states attend causally, each action token only to the two states that bracket it. Only f_θ is kept at test time; see §3.

in directions we also take, training against several steps of prediction rather than one (Gao & Xu, 2026; Liu et al., 2026) and decoding the executed action from the latent displacement between observations (Zhang et al., 2026); we read both out of a single pass. Our objects of study are LeWM (Maes et al., 2026b) and Sub-JEPA (Zhao et al., 2026), which share a compact global latent and a released checkpoint.

From a frozen latent to actions. A frozen latent is not a policy. The default procedure is test-time search in latent space, typically the cross-entropy method and its relatives (Rubinstein, 1999; Pinneri et al., 2021; Williams et al., 2015), and this is how latent-predictive world models are usually scored (Maes et al., 2026b; Sobal et al., 2025); Bai & Xiong (2026) replace the latent-Euclidean cost that such search minimises. Nguyen et al. (2026) instead amortize the search, mapping current latent, goal latent and remaining horizon straight to an action at a hundredfold lower per-decision cost, and predicting the action between two states is a representation-learning objective in its own right (Brandfonbrener et al., 2023). We adopt that amortized policy and study when it stops working. This differs from the older tradition of learning dynamics and planning inside the model (Ha & Schmidhuber, 2018; Hafner et al., 2020; 2025; Hansen et al., 2022), where what is evaluated is the return of a policy learned inside the model, not whether a frozen state supports control.

How any of this is evaluated. Offline benchmarks supply the tasks (Park et al., 2025; Liu et al., 2023; Fu et al., 2020), platforms standardise the pipeline (Maes et al., 2026a) and learned simulators replace the environment outright (Quevedo et al., 2025), but how they are scored is consequential: Park et al. (2025) report that evaluating against a single state-goal pair rather than many can reverse the ordering of two algorithms. None of the methods above is scored from the episode start; those using LeWM's benchmark adopt its protocol unchanged (Zhang et al., 2026; Gao & Xu, 2026; Bai & Xiong, 2026). A separate line probes what a latent encodes (Zhang, 2026; Molinari et al., 2025), and robustness analyses show success under nuisance perturbations falling far below headline numbers (Fei et al., 2026). Probing is rarely connected back to control, and that link is what this paper measures.

3 Method

We learn a representation from offline trajectories of frames and actions, and keep only the encoder. A frame o_t is mapped by a ViT-Tiny/14 encoder at 224 px (12 layers, hidden width 192, trained from scratch) and a projector MLP $192 \to 2048 \to 192$ with BatchNorm to a single vector $z_t =$

projector(CLS) $\in \mathbb{R}^{192}$. A predictor g_ϕ and a decoder d_ψ are used during training and discarded afterwards. Everything downstream consumes z alone: no patch tokens leave the encoder at test time, and the entire state a policy sees is one 192-d vector per frame.

A training example is 11 consecutive frames at frameskip 5, the prediction window spanning 50 environment steps. Actions are frameskip-stacked, so each of the ten action slots carries $5 \times N_a$ dimensions, where N_a is the environment's action dimension. The window is laid out as 21 interleaved tokens $(z_0, a_0, z_1, a_1, \ldots, a_9, z_{10})$ and processed by six AdaLN-zero transformer blocks (8 heads of width 64, MLP 2048). The predictor conditions on *one* state–action step, (z_0, a_0); every other input position is zero. From that single anchor it regresses all ten future latents $\hat{z}_{1:10}$ and the nine intervening actions $\hat{a}_{1:9}$ in one deterministic forward pass. What makes the two predictions compatible is the attention mask (Figure 1b): state tokens attend causally over the interleaved sequence, while each action token attends only to the two states that bracket it. Forward dynamics and inverse dynamics over its internal state representations are therefore learned in the same pass.

A convolutional decoder d_ψ (a linear map followed by three transposed convolutions of width 256) takes a 192-d latent back to the encoder's 16×16 patch grid. Its target is the encoder's own patch tokens P for the frame that latent corresponds to, taken with a stop-gradient $\mathrm{sg}(\cdot)$, so the encoder cannot make the reconstruction easy by moving the target. Four latents $\tilde{z}$ are sampled per batch from the union of the eleven encoded and ten predicted latents, $\tilde{z} \sim \{z_{0:10}\} \cup \{\hat{z}_{1:10}\}$, which costs a fraction of decoding every step and pushes encoded and predicted states into one decodable space.

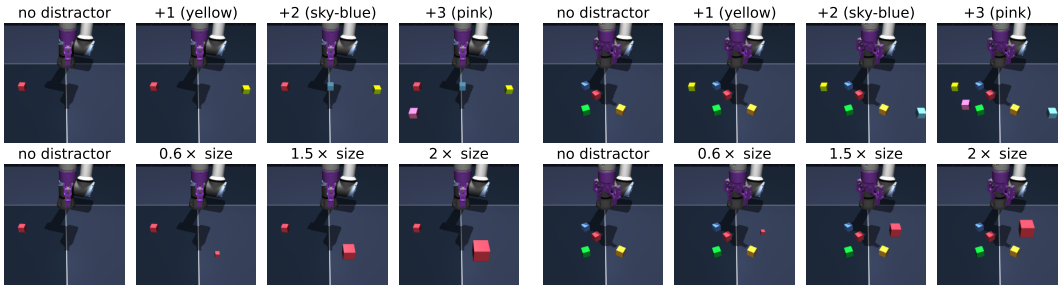
Training objective. We minimise

$$\mathcal{L} = \underbrace{\|\hat{z}_{1:10} - z_{1:10}\|^2}_{\mathcal{L}_z} + \underbrace{\|\hat{a}_{1:9} - a_{1:9}\|^2}_{\mathcal{L}_a} + \lambda_{\text{rec}} \underbrace{\|d_\psi(\tilde{z}) - \mathrm{sg}(P)\|^2}_{\mathcal{L}_{\text{rec}}} + \lambda_{\text{SIG}}\, \text{SIGReg}(z), \qquad (1)$$

with $\lambda_{\text{rec}} = 1$ and $\lambda_{\text{SIG}} = 0.09$. SIGReg is an Epps–Pulley normality statistic evaluated on 1024 random one-dimensional projections of the ambient 192-d latent, and prevents the representation from collapsing. The prediction targets are *not* stop-gradient, so the encoder receives gradient from all four terms. Further training details are given in Appendix A.

4 Experimental Setup

We run two primary sets of experiments. The first re-evaluates the prior work on its own protocol and under start-controlled evaluation (with all goal target offsets at 50 steps) to measure the impact of our evaluation protocol. We take LeWM, Sub-JEPA and GC-IDM on the four environments those papers report (OGB-Cube, Push-T, Two-Room and Reacher) and run them inside the official `stable-worldmodel` harness with the released LeWM and Sub-JEPA encoder weights and inference code; GC-IDM is trained from scratch with the authors' released training code. Neither the representations nor the planners are altered, so the only variable is where an episode begins. We fix the offset at 50 because at an offset of 25, OGB-Cube reports 100% success under start-controlled evaluation. That isolates the question of which *controller* converts a frozen latent into actions. We



(a) `pnp1` (b) `pnp4`

Figure 2: **The distractor conditions.** Top row of each panel: the colour axis; bottom row: the size axis. Within a panel every image is the same episode at the same frame. Placement is drawn once per axis, the colour axis is nested ($c_1 \subset c_2 \subset c_3$) and the three size cells put the distractor cube in the same location.

evaluate the four test-time search planners shipped with `stable-worldmodel` (CEM, iCEM, MPPI and gradient descent on the action sequence) against a single amortized policy. These results are Table 1 and §5.1. The second set is the multi-object study, which fixes the controller to GC-IDM and varies the representation and the scene (Table 2, §5.2).

We define a progressive multi-object study on the OGBench cube suite, to measure the controllability of each representation-learning algorithm as the cube count rises to four. We evaluate three task categories (pick-and-place, stacking and cyclic rearrangement) at two, three and four cubes, plus one-cube pick-and-place as a saturation anchor, giving ten tasks in total (Table 2). We visualise the four-object tasks from the start state to the goal in Appendix B (Figure 6). Episodes are a few hundred frames long (352 at four-object pick-and-place, 443 at four-object cyclic rearrangement). Task success follows the same definition as the prior work (Maes et al., 2026b; Zhao et al., 2026): the episode succeeds if the scene reaches the goal state.

For the multi-object study we compare six frozen encoders, five of them trained from scratch per task over 10,000 episodes of expert data: LeWM, Sub-JEPA, DINOv2-S, Ours, Ours without patch reconstruction, and LeWM with patch reconstruction. LeWM and Sub-JEPA are retrained with their original published hyper-parameters. DINOv2-S, whose 384-d CLS token stands in as a control, is not trained on the expert data. Encoders are never fine-tuned by the downstream GC-IDM policy.

Every encoder is trained for 200k optimisation steps to learn the visual representations used for the GC-IDM policy, which is trained for 100 epochs. Every cell reports control success over 200 episodes at goal offset 50 steps, repeated across four evaluation seeds as in every experiment we report. Each seed fixes its 200 start states and goals once, and every encoder is evaluated on that identical set; we report the mean $\pm$ standard deviation over those seeds. The horizon and the inverse-dynamics loss are ablated at the same settings in §6 (Figure 5).

To further analyse our architecture we run two sets of experiments that do not measure controllability directly but indicate representation quality: distractor robustness (Table 3, §5.3) and decodability (Figure 3, §5.4). For distractor robustness we hold the task fixed and add only task-irrelevant objects: one to three distractors in colours the task never uses, and a single distractor in the task cube's own colour at 0.6, 1.5 and $2\times$ its size (Figure 2). The task cubes' trajectories are byte-identical across conditions, so a column is exactly paired with its reference: frozen encoders and the policies are evaluated in the altered scenes. For decodability we read three things off the frozen latent. A linear probe to every cube's (x, y, z) and the end-effector; a 512–512 regression MLP to the same targets, and a pixel decoder trained on that same 192-d vector, which shows what survives of the observation itself (Figure 4). Probe and MLP use an episode-level 80/20 split and are scored on the held-out part. Each target coordinate is standardised to zero mean and unit variance over the sampled frames of its task.

5 Results

5.1 Search-based control collapses on 3D manipulation

Measured from the true episode start, every test-time search planner scores exactly 0.0% on OGB-Cube, on both representations, while the amortized GC-IDM policy solves the same task at 84.9% (Sub-JEPA) and 68.9% (LeWM). Under the random mid-episode start the same planners report up to 59.4%, so the search success reported in prior work is an artefact of where the episode begins. A policy that outputs zero actions already succeeds on 25.3% (pnp1), 12.9% (pnp4) and 13.9% (cyclic4) of random-start episodes, part of the success that protocol credits to the controller. The gap is not a constant a reader could subtract: inflation runs from $+59.4$ on OGB-Cube to -16.4 on Reacher, so the protocol changes the ordering of methods and not only their level (Table 1).

The collapse of test-time search is not an under-tuned search artefact. The solvers run LeWM's search configuration (Maes et al., 2026b), which drives the latent planning cost to near zero (Figure 8) without moving the block. A hyper-parameter sweep over the search budget (18 configurations, 3,000 to 50,000 candidate rollouts per step, a $16\times$ range containing the 9,000 we report) leaves success at exactly 0% and net block displacement at 0.0 cm in every trial, while the latent cost the optimiser minimises falls from 2.9 to 0.12 (App. C.1). The limiter is the latent-distance objective and the learned predictor it is optimised against, not the search budget.

Table 1: **Baseline representation success: start-controlled vs. random-start evaluation.** *infl.* is random minus $t{=}0$ (start-controlled). The best control method per representation under $t{=}0$ is shown in bold. Both protocols use goal offset 50.

	Sub-JEPA			LeWM		
Method	$t{=}0$	rand.	infl.	$t{=}0$	rand.	infl.
OGB-Cube (3D manipulation)						
GC-IDM	$\mathbf{84.9} \pm 1.8$	95.2 ± 1.2	$+10.4 \pm 1.8$	$\mathbf{68.9} \pm 1.7$	98.2 ± 0.6	$+29.4 \pm 1.2$
CEM	0.0 ± 0.0	55.9 ± 4.0	$+55.9 \pm 4.0$	0.0 ± 0.0	51.6 ± 5.3	$+51.6 \pm 5.3$
iCEM	0.0 ± 0.0	59.4 ± 2.9	$+59.4 \pm 2.9$	0.0 ± 0.0	52.0 ± 2.8	$+52.0 \pm 2.8$
MPPI	0.0 ± 0.0	37.0 ± 1.5	$+37.0 \pm 1.5$	0.0 ± 0.0	38.1 ± 1.3	$+38.1 \pm 1.3$
SGD	0.0 ± 0.0	22.4 ± 2.4	$+22.4 \pm 2.4$	0.0 ± 0.0	13.6 ± 1.2	$+13.6 \pm 1.2$
Push-T (2D pushing)						
GC-IDM	$\mathbf{86.6} \pm 1.8$	83.6 ± 0.4	-3.0 ± 2.1	$\mathbf{85.2} \pm 2.7$	82.8 ± 2.5	-2.5 ± 3.9
CEM	19.1 ± 1.7	38.9 ± 2.9	$+19.8 \pm 3.7$	17.5 ± 2.6	35.0 ± 2.3	$+17.5 \pm 4.3$
iCEM	9.2 ± 0.6	32.8 ± 3.4	$+23.5 \pm 3.2$	9.1 ± 1.5	27.4 ± 2.6	$+18.2 \pm 3.2$
MPPI	14.2 ± 2.4	22.4 ± 2.9	$+8.1 \pm 4.5$	10.8 ± 3.0	19.8 ± 3.8	$+9.0 \pm 3.8$
SGD	0.2 ± 0.4	0.2 ± 0.2	$+0.0 \pm 0.6$	0.0 ± 0.0	0.6 ± 0.5	$+0.6 \pm 0.5$
Two-Room (2D navigation)						
GC-IDM	$\mathbf{100.0} \pm 0.0$	100.0 ± 0.0	$+0.0 \pm 0.0$	$\mathbf{100.0} \pm 0.0$	100.0 ± 0.0	$+0.0 \pm 0.0$
CEM	51.2 ± 3.3	53.9 ± 5.4	$+2.6 \pm 8.5$	38.0 ± 2.7	47.8 ± 2.1	$+9.8 \pm 2.7$
iCEM	56.4 ± 2.6	58.4 ± 5.2	$+2.0 \pm 7.1$	43.6 ± 2.9	57.6 ± 3.5	$+14.0 \pm 2.6$
MPPI	26.8 ± 4.4	29.5 ± 2.8	$+2.8 \pm 5.2$	26.1 ± 1.3	29.0 ± 2.5	$+2.9 \pm 1.8$
SGD	18.0 ± 2.6	20.1 ± 2.4	$+2.1 \pm 1.1$	13.4 ± 2.0	16.2 ± 2.5	$+2.9 \pm 2.5$
Reacher (2D reaching)						
GC-IDM	$\mathbf{99.6} \pm 0.4$	99.6 ± 0.4	$+0.0 \pm 0.6$	$\mathbf{99.8} \pm 0.2$	99.5 ± 0.6	-0.2 ± 0.8
CEM	92.1 ± 0.5	75.8 ± 3.2	-16.4 ± 3.5	91.8 ± 2.1	76.9 ± 3.9	-14.9 ± 1.8
iCEM	93.4 ± 1.5	81.5 ± 2.5	-11.9 ± 3.8	93.8 ± 0.8	81.4 ± 1.4	-12.4 ± 0.7
MPPI	38.6 ± 2.3	38.2 ± 1.2	-0.4 ± 2.6	35.5 ± 1.8	34.8 ± 3.7	-0.8 ± 2.8
SGD	4.4 ± 0.4	4.6 ± 1.6	$+0.2 \pm 1.6$	4.1 ± 0.6	4.0 ± 2.0	-0.1 ± 1.6

Visually inspecting the CEM rollouts explains the collapse under start-controlled evaluation (Table 1): the gripper never truly approaches the cube to pick it up, and the solver succeeds only when (a) the cube already sits at the target, (b) the cube is already in hand and only a short transport remains, or (c) the gripper is initialised in a near-perfect pick pose.

5.2 Controllability under increasing object count

The delta (Δ) row of Table 2 shows the performance gradient of each representation. LeWM's slope from two to four objects is negative in every task family (-18.4 on pick-and-place, -19.8 on stacking, -7.4 on cyclic rearrangement), while ours is positive in all three ($+10.1$, $+7.5$, $+8.5$). Sub-JEPA and the frozen DINOv2-S control are near the floor by four objects (1.9, 0.6, 0.5 and 3.1, 1.2, 1.9 respectively). Adding objects is therefore not uniformly harder: it separates representations that carry object structure from ones that do not.

Every model class drops sharply in performance from one to two cubes. At two objects LeWM leads ours in every family, but by three the ordering has already reversed (35.1 vs 21.9 on pnp3, 15.2 vs 13.4 on stack3 and 25.2 vs 21.6 on cyclic3) and it widens at four. What object count changes is not which representation is better in general but which one degrades with scene information and variety.

Under random start the slope result holds: from two to four objects LeWM falls in every family (-6.0, -29.6, -6.8) and ours rises ($+2.8$, $+19.8$, $+2.3$). The ordering differs by family: LeWM leads ours on pick-and-place (e.g. 72.6 vs 62.6 at four objects) and on two-object stacking, while ours leads on stacking and cyclic rearrangement at three and four objects.

The patch decoder acts differently on the two architectures. Applied to LeWM it buys slope at the cost of performance. On our architecture it raises performance as well ($+10.4$ on pick-and-place, $+9.9$ on stacking and $+1.4$ on cyclic rearrangement over the same model without it), while leaving the gradient mostly positive.

5.3 Robustness to environmental distractors

Introducing task-irrelevant objects while maintaining the evaluation scenes, pre-trained representations and GC-IDM controller enables strict measurement of the representation robustness to environmental distractors (Table 3). On the colour axis both of our arms hold more than LeWM on both tasks: at three colour distractors we achieve 43.1 and 47.8 against LeWM's 1.5 on `pnp1`, and 10.0 and 9.5 against 6.5 on `pnp4`. On `pnp4` ours leads every cell in absolute success, but relative to each

Table 2: **Control success (%) by object count.** Start-controlled (t=0) and random-start evaluation, goal offset 50. Within each family the Δ row is the change from two to four objects; the 1-object row is a saturation anchor and is excluded from it. The best cell per task within each protocol block is shown in bold.

	start-controlled t=0						**random start**					
Task	LeWM	S-JEPA	DINO	L+rec	Ours	O–rec	LeWM	S-JEPA	DINO	L+rec	Ours	O–rec
Pick-and-place												
pnp1	80.1 ±2.6	89.8 ±3.5	30.2 ±3.2	91.2 ±2.0	**97.6** ±1.3	96.9 ±1.4	98.4 ±1.7	97.8 ±1.0	82.2 ±2.2	**99.2** ±0.6	99.1 ±0.8	98.6 ±0.5
pnp2	**35.5** ±6.3	16.4 ±3.4	7.1 ±3.4	28.9 ±2.9	27.4 ±5.3	16.9 ±2.0	**78.6** ±1.1	72.1 ±2.2	72.8 ±2.0	75.5 ±1.4	59.8 ±1.8	57.5 ±2.0
pnp3	21.9 ±3.9	5.2 ±1.4	3.6 ±2.4	21.6 ±1.8	**35.1** ±3.2	23.0 ±2.4	**72.8** ±3.5	58.2 ±2.3	69.5 ±3.6	72.6 ±2.9	63.9 ±2.4	61.0 ±4.4
pnp4	17.1 ±2.0	1.9 ±1.4	3.1 ±1.6	20.0 ±2.1	**37.5** ±3.3	27.1 ±3.4	**72.6** ±2.0	50.5 ±4.9	65.1 ±4.5	71.4 ±1.2	62.6 ±4.5	65.0 ±2.3
Δ 2→4	-18.4	-14.5	-4.0	-8.9	+10.1	+10.2	-6.0	-21.6	-7.6	-4.1	+2.8	+7.5
Stacking												
stack2	**23.0** ±1.7	3.8 ±1.0	2.5 ±0.9	5.5 ±1.5	9.9 ±2.7	7.0 ±1.1	**70.2** ±1.3	43.0 ±5.9	54.6 ±3.4	69.0 ±2.1	46.8 ±1.9	59.1 ±3.5
stack3	13.4 ±2.8	3.0 ±1.2	2.0 ±1.3	7.5 ±0.9	**15.2** ±2.3	9.8 ±1.2	60.6 ±2.7	44.1 ±2.8	52.4 ±2.8	**72.8** ±2.5	69.2 ±4.5	65.9 ±1.9
stack4	3.2 ±1.0	0.6 ±0.8	1.2 ±0.9	15.4 ±2.5	**17.4** ±4.5	7.5 ±3.1	40.6 ±2.4	28.6 ±1.5	53.1 ±1.4	69.4 ±2.9	66.6 ±2.5	**70.1** ±4.0
Δ 2→4	-19.8	-3.2	-1.2	+9.9	+7.5	+0.5	-29.6	-14.4	-1.5	+0.4	+19.8	+11.0
Cyclic rearrangement												
cyclic2	**25.4** ±0.6	3.9 ±1.4	5.4 ±1.8	18.4 ±1.7	15.9 ±2.3	14.2 ±0.9	**69.2** ±5.1	43.8 ±2.8	43.0 ±2.8	67.2 ±5.7	65.9 ±4.8	66.0 ±3.8
cyclic3	21.6 ±2.8	2.9 ±2.5	3.6 ±0.8	16.9 ±1.7	**25.2** ±1.5	23.0 ±1.8	58.5 ±2.6	34.2 ±2.1	44.6 ±4.0	63.6 ±6.9	**74.8** ±5.6	73.0 ±3.2
cyclic4	18.0 ±0.7	0.5 ±0.0	1.9 ±0.8	19.9 ±1.5	**24.4** ±2.9	23.0 ±2.4	62.4 ±3.4	26.5 ±3.9	46.4 ±3.7	62.0 ±3.2	**68.2** ±1.3	60.0 ±4.2
Δ 2→4	-7.4	-3.4	-3.5	+1.5	+8.5	+8.8	-6.8	-17.3	+3.4	-5.2	+2.3	-6.0

Table 3: **Distractor robustness.** Control success (%) with the task held fixed and only task-irrelevant objects added, start-controlled (t=0), offset 50; the colour and size axes are defined in §4. Scored on the task cubes only.

		colour axis			**size axis**		
Task	Model	1 col.	2 col.	3 col.	0.6×	1.5×	2×
pnp1	LeWM	50.6 ±3.1	45.6 ±4.2	1.5 ±1.5	1.0 ±0.7	0.2 ±0.3	0.0 ±0.0
	Ours	**92.9** ±1.3	**89.2** ±1.9	43.1 ±2.0	35.0 ±3.4	**0.4** ±0.2	0.0 ±0.0
	Ours–recon	84.4 ±1.9	81.1 ±2.0	**47.8** ±2.8	**51.1** ±1.4	0.0 ±0.0	0.0 ±0.0
pnp4	LeWM	10.1 ±3.0	8.6 ±3.0	6.5 ±1.6	16.0 ±3.5	10.4 ±2.2	7.5 ±1.4
	Ours	**27.0** ±1.6	**18.4** ±2.5	**10.0** ±2.6	**28.4** ±4.0	**18.5** ±0.6	**11.2** ±1.9
	Ours–recon	15.2 ±2.3	13.1 ±1.8	9.5 ±1.9	20.4 ±1.2	11.9 ±3.0	6.4 ±1.4

model's own undistracted score LeWM degrades least on both axes (at three colour distractors it keeps 38% of its score, ours 27%). On `pnp1` that condition is destructive for every model: a single same-coloured distractor at 1.5× or larger drives all three to zero.

5.4 Decodability of the frozen representation

Figure 3 probes object position directly from the frozen latent isolating representation quality, however these results do not directly quantify controllability. Across the four baseline encoders and the three task families, held-out R^2 falls from two to four objects in twelve cases out of twelve, with slopes from -0.11 to -0.61 (Figure 3, top). Our arms do not: their slopes run from -0.16 to $+0.11$. At four objects our method performs best, achieving > 0.8 on every family and dominating the baselines, LeWM (0.44-0.49) and Sub-JEPA (0.16-0.44).

The MLP regression probe (Figure 3, bottom) furthers our improvement. Going from two to four objects multiplies LeWM's held-out MSE by 54 on pick-and-place and 87 on cyclic rearrangement, while ours changes by at most 2× in the same cells. At four objects ours reads 0.010, 0.078 and 0.006 against LeWM's 0.218, 0.395 and 0.261.

Figure 4 shows the same thing in pixels rather than in scores. The same decoder trained on the frozen representations from each encoder recovers cube positions from both encoders. Our method's representations yield sharp reconstructions of fast-moving objects, primarily the robot itself.

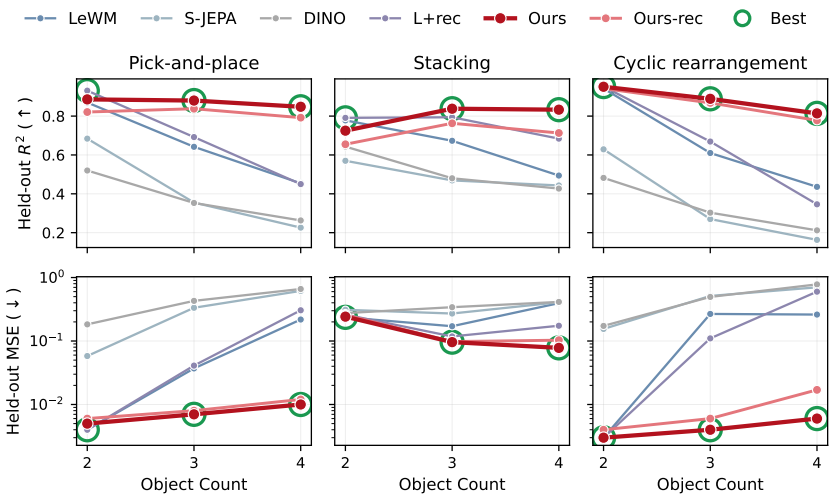


Figure 3: **Position decodability against object count.** Linear probe R^2 (top) and 512–512 MLP held-out MSE (bottom, log scale, standardised units), fit from the frozen latent to the cube coordinates on an 80/20 episode-level split. A ring marks the best model at each object count. Full numbers in Tables 5 and 6.

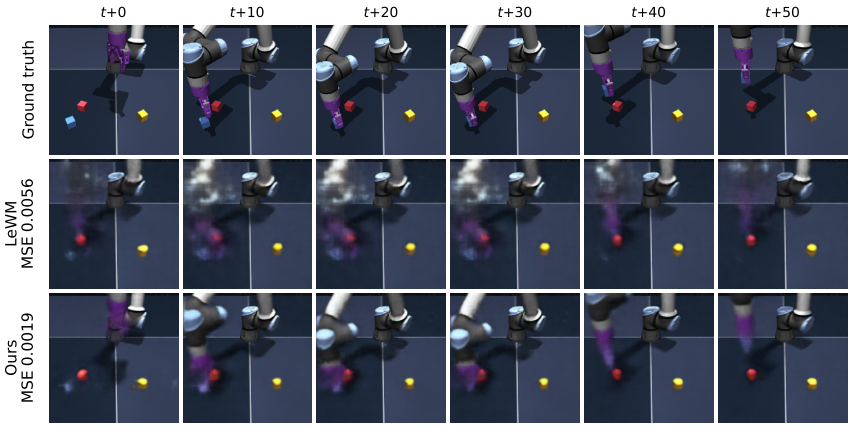


Figure 4: **Observation reconstructions from frozen latents.** One `stack3` episode at stride 10. Rows share decoder architecture, training budget and frames; only the frozen encoder supplying the representation differs.

6 Ablations

A 25-step horizon already matches the full 50-step model on both tasks, while shortening the prediction to a single 5-step step costs everything the longer horizon buys: on pnp4 it returns Ours to LeWM's level (17.4 vs. 17.1) and on cyclic4 it falls well below it (3.9 vs. 18.0), with cube-position decodability collapsing in step (Table 7). The $n{=}1$ arm also has no $\mathcal{L}_a$, but the full-horizon arm without it keeps R^2 at 0.82, indicating that this collapse is due to the horizon.

The inverse-dynamics loss term is impactful for control but not for decodability: the linear probes still reach R^2 0.82 on both tasks, against 0.85 and 0.82 for the full model. Control, however, falls from 37.5 to 25.4 and from 24.4 to 14.5 (Figure 5). A latent can therefore hold the object positions a probe asks for and still be materially worse to act on.

The patch decoder affects both controllability and decodability. Removing it costs four-object control (37.5 → 27.1 on pick-and-place, 17.4 → 7.5 on stacking, 24.4 → 23.0 on cyclic rearrangement) and decodability with it (R^2 0.85 → 0.79, 0.83 → 0.71, 0.81 → 0.78; Tables 2 and 5). This is what we expect of a term that puts object-level detail into the compressed state.

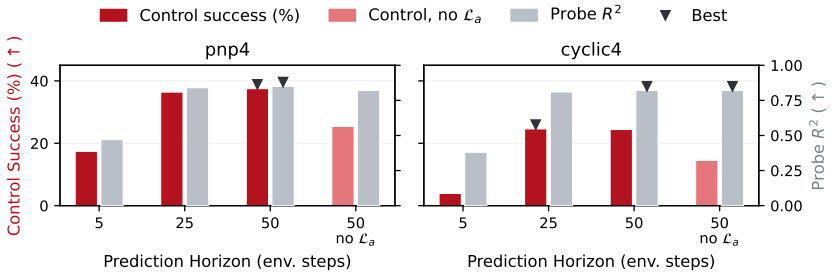


Figure 5: **Ablating prediction horizon and the inverse-dynamics loss.** Control success (left axis) and linear-probe R^2 (right axis) for Ours retrained at shorter horizons, with the full-horizon arm trained without $\mathcal{L}_a$ as a fourth group. A caret marks the best value in each series. Full numbers in Table 7.

7 Discussion and Limitations

Every task we study is built around picking: pick-and-place, stacking and cyclic rearrangement all require the gripper to approach, grasp and transport a cube. Unexpectedly, performance swings widely across tasks that look similar for all model architectures: at four objects ours scores 37.5 on pick-and-place, 24.4 on cyclic rearrangement and 17.4 on stacking. This raises the question of whether our findings translate to more complex scenes with naturally occurring backgrounds; the appropriate next step is to test the method on real-robot data such as Open X-Embodiment (O'Neill et al., 2024) rather than more simulated cubes.

No representation leads on every task and every metric. LeWM is ahead at two objects, ours at three and four; under random starts the ordering changes, and the probe tables rank the arms differently from the control tables. Claims of architectural superiority are protocol-dependent.

Our architecture trains on an eleven-frame window with a predictor and a patch decoder, so training is more expensive than a one-step baseline. Inference is unaffected: the predictor and decoder are discarded, the deployed encoder is the same ViT-Tiny/14 producing the same 192-d vector. We train a predictor but never use it for planning as a world model, because test-time search on these representations scores exactly 0.0% on 3D manipulation from the true episode start (§5.1). Its value here is as a training signal for the encoder. Whether a better planning objective would make such a predictor usable for search is left open.

8 Conclusion

We investigate whether a compact latent trained end-to-end via joint embedding and prediction is actionable, and under what conditions it stops being so. Two evaluation choices decide the answer. Scoring from the true episode start rather than a random mid-episode state takes every test-time search planner on 3D manipulation to exactly 0.0% while the amortized policy still solves the task, a result prior evaluation protocols hide and one that matches what the rollouts show; varying object count with the task type held fixed exposes a limit of prior training objectives rather than of the compact latent itself: control and held-out position decodability fall from two to four objects for LeWM, Sub-JEPA and a frozen DINOv2-S in every task we test, while control with our latent, the same 192-d size as LeWM and Sub-JEPA, improves and its R^2 exceeds 0.8 at four objects.

Our architecture reverses this slope, so that control improves with object count, and outperforms the prior encoders at four objects on every task family. Predicting 50 environment steps in one pass, reading inverse dynamics from the network's internal state representations out of that same pass through a structured attention mask, and training a patch decoder raises four-object control over LeWM on all three families and turns the two-to-four-object slope positive, with decodability following. The ablations identify that increasing the prediction horizon is critical to a point where the performance gains saturate, the inverse-dynamics term affects control performance by up to two-fifths while leaving decodability unmoved, and the patch decoder moves both together, adding up to 10.4 points of four-object control over the same model without it.

AI USE STATEMENT

We used a large language model in three ways. For writing, it revised and polished text, which the authors rewrote and checked. For retrieval and discovery, it aided in finding related work, adding to the authors' initial background research; every paper we cite was then read by the authors, and its identifier and claims verified against the source. For research execution, it wrote scripting to launch experiments, data retrieval, analysis and figure editing under our direction; all generated code was reviewed, and every number reported here is produced from committed result files by a script rather than transcribed. We did not use it to originate the research questions, the evaluation protocol or the conclusions. We have reviewed all AI-assisted work and take responsibility for the final content, including text, claims and artifacts.

ETHICS STATEMENT

This work is entirely simulated. It involves no human subjects, no personal data and no physical robot, and the datasets are generated from an open benchmark. We see no direct path from it to harmful application beyond the general dual-use character of robot learning. Our results are reported against publicly released baseline checkpoints, and we have tried to state where our protocol differs from the one those baselines were scored under rather than comparing across protocols silently.

REPRODUCIBILITY STATEMENT

Every architecture and training setting is listed in Appendix A, Table 4, and the evaluation protocol (start distribution, goal offset, budget and episode count) is specified in §4. The search configuration behind the collapse result, and the budget sweep that rules out an under-provisioned planner, are in Appendix C. The protocol comparison (Table 1) evaluates the baselines from the authors' released checkpoints inside the official `stable-worldmodel` harness, so those numbers can be reproduced without retraining; in the multi-object study LeWM and Sub-JEPA are retrained with their original hyper-parameters, and our own encoders are trained from scratch with the settings in Appendix A. We provide a simplified code release with the submission: scripts that generate the datasets, that train our encoder and its GC-IDM policy on four-object pick-and-place, and that load and evaluate every four-object representation in Table 2 under the protocol above, with released weights for our encoder, LeWM and Sub-JEPA on four-object pick-and-place. Training data is not shipped; it is the open OGBench cube suite, regenerated inside `stable-worldmodel` by those scripts.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, et al. Self-supervised learning from images with a joint-embedding predictive architecture. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629, 2023. doi: 10.1109/CVPR52729.2023.01499.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025. URL https://arxiv.org/abs/2506.09985.

Jiaxin Bai and Jiaxuan Xiong. Temporal-distance JEPA: Plan-aware representation learning for latent world model predictive control. *arXiv preprint arXiv:2607.25337*, 2026.

Randall Balestriero and Yann LeCun. Lejepa: Provable and scalable self-supervised learning without the heuristics. *arXiv preprint arXiv:2511.08544*, 2025. URL https://arxiv.org/abs/2511.08544.

Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-invariance-covariance regularization for self-supervised learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=xm6YD62D1Ub.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, et al. V-JEPA: Latent video prediction for visual representation learning, 2024. URL https://openreview.net/forum?id=WFYbBOEOtv.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, et al. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025a. doi: 10.15607/RSS.2025.XXI.010.

Kevin Black, Manuel Y. Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, et al. (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025b. URL http://papers.nips.cc/paper_files/paper/2025/hash/300ccb2187dedd4edcc07f7e76d8e553-Abstract-Conference.html.

David Brandfonbrener, Ofir Nachum, and Joan Bruna. Inverse dynamics pretraining learns good representations for multitask imitation. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/d36dfcdb14473a8526111c221660f2ab-Abstract-Conference.html.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, et al. RT-1: Robotics Transformer for Real-World Control at Scale. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.025.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, et al. LIBERO-plus: A progressive robustness benchmark for visual-language-action models. In *Conference on Computer Vision and Pattern Recognition 2026*, 2026. URL https://openreview.net/forum?id=Mi1AYOZ9uE.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, et al. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020. URL https://arxiv.org/abs/2004.07219.

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

Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, et al. Octo: An Open-Source Generalist Robot Policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.090.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, et al. Bootstrap your own latent: A new approach to self-supervised learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 21271–21284, 2020.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018. URL https://arxiv.org/abs/1803.10122.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1lOTC4tDS.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640, 2025. doi: 10.1038/s41586-025-08744-2.

Nicklas A Hansen, Hao Su, and Xiaolong Wang. Temporal difference learning for model predictive control. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 8387–8406. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/hansen22a.html.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, et al. Openvla: An open-source vision-language-action model. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Andy Zeyi Liu, Haoran Sun, Lucas Baker, Randall Balestriero, and John Sous. Semigroup-JEPA: Latent dynamics consistency for zero-shot physics generalization. *arXiv preprint arXiv:2609.10464*, 2026.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, qiang liu, Yuke Zhu, et al. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=xzEtNSuDJk.

Lucas Maes, Quentin Le Lidec, Dan Haramati, Nassim Massaudi, Damien Scieur, Yann LeCun, et al. stable-worldmodel-v1: Reproducible world modeling research and evaluation [tiny paper]. In *ICLR 2026 the 2nd Workshop on World Models: Understanding, Modelling and Scaling*, 2026a. URL https://openreview.net/forum?id=wjMSWSPNco.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, et al. Leworldmodel: Stable end-to-end joint-embedding predictive architecture from pixels. *arXiv preprint arXiv:2603.19312*, 2026b. URL https://arxiv.org/abs/2603.19312.

Shangchen Miao, Ningya Feng, Jialong Wu, Ye Lin, Xu He, Dong Li, et al. JEPA-VLA: Video predictive embedding is needed for VLA models. *arXiv preprint arXiv:2602.11832*, 2026.

Marco Molinari, Leonardo Nevali, Saharsha Navani, and Omar G. Younis. Emergent world representations in openvla. *arXiv preprint arXiv:2509.24559*, 2025. URL https://arxiv.org/abs/2509.24559.

Heejeong Nam, Quentin Le Lidec, Lucas Maes, Yann LeCun, et al. Causal-jepa: Learning world models through object-level latent masking. In *International Conference on Machine Learning (ICML)*, 2026.

Hoang Nguyen, Xiaohao Xu, and Xiaonan Huang. Latent geometry beyond search: Amortizing planning in world models. *arXiv preprint arXiv:2605.08732*, 2026. URL https://arxiv.org/abs/2605.08732.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, et al. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=a68SUt6zFt. Featured Certification.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, et al. Open x-embodiment: Robotic learning datasets and rt-x models : Open x-embodiment collaboration0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903, 2024. doi: 10.1109/ICRA57147.2024.10611477.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International Conference on Learning Representations (ICLR)*, 2025.

Cristina Pinneri, Shambhuraj Sawant, Sebastian Blaes, Jan Achterhold, Joerg Stueckler, Michal Rolinek, et al. Sample-efficient cross-entropy method for real-time planning. In Jens Kober, Fabio Ramos, and Claire Tomlin (eds.), *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pp. 1049–1065. PMLR, 16–18 Nov 2021. URL https://proceedings.mlr.press/v155/pinneri21a.html.

Julian Quevedo, Ansh Kumar Sharma, Yixiang Sun, Varad Suryavanshi, Percy Liang, and Sherry Yang. Worldgym: World model as an environment for policy evaluation, 2025. URL https://arxiv.org/abs/2506.00613.

Reuven Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodology And Computing In Applied Probability*, 1(2):127–190, September 1999. ISSN 1573-7713. doi: 10.1023/a:1010091220143. URL http://dx.doi.org/10.1023/A:1010091220143.

Vlad Sobal, Jyothir S, Siddhartha Jalagam, Nicolas Carion, et al. Joint embedding predictive architectures focus on slow features. *arXiv preprint arXiv:2211.10831*, 2022. URL https://arxiv.org/abs/2211.10831.

Vlad Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, et al. Learning from reward-free offline data: A case for planning with latent dynamics models. *arXiv preprint arXiv:2502.14819*, 2025. URL https://arxiv.org/abs/2502.14819.

Jingwen Sun, Wenyao Zhang, Zekun Qi, Shaojie Ren, Zezhi Liu, Hanxin Zhu, et al. Vla-jepa: Enhancing vision-language-action model with latent world model. *CoRR*, abs/2602.10098, February 2026. URL https://doi.org/10.48550/arXiv.2602.10098.

Grady Williams, Andrew Aldrich, and Evangelos Theodorou. Model predictive path integral control using covariance variable importance sampling. *arXiv preprint arXiv:1509.01149*, 2015.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stephane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12310–12320. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/zbontar21a.html.

Xinyu Zhang. What do world models learn in RL? probing latent representations in learned environment simulators. In *ICLR 2026 the 2nd Workshop on World Models: Understanding, Modelling and Scaling*, 2026. URL https://openreview.net/forum?id=VE7JJLVKG1.

Zhenghao Zhang, Yuanxiang Wang, Zhenyu Guan, Yujia Yang, Bingkang Shi, Tianyu Zong, et al. Delta-JEPA: Learning action-sensitive world models via latent difference decoding. *arXiv preprint arXiv:2606.31232*, 2026.

Kai Zhao, Dongliang Nie, Yuchen Lin, Zhehan Luo, et al. Sub-jepa: Subspace gaussian regularization for stable end-to-end world models. *arXiv preprint arXiv:2605.09241*, 2026. URL https://arxiv.org/abs/2605.09241.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.016.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=D5RNACOZEI.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

A Implementation and Training Details

Table 4 lists configuration details for our architecture, training and evaluation settings.

Table 4: **Implementation and training hyper-parameters.** Configuration details for our architecture, training and evaluation settings.

Encoder	
Architecture	ViT-Tiny/14, trained from scratch
Input resolution	224×224 px
Embedding width	192
Projector	$192 \to 2048 \to 192$ MLP, BatchNorm
Read-out	CLS token
Predictor	
Blocks / heads	6 AdaLN-zero / 8
Head width / MLP	64 / 2048
Attention mask	Structured: states causal, actions bracketed
Predicted steps n	10 ($\times$ frameskip $5 = 50$ env. steps)
Context anchors	1 state + 1 action, rest zeroed
Patch decoder	
Architecture	linear + 3 transposed convs, width 256
Latents decoded	4 per batch, encoded $\cup$ predicted
Target	encoder patch tokens, stop-gradient
Objective	
λ_{rec}	1
λ_{SIG}	0.09
SIGReg projections	1024 random 1-D, 1 subspace (ambient 192-d)
Representation training	
Optimizer	AdamW, lr 5×10^{-5}, weight decay 10^{-3}
Batch / precision	128, bf16, grad-clip 1
Schedule	10 epochs $\times$ 20,000 steps = 200,000
Training window	11 frames at frameskip 5
GC-IDM policy	
Head	3 blocks of [Linear 512, LayerNorm, GELU, dropout 0.1]
Horizon conditioning	FiLM from a sinusoidal embedding of $h/50$
Input / output	concat(z_t, z_g), 192-d each $\to$ 5-d action
Optimizer	AdamW, lr 10^{-3}, batch 1024
Schedule	100 epochs on frozen latents, frameskip 1
Goal sampling	$k \sim \mathcal{U}\{1, \ldots, 50\}$ steps ahead, supervised with the recorded action
$H_{\max}$	50
Evaluation	
Protocol	start-controlled (t=0, from the true episode start)
Goal offset	50 environment steps
Budget	$= H_{\max} = 50$
Episodes	200 per evaluation run
Reporting	mean $\pm$ standard deviation

B Visualisation of the Evaluation Tasks

Figure 6 shows what a start-controlled evaluation episode asks of the controller: the scene at the true episode start, the expert's progress every ten environment steps, and the goal 50 steps later.

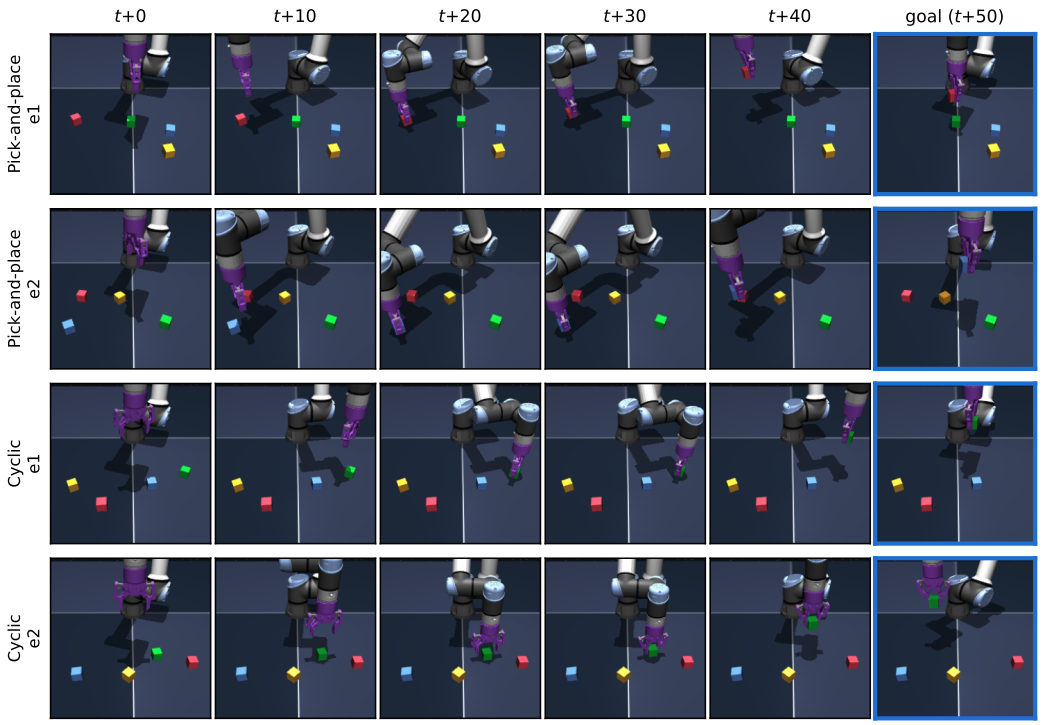


Figure 6: **Start-controlled evaluation episodes at goal offset 50.** Two episodes (e1, e2) drawn at random from the evaluation set of four-object pick-and-place and cyclic rearrangement, shown every ten environment steps from the true episode start. The last column (blue) is the goal image given to the controller, the scene 50 steps later.

C Further Analysis of the CEM Collapse

C.1 CEM hyper-parameter sweep (OGB-Cube)

To confirm that the start-controlled search collapse (§5.1) is not an artefact of an under-provisioned planner, we sweep the CEM search budget over a grid and re-measure. We vary the candidates per iteration (`num_samples` $\in \{300, 500, 1000\}$), the number of refinement iterations (`n_steps` $\in \{10, 30, 50\}$), and the planning/receding horizon ($\in \{5, 10\}$), giving 18 configurations. All other settings match the latent-cost diagnostic (OGB-Cube, Sub-JEPA, start-controlled t=0, offset 50, 12 episodes, action-block 5, top-30 elites). We evaluate 12 episodes per configuration rather than 200 to keep the sweep's compute tractable: at up to 50,000 candidate rollouts per planning step, the 18 configurations cost far more than a single evaluation at the default setting. Because every configuration scores exactly 0%, with no episode moving the cube, the smaller sample does not change the conclusion. This spans the search compute from $300 \times 10 = 3{,}000$ candidate rollouts per planning step to $1000 \times 50 = 50{,}000$, a $16\times$ range that contains the 9,000-rollout default used throughout.

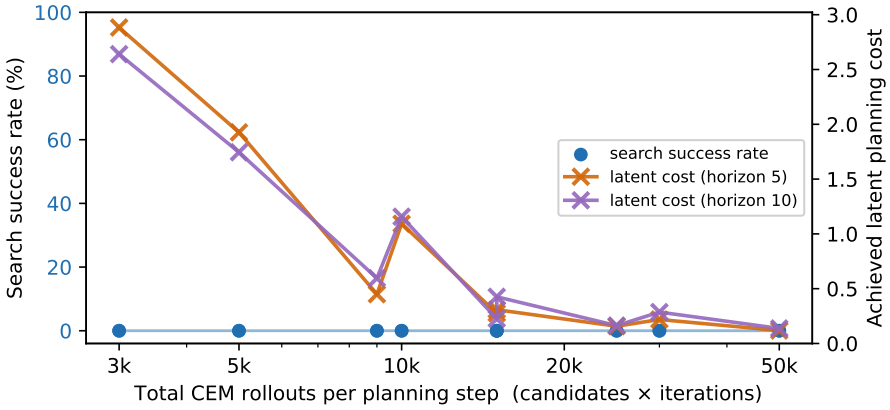


Figure 7: **Search stays collapsed as compute scales** (OGB-Cube, Sub-JEPA, start-controlled $t{=}0$). Across all 18 configurations, search success is exactly 0% and net block displacement is 0.0 cm (max 0.001 cm), while the achieved latent planning cost the optimizer minimizes falls from 2.9 to 0.12 as compute grows. More search buys a better-optimized latent objective, not task progress.

Across every configuration, search success remains exactly 0% and the cube's net displacement stays at 0.0 cm (Fig. 7). Additional compute only lets the optimizer drive the latent goal-cost it minimizes lower (from 2.9 down to 0.12), reinforcing the reading of §5.1. On this task the latent-distance objective and the frozen predictor, not the search budget, are what cap performance.

C.2 Block position under CEM

The sweep shows that more search does not help, and Figure 8 shows why. Tracking one OGB-Cube episode, CEM drives the latent goal-cost it minimises down by 99% while the block itself never moves: net displacement 0.0 cm. The amortized policy's latent cost behaves differently, falling only once the gripper has grasped the block and thereafter tracking true displacement step for step. The planning objective and the true task objective are therefore decoupled on this task, which is what a search procedure cannot repair by searching harder.

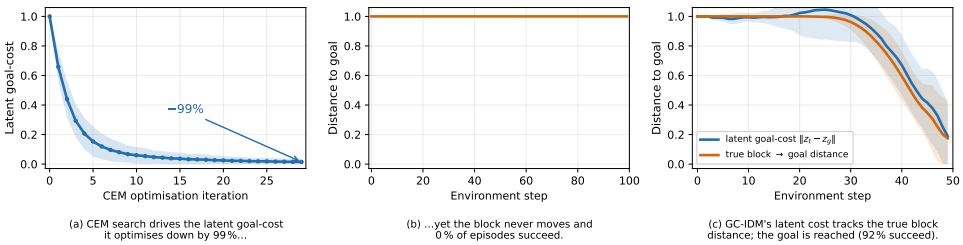


Figure 8: **Why search collapses: the planning objective is the limiter, not the budget.** CEM drives the latent goal-cost it minimises down by 99% while the true object-to-goal distance never falls: displacement 0.0 cm, 0% success. GC-IDM's latent cost instead falls only once the gripper has grasped the block, thereafter tracking true displacement step for step. Scaling search to $50{,}000$ candidate rollouts per step, more than $5\times$ the source default, leaves success at exactly 0%.

D Further Detail of Experimental Results

Two results are reported as figures in the body because the claim they support is a trend rather than a level. The tables below give the underlying numbers. Tables 5 and 6 are the decodability probes plotted in Figure 3, and Table 7 is the horizon and inverse-dynamics ablation plotted in Figure 5. Values are identical to the figures; only the presentation differs.

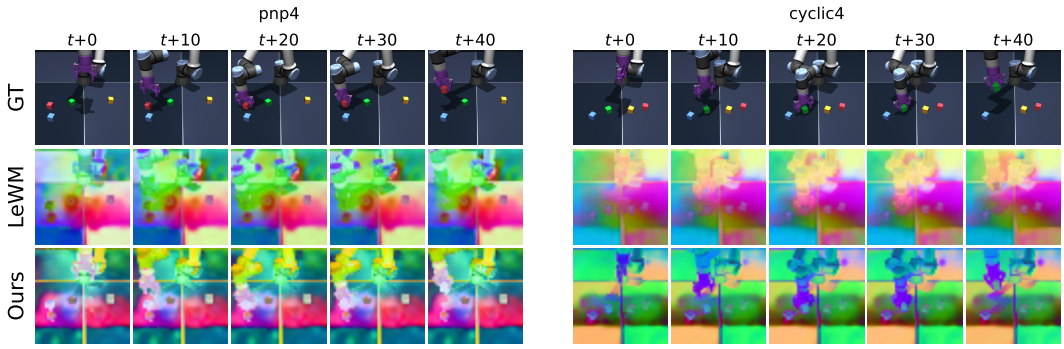


Figure 9: **Patch-feature PCA at four objects.** Last-layer ViT patch tokens, top-3 components mapped to RGB, for one episode of each of two 4-object tasks at stride 10 (stacking omitted for space). Within each (task, model) the PCA basis and the colour span are shared across the five frames, so a colour means the same thing along a row and the row reads as time. Encoded at 672 px through interpolated position embeddings, giving a 48×48 token grid from encoders trained at 224/patch-14. Cubes are present in both models' maps, but under Ours they stay as compact, saturated regions distinct from the table across the whole sequence, whereas LeWM's maps are dominated by a smooth global gradient in which the cubes appear as faint blobs. This is a qualitative illustration: PCA component identity, sign and colour are arbitrary and not comparable across panels, so the quantitative claim rests on Figure 3, not on this figure.

Table 5: **Linear probe** R^2 (↑). Ridge from the frozen latent to the cube coordinates, 80/20 episode-level split. Best per task in bold.

Task	LeWM	S-JEPA	DINO	L+rec	Ours	O–rec
pnp1	**0.991**	0.952	0.897	0.989	0.987	0.980
pnp2	0.870	0.684	0.520	**0.931**	0.886	0.821
pnp3	0.642	0.355	0.353	0.692	**0.880**	0.838
pnp4	0.454	0.226	0.263	0.450	**0.848**	0.792
stack2	0.778	0.570	0.643	**0.791**	0.725	0.655
stack3	0.673	0.469	0.480	0.794	**0.838**	0.763
stack4	0.494	0.443	0.427	0.684	**0.833**	0.713
cyclic2	0.942	0.629	0.482	**0.952**	0.951	0.942
cyclic3	0.610	0.270	0.303	0.669	**0.889**	0.868
cyclic4	0.436	0.163	0.212	0.346	**0.814**	0.778

Table 6: **MLP held-out MSE** (↓). A 512–512 MLP trained and evaluated on the same 80/20 split. Targets are standardised per coordinate, so MSE is in units of each coordinate's variance. Best per task in bold.

Task	LeWM	S-JEPA	DINO	L+rec	Ours	O–rec
pnp1	0.001	0.003	0.031	0.001	**0.001**	0.001
pnp2	**0.004**	0.058	0.183	0.004	0.005	0.006
pnp3	0.037	0.334	0.429	0.041	**0.007**	0.008
pnp4	0.218	0.616	0.660	0.305	**0.010**	0.012
stack2	**0.236**	0.310	0.275	0.261	0.242	0.258
stack3	0.171	0.271	0.341	0.118	**0.096**	0.098
stack4	0.395	0.408	0.413	0.174	**0.078**	0.103
cyclic2	0.003	0.155	0.173	**0.003**	0.003	0.004
cyclic3	0.267	0.513	0.495	0.110	**0.004**	0.006
cyclic4	0.261	0.701	0.780	0.599	**0.006**	0.017

Table 7: **Ablation results of our method.** Control success rate (%) under the start-controlled protocol (t=0), and the out-of-sample linear-probe R^2 following the 80/20 split. We ablate the prediction horizon and the inverse-dynamics loss.

	n=1 (5 steps)	n=5 (25 steps)	n=10 (50 steps) **Ours**	no $\mathcal{L}_a$
pnp4 success	17.4 $\pm$2.5	36.4 $\pm$3.3	**37.5** $\pm$3.3	25.4 $\pm$5.3
probe R^2	0.47	0.84	0.85	0.82
cyclic4 success	3.9 $\pm$1.1	**24.6** $\pm$3.6	24.4 $\pm$2.9	14.5 $\pm$2.0
probe R^2	0.38	0.81	0.82	0.82

E Visualisation of Patch Representations

Figure 9 looks below the global vector at the patch tokens the encoder produces, for one four-object episode of each of two tasks. It is a qualitative view only: PCA component identity, sign and colour are arbitrary and not comparable across panels, so the quantitative claim rests on Figures 3 and 4 rather than on this one.