PLANJEPA: LEARNING CONTROL-STRUCTURED REPRESENTATIONS FOR LATENT PLANNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning representations that support planning is essential for latent world models. However, existing representation-learning objectives focus on latent-state prediction or action recoverability and do not constrain the complexity of the control relation between actions and latent transitions, so an accurate predictor can still operate over a latent space with highly nonlinear control relations. Consequently, accurate latent prediction does not necessarily lead to successful planning, since goal costs, candidate-trajectory ranking, and gradient-based action optimization all depend on the control structure exposed by the latent space. We introduce PlanJEPA, which learns control-structured representations with a multi-horizon linear action objective. By requiring actions to be predicted through a shared linear map of current and future latent states across multiple temporal horizons, PlanJEPA encourages the encoder to organize control-relevant variations directly in the representation. We prove that the imposed linear action relation forces nontrivial control errors to induce nontrivial latent goal errors and improves the local conditioning of goal-directed planning objectives. Empirically, PlanJEPA better organizes control-relevant state changes in latent space and consistently outperforms strong latent world model baselines across closed-form control, learned policies, cross-entropy method search, and gradient-based planning.

1 INTRODUCTION

Latent world models provide an efficient framework for visual control and planning. Their core idea is to map high-dimensional observations to compact latent representations, learn action-conditioned dynamics in that latent space, and optimize actions or policies by imagining future trajectories (Ha & Schmidhuber, 2018; Hafner et al., 2019; 2020; 2021; Schrittwieser et al., 2020; Hansen et al., 2022; Hafner et al., 2025). Compared with predicting and planning directly in observation space, the latent abstraction reduces the dimensionality of the representation and relieves the dynamics model of irrelevant visual detail, which makes both dynamics learning and planning more efficient (Hansen et al., 2024; Sobal et al., 2025; Assran et al., 2025; Zhou et al., 2025; Maes et al., 2026).

In practice, however, planning in the learned latent space can remain difficult even when the world model predicts future latent states accurately. The prediction objective by itself does not guarantee that the representation retains the factors relevant to control, and prior work therefore adds auxiliary supervision such as an inverse-dynamics loss (IDM): EB-JEPA adds such a loss to an action-conditioned world model, recovering the executed action from adjacent latent states to strengthen action-relevant information in the representation (Terver et al., 2026), and Sensorimotor World Models make it the central regularizer, requiring a_t to be recoverable from (z_t, z_{t+1}) so that the encoder focuses on the controllable state changes (Ivashkov et al., 2026). Yet being able to recover the action is not the same as the control relation being simple. A high-capacity inverse-dynamics network can decode actions from a highly curved or intricately organized latent space.

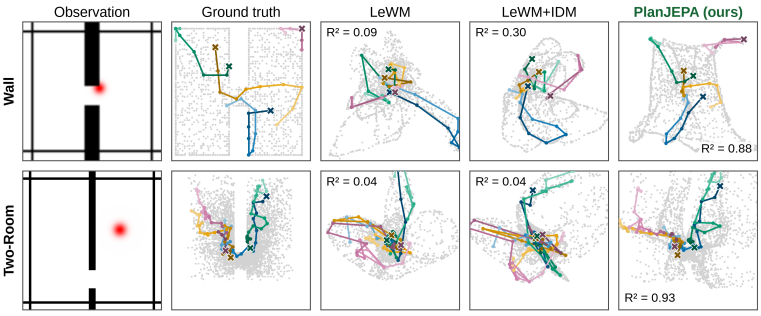
For planning, this distinction matters: latent distance usually serves directly as the goal cost, sampling-based planners rank candidate trajectories by that cost, and gradient-based planners further depend on the local geometry of the objective with respect to actions. When the control relation is highly nonlinear, the resulting planning objective can be hard to compare, search, or optimize. Figure 1(a) shows a direct manifestation of this problem: on the same frozen LeWorldModel (LeWM) (Maes et al., 2026) representation, a learned goal-conditioned MLP controller clearly outperforms

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.

Same encoder, two planners

		CEM	MLP
Wall	LeWM	73.0	**98.5**
	LeWM+IDM	71.5	**98.0**
Cube-PG	LeWM	11.5	**94.0**
	LeWM+IDM	15.5	**99.5**
Two-Room	LeWM	91.0	**98.0**
	LeWM+IDM	91.0	**96.5**
Reacher	LeWM	82.0	**88.5**
	LeWM+IDM	80.0	**90.0**

(a)



(b)

Figure 1: **Motivation and latent structure.** **(a)** Success rates of CEM search and of a learned goal-conditioned MLP controller on the same frozen representation, for LeWM with and without an inverse-dynamics head (LeWM+IDM). **(b)** Trajectories in the ground-truth state space and in each representation, projected onto its first two principal components and Procrustes-aligned to the ground-truth coordinates; R^2 is the fraction of ground-truth variance explained by the alignment.

cross-entropy method (CEM) search through the latent dynamics, and adding an inverse-dynamics head to LeWM (LeWM+IDM) raises the MLP controller further while leaving CEM essentially unchanged. This suggests that the information required for control is already present in the representation, since a nonlinear controller recovers it, but it is not organized in a form that direct latent-space planners can exploit efficiently.

What, then, makes a representation suitable for latent planning? We argue that planning benefits from representations with two properties: (i) the representation should retain enough control-relevant information for different actions and the state changes they cause to be predictable; and (ii) it should limit the complexity of the control relation itself. If the action can be recovered from the current and a future state through a simple and stable map, the effect of a change in action is exposed more directly in the representation, which gives goal comparison, search, and local optimization a more regular control geometry. We call a representation with both properties control-structured.

Motivated by these two requirements, we introduce PlanJEPA, a representation-learning method for planning with latent world models. Concretely, PlanJEPA adds to the standard nonlinear inverse-dynamics objective a strictly linear multi-horizon action objective: given the current latent state z_t and a future latent state z_{t+k}, a linear head shared across horizons predicts the first action a_t. Because the linear head cannot absorb a complex action-recovery map on its own, minimizing this objective pushes the encoder to organize control-relevant variation more directly in the latent representation. The latent space learned by PlanJEPA better preserves the trajectory structure produced by the control process, as shown in Figure 1(b).

The main contributions of this work are summarized as follows:

- We are the first to identify the complexity of the control relation as a planning-oriented representation gap: even when the latent contains sufficient action-relevant information, that relation may be organized in a form planners cannot exploit directly.
- We propose PlanJEPA, which explicitly shapes the control structure of the latent space through a strictly linear multi-horizon action objective and thereby closes the gap between representation and planning identified above.
- We analyze theoretically how this linear control relation affects planning, proving that it forces control error to be reflected in latent goal error and improves the local conditioning of goal-directed planning objectives for gradient-based planners.
- Extensive experiments show that PlanJEPA outperforms state-of-the-art latent world model baselines across closed-form control, learned policies, CEM search, and gradient-based planning, across five visual control environments and the LIBERO-Goal benchmark. On most tasks the learned representation lets a simple linear controller reach competitive success rates, which sharply reduces inference cost.

2 Related Work

Latent world models. Latent world models predict future embeddings instead of pixels: DINO-WM predicts pretrained patch features for zero-shot planning, V-JEPA 2 couples large-scale video pretraining with an action-conditioned model for robot planning, and LeWorldModel trains a compact action-conditioned JEPA end to end from pixels (Zhou et al., 2025; Assran et al., 2025; Maes et al., 2026). Their objective descends from Joint-Embedding Predictive Architectures, which predict the embedding of a target from the embedding of a context so that predictable structure is retained without reconstruction (LeCun, 2022; Assran et al., 2023; Bardes et al., 2024; Balestriero & LeCun, 2025). This line makes latent futures predictable, and RC-aux adds reachability supervision so that latent distance tracks what a planner can reach (Li et al., 2026); PlanJEPA asks the complementary question of whether the learned coordinates expose the control structure a planner requires, which predictive accuracy alone does not guarantee (Figure 1(a)).

Inverse-dynamics models. Inverse models map state changes to the actions that produced them (Jordan & Rumelhart, 1992) and have long served as self-supervision for visual and control-relevant features (Jayaraman & Grauman, 2015; Agrawal et al., 2016; Pathak et al., 2017; Islam et al., 2023; Koul et al., 2024). Sensorimotor World Models bring this objective into end-to-end latent world models, where a nonlinear inverse-dynamics head prevents collapse (Ivashkov et al., 2026); EB-JEPA and Delta-JEPA use related auxiliary action-prediction objectives (Terver et al., 2026; Zhang et al., 2026), while GC-IDM amortizes planning through goal-conditioned inverse dynamics and INTACT jointly learns its representation and intent-to-action mapping (Nguyen et al., 2026; Sun et al., 2026). These expressive inverse models establish action recoverability without requiring a simple control relation. PlanJEPA instead uses a single linear head shared across horizons, so this relation must be learned into the representation itself.

Latent linear control. Representation learning for control has long sought latent coordinates in which the dynamics become linear. E2C, its successors, and Deep Variational Bayes Filters learn locally linear latent transitions from pixels (Watter et al., 2015; Banijamali et al., 2018; Levine et al., 2020; Shu et al., 2020; Karl et al., 2017), and Koopman-based methods lift nonlinear dynamics into observables that evolve linearly under a global operator (Takeishi et al., 2017; Lusch et al., 2018; Korda & Mezić, 2018). All of these place the linear structure on the forward dynamics, which a latent world model cannot afford: its predictor has to reproduce contact, occlusion, and other transitions that are linear in no fixed coordinates. What a planner needs is a simple relation between the latent pair it compares and the action it selects, so PlanJEPA keeps the predictor nonlinear and places the linear structure on the inverse relation. Appendix D discusses the broader literature on control and plannable representations, including the linear-latent lines above.

3 Method

PlanJEPA keeps the LeWM backbone and its loss unchanged and adds two training-only action heads: a nonlinear inverse model and a strictly linear multi-horizon plan head (Figure 2). Section 3.1 recalls the backbone and the inverse-dynamics supervision that keeps action information recoverable, Section 3.2 introduces the plan objective organizing it, and Section 3.3 joins them.

3.1 Latent World Model and Inverse-Dynamics Supervision

Let o_t denote an observation, $a_t \in \mathbb{R}^{d_a}$ an action, and $z_t = E_\phi(o_t) \in \mathbb{R}^d$ its latent representation. We build on LeWM (Maes et al., 2026), a latent world model trained end to end from pixels on offline, reward-free trajectories. Its encoder E_ϕ is a vision transformer whose class-token embedding is projected to the latent z_t, and its forward model F_θ is a causal transformer that reads a history of latents and actions, injects the actions through adaptive layer normalization, and predicts the next latent, $\hat{z}_{t+1} = F_\theta(z_{\leq t}, a_{\leq t})$; training uses teacher forcing, so the prediction is scored against the encoded next observation. This objective alone admits the collapsed solution that maps every observation to the same latent, so LeWM adds SIGReg (Balestriero & LeCun, 2025), which projects the batch of latents Z onto random unit directions and penalizes the departure of each one-dimensional marginal from a standard Gaussian, driving the latent distribution toward an isotropic Gaussian. The

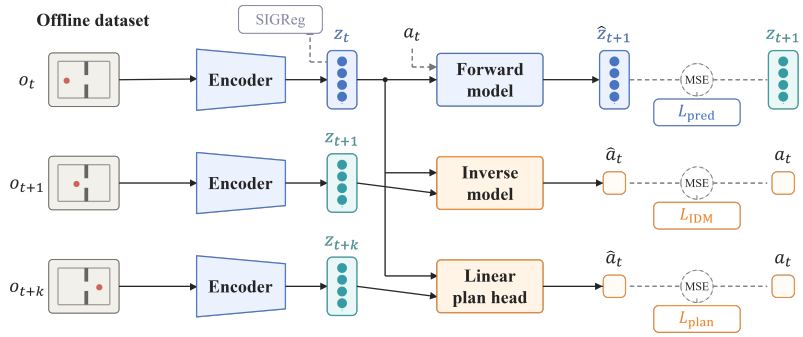


Figure 2: **PlanJEPA training.** A shared encoder maps o_t, o_{t+1}, and o_{t+k} to latents. The action-conditioned forward model predicts $\hat{z}_{t+1}$ from (z_t, a_t) and is trained with $\mathcal{L}_{\text{pred}}$, while SIGReg regularizes the latents (together, Eq. (1)). Two training-only heads, discarded at deployment, read latent pairs: a nonlinear inverse model recovers a_t from (z_t, z_{t+1}) ($\mathcal{L}_{\text{IDM}}$), and a strictly linear plan head predicts the same first action from (z_t, z_{t+k}) with one map shared across all horizons k ($\mathcal{L}_{\text{plan}}$).

two terms form the world-model loss

$$\mathcal{L}_{\text{LeWM}} = \mathcal{L}_{\text{pred}} + \lambda_{\text{sig}}\,\mathcal{L}_{\text{sig}}, \qquad \mathcal{L}_{\text{pred}} = \mathbb{E}\left[\|\hat{z}_{t+1} - z_{t+1}\|_2^2\right], \quad \mathcal{L}_{\text{sig}} = \text{SIGReg}(Z), \tag{1}$$

optimized jointly over the encoder and the forward model without stop-gradients, moving-average targets, or pretrained weights. We keep this backbone and its loss unchanged.

To further retain action-relevant information, we adopt a nonlinear inverse-dynamics model g_ψ that recovers the executed action from the current and the next latent state,

$$\mathcal{L}_{\text{IDM}} = \mathbb{E}\left[\|a_t - g_\psi(z_t, z_{t+1})\|_2^2\right]. \tag{2}$$

This objective provides flexible local control supervision: a latent transition must carry enough information to recover the action that produced it. Because g_ψ is highly expressive, the same action-recovery accuracy is compatible with latent control relations of very different complexity. Inverse-dynamics supervision therefore guarantees that control information is recoverable, without prescribing how it is structured in latent space.

3.2 Multi-Horizon Linear Action Objective

To explicitly shape a planning-oriented control structure, we introduce a multi-horizon linear action objective, the plan objective. For a horizon k, PlanJEPA predicts the first action a_t of a trajectory linearly from the current latent z_t and the future latent z_{t+k}. Given a set of horizons $\mathcal{K}$, the objective is

$$\mathcal{L}_{\text{plan}} = \frac{1}{|\mathcal{K}|}\sum_{k\in\mathcal{K}} \mathbb{E}\left[\|a_t - U z_t - V z_{t+k}\|_2^2\right], \qquad U, V \in \mathbb{R}^{d_a \times d}, \tag{3}$$

where the linear maps U and V form the plan head and are shared by all horizons in $\mathcal{K}$.

The limited expressivity of a linear head prevents the action decoder from absorbing the complexity of the control relation, which pushes the encoder toward a simpler and more consistent control structure. The multi-horizon constraint further requires this relation to hold at several temporal scales at once, so that the current action is tied to futures at different distances through one and the same structure rather than through a separate relation per horizon.

We predict the first action a_t so that the training objective corresponds directly to the basic question of goal-directed control: given the current state and a future goal, which action to take now. The multi-horizon objective thus extends local action supervision into a structural constraint that ties current control to goals at every horizon of the training window.

3.3 JOINT TRAINING OBJECTIVE

The two objectives provide complementary control supervision and map onto the two requirements of Section 1: the nonlinear model targets control sufficiency, since it can describe arbitrarily complex control relations, and the linear objective targets control structure, since it makes those relations exploitable by simple planners. Either alone falls short: an expressive decoder adapts to any latent control relation, and a linear constraint alone may fail to capture all local variation of visual control. Joint optimization balances sufficiency against structure in the control information that the encoder retains, so that neither is traded away for the other.

The final training objective adds both heads to the world-model loss of Eq. (1),

$$\mathcal{L} = \mathcal{L}_{\mathrm{LeWM}} + \lambda_{\mathrm{idm}}\mathcal{L}_{\mathrm{IDM}} + \lambda_{\mathrm{plan}}\mathcal{L}_{\mathrm{plan}}, \tag{4}$$

where λ_{idm} and λ_{plan} weight the nonlinear inverse-dynamics supervision and the plan objective.

4 WHY LINEAR CONTROL STRUCTURE HELPS PLANNING

Our multi-horizon linear plan objective is designed to shape the latent representation for downstream control. Rather than constraining a particular test-time planner, the objective encourages action-relevant information to be accessible through a linear map across multiple horizons. In this section, we formalize this property and study its implications for latent planning.

4.1 MULTI-HORIZON LINEAR CONTROL STRUCTURE

A generic nonlinear inverse-dynamics model can ensure that actions are recoverable from latent transitions, but recoverability alone does not specify how action differences are represented geometrically. A linear plan head imposes a stronger quantitative constraint.

Definition 1 (k-step linear control structure)**.** A representation is linearly control-structured at horizon k if there exist matrices U_k and V_k such that, for valid trajectory segments,

$$a_t = U_k z_t + V_k z_{t+k}. \tag{5}$$

PlanJEPA trains one pair shared by all horizons, $U_k{=}U$ and $V_k{=}V$ for every $k \in \mathcal{K}$; the statements below hold for each horizon separately and therefore for the shared head trained by Eq. (3).

Theorem 1 (Multi-horizon action separation)**.** *Suppose Definition 1 holds for horizon k. Consider two valid trajectory segments with the same initial latent z_t, but first actions a_t and a'_t and terminal latents z_{t+k} and z'_{t+k}, respectively. Then*

$$\left\|z_{t+k} - z'_{t+k}\right\|_2 \geq \frac{1}{\|V_k\|_2}\left\|a_t - a'_t\right\|_2. \tag{6}$$

Consequently, if Eq. (5) *holds for every $k \in \mathcal{K}$, controllably different trajectory segments cannot collapse to arbitrarily similar terminal representations at any of the regularized scales.*

The proof is in Appendix C.3 (residual-robust version: Appendix C.4; nonlinear inverse models: Appendix C.5). The coefficient $1/\|V_k\|_2$ is meaningful only once the latent scale is fixed, since rescaling z by c rescales V_k by $1/c$; SIGReg fixes this scale, and under its idealization $\mathbb{E}[z_t] = 0$, $\mathbb{E}[z_t z_t^\top] = I$ the coefficient is also invariant to rotations of the latent (Appendix C.2). A nonlinear IDM primarily asks whether action information is present in the representation, whereas Eq. (6) constrains how that information is geometrically organized: large action differences must induce proportionally detectable differences in the latent transitions; Appendix B.7 reports the measured coefficients on Push-T and Cube-PG, the two environments closest to linear.

4.2 IMPLICATIONS FOR GRADIENT-BASED PLANNING

For gradient-based planning, we now specialize to the one-step relation $k = 1$, for which the dependence on the current action is unambiguous. Let the twice-differentiable one-step latent dynamics be $z_{t+1} = F(z_t, a_t)$, with action Jacobian $J(a) := \partial F(z_t, a)/\partial a$.

Assumption 1 (Locally exact one-step inverse relation)**.** For a fixed z_t, there exists a neighborhood of actions in which $a = U_1 z_t + V_1 F(z_t, a)$ holds.

Theorem 2 (Non-vanishing action sensitivity)**.** *Under Assumption 1, $V_1 J(a) = I$, and therefore the smallest singular value of $J(a)$ satisfies*

$$\sigma_{\min}(J(a)) \geq \frac{1}{\|V_1\|_2}. \tag{7}$$

Hence, the one-step latent dynamics cannot become arbitrarily insensitive to action along any action-space direction throughout the neighborhood of Assumption 1.

The proof is in Appendix C.6. Consider the standard goal-reaching objective

$$\ell(a) = \tfrac{1}{2}\|F(z_t, a) - z_g\|_2^2. \tag{8}$$

Theorem 3 (Local conditioning of goal-directed optimization)**.** *Let $a^\star$ satisfy $F(z_t, a^\star) = z_g$, suppose Assumption 1 holds around $a^\star$, and let $\|J(a^\star)\|_2 \leq L$. Then the Hessian of ℓ at $a^\star$ and its condition number κ satisfy*

$$\frac{1}{\|V_1\|_2^2} I \preceq \nabla_a^2 \ell(a^\star) \preceq L^2 I \quad \textit{and} \quad \kappa\big(\nabla_a^2 \ell(a^\star)\big) \leq L^2 \|V_1\|_2^2. \tag{9}$$

Theorem 3 shows that linear control structure does more than prevent a degenerate action Jacobian: it yields an explicit local conditioning guarantee for gradient-based goal optimization on a nonlinear world model. The proof is in Appendix C.7.

The preceding results characterize a property of the learned representation rather than a particular controller. For a trajectory whose desired endpoint is $z_g = z_{t+k}$, Eq. (5) directly gives the linear controller $\pi_k^{\text{lin}}(z_t, z_g) = U_k z_t + V_k z_g$, which we fit by ridge regression in the experiments; the implications for learned policies and sampling-based search, and the residual-robust forms of the bounds, are in Appendix C.8. Overall, our analysis suggests the following mechanism: the multi-horizon plan objective yields a linearly structured control geometry, which in turn yields better behaved downstream control objectives.

5 Experiments

The experiments ask whether a single frozen PlanJEPA representation improves planners of different capacity and mechanism. Section 5.2 examines what the plan objective changes in the representation, Section 5.3 evaluates the frozen representation under four planner families and policy learning, and Section 5.4 ablates the objective.

5.1 Experimental Setup

We evaluate on five visual control environments—Two-Room (Sobal et al., 2025), Wall (Zhou et al., 2025), Reacher (Tassa et al., 2018), Push-T (Chi et al., 2025), and OGBench Cube pick-and-place (Park et al., 2025)—plus the LIBERO-Goal policy-learning benchmark (Liu et al., 2023). Cube is evaluated from pre-grasp starts (Cube-PG), so every episode requires the full pick-and-place (Appendix A.2); all world models are trained on offline demonstrations following the LeWM protocol (Maes et al., 2026), with environments, datasets, and hyperparameters in Appendices A.1–A.3.

Each frozen representation is evaluated under four planner families of different capacity and mechanism: a closed-form horizon-conditioned linear controller, a horizon-conditioned MLP readout, CEM search over the learned predictor, run receding-horizon and open-loop, and a gradient-based planner. The linear controller is fitted once per environment by ridge regression from the encoded observation, the encoded goal and the horizon to the action, so a plan is two encoder calls and one matrix multiply. Protocols are in Appendix A.4. The planner and its budget are fixed within each column, so rows differ only in the representation. Baselines are the LeWM backbone (Maes et al., 2026), Delta-JEPA (Zhang et al., 2026), Sub-JEPA (Zhao et al., 2026), Fast-LeWM (Gao & Xu, 2026), INTACT (Sun et al., 2026), DINO-WM (Zhou et al., 2025) in its native planning stack, frozen DINOv2 (Oquab et al., 2024), and two ablations of our objective, *PlanJEPA w/o plan head* (λ_{plan}=0, the nonlinear IDM alone; LeWM+IDM in Figure 1) and *PlanJEPA w/o IDM head* (λ_{idm}=0); the plan weight is 1.0 throughout, and Appendix B.6 sweeps it.

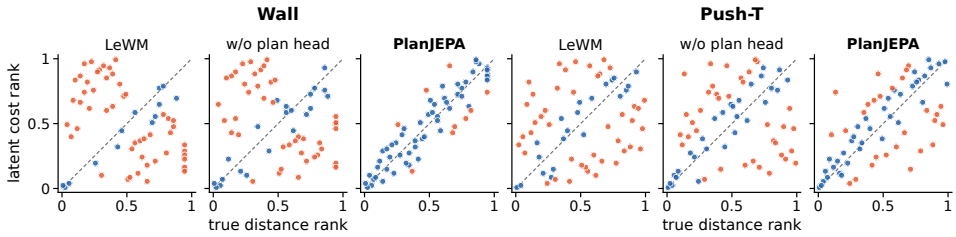


Figure 3: Ranking of reachable futures on one start–goal pair of Wall and of Push-T. From the start state, 64 random action sequences of the planning horizon are executed in the simulator; each point is one candidate, placed at the percentile rank of the true task distance of its reached state (x) and at the percentile rank of the predictor's latent cost of the same candidate (y). The dashed line is a perfect ranking; blue points lie within two deciles of it and orange points farther away.

Table 1: Ranking of reachable futures: Spearman correlation between latent goal cost and true task distance over 64 executed random action sequences per start state, mean (s.e.m.) over 100 start–goal pairs; ρ_{enc} uses the encoded true endpoint and ρ_{roll} the predictor's rollout, which is the quantity that CEM and gradient planning minimize. Higher is better.

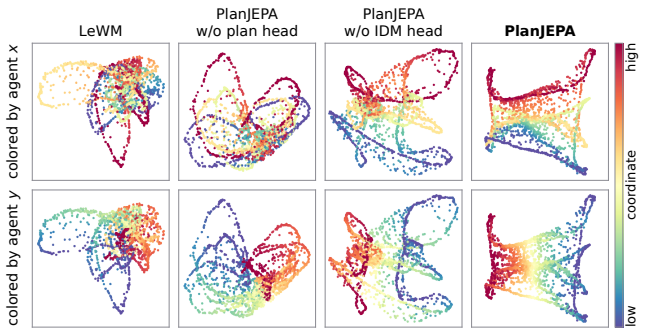
Method	Two-Room	Reacher	Push-T	Wall	Cube-PG
ρ_{enc}: encoded true endpoint					
LeWM	0.50 (0.03)	0.18 (0.02)	0.41 (0.03)	0.35 (0.03)	0.37 (0.02)
w/o plan head	0.54 (0.02)	0.32 (0.02)	0.52 (0.03)	0.58 (0.02)	**0.58** (0.02)
PlanJEPA (ours)	**0.60** (0.02)	**0.34** (0.02)	**0.58** (0.02)	**0.65** (0.02)	0.58 (0.02)
ρ_{roll}: predictor rollout					
LeWM	0.46 (0.02)	0.17 (0.02)	0.52 (0.03)	0.30 (0.03)	0.37 (0.02)
w/o plan head	0.52 (0.02)	0.31 (0.02)	0.64 (0.02)	0.35 (0.03)	**0.56** (0.01)
PlanJEPA (ours)	**0.59** (0.02)	**0.33** (0.02)	**0.74** (0.02)	**0.59** (0.02)	0.55 (0.01)

5.2 What Makes PlanJEPA Representations More Plannable?

We first ask what the plan objective changes in the representation.

Ranking of reachable futures. A sampling planner keeps the proposals whose predicted futures lie closest to the goal in latent distance, and we test this ordering directly. From one start state we execute many random action sequences and compare the latent goal cost of each reached endpoint with its true task distance. As shown in Figure 3 for one start–goal pair of Wall and of Push-T, under LeWM and under the IDM head alone the candidates scatter across the plane, while under PlanJEPA they concentrate along the diagonal, so the candidates a planner would keep are also the ones closest to the goal. More pairs and all encoders are in Appendix B.4. The same measurement over the five environments and 100 pairs is given in Table 1. We find that PlanJEPA raises the ranking correlation of both the encoded endpoint and the predicted rollout in every environment relative to LeWM, and that the full objective strengthens the improvement already present with the IDM head alone. CEM and gradient planning minimize exactly the latent goal cost of predictor rollouts, so we read the higher correlation as a more faithful planning objective.

Latent coordinates and task geometry. We next ask whether latent distance tracks task distance without fitting any readout. As shown in Figure 4, in the Wall projection under LeWM the colors fold. Frames from the two sides of the wall land in the same region of the plane, so positions that the task keeps apart become latent neighbors. The IDM head alone unfolds y and leaves x interleaved, and under PlanJEPA both coordinates vary smoothly across the plane. The smoothness scores on the right show the same ordering for Two-Room. A fold of this kind is invisible to a Euclidean terminal cost, since states far apart in the task but adjacent in latent space are then ranked as equally good goals. More projections are in Appendix B.5.



Method	Wall		Two-Room	
	x	y	x	y
LeWM	0.45	0.62	0.42	0.51
PlanJEPA (ours)	**0.83**	**0.92**	**0.93**	**0.92**
w/o plan head	0.48	0.80	0.44	0.61
w/o IDM head	0.78	0.81	**0.93**	0.91

Figure 4: Latent coordinates and task geometry. Left: the first two principal components of the Wall latents of expert episodes, each frame colored by its true agent x (top) or y (bottom), blue = low, red = high; no readout is fitted. Right: local smoothness of the true coordinate over the PCA plane for Wall and Two-Room (1 = position-consistent neighborhoods).

Table 2: One frozen representation across controller families: average success (%) over the five environments under receding-horizon (Rec.) and open-loop (Open) CEM search, the closed-form linear controller (Lin.), the horizon-conditioned MLP readout (MLP) and the gradient planner (Grad.), and LIBERO-Goal policy learning (mean over four training seeds). Per-environment results with standard deviations: Tables B.1–B.4. Best per column in bold.

Method	CEM		Readout		Gradient	Policy
	Rec.	Open	Lin.	MLP	Grad.	LIBERO
Sub-JEPA	69.7	55.4	51.2	93.3	73.9	77.2
Delta-JEPA	72.1	60.9	73.8	94.9	73.6	80.7
LeWM	70.2	56.0	50.5	91.7	69.0	80.3
PlanJEPA (ours)	**76.9**	**64.6**	**80.7**	**95.3**	**76.1**	**83.2**
w/o plan head	70.9	62.4	70.6	93.6	74.1	79.6
w/o IDM head	71.1	60.9	78.5	92.6	73.9	79.9

5.3 One Frozen Representation Across Controller Families

Results across controller families. As shown in Table 2, PlanJEPA is the only representation that attains the highest average across all controller families, and its margin over the LeWM backbone appears in every family, from 2.9 points on LIBERO-Goal to 30.2 for the linear controller. The per-environment results follow the same pattern: PlanJEPA attains the highest success among the baselines in 16 of the 20 environment $\times$ family combinations (80%; Tables B.1–B.3). We attribute the uniformity of the gain to the plan head constraining the encoder rather than any planner, so that what it changes is available to every controller that reads the latent. Two controls support this reading. Retraining the predictor from scratch on each frozen encoder leaves the CEM levels and margins unchanged (Appendix B.8), and executing plans open-loop, without per-step correction, widens the margin over LeWM rather than closing it.

Closed-form linear controller. On LeWM this controller solves half of the episodes and on PlanJEPA four in five, which on average is above every other planner except the MLP readout. We also observe that the spread between the best and the worst representation in each column of Table 2 shrinks as the controller adds more capacity of its own, from about 30 points for the fixed linear map to 7–9 under CEM and the gradient planner and under 4 for the trained MLP readout. We attribute the size of the gain here to the form in which action information is stored. A nonlinear readout recovers that information from LeWM, and the plan head is itself a linear map, so training with it likely places the same information where a ridge fit on a few hundred episodes can reach it as well.

Table 3: Planner cost on Wall: what each planner trains on the frozen representation and how long it takes, time per evaluation episode, batch-1 median time per plan, and the speedup over our CEM.

Planner	Training on frozen rep.	ms / ep.	ms / plan	Speedup
Closed-form linear	ridge fit, seconds	780	8	$145\times$
MLP readout	300 epochs, $\sim$50 min	900	8	$145\times$
CEM (ours)	none	2,630	1,160	$1\times$
Gradient planner	none	15,600	7,290	$0.16\times$
CEM (DINO-WM)	none	72,000	7,200	$0.16\times$

Table 4: Plan-head design on Push-T, one choice varied at a time against PlanJEPA.

Head	Horizons	Shared	Lin.	MLP	CEM	Grad.
Linear	$\{1\}$	–	65.5 $\pm$3.6	81.5 $\pm$2.2	93.0 $\pm$1.0	**91.0** $\pm$3.6
Linear	$\{1, 2, 3\}$	no	**85.5** $\pm$3.3	**85.0** $\pm$3.0	93.5 $\pm$2.2	89.5 $\pm$2.2
MLP	$\{1, 2, 3\}$	no	64.0 $\pm$11.0	84.5 $\pm$2.2	94.5 $\pm$5.0	89.5 $\pm$5.4
MLP	$\{1, 2, 3\}$	yes	72.0 $\pm$6.2	**85.0** $\pm$1.7	95.0 $\pm$2.2	90.0 $\pm$1.4
Linear (PlanJEPA)	$\{1, 2, 3\}$	yes	82.5 $\pm$3.8	**85.0** $\pm$3.5	**96.0** $\pm$3.7	**91.0** $\pm$4.7

Planning cost. PlanJEPA moves complexity that a planner would otherwise resolve by search at test time into the representation learned at training time. As shown in Table 3, a plan on Wall is then $145\times$ cheaper than under CEM, at a higher success rate. The same gap holds in the other environments, since what sets it is the number of predictor passes each planner needs, which is fixed by the protocol rather than by the task.

Policy learning on LIBERO-Goal. No planner is involved here. The encoder is fine-tuned jointly with an action-chunk policy, and the protocol is described in Appendix A.6. PlanJEPA has the highest success, above LeWM and above PSG-JEPA (Yan et al., 2026), which grounds its encoder in simulator state, and removing either head lowers it. This result extends the planner results above, in that the representation benefits not only goal-conditioned planners but also task-conditioned policy learning across the ten tasks of the LIBERO-Goal benchmark.

5.4 Ablations

Plan-head design. As shown in Table 4, we vary the set of supervised horizons, the sharing of the map and the form of the head, and read each choice under the four controllers. Of the five variants PlanJEPA reaches the highest CEM success, so a shared linear head organizes the latent in a form that search can use. Under the closed-form controller the shared head and one linear map per horizon come out the same, while one map takes a third of the parameters of three. What that controller is sensitive to is the form of the head. A linear head is worth 10 to 21 points there, while under the MLP readout the two forms are level, since a controller with capacity of its own absorbs the difference. Under the gradient planner the five variants span 1.5 points and the single-horizon one is at the top, which is what we expect from Theorem 3, a statement about a single step. Appendix B.3 shows the same pattern on the other environments.

6 Conclusion

In this work, we introduce PlanJEPA, which requires the first action of every k-step transition to be readable from the latents through one shared linear map, so that control-relevant information is organized in a form that is easy to plan with. The constraint adds one linear head at training time and nothing at test time, and it improves candidate ranking, task geometry and linear controllability. Across four planner families, five environments and LIBERO-Goal, PlanJEPA gives consistent gains, and a simple closed-form linear controller recovers most of the success of search at a fraction of its cost. More broadly, our results show that a world model should not only retain control information, but also keep it in a form that a simple controller can access directly.

AI USE STATEMENT

In this work, we used generative AI tools for code assistance and for improving the organization and readability of the manuscript. All AI-assisted content was reviewed and verified by the authors. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work trains and evaluates world models on simulated control benchmarks. It involves no human subjects, no personally identifiable information and no new data collection. The datasets and the released checkpoints of prior methods are public research artifacts and are used under their original terms, and the comparisons to those methods are run with their released code paths where a method supplies one. We see no direct path from these results to harmful application beyond the risks already carried by visual robot learning in general.

REPRODUCIBILITY STATEMENT

Appendix A.1 describes the five environments and the datasets, Appendix A.3 the architecture and the training budget shared by all retrained models, Appendix A.4 the protocol and the budget of each controller family, Appendix A.5 how each baseline is obtained and evaluated, and Appendix A.6 the policy-learning protocol. Every reported cell uses the four fixed seeds 42, 43, 2026 and 7 and the episode counts stated in Appendix A.4. The assumptions and complete proofs of the theoretical results are in Appendix C, and the audit that motivates the Cube initialization is in Appendix A.2.

REFERENCES

Pulkit Agrawal, Ashvin Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning to poke by poking: Experiential learning of intuitive physics. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 29, 2016.

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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629, 2023.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Randall Balestriero and Yann LeCun. LeJEPA: Provable and scalable self-supervised learning without the heuristics. *arXiv preprint arXiv:2511.08544*, 2025.

Ershad Banijamali, Rui Shu, Mohammad Ghavamzadeh, Hung Bui, and Ali Ghodsi. Robust locally-linear controllable embedding. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018.

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

Onur Beker, Mohammad Mohammadi, and Amir Zamir. PALMER: Perception–action loop with memory for long-horizon planning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, 2022.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

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

Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G. Bellemare. DeepMDP: Learning continuous latent space models for representation learning. In *International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, 2019.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, 2018.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2555–2565, 2019.

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

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with discrete world models. In *International Conference on Learning Representations (ICLR)*, 2021.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640:647–653, 2025.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. In *International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pp. 8387–8406, 2022.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations (ICLR)*, 2024.

Riashat Islam, Manan Tomar, Alex Lamb, Yonathan Efroni, Hongyu Zang, Aniket Rajiv Didolkar, Dipendra Misra, Xin Li, Harm Van Seijen, Remi Tachet des Combes, and John Langford. Principled offline RL in the presence of rich exogenous information. In *International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pp. 14390–14421, 2023.

Petr Ivashkov, Randall Balestriero, and Bernhard Schölkopf. Sensorimotor world models: Perception for action via inverse dynamics. *arXiv preprint arXiv:2606.20104*, 2026.

Dinesh Jayaraman and Kristen Grauman. Learning image representations tied to ego-motion. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1413–1421, 2015. doi: 10.1109/ICCV.2015.166.

Michael I. Jordan and David E. Rumelhart. Forward models: Supervised learning with a distal teacher. *Cognitive Science*, 16(3):307–354, 1992. doi: 10.1207/s15516709cog1603_1.

Maximilian Karl, Maximilian Soelch, Justin Bayer, and Patrick van der Smagt. Deep variational bayes filters: Unsupervised learning of state space models from raw data. In *International Conference on Learning Representations (ICLR)*, 2017.

Milan Korda and Igor Mezić. Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. *Automatica*, 93:149–160, 2018. doi: 10.1016/j.automatica.2018.03.046.

Anurag Koul, Shivakanth Sujit, Shaoru Chen, Ben Evans, Lili Wu, Byron Xu, Rajan Chari, Riashat Islam, Raihan Seraj, Yonathan Efroni, Lekan Molu, Miro Dudik, John Langford, and Alex Lamb. PcLast: Discovering plannable continuous latent states. In *International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, 2024.

Yann LeCun. A path towards autonomous machine intelligence, 2022. Version 0.9.2, position paper.

Ian Lenz, Ross A. Knepper, and Ashutosh Saxena. DeepMPC: Learning deep latent features for model predictive control. In *Robotics: Science and Systems (RSS)*, 2015. doi: 10.15607/RSS.2015.XI.012.

Nir Levine, Yinlam Chow, Rui Shu, Ang Li, Mohammad Ghavamzadeh, and Hung Bui. Prediction, consistency, curvature: Representation learning for locally-linear control. In *International Conference on Learning Representations (ICLR)*, 2020.

Wenyuan Li, Guang Li, Keisuke Maeda, Takahiro Ogawa, and Miki Haseyama. Predictive but not plannable: RC-aux for latent world models. *arXiv preprint arXiv:2605.07278*, 2026.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2023.

Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9(1):4950, 2018. doi: 10.1038/s41467-018-07210-0.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. LeWorldModel: Stable end-to-end joint-embedding predictive architecture from pixels. *arXiv preprint arXiv:2603.19312*, 2026.

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

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.

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

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2778–2787, 2017.

Reuven Y. Rubinstein and Dirk P. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. Springer, 2004.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

Rui Shu, Tung Nguyen, Yinlam Chow, Tuan Pham, Khoat Than, Mohammad Ghavamzadeh, Stefano Ermon, and Hung Bui. Predictive coding for locally-linear control. In *International Conference on Machine Learning (ICML)*, 2020.

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 (NeurIPS)*, volume 38, pp. 49126–49162, 2025. doi: 10.52202/085713-1465.

Aravind Srinivas, Allan Jabri, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Universal planning networks: Learning generalizable representations for visuomotor control. In *International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, 2018.

Junhan Sun, Hao Zhao, and Guofeng Zhang. INTACT: Isomorphic intent-to-action learning for search-free world models. *arXiv preprint arXiv:2607.26056*, 2026.

Naoya Takeishi, Yoshinobu Kawahara, and Takehisa Yairi. Learning Koopman invariant subspaces for dynamic mode decomposition. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

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.

Basile Terver, Randall Balestriero, Megi Dervishi, David Fan, Quentin Garrido, Tushar Nagarajan, Koustuv Sinha, Wancong Zhang, Mike Rabbat, Yann LeCun, and Amir Bar. A lightweight library for energy-based joint-embedding predictive architectures. In *ICLR 2026 Workshop on World Models*, 2026.

Ying Wang, Oumayma Bounou, Gaoyue Zhou, Randall Balestriero, Tim G. J. Rudner, Yann LeCun, and Mengye Ren. Temporal straightening for latent planning. In *International Conference on Machine Learning (ICML)*, 2026.

Manuel Watter, Jost Tobias Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 28, 2015.

Haodong Yan, Jiaguan Zhu, Mingyuan Jia, Ruiqing Yin, Junjie He, Zhide Zhong, Junfeng Li, Jinxuan Lu, Hengtao Li, Tianran Zhang, Jiayi Chen, Wenxuan Song, Wen Chen, Yuxiang Gao, and Haoang Li. Is forward prediction enough? Physical state grounding for JEPA world models. *arXiv preprint arXiv:2608.06799*, 2026.

Zhenghao Zhang, Yuanxiang Wang, Zhenyu Guan, Yujia Yang, Bingkang Shi, Tianyu Zong, Hongzhu Yi, Guoqing Chao, Xingchen Chen, Tiankun Yang, Chenxi Bao, Tao Yu, Jingjing Zhou, and Jungang Xu. Delta-JEPA: Learning action-sensitive world models via latent difference decoding. *arXiv preprint arXiv:2606.31232*, 2026. doi: 10.48550/arXiv.2606.31232.

Kai Zhao, Dongliang Nie, Yuchen Lin, Zhehan Luo, Yixiao Gu, Deng-Ping Fan, and Dan Zeng. Sub-JEPA: Subspace Gaussian regularization for stable end-to-end world models. *arXiv preprint arXiv:2605.09241*, 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 (ICML)*, Proceedings of Machine Learning Research, 2025.

A Additional Experimental Details

A.1 Environments and Datasets

Two-Room is a continuous 2D navigation task (Sobal et al., 2025; Maes et al., 2026): the agent must cross between two rooms connected by a single door to reach a target sampled in the other room. **Wall** is the DINO-WM two-room navigation variant (Zhou et al., 2025), with displacement actions and a narrow door. **Reacher** is the two-joint arm from the DeepMind Control Suite (Tassa et al., 2018), evaluated with the strict joint-alignment success criterion used by DINO-WM and LeWM. **Push-T** is the contact-rich T-block pushing task (Chi et al., 2025); we use the dataset and the dataset-matched environment version of Zhou et al. (2025). **Cube** is the OGBench (Park et al., 2025) single-cube pick-and-place task; Cube-PG (pre-grasp) denotes evaluation from starts fixed before the grasp (Appendix A.2). Datasets follow LeWM for Two-Room, Reacher and Cube and DINO-WM for Wall and Push-T. Observations are 224×224 RGB frames with frameskip 5, so one model action spans five environment steps, and Figure A.1 shows one rollout per environment.

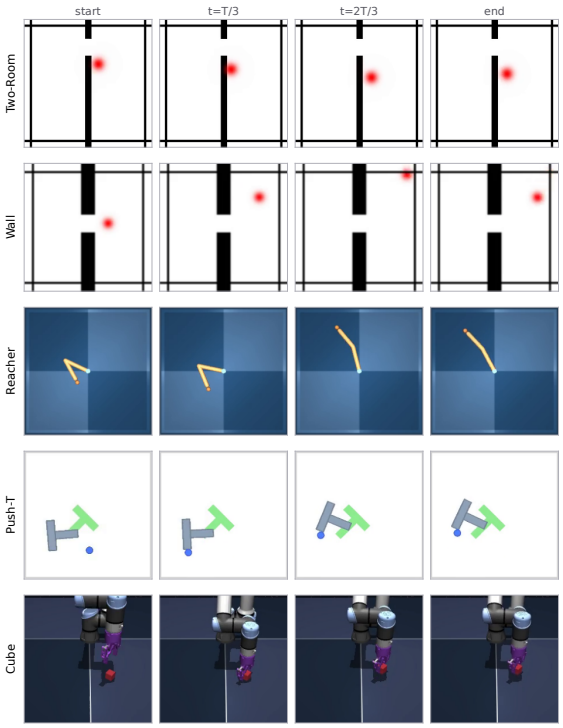


Figure A.1: Evaluation rollouts in the five environments (one episode per row; frames at the start, one third, two thirds, and the end of the episode).

A.2 Why the Pre-Grasp Cube Initialization

We first evaluated Cube under the protocol DINO-WM (Zhou et al., 2025) and LeWM (Maes et al., 2026) use, which starts from the dataset's own ("legacy") states with a 25-step goal offset. As shown in Table A.1, the three plan-regularized encoders land within half a point of each other there, 84.5 to 85.0, and the legacy numbers published with DINO-WM, 86.0, and with PLDM (Sobal et al., 2025), 65.0, fall in the same range. A random policy reaches 42 from the same starts. We therefore audited them (Table A.2). Depending on the goal offset, 16–32% of legacy episodes begin with the start–goal distance already inside the 0.04 success tolerance. Stratifying the legacy starts by grasp state shows that roughly 70% of episodes require no re-grasp at all (18.3% cube already at goal, 43.8% gripper already grasping, 7.2% hand touching), and only 30.8% demand the full approach–grasp–place. A CEM score of 64 under legacy starts decomposes into 60 from the no-grasp strata

and 4 from genuine grasps. We read the flat legacy column of Table A.1 this way: the ordering there follows how often an episode begins already solved.

Table A.1: CEM success (%) on Cube from the dataset's default ("legacy") starts and from the pre-grasp starts used throughout this paper. Every encoder cell is CEM with the same budget on the same four seeds, mean ± std; the random-policy row is a single evaluation of 50 episodes.

Method	Legacy starts	Pre-grasp
Random policy	42.0	8.0
LeWM	68.5 ±7.3	11.5 ±2.6
Sub-JEPA	85.0 ±2.2	10.5 ±1.0
Delta-JEPA	85.0 ±4.6	14.0 ±5.7
PlanJEPA (ours)	84.5 ±3.0	**16.5** ±2.9

Table A.2: Legacy-start bias on Cube (goal offsets 25/50/100) and the pre-grasp protocol.

Start protocol	Trivial episodes	Random-policy success
Legacy, offset 25	32%	42%
Legacy, offset 50	16%	28%
Legacy, offset 100	30%	38%
Pre-grasp start	0%	8%

"Trivial" = start–goal distance already within the 0.04 success tolerance. The non-monotonicity of legacy scores in the offset (offset 100 easier than 50) is explained entirely by the trivial fraction, which is why offset-based protocols are not used in this paper.

The mechanism is the structure of the demonstrations themselves (Figure A.2): in all 10,000 demonstrations the gripper closes on the cube between steps 26 and 28 and lifts it by step 33, so a start sampled at an offset *into* a trajectory lands after the grasp with high probability—the hard part of the manipulation lies behind such a start, not ahead of it.

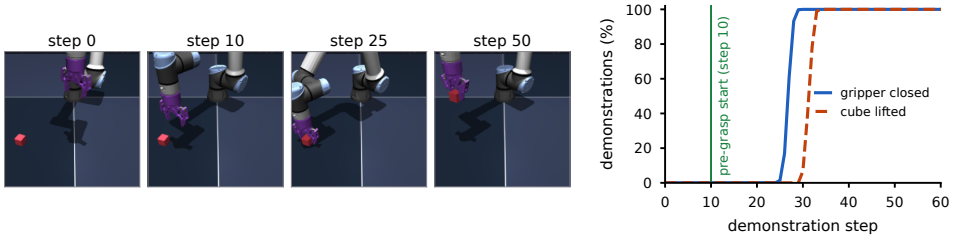


Figure A.2: When the grasp happens in the Cube demonstrations. Left: one demonstration at steps 0, 10, 25, and 50; the cube is still on the table at the pre-grasp start (step 10) and already lifted by step 50. Right: fraction of the 10,000 demonstrations in which the gripper has closed on the cube (solid) or the cube has been lifted (dashed) by a given step; both rise from 0 to 100% between steps 25 and 33.

The pre-grasp initialization instead fixes the start *inside* the pre-grasp window (trajectory step 10) with a +25-step goal, so the grasp lies ahead of every episode. Under it the random-policy floor is 8 and no episode is trivial, so every reported Cube (pre-grasp) number measures the full manipulation. The bias is specific to Cube: re-evaluating biased against unbiased starts on Push-T, Reacher and Two-Room moves CEM by a few points at most, whereas Cube collapses from 76 to 10.

A.3 Architecture and Training

All PlanJEPA and retrained-baseline world models share the LeWM backbone and budget (Table A.3). The training window is $T{=}4$ observations for Reacher, Push-T, and Cube (plan-head horizons $\mathcal{K} = \{1, 2, 3\}$) and $T{=}2$ for the fully-observable single-frame environments Two-Room and Wall ($\mathcal{K} = \{1\}$). The IDM head is a two-layer MLP (hidden 512, BatchNorm); the plan head is a single linear layer on the concatenated pair.

Table A.3: World-model training hyperparameters (shared across methods).

Encoder	ViT-Tiny, patch 14, input 224^2, $d = 192$
Predictor	causal transformer, depth 6, heads 16, MLP 2048
Optimizer	AdamW, lr 5×10^{-5}, weight decay 10^{-3}
Schedule	10 epochs, cosine, batch 128, bf16, grad clip 1.0
SIGReg	weight 0.09 (17 knots, 1024 projections, rank 64)
Action heads	$\lambda_{\text{idm}} = 1.0$, $\lambda_{\text{plan}} = 1.0$

A.4 Planner Protocols

All planners share the same terminal latent goal cost.

Closed-form linear. A bank of ridge regressors, one per horizon $k \leq 8$. From $N{=}1000$ encoded episodes we form pairs $([z_t, z_{t+k}], a_t)$, with k drawn uniformly from the horizons still available at t and a_t the action block of that step, standardize both sides, and fit each map by ridge regression ($\alpha{=}10$), so that $a_t \approx W_k[z_t, z_{t+k}] + b_k$. At test time the controller encodes the current frame and the goal frame, takes k to be the number of steps still to be planned, capped at 8, and applies the corresponding W_k; the plan is recomputed at every step. One plan is therefore two encoder calls and one matrix multiply. On Cube-PG and Push-T, refitting this controller without the episodes that supply evaluation starts and goals changes PlanJEPA's success from 96 to 100 and from 84 to 82 in a single-seed check at $N{=}1000$, and leaves the ranking against Delta-JEPA and LeWM unchanged.

Horizon-conditioned MLP. The GC-IDM-style readout (Nguyen et al., 2026), an MLP on (z_t, z_g, k) fitted on 1500 episodes for 300 epochs with AdamW.

CEM. Receding-horizon cross-entropy-method search (Rubinstein & Kroese, 2004) over the learned predictor: 300 samples, elite 30, horizon 5 action blocks, two replans per episode. The iteration budget is 10 for Two-Room and Reacher and 30 for Push-T, Wall and Cube, equal across rows.

Open-loop CEM. The same sampler with a 10-block horizon, 30 iterations and no replanning.

Gradient planner. The planner of Wang et al. (2026): Adam on the action sequence through the frozen rollout, 100 steps, learning rate 0.01, zero initialization and 100 random restarts, as released.

Every cell of Tables B.1–B.3 is four seeds $\times$ 50 episodes, except the PlanJEPA and LeWM Reacher CEM cells, which use 200 episodes per seed, and uses one checkpoint per environment trained with $\lambda_{\text{idm}}{=}\lambda_{\text{plan}}{=}1.0$. Table 3 compares what one plan costs: the fitted controllers need two encoder calls and one readout, CEM 1.5–$4.5{\times}10^4$ predictor passes, and the gradient planner $5{\times}10^4$ forward–backward passes.

A.5 Baselines

LeWM (Maes et al., 2026) is the backbone of this paper: a compact action-conditioned JEPA world model trained end to end from pixels on offline reward-free trajectories, kept from collapsing by a distributional regularizer on the latents rather than by reconstruction.

Delta-JEPA (Zhang et al., 2026) keeps that design and adds a latent-difference action decoder that predicts the action from the difference between consecutive frame latents, explicitly supervising action-relevant changes.

Sub-JEPA (Zhao et al., 2026) replaces the anti-collapse term by a Gaussian regularizer applied across multiple random orthogonal subspaces of the latent, which stabilizes end-to-end training of the same architecture.

Fast-LeWM (Gao & Xu, 2026) is a faster variant of the same world model.

RC-aux (Li et al., 2026) adds reachability supervision to LeWM, so that latent distance reflects what a planner can reach, and a planner-side cost term that reweights CEM candidates.

INTACT (Sun et al., 2026) jointly learns a structured latent representation and an isomorphic intent-to-action map, amortizing planning into a forward pass with a transformer policy head trained end to end.

DINO-WM (Zhou et al., 2025) builds the world model on frozen DINOv2 patch features instead of learning the encoder, and plans with its own predictor and cost. **DINOv2** (Oquab et al., 2024) is that frozen encoder alone, with no world-model training.

Sub-JEPA, Fast-LeWM, INTACT and RC-aux are evaluated from their official released checkpoints and Delta-JEPA from our reproduction under the shared budget of Table A.3. DINO-WM is trained and planned in its native stack without proprioception, matching our observation space.

A.6 LIBERO-Goal Protocol

The encoder is pretrained for 40 epochs on LIBERO-Goal demonstrations (Liu et al., 2023) with frameskip 5 and 35-dimensional block actions. A task-conditioned policy head (4×1024 MLP, two-frame input, 8-step action chunks, 10-dimensional task one-hot) is then trained for 30 epochs with joint encoder fine-tuning and evaluated closed-loop on 10 tasks $\times$ 50 rollouts. Every method in Table B.4 is retrained from this pipeline with four training seeds, each evaluated with three seeds.

B Additional Results

B.1 Per-Environment Results for Every Controller Family

Tables B.1–B.4 give the per-environment results summarized in Table 2.

Table B.1: Success rate (%) of CEM planning, receding-horizon and open-loop. Mean $\pm$ std over four seeds $\times$ 50 episodes, best in bold; Avg. = mean over environments.

	Two-Room		Reacher		Push-T		Wall		Cube-PG		Avg.	
Method	Rec.	Open	Rec.	Open	Rec.	Open	Rec.	Open	Rec.	Open	Rec.	Open
DINO-WM (native)	**100.0** $\pm$0.0	**99.0** $\pm$1.0	69.0 $\pm$5.4	34.0 $\pm$6.6	61.0 $\pm$3.6	17.5 $\pm$4.3	**97.5** $\pm$1.7	85.0 $\pm$5.4	0.5 $\pm$0.9	0.0 $\pm$0.0	65.6	47.1
Sub-JEPA	90.0 $\pm$3.5	91.0 $\pm$3.6	79.0 $\pm$5.9	44.0 $\pm$1.4	94.5 $\pm$4.1	53.5 $\pm$5.0	74.5 $\pm$1.7	81.0 $\pm$3.6	10.5 $\pm$1.0	7.5 $\pm$1.7	69.7	55.4
Delta-JEPA	99.0 $\pm$1.0	94.0 $\pm$1.4	77.5 $\pm$4.6	37.0 $\pm$3.0	90.5 $\pm$1.6	82.0 $\pm$3.7	79.5 $\pm$4.1	78.0 $\pm$5.5	14.0 $\pm$5.7	13.5 $\pm$0.9	72.1	60.9
LeWM	91.0 $\pm$1.0	94.0 $\pm$2.4	82.0 $\pm$1.6	43.5 $\pm$2.2	93.5 $\pm$3.6	59.0 $\pm$5.2	73.0 $\pm$1.7	75.0 $\pm$3.0	11.5 $\pm$2.6	8.5 $\pm$0.9	70.2	56.0
PlanJEPA (ours)	98.0 $\pm$1.4	96.5 $\pm$2.2	**84.1** $\pm$2.5	**48.0** $\pm$7.3	96.0 $\pm$3.7	75.0 $\pm$4.6	90.0 $\pm$2.4	**89.0** $\pm$4.1	**16.5** $\pm$2.9	**14.5** $\pm$2.2	**76.9**	**64.6**
w/o plan head	91.0 $\pm$2.2	96.0 $\pm$2.4	80.0 $\pm$5.1	46.5 $\pm$5.7	**96.5** $\pm$1.7	**84.0** $\pm$1.4	71.5 $\pm$2.6	76.0 $\pm$4.2	15.5 $\pm$0.9	9.5 $\pm$3.0	70.9	62.4
w/o IDM head	93.5 $\pm$0.9	96.0 $\pm$1.4	76.0 $\pm$4.5	44.5 $\pm$7.8	93.0 $\pm$1.0	69.5 $\pm$6.5	78.5 $\pm$2.2	83.5 $\pm$6.7	14.5 $\pm$1.7	11.0 $\pm$2.2	71.1	60.9

Table B.2: Success rate (%) of readout control on the frozen representation, closed-form and MLP readout. Conventions as in Table B.1.

	Two-Room		Reacher		Push-T		Wall		Cube-PG		Avg.	
Method	Lin.	MLP	Lin.	MLP	Lin.	MLP	Lin.	MLP	Lin.	MLP	Lin.	MLP
DINOv2 (frozen)	95.0 $\pm$1.0	96.0 $\pm$2.8	8.5 $\pm$2.6	71.5 $\pm$7.2	7.0 $\pm$4.6	23.5 $\pm$1.0	92.5 $\pm$2.2	97.0 $\pm$2.6	20.5 $\pm$4.8	82.0 $\pm$4.8	44.7	74.0
Sub-JEPA	90.5 $\pm$0.9	**99.0** $\pm$1.2	19.0 $\pm$5.4	87.0 $\pm$3.5	42.5 $\pm$4.8	84.5 $\pm$3.4	93.0 $\pm$3.3	98.0 $\pm$1.6	11.0 $\pm$1.7	98.0 $\pm$1.6	51.2	93.3
Delta-JEPA	96.5 $\pm$2.6	98.5 $\pm$1.9	29.0 $\pm$8.3	93.5 $\pm$5.5	61.0 $\pm$8.3	84.5 $\pm$4.4	93.5 $\pm$3.6	98.0 $\pm$1.6	89.0 $\pm$3.0	**100.0** $\pm$0.0	73.8	94.9
LeWM	96.0 $\pm$2.4	98.0 $\pm$2.3	16.5 $\pm$5.0	88.5 $\pm$2.9	40.5 $\pm$6.8	79.5 $\pm$4.4	93.0 $\pm$3.6	98.5 $\pm$1.0	6.5 $\pm$1.7	94.0 $\pm$3.7	50.5	91.7
PlanJEPA (ours)	**97.5** $\pm$2.2	97.5 $\pm$2.6	**31.0** $\pm$6.6	**95.5** $\pm$1.7	**82.5** $\pm$3.8	**85.0** $\pm$3.5	**94.0** $\pm$3.5	**99.0** $\pm$1.2	**98.5** $\pm$1.7	99.5 $\pm$0.9	**80.7**	**95.3**
w/o plan head	96.0 $\pm$2.4	96.5 $\pm$3.0	22.5 $\pm$4.8	90.0 $\pm$3.7	58.5 $\pm$7.9	84.0 $\pm$6.2	93.5 $\pm$3.0	98.0 $\pm$1.4	82.5 $\pm$6.5	99.5 $\pm$0.9	70.6	93.6
w/o IDM head	96.5 $\pm$1.7	98.0 $\pm$2.0	28.0 $\pm$2.4	91.5 $\pm$2.2	75.5 $\pm$8.6	75.0 $\pm$5.4	**94.0** $\pm$3.2	**99.0** $\pm$1.0	**98.5** $\pm$1.7	99.5 $\pm$0.9	78.5	92.6

Table B.3: Success rate (%) of gradient-based planning through the learned predictor. Conventions as in Table B.1.

Method	Two-Room	Reacher	Push-T	Wall	Cube-PG	Avg.
Sub-JEPA	94.0 ±3.5	80.5 ±5.7	87.5 ±3.8	**96.5** ±0.9	11.0 ±2.2	73.9
Delta-JEPA	95.0 ±3.3	77.5 ±8.2	87.5 ±3.3	94.5 ±3.6	13.5 ±2.6	73.6
LeWM	87.5 ±3.3	73.0 ±2.2	82.5 ±2.6	92.0 ±1.4	10.0 ±3.2	69.0
PlanJEPA (ours)	**97.0** ±1.7	**81.0** ±7.0	**91.0** ±4.7	**96.5** ±0.9	15.0 ±1.7	**76.1**
w/o plan head	96.5 ±3.0	79.5 ±5.5	**90.0** ±2.4	93.0 ±1.0	11.5 ±1.7	74.1
w/o IDM head	94.0 ±3.7	79.0 ±5.4	85.0 ±7.0	95.5 ±1.7	**16.0** ±3.2	73.9

Table B.4: LIBERO-Goal success (%): mean ± std over four training seeds, each evaluated with three evaluation seeds.

Method	Success
Sub-JEPA	77.2 ±3.9
Delta-JEPA	80.7 ±2.2
LeWM	80.3 ±1.0
PSG-JEPA (state-sup.)	82.7 ±2.1
PlanJEPA (ours)	**83.2** ±3.0
w/o plan head	79.6 ±3.4
w/o IDM head	79.9 ±5.1

B.2 External Checkpoints across Control Interfaces

Three LeWM-based methods release checkpoints, and Table B.5 runs each of them through the control interfaces of this paper so that what is compared is the representation rather than the planning stack. Two interfaces belong to a single method and are reported with it: INTACT's transformer policy head, trained jointly with its encoder, and RC-aux's reachability-weighted CEM, which multiplies the latent goal cost of a candidate by $\max(1 - w\,r,\ 0.05)$, where r is the largest reachability its head predicts along that candidate's rollout and w is its published per-environment weight: 0.35 on Two-Room and Reacher, 0 on Cube-PG, where the row therefore coincides with plain CEM, and 0.35 on Push-T here. The term is worth 3.0 points on Two-Room and 6.0 on Reacher. The closed-form rows are fitted on 100 episodes throughout, so they compare with each other rather than with Table B.2, and the RC-aux Push-T column uses a model we trained with its released recipe, since that checkpoint is not released.

Table B.5: Released checkpoints of RC-aux, INTACT and Fast-LeWM across control interfaces, with PlanJEPA under the same protocols. Success rate (%), four seeds; best per column within each interface in bold. INTACT and Fast-LeWM release no Wall configuration.

Method	Interface	Two-Room	Reacher	Push-T	Cube-PG	Avg.
PlanJEPA	CEM	**98.0±1.4**	**84.1±2.5**	**96.0±3.7**	**16.5±2.9**	**73.7**
PlanJEPA	closed-form linear	**94.5±1.7**	**38.5±7.0**	**67.5±5.0**	**88.0±1.4**	**72.1**
PlanJEPA	horizon-cond. MLP	97.5±2.6	**95.5±1.7**	85.0±3.5	**99.5±0.9**	**94.4**
INTACT	CEM	54.0±4.0	80.0±9.9	76.0±11.2	2.5±2.2	53.1
INTACT	direct policy head	84.0±1.4	97.5±0.9	80.0±4.0	99.0±1.0	90.1
INTACT	closed-form linear	67.5±3.0	19.5±3.0	25.5±4.8	10.5±0.9	30.8
INTACT	horizon-cond. MLP	**98.5±1.7**	88.0±2.4	81.0±3.0	98.5±1.7	91.5
RC-aux	CEM	94.0±1.4	79.4±2.5	3.5±0.9	11.5±1.7	47.1
RC-aux	reachability-weighted CEM	97.0±1.0	85.4±1.7	3.0±3.3	11.5±1.7	49.2
RC-aux	closed-form linear	76.0±4.5	20.0±5.1	21.5±3.6	7.5±1.7	31.3
RC-aux	horizon-cond. MLP	98.0±1.4	89.0±3.3	**86.0±2.4**	94.5±5.4	91.9
Fast-LeWM	CEM	90.5±1.7	78.5±5.7	90.5±5.7	7.0±2.2	66.6
Fast-LeWM	closed-form linear	75.0±6.1	20.0±4.5	25.5±5.9	11.5±2.2	33.0
Fast-LeWM	horizon-cond. MLP	**98.5±0.9**	91.0±4.6	84.5±3.6	84.0±4.9	89.5

B.3 Plan-Head Design under Every Controller Family

Table B.6 repeats the design comparison of Table 4 on every environment and under all four controller families.

Table B.6: Table 4 extended to the four controller families of Table 2 and to a third environment: closed-form linear controller, MLP readout, CEM and the gradient planner, under the protocols of Appendix A.4. Success (%), mean ± std over four seeds; all variants but the single-horizon one supervise $\mathcal{K}=\{1,2,3\}$, and the best entry per column and environment is in bold.

Plan head	Lin.	MLP	CEM	Grad.
Push-T				
Linear, single horizon	65.5 ±3.6	81.5 ±2.2	93.0 ±1.0	**91.0** ±3.6
Linear, one map per horizon	**85.5** ±3.3	**85.0** ±3.0	93.5 ±2.2	89.5 ±2.2
MLP, one map per horizon	64.0 ±11.0	84.5 ±2.2	94.5 ±5.0	89.5 ±5.4
MLP, shared map	72.0 ±6.2	**85.0** ±1.7	95.0 ±2.2	90.0 ±1.4
PlanJEPA (linear, shared)	82.5 ±3.8	**85.0** ±3.5	**96.0** ±3.7	**91.0** ±4.7
Cube-PG				
Linear, single horizon	88.0 ±3.7	**100.0** ±0.0	**16.5** ±1.7	13.5 ±1.7
Linear, one map per horizon	**98.5** ±0.9	**100.0** ±0.0	15.0 ±2.2	13.0 ±3.0
MLP, one map per horizon	94.0 ±3.5	99.5 ±0.9	14.5 ±2.2	12.5 ±3.0
MLP, shared map	92.0 ±3.7	**100.0** ±0.0	12.0 ±2.4	11.0 ±3.3
PlanJEPA (linear, shared)	**98.5** ±1.7	99.5 ±0.9	**16.5** ±2.9	**15.0** ±1.7
Reacher				
Linear, single horizon	25.5 ±3.8	93.5 ±3.8	**84.1** ±1.4	**81.0** ±5.2
Linear, one map per horizon	28.0 ±4.2	93.5 ±2.6	83.6 ±3.1	80.5 ±4.3
MLP, one map per horizon	25.0 ±3.0	89.0 ±6.4	80.4 ±2.4	**83.0** ±5.4
MLP, shared map	27.0 ±4.6	91.0 ±7.3	82.9 ±2.4	74.5 ±4.1
PlanJEPA (linear, shared)	**31.0** ±6.6	**95.5** ±1.7	**84.1** ±2.5	**81.0** ±7.0
Two-Room (single horizon; head form only)				
w/o plan head	96.0 ±2.4	96.5 ±3.0	91.0 ±2.2	96.5 ±3.0
PlanJEPA (linear)	**97.5** ±2.2	**97.5** ±2.6	**98.0** ±1.4	**97.0** ±1.7
Wall (single horizon; head form only)				
w/o plan head	93.5 ±3.0	98.0 ±1.4	71.5 ±2.6	93.0 ±1.0
PlanJEPA (linear)	**94.0** ±3.5	**99.0** ±1.2	**90.0** ±2.4	**96.5** ±0.9

B.4 Reachable-Candidate Ranking

Protocol. For each environment we draw 100 start states from dataset episodes (seed 0) and take the frame 25 steps later as the goal, the offset of the main-table protocol. From each start we sample 64 action sequences of five blocks from a Gaussian matched to the dataset's per-dimension action statistics, execute each one in the simulator, and record the reached frame and state. The encoder cost is $\|E(o_H) - E(o_g)\|^2$ on the reached frame; the rollout cost is $\|F(z_t, \mathbf{a}) - E(o_g)\|^2$ from the predictor, the quantity CEM and gradient planning minimize. The task distance uses the true state: geodesic distance over the free cells of the maze on Two-Room and Wall, chord distance on the wrapped joint angles on Reacher, standardized block position, block orientation, and agent position on Push-T, and standardized block and end-effector positions plus a unit penalty for a different grasp state on Cube-PG. Each start–goal pair gives one Spearman correlation over its 64 candidates, and we report the mean and s.e.m. over pairs.

Head ablations. Figure B.1 and Table B.7 add the w/o-IDM-head variant to Table 1. The linear plan head alone already ranks true endpoints better than the backbone in every environment, and on Two-Room it matches the full objective; adding the IDM head raises the ranking of the predictor's own rollouts further on Wall, Push-T, and Cube-PG. For every encoder in the table the latent argmin lands in the best 3–11% of candidates by task distance, against 50% for a random choice, so the correlation rather than the top choice is the discriminating statistic. Figure B.2 shows one start–goal pair of each remaining environment.

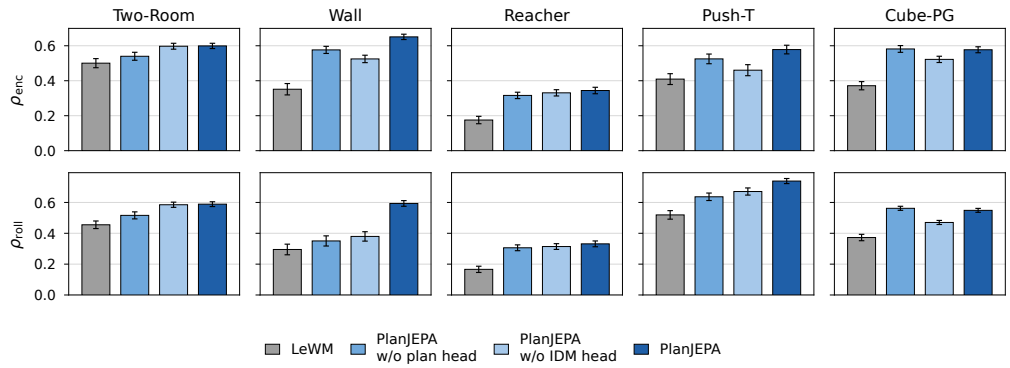


Figure B.1: Reachable-candidate ranking of the LeWM backbone and the PlanJEPA family: Spearman correlation between the latent goal cost and the true task distance over 64 reachable candidates, using the encoded true endpoint (ρ_{enc}, top) or the predictor's rollout (ρ_{roll}, bottom); mean $\pm$ s.e.m. over 100 start–goal pairs.

Table B.7: Reachable-candidate ranking of the LeWM backbone and the PlanJEPA family: Spearman correlation between the latent goal cost and the true task distance over 64 reachable candidates, mean (s.e.m.) over 100 start–goal pairs; best per column in bold.

Method	Two-Room	Reacher	Push-T	Wall	Cube-PG
ρ_{enc}: encoded true endpoint					
LeWM	0.50 (0.03)	0.18 (0.02)	0.41 (0.03)	0.35 (0.03)	0.37 (0.02)
w/o plan head	0.54 (0.02)	0.32 (0.02)	0.52 (0.03)	0.58 (0.02)	**0.58** (0.02)
w/o IDM head	0.60 (0.02)	0.33 (0.02)	0.46 (0.03)	0.52 (0.02)	0.52 (0.02)
PlanJEPA (ours)	**0.60** (0.02)	**0.34** (0.02)	**0.58** (0.02)	**0.65** (0.02)	0.58 (0.02)
ρ_{roll}: predictor rollout					
LeWM	0.46 (0.02)	0.17 (0.02)	0.52 (0.03)	0.30 (0.03)	0.37 (0.02)
w/o plan head	0.52 (0.02)	0.31 (0.02)	0.64 (0.02)	0.35 (0.03)	**0.56** (0.01)
w/o IDM head	0.59 (0.02)	0.31 (0.02)	0.67 (0.02)	0.38 (0.03)	0.47 (0.01)
PlanJEPA (ours)	**0.59** (0.02)	**0.33** (0.02)	**0.74** (0.02)	**0.59** (0.02)	0.55 (0.01)

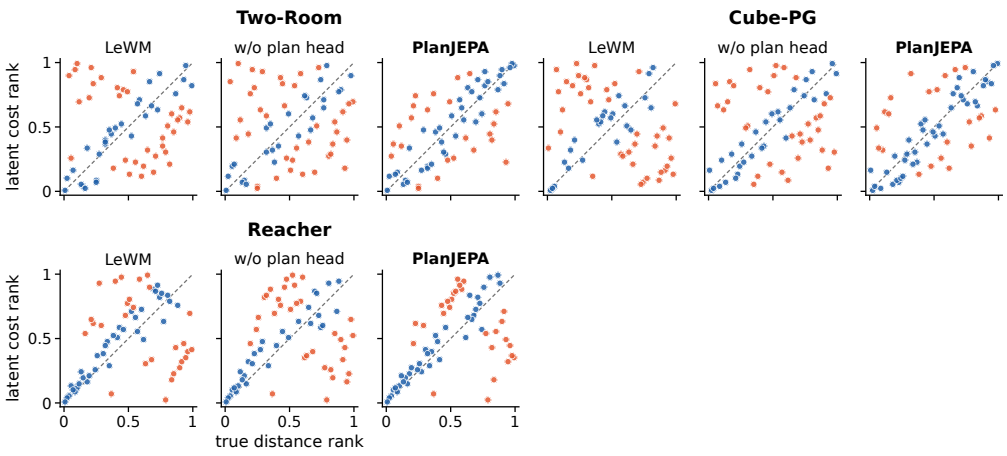


Figure B.2: Figure 3 for the remaining environments: one start–goal pair each of Two-Room, Cube-PG, and Reacher, with the same conventions.

B.5 Latent Geometry

All views here use the checkpoints of the main tables and no fitted readout. Table B.8 gives the coordinate smoothness on all five environments, measured as one minus the mean absolute true-coordinate difference between each latent and its 12 nearest neighbors in the PCA plane, relative to random pairs. We observe the effect on the navigation environments and on Reacher.

Table B.8: Coordinate smoothness over the PCA plane (1 = position-consistent neighborhoods, protocol of Figure 4) on all five environments, for the two true coordinates colored in Figures 4 and B.3.

	Two-Room		Reacher		Push-T		Wall		Cube-PG	
Method	x	y	x	y	x	y	x	y	x	y
LeWM	0.42	0.51	0.27	0.35	0.14	0.16	0.45	0.62	0.16	0.33
PlanJEPA (ours)	**0.93**	**0.92**	**0.49**	**0.54**	0.13	0.13	**0.83**	**0.92**	**0.24**	0.34
w/o plan head	0.44	0.61	0.27	0.40	**0.17**	0.15	0.48	0.80	0.20	0.29
w/o IDM head	**0.93**	0.91	0.43	0.52	0.13	0.12	0.78	0.81	0.23	**0.37**

Figure B.3 repeats the PCA view of Figure 4 on the remaining environments. On Push-T and Cube-PG the leading two components carry only 4–5% of the latent variance and no encoder's plane is coordinate-consistent, which is why Figure 1(b) is restricted to navigation.

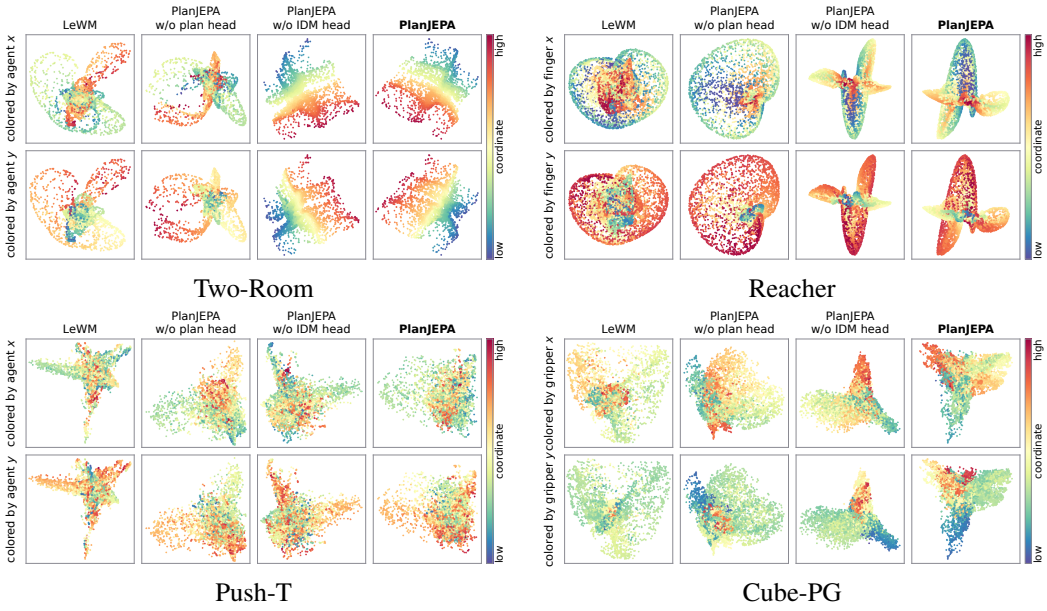


Figure B.3: The PCA view of Figure 4 on the remaining environments: the first two principal components of the latents of 260 expert episodes, each frame colored by the true coordinate named on the row (agent position on Two-Room and Push-T, fingertip position on Reacher, gripper position on Cube-PG), blue = low, red = high.

Figures B.4 and B.5 track the same measure over training, and the predictor inherits the structure, in that rolling the learned dynamics forward with the true actions reproduces the encoded trajectories most closely for PlanJEPA.

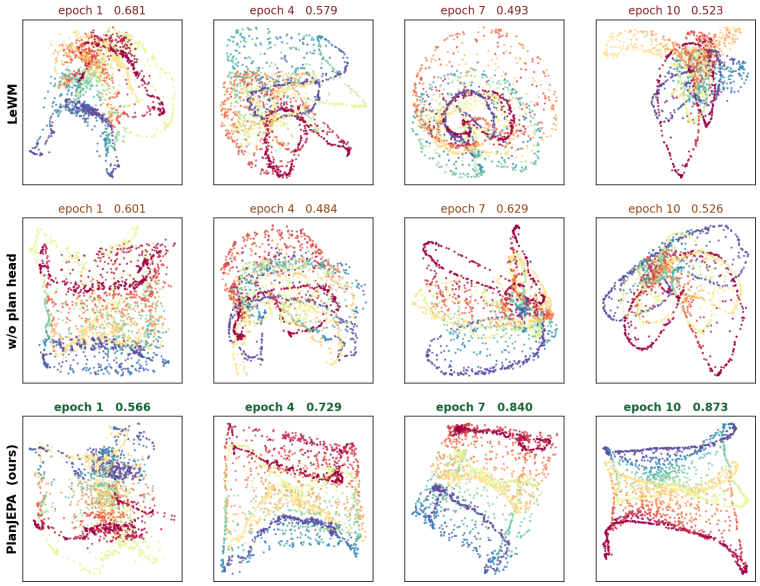


Figure B.4: Emergence of the coordinate structure on Wall. Latents of 260 expert episodes projected on their first two principal components at successive epochs, every frame colored by the true agent x (blue = low, red = high). The number above each panel is the coordinate smoothness of that coordinate on this draw of episodes.

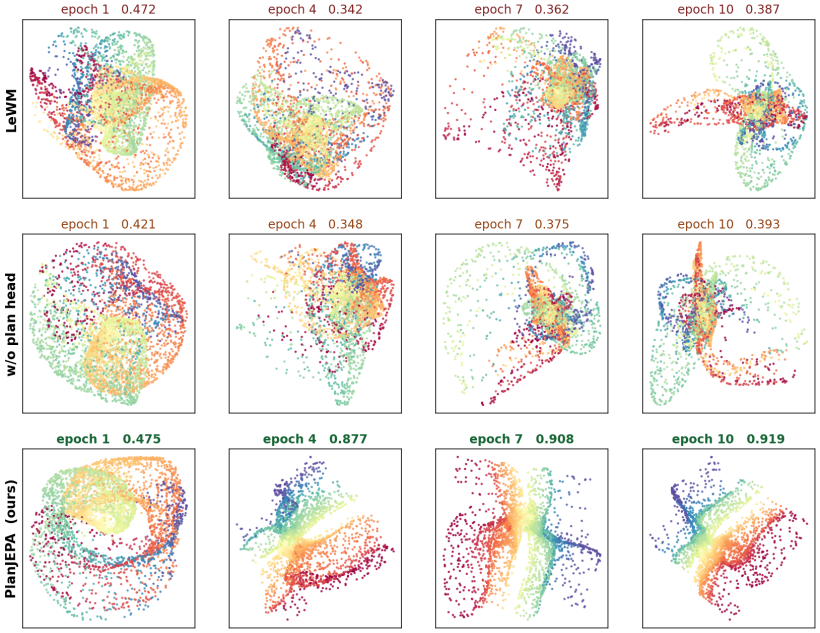


Figure B.5: Emergence of the coordinate structure on Two-Room, colored by the true agent x; conventions as in Figure B.4.

B.6 Plan-Weight Sensitivity

Tables B.9 and B.10 sweep the plan weight in the five environments (λ_{idm}=1.0; CEM and the closed-form controller under the main-table protocols). On Wall, CEM rises with the weight to 99

at 30 and stays there up to 300; Two-Room sits at its ceiling from 10 on; everywhere else the results are insensitive to the weight within seed noise, so the main tables use λ_{plan}=1.0.

Table B.9: Plan-weight sweep on the two navigation environments, whose plan head has the single horizon k=1: CEM and the closed-form controller of Table B.2 under the main-table protocols. Success rate (%), mean ± std over four seeds; λ_{plan}=1.0 is the main-table checkpoint; best per row in bold.

Environment	Planner	0.3	1.0	3	10	30	100	300
Wall	CEM	84.0 ±3.5	90.0 ±2.4	92.0 ±3.7	96.0 ±2.4	**99.0** ±1.0	98.0 ±1.4	95.5 ±3.6
	Closed-form	93.5 ±3.0	**94.0** ±3.5	93.5 ±3.0	92.0 ±2.4	93.0 ±3.0	**94.0** ±2.4	93.5 ±1.7
Two-Room	CEM	95.5 ±1.7	98.0 ±1.4	96.5 ±1.7	99.5 ±0.9	96.5 ±2.2	99.5 ±0.9	**100.0** ±0.0
	Closed-form	97.0 ±3.0	**97.5** ±2.2	97.0 ±3.0	96.0 ±2.4	97.0 ±1.0	97.0 ±1.0	96.5 ±1.6

Table B.10: Plan-weight sweep on the three multi-horizon environments, conventions as in Table B.9.

Environment	Planner	0.3	1.0	3	10
Reacher	CEM	83.5 ±1.2	**84.1** ±2.5	83.4 ±0.9	83.1 ±2.0
	Closed-form	26.0 ±3.7	**31.0** ±6.6	29.0 ±4.6	30.0 ±6.2
Push-T	CEM	**96.0** ±3.7	**96.0** ±3.7	92.0 ±4.9	83.5 ±5.7
	Closed-form	85.0 ±3.0	82.5 ±3.8	**88.0** ±3.2	82.5 ±3.8
Cube-PG	CEM	16.5 ±3.0	16.5 ±2.9	**18.0** ±2.8	13.0 ±2.2
	Closed-form	**99.0** ±1.0	98.5 ±1.7	97.0 ±2.2	98.5 ±2.6

B.7 Measured Quantities of Theorem 1

Table B.11 reports, on the two environments where the linear relation is closest to realizable, the two quantities Theorem 1 speaks about: how well the shared linear map fits held-out transitions and the separation coefficient $1/\|V\|_2$ that map implies. The plan objective raises the fit from 0.50 to 0.79 on Push-T and from 0.44 to 0.74 on Cube-PG; on Push-T it also raises the coefficient from 0.51 to 0.91, whereas on Cube-PG the coefficients of all encoders lie within 0.25–0.30 and the fit is what separates them. The separation the predictor actually produces between the endpoints of two different first actions stays above the coefficient for every encoder, as Eq. (6) requires, so it is not tabulated.

Table B.11: Measured quantities of Theorem 1 on Push-T and Cube-PG: held-out fit (R^2) of the shared linear map $a_t \approx Uz_t + Vz_{t+k}$, $k \in \mathcal{K}$, on latents normalized as in Appendix C.2 and standardized action blocks (500 fitting and 150 test episodes), and the separation coefficient $1/\|V\|_2$ of Eq. (6) for that map; best per column in bold.

	Push-T		Cube-PG	
Method	R^2	$1/\|V\|_2$	R^2	$1/\|V\|_2$
LeWM	0.497	0.510	0.440	0.256
Sub-JEPA	0.498	0.519	0.429	0.252
Delta-JEPA	0.570	0.482	0.652	0.291
DINOv2 (frozen)	0.400	0.351	0.415	0.284
PlanJEPA (ours)	0.791	**0.911**	**0.743**	**0.303**
w/o plan head	0.559	0.503	0.587	0.299
w/o IDM head	**0.806**	0.906	0.735	0.302

B.8 Frozen Encoder with a Retrained Predictor

The CEM and gradient rows of Section 5.3 plan through a predictor trained jointly with each encoder, so part of their gain could sit in that predictor rather than in the representation. For Table B.12 we freeze each encoder, retrain the predictor and action encoder from scratch with the prediction loss alone under the original architecture, data and budget, and rerun CEM unchanged. The retrained predictors reproduce the success of the jointly trained ones within two points for every encoder, and PlanJEPA's margin over the backbone is unchanged, at $+13$ to $+14$ on Wall and $+7$ to $+9$ on Two-Room. We therefore locate the search gains of Table B.1 in the encoder rather than in the predictor.

Table B.12: CEM success (%) with the predictor trained jointly with the encoder (Joint) and with a predictor retrained from scratch on the frozen encoder (New); mean $\pm$ std over seeds (Wall: seeds 42/1/2, 300 samples, 10 iterations; Two-Room: the protocol of Table B.1).

	Wall		Two-Room	
Method	Joint	New	Joint	New
LeWM	75.3 $\pm$6.6	73.3 $\pm$6.2	91.0 $\pm$1.0	89.0 $\pm$3.6
PlanJEPA (ours)	**88.7** $\pm$0.9	**87.3** $\pm$2.5	**98.0** $\pm$1.4	**98.0** $\pm$2.0
w/o plan head	75.3 $\pm$4.7	74.0 $\pm$7.1	91.0 $\pm$2.2	92.0 $\pm$2.4

C Theoretical Analysis

C.1 Setup and Notation

Throughout, the object of analysis is the linear term $\mathcal{L}_{\text{plan}}$ of Eq. (3), trained inside the full objective of Eq. (4) with the shared map of Section 3. For each horizon $k \in \mathcal{K}$, define the plan residual $e_k := a_t - U_k z_t - V_k z_{t+k}$, whose mean square is the horizon-k term of Eq. (3). For clarity, the main text states the exact realizable case $e_k = 0$. We provide below both the exact result and its residual-robust version. We additionally assume isotropic latent normalization,

$$\mathbb{E}[z_t] = 0, \qquad \mathrm{Cov}(z_t) = I, \tag{10}$$

which idealizes the effect of SIGReg and fixes the global scale of the representation.

C.2 Scale Identifiability under Isotropic Regularization

The role of the two regularizers is complementary: SIGReg fixes the global scale of the representation, whereas the linear plan objective constrains how control information is organized within that representation; Eq. (10) idealizes the former. We first explain why representation normalization is necessary for interpreting the separation coefficient $1/\|V_k\|_2$. Without Eq. (10), consider the rescaling $\widetilde{z} = cz$ with $c > 0$. Then $a_t = U_k z_t + V_k z_{t+k} = \frac{1}{c} U_k \widetilde{z}_t + \frac{1}{c} V_k \widetilde{z}_{t+k}$, hence $\widetilde{V}_k = V_k / c$ and consequently $1/\|\widetilde{V}_k\|_2 = c/\|V_k\|_2$. Thus, the numerical separation coefficient can be changed arbitrarily by rescaling the latent representation. Under Eq. (10), $\mathrm{Cov}(\widetilde{z}) = c^2 I$, so $\widetilde{z}$ satisfies the same normalization only when $c = 1$; the trivial global scaling degree of freedom is consequently removed. Moreover, the remaining orthogonal coordinate transformation $\widetilde{z} = Qz$ with $Q \in \mathbb{R}^{d \times d}$, $Q^\top Q = I$, does not change the spectral norm of the plan head, since $\widetilde{V}_k = V_k Q^\top$ and $\|V_k Q^\top\|_2 = \|V_k\|_2$. Hence $1/\|V_k\|_2$ is invariant to the natural orthogonal symmetry of an isotropically normalized latent space.

C.3 Proof of Multi-Horizon Action Separation

Proof of Theorem 1. Consider two trajectory segments sharing the same initial latent z_t, $a_t = U_k z_t + V_k z_{t+k}$ and $a'_t = U_k z_t + V_k z'_{t+k}$. Whenever $a_t \neq a'_t$, these two relations force $V_k \neq 0$, so $1/\|V_k\|_2$ is well defined in every non-vacuous instance of the bound. Subtracting the two equations

gives $a_t - a'_t = V_k(z_{t+k} - z'_{t+k})$; taking Euclidean norms and using submultiplicativity,

$$\|a_t - a'_t\|_2 \le \|V_k\|_2 \|z_{t+k} - z'_{t+k}\|_2, \tag{11}$$

and rearranging yields Eq. (6). The result holds separately for every $k \in \mathcal{K}$. □

C.4 Approximate Action Separation

The exact inverse relation is an analytical idealization. We next give a robust version for nonzero plan residuals.

Proposition 4 (Residual-robust separation)**.** *Consider two trajectory segments with the same z_t, and let their inverse residuals be e_k and e'_k, i.e., $a_t = U_k z_t + V_k z_{t+k} + e_k$ and $a'_t = U_k z_t + V_k z'_{t+k} + e'_k$. Then*

$$\|z_{t+k} - z'_{t+k}\|_2 \ge \frac{\|a_t - a'_t\|_2 - \|e_k\|_2 - \|e'_k\|_2}{\|V_k\|_2}. \tag{12}$$

The bound is understood as trivial when the numerator is negative.

Proof. Subtracting the two inverse relations gives $a_t - a'_t = V_k(z_{t+k} - z'_{t+k}) + (e_k - e'_k)$. By the triangle inequality,

$$\|a_t - a'_t\|_2 \le \|V_k\|_2 \|z_{t+k} - z'_{t+k}\|_2 + \|e_k - e'_k\|_2 \le \|V_k\|_2 \|z_{t+k} - z'_{t+k}\|_2 + \|e_k\|_2 + \|e'_k\|_2, \tag{13}$$

and rearranging proves Eq. (12). □

Proposition 4 makes explicit that the separation guarantee degrades gracefully with inverse-dynamics error.

C.5 Comparison with Nonlinear Inverse Dynamics

Suppose instead that actions are reconstructed by a generic nonlinear decoder $a_t = g_k(z_t, z_{t+k})$. Perfect reconstruction alone only guarantees injectivity with respect to action on the observed transitions: two identical decoder inputs cannot map to different actions. To obtain a quantitative geometric separation, an additional regularity condition is required. If g_k is L_g-Lipschitz in its second argument, then

$$\|a_t - a'_t\|_2 = \|g_k(z_t, z_{t+k}) - g_k(z_t, z'_{t+k})\|_2 \le L_g \|z_{t+k} - z'_{t+k}\|_2, \tag{14}$$

and hence $\|z_{t+k} - z'_{t+k}\|_2 \ge \|a_t - a'_t\|_2 / L_g$. Thus, the qualitative notion of action separation is not exclusive to the linear plan head. The distinction is that an unconstrained nonlinear decoder may realize an arbitrarily large and difficult-to-control L_g, whereas for the linear decoder the relevant quantity is explicitly given by $\|V_k\|_2$. This exposes the geometry induced by the inverse model and motivates using a deliberately weak linear decoder as a representation-shaping objective.

C.6 Proof of Non-Vanishing One-Step Action Sensitivity

Theorem 1 applies to all horizons in $\mathcal{K}$ without assuming that z_{t+k} is a function of the first action alone; we now specialize to $k = 1$. The plan head is trained on encoded next latents, so the relation below transfers to the learned predictor to the extent that Eq. (1) makes F_θ reproduce them. Let $z_{t+1} = F(z_t, a)$ with F differentiable in a; Assumption 1 states that $a = U_1 z_t + V_1 F(z_t, a)$ holds locally.

Proof of Theorem 2. Differentiate both sides with respect to a. Since z_t is fixed, $I = V_1 \, \partial F(z_t, a)/\partial a = V_1 J(a)$. Since $V_1 J(a)$ equals the identity on the action space, necessarily $\dim(a) \le d$, and $\sigma_{\min}(J(a))$ below denotes the smallest singular value of the $d \times \dim(a)$ matrix $J(a)$. For any vector x, $\|x\|_2 = \|V_1 J(a) x\|_2 \le \|V_1\|_2 \|J(a)x\|_2$, therefore $\|J(a)x\|_2 \ge \|x\|_2 / \|V_1\|_2$, and taking the minimum over all unit vectors x gives $\sigma_{\min}(J(a)) \ge 1/\|V_1\|_2$. □

The use of $k = 1$ here is essential. For $k > 1$, the endpoint generally depends on the entire action sequence, $z_{t+k} = F_k(z_t, a_t, a_{t+1}, \ldots, a_{t+k-1})$, while our inverse objective reconstructs only the first action a_t. Therefore the identity $V_k \, \partial z_{t+k}/\partial a_t = I$ does not follow without additional assumptions on the subsequent actions. For this reason, the gradient-based result is intentionally stated only for the one-step relation.

C.7 Local Hessian Conditioning

Consider $\ell(a) = \frac{1}{2}\|F(z_t, a) - z_g\|_2^2$ with F twice differentiable in a, and let $r(a) = F(z_t, a) - z_g$. Then $\nabla_a \ell(a) = J(a)^\top r(a)$, and differentiating again gives

$$\nabla_a^2 \ell(a) = J(a)^\top J(a) + \sum_{i=1}^{d} r_i(a)\, \nabla_a^2 F_i(z_t, a). \tag{15}$$

At a successful solution $a^\star$ satisfying $F(z_t, a^\star) = z_g$ we have $r(a^\star) = 0$, so the second term of Eq. (15) vanishes and $H^\star := \nabla_a^2 \ell(a^\star) = J(a^\star)^\top J(a^\star)$.

Proof of Theorem 3. By Theorem 2, $\sigma_{\min}(J(a^\star)) \geq 1/\|V_1\|_2$. Since the eigenvalues of $J^\top J$ are the squared singular values of J, $\lambda_{\min}(H^\star) \geq 1/\|V_1\|_2^2$; by the assumed upper bound $\|J(a^\star)\|_2 \leq L$ we also have $\lambda_{\max}(H^\star) = \sigma_{\max}(J(a^\star))^2 \leq L^2$. Therefore

$$\frac{1}{\|V_1\|_2^2} I \preceq H^\star \preceq L^2 I, \qquad \kappa(H^\star) = \frac{\lambda_{\max}(H^\star)}{\lambda_{\min}(H^\star)} \leq L^2 \|V_1\|_2^2. \tag{16}$$

□

Scope of the result. Unlike the globally linear dynamics considered in classical quadratic planning analyses, F may be nonlinear here. Accordingly, Theorem 3 is a local statement at a successful goal-reaching solution. It guarantees non-degenerate local curvature but does not by itself imply global convexity or global convergence of gradient descent.

Approximate one-step inverse relation. A useful perturbation form can also be obtained. Suppose locally $a = U_1 z_t + V_1 F(z_t, a) + e(a)$, where $e(a)$ is differentiable. Differentiating gives $I = V_1 J(a) + J_e(a)$ with $J_e(a) = \partial e(a)/\partial a$. If $\|J_e(a)\|_2 \leq \rho < 1$, then, for any unit vector x, $1 = \|x\|_2 \leq \|V_1 J(a)x\|_2 + \|J_e(a)x\|_2 \leq \|V_1\|_2 \|J(a)x\|_2 + \rho$, hence

$$\sigma_{\min}(J(a)) \geq \frac{1 - \rho}{\|V_1\|_2}. \tag{17}$$

Thus the non-vanishing sensitivity result is stable to sufficiently smooth inverse-model approximation error.

C.8 Implications for Downstream Controllers

Linear goal-conditioned controller. For any valid k-step transition with endpoint $z_g = z_{t+k}$, $a_t = U_k z_t + V_k z_g$. Therefore, in the realizable case, a linear goal-conditioned controller recovers the corresponding first action exactly. With nonzero residual, its squared prediction error is precisely the plan residual used in training.

MLP and policy classes. Let $\mathcal{H}_{\text{lin}}$ denote affine action predictors and let $\mathcal{H}_{\text{pol}}$ be a sufficiently expressive MLP policy class containing $\mathcal{H}_{\text{lin}}$. Then

$$\inf_{\pi \in \mathcal{H}_{\text{pol}}} \mathbb{E}\|\pi(z_t, z_g) - a_t\|_2^2 \leq \inf_{\pi \in \mathcal{H}_{\text{lin}}} \mathbb{E}\|\pi(z_t, z_g) - a_t\|_2^2. \tag{18}$$

This establishes representational compatibility with more expressive policies, but does not claim better optimization or generalization than an arbitrary baseline representation.

Sampling-based planners. Suppose one reference segment beginning at z_t reaches target endpoint z_g with first action $a_t^\star$. For another valid segment with the same initial latent and first action a_t, Theorem 1 gives

$$\|z_{t+k} - z_g\|_2 \geq \frac{1}{\|V_k\|_2}\|a_t - a_t^\star\|_2 \qquad \text{and hence} \qquad \frac{1}{2}\|z_{t+k} - z_g\|_2^2 \geq \frac{1}{2\|V_k\|_2^2}\|a_t - a_t^\star\|_2^2. \tag{19}$$

This provides a margin against first-action aliasing in latent endpoint costs.

D Extended Related Work

Section 2 positions PlanJEPA against latent linear control; two further lines shape representations for planning.

Representations shaped by the planning computation. DeepMPC learns latent features for model-predictive control (Lenz et al., 2015); Universal Planning Networks differentiate through latent-space planning so that the representation is trained by the planner's own success (Srinivas et al., 2018); DeepMDP preserves dynamics and reward structure in the latent (Gelada et al., 2019); and PALMER learns a distance tied to traversable experience for long-horizon planning (Beker et al., 2022). Each of these ties the representation to one planner or one cost. PlanJEPA imposes a planner-agnostic constraint on the control relation, and the same representation serves four controller families unchanged.

Trajectory geometry. Temporal straightening penalizes the curvature of latent trajectories so that Euclidean latent distance better reflects feasible progress and the planning Hessian is better conditioned (Wang et al., 2026). Straightening constrains the path between consecutive latents, whereas PlanJEPA constrains the map from latent pairs to actions; the two are complementary, and Section 5 uses the gradient planner of Wang et al. (2026) unchanged.