LOW-DIMENSIONAL DECISION SPACES FOR LATENT WORLD-MODEL PLANNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Latent world models can make visual planning more efficient by predicting in representation space rather than pixel space. However, conventional methods still predict a separate high-dimensional latent trajectory for every candidate action sequence, even though much visual information is shared across candidates. We study the variation among candidate outcomes for a fixed context, goal, and planning horizon, and refer to it as the *decision space*. Our analysis shows that a small number of directions captures most candidate variation while preserving much of the candidate-ranking signal used for action selection. Based on this finding, we introduce the Low-Dimensional Decision World Model (LDD-WM). LDD-WM performs one shared, action-free visual rollout and uses an action-conditioned component to update a compact r-dimensional state for each candidate. The model is trained from physical-cost ordering and avoids reconstructing a separate full latent trajectory for every candidate. Across six datasets, LDD-WM consistently outperforms conventional full-latent reconstruction world models while requiring less computation and lower latency at inference. On MetaWorld Reach-Wall, LDD-WM improves success from 29.9% to 64.6% over JEPA-WM while reducing inference computation by a factor of 150 and running $71\times$ faster. These results support decision-space modeling as an efficient alternative to full-latent prediction for visual planning.

1 INTRODUCTION

Learning to plan for decision-making in a dynamic physical world is a central objective of embodied intelligence. World models learn to predict future states from the current observation and candidate actions, and have shown significant progress in visual planning (Ha & Schmidhuber, 2018; Hafner et al., 2019b;a). Generative world models imagine future observations as images or videos and select the action sequence whose predicted trajectory reaches the visual goal (Ebert et al., 2017; Bar et al., 2025). This approach achieves promising results, however, generating a separate trajectory for each candidate is inherently expensive as rollout computation scales with both the number of candidates and the planning horizon. Recent methods instead plan in the latent space of a pretrained visual model, predicting future representations rather than pixels (Zhou et al., 2025; Assran et al., 2025; Terver et al., 2026). Although more efficient, these methods still predict a separate high-dimensional latent trajectory for every candidate action sequence.

Visual representations are high-dimensional because they encode rich spatial and semantic information. Yet a planner comparing a set of candidate actions may not need to predict all of this information separately for every candidate. When deciding what action to take, we humans do not necessarily reconstruct every possible future trajectory in full detail. Instead, we often focus on how our choices change the outcome, for example, whether an action moves us closer to the goal or toward a wall. Similarly, given the same initial context and goal, much of the latent rollout, including the scene layout and background, can be shared across candidate actions. This suggests decomposing prediction into two parts: visual information that is shared across candidates, and candidate-dependent variation that captures how each action changes the outcome. Such a decomposition avoids repeatedly predicting the same visual information and isolates the differences needed to compare actions.

In this work, we study the variation among candidate outcomes for a fixed visual context, goal, and planning horizon. We refer to the space of these candidate-dependent differences as the *decision space*. Our central finding is that, in latent world models that use representations from a pretrained

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.

visual encoder, a small number of directions can capture most of the candidate-ranking signal used for action selection. We present this analysis in Sec. 3.2.

Motivated by this finding and the intuition above, we introduce the Low-Dimensional Decision World Model (LDD-WM). LDD-WM first predicts one action-free visual rollout from the context and shares it across all candidate actions. We then train an action-conditioned component that queries the shared visual features and updates one r-dimensional state for each candidate action sequence, where r is much smaller than the dimension of the full latent rollout. This component learns the candidate-dependent dynamics needed to compare actions, without reconstructing a separate full latent trajectory for every candidate. For each fixed context, goal, and planning horizon, it is trained to order the candidate actions according to their physical costs. Figure 2 illustrates the architecture of LDD-WM and contrasts it with conventional full-latent world-model planning.

Our contributions can be summarized as follows: **(i)** We identify and measure the decision space of latent world-model planning, showing that a small number of candidate-response directions preserve much of the candidate-ranking signal used for action selection; **(ii)** we introduce LDD-WM, which predicts visual information shared across candidate actions once and models candidate-dependent dynamics in a compact decision space; and **(iii)** we evaluate planning quality and inference efficiency on six datasets spanning simulated visual navigation, robot-arm control, and real-image navigation. On MetaWorld Reach-Wall (Yu et al., 2020), LDD-WM improves success from 29.9% to 64.6% over JEPA-WM while reducing inference computation by a factor of 150 and running $71\times$ faster.

2 Related Work

Visual world models for planning. Visual world models predict the outcomes of candidate actions and use these predictions for planning. Pixel-space models generate an image or video rollout for each action sequence and compare its outcome with a visual goal (Ebert et al., 2017; Bar et al., 2025), but this generation is computationally expensive. Latent world models avoid pixel reconstruction by predicting future states in representation space. DINO-WM predicts DINOv2 (Oquab et al., 2023) features from offline action trajectories (Zhou et al., 2025), V-JEPA 2-AC learns action-conditioned dynamics on pretrained video features (Assran et al., 2025), and JEPA-WM studies architecture, objective, and planning choices for joint-embedding world models (Terver et al., 2026). These methods reduce the cost of visual prediction, but still produce a separate high-dimensional latent rollout for every candidate action sequence. Their reconstruction objectives also do not directly train the model to order candidate action sequences according to physical cost.

Efficient world models. Recent work reduces rollout cost with compact or sparse prediction. DDP-WM combines localized primary dynamics with low-rank background updates (Yin et al., 2026), Sparse Imagination processes fewer visual tokens during rollout (Chun et al., 2026), and CompACT represents each observation with a small set of discrete tokens (Kim et al., 2026). These methods reduce the cost of candidate-specific visual prediction, but repeatedly infer the visual information shared across candidates, once for each candidate action sequence. They do not separate this shared information from the candidate-dependent differences used to select an action. LDD-WM predicts the shared visual context once and learns compact candidate-dependent dynamics directly from physical-cost ordering, without a separate full-latent rollout for every candidate.

3 Visual Planning with Latent World Models

3.1 Latent world-model planning

We study goal-conditioned visual planning, in which the planner receives the most recent L observations $o_{t-L+1:t}$ and a goal image o_g, at time t. A frozen encoder ϕ maps them to context latents $z_{t-L+1:t} = \phi(o_{t-L+1:t})$ and goal latent $z_g = \phi(o_g)$. A proposal distribution samples K action sequences,

$$\mathcal{A} = \{a^k_{t:t+H-1}\}_{k=1}^{K}, \qquad a^k_{t:t+H-1} \sim \pi_{\text{prop}}, \tag{1}$$

where H denotes the planning horizon and π_{prop} denotes the proposal distribution.

A conventional autoregressive latent world model predicts a separate latent trajectory for every sampled action sequence:

$$\hat{z}^k_{t+1:t+H} = F_\theta(z_{t-L+1:t}, a^k_{t:t+H-1}). \tag{2}$$

The planner compares the final predicted latent with the goal latent. Common scores include L_1 and L_2 distances (Terver et al., 2026) and cosine distance (Zhang et al., 2025). We consider mean squared Euclidean distance,

$$s_k = \frac{1}{D}\|\hat{z}^k_{t+H} - z_g\|_2^2. \tag{3}$$

where D is the flattened latent dimension. The planner selects the sequences with the lowest s_k.

3.2 Decision Space in Latent World Models

For our considered goal-conditioned visual planning, the planner selects the candidates with the lowest score, *so only the relative ordering of candidate scores matters*. A score term shared by every candidate therefore cannot change which action is selected. For a fixed visual context, goal, planning horizon, and proposal set, we study how predicted endpoint latents vary across action sequences. We call the space spanned by this candidate-dependent variation the *decision space*. We then show how this variation determines latent scores and examine how those scores correlate with physical costs.

Candidate-relative scores. Let $\hat{z}^k_{t+H} \in \mathbb{R}^D$ denote the latent representation predicted at the final rollout step for action sequence k, and let $z_g \in \mathbb{R}^D$ denote the goal latent. For the fixed context, goal, horizon, and proposal set, choose a reference $z^{\text{ref}} \in \mathbb{R}^D$ shared by every action sequence. We write

$$\hat{z}^k_{t+H} = z^{\text{ref}} + u_k, \qquad q = z^{\text{ref}} - z_g. \tag{4}$$

Here, u_k is the candidate-dependent difference between the final latent and the reference. The vector q is the difference between the reference and the goal. Applying Eq. (3) gives

$$Ds_k = \|q + u_k\|_2^2 = \|q\|_2^2 + 2\langle q, u_k\rangle + \|u_k\|_2^2, \tag{5}$$

where $\langle\cdot,\cdot\rangle$ is the Euclidean scalar product. The first term, $\|q\|_2^2$, is shared by every action sequence and cannot affect their ordering. The second term, $2\langle q, u_k\rangle$, measures the alignment between the response and the goal residual. The third term, $\|u_k\|_2^2$, is the squared magnitude of candidate k's response. Only the last two terms vary across candidates, and together they determine the ordering.

Defining the decision space. To measure the decision space within each candidate pool, we use the mean predicted final latent representation as the reference, $z^{\text{ref}} = z_\mu = K^{-1}\sum_k \hat{z}^k_{t+H}$. Each response $u_k = \hat{z}^k_{t+H} - z_\mu$ is then the difference between a candidate's predicted final latent representation and their mean. The decision space is the span of these response vectors, $\mathcal{S} = \text{span}\{u_1, \ldots, u_K\}$. We apply principal component analysis (PCA) to the response covariance to identify its main directions of variation,

$$\Sigma_u = \frac{1}{K}\sum_{k=1}^{K} u_k u_k^\top = \sum_i \lambda_i v_i v_i^\top, \qquad \lambda_1 \geq \lambda_2 \geq \cdots \geq 0, \tag{6}$$

where v_i is a direction of variation and λ_i is its response energy. We report the percentage of total energy in the first m directions.

Ordering in a reduced decision space. We now ask how many response directions are needed to preserve the candidate-ordering performance of the full latent score. Define $p_{ki} = \langle u_k, v_i\rangle$ and $q_i = \langle q, v_i\rangle$. These are the coordinates of the response and the goal residual along direction v_i. Equation (5) can then be written as

$$Ds_k - \|q\|_2^2 = \sum_i \left(2q_i p_{ki} + p_{ki}^2\right). \tag{7}$$

Each direction contributes a goal-interaction term, $2q_i p_{ki}$, and a squared response component, p_{ki}^2. Since $\lambda_i = K^{-1}\sum_k p_{ki}^2$, PCA orders directions by response energy. Energy alone does not determine a direction's effect on candidate ordering: a low-energy direction can matter when $|q_i|$ is large, while a small $|q_i|$ reduces the interaction term even for a high-energy direction.

Using the first m PCA directions, we compute

$$s_k^{(m)} = \frac{1}{D}\sum_{i=1}^{m}\left(2q_i p_{ki} + p_{ki}^2\right). \tag{8}$$

The reduced score retains both terms for the selected directions.

Correlation with physical costs. Let J_k denote the physical cost of action sequence k, such as the final distance from a mobile robot to its goal or from a robot end effector to its target. In the settings considered here, lower costs correspond to better outcomes. During training, we obtain J_k from simulator outcomes or recorded task data. During planning, however, the final physical outcome is unknown because the candidate action sequence has not yet been executed. The planner therefore ranks candidates using s_k, which compares the final predicted latent with the goal latent.

A useful latent score should agree with the physical-cost ordering: if one action sequence has a lower physical cost than another, it should also receive a lower latent score. For each proposal set, we compute $\rho_s(s^{(m)}, J)$, the Spearman correlation between the reduced scores and physical costs. This measures how closely the reduced score follows the physical-cost ordering. Higher values indicate better agreement with physical-cost ordering. We compare it with $\rho_s(s, J)$ for the full latent score. Appendix Fig. 4 visualizes two candidate action sets with low and high agreement, respectively.

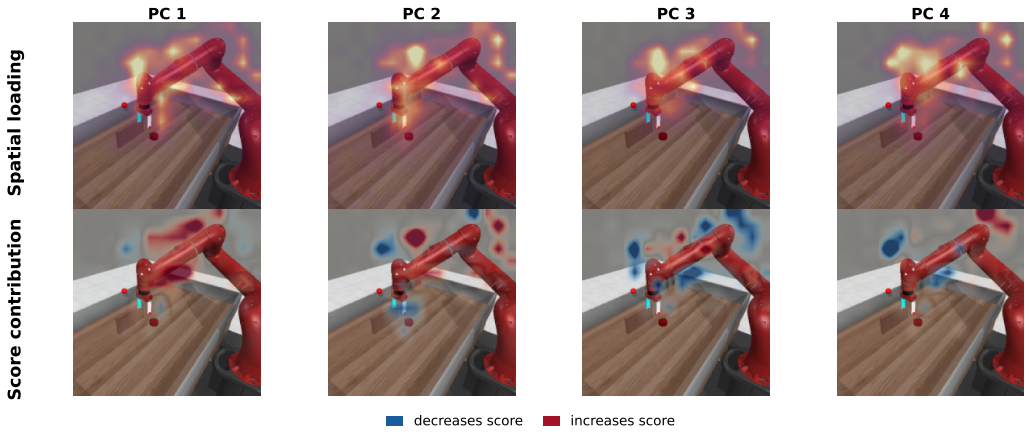

Figure 1: Leading candidate-response directions in JEPA-WM on MetaWorld Reach-Wall. Top: spatial loading. Bottom: signed contribution to the endpoint-to-goal latent score.

Throughout this analysis, the context, goal, horizon, and proposal distribution remain fixed while the action sequences vary. The response covariance measures variation across these actions, and correlation with J measures the relation between their latent scores and physical outcomes.

Empirical analysis. We analyze JEPA-WM latent rollouts on cross-wall navigation (Wall) and robot-arm control (MetaWorld Reach and Reach-Wall). These full-latent reconstruction models use a frozen DINOv2 encoder, with 16×16 vectors of dimension 384 per visual latent. We analyze the final visual latent and its mean squared Euclidean distance to the goal latent, excluding the proprioceptive distance term.

We use 96 Wall examples and 48 examples from each MetaWorld task. Each example contains an initial context and $K = 300$ candidate action sequences with planning horizon $H = 6$, corresponding to 30 simulator steps. Following JEPA-WM, we sample the candidates from the Gaussian CEM proposal distribution and execute every sequence from the same initial simulator state. The physical cost J_k is the final-state-to-goal distance in the plane for Wall and the end-effector-to-target distance for MetaWorld. We compute PCA separately for each context and compare the reduced-score correlation $\rho_s(s^{(m)}, J)$ with the full-score correlation $\rho_s(s, J)$, where s is the full visual latent score defined in Eq. (3).

Table 1 shows that sixteen PCA directions capture 90–92% of the response variation. The number of directions needed to recover the score–cost relation depends on the task. At $m = 32$, however, the correlation is within 0.006 of the full score on all three tasks. Thus, a small decision space preserves nearly all of the full latent score's correlation with physical costs for these proposal sets.

To examine where the leading candidate-response directions are expressed spatially, we apply PCA to the action-conditioned endpoint latents predicted by JEPA-WM on MetaWorld Reach-Wall and decompose each direction's score contribution across spatial locations. Figure 1 shows the four leading directions for one fixed context. The top row shows the magnitude of each direction at every spatial location. The bottom row shows each location's signed contribution to the endpoint-to-goal latent score. The leading directions emphasize regions around the robot arm, end effector, target,

Table 1: Candidate-response compression in reconstruction-trained full-latent world models. Energy is the percentage of response variation in the first m PCA directions. Spearman compares the complete reduced score with physical cost; Full uses all visual latent coordinates. Values are averaged over all examples.

	Captured response energy (%)			Spearman correlation with physical cost			
Dataset/task	$m = 8$	$m = 16$	$m = 32$	$m = 8$	$m = 16$	$m = 32$	Full
Wall	80.24	89.93	95.83	0.291	0.305	0.315	0.321
MetaWorld–Reach	83.66	91.47	96.01	0.730	0.747	0.750	0.753
MetaWorld–Reach-Wall	83.95	91.63	96.08	0.722	0.736	0.735	0.736

and intervening wall, where candidate actions produce distinct visual outcomes, rather than the background.

Appendix C.1 reports results for every retained dimension, dataset, and task considered in this work. These findings motivate the compact model of candidate-dependent dynamics introduced in Sec. 4.

4 METHOD

We introduce LDD-WM, a latent world model that represents candidate-dependent outcomes in a compact space while sharing visual prediction across action sequences. Section 3.2 shows that squared latent distance orders candidates through response–goal interaction and response energy. LDD-WM follows this decomposition using one shared visual rollout and an r-dimensional candidate state per sequence. Figure 2 summarizes the architecture and the physical-cost ordering supervision.

4.1 CANDIDATE-DEPENDENT SCORES

At rollout step h, let $\hat{z}_h^k$ denote the candidate-specific latent representation used in conventional full-latent planning. We express this state relative to a reference shared by all action sequences:

$$\hat{z}_h^k = z_h^{\mathrm{ref}} + u_h^k, \qquad q_h = z_h^{\mathrm{ref}} - z_g. \tag{9}$$

Applying Eq. (5) with normalization by D gives

$$s_h^k = \frac{1}{D}\|q_h\|_2^2 + 2I_h^k + R_h^k, \qquad I_h^k := \frac{1}{D}\langle q_h, u_h^k\rangle, \qquad R_h^k := \frac{1}{D}\|u_h^k\|_2^2. \tag{10}$$

The reference term $D^{-1}\|q_h\|_2^2$ is the same for every action sequence. Therefore, every pairwise comparison is determined by:

$$d_h^k := 2I_h^k + R_h^k, \qquad \operatorname{argsort}_k s_h^k = \operatorname{argsort}_k d_h^k. \tag{11}$$

We next describe how LDD-WM models the shared visual dynamics and the candidate-dependent interaction and response-energy terms.

4.2 SHARED VISUAL DYNAMICS

Let $\mathcal{W}_0 = \phi(o_{t-L+1:t})$ denote the initial window of L encoded context observations. At rollout step h, the reference predictor produces a latent z_h^{ref} and hidden visual tokens X_h, both shared across all candidate action sequences:

$$z_h^{\mathrm{ref}}, X_h = F_{\mathrm{ref}}(\mathcal{W}_{h-1}), \qquad \mathcal{W}_h = \mathrm{Tail}_L(\mathcal{W}_{h-1} \parallel z_h^{\mathrm{ref}}), \tag{12}$$

where $\parallel$ appends a latent and Tail_L denotes the most recent L latents. Each predicted reference is therefore used as context at the next step.

F_{ref} is an action-free deterministic autoregressive Transformer. Its architecture follows the predictor used in V-JEPA 2-AC and JEPA-WM, but it receives no candidate actions. We train F_{ref} with next-latent reconstruction and freeze it while training the compact candidate dynamics and score heads. It predicts how the scene is likely to evolve from the observed history, without conditioning on a candidate action. During planning, F_{ref} performs one prediction per rollout step, sharing its visual features across all candidate action sequences.

4.3 COMPACT CANDIDATE DYNAMICS

Motivated by the low-dimensional decision space identified in Sec. 3.2, we model the candidate-dependent variation in an r-dimensional representation rather than predicting a separate full latent

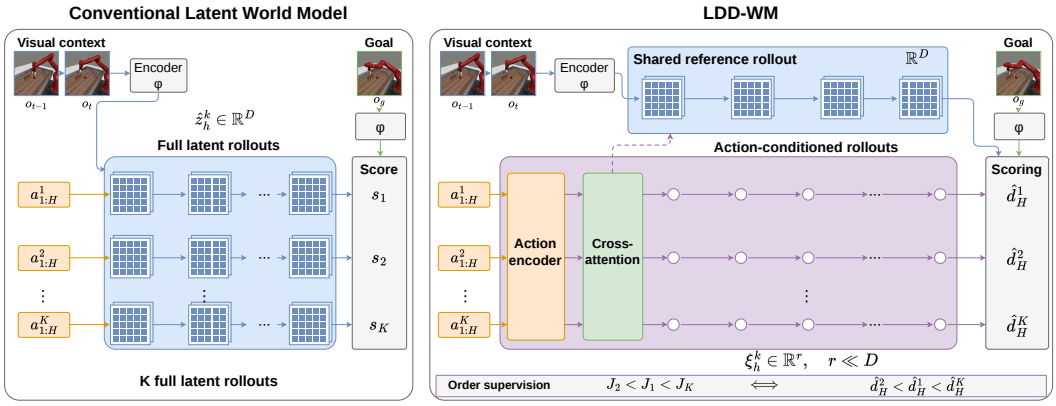

Figure 2: Conventional latent world-model planning and LDD-WM. A conventional model predicts a D-dimensional latent rollout for each of the K action sequences and compares the final latent representations with the goal latent (left). LDD-WM predicts one shared reference rollout and uses cross-attention to update an r-dimensional state for each action sequence, where $r \ll D$ (right). The model is trained to match the ordering given by physical costs, without a separate full-latent rollout for every action sequence.

trajectory for each action sequence. At rollout step h, the action encoder maps a_h^k to a query e_h^k. This query attends to the shared visual tokens X_h:

$$e_h^k = E_a(a_h^k), \tag{13}$$

$$\tilde{e}_h^k = \mathrm{LN}\big(e_h^k + \mathrm{CrossAttn}(e_h^k, X_h, X_h)\big)\,. \tag{14}$$

An MLP then produces an r-dimensional action-context coordinate:

$$\tilde{\eta}_h^k = \mathrm{MLP}_\eta\big([e_h^k, \tilde{e}_h^k]\big) \in \mathbb{R}^r. \tag{15}$$

Here, E_a and MLP_η are MLPs, $\mathrm{CrossAttn}$ is multi-head cross-attention with the action embedding as the query and visual tokens as keys and values, LN denotes layer normalization, and $[\cdot]$ denotes concatenation. The resulting representation therefore depends on the proposed action and shared visual context. We then center $\tilde{\eta}_h^k$ using a fixed set of trajectories. This gives action coordinates η_h^k whose origin does not depend on the other candidates in the proposal set. Appendix B.2 provides the full construction.

Each action sequence then updates an r-dimensional state:

$$\xi_0^k = 0, \qquad \xi_h^k = \xi_{h-1}^k + F_\xi([\xi_{h-1}^k, \eta_h^k]), \qquad \xi_h^k \in \mathbb{R}^r. \tag{16}$$

F_ξ is an MLP that updates the candidate state. The resulting ξ_h^k summarizes the first h actions of sequence k, while $z_{1:H}^{\mathrm{ref}}$ provides the shared visual trajectory. We use ξ_h^k to score the action sequence at rollout step h.

4.4 Scoring Action Sequences

The shared visual tokens define r directions in latent space:

$$\Psi_h = \mathrm{Flat}(B_\theta(X_h)) \in \mathbb{R}^{D\times r}, \tag{17}$$

where B_θ maps each token to r latent-channel vectors, and Flat arranges these outputs as r flattened latent directions. The matrix Ψ_h is computed once per rollout step and reused across all candidates. We normalize its columns as described in Appendix B.2. Two MLP heads map each candidate state to an r-dimensional scoring vector and a nonnegative scalar:

$$p_h^k = P_\theta(\xi_h^k) \in \mathbb{R}^r, \qquad n_h^k = E_\theta(\xi_h^k)^2 \geq 0. \tag{18}$$

The square ensures that n_h^k is nonnegative. Given a goal latent z_g, we map the reference–goal residual $q_h = z_h^{\mathrm{ref}} - z_g$ to an r-dimensional goal vector:

$$c_h = \frac{1}{D}\Psi_h^\top q_h. \tag{19}$$

Table 2: Visual planning on Wall, PointMaze, and MetaWorld. We report success in percentages as mean (sample standard deviation) over three random seeds. TFLOPs and latency are measured for a complete planning call, including visual encoding; latency is in seconds. Evaluation protocols, data matching, and computation are described in Appendix A.

	Wall			PointMaze			MetaWorld			
Configuration	TFLOPs ↓	Latency (s) ↓	Success (%) ↑	TFLOPs ↓	Latency (s) ↓	Success (%) ↑	TFLOPs ↓	Latency (s) ↓	Reach Success (%) ↑	Reach-Wall Success (%) ↑
DINO-WM	262.36	7.51	70.14 (2.2)	267.34	7.87	74.65 (2.6)	430.46	11.89	39.58 (9.6)	31.3 (10.4)
DDP-WM	34.75	1.58	73.61 (3.0)	34.75	1.58	75.69 (2.2)	62.84	2.81	40.97 (9.39)	29.86 (11.48)
JEPA-WM	143.89	9.49	80.9 (2.6)	143.89	9.49	83.0 (3.2)	227.79	14.86	49.3 (9.8)	29.9 (9.4)
JEPA-WM-CF	143.89	9.50	84.7 (2.4)	143.89	9.49	84.4 (2.8)	227.79	14.88	47.9 (9.5)	36.1 (11.5)
DIRECT-SCORE	24.35	0.89	85.4 (3.4)	24.35	0.89	80.2 (3.1)	60.53	2.21	21.5 (12.6)	28.5 (7.3)
LDD-WM	**1.03**	**0.14**	**92.7 (1.8)**	**1.03**	**0.14**	**86.8 (3.7)**	**1.52**	**0.21**	**60.4 (8.3)**	**64.6 (10.4)**

This vector is computed once per rollout step and shared across candidates. Motivated by Eq. (10), we define a goal-dependent interaction term and a nonnegative candidate term:

$$\widehat{I}_h^k = c_h^\top p_h^k, \qquad \widehat{R}_h^k = n_h^k. \tag{20}$$

These terms reproduce the form of the candidate-dependent score decomposition. Following Eq. (10), we combine these two learned terms to obtain the candidate score:

$$\widehat{d}_h^k = 2c_h^\top p_h^k + n_h^k. \tag{21}$$

The planner ranks action sequences in ascending order of their predicted scores. We omit the shared term $D^{-1}\|q_h\|_2^2$ because it does not affect their ordering. The resulting score combines shared visual and goal information with each candidate's compact state to rank action sequences.

4.5 LEARNING TO ORDER ACTION SEQUENCES

Planning depends on the relative ordering of candidate actions. For each context and goal, we sample a proposal set $\mathcal{A} = \{a_{1:H}^k\}_{k=1}^K$ and evaluate all sequences from the same initial state, giving the physical cost

$$J_k = J(a_{1:H}^k; o_{t-L+1:t}, o_g), \qquad k = 1, \dots, K. \tag{22}$$

Training candidates are sampled from the same proposal family used during planning. With the visual encoder and reference predictor frozen, we train the compact candidate dynamics, learned matrix Ψ_h, and score heads to order candidates by J_k. For notational clarity, we omit the rollout-step index h in the loss below. We standardize scores and costs within each proposal set:

$$\widetilde{d}_k = \frac{\widehat{d}_k - \mu_{\widehat{d}}}{\sigma_{\widehat{d}} + \epsilon}, \qquad \widetilde{J}_k = \frac{J_k - \mu_J}{\sigma_J + \epsilon}. \tag{23}$$

Here, μ and σ are computed over the K candidates, and ϵ prevents division by zero. We then apply a pairwise margin loss:

$$\mathcal{L}_{\text{ord}} = \frac{1}{|\mathcal{P}|} \sum_{(i,j)\in\mathcal{P}} \left[\widetilde{d}_i - \widetilde{d}_j + \gamma(\widetilde{J}_j - \widetilde{J}_i)\right]_+, \qquad \mathcal{P} = \{(i,j) : J_i < J_j\}. \tag{24}$$

Here, $[x]_+ = \max(x, 0)$ and γ controls the margin. The loss encourages lower-cost candidates to receive lower scores, with greater separation when their costs differ more. It directly supervises candidate ordering and requires no action-conditioned latent reconstruction target. Optimization and warmup details are provided in Appendix B.

5 EXPERIMENTS

We evaluate planning performance and inference cost in simulated control and real-image navigation, and then vary the decision dimension. Full experimental details are provided in Appendix A.

5.1 EXPERIMENTAL SETUP

Tasks and metrics. We evaluate cross-wall navigation on Wall (Zhou et al., 2025), maze navigation on PointMaze (Fu et al., 2020), and robot-arm control on MetaWorld Reach and Reach-Wall. Wall

Table 3: Goal-conditioned navigation using the NWM evaluation protocol. Results are reported as mean (sample standard deviation) over three random seeds. Success is the percentage of predictions with final position error at most 0.5 m. GNM and NoMaD trajectory errors are taken from NWM (Bar et al., 2025); success rates are unavailable in those reported results.

	ReCoN			SCAND			TartanDrive		
Planner	Success (%) ↑	ATE ↓	RPE ↓	Success (%) ↑	ATE ↓	RPE ↓	Success (%) ↑	ATE ↓	RPE ↓
GNM	–	1.870 (.000)	0.730 (.000)	–	2.120 (.000)	0.610 (.000)	–	6.650 (.000)	1.620 (.000)
NoMaD	–	1.950 (.050)	0.530 (.010)	–	2.240 (.030)	0.490 (.010)	–	6.320 (.030)	1.310 (.010)
NWM	66.7 (2.5)	1.179 (.005)	0.349 (.003)	43.0 (4.0)	1.241 (.014)	0.317 (.003)	0.0 (0.0)	5.752 (.076)	1.203 (.013)
Full-Latent WM	61.0 (1.7)	1.193 (.015)	0.367 (.007)	58.7 (2.1)	1.003 (.018)	0.280 (.002)	0.3 (0.6)	5.362 (.059)	1.123 (.011)
DIRECT-SCORE	64.0 (3.0)	1.158 (.024)	0.363 (.011)	55.0 (6.0)	1.023 (.067)	0.284 (.018)	0.3 (0.6)	5.624 (.293)	1.160 (.078)
LDD-WM	**72.0 (2.6)**	**1.129 (.011)**	**0.346 (.001)**	**64.7 (3.5)**	**0.928 (.024)**	**0.275 (.004)**	**2.0 (0.0)**	**4.571 (.113)**	**0.981 (.019)**

and PointMaze use open-loop planning, whereas MetaWorld uses closed-loop planning and replans during execution. We report task success over 96 held-out initial–goal pairs for each navigation task and 48 held-out episodes for each MetaWorld task. We also evaluate real-image navigation on ReCoN (Shah et al., 2022), SCAND (Karnan et al., 2022), and TartanDrive (Triest et al., 2022) using the NWM protocol (Bar et al., 2025). For each dataset, we report absolute trajectory error (ATE) and relative pose error (RPE) over 100 fixed contexts. We additionally report the percentage of predictions with final position error at most 0.5 m. Results are reported as the mean and sample standard deviation over three random seeds.

Planning and supervision. Following prior latent world-model planners (Zhou et al., 2025; Bar et al., 2025; Terver et al., 2026), we use CEM to sample candidate action sequences. Counterfactual training candidates are drawn from CEM proposal distributions. Simulator outcomes and recorded trajectories provide candidate-ordering supervision during training, but the models use neither simulator states nor rewards during planning. Task-specific horizons, CEM parameters, success criteria, and data-collection details are provided in Appendix A.2.

Models and baselines. On Wall, PointMaze and MetaWorld, we compare with DINO-WM (Zhou et al., 2025), DDP-WM (Yin et al., 2026), JEPA-WM (Terver et al., 2026), JEPA-WM-CF, and DIRECT-SCORE. DINO-WM and JEPA-WM predict a full latent rollout for each candidate, while DDP-WM uses localized dynamics and low-rank background updates. JEPA-WM-CF is our counterfactual variant of JEPA-WM: it reconstructs full candidate outcomes collected from the simulator and controls for the additional data used by LDD-WM.

DIRECT-SCORE removes the world-model rollout. Its candidate-specific Transformer takes frozen context and goal features together with the complete action sequence and predicts a scalar score. It uses the same counterfactual sets, ordering loss, and update budget as LDD-WM. For real-image navigation, we also construct Full-Latent WM with the same frozen visual encoder as LDD-WM. Its action-conditioned autoregressive Transformer follows the predictor architecture of JEPA-WM and is trained by latent reconstruction. It predicts a separate latent trajectory for each candidate and scores the final predicted latent by its squared distance to the goal latent. We also compare with NWM (Bar et al., 2025), GNM (Shah et al., 2023), and NoMaD (Sridhar et al., 2024).

Within each domain, all models use the same frozen visual encoder, candidate proposals, and evaluation protocol. We fix $r = 32$ for all simulation tasks (Wall, PointMaze and MetaWorld) and $r = 8$ for real-image navigation. Section 5.4 studies the effect of the decision dimension in LDD-WM. Architecture, training, and baseline details are provided in Appendix B.

5.2 VISUAL PLANNING IN SIMULATION

Results. Table 2 shows that LDD-WM achieves the highest mean success on Wall, PointMaze, MetaWorld Reach, and MetaWorld Reach-Wall. JEPA-WM-CF improves over JEPA-WM on three of the four tasks, showing the benefit of training on counterfactual outcomes. However, its reconstruction objective does not explicitly train the model to compare candidates, and it still predicts a separate full-latent rollout for every candidate at inference.

DIRECT-SCORE instead learns candidate ordering directly and avoids latent rollouts, substantially reducing inference cost. It performs comparably to JEPA-WM on the single-task Wall and PointMaze datasets, but its performance drops on the multi-task MetaWorld setting. This suggests that directly mapping the context, goal, and action sequence to a score may be insufficient when the candidate

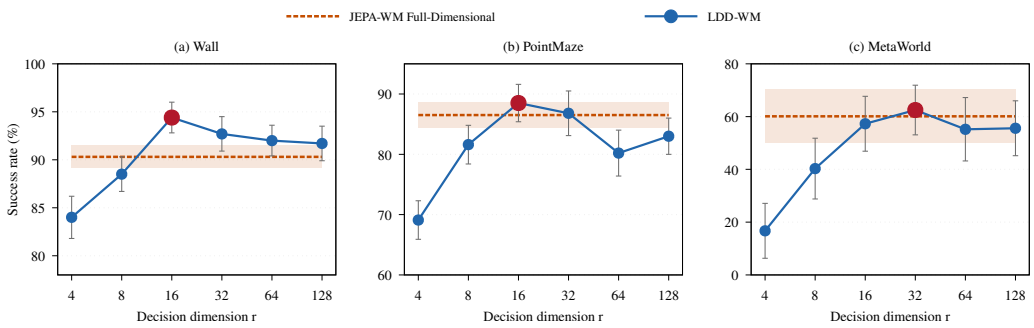

Figure 3: Effect of decision dimension at the final checkpoint. Blue points show LDD-WM; the orange dashed line shows the Full-Dimensional baseline. Error bars and shading indicate the sample standard deviation over three seeds. MetaWorld averages Reach and Reach-Wall success.

outcome depends on more varied visual dynamics. LDD-WM retains a shared visual rollout while modeling candidate-dependent dynamics in a compact representation. These results suggest that a compact candidate-dependent space can be learned to support action ordering in the evaluated tasks.

Table 2 also reports computation and latency for a complete planning call, including visual encoding. Relative to JEPA-WM, LDD-WM uses about 140–$150\times$ less computation and runs 68–$71\times$ faster. DIRECT-SCORE also avoids full-latent rollouts, but uses more computation than LDD-WM and has lower mean success on every task.

5.3 REAL-IMAGE NAVIGATION

Results. Table 3 shows that LDD-WM consistently improves over Full-Latent WM across all three real-image navigation datasets. On ReCoN, success increases from 61.0% to 72.0%. On TartanDrive, LDD-WM achieves substantially lower ATE and RPE than all baselines. As on MetaWorld, DIRECT-SCORE performs inconsistently when trained across multiple tasks: it improves some ReCoN metrics but underperforms Full-Latent WM on harder tasks SCAND and TartanDrive. Together, these results show that LDD-WM performs effectively on real-image navigation and support our hypothesis that decision-relevant outcomes admit a compact representation beyond synthetic visual environments.

5.4 EFFECT OF DECISION DIMENSION

We vary the dimension r of the candidate state on Wall, PointMaze, and MetaWorld while keeping the training protocol fixed for each domain. Figure 3 compares LDD-WM with JEPA-WM Full-Dimensional. This baseline uses the same candidate-specific full-latent architecture as JEPA-WM-CF, but is trained with the same ordering objective as LDD-WM instead of latent reconstruction. We evaluate the final checkpoint of every run and report mean success and sample standard deviation over three training seeds. MetaWorld success is averaged across Reach and Reach-Wall.

The best-performing dimension depends on the domain: $r = 16$ performs best on Wall and PointMaze, while $r = 32$ gives the highest MetaWorld average. At these dimensions, LDD-WM matches or exceeds JEPA-WM Full-Dimensional while retaining its inference-efficiency advantage. JEPA-WM Full-Dimensional changes only the JEPA-WM-CF training objective and therefore has the same TFLOPs and latency as JEPA-WM-CF. Increasing the dimension does not consistently improve success. These results suggest that separate full-latent rollouts for every candidate action are unnecessary and that a compact decision space can be sufficient for action selection.

6 CONCLUSION

We study the information that a latent world model must preserve to order candidate actions. Across simulation and real-image navigation, a compact proposal-relative decision space preserves much of the full latent score's correlation with physical costs. LDD-WM uses this observation to share visual prediction across candidates and model their differences with compact states trained from a physical-cost ordering objective. It matches or improves full-latent planning while reducing inference latency, and transfers across visual backbones. The best decision dimension depends on the task, but remains much smaller than the visual latent dimension. These results suggest that predicting the world can be high-dimensional even when the representation needed for planning is not.

AI USE STATEMENT

We used generative AI tools to assist with literature review, manuscript polishing, and to provide feedback on mathematical exposition. The authors reviewed the AI-assisted revisions and take responsibility for the final content of the paper.

REFERENCES

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, 2025.

Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15791–15801. IEEE, 2025.

Junha Chun, Youngjoon Jeong, and Taesup Kim. Sparse imagination for efficient visual world model planning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=faxcxKINBC.

Frederik Ebert, Chelsea Finn, Alex X. Lee, and Sergey Levine. Self-supervised visual planning with temporal skip connections. In *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pp. 344–356. PMLR, 2017.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2(3):440, 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019a.

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*, pp. 2555–2565. PMLR, 2019b.

Haresh Karnan, Anirudh Nair, Xuesu Xiao, Garrett Warnell, Sören Pirk, Alexander Toshev, Justin Hart, Joydeep Biswas, and Peter Stone. Socially compliant navigation dataset (scand): A large-scale dataset of demonstrations for social navigation. *IEEE Robotics and Automation Letters*, 7(4): 11807–11814, 2022.

Dongwon Kim, Gawon Seo, Jinsung Lee, Minsu Cho, and Suha Kwak. Planning in 8 tokens: A compact discrete tokenizer for latent world model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8183–8193, June 2026.

Lorenzo Mur-Labadia, Matthew Muckley, Amir Bar, Mido Assran, Koustuv Sinha, Mike Rabbat, Yann LeCun, and Nicolas Ballas. V-jepa 2.1: Unlocking dense features in video self-supervised learning. In *European Conference on Computer Vision*, pp. 671–689. Springer, 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. *arXiv preprint arXiv:2304.07193*, 2023.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635. PMLR, 2011.

Dhruv Shah, Benjamin Eysenbach, Nicholas Rhinehart, and Sergey Levine. Rapid exploration for open-world navigation with latent goal models. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 674–684. PMLR, 2022.

Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. Gnm: A general navigation model to drive any robot. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7226–7233. IEEE, 2023.

Ajay Sridhar, Dhruv Shah, Catherine Glossop, and Sergey Levine. Nomad: Goal masked diffusion policies for navigation and exploration. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 63–70. IEEE, 2024.

Basile Terver, Tsung-Yen Yang, Jean Ponce, Adrien Bardes, and Yann LeCun. What drives success in physical planning with joint-embedding predictive world models? *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856.

Samuel Triest, Matthew Sivaprakasam, Sean J Wang, Wenshan Wang, Aaron M Johnson, and Sebastian Scherer. Tartandrive: A large-scale dataset for learning off-road dynamics models. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 2546–2552. IEEE, 2022.

Shicheng Yin, Kaixuan Yin, Weixing Chen, Yang Liu, Guanbin Li, and Liang Lin. DDP-WM: Disentangled dynamics prediction for efficient world models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=gEc0iJCYxw.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pp. 1094–1100. PMLR, 2020.

Zhiwei Zhang, Hui Zhang, Kaihong Huang, Chenghao Shi, and Huimin Lu. Efficient image-goal navigation with representative latent world model. *arXiv preprint arXiv:2511.11011*, 2025.

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.