AGR-JEPA: ACTION-GROUNDED RESIDUAL JOINT-EMBEDDING PREDICTIVE ARCHITECTURE FOR PLANNING

Anonymous authors
Paper under double-blind review

ABSTRACT

In this research, we propose Action-Grounded Residual Joint-Embedding Predictive Architecture (AGR-JEPA), a reconstruction-free latent world model for planning. Our work is based on a key observation: existing approaches are typically trained to minimize next-embedding prediction error, whereas effective planning additionally requires the consequences of different actions to remain distinguishable in latent space. Through controlled experiments, we find that lower prediction error does not consistently correspond to higher planning success, indicating that latent predictability alone is not a reliable proxy for planning utility. We refer to this mismatch as the prediction–planning gap. To address this gap, AGR-JEPA adopts a residual parameterization that represents the future latent state as the combination of the current latent state and an action-conditioned latent update, focusing dynamics learning on action-induced state changes. Furthermore, AGR-JEPA introduces a control-grounding objective that uses action recoverability as a training signal, encouraging latent updates to retain information relevant to their conditioning actions and thereby improving the behavioral discriminability of action outcomes. This auxiliary objective is used only during training and incurs no additional computation at planning time. Across multiple 2D and 3D control tasks, AGR-JEPA achieves an average planning success rate of 92.06%, delivering consistent improvements over leading baselines across multiple tasks.

1 INTRODUCTION

Predicting the consequences of actions from high-dimensional observations is central to embodied planning and control. World models address this problem by learning action-conditioned environment dynamics, allowing an agent to evaluate candidate actions in an internal imagination space and thereby reduce reliance on real-environment interaction (Ha & Schmidhuber, 2018; Hafner et al., 2019; 2020; 2025). Directly modeling future observations in pixel space, however, is computationally expensive and requires capacity to reconstruct visual details, such as texture, illumination, and background, that are often irrelevant to decision making. Reconstruction-free world models instead encode observations into compact latent representations and predict future embeddings, offering an attractive route to learning action-conditioned dynamics from raw pixels (Assran et al., 2023; Bardes et al., 2024; Assran et al., 2025).

Recent JEPA-based reconstruction-free world models have demonstrated the potential of latent dynamics for planning. DINO-WM builds dynamics over pretrained visual representations (Zhou et al., 2025); PLDM jointly trains representations and predictors using multiple regularizers and an inverse-dynamics objective (Sobal et al., 2025); and LeWorldModel (LeWM) combines next-embedding prediction with SIGReg to enable stable end-to-end learning without reconstruction (Maes et al., 2026). These methods roll out action-conditioned latent trajectories and optimize candidate action sequences according to their predicted proximity to a goal embedding. However, predictive accuracy and planning utility impose fundamentally different requirements on latent dynamics. Next-embedding prediction measures the absolute accuracy of a future-state prediction, whereas planning requires the relative comparison and ranking of multiple action-conditioned outcomes. Since consecutive latent states can share substantial state-persistent information while action-induced changes occupy only a small fraction of the representation, a predictor may achieve

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

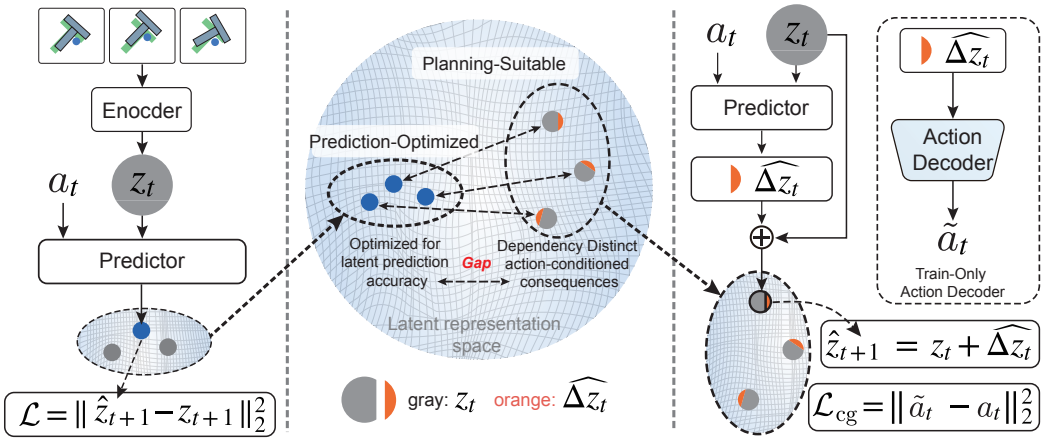

Figure 1: **Motivation of AGR-JEPA. (a) LeWM** directly predicts the next latent state by minimizing absolute next-latent prediction error. **(b) Prediction–planning gap.** Although this objective can improve individual latent prediction accuracy, it does not explicitly enforce the separation of latents, limiting reliable candidate-action comparison for planning. **(c) AGR-JEPA** parameterizes latent dynamics as an action-conditioned residual update, directing the dynamics model to focus on action-induced state changes. A training-only Action Decoder recovers the conditioning action from latent updates, encouraging action-discriminative latent transitions.

low prediction error by accurately modeling shared, action-invariant content while underrepresenting the subtle latent displacements that distinguish alternative actions.

We refer to this discrepancy between predictive accuracy and planning utility as the **prediction–planning gap** (Fig. 1). Our experiments provide direct evidence of this mismatch: lower latent prediction error does not consistently translate into higher planning success (Sec. 4.4), indicating that prediction accuracy alone is an insufficient proxy for planning utility. This gap motivates a complementary requirement for planning-oriented latent dynamics: beyond accurately predicting future states, they should preserve *behavioral discriminability*—the relative differences among actions enable reliable action comparison and ranking.

To address this gap, we propose Action-Grounded Residual Joint-Embedding Predictive Architecture (AGR-JEPA). AGR-JEPA adopts a residual parameterization that represents the next latent representation as the sum of the current latent state and an action-conditioned latent update. Rather than directly predicting the complete future representation, this design introduces an explicit identity path that preserves the current state and enables the dynamics predictor to focus on action-induced state changes. However, accurately modeling these updates alone does not guarantee that they retain sufficient action-relevant information. To this end, AGR-JEPA introduces a control-grounding objective, using action recoverability as a training signal for behavioral discriminability. Specifically, a shared lightweight Action Decoder recovers the conditioning action from a latent displacement. This supervision encourages the predictor to learn more discriminative action-conditioned transitions. The decoder is used only during training and does not participate in candidate action generation, latent rollouts, or planning-cost evaluation; thus, it introduces no additional planning-time computation. Across four continuous-control environments under a unified evaluation protocol, AGR-JEPA achieves an average planning success rate of 92.06%, delivering consistent improvements over leading baselines across multiple tasks. Ablations further show that residual parameterization and control grounding play complementary roles in modeling action-induced changes and discriminating action-conditioned transitions. Counterfactual consequence-identification analyses additionally demonstrate that control grounding improves the ability of latent rollouts to distinguish the consequences of competing action sequences. Our contributions are as follows:

- We identify and empirically characterize the *prediction–planning gap* in reconstruction-free latent world models, showing that lower next-embedding prediction error does not necessarily imply better downstream planning.

- We propose AGR-JEPA, which combines action-conditioned residual latent dynamics with an action-recoverability-based control-grounding objective, directing dynamics learning toward action-induced changes while encouraging predicted latent updates to preserve behaviorally discriminative information.
- Across diverse 2D and 3D continuous-control tasks, AGR-JEPA achieves strong and consistent planning performance, while ablations and counterfactual analyses further show that the predicted latent outcomes of different actions are more distinguishable.

2 Related Work

2.1 Joint-Embedding Predictive Architectures for Planning

World models support planning by learning action-conditioned dynamics and evaluating imagined futures before execution (Ha & Schmidhuber, 2018). Existing approaches include reconstruction-based latent world models with reward- or value-related objectives, such as PlaNet and Dreamer (Hafner et al., 2019; 2020; 2025); task-oriented latent-control models such as TD-MPC (Hansen et al., 2022; 2024); and world-action models that jointly model future evolution and action generation (Wu et al., 2023; Cheang et al., 2024; Guo et al., 2024; Cen et al., 2025; Ye et al., 2026; Wang et al., 2026). While effective, these lines introduce observation reconstruction, task-specific reward or value objectives, or action generation beyond latent consequence prediction. Reconstruction-free, reward-free planning therefore provides a focused setting for studying whether learned latent dynamics alone can support decision making.

Joint-Embedding Predictive Architectures (JEPAs) provide a reconstruction-free alternative by predicting future representations rather than future pixels (Assran et al., 2023; Bardes et al., 2024; Assran et al., 2025). For visual planning, DINO-WM learns action-conditioned dynamics over frozen pretrained DINOv2 features (Zhou et al., 2025); PLDM jointly trains representations and predictors using VICReg-style regularization, temporal constraints, and inverse dynamics (Sobal et al., 2025); and LeWorldModel (LeWM) simplifies end-to-end learning to next-embedding prediction and SIGReg (Maes et al., 2026). Sub-JEPA improves training stability through subspace Gaussian regularization (Zhao et al., 2026), while Demo-JEPA extends reconstruction-free latent planning to cross-embodiment imitation from visual demonstrations (He et al., 2026). Despite these advances, existing JEPA-based world models primarily optimize future-embedding accuracy, representation diversity, or collapse prevention. These objectives align individual predictions with observed futures but do not explicitly require the predicted consequences of competing actions to remain distinguishable, even though planning depends on their relative comparison and ranking. Because competing rollouts often share substantial state-persistent content, action-induced differences may occupy only a small part of the representation and can be underemphasized by absolute prediction objectives. This motivates latent dynamics that preserve action-relevant differences across candidate rollouts.

2.2 Action-Sensitive Latent Dynamics

Action-sensitive latent-dynamics methods introduce inverse-dynamics or action-recovery supervision to preserve control-relevant transition information. PLDM recovers actions from concatenated adjacent representations $[z_t, z_{t+1}]$ as part of its broader training objective (Sobal et al., 2025). Delta-JEPA instead recovers actions from encoded latent displacements, $\Delta z_t = z_{t+1} - z_t$, encouraging collapse-resistant and action-sensitive transition geometry while retaining next-embedding prediction (Zhang et al., 2026).

AGR-JEPA shares Delta-JEPA's use of action recovery from encoded latent displacements but couples this supervision directly to the predictive dynamics. It parameterizes the forward dynamics as residual updates and grounds both target and predictor-generated displacements; the latter directly constrains the updates recursively accumulated during planning. AGR-JEPA further retains SIGReg to separate global anti-collapse regularization from transition-level action grounding, and examines planning utility through the prediction–planning gap and the counterfactual discriminability of imagined rollouts. Thus, AGR-JEPA differs not in its use of latent differences alone, but in coupling residual prediction with direct grounding of the imagined transitions used for planning.

3 METHOD

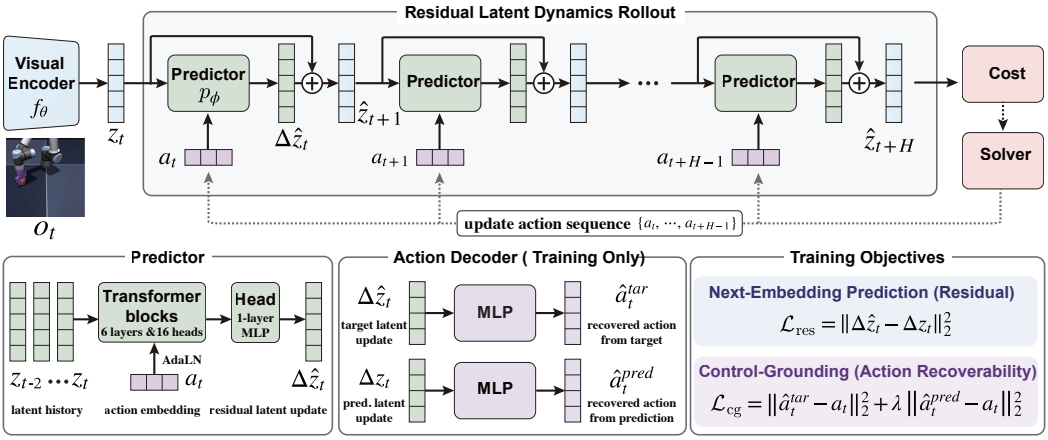

Figure 2: **Overview of AGR-JEPA.** Given an initial observation O_t, the visual encoder f_θ produces latent representations z_t, and the action-conditioned predictor p_ϕ recursively rolls out residual latent updates $\Delta\hat{z}_t$ under a candidate action sequence. The terminal latent goal-matching cost guides the solver to optimize the action sequence. During training, the residual loss $\mathcal{L}_{\text{res}}$ aligns predicted and target latent updates, while the control-grounding loss $\mathcal{L}_{\text{cg}}$ trains a shared Action Decoder to recover actions from both target and predicted updates, promoting behaviorally discriminative rollouts.

In this section, we present the design of Action-Grounded Residual Joint-Embedding Predictive Architecture (AGR-JEPA), as illustrated in Figure 2.

3.1 PRELIMINARIES: LEARNING A LATENT WORLD MODEL

We consider a fully offline and reward-free dataset $\mathcal{D} = \{\tau^{(i)}\}_{i=1}^{M}$, where each trajectory $\tau^{(i)} = \{o_1, a_1, \ldots, a_{T-1}, o_T\}$ consists of visual observations $o_t \in \mathbb{R}^{C\times H\times W}$ and actions executed between consecutive observations. Given an observation history sampled from offline trajectories, a visual encoder f_θ first maps each observation to a compact latent representation, $z_t = f_\theta(o_t) \in \mathbb{R}^{d_z}$. Following the single-step dynamics notation of LeWorldModel Maes et al. (2026), we write the predictor output at time step t as $\hat{z}_{t+1}^{\text{abs}} = p_\phi(z_t, a_t)$. During training, the predicted representation is aligned with the target latent encoded from the future observation, while representation regularization is used to prevent latent collapse. Following existing reconstruction-free latent world models, the objective is written as

$$\mathcal{L}_{\text{LeWM}} = \left\|\hat{z}_{t+1}^{\text{abs}} - z_{t+1}\right\|_2^2 + \lambda_{\text{sig}}\mathcal{L}_{\text{SIGReg}}, \tag{1}$$

where $\mathcal{L}_{\text{SIGReg}}$ Maes et al. (2026) encourages the latent distribution to approximate an isotropic Gaussian, thereby maintaining a diverse and non-collapsed latent space. Through this training process, the encoder learns to map visual states into a representation space suitable for prediction, while the predictor learns action-driven latent transitions. Although equation 1 can learn predictive reconstruction-free latent dynamics, its core supervision remains the absolute next-latent prediction error. This objective constrains each predicted state to be close to the ground-truth future state, but does not explicitly require predicted transitions induced by different actions to be sufficiently distinguishable. For planning, which relies on comparing and ranking candidate action sequences, their predicted latent outcomes must remain sufficiently distinguishable. AGR-JEPA is designed to explicitly model this property.

3.2 ACTION-GROUNDED RESIDUAL DYNAMICS

Residual Latent Dynamics. Rather than directly predicting the complete next latent representation, AGR-JEPA models state transitions as action-conditioned residual updates. Using the same single-step notation, the predicted update and the ground-truth latent displacement are defined as

$$\widehat{\Delta z}_t = p_\phi(z_t, a_t), \qquad \Delta z_t = z_{t+1} - z_t. \tag{2}$$

The next representation is obtained by adding the predicted update: $\hat{z}_{t+1} = z_t + \widehat{\Delta z}_t$. To supervise the residual update, the predicted displacement is aligned with the ground-truth displacement:

$$\mathcal{L}_{\text{res}} = \left\| \widehat{\Delta z}_t - \Delta z_t \right\|_2^2. \tag{3}$$

This parameterization preserves information about the current state through an identity path, allowing the predictor to focus on action-induced state changes rather than regenerating the complete future representation. However, accurately predicting latent displacements does not necessarily ensure that they retain sufficient information to distinguish different actions. We therefore introduce a control-grounding objective over both encoded target updates and predictor-generated updates.

Control Grounding. Let $D_\psi : \mathbb{R}^{d_z} \to \mathbb{R}^{F d_a}$ denote a shared Action Decoder, where F is the frame skip and d_a is the dimension of the original action. We use action recoverability as a training proxy for behavioral discriminability: a latent update should retain sufficient information to recover the action that produced it. The Action Decoder is applied to both the target displacement and the predicted displacement:

$$\begin{aligned} \hat{a}_t^{\text{tar}} &= D_\psi(\Delta z_t), \qquad \mathcal{L}_{\text{cg,tar}} = \left\| \hat{a}_t^{\text{tar}} - a_t \right\|_2^2, \\ \hat{a}_t^{\text{pred}} &= D_\psi(\widehat{\Delta z}_t), \quad \mathcal{L}_{\text{cg,pred}} = \left\| \hat{a}_t^{\text{pred}} - a_t \right\|_2^2. \end{aligned} \tag{4}$$

The target branch requires actions to be recoverable from the ground-truth displacement Δz_t, encouraging the encoder to learn latent transitions that contain action-relevant information. However, this constraint alone does not guarantee that the dynamics predictor preserves the same action-dependent structure in its outputs. We therefore additionally recover actions from the predicted displacement $\widehat{\Delta z}_t$. Since these predicted updates are recursively accumulated during planning, this branch more directly encourages different candidate actions to produce distinguishable imagined transitions. The two supervision terms are combined as $\mathcal{L}_{\text{cg}} = \mathcal{L}_{\text{cg,tar}} + \lambda_{\text{pred}} \mathcal{L}_{\text{cg,pred}}$. Both branches share the same decoder parameters. The complete AGR-JEPA training objective is

$$\mathcal{L}_{\text{AGR}} = \mathcal{L}_{\text{res}} + \lambda_{\text{sig}} \mathcal{L}_{\text{SIGReg}} + \lambda_{\text{cg}} \mathcal{L}_{\text{cg}}. \tag{5}$$

The three objectives play complementary roles: $\mathcal{L}_{\text{res}}$ learns predictive latent transitions, $\mathcal{L}_{\text{SIGReg}}$ maintains globally diverse and non-collapsed state representations, and $\mathcal{L}_{\text{cg}}$ preserves action-relevant information in latent changes. AGR-JEPA remains reconstruction-free throughout and does not introduce a pixel decoder.

3.3 Latent Planning

Given the current observation history and a goal observation o_g, the visual encoder first produces the corresponding latent representations, with the goal representation given by $z_g = f_\theta(o_g)$. For a candidate action sequence $A_{t:t+H-1} = \{a_t, \ldots, a_{t+H-1}\}$, AGR-JEPA recursively predicts future latent states from the encoded latent history in an autoregressive manner. Consistent with the single-step dynamics formulation, at each step the action-conditioned residual produced by the predictor is added to the current latent state:

$$\hat{z}_{t+k+1} = \hat{z}_{t+k} + p_\phi(\hat{z}_{t+k}, a_{t+k}), \quad k = 0, \ldots, H-1. \tag{6}$$

The rollout is initialized using the encoded ground-truth observation history. At subsequent steps, predicted latent states are autoregressively fed back into the predictor.

Each candidate action sequence is evaluated using a terminal latent goal-matching cost:

$$J(A_{t:t+H-1}) = \left\| \hat{z}_{t+H} - z_g \right\|_2^2. \tag{7}$$

The objective of planning is therefore to find the candidate action sequence that minimizes $J(A_{t:t+H-1})$. We approximately solve this optimization problem using the Cross-Entropy Method (CEM). At each iteration, CEM samples candidate action sequences from the current sampling distribution, computes their rollouts and costs using equation 6, and updates the sampling distribution using the elite candidates with the lowest costs. During execution, we use receding-horizon model-predictive control: only the first action of the current best sequence is executed, after which planning

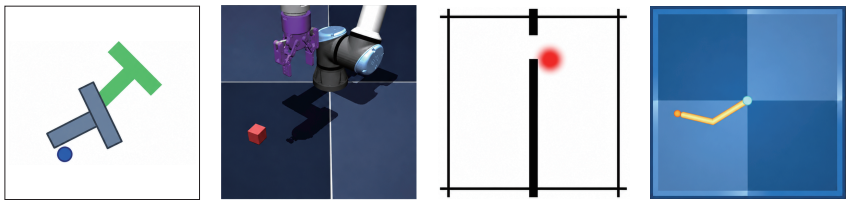

Figure 3: **Evaluation environments.** From left to right: Push-T, a contact-rich 2D task that pushes a T-shaped object to a target pose; OGB-Cube, a 3D robotic manipulation task that moves a cube to a target location; Two-Room, a 2D navigation task requiring an agent to pass through a narrow opening to reach a target; and Reacher, a planar two-link arm reaching a target configuration. Further environment and dataset details are provided in the appendix A.

is repeated upon receiving a new observation. The Action Decoder D_ψ provides control-grounding supervision only during training and is not used in latent rollouts or planning-cost computation. Thus, control grounding changes the learned latent transitions without introducing additional computation at planning time.

4 Experiments

Our experiments combine multi-task benchmarking, controlled ablations, and diagnostic analyses to assess both the planning performance of AGR-JEPA and the properties of its learned latent transitions. Beyond comparing planning success rates, the experiments isolate the contributions of each design choice and examine whether action-grounded latent dynamics provide more discriminative counterfactual consequences for planning. Specifically, we evaluate: (1) the planning performance of AGR-JEPA across navigation and manipulation tasks (Section 4.2); (2) the individual and combined effects of residual parameterization and control grounding (Section 4.3); (3) the complementarity of grounding signals from target displacements and predictor-generated displacements (Section 4.3); (4) whether next-embedding prediction error reliably reflects downstream planning utility (Section 4.4); and (5) whether action grounding improves the model's ability to distinguish the counterfactual consequences associated with different candidate actions (Section 4.5).

4.1 Experimental Setup

Evaluation environments and protocol . Following LeWorldModel (LeWM) (Maes et al., 2026), we evaluate AGR-JEPA on four continuous-control environments: Two-Room, Reacher, Push-T, and OGB-Cube (Fig. 3). Together, they cover two-dimensional navigation, planar reaching, contact-rich manipulation, and three-dimensional robotic manipulation. Further details on the environments are provided in the appendix A. We report planning success over 500 test trajectories per environment, averaged across three seeds (mean $\pm$ standard deviation).

Baselines. We compare AGR-JEPA with the following reconstruction-free or JEPA-based world models. PLDM Sobal et al. (2025) is trained end-to-end with VICReg-style regularization, temporal constraints, and inverse dynamics. LeWM Maes et al. (2026), the direct basis of AGR-JEPA, uses next-embedding prediction and SIGReg. Sub-JEPA Zhao et al. (2026) improves training stability through subspace Gaussian regularization. Delta-JEPA Zhang et al. (2026) learns action-sensitive representations through latent-difference action decoding. All retrainable methods use their official configurations and are compared under the same data, training budget, and planning protocol.

Implementation details. AGR-JEPA adopts the visual encoder and dynamics predictor of LeWM. The visual encoder is a ViT-Tiny trained from scratch with a latent dimension of 192. The dynamics predictor is a six-layer causal Transformer with action conditioning injected through AdaLN. The control-grounding decoder is a two-layer MLP with architecture $192 \to 256 \to Fd_a$, where F denotes the frame skip and d_a is the original action dimension. Unless otherwise specified, we set $\lambda_{\text{sig}} = 0.09$, $\lambda_{\text{cg}} = 1.0$, and $\lambda_{\text{res}} = 1.0$. Further details are provided in the appendix C.

Table 1: Planning success rate (%, higher is better) on four continuous-control environments. Results are reported as mean ± standard deviation over three random seeds. The average is computed over the four environments.

Method	Two-Room	Reacher	Push-T	OGB-Cube	Average
PLDM Sobal et al. (2025)	93.73 ± 1.03	64.33 ± 2.14	76.13 ± 1.70	57.27 ± 1.53	72.87
Sub-JEPA Zhao et al. (2026)	90.60 ± 0.53	81.00 ± 2.40	63.73 ± 0.12	62.67 ± 1.45	74.50
Delta-JEPA Zhang et al. (2026)	$\mathbf{100.00 \pm 0.00}$	81.33 ± 0.50	89.07 ± 1.90	79.27 ± 1.81	87.42
LeWM Maes et al. (2026)	74.93 ± 0.42	79.87 ± 0.90	84.53 ± 1.50	64.13 ± 1.89	75.87
AGR-JEPA (Ours)	98.13 ± 0.52	$\mathbf{84.07 \pm 1.31}$	$\mathbf{91.20 \pm 0.71}$	$\mathbf{94.83 \pm 0.35}$	**92.06**

4.2 Multi-task Planning Performance

Table 1 compares the planning success rates of AGR-JEPA and existing latent world models across four continuous-control environments. All results are reported as mean ± standard deviation over three random seeds. AGR-JEPA achieves the highest average planning success rate, 92.06%, across the four tasks. This result improves upon its direct basis, LeWM (75.87%), by 16.19 percentage points, and exceeds the strongest baseline on average, Delta-JEPA (87.42%), by 4.64 points. Compared with LeWM, AGR-JEPA improves success rates by 23.20, 4.20, 6.67, and 30.70 points on Two-Room, Reacher, Push-T, and OGB-Cube, respectively. On OGB-Cube, it outperforms LeWM and Delta-JEPA by 30.70 and 15.56 points, respectively, indicating that the learned action-conditioned residual transitions provide reliable candidate-action evaluation even in visually more complex three-dimensional manipulation tasks.

4.3 Module Ablation

Table 2: Module ablation on Push-T. All variants share the same visual encoder, dynamics predictor, data, and planning protocol.

Method	Transition	Control Grounding	SR (%) ↑
Baseline	Absolute	–	84.53 ± 1.50
Residual Only	Residual	–	78.90 ± 2.88
Control Grounding	Absolute	✓	89.26 ± 1.09
AGR-JEPA	Residual	✓	$\mathbf{91.20 \pm 0.71}$

Table 3: Ablation of control-grounding signals on Push-T. All variants use residual latent dynamics.

Method	Target Δz_t	Predicted $\widehat{\Delta z_t}$	SR (%) ↑
Residual Only	–	–	78.90 ± 2.88
Target Only	✓	–	90.70 ± 2.01
Predicted Only	–	✓	86.20 ± 2.78
Both	✓	✓	$\mathbf{91.20 \pm 0.71}$

Residual Dynamics and Control-Grounding Ablation. Table 2 isolates the effects of residual parameterization and control grounding. Residual dynamics alone achieves 78.90% success, 5.63 points below the LeWM baseline, showing that simply reparameterizing the predictor output as a latent displacement is not by itself sufficient to improve planning performance. Adding control grounding to absolute dynamics improves success from 84.53% to 89.26%, indicating that action recoverability provides useful supervision beyond next-latent prediction. Combining residual dynamics with control grounding achieves the highest success rate of 91.20%. Together, these results show that residual parameterization alone is insufficient, whereas coupling it with control grounding yields the strongest planning performance.

Action Control-Grounding Signal Ablation. Control grounding can be applied to the encoded target displacement Δz_t, the predictor-generated displacement $\widehat{\Delta z_t}$, or both. To analyze the effect of different supervision sources, we fix the residual dynamics and vary only the input branch of the shared Action Decoder. As shown in Table 3, relative to Residual Only ($78.90 \pm 2.8\%$), applying action recovery solely to the target displacement achieves $90.70 \pm 2.01\%$, while constraining only the predictor-generated displacement yields $86.20 \pm 2.78\%$. Combining the two branches further improves the success rate to $91.20 \pm 0.71\%$, exceeding Target Only and Predicted Only by 0.50 and 5.00 percentage points, respectively. These results indicate that the target branch provides the primary supervision for learning action-discriminative transition geometry, whereas the predicted branch directly constrains the recursively propagated latent updates and yields additional planning gains.

Table 4: Raw CCIA across candidate-set sizes using saved $K = 16$ candidate sets. Values are mean $\pm$ sample standard deviation across three evaluation seeds.

Candidate-set sizes 2–8				**Candidate-set sizes** 10–16			
k	Random	LeWM	AGR-JEPA	k	Random	LeWM	AGR-JEPA
2	50.00%	89.49 ± 0.78	92.38 ± 0.67	10	10.00%	64.68 ± 0.85	72.30 ± 1.60
4	25.00%	78.90 ± 0.91	83.93 ± 1.26	12	8.33%	61.84 ± 0.94	69.95 ± 1.65
6	16.67%	72.63 ± 0.87	78.83 ± 1.47	14	7.14%	59.43 ± 1.08	67.95 ± 1.69
8	12.50%	68.16 ± 0.83	75.17 ± 1.55	16	6.25%	57.33 ± 1.22	66.21 ± 1.75

4.4 Prediction–Planning Gap

Our investigation was motivated by an unexpected observation during LeWorldModel (LeWM) development: replacing its absolute next-latent predictor with residual dynamics reduced latent prediction error on Push-T, but did not improve planning success and instead decreased it on average. We therefore conduct a paired evaluation between an Absolute model that directly predicts the next latent state and a Residual model that predicts a latent update. Both models are evaluated with ten fixed random seeds; for each seed, closed-loop planning is performed on 50 independently sampled initial–goal pairs. Figure 4 summarizes the seed-level distributions of teacher-forced latent RMSE and planning success; small markers denote individual runs, and outlined markers denote the corresponding means.

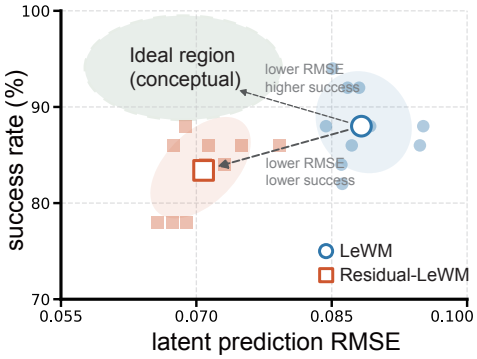

Figure 4: **Prediction–planning gap on Push-T.** Each small marker denotes one of ten fixed-seed runs; outlined markers and shaded ellipses indicate the mean and one standard deviation, respectively. Residual-LeWM achieves lower teacher-forced latent RMSE but lower mean planning success than LeWM, moving away from the conceptual ideal of lower RMSE and higher success.

The Residual model attains lower average RMSE than the Absolute model (0.0708 versus 0.0883), yet lower average success (83.4% versus 88.0%). Seed-level RMSE and success are also weakly related for the Absolute model ($r = -0.121$) and positively correlated for the Residual model ($r = 0.519$). We refer to this mismatch as the *prediction–planning gap*: prediction error measures individual transition accuracy, whereas planning additionally requires behaviorally discriminative rollouts for candidate-action ranking.

4.5 Counterfactual Consequence Discrimination

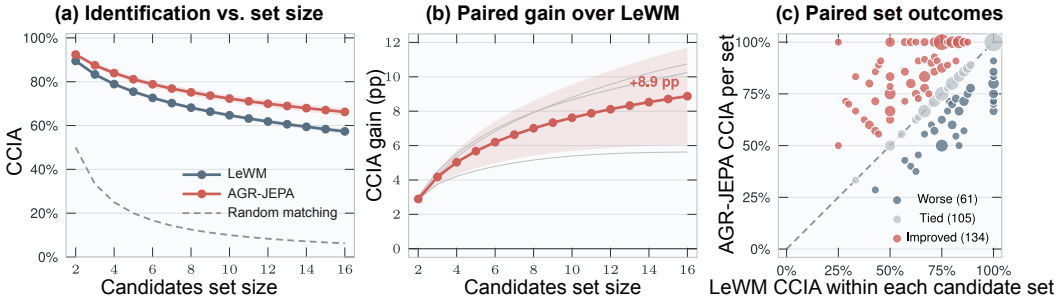

Figure 5: Counterfactual consequence discrimination on Push-T. For a common initial history, the model must match each predicted consequence to its corresponding ground-truth consequence among K counterfactual action branches. AGR-JEPA improves identification accuracy, with a larger advantage as the candidate set grows.

Table 5: Comparison of latent prediction error, counterfactual consequence identification, and planning success rate. Lower Latent RMSE is better, whereas higher CCIA and SR are better.

Metric	LeWM	AGR-JEPA
RMSE ($\times 10^{-2}$) $\downarrow$	8.828 ± 0.378	9.535 ± 0.401
CCIA@16 $\uparrow$	57.33 ± 1.22	66.21 ± 1.75
SR $\uparrow$	84.53 ± 1.50	91.20 ± 0.71

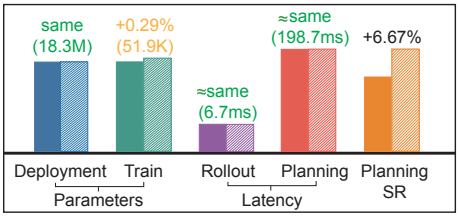

Figure 6: **Training and deployment efficiency of AGR-JEPA.**

Planning requires not only predicting a plausible future, but also distinguishing the consequences of competing action sequences. To evaluate this capability, a counterfactual consequence identification protocol is constructed. We first sample N initial histories from the training set and, for each corresponding initial state, construct and execute K candidate action sequences to obtain K ground-truth terminal observations and their latent embeddings. Each candidate sequence is also rolled out by the learned dynamics model, and its predicted terminal latent is matched against the K ground-truth terminal embeddings. A prediction is counted as correct only if its nearest ground-truth endpoint is produced by the same action sequence. We define the average matching accuracy as Counterfactual Consequence Identification Accuracy (CCIA), for which random guessing achieves $1/K$. As shown in Fig. 5, the CCIA of both models decreases as the candidate set size K increases. However, AGR-JEPA consistently outperforms LeWM, with its advantage increasing from 2.9 percentage points at $K = 2$ to 8.9 percentage points at $K = 16$. At $K = 16$, AGR-JEPA achieves a CCIA of 66.2%, compared with 57.3% for LeWM; both results remain substantially above the 6.25% chance level. These results show that action recovery on target displacements enables latent dynamics to more reliably distinguish the counterfactual consequences of different actions, thereby providing more discriminative predictions for candidate ranking during planning. Additional details of the evaluation protocol and results from larger-scale experiments are provided in Appendix B.

Table 5 further relates latent prediction accuracy and counterfactual discriminability to downstream planning performance. From LeWM to AGR-JEPA, latent RMSE increases from 8.828 to 9.535 despite a planning-success gain from 84.53% to 91.20%, whereas CCIA@16 rises alongside success, from 57.33% to 66.21%. This comparison suggests that absolute latent prediction accuracy alone is insufficient for assessing planning utility.

4.6 Training and Deployment Efficiency

This section analyzes the training- and deployment-time efficiency of AGR-JEPA. To impose control grounding, AGR-JEPA introduces a lightweight 51,978-parameter Action Decoder during training, increasing the total number of trainable parameters by only 0.29%. The decoder is discarded before deployment and is not evaluated during latent rollout, candidate generation, or CEM cost computation. Consequently, AGR-JEPA retains the same 18.03M deployment parameters as LeWM, with negligible changes in latent-rollout and CEM-planning latency (Fig. 6).

5 Conclusion

We introduced AGR-JEPA, a reconstruction-free latent world model that addresses the prediction–planning gap by combining action-conditioned residual dynamics with control grounding. Rather than optimizing next-latent prediction accuracy alone, AGR-JEPA encourages predicted latent updates to preserve the distinctions among action outcomes needed for planning. Across four continuous-control tasks, AGR-JEPA achieves strong and consistent planning performance, while ablations and counterfactual analyses show that it better distinguishes the predicted outcomes of different actions. These results suggest that planning-oriented latent world models should optimize not only predictive accuracy, but also the discriminability of action-induced latent changes.

AI use statement

We used OpenAI's ChatGPT/Codex to assist with Chinese-to-English translation and software implementation. We additionally used these tools to edit portions of the manuscript for clarity and

concision, locate and summarize relevant literature, format LaTeX and references, and check the numerical and cross-reference consistency of the manuscript. The research method and experimental protocol were determined by the authors. Generative AI was not used to generate, select, or process experimental data, and all reported measurements were obtained from experiments conducted by the authors. All AI-assisted text and code were manually reviewed, revised, and verified by the authors. Citations were checked against the original sources, and AI-assisted code was validated through author-run experiments and tests. The authors take full responsibility for the final manuscript, code, claims, and results.

Reproducibility statement

To support reproducibility, Section 4 specifies the evaluation environments, baselines, metrics, planning protocol, and random-seed setting. Appendix A details the environments and evaluation protocol; Appendix C documents the model architecture, training objective, optimization hyperparameters, CEM settings, and controls used for fair comparison; Appendix D provides the sequence-based formulations for latent-dynamics training and autoregressive planning; and Appendix B describes the counterfactual consequence identification protocol, filtering criteria, and large-scale results. Together, these details are intended to support independent reimplementation and verification. The AGR-JEPA implementation and experiment configurations are undergoing internal review and will be released publicly once that review is complete.

References

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15619–15629, 2023. URL https://openaccess.thecvf.com/content/CVPR2023/html/Assran_Self-Supervised_Learning_From_Images_With_a_Joint-Embedding_Predictive_Architecture_CVPR_2023_paper.html.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, et al. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025. doi: 10.48550/arXiv.2506.09985. URL https://arxiv.org/abs/2506.09985.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *arXiv preprint arXiv:2404.08471*, 2024. doi: 10.48550/arXiv.2404.08471. URL https://arxiv.org/abs/2404.08471.

Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, Deli Zhao, and Hao Chen. WorldVLA: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025. doi: 10.48550/arXiv.2506.21539. URL https://arxiv.org/abs/2506.21539.

Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, Hanbo Zhang, and Minzhao Zhu. GR-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024. doi: 10.48550/arXiv.2410.06158. URL https://arxiv.org/abs/2410.06158.

Yanjiang Guo, Yucheng Hu, Jianke Zhang, Yen-Jen Wang, Xiaoyu Chen, Chaochao Lu, and Jianyu Chen. Prediction with action: Visual policy learning via joint denoising process. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/cbe25fa0e7c7084049276888a09acc8d-Abstract-Conference.html.

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

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2555–2565. PMLR, 2019. URL https://proceedings.mlr.press/v97/hafner19a.html.

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

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640:647–653, 2025. doi: 10.1038/s41586-025-08744-2. URL https://doi.org/10.1038/s41586-025-08744-2.

Nick Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, pp. 47376–47405, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/cf73d57b6dcda32b293df7c2d5341f49-Paper-Conference.pdf.

Nicklas A. Hansen, Hao Su, and Xiaolong Wang. Temporal difference learning for model predictive control. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 8387–8406. PMLR, 2022. URL https://proceedings.mlr.press/v162/hansen22a.html.

Jingyang He, Guangrun Li, Jieyu Zhang, Chengkai Hou, Zhengping Che, and Shanghang Zhang. Demo-JEPA: Joint-embedding predictive architecture for one-shot cross-embodiment imitation. *arXiv preprint arXiv:2605.20811*, 2026. doi: 10.48550/arXiv.2605.20811. URL https://arxiv.org/abs/2605.20811.

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. doi: 10.48550/arXiv.2603.19312. URL https://arxiv.org/abs/2603.19312.

Uladzislau Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, Tim G. J. Rudner, and Yann LeCun. Learning from reward-free offline data: A case for planning with latent dynamics models. In *Advances in Neural Information Processing Systems*, volume 38, pp. 43905–43941. Curran Associates, Inc., 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/3e7cf447f21cd11c846463affefce665-Abstract-Conference.html.

Yuqi Wang, Xinghang Li, Wenxuan Wang, Junbo Zhang, Yingyan Li, Yuntao Chen, Xinlong Wang, and Zhaoxiang Zhang. Unified vision-language-action model. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/82f625a28d822d2748b9f5c4f9a89bb9-Abstract-Conference.html.

Hongtao Wu, Ya Jing, Chi-Lam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. GR-1: Unleashing large-scale video generative pre-training for visual robot manipulation. *arXiv preprint arXiv:2312.13139*, 2023. doi: 10.48550/arXiv.2312.13139. URL https://arxiv.org/abs/2312.13139.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, Ayaan Malik, Kyungmin Lee, William Liang, Nadun Ranawaka, Jiasheng Gu, Yinzhen Xu, Guanzhi Wang, Fengyuan Hu, Avnish Narayan, Johan Bjorck, Jing Wang, Gwanghyun Kim, Dantong Niu, Ruijie Zheng, Yuqi Xie, Jimmy Wu, Qi Wang, Ryan Julian, Danfei Xu, Yilun Du, Yevgen Chebotar, Scott Reed, Jan Kautz, Yuke Zhu, Linxi Jim Fan, and Joel Jang. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026. doi: 10.48550/arXiv.2602.15922. URL https://arxiv.org/abs/2602.15922.

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. URL https://arxiv.org/abs/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. doi: 10.48550/arXiv.2605.09241. URL https://arxiv.org/abs/2605.09241.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 79115–79135. PMLR, 2025. URL https://proceedings.mlr.press/v267/zhou25t.html.

A Environment and Evaluation Protocol Details

We evaluate AGR-JEPA in four pixel-based continuous-control environments following the protocol of LeWorldModel (LeWM) (Maes et al., 2026). The benchmark suite spans navigation, articulated reaching, contact-rich planar manipulation, and three-dimensional robot manipulation. In all environments, the world model receives only rendered visual observations and action sequences; it does not use rewards, task labels, or proprioceptive state as model inputs. Goal-conditioned planning is performed by encoding the current observation history and a goal observation, rolling out candidate action sequences in latent space, and selecting actions according to terminal latent goal-matching cost.

Two-Room. Two-Room is a continuous two-dimensional navigation environment. A point agent is initialized at a random location in one of two rooms separated by a wall, while the goal is sampled in the other room. Since the rooms are connected only by a narrow doorway, a successful trajectory must both move toward the doorway and pass through it before approaching the goal. This environment primarily tests whether a learned latent model preserves global spatial structure and can support multi-step action ranking when a direct path to the goal is obstructed.

Reacher. Reacher is a planar control task involving a two-link robotic arm. The agent controls the arm to move its end effector toward a target sampled in the workspace. Although the environment is visually and physically simpler than the manipulation settings, successful planning requires modeling the coupled, nonlinear kinematics of articulated motion. It therefore assesses whether action-conditioned latent rollouts faithfully capture fine-grained motion consequences needed to reach a specified target configuration.

Push-T. Push-T is a continuous two-dimensional non-prehensile manipulation task. A point-like end effector must push a T-shaped object on a tabletop until its pose matches a target configuration. Because interactions are restricted to pushing, the task requires the planner to reason about contact, object translation, and orientation changes; moving the end effector close to the goal alone is insufficient. Push-T consequently provides a contact-rich test of whether predicted action consequences remain distinguishable over multi-step rollouts.

OGB-Cube. OGB-Cube is a pixel-based three-dimensional robotic manipulation task from the OGBench suite. A robot arm controls its end effector to manipulate a cube toward a specified target position. The task introduces three-dimensional geometry, visually more complex scenes, and coordinated arm–object interactions, making it substantially more challenging than the planar tasks. It tests whether a world model trained from images can retain the control-relevant information required to compare candidate manipulation trajectories in a visually rich setting.

Evaluation protocol. Our main evaluation follows the random valid-start protocol inherited from LeWM (Maes et al., 2026), rather than a full-episode rollout from the beginning of each demonstration. For each evaluation seed, we first sample valid starting indices from the offline dataset such that a fixed future offset is available. The observation at the selected starting index provides the initial observation history, and the observation at the corresponding future index serves as the goal observation. This protocol evaluates whether the model can plan from diverse in-distribution states toward visually specified future states. At each decision point, CEM optimizes candidate sequences of normalized action blocks using the terminal latent goal-matching cost. When frame skipping is used, each model-level action corresponds to a block of F consecutive environment actions. We use receding-horizon MPC: the first optimized action block is executed in the environment, a new observation is received, and planning is repeated. The Action Decoder is not used during this procedure. We report planning success as mean $\pm$ standard deviation across three random seeds. For each seed, the same evaluation protocol, planning budget, and evaluation settings are used for all compared variants. The reported number of evaluation trials per seed is 500.

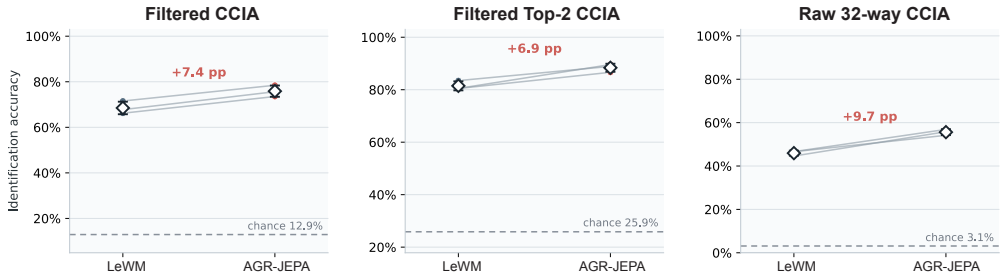

Figure 7: **Large-scale counterfactual consequence identification on Push-T ($K = 32$).** AGR-JEPA consistently outperforms LeWM across the three paired evaluation seeds. Left: Filtered CCIA, evaluated only on terminal consequences that are separable in both models' latent spaces. Middle: Filtered Top-2 CCIA. Right: Raw 32-way CCIA, evaluated over all initially sampled branches in accepted candidate sets. Diamonds denote mean $\pm$ standard deviation across three evaluation seeds; thin lines connect paired seeds.

B Counterfactual Consequence Identification Protocol and Large-Scale Results

This section presents the complete Counterfactual Consequence Identification Accuracy (CCIA) protocol and reports a large-scale evaluation with $K = 32$ counterfactual action branches. The analysis compare the full AGR-JEPA model with LeWorldModel (LeWM) on Push-T to evaluate whether AGR-JEPA improves the ability of latent rollouts to distinguish the consequences of action sequences.

B.1 Counterfactual Candidate Construction

For each sampled observation history $\mathcal{H}_t$, we construct a counterfactual candidate set containing K action sequences:

$$\mathcal{A}_t = \left\{ A_t^{(1)}, \ldots, A_t^{(K)} \right\}, \qquad A_t^{(i)} = \left(a_t^{(i)}, \ldots, a_{t+H-1}^{(i)} \right). \tag{8}$$

All action sequences within a set start from the same restored simulator state. We execute each sequence in the real environment to obtain its ground-truth terminal observation $o_{t+H}^{(i)}$, which is then encoded by the evaluated model:

$$z_{q,t+H}^{(i)} = f_{\theta_q} \left(o_{t+H}^{(i)} \right), \tag{9}$$

where q denotes either LeWM or AGR-JEPA.

Starting from the same initial observation history, we also roll out each action sequence through the latent dynamics model to obtain its predicted terminal latent $\hat{z}_{q,t+H}^{(i)}$. The two models therefore use exactly the same sampled histories, action sequences, simulator resets, and ground-truth counterfactual branches, while each model is evaluated in its own learned representation space. This protocol tests whether the predicted consequence of each action sequence can be matched to the ground-truth consequence generated by that *same* sequence among multiple competing counterfactual outcomes.

B.2 Counterfactual Consequence Identification Accuracy

For each predicted consequence, we identify its nearest ground-truth counterfactual consequence in latent space:

$$\hat{\jmath}_{q,i} = \arg\min_{j \in V} \left\| \hat{z}_{q,t+H}^{(i)} - z_{q,t+H}^{(j)} \right\|_2^2, \tag{10}$$

where V is the shared valid-candidate set defined below. A prediction is counted as correct if $\hat{\jmath}_{q,i} = i$. Counterfactual Consequence Identification Accuracy (CCIA) is the proportion of correctly

Table 6: Raw CCIA across candidate-set sizes using saved $K = 32$ candidate sets. Values are mean $\pm$ sample standard deviation across three evaluation seeds.

Candidate-set sizes 2–16				**Candidate-set sizes** 18–32			
k	Random	LeWM	AGR-JEPA	k	Random	LeWM	AGR-JEPA
2	50.00%	88.85 ± 0.76	91.85 ± 0.49	18	5.56%	54.67 ± 1.34	63.67 ± 1.15
4	25.00%	77.89 ± 1.30	83.12 ± 0.88	20	5.00%	53.06 ± 1.30	62.23 ± 1.17
6	16.67%	71.57 ± 1.46	77.96 ± 1.03	22	4.55%	51.60 ± 1.27	60.92 ± 1.20
8	12.50%	67.12 ± 1.49	74.29 ± 1.08	24	4.17%	50.28 ± 1.25	59.70 ± 1.23
10	10.00%	63.68 ± 1.48	71.43 ± 1.10	26	3.85%	49.06 ± 1.22	58.58 ± 1.26
12	8.33%	60.88 ± 1.45	69.06 ± 1.11	28	3.57%	47.94 ± 1.20	57.53 ± 1.29
14	7.14%	58.52 ± 1.41	67.04 ± 1.12	30	3.33%	46.90 ± 1.18	56.54 ± 1.33
16	6.25%	56.47 ± 1.37	65.26 ± 1.13	32	3.125%	45.94 ± 1.16	55.61 ± 1.37

identified consequences over all valid candidates from all accepted candidate sets:

$$\mathrm{CCIA}_q = \frac{1}{\sum_m |V_m|} \sum_m \sum_{i \in V_m} \mathbb{I}\left[\hat{j}_{q,i}^{(m)} = i\right], \tag{11}$$

where m indexes accepted candidate sets. Thus, CCIA evaluates whether the model preserves the identity of an action-conditioned consequence relative to the consequences of alternative actions, rather than merely whether an individual predicted endpoint is plausible.

Consequence-separability filtering. Different action sequences can occasionally produce visually and physically indistinguishable terminal states, particularly when neither sequence leads to object contact. Requiring a model to distinguish such ambiguous consequences would make the matching problem ill-posed. We therefore exclude ambiguous ground-truth consequences before computing the primary CCIA metric.

For model q, candidate i is considered valid when its ground-truth terminal latent is sufficiently separated from every other ground-truth terminal latent:

$$\min_{j \neq i} \left\| z_{q,t+H}^{(i)} - z_{q,t+H}^{(j)} \right\|_2 \geq \tau_q, \tag{12}$$

where τ_q is fixed before evaluation as the 20th percentile of pairwise terminal-latent distances computed on an independent calibration set. To ensure a strictly paired comparison, we use the intersection of the valid masks from LeWM and AGR-JEPA:

$$V = V_{\mathrm{LeWM}} \cap V_{\text{AGR-JEPA}}. \tag{13}$$

Candidate sets with fewer than four shared valid candidates are rejected and resampled.

Filtered CCIA, computed only over V, is our primary metric. We additionally report Raw CCIA, which performs matching over all K initially sampled candidates in accepted sets. Raw CCIA has a fixed K-way random baseline of $1/K$. In contrast, the chance level for Filtered CCIA depends on the number of retained candidates; after aggregation across M accepted sets, it is $M/\sum_m |V_m|$.

Evaluation configuration. We use an observation history of three model steps and roll out each candidate for $H = 5$ action blocks. Each action block contains five simulator actions; thus, every real and imagined counterfactual branch spans 25 simulator steps. For each of three evaluation seeds, we evaluate 100 accepted candidate sets, yielding 300 paired sets in total. Candidate action sequences are sampled uniformly from the environment action bounds. Both models are evaluated on the same sampled histories, action sequences, simulator resets, and shared valid-candidate masks.

B.3 Large-Scale Results with $K = 32$

We further increase the initial candidate-set size to $K = 32$, producing a substantially more challenging counterfactual matching problem. Figure 7 summarizes the results. Across the three evaluation seeds, AGR-JEPA achieves a Filtered CCIA of $75.87 \pm 2.47\%$, compared with $68.50 \pm 2.75\%$

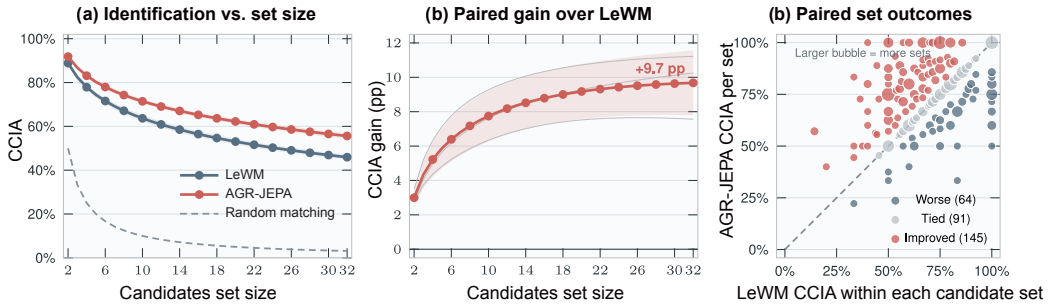

Figure 8: **Effect of counterfactual candidate-set size.** Left: Raw CCIA as a function of the number of competing counterfactual branches. Middle: AGR-JEPA's paired improvement over LeWM. Both panels are computed by averaging over subsets of accepted $K = 32$ candidate sets and therefore provide a candidate-set-size sensitivity analysis rather than independently sampled experiments at every set size. Right: paired per-set Filtered CCIA for $K = 32$. Points above the diagonal indicate candidate sets on which AGR-JEPA outperforms LeWM; bubble size denotes the number of sets at the corresponding coordinate.

for LeWM, yielding a paired improvement of 7.38 ± 0.49 percentage points. The improvement is positive for every seed, with gains of $+7.88$, $+7.34$, and $+6.91$ percentage points.

The result is consistent for the auxiliary Top-2 identification metric: AGR-JEPA attains $88.37 \pm 1.54\%$, compared with $81.48 \pm 1.73\%$ for LeWM. Moreover, the advantage remains when matching is performed over all 32 initially sampled branches. AGR-JEPA obtains a Raw 32-way CCIA of $55.61 \pm 1.37\%$, whereas LeWM obtains $45.94 \pm 1.16\%$. Both models are well above the fixed 32-way random baseline of 3.125%.

After consequence-separability filtering, the shared valid-candidate set contains 7.75 candidates on average, corresponding to a Filtered-CCIA chance level of 12.93%. The paired set-level analysis leads to the same conclusion: among the 300 accepted candidate sets, AGR-JEPA improves over LeWM on 145 sets (48.3%), ties on 91 sets (30.3%), and performs worse on 64 sets (21.3%). Its equal-weighted mean per-set CCIA is 76.46%, compared with 69.50% for LeWM. These results show that the advantage of AGR-JEPA persists when the initial counterfactual branch set is expanded to 32 candidate action sequences.

Figure 8 shows that the advantage of AGR-JEPA persists as the number of counterfactual distractors increases. Raw CCIA decreases for both models as the matching problem becomes more difficult, yet AGR-JEPA's improvement increases from approximately 2.9 percentage points at $K = 2$ to 9.7 points at $K = 32$. Together with the paired set-level analysis, this result indicates that AGR-JEPA more reliably preserves the identity of action-conditioned consequences among a large set of plausible counterfactual outcomes.

C Implementation and Reproducibility Details

Model architecture. AGR-JEPA uses the same visual encoder and latent-dynamics backbone as LeWorldModel (LeWM) (Maes et al., 2026). The visual encoder is a ViT-Tiny trained from scratch on 224×224 RGB observations, with patch size 14, 12 Transformer layers, 3 attention heads, and latent dimension $d_z = 192$. The action-conditioned dynamics predictor is a six-layer causal Transformer with 16 attention heads, MLP dimension 2048, and dropout 0.1. The predictor receives a history of latent representations and the corresponding action embeddings, and predicts one subsequent latent transition at each aligned history position.

Actions are encoded by a temporal smoothing module followed by a two-layer MLP. When frame skipping is used, a model-level action a_t denotes the normalized block of F consecutive environment actions associated with the transition from o_t to o_{t+1}. Thus, if the original action dimension is d_a, then $a_t \in \mathbb{R}^{Fd_a}$. All action inputs and action-recovery targets are normalized using dataset statistics.

For control grounding, we use a shared two-layer Action Decoder with GELU activation:

$$D_\psi : 192 \to 256 \to Fd_a.$$

For example, in Push-T, $F = 5$ and $d_a = 2$, so the decoder predicts a 10-dimensional normalized action block. The same decoder is used for the target-displacement and predicted-displacement branches; no separate decoder is introduced for either branch.

Training objective and optimization. For residual dynamics, the transition loss is

$$\mathcal{L}_{\text{res}} = \left\| \widehat{\Delta} z_t - (z_{t+1} - z_t) \right\|_2^2.$$

The complete objective is

$$\mathcal{L}_{\text{AGR}} = \mathcal{L}_{\text{res}} + \lambda_{\text{sig}} \mathcal{L}_{\text{SIGReg}} + \lambda_{\text{cg}} \left(\mathcal{L}_{\text{cg,tar}} + \lambda_{\text{pred}} \mathcal{L}_{\text{cg,pred}} \right).$$

Unless otherwise specified, we use $\lambda_{\text{sig}} = 0.09$, $\lambda_{\text{cg}} = 1.0$, and $\lambda_{\text{pred}} = 1.0$. The coefficient of the transition loss is fixed to one. We optimize all trainable parameters jointly using AdamW with learning rate 5×10^{-5} and weight decay 10^{-3}.

Training samples are contiguous windows from the offline trajectories. The default latent-history length is three model steps and the prediction offset is one step. Rather than supervising only the final position of a sampled window, the transition and grounding losses are computed over all valid aligned history–action–future tuples in the window. SIGReg is computed over the latent representations in the observation window. We train for 10 epochs using batch size 128.

Planning. At test time, AGR-JEPA receives only the current visual observation history and a visual goal observation. The encoder maps them into latent space, and CEM optimizes candidate sequences of normalized action blocks according to the terminal latent cost in equation 7. We use a planning horizon of $H = 5$ model steps. With frame skip $F = 5$, this corresponds to a maximum look-ahead of 25 environment steps.

Unless otherwise specified, each CEM iteration samples 300 candidate action sequences and updates the sampling distribution using the 30 lowest-cost elite sequences. We use at most 30 CEM iterations for Push-T and 10 iterations for the remaining environments. Planning is performed in a receding-horizon manner: only the first optimized action block is executed, after which a new observation is acquired and planning is repeated. The Action Decoder is discarded after training and is not evaluated during latent rollout, candidate generation, or cost computation; AGR-JEPA therefore has the same planning-time architecture as its residual-dynamics backbone.

Fair comparison. For the module and grounding-source ablations, all variants use the same visual encoder, action encoder, dynamics-predictor architecture, training data, optimization settings, and planning protocol. The variants differ only in (i) whether the predictor output is interpreted as an absolute next latent or a residual latent update, and (ii) whether control grounding is disabled or applied to target displacements, predicted displacements, or both. In particular, residual and absolute variants have identical predictor capacity; the residual formulation changes the transition parameterization rather than the number of model parameters.

D Sequence-Based Latent Dynamics Training and Planning

For clarity, the main paper presents AGR-JEPA using single-step notation. In practice, however, the dynamics predictor is a causal Transformer that operates on a sequence of latent representations and action blocks. This appendix describes the sequence-based formulation used for training and planning.

D.1 Sequence-Based Latent Dynamics Training

Let a training trajectory contain observations and aligned action blocks,

$$\tau = \{o_1, a_1, o_2, a_2, \ldots, o_T\}.$$

When frame skipping is used, each $a_t \in \mathbb{R}^{Fd_a}$ denotes the normalized block of F environment actions that transitions the environment from o_t to o_{t+1}. The visual encoder maps each observation in a sampled trajectory window to a latent representation,

$$z_t = f_\theta(o_t) \in \mathbb{R}^{d_z}.$$

For a context length L, the causal action-conditioned predictor receives a latent history and its aligned action history,

$$\mathbf{z}_{t-L+1:t} = (z_{t-L+1}, \ldots, z_t), \qquad \mathbf{a}_{t-L+1:t} = (a_{t-L+1}, \ldots, a_t).$$

The causal masking ensures that the prediction associated with position i depends only on the representations and actions available up to that position. For compactness, we denote the sequence of predictor outputs by

$$(r_{t-L+1}, \ldots, r_t) = p_\phi(\mathbf{z}_{t-L+1:t}, \mathbf{a}_{t-L+1:t}).$$

Each output r_i predicts the transition induced by a_i, whose target is the latent representation z_{i+1}. Thus, a sampled sequence provides multiple aligned transition-supervision terms rather than supervising only its final time step.

For the absolute-dynamics baseline, r_i is interpreted directly as the next latent representation:

$$\hat{z}_{i+1} = r_i, \qquad \mathcal{L}_{\text{abs}} = \frac{1}{L} \sum_{i=t-L+1}^{t} \|\hat{z}_{i+1} - z_{i+1}\|_2^2.$$

For AGR-JEPA, r_i is instead interpreted as an action-conditioned residual update:

$$\widehat{\Delta z}_i = r_i, \qquad \hat{z}_{i+1} = z_i + \widehat{\Delta z}_i,$$

which yields the sequence-level residual transition loss

$$\mathcal{L}_{\text{res}} = \frac{1}{L} \sum_{i=t-L+1}^{t} \left\| \widehat{\Delta z}_i - (z_{i+1} - z_i) \right\|_2^2.$$

Equivalently, this objective minimizes the next-latent prediction error after adding the predicted residual to the current latent state. The residual form makes the predicted update explicit, which permits control grounding to be applied directly to each transition in the training window.

Specifically, the control-grounding losses are aggregated over all valid aligned transitions:

$$\mathcal{L}_{\text{cg,tar}} = \frac{1}{L} \sum_{i=t-L+1}^{t} \|D_\psi(z_{i+1} - z_i) - a_i\|_2^2,$$

$$\mathcal{L}_{\text{cg,pred}} = \frac{1}{L} \sum_{i=t-L+1}^{t} \left\| D_\psi(\widehat{\Delta z}_i) - a_i \right\|_2^2.$$

The overall objective is then obtained by combining the sequence-averaged transition loss, SIGReg loss, and enabled control-grounding terms, as defined in equation 5. SIGReg is computed over the latent representations in the sampled observation window. During training, all context latents are encoded from ground-truth observations; hence, the predictor is trained under teacher forcing.

D.2 Autoregressive Latent Planning

At test time, the model receives the latest L observations and a goal observation. The current observation history is encoded as

$$\mathbf{z}_{t-L+1:t} = (f_\theta(o_{t-L+1}), \ldots, f_\theta(o_t)),$$

and the goal image is encoded as $z_g = f_\theta(o_g)$. Given a candidate action sequence

$$A_{t:t+H-1} = (a_t, \ldots, a_{t+H-1}),$$

the predictor generates an imagined rollout autoregressively. At the first rollout step, it uses the encoded history and the candidate action a_t:

$$\widehat{\Delta z}_t = p_\phi(\mathbf{z}_{t-L+1:t}, \mathbf{a}_{t-L+1:t})_t\,, \qquad \hat{z}_{t+1} = z_t + \widehat{\Delta z}_t.$$

Here, the subscript t denotes the output aligned with the most recent position of the causal sequence. The predicted latent $\hat{z}_{t+1}$ is then appended to the history, while the oldest latent is discarded. Repeating this procedure gives

$$\hat{z}_{t+k+1} = \hat{z}_{t+k} + p_\phi(\hat{\mathbf{z}}_{t+k-L+1:t+k}, \mathbf{a}_{t+k-L+1:t+k})_{t+k}\,, \quad k = 0, \ldots, H-1,$$

where $\hat{\mathbf{z}}_{t+k-L+1:t+k}$ contains encoded latents before the rollout boundary and recursively predicted latents afterwards.

The planner evaluates each candidate action sequence using only its terminal predicted latent:

$$J(A_{t:t+H-1}) = \|\hat{z}_{t+H} - z_g\|_2^2\,.$$

CEM optimizes this objective by iteratively sampling candidate action-block sequences, rolling them out through the latent dynamics model, and updating the sampling distribution from the elite candidates. The first optimized action block is executed in the environment, after which a new visual observation is received and planning is repeated. The Action Decoder is not used during this autoregressive rollout, CEM optimization, or planning-cost evaluation.