FROM STRUCTURAL PRIORS TO LOCAL CAUSALITY IN WORLD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Object-centric world models represent scenes as distinct entities, but this decomposition alone does not specify how action information should enter the scene and propagate between objects. We introduce a robot-mediated action prior and state-dependent sparse gates to organize information flow within these models. Our design separates interaction admissibility from interaction selection: a physical attention mask restricts direct action–slot exchange to robot slots, while gates select among admissible connections. The gates and predictor are jointly optimized with prediction loss and sparsity regularization, without object–object edge supervision. Multi-layer message passing enables action information to reach other objects through the robot. We define local transition dependencies through input interventions on a fixed predictor, restricting our causal analysis to the learned transition mechanism. We evaluate the approach on four visual planning benchmarks. Our model achieves 100% success on Two-Room and 92.00% on PushT. On OGBench-Cube and OGBench-Scene, it achieves 84.67% and 78.00%, respectively, compared with 75.33% and 72.67% for the reproduced LeWorldModel baseline. Structural ablations on OGBench-Cube show greater prediction-error stability under candidate nuisance-slot removal and lower action-effect error on selected contact samples. Route interventions further identify learned connections carrying local action responses within the tested eligible neighborhoods. Together, these results support compact, robot-mediated transition modeling with competitive planning performance.

1 INTRODUCTION

World models (Zhou et al., 2025; Nam et al., 2026; Balestriero & LeCun, 2025; Zhang et al., 2026b; Zhao et al., 2026; Zeng et al., 2026) learn action-conditioned dynamics in latent space, enabling agents to predict future states and plan actions through imagined rollouts. In the robotic manipulation tasks considered here, control commands act directly on the robot, whose interactions transmit their effects to other objects. Beyond predictive accuracy, this motivates us to examine whether information flow within a learned world model reflects these pathways. Explicitly representing these pathways requires distinguishing the entities involved as units of interaction. Generic visual encoders, such as Vision Transformers, provide features for prediction but do not necessarily organize them into explicit entity representations. Object-centric world models Nishimoto & Matsubara (2026); Liu et al. (2026); Vakhitov et al. (2026); Spieler et al. (2026) provide a natural foundation by representing a scene as a collection of object slots and predicting their evolution conditioned on actions. However, slot decomposition alone does not determine how entities should exchange information Locatello et al. (2020). The transition model must additionally determine which entities exchange information, under what conditions, and how action information propagates through them.

These interactions vary with the system state; for example, the onset and loss of contact change how objects influence one another. Analyses of simulator dynamics reveal limited global sparsity in the tested environments, while local sparsity patterns vary with state and contact conditions Pandaram et al. (2026). This motivates modeling state-dependent local interactions without assuming a fixed, globally sparse structure for physical dynamics. Moreover, exploiting locally factored dynamics for counterfactual data augmentation can improve data efficiency, demonstrating the practical value of local structure Pitis et al. (2020).

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.

Exploiting such local structure requires reliable estimation of dependencies. Simply thresholding soft attention in a predictive model can retain erroneous dependencies. Combining sparsity regularization with hard attention can improve local structure estimation and robustness. When complemented by mechanisms for adapting to environmental changes, such models can also generalize to unseen environments and adapt efficiently Lei et al. (2025). These findings provide empirical support for sparsity as an inductive bias for learning local causal structures. However, sparsity regularization encourages fewer connections without specifying which entity should directly receive action information. This motivates incorporating physical action priors to guide the organization of local interactions.

To this end, we introduce sparse gates constrained by a robot-mediated action prior into an object-centric world model to learn compact, state-dependent interaction pathways. Our design separates interaction admissibility from interaction selection: a physical attention mask restricts direct action–slot exchange to robot slots, while state-dependent gates select among admissible connections and are optimized jointly with the predictor using prediction loss and sparsity regularization (Fig. 2). The action interface is prescribed by the prior, whereas object–object connections are learned from data. The routing graph is recomputed for each input, and successive Transformer layers support multi-hop propagation within a prediction step, allowing action information to reach other objects through the robot. We further define transition dependencies through input interventions on a fixed predictor within local neighborhoods where the routing graph remains unchanged, distinguishing admissible information pathways from the predictive dependencies actually realized by the model. Our causal analysis concerns the learned transition mechanism and does not claim identification of the underlying physical causal graph.

2 METHOD

To make this distinction precise, we introduce two complementary definitions. The first specifies the action prior and quantifies routing sparsity; the second characterizes local predictive dependence through input interventions.

At a fixed time step, let $b = (z^1, \dots, z^K, a)$, with $K \geq 2$, collect the slot states and action token, and write the next-state prediction as $\widehat{z}^i = F_{\theta,i}(b)$. Define the input index set $\mathcal{I} = \{1, \dots, K, \mathrm{a}\}$ and a nonempty set of robot-slot indices $\mathcal{R} \subseteq \{1, \dots, K\}$. The directed routing graph $G(b) = (\mathcal{I}, E(b))$ describes direct information flow within the prediction step.

Definition 1 (Action prior and routing sparsity)**.** A routing graph $G(b)$ respects the action prior if

$$(\mathrm{a}, i) \in E(b) \ \vee \ (i, \mathrm{a}) \in E(b) \implies i \in \mathcal{R}, \qquad i = 1, \dots, K. \tag{1}$$

Let $E_{\mathrm{obj}}(b)$ denote the directed edges between distinct object slots. Their density is

$$d_{\mathrm{obj}}(G(b)) = \frac{|E_{\mathrm{obj}}(b)|}{K(K-1)}. \tag{2}$$

During training, a attention mask enforces the action prior, while sparsity regularization encourages fewer object–object connections. The routing graph specifies direct communication: an input may affect a prediction through multiple hops, while the existence of a route does not guarantee an effect on the output. To characterize the dependencies actually realized by the predictor, we consider input interventions within neighborhoods where the routing graph remains unchanged.

Definition 2 (Local transition causal parents)**.** For a fixed predictor F_θ, let $\mathcal{U} = \prod_{j \in \mathcal{I}} \mathcal{U}_j$ be an open product neighborhood on which the routing graph is constant, $G(b) = g$ for all $b \in \mathcal{U}$. The local causal parent set of prediction i is

$$\mathrm{Pa}_i^{\mathcal{U}} = \left\{ j \in \mathcal{I} : \begin{array}{l} \exists b, b' \in \mathcal{U}, \quad b_{-j} = b'_{-j}, \\ F_{\theta,i}(b) \neq F_{\theta,i}(b') \end{array} \right\}. \tag{3}$$

Thus, an input is a local parent if changing that input alone can alter prediction i within $\mathcal{U}$.

For continuous gate scores with deterministic thresholding, such a neighborhood exists around any input at which no score equals its threshold. The routing graph can therefore vary across the input space while remaining fixed within each neighborhood used for the analysis. With these definitions in place, we next describe how the physical attention mask and sparse gates parameterize the routing graph and how the gates are learned jointly with the predictor.

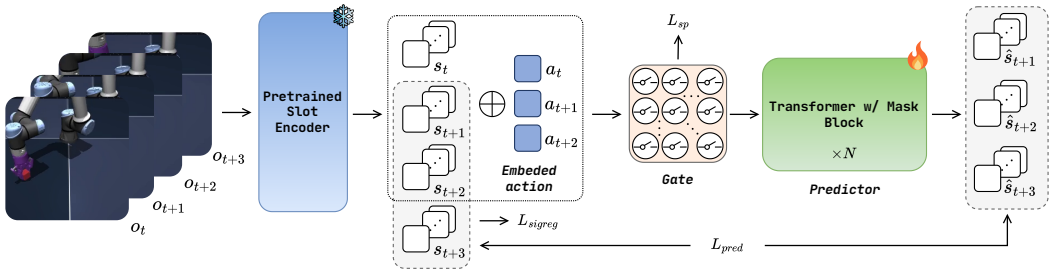


Figure 2: Overview of the training framework combining structural attention priors with learnable routing gates. The attention mask restricts direct action–slot interactions to the robot slot. The training objective combines prediction, sparsity, and SIGReg losses.

2.1 Object-Centric World Dynamics Backbone

Slot representations and robot identification We build on the joint-embedding prediction framework of LeWorldModel Maes et al. (2026). A pretrained object-centric encoder based on Slot Attention Locatello et al. (2020); Seitzer et al. (2023) and self-supervised visual features maps an observation o_t to K slots. A shared trainable projector maps each slot into the prediction space:

$$(s_t^1, \ldots, s_t^K) = E_{\text{slot}}(o_t), \qquad z_t^i = P_\theta(s_t^i) \in \mathbb{R}^d. \tag{4}$$

The pretrained slot encoder remains frozen during dynamics learning. We determine slot semantics through Hungarian matching between decoded slot masks and semantically labeled reference masks obtained with SAM2 (Ravi et al., 2024), thereby identifying the robot-slot set $\mathcal{R}_t$.

Joint slot and action tokens For a transition from t to $t+1$, the predictor receives the current slot embeddings $S_t = [s_t^1, \ldots, s_t^K]$ and action a_t. We map the slot and action embeddings to a common hidden dimension and assemble $K+1$ joint tokens:

$$X_t^{(0)} = [x_t^1, \ldots, x_t^K, x_t^{\text{a}}], \qquad x_t^i = W_s s_t^i, \quad x_t^{\text{a}} = W_a E_a(a_t), \tag{5}$$

where E_a is a trainable action encoder. The superscript (0) denotes the input to the first Transformer block, and $X_t^{(\ell)}$ denotes the token representations after ℓ blocks within the same prediction step. Positional embeddings are omitted from the notation for clarity. We index the joint tokens by $\mathcal{I} = \{1, \ldots, K, \text{a}\}$. Attention operates exclusively among these tokens at time t, and the resulting slot outputs predict the representations at $t+1$. The predictor consists of L Transformer blocks and generates multi-step predictions by autoregressively feeding predicted slot embeddings into subsequent prediction steps.

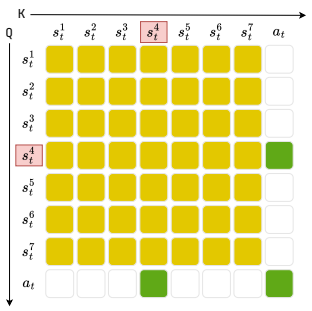


Figure 1: Physical prior attention mask at time t. Yellow cells permit slot-slot attention. Green cells permit action-robot exchange and action self-attention. White cells are masked. The highlighted slot s_t^4 represents the robot in this example.

Physical prior attention mask Let $r_t \in \{0,1\}^K$ indicate robot membership, with $r_t^i = 1$ if $s_t^i \in \mathcal{R}_t$. In query-key order, the physical mask is

$$M_t = \begin{bmatrix} \mathbf{1}_{K\times K} & r_t \\ r_t^\top & 1 \end{bmatrix}. \tag{6}$$

An entry $(M_t)_{ij} = 1$ allows query i to attend to key j, corresponding to the directed message $j \to i$. As illustrated in Fig. 1, the slot-slot block permits all object interactions before gate selection. The action column permits robot slots to receive action information, while the action row permits the action token to receive information from robot slots. Direct exchange between the action token and non-robot slots is blocked in both directions. All self-connections remain available. Thus, the mask imposes the robot interface for action exchange while leaving the object-object interaction structure to be learned.

State-dependent threshold gates Within the physical mask, we select connections using gates computed from the input tokens $X_t^{(0)}$. Separate query and key projections produce directional compatibility scores:

$$p_{t,ij} = \sigma\left(\frac{(q_{t,i}^g)^\top k_{t,j}^g/\sqrt{d_g} - \beta}{\tau}\right), \tag{7}$$

where $q_{t,i}^g$ and $k_{t,j}^g$ are learned projections of normalized input tokens, β is a learned threshold, and τ is the temperature. The forward pass uses the binary adjacency

$$G_{t,ij} = \begin{cases} 1, & i = j, \\ (M_t)_{ij}\,\mathbf{1}[p_{t,ij} \geq 1/2], & i \neq j, \end{cases} \tag{8}$$

which preserves self-connections and excludes connections forbidden by the physical mask. The graph is recomputed for each prediction step and shared across all attention heads and Transformer layers within that step.

Message passing and prediction We use a pre-normalization Transformer (Vaswani et al., 2023) to propagate information along the connections selected by G_t. For each layer and attention head, the masked attention weights are

$$A_{t,ij} = \frac{G_{t,ij}\exp(e_{t,ij})}{\sum_{k\in\mathcal{I}} G_{t,ik}\exp(e_{t,ik})}, \tag{9}$$

where $e_{t,ij}$ is the scaled query–key score; layer and head indices are omitted for clarity. Stacking L blocks enables multi-hop message passing, with action information reaching non-robot slots only through robot slots. Both slot and action tokens are updated throughout the backbone, but only the final slot tokens are decoded:

$$\widehat{s}_{t+1}^i = D_\theta\Big(\mathrm{LN}_{\mathrm{out}}(X_{t,i}^{(L)})\Big), \qquad i = 1, \ldots, K, \tag{10}$$

where D_θ maps hidden tokens to predicted slot embeddings. The resulting transition model $\widehat{S}_{t+1} = F_\theta(S_t, a_t)$ is applied autoregressively for multi-step prediction:

$$\widehat{S}_{t+h+1} = F_\theta(\widehat{S}_{t+h}, a_{t+h}), \qquad \widehat{S}_t = S_t, \tag{11}$$

with the interaction graph recomputed from the input at each step.

From routing structure to local causal dependence The architecture above parameterizes the routing graph in Definition 1. We now relate this graph to the local causal parents defined in Definition 2. Since information can propagate across multiple Transformer layers, predictive dependence is governed by graph reachability rather than individual edges. The following theorem establishes that local causal parents are contained in the set of inputs reachable within L message-passing steps.

Theorem 1 (Local causal structure of the learned transition)**.** Consider the hard-gated L-layer predictor with tokenwise input encodings, normalization, feed-forward operations, and readout. Assume that all cross-token computations use the shared binary graph with self-loops $g_{ii} = 1$, and that masked entries are excluded from both the attention numerator and denominator. At any input with nonzero admissible gate margins, there is an open product neighborhood $\mathcal{U}$ with graph g such that

$$\mathrm{Pa}_i^{\mathcal{U}} \subseteq \mathcal{P}_i(g) := \{j : (g^{[L]})_{ij} = 1\}, \tag{12}$$

where $g^{[L]}$ denotes the Boolean matrix power. Moreover, the transition admits the factorization

$$F_{\theta,i}(b) = f_i^{\mathcal{U}}\big(b_{\mathrm{Pa}_i^{\mathcal{U}}}\big), \qquad b \in \mathcal{U}, \tag{13}$$

and $\mathrm{Pa}_i^{\mathcal{U}}$ is the unique minimal input index set determining $F_{\theta,i}$ on $\mathcal{U}$. Consequently, a local intervention on an input with no active path of at most L hops to prediction i leaves that prediction unchanged.

The theorem connects the learned routing structure to local intervention-induced dependence: inputs without an active path to a prediction cannot affect it within $\mathcal{U}$. Under the physical mask, any action-to-object path must pass through a robot slot. The inclusion need not be an equality, since an available path does not necessarily produce a functional dependence. Thus, the graph bounds the predictor's local causal support without identifying the causal structure of the physical system. We next describe the objective used to learn predictive dynamics while encouraging sparse routing within these structural constraints.

2.2 TRAINING OBJECTIVE

We jointly optimize the trainable projector, action encoder, dynamics predictor, and gates while keeping the pretrained slot encoder frozen:

$$\mathcal{L} = \mathcal{L}_{\text{pred}} + \lambda_{\text{reg}}\mathcal{L}_{\text{SIGReg}} + \lambda_{\text{sp}}\mathcal{L}_{\text{sp}}. \tag{14}$$

Prediction and representation regularization The prediction loss matches corresponding next-step slots:

$$\mathcal{L}_{\text{pred}} = \mathbb{E}_{(o_t,a_t,o_{t+1})}\left[\frac{1}{Kd}\sum_{i=1}^{K}\left\|\widehat{z}_{t+1}^{i} - z_{t+1}^{i}\right\|_2^2\right]. \tag{15}$$

Gradients flow through both input and target branches of the shared projector. To discourage representation collapse, we apply SIGReg (Balestriero & LeCun, 2025) to projected embeddings pooled across samples and slots at each observation position, including prediction targets. It encourages an isotropic Gaussian distribution through random one-dimensional projections.

Sparsity and gate optimization For admissible non-self connections $\mathcal{C}_t = \{(i,j) : i \neq j,\ (M_t)_{ij} = 1\}$, we penalize the mean continuous gate value:

$$\mathcal{L}_{\text{sp}} = \mathbb{E}_t\left[\frac{1}{|\mathcal{C}_t|}\sum_{(i,j)\in\mathcal{C}_t} p_{t,ij}\right]. \tag{16}$$

This penalty encourages sparse object–object and action–robot connections without graph supervision or a fixed sparsity budget; the physical mask enforces the action prior exactly.

We train the binary gates using an attention-level straight-through estimator (Bengio et al., 2013):

$$A^{\text{train}} = A^{\text{hard}} + A^{\text{soft}} - \text{sg}(A^{\text{soft}}), \tag{17}$$

where sg denotes stop-gradient. The soft surrogate normalizes gate-weighted attention over all admissible connections, with fixed self-gates and detached attention scores. Thus, the forward pass uses hard routing, while the surrogate provides gate gradients even for currently closed connections. Inference uses hard attention only. A detailed description of the straight-through estimation procedure is provided in Appendix A.4.

3 EXPERIMENTS

3.1 EXPERIMENTAL SETTINGS

Datasets We evaluate on four visual planning benchmarks: Two-Room Zhou et al. (2025), PushT Chi et al. (2024), OGBench-Cube Park et al. (2025), and OGBench-Scene, covering navigation, reaching, planar pushing, and robotic manipulation. We further assess robustness on OGBench-Cube under appearance, cube-size, viewpoint, and lighting interventions. For experiments conducted by us, we report mean planning success rates and sample standard deviations over three random seeds.

Baselines We compare against PLDM Sobal et al. (2025), DINO-WM Zhou et al. (2025), SlotContrast-WM Nazirjonov et al. (2026), Sub-JEPA Zhao et al. (2026), Delta-JEPA Zhang et al. (2026b), PhyLatent Zeng et al. (2026), Causal-JEPA Nam et al. (2026), and LeWorldModel Maes et al. (2026) (LeWM). We reproduce LeWM and Causal-JEPA under the same experimental setup and obtain the remaining results from published sources, as specified in Table 1. The original LeWM also serves as the baseline for evaluating robustness under interventions.

3.2 DOES THE MODEL LEARN A CAUSALLY ALIGNED GRAPH STRUCTURE?

Structural ablations show that the physical attention mask and sparsity regularization play complementary roles in organizing information flow. Sparsity alone reduces connectivity but retains direct action connections to non-robot slots, whereas the mask alone enforces the robot interface while leaving

Table 1: Comparison of planning success rates (%) on four benchmarks. Higher is better. *Results collected from published literature. †Results taken from the respective original papers. ‡Results reproduced by us under the same experimental setup. For our experiments, we report mean success rates across three random seeds, with sample standard deviations ($n-1$ denominator) shown as subscripts.

Method	**Two-Room**	**PushT**	**OGBench-Cube**	**OGBench-Scene**
PLDM *	97.00	78.00	65.00	-
DINO-WM *	100.00	74.00	86.00	-
SlotContrast-WM *	-	85.00	73.00	-
Sub-JEPA †	$95.00_{\pm 2.76}$	$89.00_{\pm 5.33}$	$76.33_{\pm 5.99}$	-
Delta-JEPA †	$100.00_{\pm 0.00}$	$89.07_{\pm 1.90}$	$79.27_{\pm 1.81}$	-
PhyLatent †	$98.00_{\pm 1.00}$	$75.33_{\pm 2.08}$	$78.10_{\pm 2.80}$	-
Causal-JEPA ‡	$100.00_{\pm 0.00}$	$87.33_{\pm 5.77}$	$52.67_{\pm 5.77}$	$48.67_{\pm 11.02}$
LeWM ‡	$99.67_{\pm 0.58}$	$91.33_{\pm 4.16}$	$75.33_{\pm 5.77}$	$72.67_{\pm 3.06}$
Ours	$100.00_{\pm 0.00}$	$92.00_{\pm 2.00}$	$84.67_{\pm 5.03}$	$78.00_{\pm 3.46}$

Table 2: Interventional evaluation of causal alignment on OGBench-Cube. We report the absolute percentage change in prediction error after removing all candidate nuisance slots and the cube action-effect prediction error on contact samples. Lower is better for both metrics. Complete metric definitions and evaluation protocols are provided in Appendix A.5.

Configuration	Nuisance-slot removal ΔE (%) ↓	Cube action-effect error on contact samples (mm) ↓
(a) Mask + sparsity (Ours)	0.10	17.18
(b) Sparsity only	19.69	19.21
(c) Mask only	237.98	24.07
(d) Dense	189.74	24.42

routing nearly dense. Combining both yields compact, robot-centered routing without object–object graph supervision. Detailed rollout visualizations are provided in Appendix B and Fig. 10.

Table 1 reports planning success rates. Our model achieves 100% success on Two-Room, matching the best-performing baselines, and obtains the highest success rate on PushT (92.00%), slightly exceeding the reproduced LeWM baseline (91.33%). On OGBench-Cube, it improves over LeWM from 75.33% to 84.67%, while remaining competitive with the reported DINO-WM result (86.00%). On OGBench-Scene, our model achieves the highest success rate of 78.00%, outperforming LeWM (72.67%) by 5.33 percentage points. Under our adaptation setup, Causal-JEPA achieves low rollout success on OGBench-Cube and OGBench-Scene; diagnostics on the latter show that its predicted slots tend toward a common mean representation.

3.3 Interventional Diagnostics of Learned Dependencies

We examine how structural choices affect the learned predictive mechanism using the four configurations in Table 2. We use two complementary diagnostics: (i) the normalized absolute change in robot and cube prediction error under removal of a fixed set of candidate nuisance slots, and (ii) the coordinate-wise RMSE of the predicted paired-action effect on cube position over contact-selected, action-effective samples. Appendix A.5 provides the complete evaluation protocols and metric definitions, including the removal operator, evaluated targets, normalization of ΔE, paired simulator interventions, contact-subset construction, affine physical readout, and reference baselines.

Combining the action prior with sparsity regularization yields an absolute prediction-error change of only 0.10% under the specified candidate-slot removal intervention, compared with 19.69–237.98% for the ablated configurations. The full model also achieves the lowest cube action-effect error on the selected contact samples: 17.18 mm versus 19.21–24.42 mm. These results indicate complementary roles for the action interface and sparse interaction learning: their combination yields greater error stability under the tested removal operator and more accurate prediction of paired-action effects on the cube. These mechanism-level diagnostics complement aggregate planning success by examining

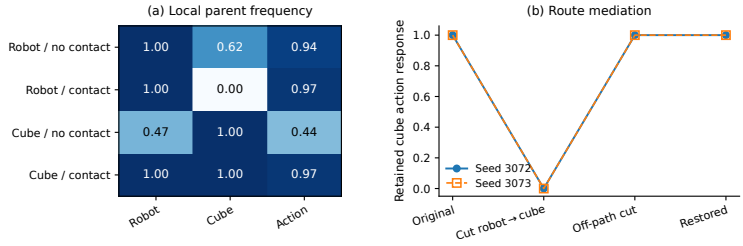


Figure 3: Local predictive dependencies and route mediation on OGBench-Cube. (a) Jacobian-witnessed parent frequencies within fixed-routing neighborhoods, averaged over two training seeds. Columns denote inputs; rows denote predicted outputs and contact conditions. (b) Normalized cube action responses in topology-eligible neighborhoods. Cutting the learned robot-to-cube connection eliminates the response; off-path cuts preserve it, and restoring the graph recovers it.

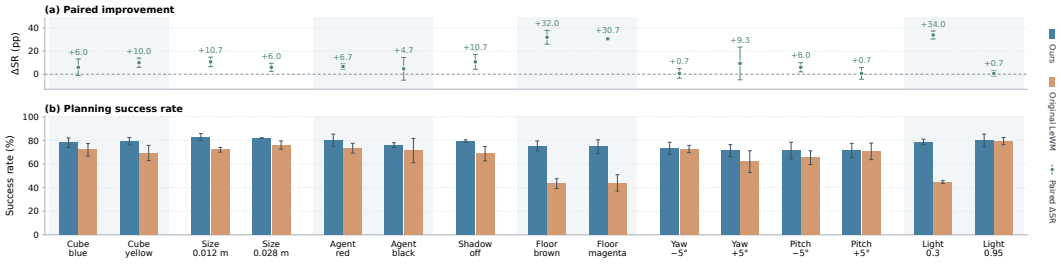


Figure 4: Planning performance under interventions on OGBench-Cube. (a) Paired success-rate differences (Ours − LeWM), measured in percentage points. (b) Mean success rates (%) of Ours and the original LeWM. Results are averaged over three evaluation seeds. Error bars denote sample standard deviations, computed from within-seed differences in (a) and per-seed success rates in (b).

how the learned predictor responds to controlled input and action interventions. The conclusions are restricted to the specified candidate slots, paired actions, contact-selected subset, and learned physical readout, and do not establish identification of the underlying physical causal graph.

3.4 Do learned routes constrain and mediate local dependencies?

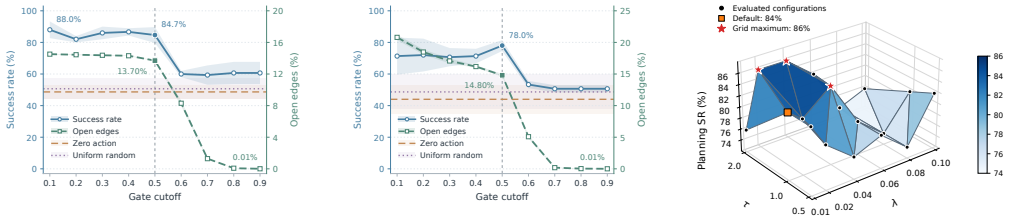
We examine local dependencies using the same 32 windows from independent episodes for each of two training seeds, with 16 contact and 16 non-contact windows. Within neighborhoods constructed to preserve the routing graph, we compute Jacobians of the fixed hard-routing predictor and confirm nonzero responses through single-input-block finite differences. The derivatives exclude the straight-through surrogate used during training.

Fig. 3(a) reveals state-associated, direction-dependent predictive dependencies. Robot inputs are witnessed parents of cube predictions in 46.9% of non-contact neighborhoods and 100% of contact neighborhoods, averaged across the two seeds. Conversely, cube inputs affect robot predictions in 62.5% of non-contact neighborhoods, with no witnesses in contact neighborhoods. These frequencies characterize local predictive dependence, which may arise through multiple communication hops.

We separately intervene on the routing mechanism while keeping the same pair of nearby actions within each neighborhood. Among neighborhoods satisfying prespecified topology criteria (22/32 and 23/32 for the two seeds), cutting the learned robot-to-cube connection eliminates the cube's local action response. Off-path cuts preserve this response, and restoring the original graph recovers it (Fig. 3(b)). These results identify learned routes that carry local action effects within the examined predictors. Full dependency classifications and numerical diagnostics are reported in Appendix B.

3.5 Planning Robustness under Visual Interventions

We evaluate fifteen visual interventions on OGBench-Cube (Fig. 4; protocols in Appendix D.1). Our model achieves higher mean success than the original LeWM across all conditions, with the



(a) OGBench-Cube: cutoff c (b) OGBench-Scene: cutoff c (c) λ–τ grid search

Figure 5: Gate-parameter ablations. (a,b) Planning success (blue, left axis) and open-edge fraction among admissible non-self connections (green, right axis), reported as mean ± 1 sample SD across three seeds. (c) Grid search on OGBench-Cube: black dots mark evaluated configurations, the orange square marks the default, and red stars mark the tied maxima.

largest gains under floor recoloring (30.7–32.0 percentage points) and reduced lighting (34.0 points). Because visual representations and transition architectures differ, this comparison does not isolate routing's contribution. Slot-based ablations (Fig. 12) yield mean success rates of 77.31%, 76.16%, and 76.64% for mask plus sparsity, sparsity alone, and dense slot attention, respectively, indicating modest planning differences. The nuisance-slot and action-effect diagnostics (Table 2), together with local route interventions, more directly characterize routing's role in the learned predictor without establishing physical causal identification.

3.6 Ablation Study

Sensitivity to the gate cutoff. Using fixed checkpoints, we sweep c from 0.1 to 0.9, activating admissible non-self connections when $p_{t,ij} \geq c$ (Figs. 5a and 5b). On OGBench-Cube, success and connectivity remain broadly stable for $c \leq 0.5$; the default retains 13.70% of admissible non-self edges. Higher cutoffs sharply reduce connectivity and lower success to approximately 60%, indicating sensitivity to excessive pruning.

Sensitivity to λ and τ. On OGBench-Cube, we fine-tune 18 (λ, τ) configurations for one additional epoch and evaluate each on 50 episodes using seed 43 and $c = 0.5$ (Fig. 5c). The highest success rate is 86%, compared with 84% for the same-budget default $(\lambda, \tau) = (0.01, 1)$, a difference of one successful episode.

Slot-to-slot routing We examine the role of action-to-slot routing by redirecting action inputs to slots representing the robot shadow or background while retaining learnable slot-to-slot routing (Table 3). The relatively strong performance of the Action $\rightarrow$ Shadow variant is expected, since the robot shadow is a dynamic visual factor that is strongly correlated with the robot pose and motion. The background-routing variants also retain nontrivial performance because the learned slot-to-slot graph can repurpose a background slot as an information relay, forming indirect paths such as action $\rightarrow$ background $\rightarrow$ robot/cube. Thus, their performance does not imply that background appearance is causally relevant to control, but rather reflects the model's ability to compensate through learned multi-hop routing. Nevertheless, among the completed epoch-matched variants, our physically grounded Action $\rightarrow$ Robot routing achieves the highest mean success rate of 84.67%.

Table 3: Ablation of action-to-slot routing on OGBench-Cube. All models are evaluated on the same clips. Results report planning success rate (%).

Action routing	Mean $\pm$ SD
Action $\rightarrow$ Background	78.00 $\pm$ 3.46
Action $\rightarrow$ Two Background Slots	78.67 $\pm$ 4.16
Action $\rightarrow$ Shadow + Background	81.33 $\pm$ 5.03
Action $\rightarrow$ Shadow	83.33 $\pm$ 4.16
Action $\rightarrow$ Robot (Ours)	84.67 $\pm$ 5.03

4 Related Work

Joint-embedding predictive architectures. Joint-embedding predictive architectures (JEPAs) learn representations through latent prediction rather than image or video reconstruction (He et al., 2022; Tong et al., 2022; Feichtenhofer et al., 2022). I-JEPA (Assran et al., 2023) predicts target-region embeddings from visible context, while data2vec (Baevski et al., 2022) learns contextualized latent targets across modalities. V-JEPA (Bardes et al., 2024) extends feature prediction to video, and V-JEPA 2.1 (Mur-Labadia et al., 2026) strengthens spatial and temporal grounding through dense supervision. Action conditioning connects latent prediction to decision making. V-JEPA 2 (Assran et al., 2025) combines video pretraining with action-conditioned post-training for robotic planning. DINO-WM (Zhou et al., 2025) learns dynamics over frozen DINOv2 features (Oquab et al., 2023), whereas LeWM (Maes et al., 2026) and PLDM (Sobal et al., 2026) jointly learn representations and dynamics. LeWM discourages collapse through SIGReg (Balestriero & LeCun, 2025); PLDM combines VICReg-inspired regularization (Bardes et al., 2022) with inverse-dynamics supervision. Latent organization provides complementary inductive biases. GeoWorld (Zhang et al., 2026a) introduces hyperbolic geometry, while Causal-JEPA (Nam et al., 2026) combines object-centric representations with selective object-level masking. Our work instead structures communication within object-centric predictors, combining a robot-mediated action prior with learned sparse gates to regulate action entry and information exchange among slots.

Local causality in world models. Local causal modeling captures dependencies that vary across regions of state–action space, permitting locally factored transitions even when global dynamics are densely coupled (Pitis et al., 2020). Research has explored both how to exploit these factorizations and how to learn them. CoDA uses locally independent transition components for counterfactual data augmentation (Pitis et al., 2020), while MoCoDA extends this approach to model-generated transitions and studies the benefits of local factorization for data efficiency and generalization to unseen states and actions (Pitis et al., 2022). To estimate local structure, ELDEN derives dependencies from the derivatives of learned dynamics and uses uncertainty in these dependencies to guide exploration (Wang et al., 2023). FCDL instead partitions the state–action space through quantization and learns sparse structures associated with the resulting contexts (Hwang et al., 2024). For object-centric representations, SPARTAN combines hard attention with sparsity regularization to learn context-dependent interactions (Lei et al., 2025). Building on this perspective, we combine a prescribed robot action interface with learned sparse routing and characterize local causal dependencies through input interventions on a fixed predictor. Our analysis distinguishes available communication paths from realized predictive dependencies within regions of unchanged routing, without claiming recovery of the underlying physical causal graph.

5 Conclusion

We presented an object-centric world model that separates interaction admissibility from selection through a robot-mediated action prior and learned sparse gates. Within fixed-routing neighborhoods, our analysis bounds local predictive dependencies by graph reachability. On OGBench-Cube, structural ablations show smaller prediction-error changes under candidate nuisance-slot removal and lower contact-conditioned action-effect error for the combined design. Local route interventions further identify connections that carry action responses. Together with the planning results, these findings support compact, intervention-tested transition modeling without implying recovery of the physical causal graph.

Limitations and future work. Our analysis concerns dependencies within the learned transition mechanism and does not establish that the learned routing recovers the underlying physical causal structure. Likewise, improved robustness under controlled interventions provides evidence of practical utility, but does not by itself demonstrate physical causal identification. Future work will investigate when model-level dependencies align with physical mechanisms through targeted physical interventions and evaluation in more diverse manipulation environments. Such efforts face a broader challenge in visual causal modeling: coupling among high-dimensional features can make distinct physical mechanisms difficult to disentangle in realistic settings.

AI USE STATEMENT

We used generative AI tools primarily to assist with manuscript drafting, language editing, literature review, and formatting. We also used these tools to help refine mathematical formulations and proof exposition, and to discuss the interpretation and scope of experimental findings. The authors reviewed and revised all AI-assisted content, including references, mathematical arguments, and empirical claims. We take full responsibility for the final manuscript, including the accuracy and integrity of all text, claims, and artifacts produced with AI assistance.

ETHICS STATEMENT

This work studies object-centric world models for visual planning in simulated environments. Although the proposed approach improves robustness under the tested distribution shifts, these results do not establish safety or reliability in real-world robotic deployment. Prediction errors and untested environmental changes could lead to unsafe actions, so practical deployment would require application-specific validation and appropriate safeguards. Our causal analysis concerns dependencies within the learned transition mechanism and should not be interpreted as establishing recovery of physical causal relations.

REPRODUCIBILITY STATEMENT

We describe the model architecture, structural constraints, and training objective in Section 2. Appendix A provides implementation details and describes slot-encoder pretraining, gate optimization, and interventional evaluation protocols. Tables 4 and 5 summarize task-specific configurations, hyperparameters, hardware, and planning settings. Section 3.1 and Table 1 identify the benchmarks and distinguish reproduced baselines from results taken from prior work. For our main planning experiments, we report mean success rates and sample standard deviations over three evaluation seeds. Appendix B details the local dependency diagnostics, including test-window selection, fixed-routing neighborhood construction, numerical thresholds, and route interventions. The assumptions underlying our theoretical results are stated in Section 2, with proofs provided in Appendix C. Appendix D.1 specifies the visual intervention settings, and Appendix E provides the training pseudocode.

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, 2023. URL https://arxiv.org/abs/2301.08243.

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. URL https://arxiv.org/abs/2506.09985.

Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. data2vec: A general framework for self-supervised learning in speech, vision and language, 2022. URL https://arxiv.org/abs/2202.03555.

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

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

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

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation, 2013. URL https://arxiv.org/abs/1308.3432.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion, 2024. URL https://arxiv.org/abs/2303.04137.

Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67, 2005.

John Ellson, Emden Gansner, Lefteris Koutsofios, Stephen C. North, and Gordon Woodhull. Graphviz— open source graph drawing tools. In Petra Mutzel, Michael Jünger, and Sebastian Leipert (eds.), *Graph Drawing*, pp. 483–484, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg. ISBN 978-3-540-45848-7.

Christoph Feichtenhofer, Haoqi Fan, Yanghao Li, and Kaiming He. Masked autoencoders as spatiotemporal learners. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=UaXD4Al3mdb.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15979–15988, 2022. doi: 10.1109/CVPR52688.2022.01553.

Inwoo Hwang, Yunhyeok Kwak, Suhyung Choi, Byoung-Tak Zhang, and Sanghack Lee. Fine-grained causal dynamics learning with quantization for improving robustness in reinforcement learning. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Anson Lei, Bernhard Schölkopf, and Ingmar Posner. SPARTAN: A sparse transformer world model attending to what matters. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=uS5ch7GjZ4.

Yushan Liu, Peibo Sun, Shoujie Li, Yifan Xie, Lingfeng Zhang, Xintao Chao, Shiyuan Dong, Fang Chen, Xiao-Ping Zhang, and Wenbo Ding. Oa-wam: Object-addressable world action model for robust robot manipulation, 2026. URL https://arxiv.org/abs/2605.06481.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworldmodel: Stable end-to-end joint-embedding predictive architecture from pixels, 2026. URL https://arxiv.org/abs/2603.19312.

Lorenzo Mur-Labadia, Matthew Muckley, Amir Bar, Mido Assran, Koustuv Sinha, Mike Rabbat, Yann LeCun, Nicolas Ballas, and Adrien Bardes. V-jepa 2.1: Unlocking dense features in video self-supervised learning, 2026. URL https://arxiv.org/abs/2603.14482.

Heejeong Nam, Quentin Le Lidec, Lucas Maes, Yann LeCun, and Randall Balestriero. Causal-jepa: Learning world models through object-level latent masking, 2026. URL https://arxiv.org/abs/2602.11389.

Shukrullo Nazirjonov, Sai Prasanna, Anna Manasyan, and Georg Martius. Better slots, better worlds: Representation quality & robustness in object-centric world models, 2026. URL https://arxiv.org/abs/2608.12078.

Yosuke Nishimoto and Takashi Matsubara. Object-centric world models for causality-aware reinforcement learning, 2026. URL https://arxiv.org/abs/2511.14262.

Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023.

Muthukumar Pandaram, Jakob Hollenstein, David Drexel, Samuele Tosatto, Antonio Rodríguez-Sánchez, and Justus Piater. Dynamic sparsity: challenging common sparsity assumptions for learning world models in robotic reinforcement learning benchmarks. In *Proceedings of the Fortieth AAAI Conference on Artificial Intelligence and Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence and Sixteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'26/IAAI'26/EAAI'26. AAAI Press, 2026. ISBN 978-1-57735-906-7. doi: 10.1609/aaai.v40i29.39658. URL https://doi.org/10.1609/aaai.v40i29.39658.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. OGBench: Benchmarking offline goal-conditioned RL. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=M992mjgKzI.

Silviu Pitis, Elliot Creager, and Animesh Garg. Counterfactual data augmentation using locally factored dynamics. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Silviu Pitis, Elliot Creager, Ajay Mandlekar, and Animesh Garg. Mocoda: model-based counterfactual data augmentation. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos, 2024. URL https://arxiv.org/abs/2408.00714.

Maximilian Seitzer, Max Horn, Andrii Zadaianchuk, Dominik Zietlow, Tianjun Xiao, Carl-Johann Simon-Gabriel, Tong He, Zheng Zhang, Bernhard Schölkopf, Thomas Brox, and Francesco Locatello. Bridging the gap to real-world object-centric learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=b9tUk-f_aG.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3, 2025. URL https://arxiv.org/abs/2508.10104.

Vlad Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, Tim G. J. Rudner, and Yann LeCun. Stress-testing offline reward-free reinforcement learning: A case for planning with latent dynamics models. In *7th Robot Learning Workshop: Towards Robots with Human-Level Abilities*, 2025. URL https://openreview.net/forum?id=jON7H6A9UU.

Vlad 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 *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=hTZ0SJCGQX.

Jonathan Spieler, Angel Villar-Corrales, and Sven Behnke. Slot-mpc: Goal-conditioned model predictive control with object-centric representations, 2026. URL https://arxiv.org/abs/2605.14937.

Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: masked autoencoders are data-efficient learners for self-supervised video pre-training. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Rodion Vakhitov, Leonid Ugadiarov, Alexey Skrynnik, and Aleksandr Panov. Causal object-centric models for planning with monte carlo tree search, 2026. URL https://arxiv.org/abs/2606.14418.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

Zizhao Wang, Jiaheng Hu, Peter Stone, and Roberto Martín-Martín. Elden: exploration via local dependencies. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Xi Zeng, Haojie Ren, and Ziying Song. Phylatent: Learning dynamics-relevant representations for jepa world models, 2026. URL https://arxiv.org/abs/2608.05720.

Zeyu Zhang, Danning Li, Ian Reid, and Richard Hartley. Geoworld: Geometric world models, 2026a. URL https://arxiv.org/abs/2602.23058.

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, 2026b. 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, 2026. 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, 2025. URL https://arxiv.org/abs/2411.04983.

A Experimental Details

A.1 Implementation Details

We summarize the implementation and evaluation settings used in our experiments. Table 4 lists the task-specific slot counts and dimensions, while Table 5 reports the shared architecture, optimization, gating, and planning parameters. Unless otherwise stated, these settings are used throughout the experiments.

Benchmark for analysis We conduct diagnostic and ablation experiments mainly on OGBench-Cube. Its well-defined manipulation task and interpretable action–robot–object interaction chain align with our modeling assumptions and support controlled interventions and systematic comparisons of our design. Extending these diagnostics to other benchmarks requires adapting the intervention protocols and evaluation criteria to their task structures. This focus concerns the diagnostic setting, and the method itself is evaluated across multiple benchmarks.

Table 4: Task-specific slot configurations.

Task	Number of slots	Slot dimension
Two-Room	4	192
PushT	4	192
OGBench-Cube	8	256
OGBench-Scene	8	256

Table 5: Shared implementation, world-model training, and evaluation settings. Task-specific slot configurations are listed in Table 4.

Setting	Value
Implementation and architecture	
Framework	PyTorch
Hardware	4 NVIDIA L40 GPUs
Precision	bfloat16 mixed precision
Observation resolution	224×224
Visual backbone	Pretrained DINOv3 Siméoni et al. (2025)
Slot Attention refinement iterations	3
Transformer layers / attention heads	6 / 16
World-model optimization	
Training epochs	10
Global batch size	128
Optimizer	AdamW Loshchilov & Hutter (2019)
Learning rate	5×10^{-5}
Weight decay	10^{-3}
Learning-rate schedule	1% linear warmup followed by cosine decay
Loss weights: prediction / SIGReg / sparsity	1 / 0.09 / 0.01
Sparse gates	
Gate decisions	Deterministic hard thresholding
Gradient surrogate	Soft-attention surrogate
Default cutoff	0.5
Planning and evaluation	
Planner	CEM De Boer et al. (2005)
CEM candidates / iterations / elites	300 / 30 / 30
Planning horizon	5 blocks of 5 control actions (25 steps)
Replanning interval	25 control steps
Evaluation budget	50 control steps
Evaluation seeds	3
Start–goal pairs per seed	50 paired samples
Reported statistics	Mean success rate and sample standard deviation across seeds

A.2 Slot attention learning

Backbone We pretrain our slot attention encoder by extending DINOSAUR with a temporal consistency loss. A frozen DINO backbone Caron et al. (2021); Oquab et al. (2023); Siméoni et al. (2025) extracts patch features from each video frame, which Slot Attention groups into object-centric slots. Pretraining combines two complementary objectives: a feature reconstruction loss $\mathcal{L}_{\text{recon}}$, which encourages the slots to retain scene information, and a temporal consistency loss $\mathcal{L}_{\text{consist}}$, which encourages consistent object representations across frames. Together, these objectives promote informative and temporally coherent slots for subsequent dynamics learning.

Slot visualization Fig. 7 visualizes the scene decomposition produced by our pretrained slot encoder. Different slots capture the controlled agent, task objects, and other visual regions, including backgrounds and shadows. These representations provide interaction units for the dynamics predictor. Identified robot or agent slots define the action interface, while dependencies among slots are learned through the gated transition model.

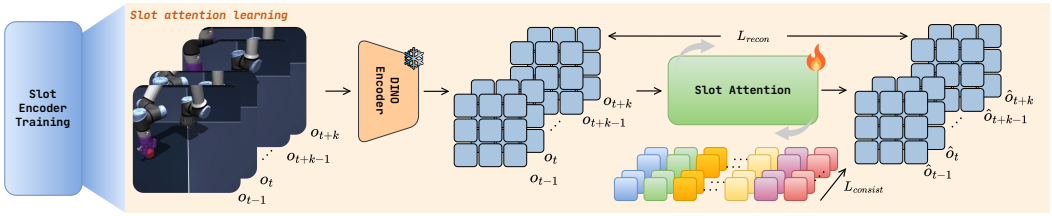


Figure 6: Pretraining the slot attention encoder. Slot Attention groups features from a frozen DINO encoder into object-centric slots, trained jointly through feature reconstruction ($\mathcal{L}_{\text{recon}}$) and temporal consistency ($\mathcal{L}_{\text{consist}}$). Snowflake and flame icons indicate frozen and trainable modules, respectively.

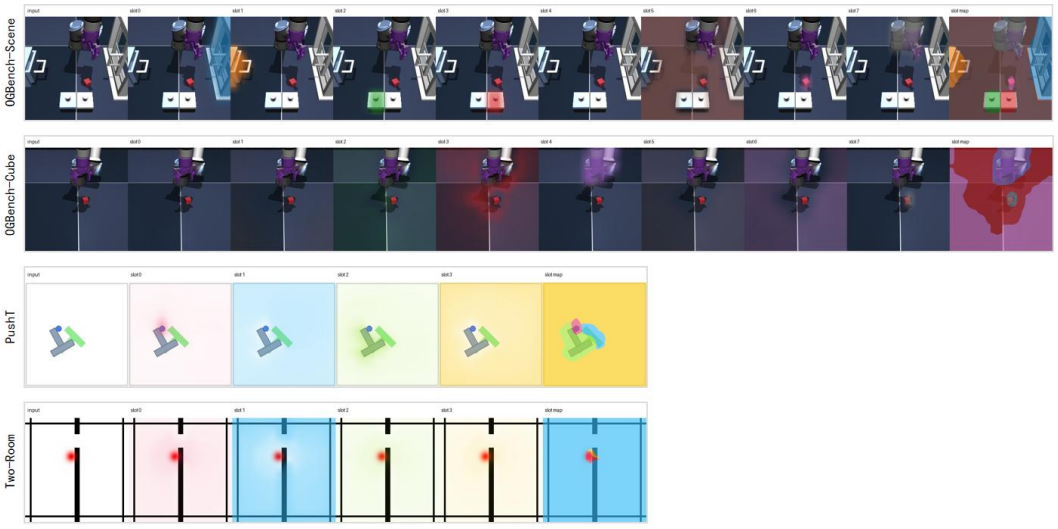


Figure 7: Decoded slot masks across four benchmarks. Each row shows the input observation, individual slot-mask overlays, and the combined slot map. We use eight slots for OGBench-Scene and OGBench-Cube and four for PushT and Two-Room. Colors distinguish slots within each example.

A.3 Decoded Images

Visualization We train an auxiliary RGB decoder while keeping the world model frozen, using slot embeddings produced by the encoder and projector as inputs and the corresponding RGB observations as reconstruction targets. For PushT and OGBench-Scene, we additionally use convolutional refinement and multiscale foreground and edge losses. During open-loop rollout, the predictor and prediction projector autoregressively generate future embeddings from a single history frame, conditioned on the action sequence. The decoder converts these embeddings into 224×224 imagined images over 25 prediction steps. Ground-truth future frames are used only for comparison and do not participate in generation. Fig. 8 presents results on OGBench-Scene, OGBench-Cube, PushT, and Two-Room, enabling qualitative assessment of scene configurations and spatial deviations over the rollout horizon.

A.4 Attention-Level Straight-Through Estimation

Hard thresholding provides no useful gradient for optimizing the gate scores. We therefore use an attention-level straight-through estimator (Bengio et al., 2013), as defined in Eq. equation 17. The correction $A^{\text{soft}} - \text{sg}(A^{\text{soft}})$ evaluates to zero during the forward pass, so A^{train} equals A^{hard} and predictions follow the binary routing pattern. During backpropagation, the stop-gradient term contributes no derivative, allowing the soft branch to provide surrogate gradients for gate learning.

The soft surrogate weights attention by continuous gate values and normalizes over all admissible connections, including those currently closed by hard thresholding. Such connections can therefore receive gate gradients and potentially reopen during training. Self-gates remain fixed, and connections prohibited by the physical mask remain excluded. Attention scores are detached only when constructing A^{soft}, so the surrogate differentiates through the gate weights without introducing gradients through its attention-score computation. Meanwhile, A^{hard} retains the usual gradients through active attention connections. This estimator supplies an optimization surrogate rather than the exact derivative of the discrete routing decisions. At inference, the soft branch is discarded and only hard attention is used.

A.5 Interventional diagnostic protocals

Nuisance-slot removal We evaluate removal sensitivity on 1,024 fixed prediction windows from 128 independently generated episodes, shared across all configurations. Using zero-based indices, the candidate removal set is $\mathcal{N} = \{0, 1, 2, 3, 5, 6\}$, while the evaluated targets are $\mathcal{T} = \{4, 7\}$, corresponding to the robot and cube. The candidate indices are fixed using archived visual-role assignments, rather than selected from the learned gates. Slots $\{0, 1, 2, 5, 6\}$ are background-appearance candidates, and slot 3 was initially designated as a shadow candidate. These assignments do not establish ground-truth non-parent status: slot 3 exhibits robot overlap, and candidate features may encode robot or cube state information.

Removal is implemented as zeroing plus attention isolation. The selected projected slot vectors are set to zero in all three history frames, and all incoming and outgoing attention connections involving these tokens are disabled in every predictor layer, except their self-loops. Token indices and positional encodings are preserved. Thus, the operation does not delete or reindex tokens, and does not replace them with mean features. Actions, retained slot inputs, and ground-truth targets remain unchanged. For configurations with learned gates, gates are recomputed from the intervened inputs under the modified attention mask; the original routing graph is not forcibly retained.

Only the robot and cube predictions are scored. For sample n, define the clean prediction error as

$$e_n = \frac{1}{|\mathcal{T}|d} \sum_{i \in \mathcal{T}} \left\| \widehat{z}_n^i - z_n^i \right\|_2^2, \qquad d = 256, \tag{18}$$

where z_n^i is the projected representation of the true next observation. Let $e_n^{-\mathcal{N}}$ denote the corresponding error after removal, using the same targets. The reported metric is

$$\Delta E = 100 \, \frac{\sum_{n=1}^{N} |e_n^{-\mathcal{N}} - e_n|}{\sum_{n=1}^{N} e_n + N\epsilon}, \qquad \epsilon = 10^{-8} \frac{1}{|\mathcal{T}|} \sum_{i \in \mathcal{T}} v_i, \tag{19}$$

where v_i is the mean per-coordinate variance of projected slot i on the calibration fitting set, floored at 10^{-8}. The absolute difference is taken per sample before aggregation. The denominator is the aggregate clean error, not each sample's error separately; therefore, this metric is not the mean of per-sample percentage changes. Both improvements and degradations contribute positively to ΔE. A small value indicates error stability under this particular removal operator, rather than establishing physical irrelevance of the removed slots.

Paired action interventions and contact samples The action-effect metric compares two action sequences starting from the same simulator state. Each sequence contains five consecutive primitive controls. The factual sequence A is taken from the expert trajectory. The alternative A' negates the XYZ components of every control while preserving its yaw and gripper components, before dataset standardization. It is therefore a two-action counterfactual contrast, not a contrast against a zero-action rollout.

Both branches restore the same complete simulator and controller state, including the random-number-generator state. Factual continuation and repeated execution are checked for pixel equality and state discrepancies below 10^{-9}. The model receives the same three-frame observation history and past actions for both predictions; only the current five-control action block changes. The prediction horizon is one world-model transition, corresponding to five primitive simulator steps.

The reported contact subset contains 128 accepted pairs from distinct episodes, selected without using model predictions. At the initial state, the environment's `proprio/gripper_contact`

score must exceed 0.8. This score is a right-gripper-pad external-wrench proxy, $\mathrm{clip}(\|w_{\mathrm{right_pad}}\|_2/50, 0, 1)$. We additionally require the paired simulator outcomes to differ by at least 10 mm in end-effector position and 5 mm in cube position. Thus, "contact samples" denotes a contact-proxy-selected, action-effective subset; it is distinct from the geometric-contact labels used in the local-neighborhood diagnostics.

Physical readout and action-effect error For each model m, we fit a separate affine ridge readout R_m from its projected cube-slot features to world-frame XYZ coordinates in meters. Features are standardized using 4,000 calibration fitting frames; 1,000 additional frames select the ridge coefficient from $\{0.01, 0.1, 1, 10, 100, 1000\}$, and a further 1,000 frames audit readout accuracy. These calibration frames are sampled from the original Cube dataset and are separate from the newly generated evaluation trajectories. The readout is fixed during evaluation and does not modify the world model. No RGB decoder is used to obtain the millimeter-valued metric.

Let $\mathbf{p}_n^A$ and $\mathbf{p}_n^{A'}$ be the true terminal cube positions from the paired simulator branches. The reference and predicted effects are

$$\delta\mathbf{p}_n = \mathbf{p}_n^{A'} - \mathbf{p}_n^A, \qquad \widehat{\delta\mathbf{p}}_n = R_m(\widehat{z}_{n,c}^{A'}) - R_m(\widehat{z}_{n,c}^{A}), \tag{20}$$

where $c = 7$ denotes the cube slot. The reported action-effect error is

$$E_{\mathrm{effect}} = 1000\sqrt{\frac{1}{3N_c}\sum_{n=1}^{N_c}\left\|\widehat{\delta\mathbf{p}}_n - \delta\mathbf{p}_n\right\|_2^2}, \qquad N_c = 128. \tag{21}$$

This is coordinate-wise RMSE pooled over samples and XYZ coordinates, converted from meters to millimeters. It is neither an absolute terminal-position error nor a mean Euclidean distance error.

Effect scale and reference baselines On the fixed contact subset, the mean Euclidean magnitude of the true counterfactual cube effect is 155.20 mm. A zero-effect predictor, which outputs $\widehat{\delta\mathbf{p}}_n = \mathbf{0}$ for every pair, obtains

$$E_{\mathrm{zero}} = 1000\sqrt{\frac{1}{3N_c}\sum_{n=1}^{N_c}\|\delta\mathbf{p}_n\|_2^2} = 96.06\ \mathrm{mm}. \tag{22}$$

This is a zero-response baseline, not an additional simulated zero-action branch. Performance below this baseline demonstrates useful action-effect prediction on the selected cases, but does not by itself establish recovery of physical causal structure.

To assess the representation/readout contribution, we also apply the same readout to the encoded true future observations from both branches, bypassing the transition predictor. For the seed-3072 checkpoints, these oracle-representation effect errors are 22.55, 22.86, 22.76, and 23.13 mm for configurations (a)–(d), respectively. They are diagnostic references rather than guaranteed lower bounds: the reported action-effect error reflects both transition prediction and the accuracy of the learned coordinate readout.

B Local Dependency Diagnostics

Evaluation setup. We evaluate the epoch-9 checkpoints for training seeds 3072 and 3073 on fixed test trajectories generated independently of training. Both models use the same 32 windows from independent episodes, comprising 16 contact and 16 non-contact windows selected solely using simulator geometry labels. The intervened inputs comprise eight current slot blocks and a 25-dimensional normalized action block; the history is held fixed. We examine the robot and cube predictions, giving $32 \times 2 \times 9 = 576$ input–output relations per model. These counts refer to relations across neighborhoods, not independent episodes. Because the predictor permits information exchange only within the same time block, we evaluate the final block separately while retaining its original positional encodings, and check its outputs and binary routing against the full-history forward pass.

Fixed-routing neighborhoods. For each selected input b, we construct an open convex product neighborhood

$$\mathcal{U} = \prod_{j \in \mathcal{I}} B(b_j, r_j), \qquad r_j = \rho / C_j, \tag{23}$$

where C_j bounds the Lipschitz constant of the corresponding token encoder. The slot intervention boundary is after the visual encoder and learned projector, but before the predictor's tokenwise slot projection. This projection is the identity in both evaluated models, giving $C_j = 1$ for all eight slot inputs. Action interventions are applied after normalization and before the action encoder; the predictor's action projection is also included in its encoding bound. Positional encodings remain fixed.

This choice bounds each encoded-token perturbation by ρ. We bound the action encoder using products of layer operator norms and a conservative global SiLU derivative bound of 1.5. For a gate LayerNorm with dimension $d = 256$, centered RMS σ_j at the reference token, and scale parameter γ, we use the bound

$$L_{\mathrm{LN},j}(\rho) \le \frac{\|\gamma\|_\infty}{\sqrt{\max(\sigma_j - \rho/\sqrt{d}, 0)^2 + \epsilon}}. \tag{24}$$

Combining this bound with the query and key projection norms gives perturbation bounds $B_{Q,i}$ and $B_{K,j}$. Writing a gate logit as $\ell_{ij} = \alpha q_i^\top k_j + \beta$, where α includes attention scaling and temperature and β is an input-independent offset, yields

$$|\Delta \ell_{ij}| \le \alpha \left(\|q_i\| B_{K,j} + \|k_j\| B_{Q,i} + B_{Q,i} B_{K,j} \right). \tag{25}$$

We choose ρ so that these bounds are strictly below half of the corresponding threshold margins for all gated admissible connections. Specifically, centers with minimum admissible margin at most 10^{-12} are rejected; none were rejected in this experiment. We perform 55 bisection iterations over $\rho \in [0, 0.1]$, retaining the feasible lower endpoint, and multiply that endpoint by 0.99 to provide the additional safety margin.

For seed 3072, the minimum, median, and maximum slot radii are $(3.623 \times 10^{-5}, 9.692 \times 10^{-3}, 3.131 \times 10^{-2})$, and the corresponding action radii are $(6.818 \times 10^{-7}, 1.824 \times 10^{-4}, 5.894 \times 10^{-4})$. For seed 3073, the corresponding triples are $(8.254 \times 10^{-4}, 7.493 \times 10^{-3}, 3.995 \times 10^{-2})$ for slots and $(1.347 \times 10^{-5}, 1.222 \times 10^{-4}, 6.517 \times 10^{-4})$ for the action block.

Thus, neighborhood construction uses analytical bounds over the entire product region, rather than only checking sampled endpoints. The bounds are evaluated conservatively in floating-point arithmetic and are not certified by interval arithmetic. The neighborhoods lie in representation and normalized-action space; their points need not correspond to physically realizable observations.

Dependency classification and derivative checks. Within each neighborhood, we compute the full Jacobian blocks of the robot and cube predictions with respect to all nine inputs at the center and two interior points. We differentiate the actual hard-routing predictor in float64 and evaluation mode, excluding the straight-through surrogate used during training. Forced float32 casts in the action encoder and gate computation are disabled for this diagnostic. Within a region of fixed routing, the binary gates have zero derivative. At each center, we require the maximum absolute difference from the original float32 full-history predictions to be below 2×10^{-5}, with identical binary routing.

For each model, a NumPy default random generator initialized with seed 2026092513 is consumed sequentially. Each interior point independently displaces every input block by $(r_j/4)v_j$, where v_j is obtained by normalizing a standard Gaussian vector in that block's dimension. For finite differences, a fresh direction is drawn for each point and input block and reused across the three step sizes

$$h \in \{0.5 r_j,\ 0.1 r_j,\ 0.02 r_j\}. \tag{26}$$

Only the selected input block is perturbed. All endpoints remain within the constructed neighborhood: their displacement from its center is at most $0.75 r_j$ in the perturbed block. We explicitly verify that their binary routing remains unchanged.

Let $J_{ij}(x) = \partial F_{\theta,i}(x) / \partial b_j$. The normalized block sensitivity is

$$S_{ij}(x) = \frac{\nu_j}{16 \nu_i} \|J_{ij}(x)\|_{\mathrm{F}}. \tag{27}$$

For slot inputs, the calibration scales are

$$\nu_j^2 = \max\left\{\frac{1}{256}\sum_{k=1}^{256}\mathrm{Var}_{\mathrm{cal}}(z_k^j), 10^{-8}\right\}, \qquad \nu_a = 1. \tag{28}$$

Population variances are estimated from 4,000 calibration frames sampled from the original Cube dataset, using each model's projected features. The divisor $16 = \sqrt{256}$ normalizes the output dimension for every input, including the 25-dimensional action block.

Six-layer Boolean reachability provides the structural support bound in Theorem 1. We classify each input–output relation as structurally excluded if no route of at most six hops exists, numerically witnessed if the maximum $S_{ij}(x)$ over the three sampled points exceeds the detection threshold, and unresolved if it is reachable but has no detected witness. A zero derivative at the sampled points alone is not used to establish absence of dependence. The default normalized block-norm threshold is 10^{-6}; we also report sensitivity at 10^{-8} and 10^{-4}.

Writing $\Delta_h F_i = F_{\theta,i}(x^+) - F_{\theta,i}(x^-)$ for endpoints differing only in block j by $\pm h v_j$, the relative Jacobian-vector-product discrepancy is

$$\eta_{ij}(x,h) = \frac{\|\Delta_h F_i/(2h) - J_{ij}(x)v_j\|_2}{\max\{\|J_{ij}(x)v_j\|_2, 10^{-12}\}}. \tag{29}$$

The implementation records this discrepancy without imposing a separate prespecified pass/fail threshold; the observed maximum is 9.053×10^{-6}. A finite-response witness requires

$$\|\Delta_h F_i\|_2 > 10^{-11} + 10^{-10}\|F_{\theta,i}(x)\|_2. \tag{30}$$

Relation-level finite-difference witnesses are accumulated across the three points and three step sizes. We additionally perturb all unreachable inputs jointly while holding reachable inputs fixed to check the local factorization predicted by the theorem.

Numerical results. Table 6 summarizes the classifications. Every numerically witnessed relation is corroborated by at least one finite-difference response above the detection threshold, and classifications are unchanged across the three Jacobian thresholds. Unreachable relations have zero Jacobian blocks and zero finite responses in the diagnostic computation. Joint perturbations of unreachable inputs also leave the examined outputs numerically unchanged. Fig. 9 reports parent-witness frequencies for all nine input blocks, separately for each seed and contact condition. These are frequencies on the selected contact-balanced test windows, rather than estimates of their prevalence under a natural rollout distribution.

Table 6: Local dependency classifications over 32 neighborhoods per seed. Each model contributes 576 input–output relations. The last two columns report maxima over structurally excluded relations; zeros refer to the diagnostic computation.

Seed	Excluded	Witnessed	Unresolved	Max. Jacobian	Max. response
3072	426	150	0	0	0
3073	425	151	0	0	0

Route interventions and sample eligibility. We separately intervene on the routing mechanism to examine whether selected learned connections carry local action responses. For each eligible neighborhood, the same pair of nearby actions is used under four routing conditions: the original graph, removal of the learned robot-to-cube connection, an off-path cut, and restoration of the original graph. Other input blocks and model parameters remain fixed. Eligibility is determined from the routing topology before comparing responses, yielding 22 of 32 neighborhoods for seed 3072 and 23 of 32 for seed 3073.

Using zero-based indices, the robot, cube, and action tokens are 4, 7, and 8. Graph entries use query–key order: $g_{qk} = 1$ denotes the connection $k \to q$. Let g^- equal g with entry $(7, 4)$ set to zero. Eligibility requires

$$g_{7,4} = 1, \qquad (g^{[6]})_{7,8} = 1, \qquad ((g^-)^{[6]})_{7,8} = 0, \tag{31}$$

together with the existence of an off-path control. Thus, the original action-to-cube route must exist, and no route of at most six hops may remain after removing robot-to-cube.

The control is the first open edge (q, k) satisfying $(g^{[9]})_{7,q} = 0$. Candidates are scanned first as $(q, 4)$ for $q = 0, 1, 2, 5, 6$, then lexicographically over $q = 0, \ldots, 7$ and $k = 0, \ldots, 8$, excluding $q = k$. With self-loops, the nine-hop power supplies transitive reachability, so the control edge's receiver cannot reach the cube. Each control cut starts from the original graph. Each cut modifies the shared graph in all six predictor layers; every other binary gate and self-loop is held fixed, while attention is renormalized over the surviving connections.

For seed 3072, eight neighborhoods are excluded because robot-to-cube is closed, and two because that edge is open but the original action-to-cube path is absent. For seed 3073, all nine exclusions have robot-to-cube closed. No further exclusions arise from surviving alternative paths or missing controls. These decisions do not use measured response magnitudes.

For each neighborhood, we draw a normalized Gaussian direction v_a in the 25-dimensional action space and construct the one-sided pair

$$a' = a + \frac{r_a}{4} v_a. \tag{32}$$

The same pair (a, a') is used under all four routing conditions, with the slot inputs unchanged. Both endpoints lie inside the fixed-routing neighborhood.

For routing condition m, the response of target i in neighborhood n is the raw projected-feature difference

$$d_{n,i}^{(m)} = \| F_i^{g_n^{(m)}}(b'_n) - F_i^{g_n^{(m)}}(b_n) \|_2. \tag{33}$$

Responses are not divided by action-step magnitude or feature variance. We first aggregate over the eligible set $\mathcal{E}$ and then normalize by the original-graph response:

$$R_i^{(m)} = \frac{\sum_{n \in \mathcal{E}} d_{n,i}^{(m)}}{\max\{\sum_{n \in \mathcal{E}} d_{n,i}^{(\text{original})}, 10^{-30}\}}. \tag{34}$$

This is a ratio of sums, not an average of per-neighborhood ratios. No response-based sample exclusion is used. Every eligible original cube response is strictly positive, so the denominator floor is inactive for the reported results.

Interpretation. The input interventions characterize dependencies of a fixed predictor, whereas route interventions modify its computational mechanism. Together, these diagnostics examine the support of local predictive dependencies and whether selected routes carry action responses within the evaluated models. They provide numerical checks of the implemented mechanism, not a replacement for the theorem's proof or evidence of recovery of physical causal relations.

Visualization of Learned Routing Fig. 10 visualizes the active routing connections at selected steps of planning rollouts on OGBench-Cube. We compare four configurations to examine how the physical attention mask and sparsity regularization jointly shape information flow.

Without either component, the routing graph is fully connected. Sparsity regularization alone reduces connectivity but retains direct action connections to non-robot slots. The physical mask enforces the prescribed robot interface, but routing remains nearly dense without sparsity regularization. Combining both components yields compact, robot-centered routing. Only the action interface is prescribed, and object–object connections are learned without graph supervision.

These visualizations describe direct communication within each prediction step. Successive Transformer layers permit multi-hop propagation, so predictive dependencies can extend beyond individual edges. Conversely, an active edge does not necessarily produce a nonzero effect on the prediction. The local input-intervention analysis distinguishes these available routes from the dependencies actually realized by the learned predictor.

C Proofs and Additional Analysis

C.1 Local Causality of the Learned Transition

This section proves Theorem 1 and discusses its implications for the learned transition mechanism. We adopt the notation and assumptions stated in the theorem. Throughout, the predictor is fixed, and interventions modify individual input blocks while leaving the remaining inputs unchanged.

Proof of Theorem 1. We first establish local constancy of the routing graph. At the input under consideration, continuity of the gate scores and their nonzero margins from the thresholds imply a neighborhood in which all hard gates retain their values. Choose an open product neighborhood $\mathcal{U}$ inside this region, and denote the fixed routing pattern by g.

We next trace input dependence through the predictor. Define $\mathcal{S}_i^{(0)} = \{i\}$ and

$$\mathcal{S}_i^{(\ell+1)} = \bigcup_{k:g_{ik}=1} \mathcal{S}_k^{(\ell)}.$$

By induction, token i at layer ℓ depends only on inputs indexed by $\mathcal{S}_i^{(\ell)}$. Self-loops account for query and residual dependencies, while exact masking excludes closed keys from attention normalization. Tokenwise operations introduce no additional input dependencies. Thus, after L layers, the prediction depends only on inputs in $\mathcal{S}_i^{(L)} = \mathcal{P}_i(g)$, establishing Eq. equation 12.

Finally, we establish the factorization and its minimality. Take two inputs in $\mathcal{U}$ that agree on $\mathrm{Pa}_i^{\mathcal{U}}$ and replace their remaining coordinates one at a time. Every intermediate input remains in the product neighborhood, and each replacement preserves the output by Definition 2. The prediction is therefore determined by the parent inputs alone, making the factorization in Eq. equation 13 well-defined. Any determining input set must contain every member of $\mathrm{Pa}_i^{\mathcal{U}}$: otherwise, the intervention witness for an omitted parent gives two inputs that agree on the determining set but produce different outputs. This proves minimality and uniqueness. □

Robot-mediated action effects. Under the action prior, every path from the action input to a non-robot prediction in the layer-expanded routing graph passes through a robot token. Theorem 1 therefore implies that any nonzero local action effect on a non-robot prediction within $\mathcal{U}$ must propagate through robot tokens.

Interpretation and scope. The theorem bounds local causal support by the learned routing structure; an open edge does not by itself establish a nonzero effect. This support is defined over vector-valued slot blocks, so sparse connections between slots are compatible with dense dependencies within each block. Locality refers to regions where the hard routing pattern remains unchanged; interventions crossing gate thresholds may activate or deactivate routes. The result characterizes the implemented transition mechanism and does not establish recovery of physical causal relations.

Differential characterization. When the predictor is smooth within such a neighborhood, the intervention-based definition can also be expressed through block Jacobians.

Corollary 1 (Differential characterization)**.** Let $\mathcal{U}$ be an open, convex product neighborhood on which the routing graph is constant. If F_θ is continuously differentiable on $\mathcal{U}$, then

$$j \in \mathrm{Pa}_i^{\mathcal{U}} \iff \exists b \in \mathcal{U} : \frac{\partial F_{\theta,i}(b)}{\partial b_j} \neq 0. \tag{35}$$

Proof. A nonzero block Jacobian yields a direction in which a sufficiently small intervention changes the prediction. Since $\mathcal{U}$ is open, this intervention remains within $\mathcal{U}$, establishing membership in $\mathrm{Pa}_i^{\mathcal{U}}$. Conversely, take an intervention witness and connect its two inputs by the coordinate line segment contained in $\mathcal{U}$. The fundamental theorem of calculus implies that their nonzero output difference requires a nonzero block Jacobian somewhere along this segment. □

Table 7: Visual interventions in OGBench-Cube. Seven intervention families comprise fifteen evaluated variants. Each variant is applied separately.

Level	Intervention	Original	Intervened values	Variants
Object-level	Cube color	Crimson	Blue; yellow	2
	Cube visual size	$0.020\,\mathrm{m}$	$0.012\,\mathrm{m}$; $0.028\,\mathrm{m}$	2
	Agent color	Purple	Red; black	2
Scene-level	Robot shadow	On	Off	1
	Floor color	Slate	Brown; magenta	2
	Camera angle offset	$(0^\circ, 0^\circ)$	$(\pm 5^\circ, 0^\circ)$; $(0^\circ, \pm 5^\circ)$	4
	Global light intensity	0.7	0.3; 0.95	2
			Total	15

Notes. All interventions affect rendering only. Cube size denotes the MuJoCo box half-extent; collision geometry, mass, and inertia remain unchanged. Cube color and visual size also apply to the target marker when visible. Agent color modifies the Robotiq gripper's colored materials. Shadow removal suppresses robot cast shadows while retaining robot pixels. Camera offsets are ordered as (yaw, pitch), with one axis varied at a time. Following CubeEnv's variation-space convention, light intensity changes the diffuse coefficient of the named `global` light; the MuJoCo headlight diffuse coefficient remains fixed at 0.6.

D Interventions

D.1 Intervention of OGBench-Cube

Inspired by the intervention strategy in Nazirjonov et al. (2026), we construct fifteen visual intervention variants across seven families to assess robustness to changes in object appearance and scene rendering. These interventions are illustrated in Fig. 11 and summarized in Table 7. Object-level interventions modify the cube color (crimson to blue or yellow), the cube's rendered half-extent ($0.020\,\mathrm{m}$ to $0.012\,\mathrm{m}$ or $0.028\,\mathrm{m}$), and the gripper color (purple to red or black). Scene-level interventions remove the robot's cast shadow, change the floor color (slate to brown or magenta), perturb the camera orientation (yaw or pitch by $\pm 5^\circ$, with one axis varied at a time), and adjust the global light intensity (0.7 to 0.3 or 0.95). Following the CubeEnv variation-space convention, the lighting intervention modifies only the diffuse coefficient of the named global light, while the MuJoCo headlight remains fixed at 0.6. All interventions operate exclusively on rendering: cube-size changes preserve collision geometry, mass, and inertia, while shadow removal preserves the robot's rendered pixels. For each evaluation-sampled clip, we reconstruct the recorded simulator states and apply each intervention separately, without advancing the simulation during paired observation generation. The compared models use the same sampled clips and intervention settings. This protocol probes sensitivity to visual changes that preserve the underlying task dynamics and provides evidence about robustness to potentially spurious visual correlations.

Across the 15 visual intervention variants on OGBench-Cube, combining structural masking with sparsity regularization achieves an equally weighted mean success rate of 78.00%, compared with 76.82% for sparsity alone and 77.31% for dense slot attention. The combined design yields modest improvements of 1.18 and 0.69 percentage points, respectively.

E Pseudo Code

Algorithm 1 summarizes the joint learning of routing and dynamics. A frozen slot encoder and a shared trainable projector encode the current and next observations. The physical mask specifies admissible connections, while state-dependent threshold gates construct a binary routing graph shared across all attention heads and layers. The predictor uses hard attention for message passing, with an attention-level straight-through estimator enabling gate optimization. Prediction, SIGReg, and sparsity losses jointly optimize the trainable components, with gradients flowing through both projector branches. During autoregressive inference, the routing graph is recomputed at each step, and only hard attention is used.

Algorithm 1 Learning Causally Aligned Routing and Dynamics

Require: Dataset $\mathcal{D}$, frozen encoder E_{slot}, fixed robot-slot indicator r
Require: Depth L, temperature $\tau > 0$, loss weights $\lambda_{\text{reg}}, \lambda_{\text{sp}}$
Ensure: Trained parameters θ, including gate parameters and threshold β

Initialize all trainable parameters θ
$M \leftarrow \begin{bmatrix} \mathbf{1}_{K\times K} & r \\ r^\top & 1 \end{bmatrix}$; $\mathcal{C} \leftarrow \{(i,j) : i \neq j,\ M_{ij} = 1\}$
while not converged **do**
  Sample a minibatch $\mathcal{B}$ of transitions (o_t, a_t, o_{t+1}) from $\mathcal{D}$
  $S_t \leftarrow P_\theta(E_{\text{slot}}(o_t))$; $S_{t+1} \leftarrow P_\theta(E_{\text{slot}}(o_{t+1}))$
  Shared projector; gradients through both branches.
  $X_t^{(0)} \leftarrow [W_s s_t^1, \ldots, W_s s_t^K, W_a E_a(a_t)]$
  $q_{t,i}^g \leftarrow W_Q^g \,\mathrm{LN}_g(x_t^i)$; $k_{t,i}^g \leftarrow W_K^g \,\mathrm{LN}_g(x_t^i)$ for all tokens i
  $p_{t,ij} \leftarrow \sigma\Big(\big((q_{t,i}^g)^\top k_{t,j}^g / \sqrt{d_g} - \beta\big)/\tau\Big)$
  $G_{t,ij} \leftarrow M_{ij}\mathbf{1}[p_{t,ij} \geq 1/2]$; $\bar{p}_t \leftarrow p_t$
  Set $G_{t,ii} \leftarrow 1$ and $\bar{p}_{t,ii} \leftarrow 1$ for every token i
  for $\ell = 0, \ldots, L-1$ **do**
    Compute $(e^{(\ell,h)}, V^{(\ell,h)})$ from $\mathrm{LN}_1^{(\ell)}(X_t^{(\ell)})$ for all heads h
    $A_{\text{hard}}^{(\ell,h)} \leftarrow \mathrm{RN}\Big(G_t \odot \exp(e^{(\ell,h)})\Big)$
    $A_{\text{soft}}^{(\ell,h)} \leftarrow \mathrm{RN}\Big(M \odot \bar{p}_t \odot \exp(\mathrm{sg}(e^{(\ell,h)}))\Big)$
    $A_{\text{train}}^{(\ell,h)} \leftarrow A_{\text{hard}}^{(\ell,h)} + A_{\text{soft}}^{(\ell,h)} - \mathrm{sg}(A_{\text{soft}}^{(\ell,h)})$
    $Y_t \leftarrow X_t^{(\ell)} + \mathrm{MultiHead}^{(\ell)}\Big(\{A_{\text{train}}^{(\ell,h)} V^{(\ell,h)}\}_h\Big)$
    $X_t^{(\ell+1)} \leftarrow Y_t + \mathrm{FFN}^{(\ell)}(\mathrm{LN}_2^{(\ell)}(Y_t))$
  end for
  $\hat{s}_{t+1}^i \leftarrow D_\theta(\mathrm{LN}_{\text{out}}(X_{t,i}^{(L)}))$, $i = 1, \ldots, K$
  $\mathcal{L}_{\text{pred}} \leftarrow \mathrm{mean}_{\mathcal{B}}\Big[\frac{1}{Kd}\sum_{i=1}^K \|\hat{s}_{t+1}^i - s_{t+1}^i\|_2^2\Big]$
  $\mathcal{L}_{\text{sp}} \leftarrow \mathrm{mean}_{\mathcal{B}}\Big[\frac{1}{|\mathcal{C}|}\sum_{(i,j)\in\mathcal{C}} p_{t,ij}\Big]$
  $\mathcal{L}_{\text{SIGReg}} \leftarrow \frac{1}{2}\sum_{u\in\{t,t+1\}} \mathrm{SIGReg}(\mathrm{Pool}(S_u))$
  $\mathcal{L} \leftarrow \mathcal{L}_{\text{pred}} + \lambda_{\text{reg}}\mathcal{L}_{\text{SIGReg}} + \lambda_{\text{sp}}\mathcal{L}_{\text{sp}}$
  Update θ using $\nabla_\theta \mathcal{L}$, keeping E_{slot} frozen
end while
Notation: $\mathrm{RN}(B)_{ij} = B_{ij}/\sum_k B_{ik}$; sg denotes stop-gradient.
MultiHead concatenates heads and applies the output projection;
Pool pools samples and slots separately at each observation position.

F Loss Curves

Fig. 13 shows the training curves on OGBench-Scene, PushT, OGBench-Cube, and Two-Room. Across all four datasets, the prediction, SIGReg, and total losses decrease rapidly during early training and then change more gradually. The structural sparsity loss and gate open ratio exhibit dataset-dependent early transients before reaching relatively stable levels. The nonzero gate open ratios indicate that the learned routing retains a subset of candidate connections. These curves characterize optimization and gate usage; they do not, by themselves, establish that the retained connections recover physical causal interactions.



Figure 8: Open-loop rollouts on OGBench-Scene, OGBench-Cube, PushT, and Two-Room. Given a single history frame at $t = 0$, the predictor autoregressively generates 25 future slot states conditioned on the action sequence. For each benchmark, ground-truth observations (top) and decoded predictions (bottom) are shown at five-step intervals.

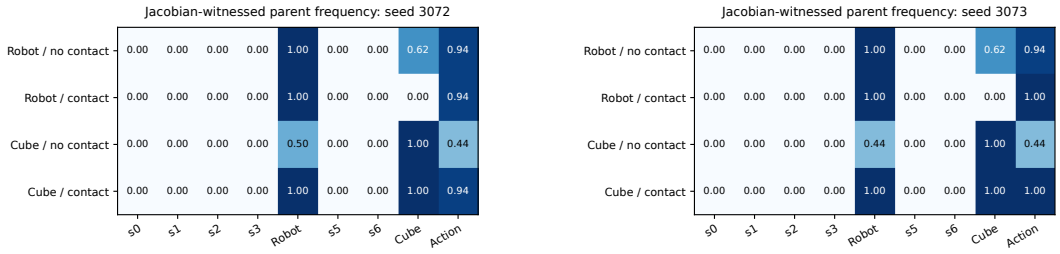
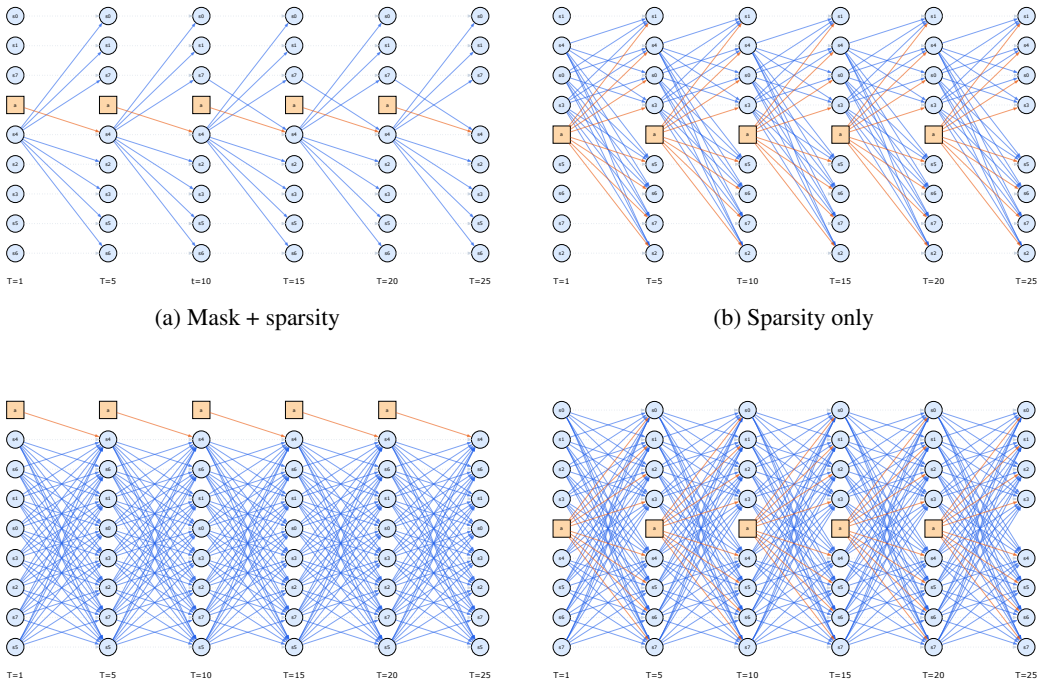


Figure 9: Jacobian-witnessed local parent frequencies on OGBench-Cube for two training seeds. Columns denote eight slot inputs and the action block; rows specify predicted outputs and contact conditions. Each entry summarizes 16 fixed-routing neighborhoods. Both seeds use the same test windows.



(a) Mask + sparsity

(b) Sparsity only

(c) Mask only

(d) Neither mask nor sparsity

Figure 10: Routing visualizations on OGBench-Cube under structural ablations. The reported edge densities are (a) 15.1%, (b) 44.4%, (c) 98.9%, and (d) 100%. Blue circles represent object slots, and orange squares represent the action token. Blue edges indicate active slot–slot connections, while orange edges indicate active action–slot connections. Graphs are rendered using Graphviz Ellson et al. (2002).

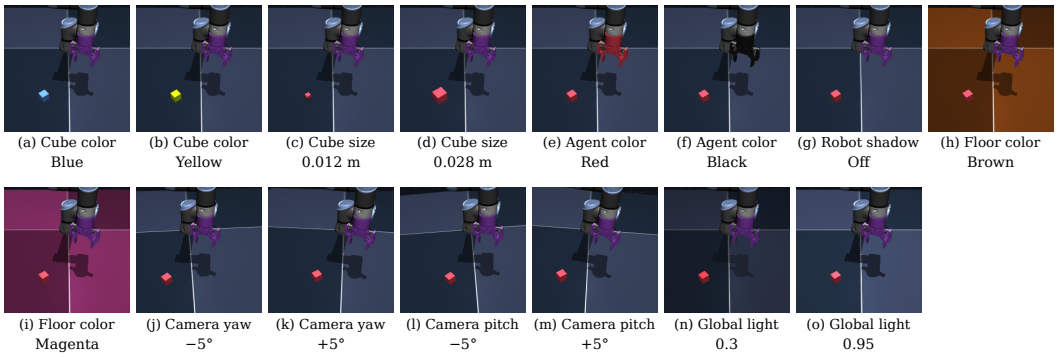


Figure 11: Visualizations of all fifteen OGBench-Cube intervention variants. Panels (a)–(f) show object-level interventions and panels (g)–(o) show scene-level interventions. All panels use the same simulator state from an evaluation-sampled clip (evaluation seed 43, episode 140, step 89). Cube size denotes the visual half-extent; physical geometry remains unchanged. Camera yaw and pitch are varied separately. Global light intensity changes the named global light, with the headlight fixed at 0.6. The final cell is intentionally left blank.

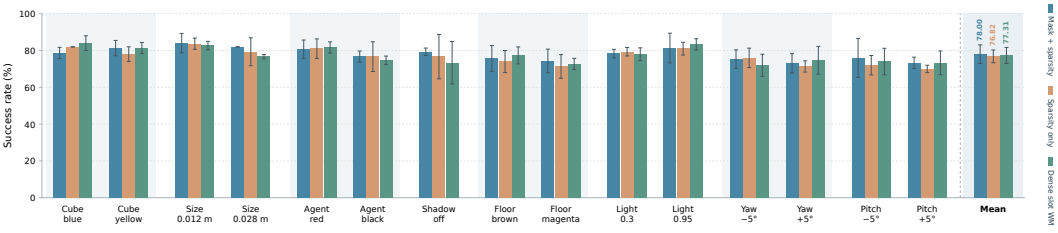


Figure 12: Planning success rates under 15 visual interventions on OGBench-Cube. We compare structural masking with sparsity regularization, sparsity alone, and dense slot attention. The final group reports the equally weighted mean success rate across all intervention variants.

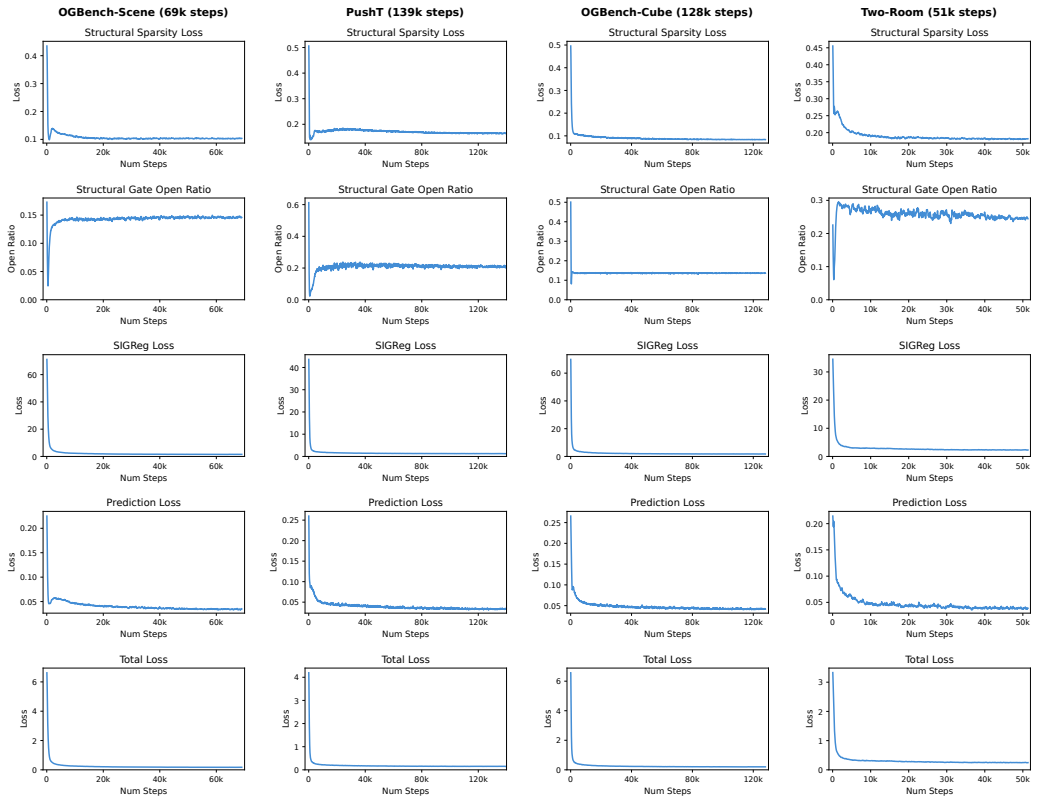


Figure 13: Training curves on OGBench-Scene, PushT, OGBench-Cube, and Two-Room (left to right). Rows show the structural sparsity loss, structural gate open ratio, SIGReg loss, prediction loss, and total loss. Each panel shows one run, with exponential moving average smoothing applied for visualization ($\alpha = 0.24$). Horizontal axes denote optimization steps, and axis ranges are set separately for each dataset and metric.