WORLD MODELS WITH CONFIGURABLE PERCEPTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Real-world observations are noisy and often contain excessive task-irrelevant information. Effective world modeling and planning need to abstract observations into world state representations while filtering out irrelevant information. However, encoders in existing world models are typically non-adaptive, leading to state representations that are easily distracted by task-irrelevant information. As a result, the model may not selectively filter observations to cater to different task demands. To address this problem, we introduce ConfigWM, which realizes configurable perception through task-conditioned state extraction. Given a language description of the task at hand, ConfigWM uses it as a configuration signal to steer how visual embeddings are mapped into a task-specific state space, selectively preserving information relevant to the current task while suppressing task-irrelevant variations. Action-conditioned dynamics are then learned in this task-specific state space for planning. We evaluate ConfigWM in environments with increasing levels of distraction. While existing methods exhibit substantial declines in planning success rate as the environments become increasingly noisy, ConfigWM remains comparatively stable and outperforms the strongest competing method by 22.3% on average across diverse environments. Consistent gains in physical-quantity probing ($+0.365$ Pearson r) and distracted-to-clean retrieval ($+75.5\%$) further indicate that ConfigWM learns states that preserve task-relevant information while remaining robust to distractors, supporting reliable world modeling and planning in noisy worlds.

1 INTRODUCTION

Real-world environments contain far more information and dynamics than are needed for any single task (Chen et al., 2025b; Bharadhwaj et al., 2022; Toso et al., 2026). However, what constitutes a sufficient state depends on the task: different tasks may require different information from the same observation. This challenge becomes particularly pronounced in dynamic environments, where multiple objects or agents may coexist and evolve simultaneously (Wu et al., 2026; Wang et al., 2026b). In such settings, only some of these dynamics may be relevant to the current task. If task-irrelevant dynamics are retained in the state, the world model must account for transitions that do not contribute to the current objective, making prediction and planning inevitably difficult.

Existing world models do not explicitly resolve this challenge. In particular, feature-based world models construct states from task-agnostic representations (Zhou et al., 2024; Assran et al., 2025; Baldassarre et al., 2025). However, a desirable state should act as an information bottleneck, retaining only the information sufficient for current task while suppressing the task-irrelevant information (Pacelli & Majumdar, 2019; Eftekhar et al., 2024; Cui et al., 2026). This is difficult to achieve with task-agnostic methods, since the relevance of an object can change with the task: the same object may be the target under one instruction but a distractor under another. The same observation can therefore require different representations under different task queries, whereas task-agnostic state representations remain unchanged.

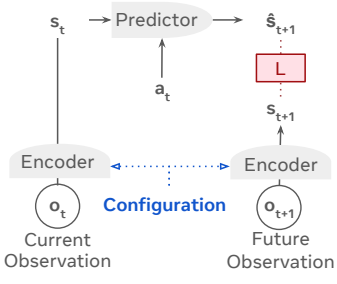

Figure 1: ConfigWM.

In the literature, it has also be noted that task-agnostic approaches may retain both target and distractor dynamics in the state (Nikulin et al., 2026). These limitations motivate adapting representations

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.

to better preserve task-relevant information. A more direct alternative is to introduce task conditioning directly into perception. This motivates *configurable perception* (LeCun et al., 2022; Shi et al., 2023; Marza et al., 2024), where the current task configures visual encoding to construct a task-relevant state.

In this work, we introduce *configurable perception* into world modeling and propose **Config World Model (ConfigWM)**, an action-conditioned world model built on task-relevant state abstractions, as illustrated in Fig. 1. Inspired by VL-JEPA (Chen et al., 2025a), we build ConfigWM on a configurable encoder that steers visual prediction with a task query and is therefore naturally suited to realizing *configurable perception*. The resulting query-conditioned representations define the task-relevant state space, over which ConfigWM learns action-conditioned dynamics for planning.

To evaluate whether *configurable perception* improves world modeling under dynamic distraction, we augment Two-Room, Push-T, and OGBench-Cube (Maes et al., 2026a) with increasing numbers of dynamic, target-like distractors and conduct controlled comparisons across different training and testing conditions. Baseline methods degrade substantially when trained on clean observations and evaluated under distraction. More importantly, simply exposing them to distractors during training does not solve the problem and can even further impair planning performance. In contrast, ConfigWM remains consistently robust as distraction increases, with an average planning improvement of 22.3%. These results show that robustness does not arise merely from exposure to distractors during training. Rather, conditioning perception on the task allows the world model to selectively preserve task-relevant dynamics while suppressing irrelevant motion, thereby supporting reliable prediction and planning in dynamically distracting environments.

Apart from planning success, we further characterize the quality of the learned states from two complementary perspectives. Physical-quantity probing (Maes et al., 2026b; Alain & Bengio, 2016) measures how well the representations preserve task-relevant physical information, while distracted-to-clean retrieval (Garrido et al., 2024) evaluates their robustness to dynamic distractors. Across both analyses, baseline representations are increasingly affected as distractors are introduced, whereas ConfigWM remains substantially more stable. These results indicate that ConfigWM better preserves task-relevant information while suppressing noise.

In summary, the contributions of this paper are as follows:

- We propose ConfigWM, which realizes *configurable perception* by using task instructions to construct task-relevant latent states and learn action-conditioned dynamics over them.
- We augment Two-Room, Push-T, and OGBench-Cube with progressively increasing dynamic, target-like distractors, establishing a controlled evaluation setting for world models in noisy environments.
- We show that ConfigWM maintains stable planning under dynamic distractors and achieves consistent gains in physical-quantity probing and distracted-to-clean retrieval, indicating better preservation of task-relevant information.

2 Related Works

World State Representations. Recent world models have explored several ways to obtain latent states suitable for prediction and planning, from directly reusing pretrained visual features to jointly learning or further adapting the state representation. A common approach is to build predictive dynamics directly over pretrained or frozen visual features, using these representations as the state space for world modeling and planning (Zhou et al., 2024; Assran et al., 2025). Rather than fixing this state space, other approaches further shape the representation for downstream control. Some jointly learn the latent state with dynamics, reward prediction, and value-learning objectives, allowing task returns to directly influence the learned representation (Hansen et al., 2022), others improve predictive state learning through latent regularization (Maes et al., 2026b; Zhao et al., 2026; Terver et al., 2026), explicitly ground the latent space in physical state and motion (Yan et al., 2026), or map generic visual features into more compact and control-relevant representations (Wang et al., 2026a; Fu et al., 2026a; Toso et al., 2026). Despite these advances, task relevance is typically introduced by shaping the learned state, rather than by allowing the current task to determine what visual information is encoded in the first place.

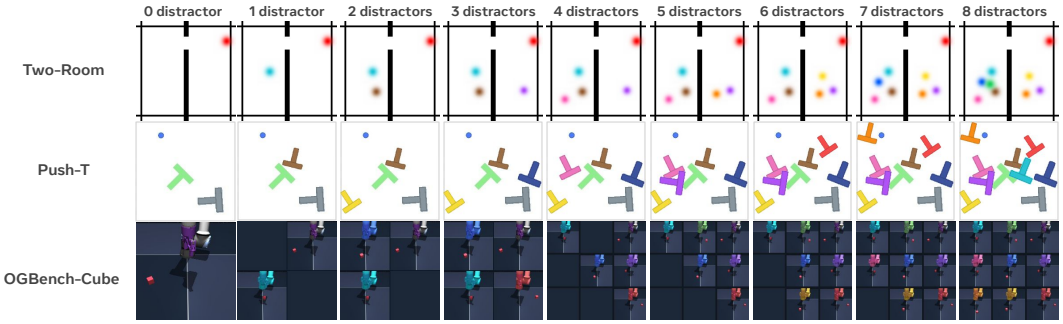

Figure 2: **Constructing noisy worlds.** Columns show observations with an increasing number of dynamic, target-like distractors. Rows show Two-Room, Push-T, and OGBench-Cube. Across all settings, the underlying task, action sequence, goal, and target trajectory remain unchanged.

Steerable Representations. Recent work has explored visual representations that can be steered by task or language signals, allowing the same observation to produce different representations depending on the current query. Some methods condition the visual encoder directly, enabling its features to emphasize queried objects, regions, or attributes (Ruthardt et al., 2026; Mao et al., 2026), while others introduce textual conditioning into predictive representation learning, where the query guides the features being predicted (Huang et al., 2026; Chen et al., 2025a). However, these methods primarily focus on learning task-conditioned visual representations, leaving open whether the resulting representations can support better action-conditioned world modeling and therefore improve downstream planning. Language has also been used to condition future prediction and action generation (Cheang et al., 2024; Brohan et al., 2023), ConfigWM differs by applying language at the state construction stage to obtain the task-relevant states for planning, extending the steerable representations to action-conditioned world modeling.

3 Constructing Noisy Worlds

To evaluate world models under dynamic distraction, we construct **controlled noisy variants** of three widely used visual control environments: Two-Room (Sobal et al., 2025), Push-T (Chi et al., 2025), and OGBench-Cube (Park et al., 2025). Together, they span 2D navigation, planar object manipulation, and 3D robotic manipulation, covering representative goal-conditioned planning settings studied in recent latent world models (Zhou et al., 2024; Wang et al., 2026a; Maes et al., 2026b).

Unlike existing noisy settings that primarily modify background appearance or introduce static clutter (Pumacay et al., 2024; Rasouli et al., 2025), we introduce multiple *dynamic, target-like distractors* that move independently and compete with the real target, while preserving the original task and target motion. By progressively increasing the number of such distractors, we create nine difficulty levels for each environment, allowing us to systematically vary dynamic distraction without changing the underlying control problem. The resulting matched clean and noisy observations are also used as training pairs for *configurable perception in our model*. We next describe the distractor construction (§ 3.1) and its instantiation in the three environments (§ 3.2).

3.1 Dynamic Distractor Construction

We construct dynamic distractors directly in the observation space, without modifying the underlying simulator state or task dynamics. We first use SAM 3 (Carion et al., 2025) to segment and track target-like objects from valid task rollouts. The extracted object masks are recolored and composited onto another rollout while preserving their original trajectories, producing visually similar objects whose motion remains consistent with the environment but is unrelated to the current task. Since this procedure operates only on visual observations, it can also be adapted to other environments without modifying their underlying simulators.

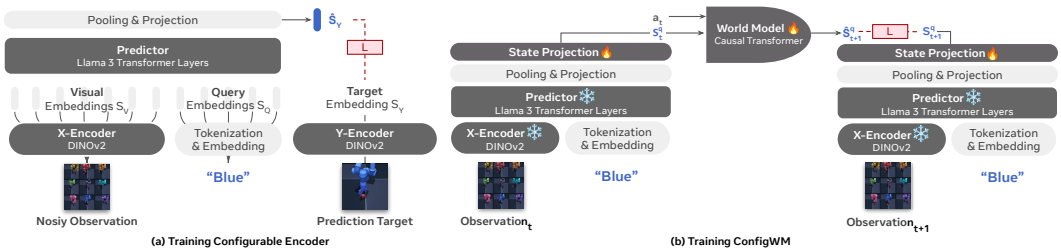

Figure 3: **Training pipeline.** (a) VL-JEPA predicts the representation of the query-specified target from a noisy observation. (b) Frozen VL-JEPA encodes the current and next observations under the same query, and ConfigWM learns action-conditioned dynamics over the projected states.

As illustrated in Fig. 2, each clean rollout is used to construct nine matched visual conditions. The zero-distractor condition d_0 contains no additional distractors, while d_ℓ contains ℓ independently moving distractors for $\ell = 1, \ldots, 8$. All levels share the same task query, action sequence, goal, and target trajectory, and differ only in the amount of competing object motion. Consequently, every noisy observation has a paired clean observation from the same rollout and time step. This matched construction provides both a controlled difficulty axis for studying world models under increasing distraction and the noisy–clean pairs used to train the VL-JEPA in Sec. 4.1.

3.2 Environment Instantiations

Two-Room (Sobal et al., 2025). A continuous 2D navigation task in which an agent must move through two connected rooms to reach a target location. We introduce additional agents with similar appearance, creating competing object motion around the query-specified agent.

Push-T (Chi et al., 2025). A planar object-pushing task in which a circular pusher moves a T-shaped block to a target pose. We introduce additional recolored T-shaped blocks, requiring the model to distinguish and track the query-specified block among visually similar moving objects.

OGBench-Cube (Park et al., 2025). A 3D robotic manipulation task in which a robotic arm moves a designated cube to a target location. We recolor the robot arms in complete manipulation scenes and combine these scenes into a tiled observation. This creates multiple simultaneous arm–cube interactions, while the task query identifies the scene relevant for prediction and planning.

4 Methodology

We aim to learn a world model with adaptive state abstractions so as to support planning in noisy environments, where multiple dynamic, realistic, and target-like distractors coexist with the main target. Following the configurator view of autonomous intelligence (LeCun et al., 2022), ConfigWM uses language to *configure perception*, thereby constructing task-relevant predictive states that serve as the state space for action-conditioned world modeling and planning.

Below, we first introduce a configurable encoder for configurable perception (§ 4.1), then describe how ConfigWM learns action-conditioned dynamics over the resulting predictive states (§ 4.2), and finally present latent-space planning with the learned world model (§ 4.3).

4.1 Configurable Encoder

To realize *configurable perception*, we introduce a configurable encoder, where a task query conditions the extraction of task-relevant visual representations. While language naturally specifies what is relevant to the task, text descriptions alone may not capture the fine-grained visual details required for low-level planning (e.g., the precise position of an agent). We therefore use language as a configuration signal while grounding the learned state representation in visual embeddings, allowing it to remain task-selective while preserving the detailed visual information needed for planning.

Formally, our model is trained with triplets $\langle X_V, X_Q, Y \rangle$, where X_V denotes a visual observation containing the task target and distractors, X_Q is the task query, and Y is the corresponding distractor-free visual target. The model comprises three components:

$$X_V \xrightarrow{\text{X-Encoder}} S_V, \qquad Y \xrightarrow{\text{Y-Encoder}} S_Y, \qquad \langle S_V, X_Q \rangle \xrightarrow{\text{Predictor}} \hat{S}_Y. \tag{1}$$

As illustrated in Fig. 3(a), both visual encoders are instantiated with DINOv2 (Oquab et al., 2023), whose dense spatial representations are well suited to low-level visual prediction and latent planning (Terver et al., 2025). A predictor implemented with Llama 3 Transformer layers jointly processes S_V and the embedded task query, followed by pooling and projection to produce $\hat{S}_Y$. The prediction is trained to match the visual target representation S_Y using an InfoNCE objective (Radford et al., 2021), encouraging the model to build visual representations that well suit the task query.

4.2 ConfigWM: Modeling Noisy Worlds

VL-JEPA constructs a task-relevant predictive state by conditioning visual state construction on the task query. As illustrated in Fig. 3(b), ConfigWM then learns how this state evolves under actions. In this formulation, the query determines what enters the predictive state, while the action determines how the state changes.

Formally, given a transition (O_t, a_t, O_{t+1}) from the noisy environments and a task query q, where O_t and O_{t+1} are the current and next observations and a_t the action taken at time t, we apply the frozen VL-JEPA to both observations under the same query, followed by a shared state projector P_ϕ:

$$S_t^q = P_\phi(E_{\text{VL-JEPA}}(O_t, q)), \qquad S_{t+1}^q = P_\phi(E_{\text{VL-JEPA}}(O_{t+1}, q)). \tag{2}$$

We instantiate F_θ as an action-conditioned causal Transformer (Maes et al., 2026b), which predicts the next task-relevant state:

$$\hat{S}_{t+1}^q = F_\theta(S_t^q, a_t). \tag{3}$$

For each task query q, S_t^q, $\hat{S}_{t+1}^q$, and S_{t+1}^q are represented in the same task-relevant latent space, where distractor dynamics are suppressed. This provides a stable predictive state space for learning action-conditioned dynamics in noisy environments. We jointly optimize the state projector P_ϕ and dynamics model F_θ by minimizing the prediction error between the predicted and encoded next states, together with SIGReg regularization (Balestriero & LeCun, 2025):

$$\mathcal{L}_{\text{WM}} = \left\| \hat{S}_{t+1}^q - S_{t+1}^q \right\|_2^2 + \lambda_{\text{sig}} \mathcal{L}_{\text{SIGReg}}. \tag{4}$$

4.3 Planning with ConfigWM

As illustrated in Fig. 4, ConfigWM performs goal-conditioned planning in the task-relevant latent space. Given an initial observation O_0, a goal observation O_g, and a task query q, the frozen VL-JEPA followed by the learned state projector encodes them as S_0^q and S_g^q, respectively. For a candidate action sequence $\mathbf{a}_{0:H-1} = (a_0, \ldots, a_{H-1})$, the world model recursively rolls out future states from S_0^q by applying $\hat{S}_{h+1}^q = F_\theta(\hat{S}_h^q, a_h)$. We score each sequence by the terminal latent distance $\|\hat{S}_H^q - S_g^q\|_2^2$. We use the cross-entropy method (CEM) (Rubinstein & Kroese, 2004) to optimize the candidate action sequences within a model predictive control framework (MPC) (Mayne et al., 2000). Since the predicted states and the goal state are constructed under the same query, the terminal latent distance measures progress toward the task-relevant goal. After executing the first action, planning is repeated from the new observation.

5 Experiments

5.1 Experimental Setup

We compare ConfigWM with LeWM (Maes et al., 2026b), TC-LeWM (Liu et al., 2026), Sub-JEPA (Zhao et al., 2026), and EB-JEPA (Terver et al., 2026), with EB-JEPA evaluated only on Two-Room following its official implementation. These methods retain their original state-learning objectives and regularization strategies. To ensure a fair comparison and isolate the role of VL-JEPA, ConfigWM uses the frozen VL-JEPA to construct query-conditioned predictive states while

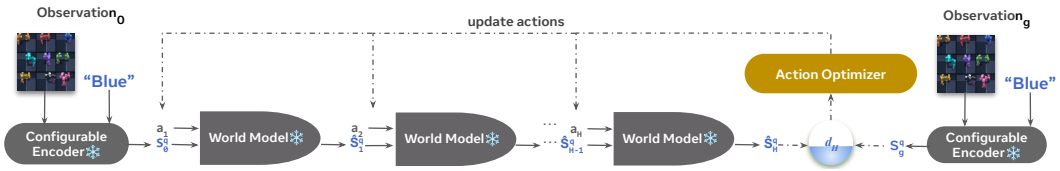

Figure 4: **Planning with ConfigWM.** VL-JEPA encodes the initial and goal observations under the task query. ConfigWM rolls out candidate actions in the query-conditioned latent space, where the terminal distance to the goal guides iterative action optimization.

adopting the LeWM-style action-conditioned causal Transformer as its dynamics model. A trainable state projector maps the spatial encoder features into the 192-dimensional global state required by the dynamics model.

We additionally construct DINOv2-WM as a controlled encoder baseline. It replaces VL-JEPA with a frozen DINOv2 encoder while retaining an identical projector architecture, dynamics model, training objective, and planning protocol. Therefore, the comparison between DINOv2-WM and ConfigWM changes only how the visual predictive state is constructed. We also report Unified ConfigWM, which uses a single configurable encoder jointly trained on pooled data from all environments and distractor levels. With this encoder frozen, separate world models are trained under the same per-environment training and evaluation protocol as the standard comparisons. All methods are trained on the same transition data and evaluated under the same planning protocol. Further optimization and evaluation details are provided in Appx. B.

5.2 Planning under Dynamic Distractors

We first examine whether world models trained in clean environments can generalize to dynamic distractors. As shown in Fig. 13, when models trained on d_0 are evaluated on d_1–d_8, the planning performance of all baselines decreases substantially as the distractor level increases. This result shows that predictive states learned in clean scenes do not readily generalize when multiple target-like objects introduce unseen competing motion.

We further consider the matched setting, where each model is trained and evaluated at the same distractor level ($d_k \rightarrow d_k$). As shown in Fig. 5, even level-matched training does not reliably recover planning performance across baselines, particularly at higher distractor levels. In contrast, ConfigWM consistently outperforms all competing baselines across distracted levels in all three environments. Beyond its consistent performance gains, ConfigWM exhibits substantially greater robustness to increasing distraction. At d_8, it retains an average of 83.8% of its d_0 success rate across the three environments, whereas the baselines retain only 42.0%. Notably, ConfigWM remains robust under distraction even when its dynamics model is trained only on d_0 transitions, achieving performance comparable to its level-matched training. Full results are provided in Appx. C.1.

These results suggest that training on distracted observations alone is not sufficient. When multiple visually similar objects exhibit valid task dynamics, existing world models struggle to identify which dynamics are relevant to the current task. By using the task query to *configure perception*, ConfigWM selects task-relevant predictive states before dynamics modeling and thereby achieves more robust planning under dynamic distractors.

5.3 Is Task-Relevant Physical Information Preserved?

We evaluate which task-relevant physical quantities remain linearly recoverable from the learned predictive states. At each distractor level, we train a linear probe on frozen representations and report the macro-average Pearson correlation r across the physical quantities defined for each environment (Maes et al., 2026b). Fig. 6 summarizes the results, while quantity-wise probing results are provided in Appx. C.2.

ConfigWM consistently preserves linearly recoverable physical information as distractor difficulty increases. From d_0 to d_8, its probing score decreases by only 0.005 on average across the three envi-

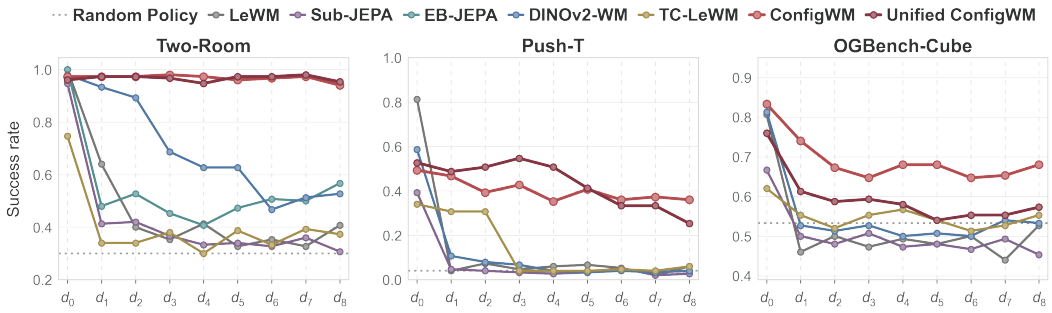

Figure 5: **Matched planning performance under dynamic distractors.** Planning success rates are shown across distractor levels d_0–d_8 on Two-Room, Push-T, and OGBench-Cube, with each model trained and evaluated at the same level ($d_k \to d_k$). Shaded regions indicate one standard deviation across seeds, and the random policy is shown as a dotted horizontal reference. Higher is better.

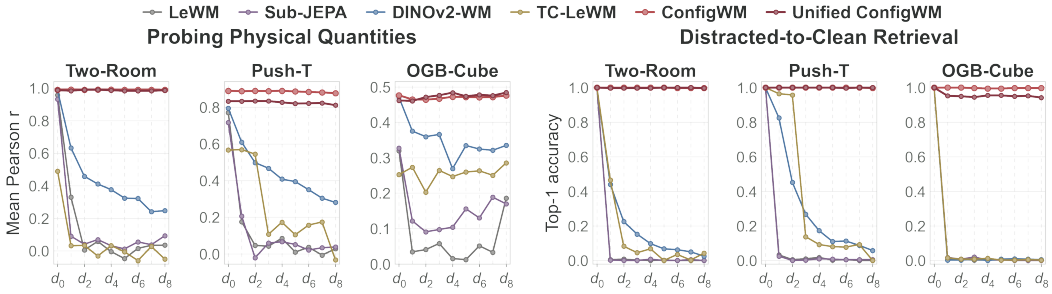

Figure 6: **Representation analysis.** Left: linear probing of task-relevant physical quantities, reported as the macro-average of the per-quantity Pearson correlations. Right: distracted-to-clean Top-1 retrieval accuracy. The three panels in each group correspond to Two-Room, Push-T, and OGBench-Cube. The clean setting is denoted by d_0, while d_1–d_8 introduce increasing levels of dynamic distractors. Higher is better.

ronments, compared with an average decrease of 0.463 for the baselines in each environment. This contrast suggests that query-configured predictive states preserve task-relevant physical information even in the presence of multiple dynamic and target-like distractors.

5.4 Do Distracted Representations Align with Clean Counterparts?

Nearest-neighbor retrieval provides a direct way to test whether a learned representation maps a transformed observation to its corresponding target in latent space (Garrido et al., 2024). For the main evaluation, we randomly sample 400 paired clean–distracted representations. Each distracted representation is used as a query over the 400 clean representations, with its paired clean representation as the positive candidate and the remaining 399 as negatives. A high retrieval score indicates that the representation preserves the identity of the underlying task-relevant state despite the added distractor motion. Detailed level-wise results are provided in Appx. C.3.

As shown in the right panels of Fig. 6, ConfigWM maintains strong clean–distracted correspondence across all three environments. This suggests that query-configured predictive states are less affected by distractor motion and better preserve task-relevant information, consistent with ConfigWM's stronger retention of clean-setting planning performance in Fig. 13.

5.5 Does the Query Configure the Predictive State?

To examine whether the task query determines which visual information is retained in the predictive state, we freeze the configurable encoder and train a single shared ViT decoder (Terver et al., 2025) on its predictor outputs. As shown in Fig. 7, changing only the queried color produces distinct reconstructions from the same input observation. Each decoded observation retains the queried

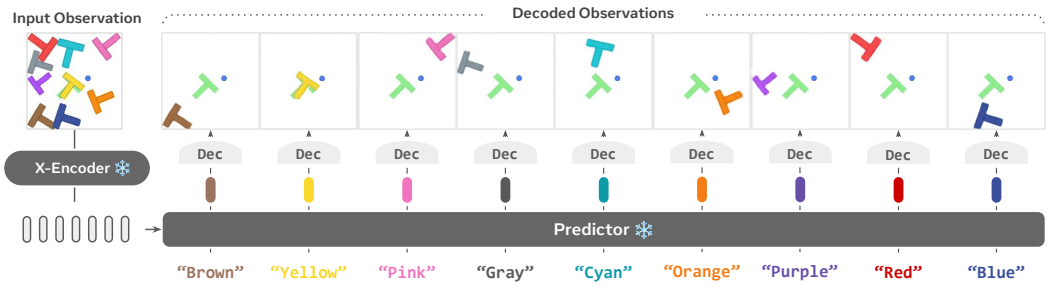

Figure 7: **Decoding query-configured predictive states.** Given the same distracted observation, we vary only the color query and decode the resulting predictive state using a shared decoder. Each decoded observation recovers the queried block and task-relevant state, while preserving other task-relevant elements (e.g., the green goal block).

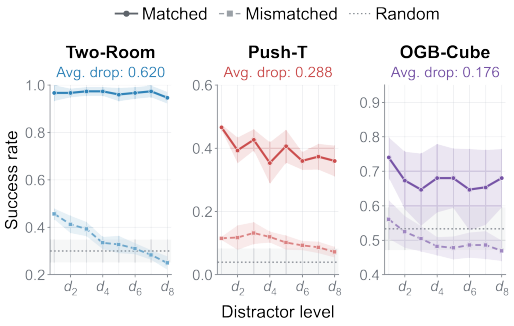

Figure 8: **Planning with mismatched queries.** Success rates (%) are averaged over distractor levels d_1–d_8. Mismatched results replace the correct query with valid queries referring to other objects. Drop denotes the decrease caused by query mismatch.

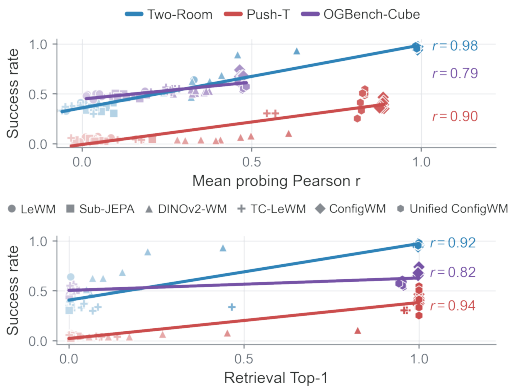

Figure 9: **Representation metrics correlate with planning performance.** Each marker pairs the representation score and success rate for different models and distractor levels. Regression lines and Pearson r are computed separately within each environment.

block together with the task-relevant agent and target pose, while largely suppressing the remaining distractors. Since the decoder has no direct access to the query, these differences provide qualitative evidence that the query configures which visual content is retained in the predictive state.

5.6 Does Perceptual Configuration Affect Planning?

While the decoding results provide qualitative evidence that configurable encoder realizes *configurable perception*, we further examine whether this query-dependent state construction affects downstream planning. We keep the task, observations, model, and planning procedure fixed, but replace the correct query with the other valid but mismatched queries referring to other objects in the scene.

As shown in Fig. 8, mismatched queries consistently reduce planning success across all three environments. Averaged over d_1–d_8 and the three environments, planning success decreases from 67.8% with the correct query to 31.6% with mismatched queries, approaching the 29.1% performance of a random policy. Since the task query is the only changed input, the resulting performance drop provides direct evidence that downstream planning depends on the query-configured predictive states constructed through *configurable perception*. Complete results are provided in Appx. C.5.

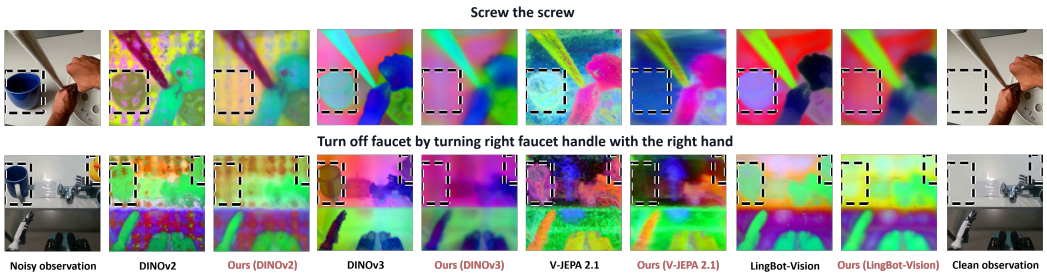

Figure 10: **Configurable encoder across visual backbones.** We compare each backbone with our configurable encoder built upon the same backbone. Across DINOv2, DINOv3, V-JEPA 2.1, and LingBot-Vision, ours suppresses task-irrelevant distractors while preserving task-relevant state information. Black dashed boxes mark distractors.

5.7 Do Representation Metrics Reflect Planning Performance?

We further examine whether these representation metrics reflect downstream planning performance using correlation analysis (Dong et al., 2026; Shen et al., 2025). Within each environment, we correlate the representation scores with matched planning success across six methods and distractor levels d_1–d_8. As shown in Fig. 9, physical-quantity probing and retrieval both achieve an average Pearson r of 0.89 across the three environments, suggesting that they provide informative intermediate measures of planning performance.

6 Scaling the Configurable Encoder with Real-World Data

We explore a scalable training route that converts existing egocentric (Grauman et al., 2022; Damen et al., 2022), robot manipulation (Khazatsky et al., 2024; O'Neill et al., 2024) and internet instructional video (Tang et al., 2019; Chen et al., 2026) data into supervision for *configurable perception*. Given a clean observation and its task description, a VLM generates editing prompts specifying reasonable but task-irrelevant objects, which are inserted into the observation using an image-editing model. The task-relevant scene content and task description remain unchanged, and the original observation serves as the clean target. The configurable encoder is then trained to predict the representation of the clean target from the noisy observation conditioned on the task description.

Fig. 10 shows held-out PCA visualizations across DINOv2, DINOv3, V-JEPA 2.1, and LingBot-Vision. Complementary patch-level linear probing for distractor and task-relevant region prediction shows that the configurable encoder reduces distractor average precision (AP) by an average of 17.9 percentage points relative to the frozen backbones, while task-core AP remains nearly unchanged, indicating weaker responses to task-irrelevant objects while preserving task-relevant states. Full quantitative results and the evaluation protocol are provided in Appendix D. These results suggest that our encoder can benefit different visual backbones, providing a practical route for scaling *configurable perception* to large real-world video datasets, without requiring manual annotation.

7 Conclusion

We introduced ConfigWM, an action-conditioned world model that brings *configurable perception* into state construction. Rather than modeling dynamics over a fixed task-agnostic representation, ConfigWM uses the task query to construct a task-relevant state and learns action-conditioned dynamics over it. Across three environments with increasing dynamic, target-like distractors, ConfigWM maintains stronger planning performance than the baselines, while representation analyses show that its states preserve task-relevant physical information and remain more consistent with their clean counterparts as distraction increases. We further examine whether the query actually configures the learned state, finding that decoding the states under different queries reveals different task-relevant state representations, while replacing the correct query with a mismatched one leads to a substantial drop in planning success. Beyond these controlled environments, results on real-world data suggest a promising path to scaling the configurable encoder across different visual backbones.

AI USE STATEMENT

In this work, generative AI tools were used for manuscript refinement and for the synthetic distractor-generation pipeline described in Appx. D. All AI-assisted outputs were manually reviewed and verified by the authors, who take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

We provide noisy-world construction and dataset statistics in Appx. A, implementation and evaluation details in Appx. B, additional experimental results in Appx. C, and the real-world scaling setup and full synthesis prompts in Appx. D.

REFERENCES

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Federico Baldassarre, Marc Szafraniec, Basile Terver, Vasil Khalidov, Francisco Massa, Yann LeCun, Patrick Labatut, Maximilian Seitzer, and Piotr Bojanowski. Back to the features: Dino as a foundation for video world models. *arXiv preprint arXiv:2507.19468*, 2025.

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

Homanga Bharadhwaj, Mohammad Babaeizadeh, Dumitru Erhan, and Sergey Levine. Information prioritization through empowerment in visual model-based rl. *arXiv preprint arXiv:2204.08585*, 2022.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pp. 961–970, 2015.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. Sam 3: Segment anything with concepts. *arXiv preprint arXiv:2511.16719*, 2025.

Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

Delong Chen, Mustafa Shukor, Theo Moutakanni, Willy Chung, Jade Yu, Tejaswi Kasarla, Yejin Bang, Allen Bolourchi, Yann LeCun, and Pascale Fung. Vl-jepa: Joint embedding predictive architecture for vision-language. *arXiv preprint arXiv:2512.10942*, 2025a.

Delong Chen, Tejaswi Kasarla, Yejin Bang, Mustafa Shukor, Willy Chung, Jade Yu, Allen Bolourchi, Theo Moutakanni, and Pascale Fung. Action100m: A large-scale video action dataset. *arXiv preprint arXiv:2601.10592*, 2026.

Yuxin Chen, Jianglan Wei, Chenfeng Xu, Boyi Li, Masayoshi Tomizuka, Andrea Bajcsy, and Ran Tian. Reimagination with test-time observation interventions: Distractor-robust world model predictions for visual model predictive control. *arXiv preprint arXiv:2506.16565*, 2025b.

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

Jingyi Cui, Qi Zhang, Hongwei Wen, and Yisen Wang. A generalization theory for jepa-based world models. *arXiv preprint arXiv:2606.27014*, 2026.

Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Antonino Furnari, Evangelos Kazakos, Jian Ma, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100. *International Journal of Computer Vision*, 130(1):33–55, 2022.

Jiahua Dong, Yunze Man, Pavel Tokmakov, and Yu-Xiong Wang. Capturing visual environment structure correlates with control performance. In *International Conference on Learning Representations*, volume 2026, pp. 35435–35462, 2026.

Ainaz Eftekhar, Kuo-Hao Zeng, Jiafei Duan, Ali Farhadi, Aniruddha Kembhavi, and Ranjay Krishna. Selective visual representations improve convergence and generalization for embodied ai. In *International Conference on Learning Representations*, volume 2024, pp. 13256–13277, 2024.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, pp. 1407–1416. PMLR, 2018.

Minghao Fu, Fan Feng, Nicklas Hansen, and Biwei Huang. Back to parsimonious latents: Learning task-centric world models from visual foundations. *arXiv preprint arXiv:2605.25620*, 2026a.

Zelin Fu, Bin Tan, Changjiang Sun, Shaohui Liu, Kecheng Zheng, Yinghao Xu, Xing Zhu, Yujun Shen, and Nan Xue. Vision pretraining for dense spatial perception. *arXiv preprint arXiv:2607.05247*, 2026b.

Quentin Garrido, Mahmoud Assran, Nicolas Ballas, Adrien Bardes, Laurent Najman, and Yann LeCun. Learning and leveraging world models in visual representation learning. *arXiv preprint arXiv:2403.00504*, 2024.

Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 18995–19012, 2022.

Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, et al. Ego-exo4d: Understanding skilled human activity from first-and third-person perspectives. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 19383–19400, 2024.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. *arXiv preprint arXiv:2203.04955*, 2022.

Chen Huang, Xianhang Li, Vimal Thilak, Etai Littwin, and Josh Susskind. Text-conditional jepa for learning semantically rich visual representations. *arXiv preprint arXiv:2605.03245*, 2026.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

Yann LeCun et al. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62(1):1–62, 2022.

Chang Liu, Fei Suo, Yanzhou Jin, Yusuke Iwasawa, Yutaka Matsuo, and Yaonan Zhu. Temporally centered sigreg improves multi-task leworldmodel learning: From analysis to method. *arXiv preprint arXiv:2607.26924*, 2026.

Lucas Maes, Quentin Le Lidec, Dan Haramati, Nassim Massaudi, Damien Scieur, Yann LeCun, and Randall Balestriero. stable-worldmodel-v1: Reproducible world modeling research and evaluation. *arXiv preprint arXiv:2602.08968*, 2026a.

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*, 2026b.

Chengzhi Mao, Xudong Lin, and Wen-Sheng Chu. Language-instructed vision embeddings for controllable and generalizable perception. *arXiv preprint arXiv:2606.19584*, 2026.

Pierre Marza, Laetitia Matignon, Olivier Simonin, and Christian Wolf. Task-conditioned adaptation of visual features in multi-task policy learning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17847–17856. IEEE, 2024.

David Q Mayne, James B Rawlings, Christopher V Rao, and Pierre OM Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

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. *arXiv preprint arXiv:2603.14482*, 2026.

Alexander Nikulin, Ilya Zisman, Albina Klepach, Denis Tarasov, Alexander Derevyagin, Andrei Polubarov, Lyubaykin Nikita, and Vladislav Kurenkov. Vision-language models unlock task-centric latent actions. *arXiv preprint arXiv:2601.22714*, 2026.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

Vincent Pacelli and Anirudha Majumdar. Task-driven estimation and control via information bottlenecks. In *2019 international conference on robotics and automation (ICRA)*, pp. 2061–2067. IEEE, 2019.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International Conference on Learning Representations*, volume 2025, pp. 94937–94982, 2025.

Wilbert Pumacay, Ishika Singh, Jiafei Duan, Ranjay Krishna, Jesse Thomason, and Dieter Fox. The colosseum: A benchmark for evaluating generalization for robotic manipulation. *arXiv preprint arXiv:2402.08191*, 2024.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Amir Rasouli, Montgomery Alban, Sajjad Pakdamansavoji, Zhiyuan Li, Zhanguang Zhang, Aaron Wu, and Xuan Zhao. Distracted robot: How visual clutter undermine robotic manipulation. *arXiv preprint arXiv:2511.22780*, 2025.

Reuven Y Rubinstein and Dirk P Kroese. *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation, and machine learning*, volume 133. Springer, 2004.

Jona Ruthardt, Manu Gaur, Deva Ramanan, Makarand Tapaswi, and Yuki M Asano. Steerable visual representations. *arXiv preprint arXiv:2604.02327*, 2026.

Dian Shao, Yue Zhao, Bo Dai, and Dahua Lin. Finegym: A hierarchical video dataset for fine-grained action understanding. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2613–2622. IEEE, 2020.

Yijun Shen, Delong Chen, Fan Liu, Xingyu Wang, Chuanyi Zhang, Liang Yao, and Yuhui Zheng. Chain-of-talkers (cotalk): Fast human annotation of dense image captions. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 4444–4464, 2025.

Baifeng Shi, Trevor Darrell, and Xin Wang. Top-down visual attention from analysis by synthesis. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2102–2112. IEEE, 2023.

Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

Vlad Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, Tim GJ 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.

Yansong Tang, Dajun Ding, Yongming Rao, Yu Zheng, Danyang Zhang, Lili Zhao, Jiwen Lu, and Jie Zhou. Coin: A large-scale dataset for comprehensive instructional video analysis. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1207–1216. IEEE, 2019.

Basile Terver, Tsung-Yen Yang, Jean Ponce, Adrien Bardes, and Yann LeCun. What drives success in physical planning with joint-embedding predictive world models? *arXiv preprint arXiv:2512.24497*, 2025.

Basile Terver, Randall Balestriero, Megi Dervishi, David Fan, Quentin Garrido, Tushar Nagarajan, Koustuv Sinha, Wancong Zhang, Mike Rabbat, Yann LeCun, et al. A lightweight library for energy-based joint-embedding predictive architectures. *arXiv preprint arXiv:2602.03604*, 2026.

Leonardo F Toso, Davit Shadunts, Yunyang Lu, Nihal Sharma, Donglin Zhan, Nam H Nguyen, and James Anderson. Learning invariant visual representations for planning with joint-embedding predictive world models. *arXiv preprint arXiv:2602.18639*, 2026.

Ying Wang, Oumayma Bounou, Gaoyue Zhou, Randall Balestriero, Tim GJ Rudner, Yann LeCun, and Mengye Ren. Temporal straightening for latent planning. *arXiv preprint arXiv:2603.12231*, 2026a.

Zizhao Wang, Chang Shi, Jiaheng Hu, Kevin Rohling, Roberto Martín-Martín, Amy Zhang, and Peter Stone. Factored latent action world models. *arXiv preprint arXiv:2602.16229*, 2026b.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025.

Haoyu Wu, Jiwen Yu, Yingtian Zou, and Xihui Liu. Multiworld: Scalable multi-agent multi-view video world models. *arXiv preprint arXiv:2604.18564*, 2026.

Kun Wu, Chengkai Hou, Jiaming Liu, Zhengping Che, Xiaozhu Ju, Zhuqin Yang, Meng Li, Yinuo Zhao, Zhiyuan Xu, Guang Yang, et al. Robomind: Benchmark on multi-embodiment intelligence normative data for robot manipulation. *arXiv preprint arXiv:2412.13877*, 2024.

Haodong Yan, Jiaguan Zhu, Mingyuan Jia, Ruiqing Yin, Junjie He, Zhide Zhong, Junfeng Li, Jinxuan Lu, Hengtao Li, Tianran Zhang, et al. Is forward prediction enough? physical state grounding for jepa world models. *arXiv preprint arXiv:2608.06799*, 2026.

Kai Zhao, Dongliang Nie, Yuchen Lin, Zhehan Luo, Yixiao Gu, Deng-Ping Fan, and Dan Zeng. Sub-jepa: Subspace gaussian regularization for stable end-to-end world models. *arXiv preprint arXiv:2605.09241*, 2026.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. Dino-wm: World models on pre-trained visual features enable zero-shot planning. *arXiv preprint arXiv:2411.04983*, 2024.

A Details on Noisy-World Construction

This section details the construction of our noisy-world datasets. We first describe the shared distractor-generation pipeline in Sec. A.1, then its environment-specific instantiations in Sec. A.2, and finally the resulting dataset scale in Sec. A.3.

A.1 Dynamic Distractor Generation

Object Extraction. For each environment, we begin with the official clean rollouts and use SAM3 (Carion et al., 2025) to identify the task-relevant target object. Specifically, SAM3 segments the object and tracks its mask consistently across frames, producing a sequence of per-frame masks and image-space trajectories. These masks and tracks provide the spatial extent and motion information required to replay the extracted objects as distractors in other rollouts. We then reconstruct each extracted object to match the native appearance of the corresponding environment. This reconstruction preserves its characteristic geometry, boundary profile, and shading while allowing controlled recoloring. Finally, the reconstructed object is composited into a recipient rollout by replaying its tracked donor trajectory frame by frame. Consequently, each distractor follows motion observed in a valid rollout rather than being independently repositioned in every frame, resulting in temporally stable and visually plausible distractors.

Progressive noise levels. We construct the noise levels cumulatively. The d_0 condition contains only the recolored task-relevant objects and no additional distractors. For each trajectory, we sample eight donor trajectories and keep their identities and tracks fixed throughout the construction. Starting from d_0, we add one additional donor trajectory at each noise level. This progressive construction enables controlled comparisons while keeping all previously introduced distractors unchanged.

Color-balanced and query-conditioned targets. In this work, we use object color as a concrete and easily controlled visual configuration attribute, although the formulation itself is not inherently restricted to color. For example, the Two-Room environment uses nine semantically named colors: red, blue, green, orange, purple, cyan, yellow, pink, and brown. To prevent the clean target from being associated with a fixed appearance inherited from the original dataset, such as the red ball in Two-Room, we recolor the task-relevant core object and balance the nine colors across trajectories. At every noise level d_ℓ, the core object and its ℓ distractors are assigned mutually distinct colors.

In addition to the standard source-to-clean pairs derived from the official datasets, we construct a same-source multi-query dataset from d_8, which contains the complete set of one core object and eight distractors. At a temporal stride of five, the same noisy observation is paired with nine color queries and nine corresponding query-selected clean targets. This construction provides a direct realization of *configurable perception*, changing only the language query changes the desired target while keeping the source pixels fixed. Each target retains only the object or manipulation scene specified by the query (task-relevant) and removes the non-queried (task-irrelevant) distractors, while preserving task-relevant contextual elements (e.g., goals and interaction objects).

A.2 Environment-Specific Construction

Two-Room. The task-relevant object is the moving ball. SAM3 is used to extract and track balls from donor trajectories. Their soft, Gaussian-like appearance is reconstructed with the native style of the environment, recolored, and replayed along the extracted trajectories. The room walls, and other static geometry are preserved during compositing. Queries follow the template `{color} ball`, and the corresponding target contains only the queried ball in the matched clean room observation.

Push-T. The task-relevant object is the movable T-shaped block, while the green T-shaped region and blue pusher provide task context. Additional T-block trajectories are extracted from valid donor rollouts. We reconstruct each donor with the geometry and edge appearance of the original T-block, recolor it, and replay it along its donor trajectory. Only the donor T-block is transferred, while the green goal and blue pusher are retained from the recipient observation. Natural occlusions between the pusher, selected T-block, distractors, and goal are preserved during compositing. Queries follow the template `{color} T block`.

OGBench-Cube. Each donor corresponds to a complete arm–cube manipulation trajectory, identified by the color of its robot arm. SAM3 is used to extract and track the relevant manipulation

Table 1: Scale of the constructed noisy-world datasets.

Environment	Trajectories	Avg. Steps	Source–Clean Pairs
Two-Room	10,000	92.1	10,002,339
Push-T	18,685	125.1	25,301,106
OGBench-Cube	10,000	201.0	21,780,000
Total	38,685	136.2	57,083,445

content, after which the robot arm is reconstructed with an environment-matched appearance and recolored. The associated cube and interaction context are retained so that each donor remains a coherent manipulation scene. To keep all manipulation scenes visible, we use an adaptive tiled layout: d_1–d_3 use a 2×2 grid, whereas d_4–d_8 use a 3×3 grid. This layout change is introduced only to accommodate the increasing number of distractors, while the controlled task, action sequence, goal, and target trajectory remain unchanged. Queries follow the template `{color} arm` and select the clean manipulation observation associated with the correspondingly colored arm.

A.3 Dataset Scale

Our noisy-world dataset spans three environments: Two-Room (Sobal et al., 2025), Push-T (Chi et al., 2025), and OGBench-Cube (Park et al., 2025). All three datasets are constructed from the official clean trajectories released by Maes et al. (2026a). Their detailed statistics are summarized in Table 1 (the average trajectory length in the total row is weighted by the number of trajectories in each environment). A source–clean pair consists of a source observation, a language query, and the corresponding query-conditioned clean target at the same time step. Source-clean pairs include all samples from d_0–d_8. In addition to adding distractors to each frame, we also construct samples in which the same source frame is paired with different queries and their corresponding clean targets.

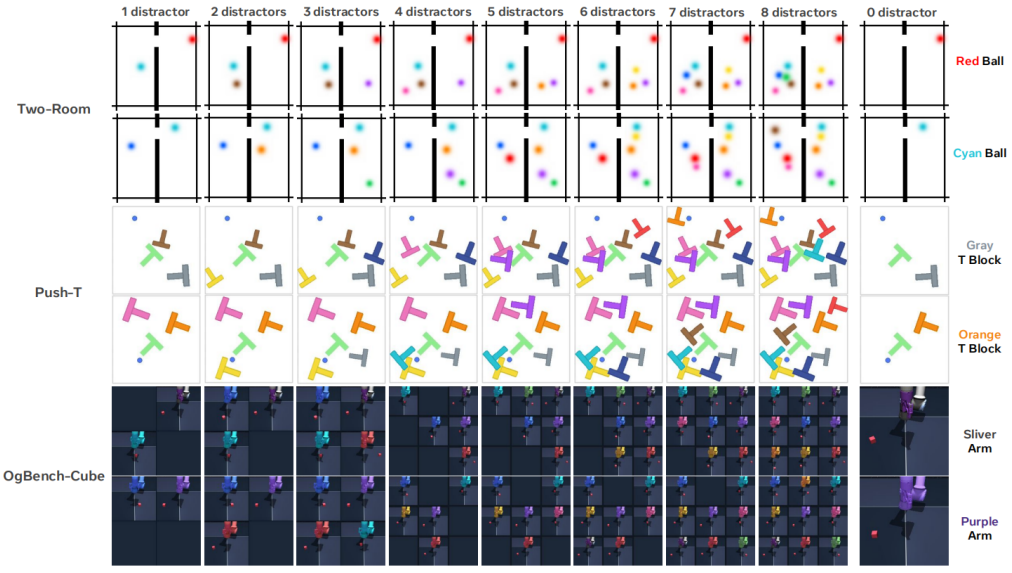

Figure 11: **Progressive noisy-world construction and query-conditioned targets.** Within each row, d_1–d_8 cumulatively add distractors to the same trajectory frame. The rightmost column shows the d_0 target selected by the displayed query.

B Implementation Details

B.1 The Configurable Encoder Training

We train the configurable encoder on the three controlled environments used in our main experiments: Two-Room, Push-T, and OGBench-Cube. We later extend the same training formulation to real-world visual data in Sec. D. Below, we describe the model architecture, the patch-wise contrastive objective, and the training data and optimization setup.

Architecture. Both the input encoder E_X and target encoder E_Y are initialized from pretrained DINOv2-S. All images are resized to 224×224 and processed using the standard DINOv2 preprocessing pipeline. Each image is encoded into one CLS token and a 16×16 grid of 384-dimensional patch tokens. During the configurable encoder training, E_X is optimized jointly with the predictor, whereas E_Y is kept as an independent frozen copy of DINOv2, providing a fixed target representation space throughout training.

The predictor P_θ consists of the first three Transformer layers of pretrained Llama-3.2-1B. A trainable linear layer maps the 384-dimensional visual tokens to the Llama hidden dimension, while the task query is tokenized with the Llama tokenizer and padded or truncated to at most 32 tokens. The pretrained Llama token-embedding layer is kept frozen, whereas the Transformer layers and the visual input and output projections are trainable. The predictor processes the concatenation of the visual CLS and patch tokens with the query tokens, and its visual patch outputs are projected back to 384 dimensions, matching the dimensionality of the DINOv2 feature space. This provides a common feature interface for the subsequent world-model experiments and facilitates controlled comparison with DINOv2-WM. The target encoder receives only the corresponding clean target image and does not take the task query as input. This keeps the target representation query-independent, requiring the predictive branch to use the query to select task-relevant visual information from the noisy observation.

Training Data. The configurable encoder is trained using the noisy-world datasets described in Appendix A. Each training sample consists of a noisy source observation, a task query, and the corresponding clean target. We use source-clean pairs constructed across different distractor levels and additionally include same-source multi-query samples, where the same observation is paired with multiple valid queries and clean targets. These samples provide diverse query-conditioned training signals while preserving the same underlying visual state. The overall scale of the constructed training data is summarized in Table 1.

Training Objective. The configurable encoder is trained with a spatially aligned patch-level multi-positive InfoNCE objective. Given the visual embeddings S_V and query embeddings S_Q, the predictor produces $\hat{S}_Y$, which is trained to match the target representation S_Y from the frozen Y-Encoder. Since the source and target observations are spatially aligned, each predicted patch is contrasted against target patches at the corresponding spatial location across the contrastive bank. Targets sharing the same query-conditioned target identity are treated as positives, whereas same-source samples associated with different queries remain negatives. Following the bidirectional InfoNCE formulation of VL-JEPA (Chen et al., 2025a), we average the loss across spatial locations and compute it symmetrically in both directions:

$$\mathcal{L}_{\text{VL-JEPA}} = \frac{1}{2}\left(\mathcal{L}_{\hat{S}_Y \to S_Y} + \mathcal{L}_{S_Y \to \hat{S}_Y}\right). \tag{5}$$

Optimization Details. We train the configurable encoder for one epoch using AdamW with the settings summarized in Table 2.

Table 2: **Optimization settings for the configurable encoder training.**

Setting	Value	Setting	Value
Learning rate	5×10^{-5}	Weight decay	0.05
LR schedule	100-step warmup; cosine decay	Global batch size	512
Temperature τ	0.07	Gradient clipping	1.0
Numerical precision	BF16	Training hardware	1× A100 (80 GB)

B.2 CONFIGWM TRAINING

ConfigWM follows the action-conditioned latent dynamics formulation of LeWM (Maes et al., 2026b). To isolate the contribution of the configurable encoder under a controlled comparison, we replace LeWM's end-to-end visual encoder with the frozen configured-state representation model while retaining its state dimensionality, causal Transformer dynamics core, and training objective. Accordingly, a trainable spatial projector aggregates the configurable encoder's patch features into the 192-dimensional state S_t^q, matching the state dimensionality used by LeWM. The projector is shared across input observations and target observations, and trained jointly with the world model, while all the configurable encoder parameters remain frozen.

The dynamics predictor is a causal Transformer with AdaLN-zero action conditioning. During training, it takes a three-state context sampled at intervals of $\Delta = 5$ environment steps, with each state conditioned on the intervening Δ-step action block, and predicts the subsequent states using teacher forcing. Actions are normalized using the training-set statistics. Following LeWM, we optimize the mean-squared latent prediction error together with SIGReg:

$$\mathcal{L}_{\text{ConfigWM}} = \mathcal{L}_{\text{pred}} + \lambda_{\text{sig}}\mathcal{L}_{\text{SIGReg}}, \qquad \lambda_{\text{sig}} = 0.09, \tag{6}$$

where $\mathcal{L}_{\text{pred}}$ denotes the mean-squared error between the predicted and target states.

B.3 BASELINE IMPLEMENTATION DETAILS

All comparison methods are trained on the same trajectories for 10 epochs, and we report the best-performing checkpoint selected using the same procedure. LeWM, Sub-JEPA, TC-LeWM, DINOv2-WM, and ConfigWM use a three-state context with $\Delta = 5$ environment steps and share the same action normalization and online planning protocol. We describe only the method-specific differences below.

LeWM-Family Baselines. LeWM (Maes et al., 2026b), Sub-JEPA (Zhao et al., 2026), and TC-LeWM (Liu et al., 2026) use a randomly initialized ViT-Tiny encoder and are trained end to end, retaining LeWM's original 192-dimensional state representation and predictor projection. These baselines use a global batch size of 128 and cosine learning-rate decay. LeWM uses its original latent prediction objective with SIGReg. Sub-JEPA replaces SIGReg with its multi-subspace variant, using $K = 16$ for Push-T and $K = 32$ for Two-Room and OGBench-Cube, following the best-performing settings reported in the original paper. TC-LeWM instead uses temporally centered SIGReg with a temporal window of $W = 4$, covering the three context states and their prediction target.

DINOv2-WM. As a controlled encoder baseline for ConfigWM, DINOv2-WM replaces the configurable encoder with a frozen pretrained DINOv2-S visual encoder (Oquab et al., 2023) and does not use language conditioning. Its patch-token features are mapped to the 192-dimensional state space using the same spatial projector as ConfigWM. All remaining components and settings, including the action encoder, dynamics predictor, training objective, and optimization protocol, are identical to those of ConfigWM. The DINOv2 encoder remains frozen, and only the state projector, action encoder, and dynamics predictor are trained.

EB-JEPA. EB-JEPA (Terver et al., 2026) is evaluated only on Two-Room, following its official implementation. We retain its original IMPALA-based (Espeholt et al., 2018) encoder, training objective, and optimizer settings and train the model end to end from random initialization. It uses the same Two-Room training trajectories, checkpoint-selection procedure, and online planning protocol as the other methods.

B.4 PLANNING AND EVALUATION PROTOCOL

We evaluate all world models using the same CEM-based model predictive control protocol, following LeWM (Maes et al., 2026b). At each replanning step, the current and goal observations are encoded using the corresponding representation model. CEM optimizes candidate action sequences by rolling them out through the learned dynamics model and minimizing the distance between the terminal predicted state and the encoded goal state. A prefix of the optimized action sequence is executed before replanning from the updated observation. The same planner configuration is used across methods within each environment. All reported results are averaged over evaluation seeds 40, 42, and 45.

C Additional Experimental Results

C.1 Detailed Planning Results

Table 3: **Matched planning performance.** Models are trained and evaluated at the same distractor level ($d_k \to d_k$). Values are mean success rates (%) over seeds 40/42/45. Light-blue tags indicate standard deviation. Average is computed by first averaging d_0–d_8 within each seed and then aggregating the three seed-level averages. Column-best means are bold.

Method	Distractor Level									Average
	d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	
Two-Room										
Random Policy					30.0 ±4.9					
LeWM	**100.0** ±0.0	64.0 ±4.3	40.0 ±5.9	35.3 ±2.5	41.3 ±10.9	32.7 ±1.9	35.3 ±3.4	32.7 ±4.1	40.7 ±0.9	46.9 ±0.8
Sub-JEPA	94.7 ±3.4	41.3 ±2.5	42.0 ±5.9	36.7 ±4.7	33.3 ±8.2	34.0 ±8.5	32.7 ±5.7	36.0 ±4.3	30.7 ±2.5	42.4 ±3.2
TC-LeWM	74.7 ±2.5	34.0 ±5.9	34.0 ±2.8	38.0 ±4.3	30.0 ±8.6	38.7 ±3.8	33.3 ±5.2	39.3 ±5.0	37.3 ±0.9	39.9 ±2.1
EB-JEPA	**100.0** ±0.0	48.0 ±4.3	52.7 ±0.9	45.3 ±3.4	40.7 ±2.5	47.3 ±6.6	50.7 ±6.6	50.0 ±4.3	56.7 ±0.9	54.6 ±2.6
DINOv2-WM	98.0 ±1.6	93.3 ±1.9	89.3 ±1.9	68.7 ±0.9	62.7 ±0.9	62.7 ±2.5	46.7 ±12.3	51.3 ±9.6	52.7 ±1.9	69.5 ±2.5
ConfigWM	97.3 ±0.9	**97.3** ±2.5	**97.3** ±0.9	**98.0** ±1.6	**97.3** ±2.5	96.0 ±2.8	96.7 ±2.5	97.3 ±2.5	94.0 ±1.6	**96.8** ±1.8
Unified ConfigWM	96.0 ±3.3	**97.3** ±2.5	**97.3** ±0.9	96.7 ±1.9	94.7 ±3.4	**97.3** ±1.9	**97.3** ±2.5	**98.0** ±1.6	**95.3** ±4.1	96.7 ±2.4
Push-T										
Random Policy					4.0 ±4.3					
LeWM	**81.3** ±5.7	4.0 ±2.8	7.3 ±2.5	4.7 ±2.5	6.0 ±4.3	6.7 ±2.5	5.3 ±1.9	2.0 ±0.0	6.0 ±1.6	13.7 ±1.0
Sub-JEPA	39.3 ±1.9	4.7 ±0.9	4.0 ±1.6	3.3 ±0.9	2.7 ±2.5	3.3 ±0.9	5.3 ±3.4	2.0 ±0.0	2.7 ±0.9	7.5 ±0.8
TC-LeWM	34.0 ±1.6	30.7 ±0.9	30.7 ±3.4	4.0 ±2.8	4.0 ±2.8	4.0 ±3.3	4.7 ±3.4	4.0 ±1.6	6.0 ±1.6	13.6 ±1.5
DINOv2-WM	58.7 ±0.9	10.7 ±1.9	8.0 ±3.3	6.7 ±1.9	4.0 ±0.0	3.3 ±0.9	4.0 ±1.6	3.3 ±0.9	4.0 ±1.6	11.4 ±0.6
ConfigWM	49.3 ±4.1	46.7 ±0.9	39.3 ±4.1	42.7 ±3.4	35.3 ±6.6	40.7 ±5.2	**36.0** ±2.8	**37.3** ±4.1	**36.0** ±4.9	40.4 ±2.7
Unified ConfigWM	52.7 ±6.6	**48.7** ±2.5	**50.7** ±6.6	**54.7** ±3.4	**50.7** ±7.5	**41.3** ±6.8	33.3 ±8.1	33.3 ±5.0	25.3 ±0.9	**43.4** ±4.6
OGBench-Cube										
Random Policy					53.3 ±6.2					
LeWM	80.7 ±6.2	46.0 ±4.3	50.0 ±5.7	47.3 ±3.8	49.3 ±4.1	48.0 ±7.5	50.0 ±1.6	44.0 ±1.6	52.7 ±6.6	52.0 ±4.4
Sub-JEPA	66.7 ±5.0	50.0 ±2.8	48.0 ±5.9	50.7 ±6.6	47.3 ±2.5	48.0 ±2.8	46.7 ±2.5	49.3 ±4.1	45.3 ±2.5	50.2 ±2.7
TC-LeWM	62.0 ±8.6	55.3 ±6.8	52.0 ±8.2	55.3 ±9.0	56.7 ±9.0	54.0 ±9.1	51.3 ±5.7	52.7 ±1.9	55.3 ±10.9	55.0 ±7.4
DINOv2-WM	81.3 ±5.7	52.7 ±4.1	51.3 ±6.8	52.7 ±5.7	50.0 ±3.3	50.7 ±4.7	50.0 ±3.3	54.0 ±2.8	53.3 ±5.7	55.1 ±4.3
ConfigWM	**83.3** ±8.4	**74.0** ±5.9	**67.3** ±8.4	**64.7** ±10.5	**68.0** ±9.9	**68.0** ±8.6	**64.7** ±11.8	**65.3** ±10.9	**68.0** ±8.5	**69.3** ±9.1
Unified ConfigWM	76.0 ±1.6	61.3 ±8.1	58.7 ±7.7	59.3 ±6.8	58.0 ±4.3	54.0 ±7.1	55.3 ±6.8	55.3 ±1.9	57.3 ±9.6	59.5 ±5.5

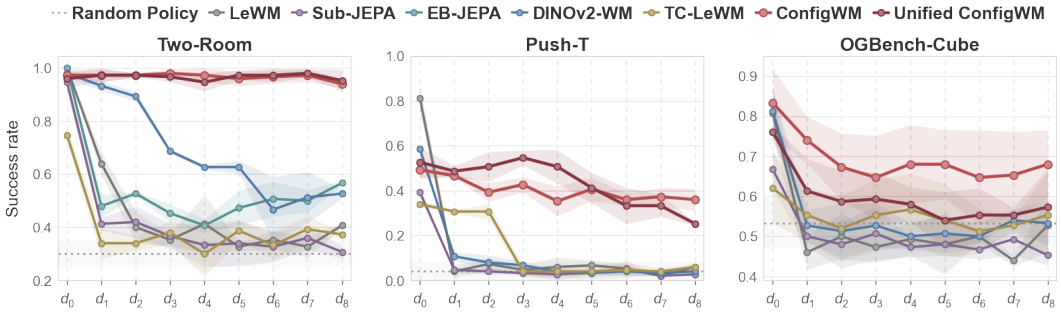

Figure 12: **Matched planning performance across distractor levels.** Success rate (%) under the matched setting, where each model is trained and evaluated at the same distractor level ($d_k \to d_k$). Results are averaged over seeds 40/42/45, with shaded regions indicating standard deviation.

Table 4: **Clean-trained transfer performance.** Each model uses its d_0-trained checkpoint and is evaluated across d_0–d_8 ($d_0 \rightarrow d_k$). Values follow the same reporting conventions as Table 3.

Method	Evaluation Distractor Level									Average
	d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	
Two-Room										
Random Policy	30.0 ±4.9									
LeWM	**100.0** ±0.0	72.7 ±1.9	66.0 ±4.9	58.7 ±5.2	51.3 ±5.0	50.7 ±3.4	56.7 ±6.6	52.0 ±7.5	50.0 ±5.7	62.0 ±3.3
Sub-JEPA	94.7 ±3.4	60.7 ±1.9	45.3 ±12.5	53.3 ±7.4	45.3 ±3.4	44.0 ±5.7	44.7 ±6.6	46.7 ±7.5	46.0 ±7.5	53.4 ±4.4
TC-LeWM	74.7 ±2.5	43.3 ±0.9	38.7 ±7.4	32.7 ±4.1	34.7 ±0.9	30.0 ±5.9	31.3 ±8.1	31.3 ±5.7	27.3 ±1.9	38.2 ±0.5
EB-JEPA	**100.0** ±0.0	58.0 ±7.1	43.3 ±12.3	43.3 ±5.0	42.7 ±5.0	42.7 ±2.5	37.3 ±2.5	38.0 ±5.9	38.0 ±3.3	49.3 ±4.0
DINOv2-WM	98.0 ±1.6	47.3 ±6.8	46.0 ±3.3	35.3 ±9.3	36.0 ±2.8	38.0 ±4.9	38.7 ±4.7	44.7 ±3.4	40.7 ±5.0	47.2 ±3.2
ConfigWM	97.3 ±0.9	**98.0** ±1.6	**98.0** ±1.6	**96.7** ±0.9	**98.0** ±1.6	**98.0** ±1.6	**98.0** ±1.6	**97.3** ±0.9	**96.7** ±0.9	**97.6** ±1.2
Unified ConfigWM	96.0 ±3.3	96.7 ±3.4	96.0 ±4.3	96.0 ±2.8	96.0 ±2.8	94.0 ±1.6	94.7 ±3.4	94.7 ±3.4	95.3 ±2.5	95.5 ±3.0
Push-T										
Random Policy	4.0 ±4.3									
LeWM	**81.3** ±5.7	22.0 ±3.3	15.3 ±3.4	11.3 ±1.9	8.7 ±2.5	7.3 ±0.9	9.3 ±0.9	6.7 ±0.9	2.7 ±0.9	18.3 ±0.4
Sub-JEPA	39.3 ±1.9	5.3 ±3.4	4.0 ±2.8	3.3 ±2.5	3.3 ±0.9	2.0 ±1.6	3.3 ±0.9	4.0 ±1.6	2.0 ±1.6	7.4 ±0.4
TC-LeWM	34.0 ±1.6	27.3 ±5.0	26.7 ±3.4	23.3 ±3.8	22.7 ±5.0	24.7 ±1.9	23.3 ±4.1	22.0 ±3.3	20.0 ±7.5	24.9 ±3.6
DINOv2-WM	58.7 ±0.9	12.0 ±3.3	2.7 ±0.9	5.3 ±1.9	2.7 ±1.9	2.7 ±0.9	1.3 ±0.9	1.3 ±0.9	2.0 ±1.6	9.9 ±0.7
ConfigWM	49.3 ±4.1	41.3 ±7.4	40.0 ±2.8	43.3 ±3.4	40.7 ±2.5	41.3 ±0.9	41.3 ±6.2	36.7 ±3.4	36.7 ±2.5	41.2 ±2.5
Unified ConfigWM	53.3 ±6.2	**52.0** ±5.7	**48.0** ±4.9	**51.3** ±4.1	**54.7** ±4.1	**55.3** ±0.9	**50.7** ±6.6	**50.7** ±6.6	**40.7** ±5.2	**50.7** ±4.0
OGBench-Cube										
Random Policy	53.3 ±6.2									
LeWM	80.7 ±6.2	45.3 ±4.1	47.3 ±4.1	47.3 ±6.8	46.7 ±4.7	49.3 ±3.8	47.3 ±2.5	46.0 ±4.9	44.7 ±5.0	50.5 ±4.6
Sub-JEPA	66.7 ±5.0	45.3 ±3.8	50.0 ±7.5	54.0 ±5.9	47.3 ±5.2	48.0 ±1.6	45.3 ±2.5	48.7 ±3.4	46.0 ±3.3	50.1 ±3.9
TC-LeWM	62.0 ±8.6	52.7 ±5.0	50.7 ±4.1	50.7 ±8.2	51.3 ±4.1	46.0 ±5.9	49.3 ±5.2	50.7 ±7.5	48.0 ±2.8	51.3 ±5.5
DINOv2-WM	81.3 ±5.7	54.0 ±4.9	53.3 ±6.8	50.7 ±3.4	50.7 ±4.1	52.7 ±3.4	54.7 ±7.4	55.3 ±5.2	56.7 ±6.2	56.6 ±4.8
ConfigWM	**83.3** ±8.4	**74.0** ±9.1	**68.7** ±10.0	**68.0** ±8.2	**66.7** ±7.5	**65.3** ±10.9	**68.7** ±6.2	**67.3** ±10.9	**68.7** ±10.4	**70.1** ±8.7
Unified ConfigWM	74.7 ±7.7	60.0 ±9.8	58.0 ±4.3	60.7 ±6.6	60.7 ±5.0	58.7 ±8.2	58.0 ±5.7	58.7 ±5.0	56.7 ±8.2	60.7 ±6.6

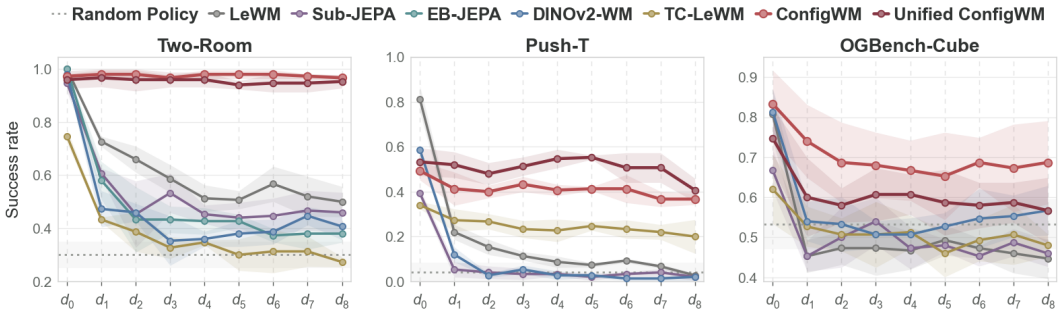

Figure 13: **Generalization from clean training to dynamic distractors.** Each world model is trained only on clean observations (d_0) and evaluated on d_0–d_8. The three panels correspond to Two-Room, Push-T, and OGBench-Cube. Curves report mean success rate across evaluation seeds 40, 42, and 45. Higher is better.

C.2 Quantity-Wise Physical Probing Results

For each model and distractor level, we freeze the representation model and extract the 192-dimensional states supplied to the dynamics predictor. Episodes are split into disjoint probe-training and probe-test partitions using a fixed $90/10$ split, from which we sample 1000 and 400 frames, respectively. The same split and sampled frames are used for all methods. For each physical quantity, we standardize the representations and targets using probe-training statistics and fit a separate linear probe using ridge regression with a coefficient of 10^{-6}. We report Pearson correlation r on the probe-test set, averaging across coordinates for vector-valued quantities. The macro-averaged results are shown in Fig. 6, with quantity-wise results reported below.

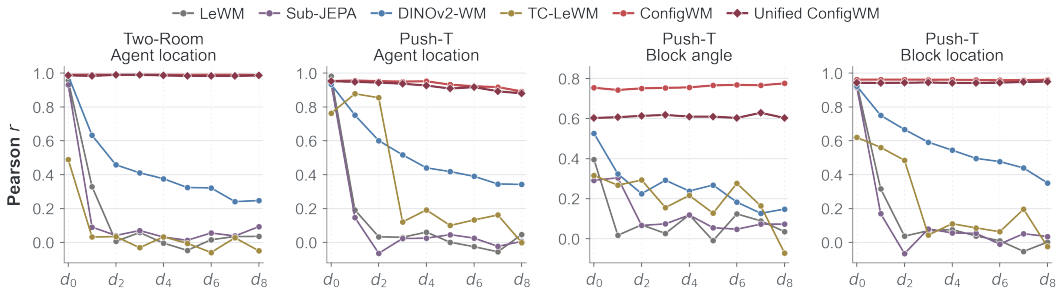

Figure 14: **Quantity-wise physical probing on Two-Room and Push-T.** Each panel reports the Pearson correlation (r) of a linear probe trained on frozen representations for one task-relevant physical quantity across distractor levels d_0–d_8. Higher is better.

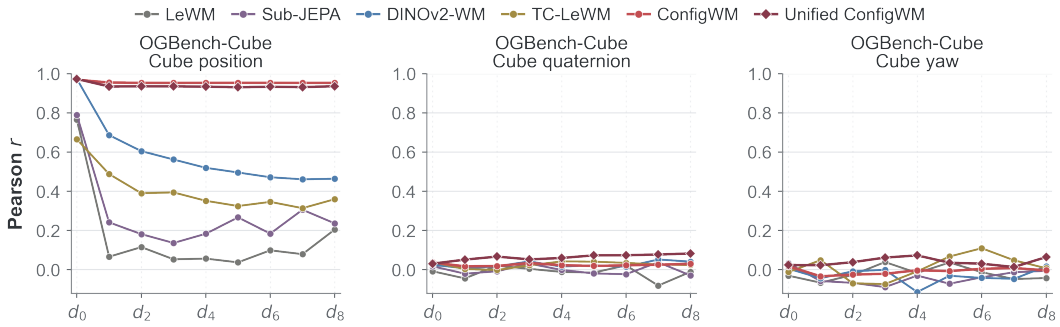

Figure 15: **Quantity-wise physical probing on OGBench-Cube: cube state.** Pearson correlation (r) is reported across distractor levels d_0–d_8 for cube position, quaternion, and yaw. Higher is better.

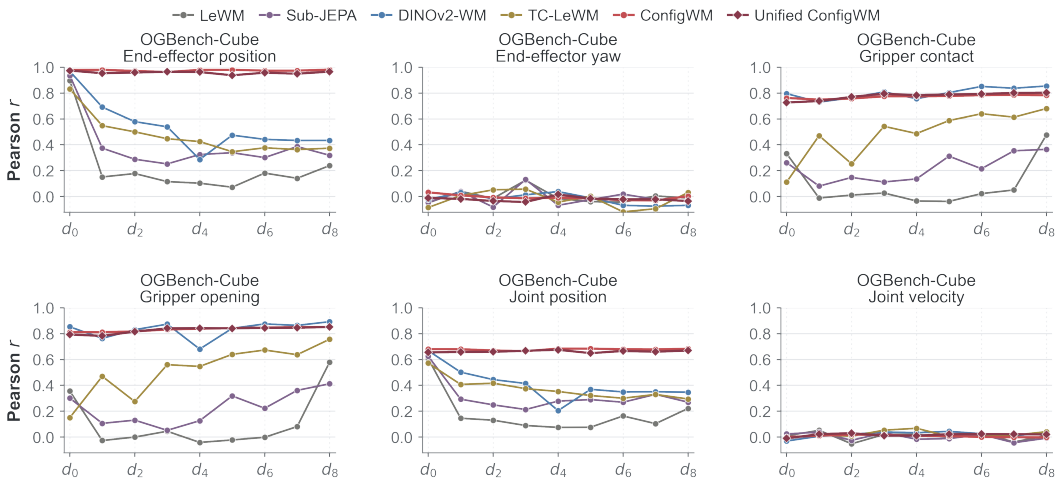

Figure 16: **Quantity-wise physical probing on OGBench-Cube.** Pearson correlation (r) is reported across distractor levels d_0–d_8. Top: end-effector position, end-effector yaw, and gripper contact. Bottom: gripper opening, joint position, and joint velocity. Higher is better.

C.3 Detailed Clean–Distracted Retrieval Results

For each model and distractor level, we sample 400 paired distracted and clean predictor-input states from matching episode and frame positions. Each distracted state is used as a query over the 400 clean candidates, which are ranked by cosine similarity. Top-k retrieval is successful when the corresponding clean state appears among the top-k candidates. We also report mean reciprocal rank, $\mathrm{MRR} = \frac{1}{400}\sum_{i=1}^{400} 1/\,\mathrm{rank}_i$, where rank_i is the rank of the paired clean state for query i. The same sampled pairs are used for all methods. Top-1 results are reported in Fig. 6, with Top-5 and MRR results shown below.

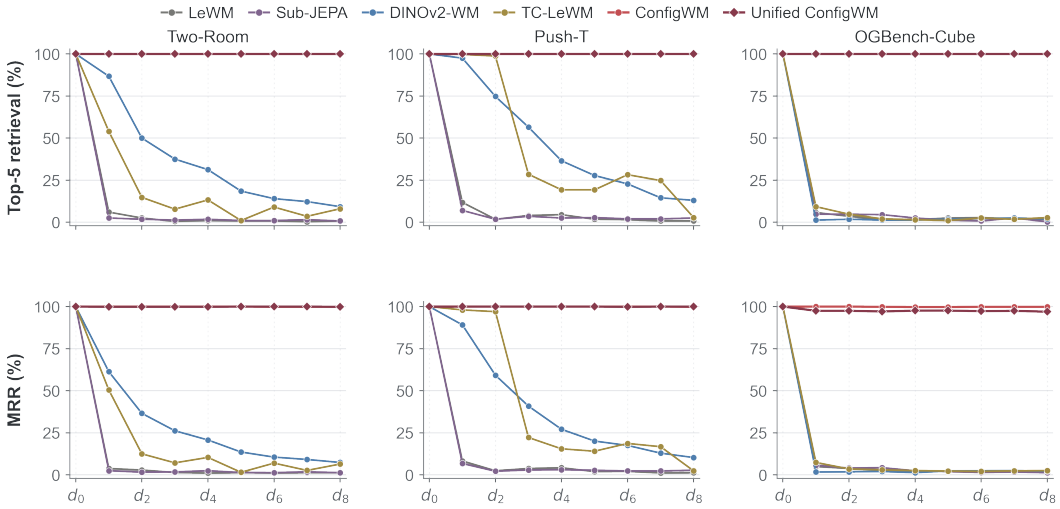

Figure 17: **Detailed distracted-to-clean retrieval results.** Top: Top-5 retrieval accuracy. Bottom: mean reciprocal rank (MRR). Each distracted state is queried against 400 clean candidates ranked by cosine similarity. Results are reported across distractor levels d_0–d_8; higher is better.

C.4 Detailed Query-Conditioning Results

Figure 18 presents additional decoding results obtained by varying only the task query while keeping the input observation fixed. For each environment, we freeze the configurable encoder and train one ViT patch decoder shared across all query variants. The decoder contains six Transformer blocks with width 768 and 12 attention heads and reconstructs 224×224 RGB observations from the predictor outputs. It is trained for one epoch with a global batch size of 256 in BF16. Because the decoder receives no direct access to the query, query-dependent reconstructions reflect information already encoded in the predictive state.

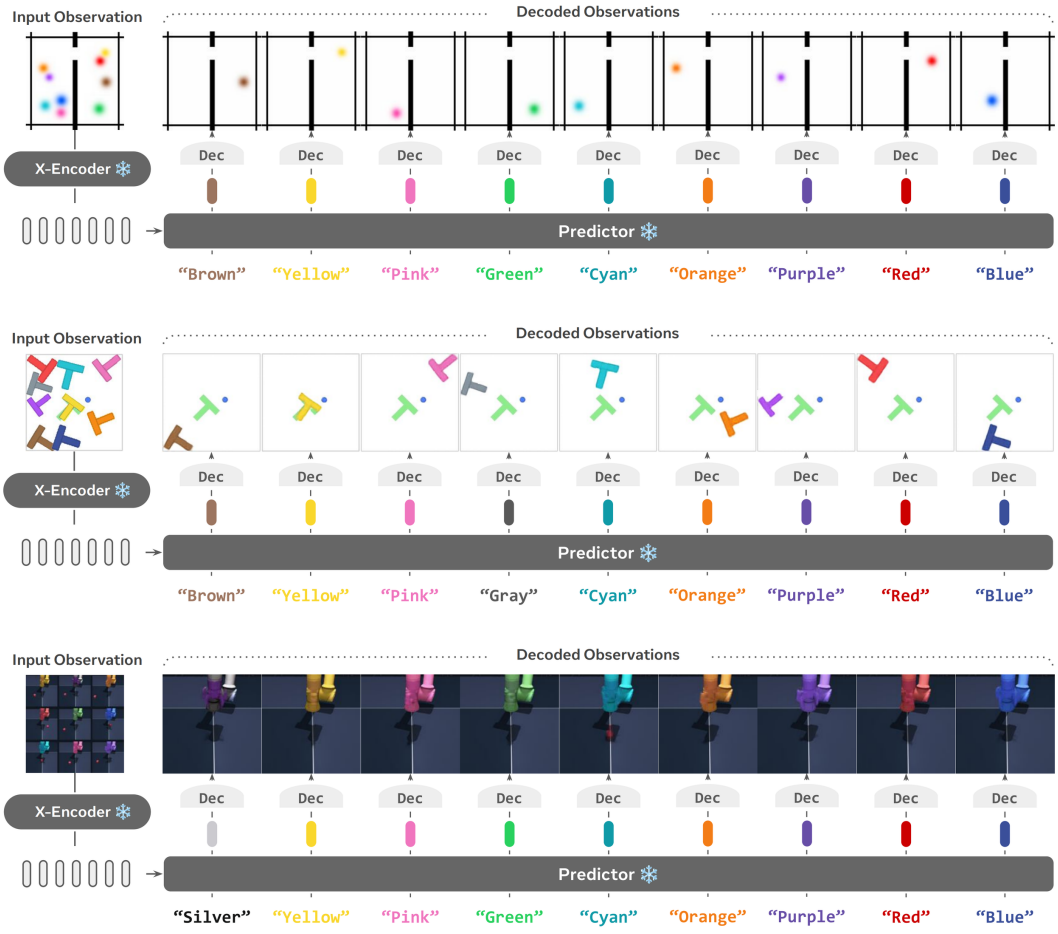

Figure 18: **Additional query-conditioned decoding results.** From top to bottom, the three rows show Two-Room, Push-T, and OGBench-Cube. Within each environment, the input observation is held fixed while only the queried color is varied. The decoded observations show that different queries retain different visual content in the predictive state.

C.5 Detailed Query-Conditioned Planning Results

To complement the aggregate query-mismatch results in Fig. 8, Table 5 reports performance at each distractor level, while Table 6 further decomposes the results by individual mismatched query. All experimental settings other than the task query are held fixed. The correct-query results are independently re-evaluated in this experiment; small differences from Table 3 arise from stochastic CEM planning.

Table 5: **Complete query-conditioned planning results.** Mean success rate (%) over three seeds at each distractor level; light-blue tags show standard deviation. Mismatched results average eight valid queries within each seed. Drop is computed from the unrounded seed-level results.

Level	Correct query	Mismatched-query average	Drop
	Two-Room		
d_1	96.7 ± 3.4	45.7 ± 2.3	51.0 ± 5.5
d_2	96.7 ± 1.9	41.3 ± 3.7	55.4 ± 4.3
d_3	97.3 ± 1.9	39.3 ± 3.0	58.1 ± 4.9
d_4	97.3 ± 1.9	33.4 ± 2.6	63.9 ± 4.5
d_5	96.0 ± 2.8	32.8 ± 3.7	63.2 ± 4.6
d_6	96.7 ± 2.5	31.0 ± 2.4	65.7 ± 4.5
d_7	97.3 ± 2.5	28.3 ± 2.1	69.0 ± 3.6
d_8	94.7 ± 2.5	24.9 ± 2.9	69.7 ± 5.3
Avg.	96.6 ± 2.1	34.6 ± 2.7	62.0 ± 4.5
	Push-T		
d_1	46.7 ± 0.9	11.5 ± 0.5	35.2 ± 0.4
d_2	39.3 ± 4.1	11.8 ± 3.9	27.6 ± 8.0
d_3	42.7 ± 3.4	13.2 ± 3.5	29.5 ± 5.6
d_4	35.3 ± 6.6	11.9 ± 1.8	23.4 ± 6.5
d_5	40.7 ± 5.2	10.2 ± 2.2	30.5 ± 3.7
d_6	36.0 ± 2.8	9.2 ± 1.5	26.8 ± 1.5
d_7	37.3 ± 4.1	8.7 ± 1.8	28.7 ± 3.7
d_8	36.0 ± 4.9	7.2 ± 1.7	28.8 ± 3.9
Avg.	39.2 ± 2.6	10.4 ± 2.1	28.8 ± 2.2
	OGBench-Cube		
d_1	74.0 ± 5.9	56.1 ± 5.6	17.9 ± 2.5
d_2	67.3 ± 8.4	52.4 ± 4.6	14.9 ± 4.7
d_3	64.7 ± 10.5	50.4 ± 3.7	14.2 ± 6.8
d_4	68.0 ± 9.9	48.3 ± 2.7	19.7 ± 7.6
d_5	68.0 ± 8.6	47.8 ± 3.3	20.2 ± 5.3
d_6	64.7 ± 11.8	48.6 ± 3.7	16.1 ± 8.2
d_7	65.3 ± 10.9	48.6 ± 4.1	16.7 ± 7.1
d_8	68.0 ± 8.5	46.9 ± 3.0	21.1 ± 5.6
Avg.	67.5 ± 9.2	49.9 ± 3.7	17.6 ± 5.5

Table 6: **Per-query planning performance under mismatched queries.** Mean success rate (%) over d_1–d_8 and three seeds; light-blue tags show standard deviation. Drop is computed from the unrounded results.

Environment	Correct query	Mismatched query	Avg. d_1–d_8	Drop
Two-Room	Red ball	Blue ball	29.4 ± 0.7	67.2 ± 1.7
		Brown ball	32.4 ± 3.2	64.2 ± 5.3
		Cyan ball	29.8 ± 3.1	66.8 ± 4.6
		Green ball	35.4 ± 4.7	61.2 ± 6.3
		Orange ball	46.6 ± 2.6	50.0 ± 4.7
		Pink ball	39.0 ± 6.8	57.6 ± 7.9
		Purple ball	31.9 ± 2.0	64.7 ± 3.6
		Yellow ball	32.1 ± 1.6	64.5 ± 3.7
Push-T	Gray T block	Brown T block	10.1 ± 4.1	29.2 ± 2.9
		Cyan T block	10.4 ± 1.8	28.8 ± 0.8
		Navy T block	11.7 ± 0.7	27.6 ± 2.6
		Orange T block	11.0 ± 2.4	28.2 ± 0.6
		Pink T block	11.2 ± 4.4	28.0 ± 4.1
		Purple T block	10.2 ± 2.7	29.1 ± 4.2
		Red T block	7.8 ± 1.4	31.4 ± 2.6
		Yellow T block	11.2 ± 2.6	28.1 ± 2.8
OGBench-Cube	Silver arm	Blue arm	50.4 ± 3.0	17.1 ± 6.4
		Cyan arm	51.2 ± 3.8	16.3 ± 5.4
		Green arm	50.7 ± 3.4	16.8 ± 5.9
		Orange arm	49.2 ± 3.9	18.3 ± 5.7
		Pink arm	48.0 ± 3.4	19.5 ± 5.9
		Purple arm	50.8 ± 4.1	16.7 ± 5.2
		Red arm	48.1 ± 4.8	19.4 ± 5.5
		Yellow arm	50.8 ± 4.4	16.7 ± 4.8

C.6 Qualitative Planning Rollouts

We visualize complete real-environment rollouts under clean observations and severe distractors. For each environment, we compare ConfigWM with the best-performing baseline in the matched planning results: DINOv2-WM for Two-Room and OGBench-Cube, and LeWM for Push-T. Frames are sampled at normalized rollout times; the final column shows a condition-matched goal observation and is not part of the rollout.

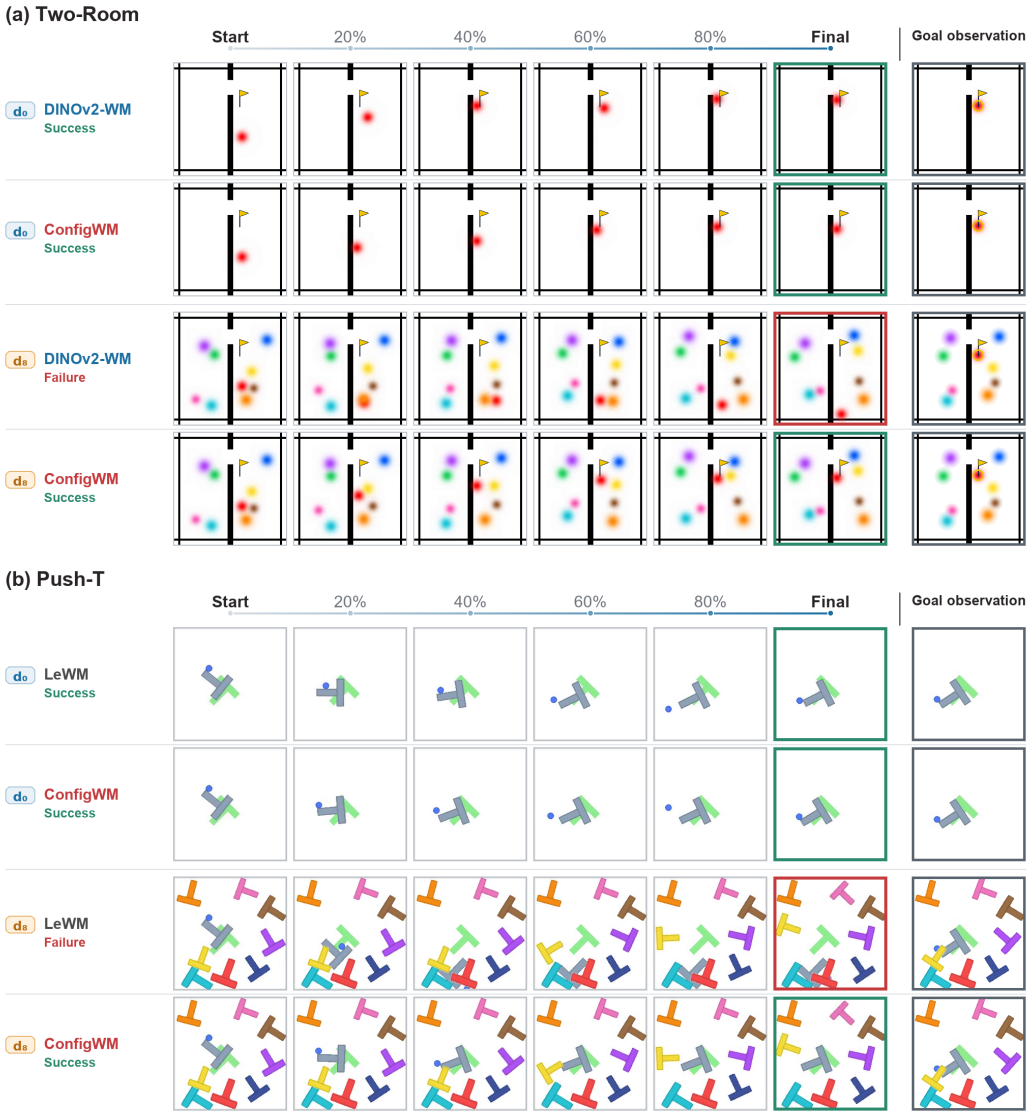

Figure 19: **Qualitative planning rollouts on Two-Room and Push-T.** Both methods complete the task under clean observations (d_0). Under severe distractors (d_8), ConfigWM reaches the goal, whereas the corresponding comparison model fails.



Figure 20: **Qualitative planning rollouts on OGBench-Cube.** Both models complete the task under clean observations (d_0). Under severe distractors (d_8), ConfigWM reaches the goal, whereas DINOv2-WM fails. The yellow box identifies the controlled task view.

D SCALING THE CONFIGURABLE ENCODER WITH REAL-WORLD DATA

To assess whether the configurable encoder extends beyond controlled environments, we conduct an additional representation-learning study using diverse real-world video data and four large visual backbones, demonstrating that the framework can be readily instantiated with a range of widely used visual encoders.

Real-World Data Construction. We draw candidate cases from task-annotated clips in public datasets spanning three domains: egocentric video (e.g., Ego4D (Grauman et al., 2022), Ego-Exo4D (Grauman et al., 2024), and EPIC-KITCHENS-100 (Damen et al., 2022)), robotic manipulation (e.g., DROID (Khazatsky et al., 2024), Open X-Embodiment (O'Neill et al., 2024), and RoboMind (Wu et al., 2024)), and instructional video (e.g., ActivityNet (Caba Heilbron et al., 2015), COIN (Tang et al., 2019), and FineGym (Shao et al., 2020)). For each case, we sample frames as the clean target observation and its task description as the query.

To construct paired noisy and clean observations, we use a task-conditioned distractor-synthesis pipeline based on VLM-guided image editing and differential quality control. Given a clean observation and its task query, a VLM (Qwen3-VL-32B-Instruct (Bai et al., 2025)) identifies the task-critical entities and proposes one or two visually plausible but task-irrelevant objects. It then produces a structured edit instruction specifying the appearance and placement of each addition, which is executed by an image-editing model (Qwen-Image-Edit-2511 (Wu et al., 2025)) to obtain the noisy input observation. A second pass with Qwen3-VL-32B-Instruct compares the edited and clean observations and rejects edits that alter task-relevant content or scene geometry.

Training and Qualitative Evaluation. We retain the configurable encoder architecture and training objective described in Section B.1, and instantiate it with four large visual backbones: DINOv2-Giant (Oquab et al., 2023), DINOv3-7B (Siméoni et al., 2025), V-JEPA 2.1-Gigantic (Mur-Labadia et al., 2026), and LingBot-Vision-Giant (Fu et al., 2026b). All variants produce a 64×64 patch grid. We train each variant for 1000 steps on $8\times$ NVIDIA A100 GPUs using AdamW with a learning rate of 10^{-5}, weight decay of 0.05, 50 warm-up steps followed by cosine decay, a global batch size of 32, BF16 precision, and gradient clipping at 1.0. For qualitative evaluation, we apply the trained models to held-out cases and visualize PCA projections of the frozen-backbone and our

representations. The results below examine whether ours suppresses the injected distractors while preserving task-relevant states.

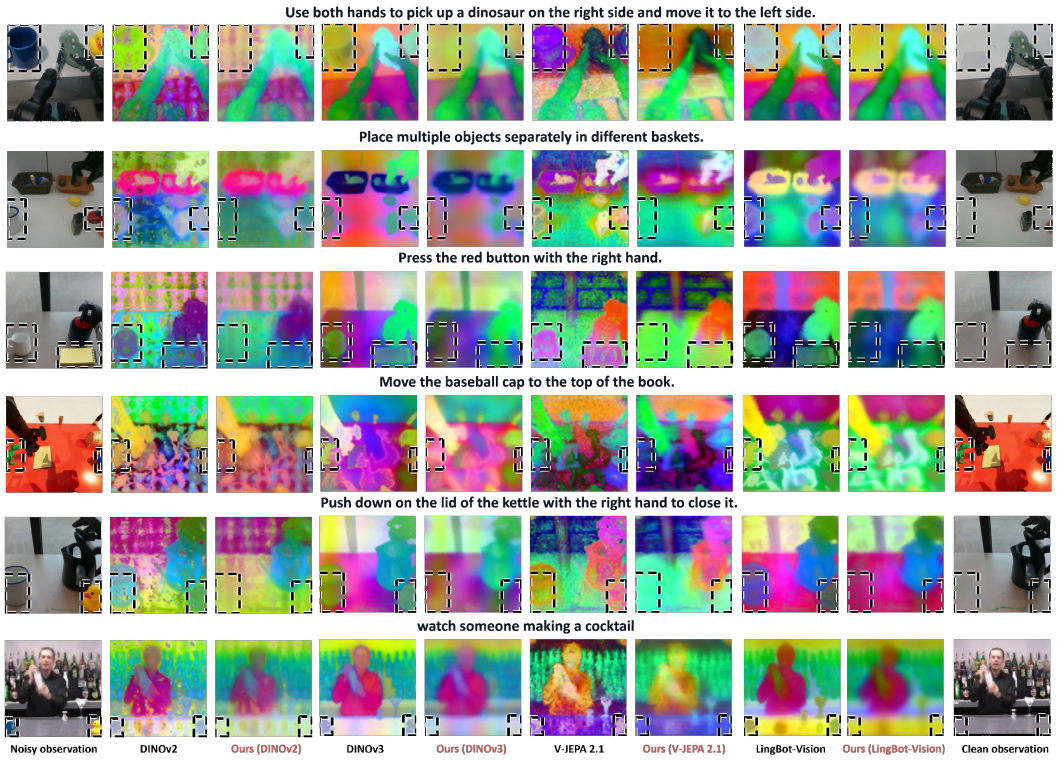

Figure 21: **Configurable encoder across visual backbones.** The configurable encoder suppresses feature regions associated with the injected distractors while retaining the task-relevant entities and spatial structure visible in the clean target. Six held-out cases are shown across DINOv2-Giant, DINOv3-7B, V-JEPA2.1-Gigantic, and LingBot-Vision-Giant. Black dashed boxes mark distractors.

Quantitative Evaluation. We quantitatively evaluate whether the configurable encoder can effectively suppress distractors while preserving task-relevant states across different visual backbones. We use 300 cases, with 80% cases for training simple linear classifiers and 20% cases for testing. As shown in Table 7, the results suggest that the configurable encoder trained with different backbones consistently suppresses distractors while largely preserving task-relevant regions.

Table 7: Quantitative analysis of real-world representations across large visual backbones. The configurable encoder reduces distractor decodability while largely preserving task-relevant spatial information. We report patch-level linear-probing AP across DINOv2-Giant, DINOv3-7B, V-JEPA2.1-Gigantic, and LingBot-Vision-Giant.

	Frozen backbone		The configurable encoder (Ours)	
Backbone	Distractor AP ↓	Task-core AP ↑	Distractor AP ↓	Task-core AP ↑
DINOv2-Giant	0.523	0.603	**0.377**	0.594
DINOv3-7B	0.575	0.609	**0.367**	0.596
V-JEPA 2.1-Gigantic	0.544	0.605	**0.376**	0.602
LingBot-Vision-Giant	0.561	0.621	**0.368**	0.603

Data Construction Prompts. We provide the full prompts used to generate structured edit instructions and produce the corresponding noisy observations.

Edit-Instruction Generation Prompt *Qwen3-VL-32B-Instruct*

Clean Observation:
`{clean_observation}`
Task Description:
`{task_description}`

1. **Objective.** Plan synthetic visual distractors for training a model to recover the untouched clean input image. Given one image and one task description, choose one or two new task-irrelevant objects and write an image-editing instruction. Return only valid JSON matching the specified format.
2. **Identify protected content.** Identify all visible task-critical entities, including the target, destination, tool, acting hand or robot, directly involved container, contact region, action path, and support surfaces needed to understand the task. All identified entities must be protected.
3. **Select task-irrelevant distractors.** Every distractor must be unrelated to the task. Never add another instance, substitute, semantic alternative, or look-alike of a task entity. Prefer an ordinary, scene-plausible object whose category, shape, color role, and function are visibly distinct from all salient original objects. Use one distractor in dense scenes or when only one isolated empty region exists. Use two only when the visible support contains two clearly separated empty regions.
4. **Determine safe placements.** Place additions on visible existing support surfaces in genuinely empty regions. They must not overlap, touch, cover, replace, move, recolor, resize, or merge with any original entity. Never place an addition inside the task area, between an actor and its target, or along the task motion path.
5. **Construct the editing instruction.** The `edit_prompt` must contain exactly three concise sentences:
   - **(a)** Start with "`Add only these new task-irrelevant objects:`" and specify the appearance and realistic size of each object.
   - **(b)** Start with "`Place them separately`" and specify the existing support surface and an unambiguous empty relative location for each object. Explicitly forbid touching or overlapping any original object.
   - **(c)** Start with "`Preserve the input image exactly`" and require every original pixel and object to remain unchanged in identity, count, color, pose, position, scale, contact, shadow, geometry, exposure, camera view, framing, and spatial relationship.
6. **Construct the negative prompt.** The `negative_prompt` must forbid cropping, zooming, recentering, restaging, reconstruction, deletion, replacement, relocation, recoloring, duplicate additions, unrequested objects, occlusion, overlap, floating objects, altered support surfaces, altered hands or robots, and changed task relationships.
7. **Return JSON only.**

```
{
  "task_entities": [
    "..."
  ],
  "protected_entities": [
    "..."
  ],
  "distractors": [
    {
      "name": "...",
      "appearance": "...",
      "task_irrelevance": "...",
      "support": "...",
      "placement": "..."
    }
  ],
  "edit_prompt": "...",
  "negative_prompt": "..."
}
```

Image-Editing Prompt *Qwen-Image-Edit-2511*

Input image.
`{clean_observation}`

1. **Add the distractors.** Add only these new task-irrelevant objects: `{distractor names, appearances, and realistic sizes}`.
2. **Place the distractors.** Place them separately at `{support surfaces and relative locations produced by the VLM}`, without touching or overlapping any original object.
3. **Preserve the source observation.** Preserve the input image exactly. Keep every original pixel and object unchanged in identity, count, color, pose, position, scale, contact, shadow, geometry, exposure, camera view, framing, and spatial relationship.