BEYOND VISUAL FIDELITY: COMPARING CATEGORICAL AND CONTINUOUS WORLD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Visual world models can now generate compelling interactive worlds, but visual realism does not guarantee faithful dynamics. To understand how modeling choices affect world model behavior, we present a systematic comparison that disentangles prediction representation from generation procedure, choices often bundled in approaches such as autoregressive token models and whole-frame flow matching. In controlled 2D and 3D environments, we evaluate dynamics validity, calibration, and visual quality. Failures such as disappearing game characters echo those observed in more complex environments, but here their impact can be measured against known dynamics. With lossless visual encoding in 2D, discrete-token models achieve higher rollout validity at lower training compute, particularly for stochastic entities. However, high validity can coexist with poor calibration: generating permissible trajectories does not imply reproducing their probabilities. In 3D, visual degradation from lossy tokenization does not consistently translate into worse long-horizon metrics. These results show how representation and decoding choices affect validity and calibration differently, demonstrating the importance of evaluating world models beyond visual fidelity.

1 INTRODUCTION

Recent advances in video generation have made visual world models, i.e., learned simulators of how environments evolve under actions, promising tools for planning, policy learning, and interactive applications (Du et al., 2023; Alonso et al., 2024; Kanervisto et al., 2025; Valevski et al., 2025). However, many applications require more than visual realism: generation must respond faithfully to actions and preserve the underlying dynamics. How best to learn these dynamics remains unclear.

Approaches to learning these dynamics include autoregressive prediction of discrete visual tokens (categorical prediction), building on language-model architectures (Vaswani et al., 2017), and continuous generation methods such as flow matching (Lipman et al., 2023). Continuous approaches are often preferred for visual generation because token-based models typically rely on lossy encodings that can limit visual quality. Research on narrowing this gap (Tian et al., 2024; Sun et al., 2024; Yu et al., 2024) shows that improvements in tokenization can change their relative performance. Yet visually compelling predictions can still violate physical interactions, change object state incorrectly, or lose consistency over time (Liu et al., 2024). Mousakhan et al. (2026) directly compare categorical and continuous world models using a shared hybrid tokenizer, finding an advantage for continuous modeling in driving scenes. Yet tokenization-induced differences in visual fidelity make it difficult to attribute this advantage to dynamics modeling alone. We therefore ask: how do categorical and continuous prediction compare when visual encoding introduces no information loss?

We address this question through a controlled comparison that separates prediction representation from within-frame generation procedure. Both categorical and continuous predictions can be generated sequentially, patch by patch (Micheli et al., 2023; Li et al., 2024), or jointly through iterative updates (Bruce et al., 2024; Alonso et al., 2024). Flow matching, for example, can generate either individual patches or a complete frame. Our resulting four-model design crosses these choices (Figure 1). This allows us to examine how each modeling and representation choice affects model performance. We study these choices in two complementary environments. A Pac-Man-inspired environment admits lossless encoding and lets us vary the number and behavior of interacting entities. Its known rules allow us to assess whether generated transitions and trajectories are permissible,

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

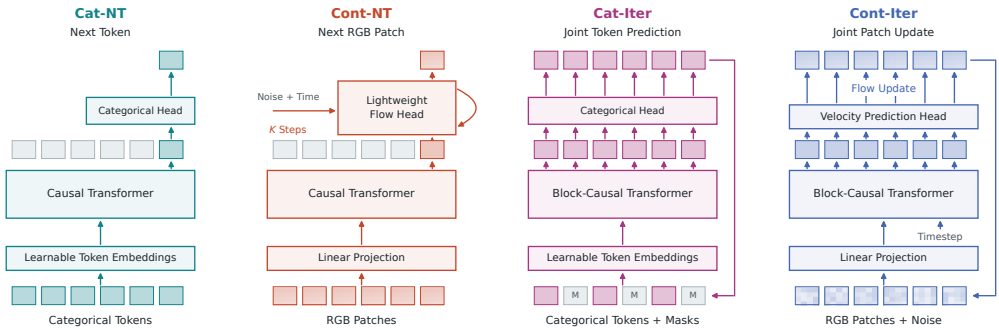


Figure 1: **Overview of the four core architectures evaluated in this work.** We follow a controlled 2×2 design that crosses categorical versus continuous prediction with next-token versus iterative whole-frame generation, enabling systematic comparison of these two modeling choices.

distinguishing violations of the dynamics from legitimate stochastic alternatives without requiring agreement with a single reference. A partially observed 3D maze extends the comparison to examine how these findings change in 3D spaces and when learned lossy tokenization is introduced.

Contributions. Our central result is that, with lossless visual encoding in Pac-Man, categorical world models sustain valid trajectories at lower training compute than continuous models, with largest differences in performance arising from stochastic dynamics. These failures echo reports of disappearing or temporally inconsistent non playable entities in large, high-fidelity diffusion and flow-based world models Alonso et al. (2024); Parker-Holder et al. (2024), but our controlled setting makes their frequency and consequences measurable. The categorical advantage does not extend uniformly to reproducing outcome probabilities: continuous models can outperform iterative categorical prediction in calibration despite weaker validity. This distinguishes learning which futures are possible from learning how likely they are. The 3D maze experiments further separate reconstruction fidelity from rollout performance: lossy tokenization reduces next-frame visual accuracy without consistently degrading long-horizon semantic or video-level metrics. Decoding choices also matter, with sampling improving maze Fréchet Video Distance (FVD) while greedy decoding favors Pac-Man validity. Together, these results show why representation and generation choices must be assessed against the properties required by the intended application, and how controlled environments complement large-scale evaluation, which is typically driven by visual quality.

2 Related Work

Discrete and continuous world models. Discrete-token world models (Kanervisto et al., 2025; Guo et al., 2025; Bruce et al., 2024) demonstrate controllable interactive generation using categorical sequence models. In contrast, many recent frontier systems favor continuous diffusion-based generation (Zhang et al., 2025; Parker-Holder et al., 2024; Alonso et al., 2024; Valevski et al., 2025), motivated in part by improved visual fidelity and the avoidance of lossy quantization. Together, these works highlight a potential trade-off between visually faithful observations and stable, controllable dynamics over long rollouts.

Comparing output parameterizations. Mousakhan et al. (2026) provide the closest direct comparison between continuous and discrete world models, finding that continuous flow matching outperforms discrete autoregression in visually complex driving scenes. Conversely, image-generation studies show that discrete models can match or outperform diffusion with sufficiently strong tokenizers and architectures (Sun et al., 2024; Yu et al., 2024; Tian et al., 2024). These end-to-end comparisons couple output parameterization with tokenization, architecture, model scale, and compute, making their individual contributions difficult to distinguish. We address this gap by comparing both formulations under lossless visual encoding settings, within sequential and iterative generation procedures, and evaluating their predictions against known transition rules and probabilities.

Long-horizon stochastic dynamics. Evaluation of world models remains an emerging topic, with existing popular benchmarks often emphasizing perceptual quality over exact mechanics, entity persistence, or action-conditioned transition validity (Huang et al., 2024). Generated-context mismatch

can cause local errors to accumulate during autoregressive rollout (Valevski et al., 2025; Chen et al., 2025; Huang et al., 2025). However, how this accumulation differs between categorical and flow-matching models as the number and stochasticity of interacting entities increase remains unclear. We compare both formulations under lossless encoding, within sequential and iterative generation procedures, varying dynamics complexity and evaluating predictions against known transition rules and probabilities.

3 World Model Architectures For a Controlled Comparison

We compare four action-conditioned world models in a 2×2 design crossing output representation, namely categorical or continuous, with generation procedure, namely next-token prediction or iterative whole-frame generation (see Fig. 1 for a schematic overview). All use Transformer backbones (Vaswani et al., 2017). Next-token variants use pre-normalization decoder-only blocks (Radford et al., 2019; Xiong et al., 2020), while iterative models use DiT-style blocks (Peebles & Xie, 2023). We hold training data and optimization schedule fixed, and approximately match backbone capacity. Because the models require different input and output heads, enforcing identical total parameter counts would introduce additional architectural differences. We instead report performance as a function of cumulative training compute. Further implementation details are provided in App. A.

Observation and action representation. For each environment, an observation x_t is represented as a sequence of P image patches. Categorical models encode each patch as a token $z_{t,p} \in \{1, \dots, V\}$, where V is the categorical vocabulary size. Continuous models represent each patch by a vector $y_{t,p} \in \mathbb{R}^d$, where d is the continuous patch dimension and $p \in \{1, \dots, P\}$. Within each environment, the two categorical models share the same observation encoding, as do the two continuous models. Environment-specific encodings are described in Sections 4.1 and 4.2. All models predict x_{t+1} conditioned on a context history of at most L frames and actions $\mathcal{C}_t = (x_{t-L+1}, a_{t-L+1}, \dots, x_t, a_t)$, where a_t is the action leading from x_t to the target frame x_{t+1}. The context length L is fixed within each experiment. All models use learned absolute positional embeddings over their flattened observation–action sequences and are trained to predict the target frame from a ground-truth context (teacher-forcing). At inference time, generated frames are fed back into the temporal context.

3.1 Next-token World Models

We call "next-token models" those action-conditioned autoregressive world models that represent environment dynamics as next-token prediction (Micheli et al., 2023; Kanervisto et al., 2025), meaning that the next frame is generated as an ordered sequence of patches generated autoregressively. Our two variants use the same sequentially-ordered factorization with either categorical or continuous patch distributions:

$$p_\theta(x_{t+1} \mid \mathcal{C}_t) = \prod_{p=1}^{P} p_\theta(x_{t+1,p} \mid \mathcal{C}_t, x_{t+1,<p}), \tag{1}$$

where $x_{t+1,<p}$ denotes the target patches preceding position p. Both variants use the same decoder-only transformer backbone. Context patches and actions are interleaved in a single sequence, and each generated patch is appended to this sequence before the next patch is predicted.

Categorical next-token model. CAT-NT instantiates the causal factorization with categorical patch distributions. It embeds visual patches as categorical tokens. Discrete actions share the embedding table using a separate identifier range, whereas continuous actions are linearly projected into the hidden space. The visual embeddings are tied to the output classifier, and the model minimizes cross-entropy over the target-frame tokens:

$$\mathcal{L}_{\mathrm{CatNT}}(\theta) = -\sum_{p=1}^{P} \log p_\theta \left(z_{t+1,p} \mid \mathcal{C}_t, z_{t+1,<p} \right). \tag{2}$$

At inference, visual tokens are generated sequentially, either greedily or using categorical sampling.

Continuous next-token model. To our knowledge, no prior world model combines causal, patch-by-patch next-frame prediction with a separate continuous generative process for each patch. We

select this architecture to enable a controlled comparison between categorical and continuous heads. CONT-NT adapts per-token diffusion heads from autoregressive image and multimodal generation (Li et al., 2024; Sun et al., 2026) to the action-conditioned world-model setting, and replaces the diffusion objective with conditional flow matching. It encodes RGB patches into the transformer hidden space and uses learned embeddings for discrete actions or linear projections for continuous actions. For target position p, the transformer produces the conditioning vector $h_{t+1,p} = f_\theta(\mathcal{C}_t, y_{t+1,<p})$. A conditional flow head then generates the patch from Gaussian noise.

Following conditional flow matching (Lipman et al., 2023), let $\epsilon \sim \mathcal{N}(0, I)$ and $\tilde{y}_\tau = (1 - \sigma_\tau)y + \sigma_\tau\epsilon$, where σ_τ specifies the flow schedule. The target velocity with respect to the noise level is $\epsilon - y$. For CONT-NT, the clean sample is a target patch $y = y_{t+1,p}$ and the conditioning variable is $h = h_{t+1,p}$:

$$\mathcal{L}_{\text{ContNT}}(\theta, \phi) = \mathbb{E}_{\tau,\epsilon}\left[\|v_\phi(\tilde{y}_\tau, \tau \mid f_\theta(\mathcal{C}_t, y_{t+1,<p})) - (\epsilon - y_{t+1,p})\|_2^2\right]. \tag{3}$$

At inference, the velocity field is integrated from noise level one to zero separately for each patch. The resulting patch is encoded into the transformer hidden space and appended to the sequence before the next patch is generated. Thus, CAT-NT and CONT-NT share similar factorization and transformer backbone but differ in whether each conditional distribution is represented by categorical logits or a continuous flow.

3.2 ITERATIVE WHOLE-FRAME WORLD MODELS

Iterative whole-frame generation has been adopted in world models through masked categorical refinement (Bruce et al., 2024; Microsoft Research, 2025) and continuous diffusion or flow-based generation (Alonso et al., 2024; Valevski et al., 2025; Mousakhan et al., 2026). Rather than imposing an ordering within the target frame, they represent all target patches simultaneously and update them over multiple iterations. Both our iterative model variants use DiT-style blocks with block-causal temporal attention: patches interact bidirectionally within a frame but attend only to the current and preceding temporal blocks. The variants differ in their input and output layers, conditioning mechanisms, and update rules.

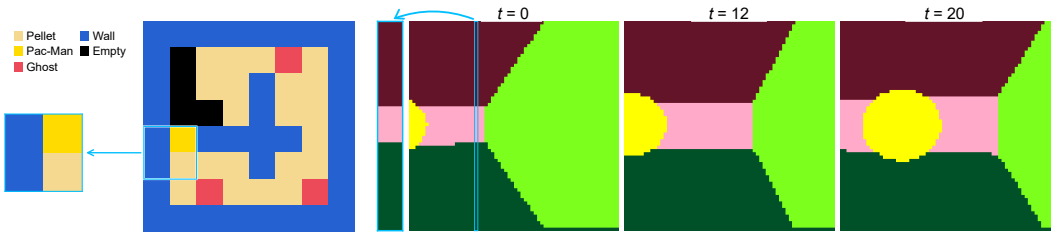
Categorical iterative model. CAT-ITER uses the MaskGIT training and decoding procedure (Chang et al., 2022), which has been adapted to world modeling in different ways. Genie applies masked prediction within a spatiotemporal dynamics model conditioned on learned latent actions (Bruce et al., 2024), whereas WHAMM! combines a temporal context model with lightweight iterative masked refinement of the next-frame tokens (Microsoft Research, 2025). Here, we use the same core masked-token principle while keeping the backbone and conditioning aligned with the controlled comparison.

During training, a random subset $\mathcal{M} \subseteq \{1, \ldots, P\}$ of target positions is replaced by a learned mask embedding. The model predicts the masked tokens conditioned on the context, action, and visible target tokens $z_{t+1,\overline{\mathcal{M}}}$, where $\overline{\mathcal{M}} = \{1, \ldots, P\} \setminus \mathcal{M}$:

$$\mathcal{L}_{\text{CatIter}}(\theta) = -\sum_{p \in \mathcal{M}} \log p_\theta\left(z_{t+1,p} \mid \mathcal{C}_t, z_{t+1,\overline{\mathcal{M}}}\right). \tag{4}$$

At inference, all target positions are initially masked. At each iteration, the model predicts the remaining tokens and permanently reveals the highest-confidence predictions according to a linear schedule, continuing until the target frame is complete.

Continuous iterative flow model. CONT-ITER instantiates the continuous iterative variant using a flow-matching objective. Continuous context patches and their corresponding actions are interleaved in the context sequence, and the noised target patches are appended as the newest temporal block. Flow-time embeddings modulate the DiT-style blocks and output layer through adaptive normalization. It uses the same conditional flow-matching objective as Equation 3, but applies it jointly to the complete patchified target frame $y_{t+1} = (y_{t+1,1}, \ldots, y_{t+1,P})$, so $\tilde{y}_\tau = (1 - \sigma_\tau)(y_{t+1,1}, \ldots, y_{t+1,P}) + \sigma_\tau\epsilon$. Conditioned on $\mathcal{C}_t$, its backbone predicts a coupled velocity for all target patches rather than a separate conditional flow for each patch. At inference, the complete target frame is initialized from Gaussian noise and all patches are updated jointly by integrating the learned velocity field. This contrasts with CONT-NT, which integrates a separate conditional flow for each patch and feeds the generated patches back sequentially.



(a) Pixel Pac-Man (b) 3D maze

Figure 2: **Visualisation of Pixel Pac-Man and 3D Maze environments**. Cyan outlines and arrows indicate extracted patches: a 2×2 tile patch in Pac-Man and a 64×1 pixel column in the maze.

4 EXPERIMENTS

In this section we describe our controlled toy environments and provide the main findings obtained with each of them. See other implementation, training and evaluation details in Appendix A.

4.1 "PIXEL PAC-MAN" CONTROLLED ENVIRONMENT

Our first environment is a Pac-Man-inspired pixel-navigation task defined on an $N \times N$ grid. Walls and pellets occupy individual cells, as do the controllable agent and each of the K ghosts. The agent aims to consume all pellets while avoiding collisions with ghosts. The primary comparison uses an 8×8 grid, three ghosts, and a fixed wall layout. Figure 2a shows an example. The agent has five actions: up, down, left, right, and stay. Movement is restricted to orthogonally adjacent non-wall cells, and actions directed into walls leave the agent stationary. Moving onto a pellet consumes it. After the agent acts, ghosts are updated sequentially, avoiding cells occupied by other ghosts. The number of ghosts and their behaviour are configurable. We consider two ghost policies: uniform *random* movement and *flee*, which maximizes the Manhattan distance to the agent, breaking ties randomly. The primary comparison uses random ghosts that move at every step. An episode terminates when the agent collides with a ghost or consumes the final pellet. Following termination, the wall layout is preserved, pellets are restored to all traversable cells except the agent's starting cell, and the agent and ghosts return to deterministic reset positions. Initial states used to generate training trajectories instead randomize the agent and ghost positions and sample different pellet densities. Training actions are sampled uniformly from the five-action space.

Observation encoding. Each observation is a four-channel binary image indicating walls, pellets, the agent, and ghosts. The underlying environment state admits seven valid per-pixel occupancy patterns: empty, wall, pellet, agent, ghost, ghost with pellet, and an agent-ghost collision. In the primary observation setting, pellets covered by ghosts remain in the underlying state but are hidden from the observed pellet channel. We divide each 8×8 observation into sixteen non-overlapping 2×2 patches, i.e., $P = 16$. Categorical models encode each patch by enumerating all combinations of the seven valid cell states across its four pixels, giving a vocabulary of $V = 7^4 = 2{,}401$ tokens. This encoding is lossless for every valid observation and therefore introduces no tokenizer reconstruction error. For continuous models, we render each occupancy pattern using a distinct fixed RGB colour. Each 2×2 RGB patch is flattened and rescaled to $[-1, 1]^{3\times2\times2}$, giving $d = 12$. Continuous predictions are decoded by mapping each generated pixel to its nearest palette entry. The projected frames are used for evaluation and fed back as context during rollout. All Pixel Pac-Man models condition on at most $L = 15$ preceding frames.

Evaluation metrics. The Pac-Man environment has stochastic dynamics, so disagreement with a sampled reference trajectory does not necessarily indicate a prediction error. A key advantage of our constructed environment is that its dynamics are known, allowing us to assess whether generated transitions are permissible rather than requiring agreement with a single reference trajectory. We evaluate this validity at two levels. **Transition validity** is the fraction of predicted transitions satisfying the relevant simulator constraints, irrespective of earlier failures. **Rollout validity** measures consistency over time: at each horizon, it is the fraction of trajectories that remain valid throughout their generated prefix. We evaluate 64-step autoregressive rollouts. Once a trajectory fails, subsequent valid transitions do not restore its rollout validity. Where appropriate, we further split

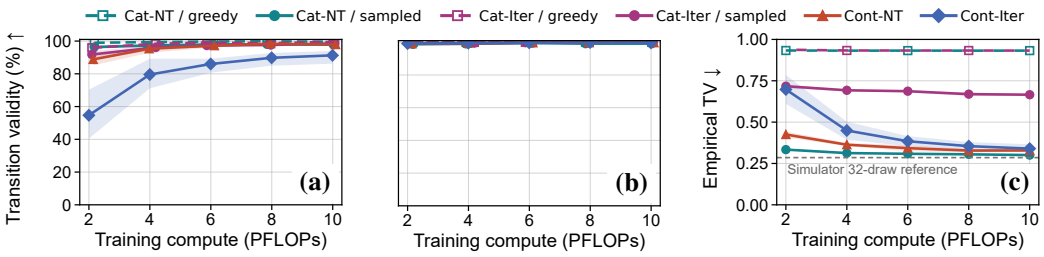


Figure 3: **Next-frame transition validity and calibration across training-compute budgets, conditioned on 15 observed context frames and the next action. (a)** Overall transition validity across 512 shared test contexts. **(b)** Deterministic transition validity excludes ghost-dynamics checks while retaining Pac-Man and board constraints, showing that all approaches learn the deterministic elements nearly perfectly. **(c)** Empirical total variation between world-model samples and the true simulator distribution. The dashed line shows the expected empirical TV from 32 simulator samples. We see that producing valid transitions does not guarantee distributional calibration.

transition and rollout validity by component (ghosts, Pac-Man, board), or into stochastic (ghosts) and deterministic (Pac-Man and board) elements. Further details are provided in Appendix A.2.

4.1.1 Findings in Pixel Pac-Man

Finding 1: All models achieve near-perfect next-frame validity on deterministic elements. We first compare next-frame transition validity across training-compute budgets (Figure 3(a)). Overall validity improves with training compute for all approaches. The categorical models perform well even at low compute, whereas the continuous models, especially Cont-Iter, show larger improvements with training but remain less reliable. To identify the source of these differences, Figure 3(b) excludes ghost dynamics. All approaches achieve near-perfect deterministic validity from 2 PFLOPs, indicating that the differences in overall validity may arise from modeling the stochastic transitions.

Finding 2: Continuous models struggle more to learn stochastic elements at long horizon. Next, we evaluate models' behaviors at long horizon, through rollouts of 64 steps. For each rollout step, we verify the validity of generated transitions against the expected environment behavior. We report rollout validity (Figure 4), i.e. how many trajectories have all valid transitions at each rollout step. We report validity on deterministic elements only, as well as all elements (including the stochastic ghosts). Detailed results are available in Appendix C. Our results show that models achieve similar performance for deterministic transitions, with categorical models (greedy inference) outperforming models with stochastic inference. In this setting, this shows that all models can effectively learn to model deterministic environments. We observe substantially more variability for the stochastic elements, with categorical models consistently achieving better performance at all compute budgets. Greedy inference again achieves the best overall results. Cont-NT consistently improves throughout training, slowly closing the gap with categorical models, although not matching their performance. The two categorical models approach similar rollout validity at the largest budget, with Cat-NT achieving stronger results at lower compute. Lastly Cont-Iter appears to struggle the most to learn the ghost mechanics, with very poor initial performance, slow improvements over compute budgets and a substantial gap remaining at our maximum compute budgets considered. While we expect all models to eventually be capable of modeling this environment given sufficient resources, this experiment suggests that continuous models struggle to model stochastic dynamics. This aligns with qualitative observations that generative world models can struggle to preserve visually and dynamically uncertain entities over time, including blurry or temporally inconsistent opponent predictions (Alonso et al., 2024; Parker-Holder et al., 2024).

Finding 3: Categorical models demonstrate better compute efficiency than continuous models. As shown in Figure 4, categorical models consistently achieve better overall performance with smaller compute budget. With greedy decoding, Cat-NT retains approximately 86% rollout validity at step 64 after only 2 PFLOPs of training. Comparing models on their capability to model determin-

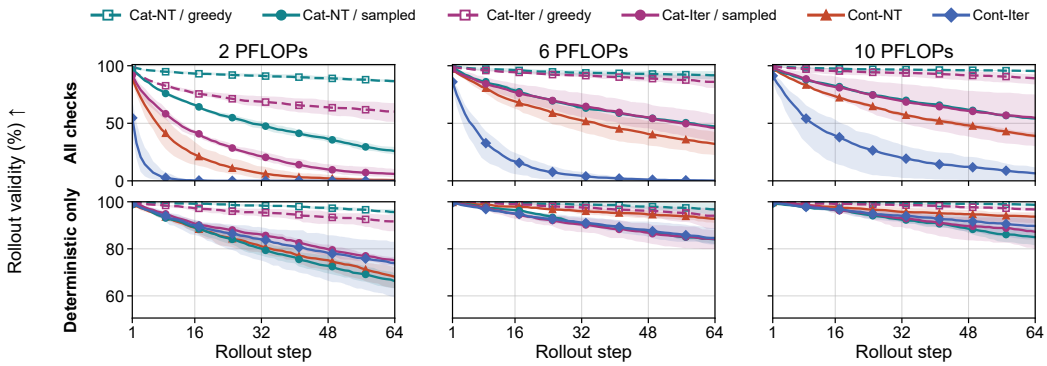


Figure 4: **Rollout validity over 64 autoregressive steps across training compute budgets**. Curves show the percentage of rollouts remaining valid throughout the generated prefix, under full evaluation (top) and ghost-excluded evaluation (bottom).

istic elements only shows much closer capability, highlighting that increased compute requirements are tied to models' ability to model stochastic elements accurately.

Finding 4: Transition validity does not imply calibrated stochastic predictions. The validity metrics reported above measure whether generated transitions obey the environment's rules, but not whether different valid outcomes occur with the correct probabilities. To understand how these relate, we assess calibration using total variation (TV) between the empirical distribution of predicted next-step ghost configurations and the simulator's data distribution. For each model checkpoint, we generate 32 next-frame predictions conditioned on each real 15-frame history and its next action. We evaluate a fixed, model-independent subset of 232 context histories for which the ghost-configuration distribution is exactly identifiable. Figure 3c shows that sampled CAT-NT achieves the lowest empirical total variation at every evaluated compute budget. At 10 PFLOPs, mean TV is 0.30 for sampled CAT-NT, 0.33 for CONT-NT, 0.34 for CONT-ITER, and 0.67 for sampled CAT-ITER. Both continuous approaches substantially improve between 2 and 10 PFLOPs and outperform sampled CAT-ITER at the largest budget, despite their weaker validity results. This reversal in model ranking highlights the need to evaluate calibration alongside validity. The results also suggest an important direction for future work on CAT-ITER-type models specifically to further understand and improve the potential trade-offs between validity, calibration, and inference speed.

Finding 5: Increasing dynamics complexity disproportionately degrades continuous models. We examine how the models' rollout validity changes as the ghost dynamics become more demanding. We vary the number of ghosts (one, three, or five) and their behavior (Random or Flee), while keeping the remaining environment settings and training horizon fixed. Figure 5 shows that categorical models achieve higher rollout validity across most conditions. Greedy CAT-NT is the most robust approach overall, while greedy CAT-ITER also performs strongly, although its validity decreases as the number of Flee ghosts increases. Greedy decoding improves rollout validity for both models. With five Random ghosts, sampled CAT-ITER retains approximately 50% rollout validity at step 64. Increasing the number of ghosts generally reduces rollout validity, with a substantially steeper reduction for the continuous models. With five Random ghosts, greedily decoded CAT-NT and CAT-ITER retain approximately 95% and 86% validity at step 64, respectively, whereas CONT-NT and CONT-ITER approach 0%. The effect of ghost behavior also differs between model classes: the continuous models achieve higher validity under Flee than Random dynamics, whereas categorical models generally show the opposite pattern. Appendix C.2 further examines structural failures and adherence to the ghost policy in Flee rollouts.

4.2 "3D MAZE" CONTROLLED ENVIRONMENT

In addition to the 2D pixel-navigation environment, we created a partially observable 3D environment, which renders the same grid-based map in 3D from a first-person perspective, with continuous, rather than discrete navigation. Non static elements are included in the form of three bouncing

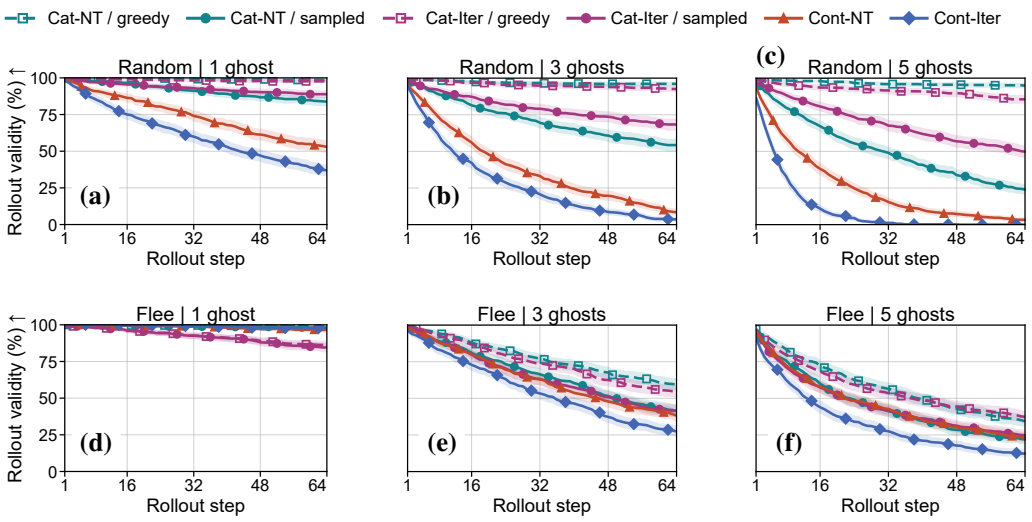
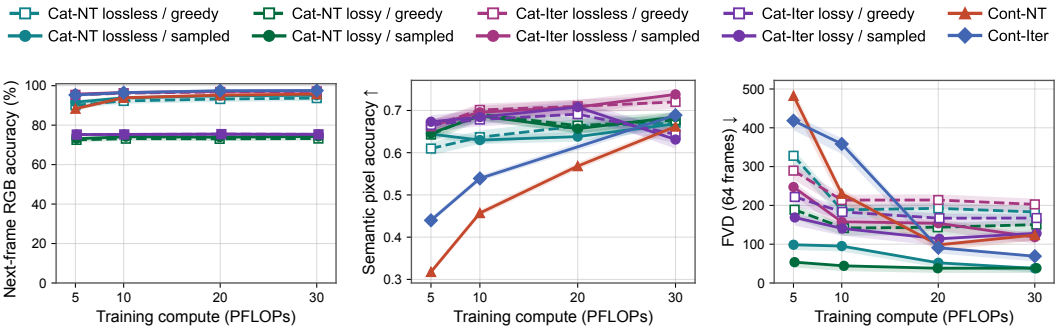


Figure 5: **Rollout validity as ghost-dynamics complexity increases.** Panels vary the number of ghosts under Random dynamics **(a–c)** and Flee dynamics **(d–f)**. Each curve is the percentage of 512 rollouts, conditioned on 15 context frames. Training compute is approximately 12 PFLOPs (one training seed only): Random/3 uses replacement runs at 11.42–12.20 PFLOPs; other panels retain the original 11.42-PFLOP ceiling-selected checkpoints. Categorical models use greedy ($T = 0$) or sampled ($T = 1$) decoding. Shading shows pointwise 95% intervals from 2,000 bootstrap resamples.



(a) Next-frame RGB accuracy. (b) Semantic pixel accuracy. (c) 64-frame FVD.

Figure 6: **Results for lossless and lossy models on 3D Maze.** (a) Exact next-frame RGB accuracy, (b) mean semantic pixel accuracy over 64-step rollouts, and (c) 64-frame FVD (lower is better). Shading gives pointwise 95% bootstrap intervals (2, 000 samples) over 512 shared evaluation cases.

spheres, randomly positioned at the start of the trajectory, with random initial velocities. These bounce off walls but do not interact with the agent. Sphere dynamics are deterministic given their positions and velocities, but these may be uncertain from the observed history. To prevent visual complexity from obscuring other signals, the rendering uses only solid colors, with no textures or shading. Spheres, walls, floor, and ceiling are distinguished by color. Training trajectories are generated by traversing random routes through the maze, resetting the environment when a randomly chosen goal is reached. See a visual example in Figure 2b and additional details in Appendix B.

Image encoding. We consider three representations of the environment: discrete lossless, discrete lossy, mirroring common categorical modelling approaches, and continuous. We begin by rendering each frame as a $64 \times 64 \times 3$ RGB image and divide this frame into $64 \times 1 \times 3$ vertical strips. Continuous models directly operate with these strips as inputs. For the lossless discrete representation, we tokenize each strip using a lookup table. The simple rendering allows a vocabulary of 10,000 entries to cover all column patterns encountered in the training and evaluation data. For the lossy

representation, we use a small VQ-VAE (Van Den Oord et al., 2017) whose encoder comprises two convolutional layers with 8×3 kernels and stride $(8, 1)$. This produces 64 discrete tokens per frame, one per column, while incorporating neighboring-column context.

Evaluation metrics. Due to its partial observability and continuous actions, we cannot directly measure component validity as in the Pac-Man environment. We rely on the following three metrics: **Next-frame pixel accuracy** measures exact RGB agreement with the reference next frame. **Fréchet Video Distance (FVD)** (Unterthiner et al., 2019) is measured over long horizon rollouts to evaluate distributional alignment with ground truth rollouts. **Semantic accuracy** measures pixel-wise agreement with the reference on three classes: wall, spheres, and background. Pixels are classified using the environment's color palette.

4.2.1 Findings in 3D Maze

Finding 1: Lossy tokenization limits predicted visual quality, while lossless and continuous models reach similarly high accuracy. As in our first Pixel Pac-Man experiment, we measure our models' ability to generate the next frame in a sequence correctly. Figure 6a compares next frame RGB accuracy over training budgets. It shows that lossy models quickly plateau, while all other methods steadily improve to similarly near perfect accuracy.

Finding 2: The visual accuracy gap between lossy and lossless categorical models does not translate into a consistent disadvantage in long-horizon metrics. Next, we evaluate how models behave over long horizons, computing both FVD (Figure 6c) and semantic accuracy (Figure 6b) over 64-step rollouts. Lossy models remain competitive with their lossless counterparts, indicating that lower next-frame accuracy does not consistently translate into worse semantic accuracy or FVD. We note that compression may have a greater impact on dynamics that depend on fine visual details. Nevertheless, this result highlights the importance of disentangling visual quality from temporal modeling performance.

Finding 3: Categorical models achieve stronger long-horizon metrics at low training budgets, while continuous models narrow the gap with additional compute. Similarly to our Pixel Pac-Man experiments, we see that categorical models show much better training efficiency than continuous models. Continuous models substantially narrow the gap by 30 PFLOPs, although differences between models and decoding procedures remain. Unlike the stochastic ghost dynamics in Pixel Pac-Man, uncertainty in the maze arises from partial observability, including spheres entering view with previously unobserved positions or velocities. In addition, while Pixel Pac-Man validity results heavily favored greedy inference, sampling substantially improves categorical FVD with respect to greedy decoding in this maze setting, without a correspondingly large change in other metrics.

5 Conclusion

We presented a controlled comparison of categorical and continuous visual world models across 2D and 3D environments that were specifically constructed for systematic analysis. In our lossless 2D environment, categorical models achieved higher rollout validity at lower training compute, particularly for stochastic dynamics and increasingly complex environments. We also pinpointed how validity and calibration measure distinct properties: generating permissible trajectories does not guarantee reproducing their correct distribution. In the 3D maze, the visual accuracy penalty from lossy tokenization did not systematically translate into worse long-horizon metrics. These findings demonstrate that controlled environments with known dynamics provide a valuable complement to large-scale world-model evaluation, enabling controlled comparisons of representation, tokenization, and generation procedures beyond visual-fidelity metrics. Limitations include that our experiments are carried out in controlled environments with relatively small model sizes and training budgets. While there is uncertainty on how the observed trade-offs will evolve at scale and with rollout-aware training methods such as self-forcing (Huang et al., 2025), our experiments reproduce qualitative failures observed in larger world models. Our findings support targeted model design choices based on application requirements and resource constraints.

5.1 AI USE STATEMENT

The authors used generative AI to support literature searches and summaries, manuscript drafting and revision, preparation of plotting and data-processing code, reference and table formatting, and the presentation and interpretation of experimental results. It was also used to inspect saved experiment artifacts and suggest revisions to figures and captions. The authors manually reviewed the resulting text, code, figures, citations, and analyses, checking factual claims against primary sources and numerical claims against the underlying experimental artifacts. Responsibility for the accuracy, originality, and integrity of the submitted work rests entirely with the authors.

5.2 REPRODUCIBILITY STATEMENT

We have made efforts to make our work as reproducible as possible. All architecture details, training and inference parameters are detailed in the paper. The design of each synthetic environment is described in as much detail as possible, with no proprietary data or model weights. All our models are trained from scratch on procedurally generated data. Section 3 contains details of the methods used; training and inference settings are listed in Appendix A, with evaluation details in Appendices A.2 and A.3 and parameter ablations in Appendix C.1.

REFERENCES

Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari. *Advances in Neural Information Processing Systems*, 37:58757–58791, 2024.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first international conference on machine learning*, 2024.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11305–11315. IEEE, 2022.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2025.

Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in neural information processing systems*, 36:9156–9172, 2023.

Junliang Guo, Yang Ye, Tianyu He, Haoyu Wu, Yushu Jiang, Tim Pearce, and Jiang Bian. Mineworld: a real-time and open-source interactive world model on minecraft. *arXiv preprint arXiv:2504.08388*, 2025.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *arXiv preprint arXiv:2506.08009*, 2025. doi: 10.48550/arXiv.2506.08009.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Anssi Kanervisto, Dave Bignell, Linda Yilin Wen, Martin Grayson, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Tabish Rashid, Tim Pearce, Yuhan Cao, et al. World and human action models towards gameplay ideation. *Nature*, 638(8051):656–663, 2025.

Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *Advances in Neural Information Processing Systems*, 37: 56424–56445, 2024.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The 11th International Conference on Learning Representations*, 2023.

Yixin Liu, Kai Zhang, Yuan Li, Zhiling Yan, Chujie Gao, Ruoxi Chen, Zhengqing Yuan, Yue Huang, Hanchi Sun, Jianfeng Gao, Lifang He, and Lichao Sun. Sora: A review on background, technology, limitations, and opportunities of large vision models, 2024. URL https://arxiv.org/abs/2402.17177.

Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models. In *The 11th International Conference on Learning Representations*, 2023.

Microsoft Research. WHAMM! real-time world modelling of interactive environments. https://www.microsoft.com/en-us/research/articles/whamm-real-time-world-modelling-of-interactive-environments/, April 2025.

Arian Mousakhan, Sudhanshu Mittal, Silvio Galesso, Karim Farid, and Thomas Brox. Orbis: Overcoming challenges of long-horizon prediction in driving world models. *Advances in Neural Information Processing Systems*, 38:4741–4770, 2026.

Jack Parker-Holder, Philip Ball, Jake Bruce, Vibhavari Dasagi, Kristian Holsheimer, Christos Kaplanis, Alexandre Moufarek, Guy Scully, Jeremy Shar, Jimmy Shi, Stephen Spencer, Jessica Yung, Michael Dennis, Sultan Kenjeyev, Shangbang Long, Vlad Mnih, Harris Chan, Maxime Gazeau, Bonnie Li, Fabio Pardo, Luyu Wang, Lei Zhang, Frederic Besse, Tim Harley, Anna Mitenkova, Jane Wang, Jeff Clune, Demis Hassabis, Raia Hadsell, Adrian Bolton, Satinder Singh, and Tim Rocktäschel. Genie 2: A large-scale foundation world model. https://deepmind.google/discover/blog/genie-2-a-large-scale-foundation-world-model/, December 2024.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024.

Yutao Sun, Hangbo Bao, Wenhui Wang, Zhiliang Peng, Li Dong, Shaohan Huang, Yaoyao Chang, Jianyong Wang, and Furu Wei. Multimodal latent language modeling with next-token diffusion. In *Forty-third International Conference on Machine Learning*, 2026.

Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in Neural Information Processing Systems*, 37:84839–84865, 2024.

Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges, 2019. URL https://arxiv.org/abs/1812.01717.

Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter. Diffusion models are real-time game engines. In *International Conference on Learning Representations*, volume 2025, pp. 73754–73776, 2025.

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International conference on machine learning*, pp. 10524–10533. PMLR, 2020.

Lijun Yu, José Lezama, Nitesh Bharadwaj Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Agrim Gupta, Xiuye Gu, Alexander G Hauptmann, et al. Language model beats diffusion-tokenizer is key to visual generation. In *International Conference on Learning Representations*, volume 2024, pp. 765–783, 2024.

Yifan Zhang, Chunli Peng, Boyang Wang, Puyi Wang, Qingcheng Zhu, Fei Kang, Biao Jiang, Zedong Gao, Eric Li, Yang Liu, et al. Matrix-game: Interactive world foundation model. *arXiv preprint arXiv:2506.18701*, 2025.

APPENDIX

A IMPLEMENTATION AND TRAINING DETAILS

Table 1 reports the shared optimization and capacity settings.

Table 1: Training and capacity settings. Compute budgets refer to the maximum reported comparison budget.

Setting	**Pixel Pacman**	**3D Maze**
Trajectories per epoch	2,048	2,048
Frames per trajectory	16	16
Batch size	128	128
Training budget	10 PFLOPs (12 PFLOPs for the ghost-dynamics experiment)	30 PFLOPs
Training regime	Teacher forcing	Teacher forcing
Optimizer	AdamW	AdamW
Learning rate	Constant 4×10^{-4}	Constant 4×10^{-4}
Weight decay	0	0
Gradient clipping	Global norm 1.0	Global norm 1.0
Transformer depth	6 blocks	6 blocks
Attention heads	4	4
MLP ratio	4	4
nanoGPT width	44	44
DiT width	40	36
Continuous encoder	Linear projection	2-layer 1D CNN, 48 channels, then linear projection
Validation trajectories	128	512
Training seeds	3	1

Table 2 details architecture specific design choices.

Table 3 provides implementation details for the VQ-VAE trained on the 3D maze dataset.

A.1 PAC-MAN IMPLEMENTATION DETAILS

We train all four models on the Pixel Pac-Man environment for a compute budget of up to 10 PFLOPs, with matching training parameters. Each epoch comprises 2048 randomly sampled trajectories of 16 frames. Models are evaluated at multiple matching compute budgets across 64 rollout steps. Categorical models are evaluated with greedy sampling (temperature T=0) and stochastic (T=1). DiT and NanoGPT backbones are aligned in terms of parameter count as much as possible, with each model type having additional capacity dedicated to their modeling type (e.g. patch embeddings, flow matching head). Continuous models use 50 inference steps, and CAT-ITER uses 16. All models are trained across 3 random seed initializations. Unless noted otherwise, result plots show lines to denote means over training seeds and shading shows seed minimum-maximum ranges. See Appendix C.1 for hyperparameter ablations.

Table 2: Model-specific training and inference settings for the main comparisons.

Model	Training details	Inference details
CAT-NT	Cross-entropy on the final target frame; visual input embeddings are tied to the output classifier.	Sequential patch generation with greedy decoding ($T = 0$) or categorical sampling ($T = 1$), without top-k truncation.
CONT-NT	Conditional flow matching with three residual MLP blocks of hidden width 23; logit-normal time sampling with parameters $\mu = 0$, $\sigma = 1$, over 1,000 discrete timesteps; training flow shift 3.	Each patch is initialized from Gaussian noise and generated separately before being fed back into the causal sequence. Pac-Man: 50 Euler evaluations per patch, inference shift 1. 3D maze: 20 Euler evaluations per patch, inference shift 3.
CAT-ITER	The number of masked target patches is sampled uniformly from $\{0, \ldots, P\}$, where P is the number of patches per image; cross-entropy is evaluated only at masked positions. At least one masked position is enforced per training batch.	16 reveal iterations for Pac-Man and 20 for the 3D maze, using a linear schedule. Remaining tokens are predicted using greedy decoding ($T = 0$) or categorical sampling ($T = 1$); the highest-confidence predictions are permanently committed at each iteration.
CONT-ITER	Whole-frame conditional flow matching with logit-normal time sampling with parameters $\mu = 0$, $\sigma = 1$, over 1,000 discrete timesteps; training flow shift 3.	The complete frame is initialized from Gaussian noise and updated jointly. Pac-Man: 50 Euler evaluations per frame. 3D maze: 20 Euler evaluations per frame. Inference shift 3 in both environments.

Table 3: Column VQ-VAE architecture and training configuration.

Setting	Value
Input	$64 \times 64 \times 3$ RGB, scaled to $[0, 1]$
Tokens per frame	64 (1×64 grid)
Codebook size	256
Embedding dimension	32
Hidden channels	64
Encoder	Two convolutions, ReLU between layers
Encoder kernels / strides	8×3 / $(8, 1)$ in both layers
Decoder	Two transposed convolutions
Decoder residual blocks	2, before upsampling
Reconstruction loss	Mean absolute error (L_1)
Codebook loss	Mean squared error
Commitment coefficient	0.25
Optimiser	Adam
Learning rate	10^{-3}, constant
Gradient clipping	Global norm of 1.0
Training steps	300,000
Batch size	64
Data sampling	Shuffled frames from 8 fresh trajectories per buffer

A.2 PAC-MAN EVALUATION DETAILS

Evaluation contexts and action sequences are shared across models. For each checkpoint and decoding condition, we evaluate *512* contexts. Results are averaged across contexts, and we report the min, max (shaded region) and mean (line and markers) over three independent training runs, where applicable.

We evaluate one-step predictions conditioned on 15 observed frames and the next action. **Transition validity** is the fraction of predicted transitions satisfying the selected simulator constraints, irrespective of earlier failures. For long-horizon evaluation, we generate $H = 64$ frames autoregressively under a prescribed action sequence, feeding generated observations back into the model. **Rollout validity** at horizon h is the fraction of trajectories that remain valid throughout their first h generated transitions. Once a trajectory fails, subsequent valid transitions do not restore its rollout

validity. The first generated transition is indexed by $t = 1$, as conditioning frames are excluded from all reported metrics.

State representation and validity criteria. All predictions are evaluated in a common representation with wall, pellet, Pac-Man, and ghost occupancy channels. Categorical tokens decode into patches of allowed cell states. Continuous RGB predictions are mapped to the nearest allowed palette colour at each cell; RGB-native rollouts also apply this projection before feeding predictions back into the model. Consequently, cell-state validity is enforced by construction. This does not enforce board-level or temporal consistency, which require the checks below.

Component checks. Pac-Man validity requires exactly one Pac-Man and movement consistent with the supplied action and wall layout. Ghost validity checks occupancy and movement permitted by the configured ghost policy. For stochastic ghosts, any permitted movement is accepted, regardless of whether it matches the movement in a reference trajectory. Board validity requires wall preservation and pellet consistency jointly. Pellet checks distinguish changes in visibility from changes in underlying presence: ghosts may hide pellets without consuming them, and revealing a previously hidden pellet is not counted as creation.

History consistency. A sequence of locally valid transitions need not admit a consistent hidden-state history. The full rollout validator therefore propagates compatible hidden-state hypotheses from the observed context through the generated sequence, including hypotheses about occluded pellets. Local transition checks do not require the earlier generated prefix to have remained valid.

Episode boundaries. Evaluation continues across episodes. An observed reset crossing is checked against the simulator's clean-reset rule, including the prescribed agent positions and restored pellet layout, accounting for occlusion. The action supplied at the reset boundary is ignored, as in the simulator. Capture and episode termination are not themselves validity failures. A terminal observation at the final forecast step provides no subsequent reset transition to evaluate.

Ghost-excluded evaluation. We score the same generated trajectories with and without ghost-specific constraints. Ghost-excluded evaluation omits ghost checks and joint reset reachability, while retaining local Pac-Man and board checks, fresh-pellet requirements at resets, and collision rejection at definite reset crossings.

A.3 Maze environment evaluation details

We follow the same training and evaluation protocol as for Pixel Pac-Man, with a compute budget of up to 30 PFLOPs. Continuous models and CAT-ITER use 20 steps. Each model configuration is trained with a single random seed.

B Additional maze environment details

We provide additional detail on the maze environment design and data construction.

The agent navigates via two continuous joystick values – a steering value (where -1 means turn full-speed left, 1 means turn full-speed right, and 0 is a turn no-op), and a forwards motion value (where 1 means full-speed forwards, and 0 means stand still). Agents cannot walk through walls, but can slide along them, so a wall collision does not necessarily leave the agent stationary.

Camera pitch is fixed so that the agent is always looking straight ahead in the direction of movement, and wall height and ball elevation are uniform, such that the geometry of the scene is always symmetrical about the horizontal axis. Internal walls are all one color, while the North, South, East and West external walls each have their own color. Spheres are drawn in solid yellow, the floor in solid green, and the ceiling in solid red.

C DETAILED EXPERIMENT RESULTS

In this section, we provide more additional and more detailed results to the experiments provided in the main manuscript.

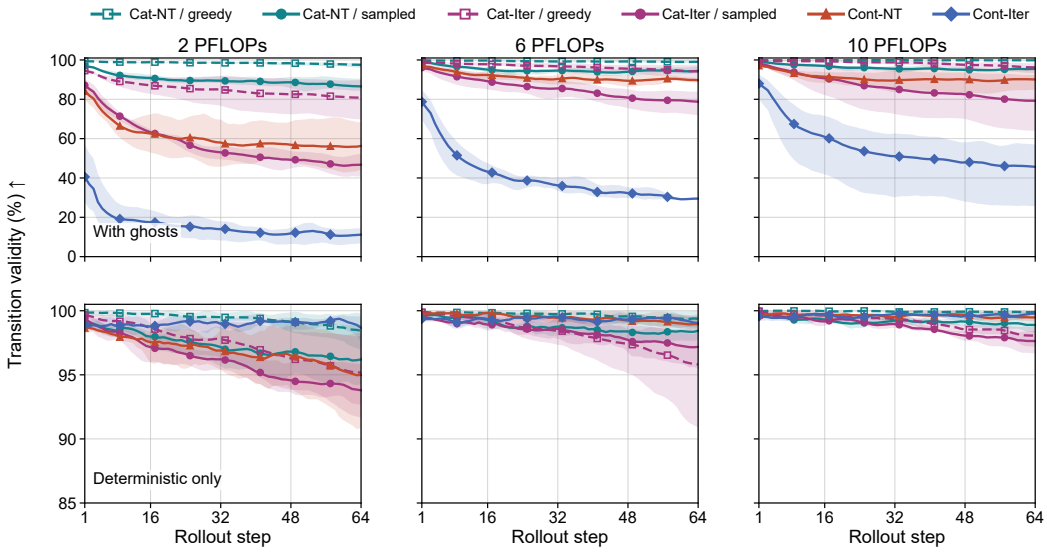


Figure 7: Local transition validity over 64 autoregressive rollout steps. Complementing the strict rollout-validity results in Figure 4, local transition validity shows that the observed long-horizon failures are not solely caused by permanently recording isolated early errors: when ghost dynamics are included, the continuous models, particularly CONT-ITER, continue to produce locally invalid transitions, while deterministic-only validity remains high.

First, we complement the strict rollout-validity results in Figure 4 with local transition validity in Figure 7. This metric evaluates each predicted transition independently of earlier errors, showing whether a model returns to producing locally valid transitions after a failure; such local recovery does not restore the validity of the preceding rollout history. As before, we compare evaluation with all checks against deterministic-only evaluation, which excludes ghost-dynamics checks while retaining Pac-Man, board, collision, and reset constraints. Deterministic-only transition validity remains high for all approaches, whereas including ghost dynamics reveals persistent local errors, particularly for the continuous models and most strongly for CONT-ITER. Thus, the decay in strict rollout validity observed in Figure 4 is not solely an artefact of permanently recording an isolated early failure: the continuous models continue to produce locally invalid transitions throughout the rollout.

In addition, Figure 8 shows cumulative per component failure at the maximum compute budget (approx. 10 PFLOPs). These results support findings discussed in the main paper, with all models achieving very similar performance in deterministic settings, with stochastic ghosts being by far the biggest source of failure. Wall failures are concentrated in sampled categorical models, with only negligible failure rates under greedy decoding.

C.1 PARAMETER ANALYSIS EXPERIMENTS

Continuous model inference parameter study. We evaluate the impact of increasing the number of flow matching inference steps for continuous models, on rollout quality. We compute long horizon rollouts for all models seeds across 20, 50 and 100 inference steps. Figure 9 reports transition validity, strict rollout validity, and mean valid-prefix length. These measure, respectively, the percentage of valid transitions, the percentage of rollouts remaining valid through all 64 generated frames, and the average number of consecutive valid frames before the first failure. Our results suggest that 50

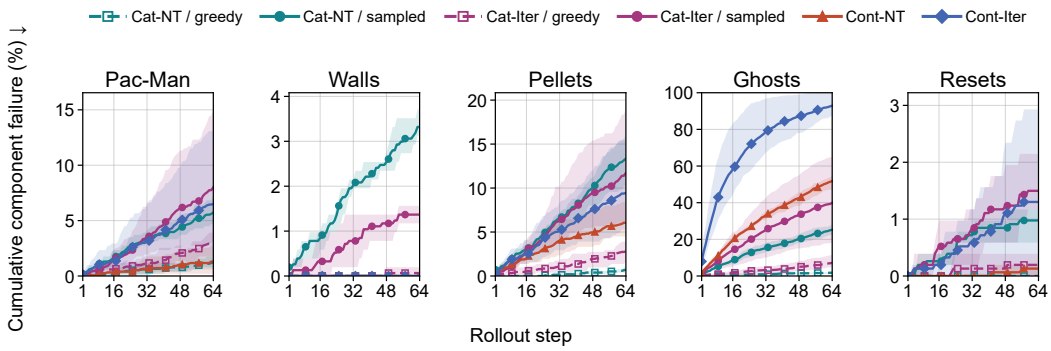


Figure 8: Cumulative component failures over autoregressive rollouts at approximately 10 PFLOPs of training compute. Each panel shows the percentage of rollouts with at least one violation of the indicated component by each horizon. A rollout may contribute to multiple panels.

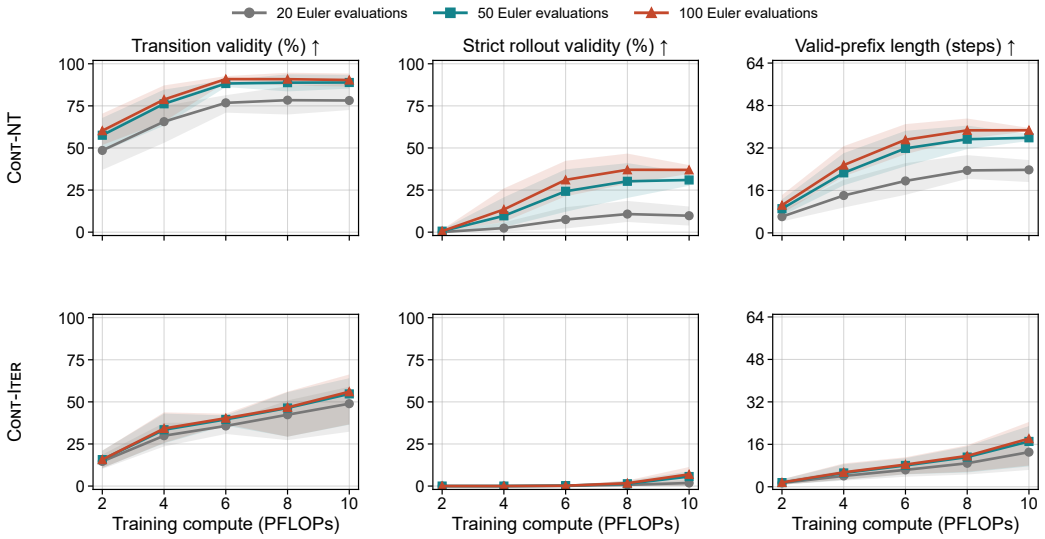


Figure 9: Impact of number of inference steps for continuous models on overall performance.

inference steps achieves a balance between maximising model performance while maintaining reasonable inference costs. We can see that Cont-Iter appears to plateau after 50 steps, while Cont-NT keeps improving, albeit much more modestly than between 20 and 50 steps. We therefore use 50 inference steps for continuous models in the main Pac-Man comparison; the 3D maze evaluations use 20 steps. Additionally, we investigate the impact of the flow shift parameter at inference time, which influences where inference steps are sampled on the noise-to-target trajectory. The higher the flow shift, the more steps are sampled during early noisy steps. Our experiments show that while the parameter has very little influence on Cont-Iter models, Cont-NT models tend to favour smaller flow shift values. The parameter has however a much smaller impact than the number of inference steps.

Cat-Iter model inference parameter study. In addition to continuous models, we run a parameter analysis on Cat-Iter models, both in terms of number of iterations and sampling temperature. We run tests with $4, 8, 10, 12$ and 16 inference steps, and temperature $T = \{0, 0.5, 1, 1.5\}$. Our results show that 16 steps substantially outperform other modes, with greedy inference achieving the overall best performance. We further note that the gap between 16 steps inference and other parameter values narrows in greedy mode.

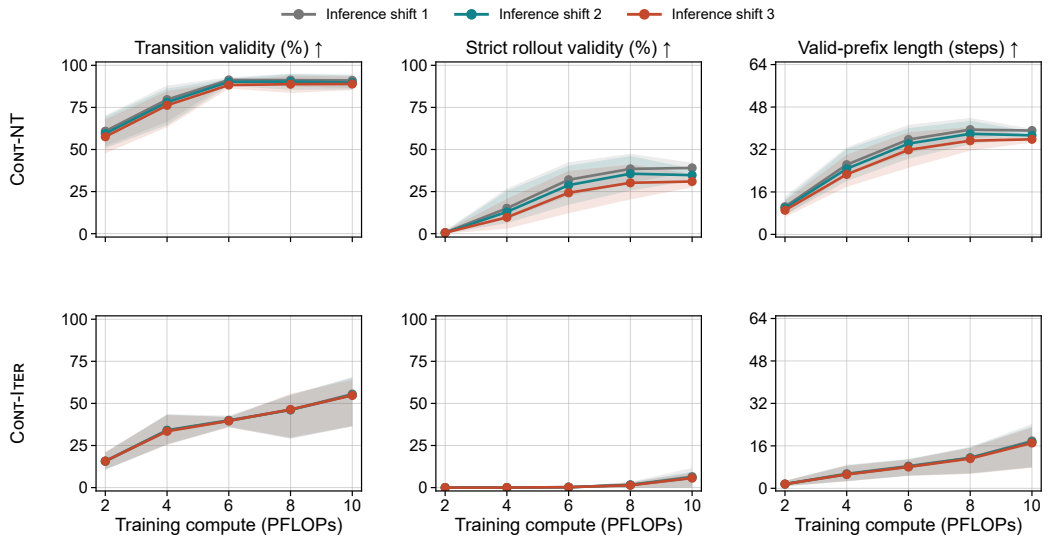


Figure 10: Impact of the flow shift parameter for continuous models on overall performance.

C.2 Structural and Semantic Validity Under Different Ghost Dynamics

Figure 5 shows that continuous models achieve higher rollout validity under Flee than Random dynamics, while categorical models generally exhibit the opposite pattern. To investigate the sources of failure, Figure 12 separates structural consistency from adherence to the fleeing policy at the approximately 12-PFLOP checkpoints.

We measure *structural validity* by relaxing the ghost-policy constraint: ghosts may take any legal direction rather than following the fleeing preference and tie-break. All other checks remain active, including walls, ghost occupancy and count, legal movement, Pacman dynamics, pellet history, collisions, and resets. A sequence is structurally valid through step h only if its entire prefix satisfies this relaxed evaluator.

Let S_h denote structural validity through step h and V_h denote validity under the original fleeing rules. Since $V_h \subseteq S_h$, we report conditional semantic validity as

$$100 \Pr(V_h \mid S_h) = 100 \frac{N(V_h)}{N(S_h)},$$

where N counts sequence prefixes satisfying each event. This is the percentage of structurally valid prefixes that also satisfy the fleeing policy; it is undefined when no structural survivors remain.

Structural failures account for much of the continuous-model deficit. The first row shows substantial losses of structural validity for continuous models as the number of ghosts and rollout length increase. Among the prefixes that remain structurally valid, their conditional semantic validity is considerably stronger; CONT-ITER, in particular, retains high conditional validity despite pronounced structural degradation. Thus, structural failures dominate the continuous models' long-horizon failures under fleeing conditions, rather than an inability to follow the fleeing policy alone.

Possible explanation for the behavior asymmetry. One hypothesis is that continuous models predict sharper, more coherent whole frames when the future is closer to unimodal. Conditional on a fully specified state and action, fleeing reduces transition ambiguity relative to random ghost motion. With multiple plausible futures, continuous generation may instead produce less coherent combinations of object positions or appearances, increasing structural failures. A complementary hypothesis is that continuous models can exploit a spatial shortcut, such as learning a smooth "move away" vector field, that approximates fleeing dynamics. Random motion offers no equivalent directional regularity. Discrete models may preserve valid frame structure more reliably under stochastic

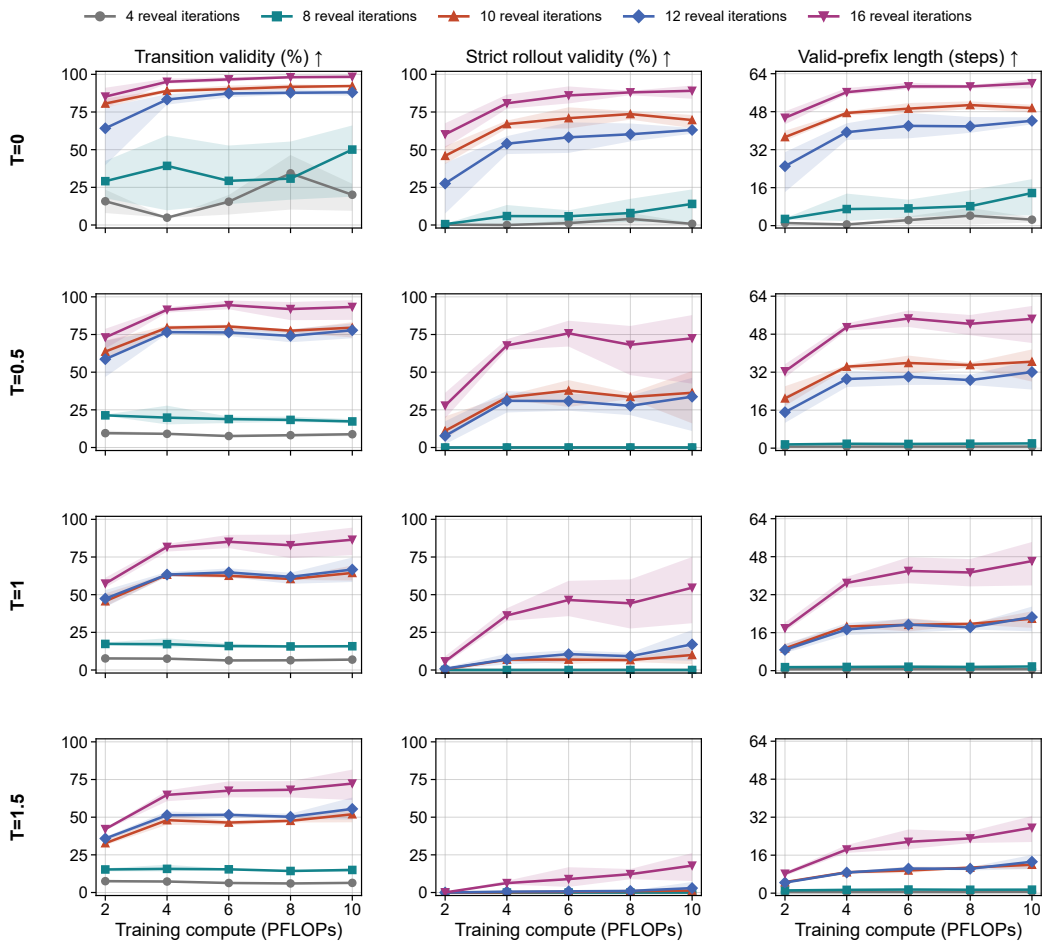


Figure 11: Impact of number of inference steps and sampling temperature for the Cat-Iter model on overall performance.

Table 4: Normalized discrete AUC (%) for the curves in Figure 5, averaged over steps 1–64 for 512 sequences at 12 PFLOPS compute budget. Higher is better. Random and Flee denote random and flee ghost behavior; parentheses give the ghost count.

Method	Random (1)	Random (3)	Random (5)	Flee (1)	Flee (3)	Flee (5)
Cat-NT T=0	99.33	96.95	96.53	99.49	77.86	58.42
Cat-NT T=1	91.39	71.62	51.28	98.92	66.85	45.80
Cat-Iter T=0	98.39	95.44	91.57	92.69	74.66	57.19
Cat-Iter T=1	93.06	79.91	69.36	92.25	65.11	45.96
Cont-NT	73.92	38.60	25.09	98.51	63.85	46.04
Cont-Iter	61.66	27.67	10.44	98.99	55.75	33.24

dynamics, while still finding the coordinated fleeing rule difficult. These explanations remain speculative: Figure 12 evaluates only Flee rollouts and does not directly measure sharpness or learned vector fields.

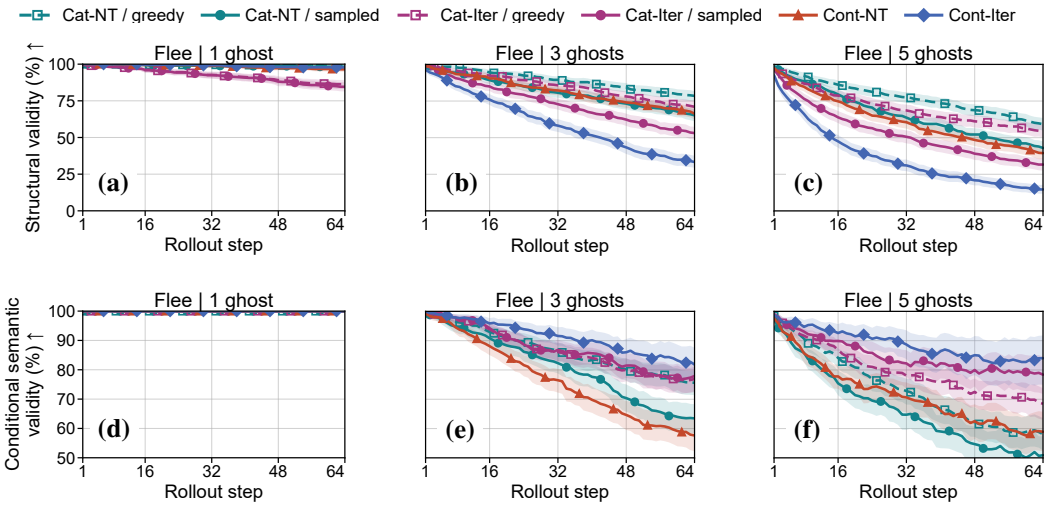


Figure 12: Structural and conditional semantic validity of rollouts from flee models under a common training-compute ceiling of approximately 12 PFLOPs (512 sequences). Top: percentage of sequence prefixes valid when the fleeing-policy requirement is relaxed to allow any legal ghost direction, retaining all other constraints. Bottom: percentage satisfying the original fleeing rules conditioned on structural validity. Shaded bands show pointwise 95% intervals from 2,000 trajectory-bootstrap resamples. Structural failures account for much of the continuous-model survival loss, while surviving structurally valid prefixes show stronger policy consistency.