WHY DO CONVENTIONAL WORLD MODELS FAIL TO LEARN CELLULAR AUTOMATA?

Anonymous authors
Paper under double-blind review

ABSTRACT

Although conventional world models — auto-regressive or diffusion models based on transformers or convolutional networks — may learn surface statistics of world dynamics, can they learn the exact world dynamics from its observed history? Leveraging cellular automata as a simple testbed, we find the answer to be **no** in many cases. Conventional architectures predict most pixels correctly yet rarely complete a rollout: a CNN predicts 96.3% of cells but completes 18.9% of rollouts; a joint diffusion model completes none. We trace the gap to three failure modes of these world models — namely, they fail to exactly capture spatial locality, temporal locality or temporal stability. Simple changes repair each: (1) for **spatial locality**, two-dimensional rotary positions lift a transformer from 39.1% to 100% on the Game of Life; (2) for **temporal locality**, handing each token its cell's previous-frame neighbourhood lifts the same transformer from 25.8% to 99.9% on unseen rules; (3) for **temporal stability**, causal freezing lifts the same diffusion weights from 42.2% to 99.9%. None of the three changes touches the architectural backbone; each only modifies the information flow within it. The same diagnosis applies to billiards and to a simulated Burgers equation.

1 INTRODUCTION

World models aim to learn the rules governing a world from observations and use them to predict how the world evolves (Ha and Schmidhuber, 2018; LeCun, 2022). Existing work has explored a range of objectives, including learning physical dynamics (Kang et al., 2025), predicting future states (Hafner et al., 2023), maintaining scene consistency across time and viewpoints (OpenAI, 2024), and capturing the interactions among objects and agents (Battaglia et al., 2018; Micheli et al., 2023). However, generative models often predict well without recovering the world model behind the data (Vafa et al., 2024; 2025; Kang et al., 2025), and the failure is hard to analyse because the rule itself is unknown. Cellular automata (CA, Figure 2) provide a simple yet expressive setting for addressing this question. Wolfram's studies of cellular automata showed how simple local rules can generate rich and complex behaviour (Wolfram, 1983). In a CA, each cell updates according to its own state and those of its neighbours; in Conway's Game of Life, such local interactions produce patterns that move, interact, and collide (Gardner, 1970). This combination of simple, well-defined rules and rich behaviour makes cellular automata a clean testbed for investigating why standard world models fail to infer underlying world rules.

Research question

Why can't standard world models — autoregressive or diffusion models based on transformers or convolutional networks — infer an unseen cellular automaton's rule from observed history?

Several studies have investigated predicting future states of cellular automata from observations (Gilpin, 2019; Springer and Kenyon, 2021; Burtsev, 2024; Berkovich and Buehler, 2025; Berkovich et al., 2026). However, these works have primarily focused on one-step prediction or pixel-level accuracy, rather than whether a model can generate a correct long-horizon continuation (rollout) of the system. For example, a model can achieve 99.8% pixel accuracy while maintaining a correct rollout for only 64% of the evaluated trajectories (Figure 3). Here, we evaluate standard architectures, including convolutional neural networks, transformers, and diffusion models, on cellular automata using rollout success as a metric for long-horizon prediction. Surprisingly, we find that

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.

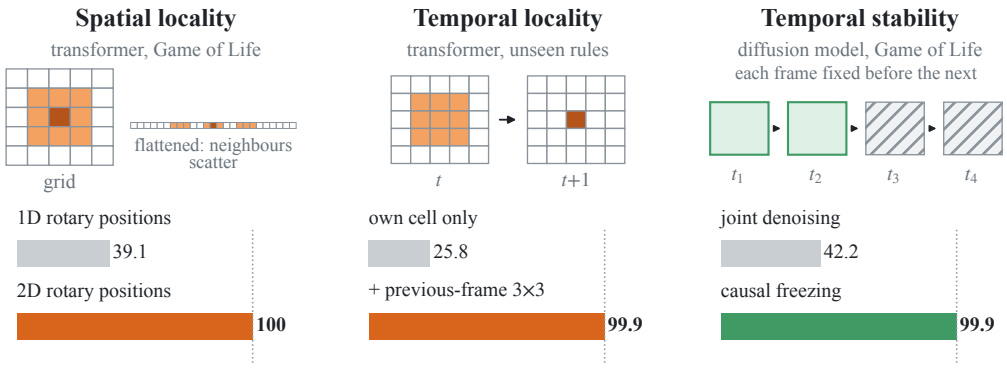


Figure 1: **Three failures of standard models, and the change that repairs each** (bars: exact rollouts, %). Each panel isolates one property; the grey boxes below state the failure and its fix, and Sections 3–5 give the evidence. *Left, spatial locality*: a transformer that reads the grid as a flat token sequence must know which tokens are neighbours; one-dimensional positions lose this at the torus edge, two-dimensional positions restore it. *Middle, temporal locality*: on an unseen rule the model must pair each situation with what followed it one frame later; handing each token its cell's previous-frame 3×3 neighbourhood supplies the pair. *Right, temporal stability*: a predicted frame must be settled before the next is built on it; the same diffusion weights fail when all frames are denoised jointly and succeed when frames are fixed in order (causal freezing).

these architectures fail to reliably complete the task, despite their high short-term prediction accuracy. This motivates us to investigate why current architectures struggle with this seemingly simple task and how they can be improved. We identify a set of simple yet general inductive biases that are absent from current architectures; incorporating these biases enables the models to continue cellular automata exactly over long horizons.

The three inductive biases we identify are as follows (Figure 1):

Spatial locality — step 1, seeing the situation (Section 3)

Failure mode: flattening the grid into one token sequence hides which cells are neighbours. With one-dimensional rotary positions, a transformer executes the known Game of Life in only 39.1% of rollouts, and its errors gather on the wrap-around edge.
Solution: two-dimensional positions, with the parameters unchanged: 100%.

Temporal locality — steps 2–3, matching a situation to what followed it (Section 4)

Failure mode: a situation and its outcome lie one frame apart, and standard models rarely learn to bind them: the standard CNN completes 18.9% of unseen-rule rollouts; no standard transformer within our budget is exact.
Solution: store each situation with its successor—a two-frame convolution, a KV shift (Xu et al., 2024), or the previous-frame neighbourhood on each token, which lifts the same transformer from 25.8% to 99.9% (Figure 5).

Temporal stability — the rollout loop, settling each frame before it is used (Section 5)

Failure mode: joint denoising computes later frames from earlier ones that are never settled; more network calls and deeper denoisers do not fix it.
Solution: causal freezing (Chen et al., 2024a)—denoise the frames in order, keeping each as clean context: with weights unchanged, a depth-8 plain denoiser rises from 42.2% to 99.9% on the Game of Life, while billiards barely changes (Figure 6).

Each section asks the same three questions of one step: is the needed information reachable, does training learn to use it, and which small change supplies the missing link? Reachability is checked before training (Section 2): in the CNN and the transformer the evidence was within reach—inside the receptive field or the context—and what failed was learning to use it, whereas the standard

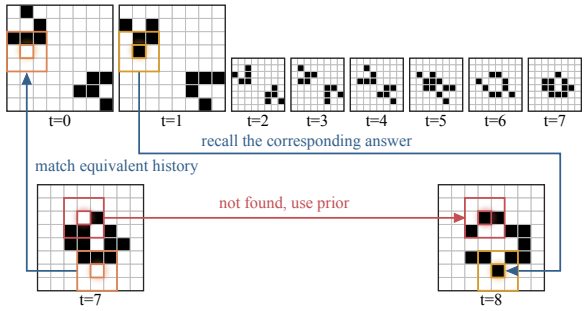


Figure 2: **The information flow required for prediction.** The grid is observed at $t = 0, \ldots, 7$; the task is to predict the state of a queried cell at $t = 8$.
Step 1: identify the current situation. At $t = 7$, record the queried cell's state and the number of black cells among its eight neighbours (orange box).
Step 2: match a past situation (blue). Find an earlier observed transition with the same central state and neighbour count; the match shown occurs at $t = 0$.
Step 3: recall its outcome (blue). Read the matched cell's state in the following frame ($t = 1$) and copy that bit to the queried cell at $t = 8$; the same fixed rule maps identical situations to identical outcomes.
No match (red). Direct retrieval cannot determine the next state, so prediction relies on prior knowledge of the rule family; the L2 and L3 test corpora contain no such cells, and L4 tests them separately. The 8×8 torus has periodic boundaries (edges wrap).

Goal: see 8 frames, predict next 8 exactly	
L1	a fixed rule (Game of Life)
L2	seen worlds; context tells which
L3	unseen worlds; context gives the rule
L4	evidence withheld; a prior is needed

Table 1: The four evaluation levels.

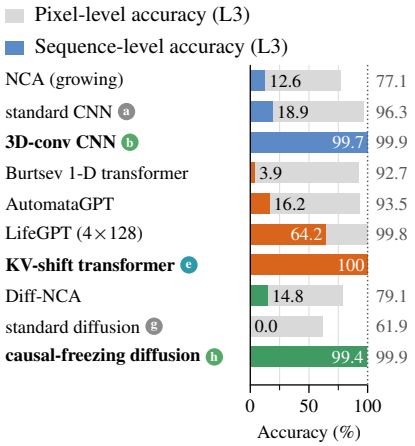


Figure 3: **Two metrics on held-out rules.** Grey: pixel-level accuracy, the fraction of predicted cells correct. Coloured: SeqAcc, the fraction of rollouts with all $8 \times 64 = 512$ cells correct. Means over three seeds; per-seed values in Appendix D.

denoiser's four-layer field does not even reach the oldest observed frames. Each comparison changes one link and holds the rest of the model fixed. None of the devices is new—the KV shift comes from language models, causal freezing from video diffusion; what the automaton adds is a world in which the reason each one is needed can be stated in advance and tested. Related work is discussed in Appendix A.

2 Cellular Automata as a Testbed

In the CA testbed, each world is a binary outer-totalistic cellular automaton on an 8×8 torus. Its rule is an 18-row lookup table

$$r : \{0,1\} \times \{0,\ldots,8\} \to \{0,1\}, \tag{1}$$

which maps a cell's own state s and its number of black neighbours n to its next state. We call (s, n) the cell's *situation*. There are $2^{18} = 262{,}144$ rules, including the Game of Life. A fixed-seed permutation splits them into two disjoint halves of 131,072 rules, one for training and one held out.

A model observes 8 frames and predicts the next 8, a **rollout**: autoregressive models generate it one frame at a time, feeding each prediction back as input, while a diffusion model denoises all eight frames jointly.

Four levels grade the difficulty (Table 1). L1 fixes one world, the Game of Life: can the model execute a known rule? L2 uses new trajectories of training-half worlds. L3, the headline level, uses held-out worlds, so the rule must be inferred from the eight observed frames alone. L4 builds on L3, but the context may not contain the answer: a queried situation may never appear among the observed transitions. The model must then use a prior we build into the rule family: two situations with opposite centre cells and the same neighbour count have opposite outcomes, $r(1, n) = 1 - r(0, n)$ for $n \in \{3, 4\}$. So it looks for the centre-flipped situation in the context, which is guaranteed to be there, and negates what followed it.

The metric is strict sequence accuracy,

$$\mathrm{SeqAcc} = \Pr[\text{all } 8 \times 64 = 512 \text{ predicted cells correct}]\,. \tag{2}$$

Test corpora are *self-consistent*: every queried situation appears among the observed transitions, except at L4, which withholds it on purpose. Without this filter, a third of the trajectories query a situation that the observed frames never show, so the observations do not determine their continuation. Training streams are filtered the same way. Unless a caption says otherwise, numbers are final-step values without checkpoint selection, averaged over three seeds; the few exceptions—figure cells with fewer registered seeds, and the particle experiment of Appendix G.2, whose checkpoint is validation-selected—are marked where they appear. Per-seed values and training budgets are in Appendices C and D.

Standard models. We build three standard next-frame models as a language or video modeller would, and train them on identical data. The standard CNN (a) stacks the eight observed frames as input channels; its 9×9 receptive field covers the torus. The standard transformer (c) is a causal decoder over raster tokens: sixteen frames flattened into 1024 tokens with learned absolute positions, at four widths (0.17M–6.6M parameters, 2–8 layers). The standard diffusion model (g) is a four-layer 3D-convolutional denoiser in which all eight predicted frames share one noise level. In the CNN and the transformer every observed transition can reach every prediction; the standard denoiser is narrower, its depth-4 temporal field reaching only the four most recent observed frames.

Scored per pixel, the metric of prior work, every baseline looks nearly solved; scored per sequence, all fail (Figure 3). Pixel-level accuracy (Berkovich and Buehler, 2025; Berkovich et al., 2026; Burtsev, 2024) hides rule errors: at 99.9% per pixel, independent errors would still spoil about two in five rollouts (Appendix B). Every baseline in Figure 3 with a one-step prediction path—ours, and those adapted from prior work within a predeclared budget of 0.5–1.5M parameters—gets at least 77% of pixels right one step ahead, yet few rollouts right: the best, the LifeGPT of the introduction (Berkovich and Buehler, 2025), completes only 64.2% of held-out rollouts, and every other at most 18.9%. The failures are not shared evenly: flattening the grid is the transformer's problem, joint denoising the diffusion model's, and binding a situation to its outcome everyone's. Figure 4 lists our models, grouped by the step of Figure 2 that Sections 3–5 supply.

3 Spatial locality: keep neighbours together

With one-dimensional rotary positions, a transformer executes the Game of Life in 39.1% of rollouts, and 95% of its errors fall on the 28 border cells. A cell's next state depends on its 3×3 neighbourhood. A convolution sees this neighbourhood directly, and the standard CNN executes the Game of Life exactly (L1 in Figure 4). A raster transformer instead reads each frame as 64 consecutive tokens: a cell's previous-frame neighbourhood becomes three runs of three tokens about 64 positions earlier, and at the edge of the torus each run wraps to the other side of its row. With learned absolute positions, the standard transformer learns this layout and is exact. With rotary position encoding (RoPE) along the raster sequence, as in most language models, the d64 transformer is not (Table 5, Appendix E): at the wrap-around edge a fixed raster offset no longer points to a fixed neighbour, and that is where the errors sit.

Two-dimensional positions, with the same parameters, give 100%. Rotating by row and column separately (axial 2D RoPE), with identical parameters and initialization, gives 100% on every seed. Toroidal and deliberately mis-periodic variants also reach 99.4–100%: what matters is two-dimensional position, not the exact period. On unseen rules with one token per cell, 2D rotary positions lift the d128 transformer from 82.6% to 99.3%, higher on every seed, and at d64 neither scheme exceeds 18%; with the previous-frame neighbourhood added to each token, the preregistered 2D-over-1D gains missed their per-seed thresholds (Table 6, Appendix E).

Attention can learn the layout; what it lacks is reliability. With learned positions the same stack is exact on the known rule and reaches 85.6% on unseen rules at d128—yet no width is exact within our budget (Appendix E.3). The previous-frame binding of Section 4 makes the same decoders near-exact with a few hundred added parameters. The lesson is not that attention cannot represent locality but that it does not reliably learn it. Structure buys reliability, not capability—and reliability is the reason to build locality in rather than stack width.

	L1 known rule	L2 seen worlds	L3 unseen worlds	L4 with prior	params
Standard models					
a CNN	100	22.6	18.9	10.0	116k
c transformer, 2 layers (d64)	100	28.7	25.8	5.3	166k
c transformer, 4 layers (d128)	100	86.7	85.6	78.6	925k
c transformer, 6 layers (d192)	100	39.5	37.3	43.1	2.87M
c transformer, 8 layers (d256)	100	54.1	52.7	0.2	6.58M
g diffusion	24.9	0.0	0.0	0.0	86k
Spatial locality (Sec. 3)					
transformer, raster RoPE, 2 layers (d64)	39.1	20.3	17.2	1.3	100k
transformer, axial RoPE, 2 layers (d64)	100	19.2	15.8	0.9	100k
Temporal locality (Sec. 4)					
transformer, 2 layers (d64) + last-frame 3×3	100	99.9	99.9	73.2	167k
transformer, 4 layers (d128) + last-frame 3×3	100	99.9	99.9	97.2	926k
d 2D-conv transformer, 2 layers	100	89.1	88.8	79.9	142k
e KV-shift transformer, 1 memory read	100	100	100	42.2	2.6k
e KV-shift transformer, 4 memory reads*	100	100	100	99.5	9.8k
f 3D-conv transformer, 1 memory read	100	100	100	71.5	3.8k
f 3D-conv transformer, 4 memory reads	100	100	100	99.5	11k
b 3D-conv CNN	100	99.8	99.7	98.8	18k
Temporal stability (Sec. 5)					
per-frame noise + backward encoder + anchor	32.0	6.1	4.5	0.1	30k
h′ same denoiser, frame-ordered sampling	100	30.9	27.9	24.8	30k
h same denoiser, order-matched training	100	99.3	99.4	93.1	30k

Figure 4: **Main results.** SeqAcc (%) at the four levels, mean over registered seeds (per-seed values in Figure 10). Rows are grouped by the property of Sections 3–5 that they test. Standard-transformer rungs tie depth to width (d64–d256, 2–8 layers, four heads). A "memory read" is one attention retrieval over the frozen evidence memory (Table 14); * marks an L4 cell trained with the evidence-balanced loss. The three denoiser rows share one architecture—per-frame noise levels, a backward-looking canvas encoder, and a clean anchor frame added to the standard denoiser—sampled jointly, sampled in frame order with the same weights (h′, causal freezing), or trained matched to that order (h). Coloured letters index the design-space panels (Figures 7–9); protocol exceptions are in Table 2.

Spatial locality is necessary but not sufficient. The standard CNN, local by construction, executes the Game of Life exactly yet fails most held-out rollouts; what it lacks is temporal locality.

4 Temporal locality: bind each state to its successor

Induction needs each observed situation and the outcome it produced in one place that the current query can find. In the input the two lie a frame apart, and no standard model within our budget reliably learns to bring them together. We call models given this binding *momentum induction* models: each observed situation is stored with its continuation, so recalling the situation carries the continuation with it. The models of the main text receive structure and initialization only, never extra supervision, apart from the L4 cells marked * in Figure 4, whose loss uses a training-side task label (Appendix C). *Learned* models train their detectors from random initialization; *analytic* models use hand-set detectors and train only the head.

The standard CNN sees the whole torus and all eight frames, yet completes 18.9% of held-out rollouts, and five times the training steps do not change this (Appendix D). Convolutions can represent the rule exactly (Gilpin, 2019), but even the fixed Life rule is hard to learn (Springer and Kenyon, 2021). The standard transformer leaves the marginal predictor, which ignores the evidence and scores 4.1%, within its first 18k steps or never: 200k steps and a halved learning rate free no

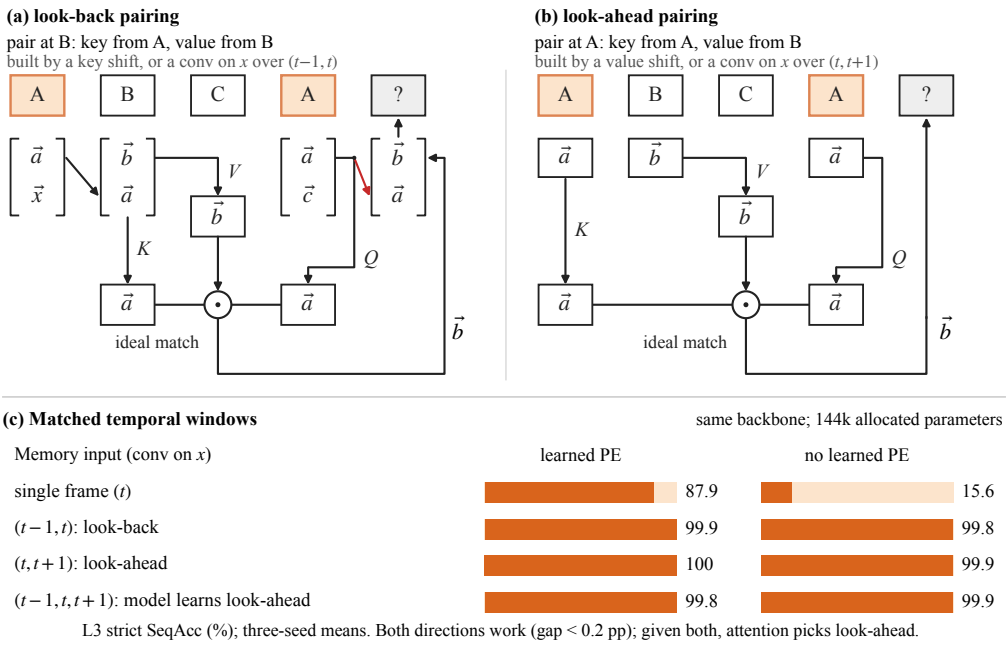


Figure 5: **Two kinds of induction.** (a) Look-back pairing stores the observed pair A→B at B, whose key carries its predecessor A. (b) Look-ahead pairing stores it at A, whose value carries its successor B. Either way, the second A retrieves B. (c) One memory-attention backbone (144k parameters, identical initialization) whose memory is a convolution on the input x over one of four temporal windows, with and without learned positions (L3 SeqAcc, %). A KV shift builds (b) by index; a convolution can build either. Every window that spans a transition works, in either direction; the six symmetric-window models (two position settings, three seeds) attend in the look-ahead convention.

stuck run, and at the two widest models one seed in three stays stuck (Appendix E.3). Runs that escape infer rather than recall—L2 tracks L3 at every width—yet none is exact. This falsifies both of our preregistered predictions for this model: that it would stay near zero, and that accuracy would rise monotonically with width (Appendix B.1).

4.1 Convolution: count, then look up

A convolution over two adjacent frames makes the CNN nearly exact. The 3D-conv CNN introduces the front end that the structured models below share (Appendix Figure 7). A 3×3 convolution maps each cell to an 18-dimensional situation code, and a 2×3×3 convolution over two adjacent frames maps each observed transition to a 36-dimensional (situation, outcome) code. A linear aggregator sums the transition codes into a tally (18 situations × 2 outcomes), and a gated head looks up the current situation in it. With analytic detectors the model is exact at L3; learned, it reaches 99.7%. Every inexact learned run misses the detector for one of the two rarest situations (each about 0.2% of cells), and more budget, width, or batch does not repair it (Appendix F).

4.2 Transformer: look-back and look-ahead pairing

The classic induction circuit of language models pairs by looking back (Elhage et al., 2021; Olsson et al., 2022). In the sequence A B C A (Figure 5a), a previous-token head copies A into the residual stream at B, so the key at B carries A and its value carries B; the second A matches that key and retrieves B. In look-ahead pairing (Figure 5b), each token keeps its own content as key and its value carries its successor, so the pair is stored at A. Both are causal here: the current frame only queries, and every stored pair belongs to a frame whose successor has been observed. A short convolution builds either pair (So et al., 2021; Fu et al., 2023; Arora et al., 2023; Allen-Zhu, 2025), and so does a KV shift (Xu et al., 2024).

Supplying the pair matters more than scale. The 3D-conv transformer (f) turns each observed transition into one token whose key is the situation and whose value is the outcome, and retrieves with a single head at a frozen temperature of 20 and no positional encoding. With 3.8k learned parameters it is exact at L1–L3 on every seed. The KV-shift transformer (e) builds the same pair by index—key from frame t, value from frame $t{+}1$ (Lin et al., 2019; Peng et al., 2023)—and is exact on every seed with 2.6k parameters, over two thousand times fewer than the 6.6M-parameter standard transformer, which is not exact (Figure 4). The 2D-conv transformer (d) must learn the pair: it encodes each frame separately and passes the unpaired tokens, with learned positions, through two attention layers. With 142k parameters it reaches 88.8% (Table 9, Appendix F).

The 2D-conv transformer learns to pair by looking back, but its retrieval stays soft. On every seed its first hop attends to the same cell in the previous frame and almost never to the next one (Table 8, Appendix F). Its learned temperature settles below 1, so attention stays spread and small first-step errors compound over the rollout. Errors concentrate on situations observed only once or twice, where many near matches outweigh a lone exact one (Singh et al., 2024; Veličković et al., 2024). On an unfiltered stream, freezing the temperature, five times the budget, and doubled width do not repair it (Appendix F).

With the backbone fixed, any window that spans a transition works (Figure 5c). One memory-attention model, with 144,050 allocated parameters and identical initialization in every arm, builds its memory from one of four temporal windows, with or without learned positions. Every window spanning a transition—$(t{-}1, t)$, $(t, t{+}1)$, or $(t{-}1, t, t{+}1)$—reaches at least 99.8% with or without positions, against 87.9% and 15.6% for the single frame (t) (Table 11, Appendix F). The two directions perform alike, and the six symmetric-window models ($(t{-}1, t, t{+}1)$; two position settings, three seeds) attend mostly to the historical situation itself rather than to its successor (Table 12, Appendix F): they match A and read what followed it, the look-ahead convention of the KV-shift transformer.

The same binding rescues the standard transformer, given positions. Attaching to each raster token its cell's $3{\times}3$ neighbourhood in the previous frame (704 parameters at d64) lifts the d64 transformer from 25.8% to 99.9%, and d128 from 85.6% to 99.9% (the "last-frame $3{\times}3$" rows of Figure 4). With no position encoding at all, the same decoders reach only 8.1–8.2% (Table 10, Appendix F). This feature supplies spatial alignment and the temporal pair at once; the matched windows above isolate the temporal part. Binding is not a drop-in replacement everywhere: as the front end of the structured diffusion model of Section 5, the KV shift reaches 49.0% against 97.7% for the convolution (Appendix F). What carries across designs is the requirement—each observed outcome bound to a situation the query can match—not one implementation.

4.3 Level 4: when induction needs a prior

When the evidence does not contain the answer, the model needs a prior and a route to apply it. At L4 the queried situation never appears in the observed frames; the answer follows from the partner row's observed outcome only through knowledge of the rule family—here, that the two rows are negations of each other. Matching alone cannot supply it. A tally over all evidence is one route: the 3D-conv CNN reaches 98.8%, and the structured diffusion model (Section 5), same tally, 93.1%. A single attention read is not: the KV-shift and 3D-conv transformers, exact at L3, drop to 42.2% and 71.5%. Four stacked reads are another route: 99.5% for the 3D-conv transformer under the ordinary loss, and 99.5% for the KV-shift transformer once the loss is balanced over the 0.18% of training cells that need the partner (Table 14). The standard d128 decoder with the previous-frame neighbourhood reaches 97.2%. Targeted data matters more than training time: at the natural rate of the event, three times the steps never match the 10% dose (Table 13, Appendix F; cf. Chan et al., 2022; Reddy et al., 2024). A matched inductive bias lowers how much targeted data a prior needs, a cost that matters most where such data are scarce.

5 Temporal stability: settle each frame before it is used

Rollout adds a third requirement: a predicted frame that becomes the input for the next must be settled first. Diffusion models make this testable, because the same weights can generate all frames

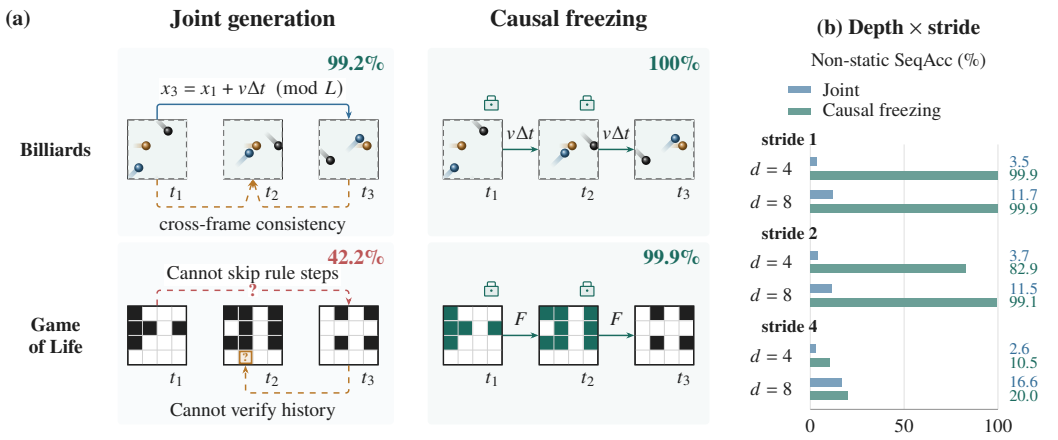


Figure 6: **Billiards and the Game of Life under the same denoiser architecture.** (a) One plain denoiser trained per world, weights fixed across samplers, sampled jointly or with causal freezing at eight calls per rollout (SeqAcc). Billiards can be extrapolated across frames; in Life one call cannot skip rule steps, and a later frame cannot verify an earlier one. The drawings illustrate these properties, not model outputs. (b) Life, both samplers, across network depth d and the number of rule steps per displayed frame (stride).

jointly or one at a time. We use two settings: structured denoisers on unseen rules, and a plain denoiser on the Game of Life and on billiards.

The standard diffusion model is correct for two or three frames and then collapses. It completes no held-out rollout on any seed, and only 24.9% even on the Game of Life. Under uniform joint denoising no predicted frame is ever settled, so each prediction is revised together with the frames it depends on. Joint denoising could carry out the computation in principle: given an exact transition F, the recurrence $\hat{x}_k^{(r+1)} = F(\hat{x}_{k-1}^{(r)})$ with $\hat{x}_0^{(r)} = x_0$ extends a correct prefix by one frame per round. Training does not find it.

With the weights fixed, changing only the order of denoising lifts held-out completion from 4.7% to 76.3%. We train two structured denoisers (main and curriculum recipes) with independent per-frame noise levels, a backward-looking canvas encoder, and a clean anchor frame (Appendix Figure 9). Sampled jointly over 49 network calls, they complete 4.7% of held-out rollouts. Sampled with *causal freezing* —denoise each frame in order, then keep it as clean context (Chen et al., 2024a)—the same weights reach 76.3% with 48 calls, and 88.2% with one call per frame. Retrained under the common recipe of Figure 4, the same swap lifts the denoiser from 4.5% to 27.9%: the direction holds under both recipes, the size does not. The weights already encode the transition; the joint schedule does not turn it into a continuation.

Freezing matters for the Game of Life, not for billiards (Figure 6a). We train one plain depth-8 3D-convolutional denoiser (196k parameters) separately on free billiards and on the Game of Life, and change only the sampler, at eight network calls either way. Joint generation reaches 99.2% on billiards against 100% with freezing, but 42.2% on Life against 99.9%. A billiard ball's later position follows from its current position and velocity in one step. Life must be composed one local update at a time, and a later Life state can have several predecessors, so it cannot uniquely confirm an earlier guess.

Depth does not rescue joint generation; freezing degrades only when one call must compose several rule steps (Figure 6b). Going from four to eight layers leaves joint generation far behind freezing. When each displayed frame is two rule steps after the last, depth-8 freezing still reaches 99.1%; at four steps, both samplers fail on most worlds. The successful sampler behaves as one-step autoregression trained with a denoising objective: a shallow denoiser composes few rule steps per call, and freezing supplies the intermediate states that joint training does not learn to keep.

Training matched to the schedule gives 99.4%. A clean prefix, one noisy target frame, noise beyond it, and loss on that frame only—the diffusion analogue of exposure-bias remedies (Bengio

et al., 2015; Huang et al., 2025)—define the structured diffusion model : 99.4% on held-out rules with 30k parameters (Appendix F.7). Per-frame noise levels and ordered generation already appear in streaming video diffusion (Chen et al., 2024a; Ruhe et al., 2024; Kim et al., 2024; Yin et al., 2025; Song et al., 2025; Sand AI, 2025); our experiments show their role in exact continuation.

Left to choose, the denoisers commit frames in causal order. Committing the most confident frame at each stage (Chang et al., 2022; Kim et al., 2025) selects exactly the order $1 \to 8$ in about 98% of worlds and matches the imposed order's accuracy; least-confident-first and random orders stay below 5% (Table 16, Appendix F). These weights carry directional priors (the backward encoder, the anchor), so this is order selection under those priors, not discovery of causality. Order need not mean one frame per call: the plain denoiser commits several Life frames per call and keeps 99.8% (Appendix G.1).

6 What the three properties have in common

Each property is one link in Figure 2: spatial locality brings a cell's neighbours together (step 1), temporal locality (momentum induction) pairs each observed situation with its outcome (steps 2–3), and temporal stability settles each prediction before later ones use it (the rollout loop). Reachability can be checked before training; learnability only by training, and that is where every standard model within our budget fell short. When the evidence does not contain the answer (Section 4.3), the missing link is a prior about the world's rule family, and the same two tests apply: does a route exist, and does training use it?

This suggests a short checklist for a new world. (i) Write down which cells and frames must meet, and which parts of the future depend on which. (ii) Check that some part of the model can see both members of each pair. (iii) Supply the links that training is unlikely to find: a position scheme, a pairing, a commitment order. It is not a new method, and is close in spirit to algorithmic alignment (Xu et al., 2020) and to Canon layers (Allen-Zhu, 2025). Settling intermediate states resembles the role of chain of thought in transformers (Feng et al., 2023; Merrill and Sabharwal, 2023).

7 Future Work

Our evidence comes chiefly from a deterministic family of binary automata on an 8×8 grid; the physical and video directions below remain tests of transfer. Small exploratory experiments on continuous dynamics (Appendix G.2, Figures 12–13) already suggest that the answer depends on the dynamics: with shared weights, causal freezing helps Burgers but not linear advection–diffusion, while the locality and pairing results are mixed.

Exact physical generation. We will test the three properties in richer systems—interacting particles and continuous dynamics—where a known simulator makes fidelity checkable over a specified horizon: exact agreement for discrete states, declared tolerances for continuous ones. This separates executing the dynamics from the visual plausibility of a rendered rollout; the experiments of Appendix G.2 are a first step.

From latent dynamics to rendered video. Instead of showing CA states directly to the model, a rendered automaton keeps the simple rules while adding textures, lighting, and camera motion. A video model could encode the observed frames, infer transitions with the two-frame pairing in a learned latent space, and decode the predicted states into frames; we will ask whether this improves state-level consistency without sacrificing visual quality, relative to an otherwise matched predictor (Appendix H). The question is whether the same information flow survives perception and rendering.

Ethics Statement

This work studies small synthetic cellular-automaton systems generated by a documented simulator. It involves no human subjects, personal data, or deployed systems, and we foresee no direct ethical risk from the models or the data.

Reproducibility Statement

Every number in this paper is regenerated from per-run logs: each registered run records its command, seeds, hardware, a SHA-256 fingerprint of the exact training code, and the checksums of the pinned evaluation corpora used, and every reported number is transcribed from these logs without manual editing. One supplementary probe—the attention-depth ladder at L4—was registered from a retrospectively preserved source snapshot rather than a prelaunch code fingerprint; its predictions and checkpoints passed the same audits. Training data are generated deterministically from (seed, index) by a version-frozen simulator, so no training corpus needs to be shipped; evaluation corpora are pinned files with SHA-256 sidecars that regenerate identically on any machine. Protocol, architecture, and recipe details are in Appendices B–D. The full codebase—simulator, single-entry training script, evaluation and analysis tooling, and the pinned evaluation corpora—is included as anonymized supplementary material with this submission, and will be released publicly upon publication.

Statement on the Use of AI Assistance

In this work, we used generative AI tools (large-language-model assistants) for: providing feedback on research methodology and experiment design, with several of the paper's techniques—including the constructor-swap and sampler-swap controls—originating in human–AI collaborative design sessions; helping develop the conceptual framework (the information-flow requirements and the audit protocol); implementing methods (experiment scripts, model code, and analysis tooling); interpreting results (drafting diagnostic summaries from run logs); assisting with translation between the authors' working language and English; and drafting and editing parts of the paper for readability. Additionally, we used them for identifying and verifying literature, formatting references, creating figure-generation scripts, and suggesting experimental parameters. We did not use generative AI tools to generate synthetic datasets—all data are produced by a deterministic, version-frozen simulator from (seed, index)—and this work contains no mathematical proofs, so proof-related assistance is not applicable. We have reviewed all AI-assisted work: every experimental number was checked by the authors against the registered run logs; citation keys were checked against the bibliography, and bibliographic metadata was checked against available arXiv, CrossRef, and publisher records where applicable; AI-assisted code is covered by a test suite, including bit-identity tests for every compatibility flag; and the authors directed and adjudicated all design proposals. We take full responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Z. Allen-Zhu. Physics of Language Models: Part 4.1, Architecture Design and the Magic of Canon Layers. *arXiv:2512.17351*, 2025.

S. Arora et al. Zoology: Measuring and Improving Recall in Efficient Language Models. *arXiv:2312.04927*, 2023.

H. Bansal et al. VideoPhy: Evaluating Physical Commonsense for Video Generation. *arXiv:2406.03520*, 2024.

P. W. Battaglia et al. Relational inductive biases, deep learning, and graph networks. *arXiv:1806.01261*, 2018.

D. M. Bear et al. Physion: Evaluating Physical Prediction from Vision in Humans and Machines. In *NeurIPS Datasets and Benchmarks*, 2021. arXiv:2106.08261.

S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *NeurIPS*, 2015.

J. A. Berkovich and M. J. Buehler. LifeGPT: Topology-agnostic generative pretrained transformer model for cellular automata. *npj Artificial Intelligence*, 1:23, 2025. arXiv:2409.12182.

J. A. Berkovich, N. S. David, and M. J. Buehler. AutomataGPT: Transformer-based forecasting and ruleset inference for two-dimensional cellular automata. *Advanced Science*, 13(33):e11352, 2026. arXiv:2506.17333.

T. Bricken et al. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023.

J. Bruce et al. Genie: Generative Interactive Environments. In *ICML*, 2024. arXiv:2402.15391.

M. Burtsev. Learning elementary cellular automata with transformers. In *MATH-AI: The 4th Workshop on Mathematical Reasoning and AI (NeurIPS 2024 Workshop)*, 2024. arXiv:2412.01417.

S. Chan, A. Santoro, A. Lampinen, J. Wang, A. Singh, P. Richemond, J. McClelland, and F. Hill. Data distributional properties drive emergent in-context learning in transformers. In *NeurIPS*, 2022.

H. Chang, H. Zhang, L. Jiang, C. Liu, and W. T. Freeman. MaskGIT: Masked Generative Image Transformer. In *CVPR*, 2022. arXiv:2202.04200.

R. T. Q. Chen et al. Neural Ordinary Differential Equations. In *NeurIPS*, 2018. arXiv:1806.07366.

B. Chen, D. Martí Monsó, Y. Du, M. Simchowitz, R. Tedrake, and V. Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In *NeurIPS*, 2024.

M. Cranmer et al. Lagrangian Neural Networks. In *NeurIPS*, 2020. arXiv:2003.04630.

B. L. Edelman et al. The Evolution of Statistical Induction Heads: In-Context Learning Markov Chains. In *NeurIPS*, 2024. arXiv:2402.11004.

C. Ekbote et al. What One Cannot, Two Can: Two-Layer Transformers Provably Represent Induction Heads on Any-Order Markov Chains. In *NeurIPS*, 2025. arXiv:2508.07208.

N. Elhage et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021.

N. Elhage et al. Toy models of superposition. *Transformer Circuits Thread*, 2022.

G. Feng et al. Towards Revealing the Mystery behind Chain of Thought: A Theoretical Perspective. In *NeurIPS*, 2023.

D. Y. Fu, T. Dao, K. K. Saab, A. W. Thomas, A. Rudra, and C. Ré. Hungry Hungry Hippos: Towards Language Modeling with State Space Models. In *ICLR*, 2023.

M. Gardner. Mathematical games: The fantastic combinations of John Conway's new solitaire game "Life". *Scientific American*, 223(4):120–123, 1970.

N. Gillman et al. Force Prompting: Video Generation Models Can Learn and Generalize Physics-based Control Signals. *arXiv:2505.19386*, 2025.

W. Gilpin. Cellular automata as convolutional neural networks. *Physical Review E*, 100(3):032402, 2019.

S. Greydanus et al. Hamiltonian Neural Networks. In *NeurIPS*, 2019. arXiv:1906.01563.

D. Ha and J. Schmidhuber. World models. *arXiv:1803.10122*, 2018.

D. Hafner et al. Mastering Diverse Domains through World Models. *arXiv:2301.04104*, 2023.

A. Haviv, O. Ram, O. Press, P. Izsak, and O. Levy. Transformer Language Models without Positional Encodings Still Learn Positional Information. In *Findings of EMNLP*, 2022.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

J. Ho, T. Salimans, A. Gritsenko, W. Chan, M. Norouzi, and D. J. Fleet. Video diffusion models. In *NeurIPS*, 2022.

X. Huang et al. Self Forcing: Bridging the Train-Test Gap in Autoregressive Video Diffusion. *arXiv:2506.08009*, 2025.

B. Heo, S. Park, D. Han, and S. Yun. Rotary Position Embedding for Vision Transformer. In *ECCV*, pages 289–305, 2024. arXiv:2403.13298. DOI: 10.1007/978-3-031-72684-2_17.

J. Kalkhof, A. Kühn, Y. Frisch, and A. Mukhopadhyay. Frequency-Time Diffusion with Neural Cellular Automata. *arXiv:2401.06291*, 2024.

B. Kang, Y. Yue, R. Lu, Z. Lin, Y. Zhao, K. Wang, G. Huang, and J. Feng. How far is video generation from world model: A physical law perspective. In *International Conference on Machine Learning*, 2025.

A. Kazemnejad et al. The Impact of Positional Encoding on Length Generalization in Transformers. In *NeurIPS*, 2023.

J. Kim et al. FIFO-Diffusion: Generating Infinite Videos from Text without Training. In *NeurIPS*, 2024. arXiv:2405.11473.

J. Kim et al. Train for the Worst, Plan for the Best: Understanding Token Ordering in Masked Diffusions. In *ICML*, 2025.

Y. LeCun. A path towards autonomous machine intelligence. Technical report, OpenReview, 2022.

K. Li, A. K. Hopkins, D. Bau, F. Viégas, H. Pfister, and M. Wattenberg. Emergent World Representations: Exploring a Sequence Model Trained on a Synthetic Task. In *ICLR*, 2023. arXiv:2210.13382.

J. Lin et al. TSM: Temporal Shift Module for Efficient Video Understanding. In *ICCV*, 2019. arXiv:1811.08383.

B. Liu, J. T. Ash, S. Goel, A. Krishnamurthy, and C. Zhang. Transformers Learn Shortcuts to Automata. In *ICLR*, 2023.

Z. Liu, S. Sanborn, S. Ganguli, and A. Tolias. From Kepler to Newton: Inductive biases guide learned world models in transformers. *arXiv:2602.06923*, 2026.

S. McLeish et al. Transformers Can Do Arithmetic with the Right Embeddings. In *NeurIPS*, 2024. arXiv:2405.17399.

W. Merrill and A. Sabharwal. The Expressive Power of Transformers with Chain of Thought. *arXiv:2310.07923*, 2023.

V. Micheli et al. Transformers are Sample-Efficient World Models. In *ICLR*, 2023. arXiv:2209.00588.

A. Mordvintsev, E. Randazzo, E. Niklasson, and M. Levin. Growing neural cellular automata. *Distill*, 2020.

S. Motamed et al. Do generative video models understand physical principles? In *WACV*, 2026. arXiv:2501.09038.

E. Nichani et al. How Transformers Learn Causal Structure with Gradient Descent. In *ICML*, 2024. arXiv:2402.14735.

C. Olsson et al. In-context learning and induction heads. *Transformer Circuits Thread*, 2022.

OpenAI. Video generation models as world simulators. Technical report, 2024.

B. Peng et al. RWKV: Reinventing RNNs for the Transformer Era. In *Findings of EMNLP*, 2023. arXiv:2305.13048.

G. Reddy et al. The mechanistic basis of data dependence and abrupt learning in an in-context classification task. In *ICLR*, 2024. arXiv:2312.03002.

D. Ruhe et al. Rolling Diffusion Models. In *ICML*, 2024. arXiv:2402.09470.

A. Sanchez-Gonzalez et al. Learning to Simulate Complex Physics with Graph Networks. In *ICML*, 2020. arXiv:2002.09405.

Sand AI. MAGI-1: Autoregressive Video Generation at Scale. *arXiv:2505.13211*, 2025.

C. Sanford et al. One-layer transformers fail to solve the induction heads task. *arXiv:2408.14332*, 2024.

R. Schaeffer et al. Are Emergent Abilities of Large Language Models a Mirage? In *NeurIPS*, 2023. arXiv:2304.15004.

A. K. Singh et al. What needs to go right for an induction head? A mechanistic study of in-context learning circuits and their formation. In *ICML*, 2024. arXiv:2404.07129.

D. R. So et al. Primer: Searching for Efficient Transformers for Language Modeling. In *NeurIPS*, 2021.

K. Song et al. History-Guided Video Diffusion. *arXiv:2502.06764*, 2025.

J. M. Springer and G. T. Kenyon. It's hard for neural networks to learn the Game of Life. In *International Joint Conference on Neural Networks (IJCNN)*, 2021.

J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding. *Neurocomputing*, 568:127063, 2024. arXiv:2104.09864. DOI: 10.1016/j.neucom.2023.127063.

T. Unterthiner et al. Towards Accurate Generative Models of Video: A New Metric & Challenges. *arXiv:1812.01717*, 2018.

K. Vafa, J. Y. Chen, A. Rambachan, J. Kleinberg, and S. Mullainathan. Evaluating the world model implicit in a generative model. In *Advances in Neural Information Processing Systems*, 2024.

K. Vafa, P. G. Chang, A. Rambachan, and S. Mullainathan. What has a foundation model found? Using inductive bias to probe for world models. In *International Conference on Machine Learning*, 2025.

D. Valevski et al. Diffusion Models Are Real-Time Game Engines. *arXiv:2408.14837*, 2024.

P. Veličković, R. Ying, M. Padovano, R. Hadsell, and C. Blundell. Neural execution of graph algorithms. In *ICLR*, 2020.

P. Veličković, C. Perivolaropoulos, F. Barbero, and R. Pascanu. Softmax is not Enough (for Sharp Size Generalisation). *arXiv:2410.01104*, 2024.

S. Wolfram. Statistical mechanics of cellular automata. *Reviews of Modern Physics*, 55(3):601–644, 1983.

K. Xu, J. Li, M. Zhang, S. S. Du, K. Kawarabayashi, and S. Jegelka. What can neural networks reason about? In *ICLR*, 2020.

M. Xu, W. Cheng, B. Wang, and W. Chen. KV Shifting Attention Enhances Language Modeling. *arXiv:2411.19574*, 2024.

T. Yin et al. From Slow Bidirectional to Fast Autoregressive Video Diffusion Models. In *CVPR*, 2025. arXiv:2412.07772.

A Related work

Cellular automata and rule learning. Cellular automata make global dynamics from local rules (Wolfram, 1983; Gardner, 1970). Convolutional networks can implement cellular-automaton updates by construction (Gilpin, 2019), although learning even the Game of Life from examples can be difficult (Springer and Kenyon, 2021). Neural cellular automata learn a parameterized local update (Mordvintsev et al., 2020); related sequence-model studies include shortcuts on finite automata (Liu et al., 2023), elementary one-dimensional rules with rule-disjoint free runs (Burtsev, 2024), recursive prediction under the fixed Game of Life rule (Berkovich and Buehler, 2025), and two-dimensional rule-supplied forecasting alongside inverse rule inference (Berkovich et al., 2026). Our setting isolates prefix-conditioned continuation on held-out rules: queried situations are covered by observed transitions, and success requires every cell in every predicted frame to be correct (§2). We diagnose known devices under this exact metric without claiming priority for learning across rules.

Spatial locality and trainability. Convolutions expose neighbouring cells directly, whereas a raster sequence requires the model to recover two-dimensional adjacency, including boundary wrap-around. Rotary position embeddings originated in sequence models (Su et al., 2024); two-dimensional variants are established for image tokens (Heo et al., 2024). The choice of position scheme exemplifies how architecture and target computation should align (Battaglia et al., 2018; Xu et al., 2020; Veličković et al., 2020); input parameterization can change which algorithm a transformer learns (McLeish et al., 2024). In our controlled spatial comparisons, the same architecture can represent the rule but its reliability changes when positions or local inputs expose the grid structure (§3). Thus a locality benefit in this testbed does not imply a general inability of attention to compute local updates.

Induction and context-dependent retrieval. Induction-head accounts describe a previous-token operation followed by matching and copying (Elhage et al., 2021; Olsson et al., 2022); controlled work studies how such behaviour forms and depends on data (Singh et al., 2024; Edelman et al., 2024; Chan et al., 2022; Reddy et al., 2024). Theory characterizes representational and training conditions for induction on restricted tasks (Sanford et al., 2024; Ekbote et al., 2025; Nichani et al., 2024). Our coverage-filtered cellular-automaton task calls for matching a local situation in the observed frames and retrieving its successor. The attention patterns we report are consistent with this two-step account, but observational attention and approximate matches do not identify a unique causal circuit (§4.2, Appendix F).

Putting situations beside outcomes. Temporal shifts in video and sequence models (Lin et al., 2019; Peng et al., 2023), short convolutions for recall (Fu et al., 2023; Arora et al., 2023), and shifted keys and values for induction (Xu et al., 2024) provide precedents for placing information from adjacent times in one representation. Our convolution over adjacent frames, index-aligned key–value shift, and previous-frame neighbourhood input adapt such existing devices to observed situation–successor pairs. They differ in where the pair is stored, but each makes the transition available before retrieval. Matched temporal-window controls test the pairing requirement without assigning special status to one direction (§4). The observed gain concerns learnability under this task and training budget, not a new pairing operation.

Diffusion and commitment order. Denoising diffusion underlies video generation (Ho et al., 2020; 2022); per-frame noise levels and ordered or streaming generation already appear in Diffusion Forcing, rolling diffusion, FIFO-Diffusion, and causal video generation (Chen et al., 2024a; Ruhe et al., 2024; Kim et al., 2024; Yin et al., 2025). Decoding order also matters in masked discrete generation (Kim et al., 2025). In our controlled comparison, one denoiser's weights are held fixed while joint denoising and causal freezing use different schedules with 49 and 48 network calls, respectively (§5). The result tests whether settling each predicted frame before later frames use it helps exact continuation in these worlds; it neither introduces frame ordering nor establishes a universal preference for it.

World models and exact evaluation. Othello-GPT probes a learned board-state representation (Li et al., 2023). Other work on learned world models asks whether plausible trajectories reflect the underlying dynamics (Ha and Schmidhuber, 2018; Vafa et al., 2024; 2025); video and physical-reasoning evaluations likewise probe the gap between appearance and physical prediction (Bear et al., 2021; Bansal et al., 2024; Motamed et al., 2026). Measurement choices can change apparent capability (Schaeffer et al., 2023), while video quality measures such as FVD (Unterthiner et al., 2018) answer a different question from whether every predicted cell in every frame follows the rule. We therefore report strict sequence accuracy on rule-disjoint, coverage-filtered continuations and check that the observed evidence determines their targets (§2). The local pairing and frame-order results suggest hypotheses for broader generators, rather than verified transfer beyond this controlled family (§7).

B Protocol and reporting

This appendix fixes the rule split, the test corpora, and the reporting rules.

Rule indexing and split. A rule is an 18-bit integer whose bit $9s + n$ is the next state of situation (s, n). The split is a fixed-seed permutation of the $262{,}144$ indices, because table entries are index bits and a parity or modulo split would tie some entries to one half.

Coverage and the self-consistency filter. Without the filter, 33% of L3 trajectories query a situation that the observed frames never show. If u counts such situations, a uniform prior on the unseen rule entries gives the realized continuation posterior mass 2^{-u}, with mean 0.8099. This is not an upper bound on SeqAcc: averaged over the 2048 observed prefixes, the most probable continuation carries mass 0.8159, or 0.8732 under the prior restricted to the held-out half—conditional references, not measured accuracies. The analytic 3D-conv CNN (Section 4.1) scores 0.8032 on the unfiltered corpus and 1.0000 on its covered subset. Rejection sampling keeps 67% of draws at L2 and L3. Masking the loss on unobserved situations instead of filtering the stream is neutral ($\Delta = 0.0001$, paired).

Level 4 corpora. Level 4 worlds come from held-out rules with the prior of Section 2 built in: the two situations with opposite centre cells and $n \in \{3, 4\}$ black neighbours have opposite outcomes, $r(1, n) = 1 - r(0, n)$. L4B, the Level 4 test of the main text, holds 768 trajectories; in each, a queried situation never appears among the observed transitions while its centre-flipped partner does. They were selected from 3.86M draws, so the event's natural rate is about 2×10^{-4}. Level 4 training therefore replaces 10% of each batch (the *10% dose*) with such trajectories, drawn from a pool of 65,536 training-half trajectories generated with a seed of its own; *natural rate* denotes training without the dose. L4A, a companion test of 2048 trajectories, instead fixes four rule rows across the family and queries one of them without showing it, so the model must recall the family's fixed value.

Pixel-level versus sequence-level accuracy. At 0.999 per pixel, independent errors would still spoil about two in five eight-frame rollouts on the 8×8 torus ($1 - 0.999^{512} \approx 0.4$), and the metric cannot separate a model that has the rule from one that is approximately right.

Reporting conventions. Numbers are final-step values on exponential-moving-average (EMA) weights, without checkpoint selection, over three preregistered seeds (42/43/44). Main-text numbers

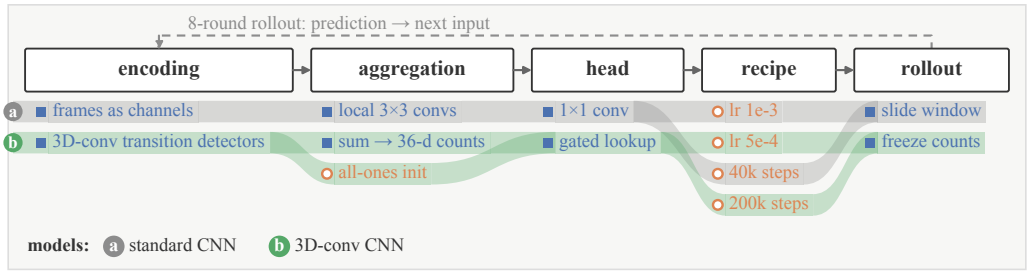


Figure 7: **Design space of the convolutional models.** Columns are pipeline stages; each coloured band traces one model. Squares are architectural choices, circles training choices.

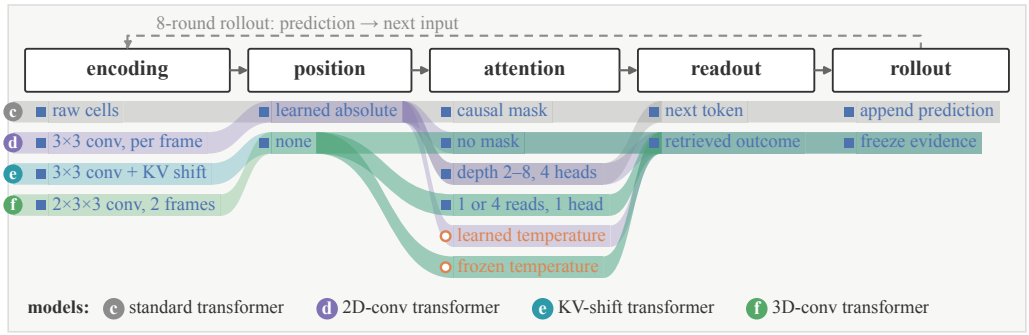


Figure 8: **Design space of the attention models**, read as Figure 7. The encoding column names each model: raw cells, a per-frame 3×3 convolution, the same with a KV shift, or a 2×3×3 convolution over two adjacent frames.

are three-seed means in percent, truncated, not rounded, to one decimal: 0.9995 shows as 99.9, and only an exact 1.0 shows as 100. Appendix tables give the per-seed values behind them. Figure cells with fewer registered seeds are marked where they appear. Two choices used validation data, never test data: the checkpoint of one continuous-dynamics experiment (Appendix G.2) and the number of attention reads in Table 14. Data are generated deterministically from (seed, index); evaluation corpora carry SHA-256 sidecars and hold 512 trajectories at L1, 2048 at L2 and L3, and 768 at L4 (L4B). Models train on the filtered stream with bf16 autocast and are evaluated in fp32 unless a table says otherwise; runs on the unfiltered stream appear only where labelled.

B.1 Preregistered predictions and corrections

Two predictions registered for the standard transformer failed on the filtered stream (Table 7). The first was that it would stay near zero on held-out rules (≈0–5%); d128 reaches 0.9263/0.7646/0.8784 on L3. The second was that accuracy would rise monotonically with width; d128 is the best rung, while d192 and d256 each leave one seed at the marginal predictor. Its other half, that no rung would be exact, held.

C Models and training

Figures 7, 8, and 9 place every model in its family's design space; each band carries the model's letter from Figure 4.

Detection front end. The three structured families share two detectors. A 3×3 circular convolution with two channels, followed by two small ReLU layers, maps each cell to an 18-dimensional situation code (one-hot at the analytic solution); the same stack with a time kernel of 2 and three channels maps each adjacent frame pair to a 36-dimensional (situation, outcome) code per cell. Two

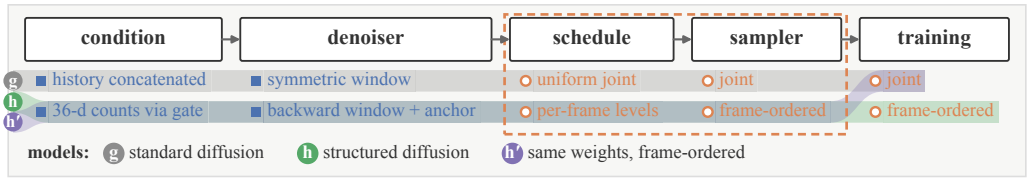


Figure 9: **Design space of the diffusion models**, read as Figure 7. Circles are training and sampling choices; frame-ordered sampling is causal freezing.

ReLU layers are needed because "exactly n black neighbours" is not linearly separable from a single convolution.

How each family uses the codes. The 3D-conv CNN gates each transition channel, sums it with a grouped spatial-sum pyramid, combines the 7 frame pairs linearly per channel, and applies a fixed $\log(1+x)$; a per-cell head reads the resulting tally beside the situation code. The 3D-conv transformer makes each of the 448 observed transitions a token (key: its situation channels; value: its outcome) and queries with the current situation code. Scores are $20\,q \cdot k$, leaking about 10^{-6} of the softmax weight onto non-matching one-hot tokens; a head MLP ($36 \to 24 \to 1$) reads the retrieved value and the code. The structured diffusion model noises each predicted frame by a variance-preserving first-order autoregressive chain (retention 0.9, $T = 50$), with targets ± 5 and belief $\hat{x}_0 = 5\tanh(u/2)$ for network output u. Its canvas encoder looks only backward (frame k reads frames $k-1$ and k), the last observed frame is a clean anchor, and a bilinear gate combines canvas features with the tally. Training matches causal freezing: clean ground truth before the denoised frame, that frame at its noise level, noise beyond, and loss on that frame only.

Parameter counts. 3D-conv CNN 17,994 (analytic 15,361); 3D-conv transformer 3,798 (analytic 913); KV-shift transformer 2,625; 2D-conv transformer 141,698 (104,834 without learned positions); structured diffusion model 29,963; standard transformer 165,825, 924,801, 2,866,753, and 6,581,505 at d64, d128, d192, and d256 (2, 4, 6, and 8 layers), plus 704 at d64 for the last-frame 3×3 input; standard CNN 116,097; standard diffusion model 85,697 (depth 4); plain depth-8 denoiser of the billiards comparison 196,417.

Supervision. All models train on their ordinary loss and are given structure and initialization only, with one exception: the evidence-balanced loss of Table 14 (the L4 cells marked * in Figure 4) averages the loss separately over direct-evidence and partner-only cells, a training-side task label that never enters the model as input. The filtered stream and the 10% L4 dose choose training trajectories and add no loss term.

Scaffolds. Given, not learned: the 3D convolution (time kernel 2); in the 3D-conv transformer, the frozen attention temperature 20 and no positional encoding; in the learned 3D-conv CNN and structured diffusion model, an analytic all-ones initialization of the aggregation subnets; in the structured diffusion model, the backward receptive field, the clean anchor frame, and the per-frame noise schedule. Diffusion has no analytic version; its results rest on learned weights.

Training recipe. Adam ($\beta_2 = 0.999$), gradient clipping at 1.0, 500 warmup steps then cosine decay to 10%, and EMA 0.999; all evaluation uses the EMA weights. All transformers, the standard CNN, and the standard diffusion model train for 40k steps at batch 512 and learning rate 10^{-3}; the 3D-conv CNN trains for 200k steps at batch 128 and the structured diffusion model for 200k steps at batch 2048, both at learning rate 5×10^{-4}. Recipe variants and per-cell exceptions are in Appendix D.

D PER-SEED RESULTS AND TRAINING BUDGETS

Figure 10 gives the three seeds behind every cell of Figure 4, in the same layout. Table 2 repeats the L1, L3 and L4 values at four decimals with exact parameter counts and lists the protocol exceptions;

	params	L1 known rule			L2 seen worlds			L3 unseen worlds			L4 with prior		
Standard models													
a CNN	116k	100	100	100	22.7	22.5	22.6	19.0	18.8	19.0	11.3	7.8	10.9
c transformer, 2 layers (d64)	166k	100	100	100	21.0	31.6	33.7	17.8	28.6	31.2	3.3	5.8	6.7
c transformer, 4 layers (d128)	925k	100	100	100	91.6	79.0	89.6	92.6	76.4	87.8	82.2	65.3	88.4
c transformer, 6 layers (d192)	2.87M	100	100	100	50.9	5.4	62.1	49.0	4.1	58.9	67.7	0.2	61.5
c transformer, 8 layers (d256)	6.58M	100	100	100	5.4	86.2	70.6	4.1	84.7	69.2	0.2	0.2	0.2
g diffusion	86k	19.1	28.1	27.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Spatial locality (Sec. 3)													
transformer, raster RoPE, 2 layers (d64)	100k	39.2	39.6	38.6	19.5	20.7	20.8	16.6	17.4	17.5	1.3	1.4	1.3
transformer, axial RoPE, 2 layers (d64)	100k	100	100	100	19.2	19.2	19.3	16.3	15.5	15.7	0.6	0.9	1.3
Temporal locality (Sec. 4)													
transformer, 2 layers (d64) + last-frame 3×3	167k	100	100	100	99.9	99.9	99.9	99.8	99.9	99.9	66.2	59.3	94.1
transformer, 4 layers (d128) + last-frame 3×3	926k	100	100	100	99.9	99.9	99.9	99.9	100	99.9	98.8	100	92.8
d 2D-conv transformer, 2 layers	142k	100	100	100	95.8	91.1	80.5	96.1	90.9	79.5	84.3	81.7	73.8
e KV-shift transformer, 1 memory read	2.6k	100	100	100	100	100	100	100	100	100	16.1	51.8	58.7
e KV-shift transformer, 4 memory reads*	9.8k	100	100	100	100	100	100	100	100	100	98.8	99.8	99.8
f 3D-conv transformer, 1 memory read	3.8k	100	100	100	100	100	100	100	100	100	71.3	99.8	43.3
f 3D-conv transformer, 4 memory reads	11k	100	100	100	100	100	100	100	100	100	100	99.8	98.7
b 3D-conv CNN	18k	100	100	100	100	100	99.4	99.8	100	99.5	100	100	96.6
Temporal stability (Sec. 5)													
per-frame noise + backward encoder + anchor	30k	31.8	32.0	32.2	6.2	6.2	6.1	4.5	4.7	4.3	0.0	0.0	0.3
h′ same denoiser, frame-ordered sampling	30k	100	100	100	25.1	33.4	34.1	21.0	30.1	32.8	36.8	9.5	28.2
h same denoiser, order-matched training	30k	100	100	100	100	99.1	98.8	100	99.2	98.9	100	84.1	95.4

Figure 10: **Main results, per seed.** The rows and columns of Figure 4; each cell gives SeqAcc (%) for seeds 42, 43 and 44 at the final step, truncated to one decimal, and is shaded by their mean. Red: standard-transformer runs that never left the marginal predictor. The 200k-step runs of the standard models are in Tables 4 and 7.

the four-read rows are detailed in Table 14 and the sampler comparison of the structured denoiser in Appendix F.7. Table 3 gives the per-seed values of Figure 3, and Table 4 the controls on training budget and recipe.

LifeGPT configuration. The LifeGPT row of Figure 3 (Berkovich and Buehler, 2025) is the 4×128 configuration (1,052,800 parameters), designated in advance and within the 0.5–1.5M budget for prior-work configurations. It keeps raster next-token generation, the original split-half partial RoPE, and forgetful causal masking during training only. It uses our binary vocabulary, the common training protocol of 40k steps at batch 512, and greedy evaluation of the final EMA weights. This is architecture-faithful retraining under our protocol. Across three seeds its L3 SeqAcc is $64.26 \pm 26.68\%$ (sample SD).

Training budgets and recipe choices. Table 4 lists the controls on training budget, learning rate and precision. At five times the steps, the standard CNN ends at 0.2070/0.2065/0.2051: its floor is converged, not budget-starved.

Both the 3D-conv CNN and the structured diffusion model were first registered at learning rate 10^{-3}; the reported 5×10^{-4} replaced it after one seed of each ended low, and both recipes are listed with all their seeds. For the CNN, the 0.8350 cell is a late loss spike, which the lower rate removes. For the diffusion model, the registered rate leaves one of five seeds stuck at 0.9175; neither continued annealing nor a warm restart from that checkpoint frees it, and only retraining from initialization at 5×10^{-4} lifts it to 0.9805. The remaining gap on that seed survives batch 4096 and changes of width, consistent with the rare-situation tail of Appendix F. In fp32 and in bf16 alike, both models stay inexact on one or two seeds; the choice of precision does not remove the tail.

Table 2: **Main results by number.** Strict SeqAcc (fraction) at the final step on EMA weights. Three numbers are seeds 42/43/44 and a single number is the single run of an analytic model; for the d64, d192 and d256 standard transformers the entry is the min–max range over seeds 42/43/44 (per-seed L3 values in Table 7, all per-seed values in Figure 10). Training uses the filtered stream, bf16 and the reported recipes of Appendix C; L4 is scored on the 768 forced test trajectories after training with the 10% dose (Appendix B). Exceptions: the L4 cells of the 3D-conv CNN, the 3D-conv transformer and the structured diffusion model are fp32 runs, the 3D-conv CNN's at learning rate 10^{-3}; the 3D-conv transformer's L4 cells are the model without the tally (with the tally: 0.9974 / 0.9987 / 0.9180, Table 13); L3 values of the 2D-conv, KV-shift and 3D-conv transformers trained on the unfiltered stream in fp32 are in Appendix F. "—" marks cells not registered.

Model	Params	L1	L3	L4
3D-conv CNN (analytic)	15,361	1.0	1.0000	—
3D-conv CNN (learned)	17,994	1.0 / 1.0 / 1.0	0.9980 / 1.0000 / 0.9951	1.0000 / 1.0000 / 0.9661
3D-conv transformer (analytic)	913	1.0	1.0000	—
3D-conv transformer (learned)	3,798	1.0 / 1.0 / 1.0	1.0 / 1.0 / 1.0	0.7135 / 0.9987 / 0.4336
Structured diffusion model	29,963	1.0 / 1.0 / 1.0	1.0000 / 0.9927 / 0.9893	1.0000 / 0.8411 / 0.9544
2D-conv transformer	141,698	1.0 / 1.0 / 1.0	0.9614 / 0.9097 / 0.7954	0.8438 / 0.8177 / 0.7383
KV-shift transformer	2,625	1.0 / 1.0 / 1.0	1.0000 / 1.0000 / 1.0000	0.1615 / 0.5182 / 0.5872
Standard transformer d128	924,801	1.0 / 1.0 / 1.0	0.9263 / 0.7646 / 0.8784	0.8229 / 0.6536 / 0.8841
Standard transformer d64 + last-frame 3×3	166,529	1.0 / 1.0 / 1.0	0.9985 / 0.9995 / 0.9995	0.6628 / 0.5938 / 0.9414
Standard transformer d128 + last-frame 3×3	926,209	1.0 / 1.0 / 1.0	0.9990 / 1.0000 / 0.9995	0.9883 / 1.0000 / 0.9284
Standard transformer d64	165,825	1.0 / 1.0 / 1.0	0.1787–0.3120	0.0339–0.0677
Standard transformer d192	2,866,753	1.0 / 1.0 / 1.0	0.0410–0.5894	0.0026–0.6771
Standard transformer d256	6,581,505	1.0 / 1.0 / 1.0	0.0410–0.8477	0.0026 ×3
Standard CNN	116,097	1.0 / 1.0 / 1.0	0.1909 / 0.1880 / 0.1904	0.1133 / 0.0781 / 0.1094
Standard diffusion model	85,697	0.1914 / 0.2812 / 0.2754	0.0000 / 0.0000 / 0.0000	0.0000 / 0.0000 / 0.0000

Table 3: **Per-seed values of Figure 3.** L3, fractions to four decimals (a value below 1 is never shown as 1.0000), seeds 42/43/44. Prior-work baselines: NCA (Mordvintsev et al., 2020), the one-dimensional transformer of Burtsev (2024), AutomataGPT (Berkovich et al., 2026), LifeGPT (Berkovich and Buehler, 2025), and Diff-NCA (Kalkhof et al., 2024). Pixel-level accuracy is teacher-forced (one step ahead) except in the two diffusion rows marked †, which score the rollout; SeqAcc always scores the full rollout.

Model	Pixel-level accuracy	SeqAcc
NCA (growing)	0.7715 / 0.7711 / 0.7718	0.1387 / 0.1021 / 0.1377
Standard CNN	0.9625 / 0.9645 / 0.9636	0.1909 / 0.1880 / 0.1904
3D-conv CNN	0.9999 / 1.0000 / 0.9999	0.9980 / 1.0000 / 0.9951
Burtsev 1-D transformer	0.8243 / 0.9815 / 0.9768	0.0386 / 0.0405 / 0.0405
AutomataGPT	0.9303 / 0.9416 / 0.9351	0.1528 / 0.1689 / 0.1655
LifeGPT (4 × 128)	0.9999 / 0.9982 / 0.9980	0.9497 / 0.5098 / 0.4683
KV-shift transformer	1.0000 / 1.0000 / 1.0000	1.0000 / 1.0000 / 1.0000
Diff-NCA	0.7932 / 0.7888 / 0.7913	0.1470 / 0.1489 / 0.1484
Standard diffusion model†	0.6167 / 0.6218 / 0.6211	0.0000 / 0.0000 / 0.0000
Structured diffusion model†	1.0000 / 0.9992 / 0.9984	1.0000 / 0.9927 / 0.9893

E Spatial locality in the raster transformer

The standard transformer of Section 2 reads a world's sixteen frames as one frame-major raster sequence of 1024 tokens with learned absolute positions, trains with a causal next-token loss under teacher forcing, and rolls out greedily; depth grows with width, from d64 with 2 layers to d256 with 8 (Appendix C). Runs use 40k steps and seeds 42/43/44 unless stated.

E.1 Rotary positions on the Game of Life

Table 5 gives the d64 transformer on the Game of Life (L1) under six position encodings, adapting one-dimensional RoPE (Su et al., 2024) and two-dimensional visual RoPE (Heo et al., 2024) to the toroidal grid. With rotary positions along the raster sequence (1D RoPE) it completes 39.1% of rollouts, and 95.4% of its teacher-forced errors fall on the 28 border cells (43.75% of the board). In raster order a cell's left neighbour in the previous frame is 65 tokens back, but in column 0 it wraps

Table 4: **Training budgets and recipe choices.** Strict L3 SeqAcc (fraction) at the final step on EMA weights, one value per listed seed. The second column lists every difference from the model's row in Table 2; indented rows modify the nearest unindented row above them.

Model	Difference from Table 2	Seeds	L3
Standard CNN	200k steps ($5\times$), fp32	42/43/44	0.2070 / 0.2065 / 0.2051
3D-conv CNN	learning rate 10^{-3}, fp32	42/43/44	0.9937 / 0.8350 / 0.9946
3D-conv CNN	fp32	42/43/44	0.9917 / 1.0000 / 0.9814
Structured diffusion model	learning rate 10^{-3}, fp32	42/43/44/45/46	0.9990 / 1.0 / 0.9175 / 0.9941 / 1.0
from the stuck checkpoint	continued annealing	44	0.9165
from the stuck checkpoint	warm restart	44	0.9229
Structured diffusion model	fp32	42/43/44	1.0000 / 1.0000 / 0.9805
same recipe	batch 4096	44	0.9873

Table 5: **Rotary positions on the Game of Life (L1).** Standard transformer d64, 2 layers; SeqAcc (%) at seeds 42/43/44 and their mean. The five rotary variants have identical parameter counts (100,289) and initialization; the learned reference has 165,825 parameters. Border error share: fraction of teacher-forced cell errors, pooled over seeds, that fall on the 28 border cells (43.75% of the board).

Position encoding	SeqAcc (%)	Mean	Border error share
1D raster RoPE	39.26 / 39.65 / 38.67	39.1	95.4%
2D axial RoPE	100 / 100 / 100	100	no errors
Toroidal RoPE (period 8)	100 / 100 / 99.80	99.9	0% (1 error)
Period-16 control	100 / 100 / 100	100	no errors
Half-integer detuned control	100 / 98.44 / 100	99.4	75% (8 errors)
Learned absolute (reference)	100 / 100 / 100	100	no errors

to the end of the row and is 57 tokens back. Both offsets always land on some token, so an encoding that sees only raster offsets cannot tell which applies to which cell; absolute positions, or rotation by row and column separately (2D axial RoPE), can. This concerns what the encodings can express, not a mechanism located in the trained models.

Every two-dimensional variant repairs the failure (99.4–100%), including the period-16 and half-integer detuned controls, whose periods are deliberately wrong. The result supports two-dimensional position, not the period, as what matters; the preregistered prediction that the board's own period would beat the other two-dimensional variants was not met. All arms keep the raster causal mask, and time is never periodic. The learned reference has more parameters than the rotary arms, so it is a reference, not a matched control. The rule is known, so rule inference is not tested.

E.2 Rotary positions on unseen rules

Table 6 crosses the two rotary schemes with two kinds of token on unseen rules (L3), at d64 and d128. A pair token also carries the cell's previous-frame 3×3 neighbourhood (the binding of Section 4.2), so pair arms have slightly more parameters. With cell tokens, 2D RoPE lifts d128 from 82.6% to 99.3%; at d64 neither scheme exceeds 18%.

Four contrasts were preregistered; each is met only if two of three seeds clear its per-seed threshold. (i) 2D minus 1D RoPE, pair tokens: $+13.9$ pp at d64, -0.9 pp at d128; not met. (ii) Pair minus cell tokens, 2D RoPE: $+28.4$ pp at d64, where only one seed clears, and -0.3 pp at d128; not met. (iii) 2D minus 1D RoPE, cell tokens, registered as descriptive: -1.3 pp at d64, $+16.7$ pp at d128, higher on every d128 seed. (iv) Border share of teacher-forced errors, pair tokens with 1D RoPE: met at d64 (mean 98.95%); undefined at d128, where two seeds make no teacher-forced errors. The experiment thus does not establish two-dimensional rotary positions as a robust improvement on unseen rules; the d128 cell-token gain stays descriptive. At d64 the rotary pair arms reach only 30.4% and 44.3%, against 99.9% with learned positions (Table 10).

Table 6: **Rotary positions and previous-frame neighbourhoods on unseen rules (L3).** Standard transformer, L3 SeqAcc (%) at seeds 42/43/44 and mean. A cell token carries only its own cell; a pair token adds the cell's 3×3 neighbourhood in the previous frame. With pair tokens and 1D RoPE at d64, 98.2–99.5% of teacher-forced errors fall on the 28 border cells.

Width	Tokens	Positions	Seeds 42 / 43 / 44	Mean
d64	cell	1D RoPE	16.60 / 17.48 / 17.53	17.2
d64	cell	2D RoPE	16.31 / 15.58 / 15.77	15.8
d64	pair	1D RoPE	39.84 / 23.44 / 28.03	30.4
d64	pair	2D RoPE	34.77 / 31.01 / 67.19	44.3
d128	cell	1D RoPE	56.64 / 94.04 / 97.12	82.6
d128	cell	2D RoPE	99.41 / 99.76 / 98.83	99.3
d128	pair	1D RoPE	100 / 100 / 99.95	99.9
d128	pair	2D RoPE	97.31 / 100 / 99.80	99.0

Table 7: **Width ladder of the standard transformer.** SeqAcc (fraction of rollouts exact) on L2 validation and L3. Top: 40k steps, seeds 42 / 43 / 44. Middle: separate 200k-step runs. Bottom: depth by width at seed 42, L2 validation only.

Arm	Steps	L2 validation	L3
d64, 2 layers	40k	0.2100 / 0.3164 / 0.3374	0.1787 / 0.2861 / 0.3120
d128, 4 layers	40k	0.9160 / 0.7900 / 0.8960	0.9263 / 0.7646 / 0.8784
d192, 6 layers	40k	0.5098 / 0.0547 / 0.6216	0.4902 / 0.0410 / 0.5894
d256, 8 layers	40k	0.0547 / 0.8628 / 0.7061	0.0410 / 0.8477 / 0.6929
d192, 6 layers, seed 42	200k	0.8931	0.8828
d256, 8 layers, seed 42	200k	0.0186	0.0151
d256, 8 layers, seed 43	200k	0.0273	0.0234
d256, 8 layers, seed 44	200k	0.6538	0.6372
d64, 2 layers, seed 42	40k	0.2100	—
d64, 6 layers, seed 42	40k	0.6616	—
d192, 2 layers, seed 42	40k	0.4614	—
d192, 6 layers, seed 42	40k	0.5098	—

E.3 Width ladder

Table 7 gives the standard transformer with learned positions at four widths (filtered stream, learning rate 10^{-3}). A run either leaves the marginal predictor, which ignores the evidence and scores 0.0410 on L3, before 18k steps or never (Figure 11); no run escapes between 40k and 200k steps. At 40k steps d192 and d256 each leave one seed of three at the floor. In separate 200k-step runs (middle block) d192 seed 42 improves (L3 0.8828 against 0.4902 at 40k), while d256 seeds 42 and 43 stay stuck, although seed 43 escaped in its 40k run, so the seed alone does not decide escape. Halving the learning rate to 5×10^{-4} leaves d256 seed 42 at the floor too (L3 0.0405 against 0.0410; not tabulated).

Accuracy does not rise monotonically with width: d128 is the best rung on average (85.6% on L3), and no arm is exact. L2 validation (seen rules) tracks L3 in every run, so escaped runs infer the rule rather than recall it. The bottom block, a single-seed depth-by-width check, is descriptive: six layers instead of two raise d64 from 0.2100 to 0.6616; at d192, two layers give 0.4614 against 0.5098.

F Diagnostics for temporal locality, Level 4, and temporal stability

All measurements below use the final-step EMA weights of registered runs, evaluated in fp32 on the self-consistent test corpora of Section 2. Attention and order probes change no weights; the position-free arms and the matched-window experiment are fresh training. Protocols and primary probes were frozen before launch; later diagnostics are labelled post hoc.

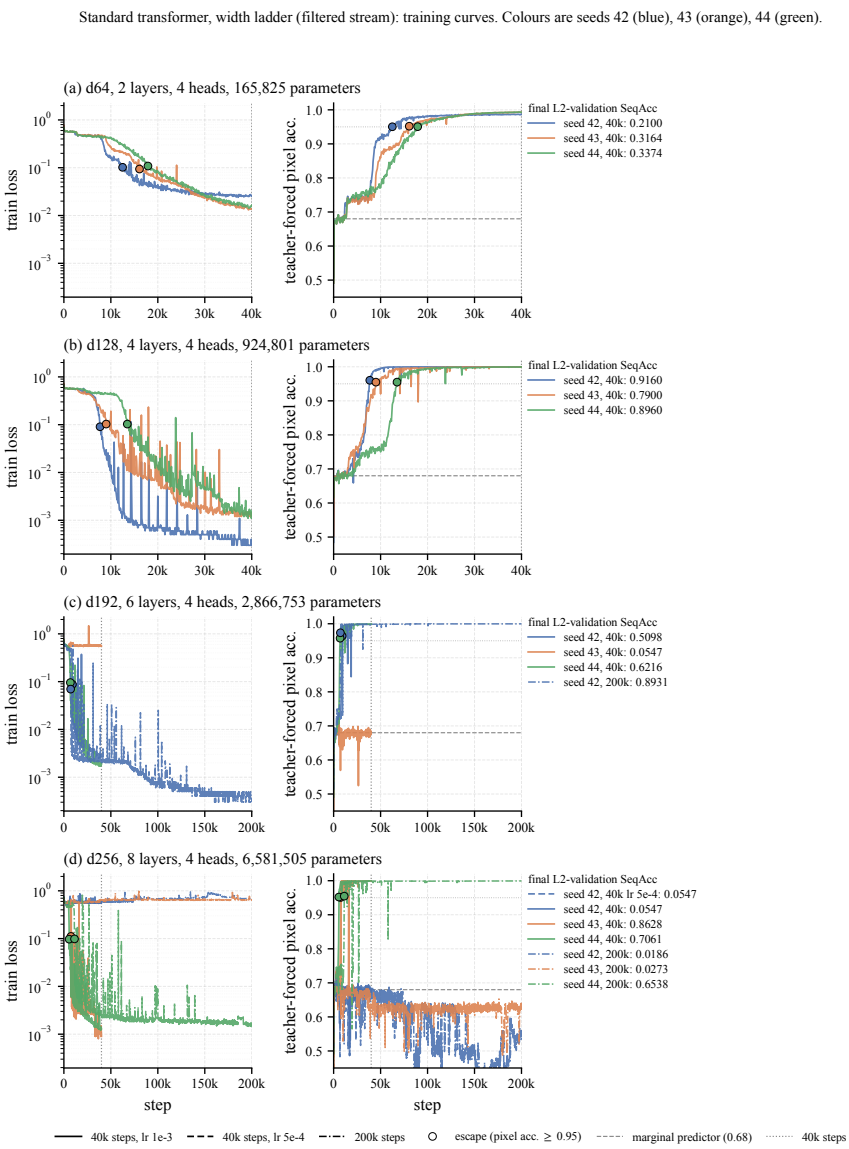


Figure 11: **Training curves of the width ladder.** Training loss and teacher-forced pixel accuracy, one row per width; colours are seeds 42/43/44; solid lines are 40k-step runs, dash-dotted lines 200k-step runs, and the dashed line the halved learning rate. Circles mark escape, the first step with teacher-forced pixel accuracy ≥ 0.95; stuck runs are flat from $\sim$1k steps.

F.1 Convolution: the rare-situation tail

Every learned 3D-conv CNN run below 1.0 (Table 2 and Appendix D) concentrates its first errors on one of the two rarest situations (each about 0.2% of cells), for which no detector forms. Which run misses a detector depends on the seed: more budget, width, or batch does not repair it, and the learning rate helps only weakly.

F.2 Which pairing the 2D-conv transformer learns

We capture attention per block and head at the teacher-forced first prediction step on all 2048 L3 worlds (Table 8). Unmasked attention lets the first hop point either way; on every seed it points back, to the same cell in the previous frame, as if each pair were stored at its outcome (look-back pairing). The final match is content-correct under that reading and near chance under the look-ahead one.

The pattern does not identify a unique mechanism. What stays soft is retrieval: errors concentrate where the queried situation is rare in the observed frames, and wrong cells spread their attention over situations one live neighbour away. The evidence-count rows were added after the first pass and are descriptive.

Table 8: **Attention readout of the 2D-conv transformer.** Filtered training stream; teacher-forced first prediction step on all 2048 L3 worlds; seeds 42/43/44, whose L3 SeqAcc is 0.9614/0.9097/0.7954. Same-cell entries give the largest same-cell attention mass over heads; the uniform level is 0.002.

Measure	s42	s43	s44
Learned temperature (3D-conv transformer: frozen at 20)	0.846	0.750	0.662
Same-cell attention to the previous frame (look-back)	0.472	0.445	0.388
Same-cell attention to the next frame (look-ahead; uniform 0.002)	0.003	0.002	0.003
Final match content-correct, look-back reading	0.9995	0.9986	0.9955
Final match content-correct, look-ahead reading	0.1836	0.1734	0.2109
Top-1 attention mass	0.138	0.144	0.176
First-step cell accuracy	0.99979	0.9995	0.99832
Error rate, situation seen once among the observed transitions (276 cells)	0.069	0.109	0.228
Error rate, seen twice (400 cells)	0.020	0.058	0.150
Error rate, seen 3–4 times (983 cells)	0	0.011	0.052
Error rate, seen 17 or more times (119,722 cells)	0	0	0
Mean exact matches among the observed transitions, wrong / correct cells	1.3 / 61.0	1.8 / 61.0	3.1 / 61.1
Attention to one-neighbour-off situations, wrong ÷ correct	24.5	37.7	15.7

Soft retrieval resists the obvious fixes. On the unfiltered stream the seed-42 model completes 0.8047 of L3 rollouts; freezing the temperature at 20 (0.793, and 0.758 after five times the budget) and doubling the width with $3.4\times$ the parameters (0.778) do not repair it. Each is a single run.

F.3 Positions and pairing across designs

Table 9 sets the 2D-conv transformer beside the two constructors that are given the pair. Removing its token and query position tables (recipe otherwise identical) makes its retrieval a function of the set of observed tokens, so the first hop cannot find the previous frame. It then ends below the standard CNN (L3 0.1553/0.1558/0.1538; L2 0.1851/0.1841/0.1851), and the readout of Table 8 shows why: attention to the previous and to the next frame both stay at the uniform level (0.002 on every seed), top-1 mass is 0.02, and first-step cell accuracy is 0.73.

Table 9: **Pairing across designs.** L3 SeqAcc, filtered stream, seeds 42/43/44. Architectures, temperatures, and parameter counts differ; index alignment is not a learned positional embedding.

Model	Params	Pairing and position structure	L3 SeqAcc
KV-shift transformer	2,625	index-aligned pairs; no learned positions	1.0 / 1.0 / 1.0
3D-conv transformer	3,798	adjacent-frame pairs; no learned positions	1.0 / 1.0 / 1.0
2D-conv transformer	141,698	unpaired input; learned positions	0.9614 / 0.9097 / 0.7954
2D-conv transformer	104,834	unpaired input; no learned positions	0.1553 / 0.1558 / 0.1538

Constructors given the pair need no learned positions, but pairing does not make positions dispensable in every design (Table 10). Without positions, the standard d128 transformer (793,729 parameters) has only its causal mask as an order signal. Causal decoders can recover position from the mask (Haviv et al., 2022; Kazemnejad et al., 2023), but within 40k steps none of the three seeds leaves the marginal predictor (L3 0.0410 and L2 0.0547 on every seed; teacher-forced pixel accuracy 0.66–0.67), and the previous-frame neighbourhood on each token does not rescue either width.

F.4 Matched temporal windows

One memory-attention model isolates the temporal window (Figure 5c). It keeps the 2D-conv transformer's backbone ($d = 64$, two unmasked four-head memory-attention blocks, a learned temperature initialized to 1) and builds its memory with a $3\times3\times3$ convolution whose input gates expose one

Table 10: **Previous-frame neighbourhood and learned positions in the standard transformer.** L3 SeqAcc (%), mean over seeds 42/43/44. Removing the position table also changes parameter count and initialization; the causal mask and the neighbourhood input still carry temporal and spatial structure. All cells share the recipe and were scored in one evaluation pass: the learned-position d64 cells and the d128 own-cell cell are the checkpoints of Table 2 re-scored here (d128 own cell 0.9282/0.7681/0.8906, against 0.9263/0.7646/0.8784 in Table 2 and 85.6% in the main text), and the d128 cell with the previous-frame neighbourhood and learned positions comes from three reference runs of the same recipe (1.0/1.0/0.9990).

Width	Token input	Learned positions	No position encoding
d64	own cell	25.90	4.10
d64	+ previous-frame 3×3	99.92	8.12
d128	own cell	86.23	4.10
d128	+ previous-frame 3×3	99.97	8.27

of four windows at each of the eight observed anchors. Unavailable temporal neighbours are zero, never wrapped. Every arm allocates 144,050 parameters with identical same-seed initialization, but unused slices and gated position tables leave different weights in use: equal allocation is not equal effective capacity. All 24 runs (four windows, two position settings, seeds 42/43/44) train on the filtered stream for 40k updates and are scored on all 2048 unseen worlds; within-seed comparisons share hardware.

The preregistered performance predictions are met (Table 11): the forward window $(t, t{+}1)$ without positions exceeds 95% with an on/off gap below two points, and it improves on the single frame (t) by 12.061 points with learned positions and 84.310 without. The direction prediction is inconclusive: forward minus backward is 0.033 and 0.163 points, inside the preregistered two-point region.

Table 11: **Matched temporal windows with and without learned positions.** L3 SeqAcc (%), seeds 42/43/44. The first four rows form the main 2×2 design; the rest compare direction and a symmetric window. All arms allocate 144,050 parameters with identical same-seed initialization; the weights in use differ.

Window	Learned positions	Mean ± SD	Seeds 42 / 43 / 44
(t)	on	87.939 ± 2.511	85.986 / 90.771 / 87.061
(t)	off	15.674 ± 0.000	15.674 / 15.674 / 15.674
$(t, t{+}1)$	on	100.000 ± 0.000	100.000 / 100.000 / 100.000
$(t, t{+}1)$	off	99.984 ± 0.028	99.951 / 100.000 / 100.000
$(t{-}1, t)$	on	99.967 ± 0.056	100.000 / 100.000 / 99.902
$(t{-}1, t)$	off	99.821 ± 0.310	100.000 / 100.000 / 99.463
$(t{-}1, t, t{+}1)$	on	99.870 ± 0.226	99.609 / 100.000 / 100.000
$(t{-}1, t, t{+}1)$	off	99.967 ± 0.056	99.902 / 100.000 / 100.000

Table 12 asks where the most-attended memory token sits relative to the matching historical situation A: at A itself, whose successor supplies the outcome (cause-side, the look-ahead convention), or one frame later, at A's outcome (effect-side, the look-back convention). The six symmetric-window models (two position settings, three seeds each) all favour the cause side. A post-hoc partition into cause-only, effect-only, both, and neither keeps this reading: in every symmetric arm the effect-only fraction stays below 0.9% (maximum 0.86594%). The single frame with learned positions shows the opposite, predominantly effect-side pattern (0.974/0.995/0.913), like the look-back pairing of Table 8.

These are anchor associations, not an identified key/value mechanism, and the convolution window is not a literal KV shift. A final token has passed through two unmasked blocks, which can relocate evidence: the forward window with learned positions completes every rollout at seed 42 with a cause-side match of only 0.323 (0.999 at seeds 43 and 44), and the backward window at seed 43 is exact with an effect-side match of 0.077 and 50.6% of final attention on boundary anchors. High match rates also need not mean large attention mass.

Table 12: **Which historical anchor matches the query?** Fraction of first-step queries whose most-attended interior memory token sits at the matching situation (cause-side) or one frame after it, at its outcome (effect-side). All 2048 L3 worlds, seeds 42/43/44, common interior anchors 1–6. The two readings overlap and are observational: two memory-attention blocks can relocate evidence before the final readout.

Window	Learned positions	Cause-side top-1	Effect-side top-1
(t)	on	0.256 / 0.249 / 0.239	0.974 / 0.995 / 0.913
(t)	off	0.090 / 0.136 / 0.110	0.091 / 0.125 / 0.086
$(t, t{+}1)$	on	0.323 / 0.999 / 0.999	0.163 / 0.242 / 0.229
$(t, t{+}1)$	off	0.563 / 0.999 / 0.999	0.101 / 0.107 / 0.106
$(t{-}1, t)$	on	0.242 / 0.137 / 0.251	0.999 / 0.077 / 0.970
$(t{-}1, t)$	off	0.211 / 0.234 / 0.158	0.999 / 0.659 / 0.725
$(t{-}1, t, t{+}1)$	on	0.931 / 0.817 / 0.999	0.211 / 0.203 / 0.207
$(t{-}1, t, t{+}1)$	off	0.806 / 0.904 / 0.999	0.064 / 0.096 / 0.159

Table 13: **Level 4 grids: evidence withheld.** SeqAcc on 2048 forced arbitrations (L4A) and 768 forced pair inferences (L4B); seeds 42/43/44. *Natural rate*: the event occurs in training at its natural frequency; *10% dose*: 10% of every training batch is replaced by forced training-half trajectories. [a]Recall-only L4A is budget-limited: 0.6152 at 40k steps, 1.0000 at 120k or with head width 48. [b]Separate d64 runs without a dose arm, so the last column is blank; on the unforced L4B corpus, where the event occurs at its natural rate, they read 0.2026/0.2031/0.1953.

Model	L4A	L4B, natural rate	L4B, 10% dose
3D-conv transformer, recall only	$0.6152 \rightarrow 1.0000$[a]	$0.5807 \approx$ chance	0.7135 / 0.9987 / 0.4336
3D-conv transformer + tally	1.0000 / 0.9946 / 1.0000	0.9922 / 0.5807 / 0.9167	0.9974 / 0.9987 / 0.9180
3D-conv CNN	1.0000 / 1.0000 / 0.9995	0.9805 / 0.8372 / 0.8555	1.0000 / 1.0000 / 0.9661
Structured diffusion model	1.0000 / 0.5190 / 1.0000	0.9935 / 0.8516 / 0.9987	1.0000 / 0.8411 / 0.9544
Standard transformer d64[b]	0.0044 / 0.0249 / 0.1802	0.0378 / 0.0299 / 0.0299	—

F.5 Binding across designs and training streams

The pairing requirement carries across designs; a particular constructor need not. Replacing the $2{\times}3{\times}3$ convolution of the structured diffusion model by the index-aligned KV shift, with the 200k-step recipe otherwise unchanged, gives L3 SeqAcc 0.5962/0.2490/0.6250 (mean 49.0%) against 1.0000/1.0000/0.9316 (97.7%) for the convolution retrained on the same machine. The preregistered prediction that the two constructors are interchangeable was not met. Parameter counts (29,206 against 29,963) and initialization trajectories differ, so this is a recipe-specific comparison of constructors, not an isolated cause.

The training stream separates the attention constructors too. On the unfiltered stream about a third of training trajectories query a situation the observed frames never show, whose label is a fair coin. L3 SeqAcc at seeds 42/43/44, filtered stream (bf16 training) against unfiltered stream (fp32 training): 2D-conv transformer 0.9614/0.9097/0.7954 against 0.8047/0.8853/0.8926; KV-shift transformer 1.0000/1.0000/1.0000 against 0.8442/1.0000/0.9297; 3D-conv transformer 1.0000 on every seed of both. The index-shift constructor alone is exact on one stream and not the other; since precision also differs, the comparison is descriptive.

F.6 Level 4: evidence withheld

Table 13 gives the full grids behind Section 4.3. L4B is the level called L4 in the main text, with the centre-flip prior of Section 2 (Appendix B). L4A fixes four rule rows across the whole family and queries one of them without showing it, so the answer is the family's fixed value. "Recall only" is the 3D-conv transformer of the main text; "+ tally" adds the 3D-conv CNN's tally to its head.

Training time does not substitute for the dose. At the natural rate the tally model trains three times as long as with the dose (120k against 40k steps) and still ends below its dose runs on every seed; single seed-42 natural-rate runs of the 3D-conv CNN and the structured diffusion model at 600k steps, three times their 200k budget, also stay below their dose cells. The 3D-conv CNN's dose cells

Table 14: **Attention depth at L4.** Strict L4B SeqAcc (%) on 768 held-out worlds that each need the partner inference; three-seed mean (worst seed). R sequential attention reads over the frozen memory of observed transitions, each with its own key and value projections and a residual update of the query; same 40k-step recipe and 10% dose throughout. Depth was chosen on a training-half validation corpus before this test was scored: four reads for both models. *Evidence-balanced*: a training loss that averages cells with direct evidence and the 0.18% of training cells that need the partner separately (a training-side label, never a model input). The R=1 rows are the single-read models of Table 2. Parameters: 3D-conv 3.8k/6.2k/8.6k/11.0k/15.8k and KV-shift 2.6k/5.0k/7.4k/9.8k/14.6k for R=1/2/3/4/6.

Model	Loss	R=1	R=2	R=3	R=4	R=6
3D-conv transformer	ordinary	71.5 (43.3)	96.6 (92.7)	98.2 (95.0)	99.5 (98.6)	99.6 (98.8)
3D-conv transformer	evidence-balanced	—	93.5 (83.8)	99.8 (99.7)	99.9 (99.8)	96.9 (90.8)
KV-shift transformer	ordinary	42.2 (16.1)	79.6 (55.8)	79.7 (49.8)	94.1 (84.7)	92.5 (77.6)
KV-shift transformer	evidence-balanced	—	99.2 (97.9)	98.6 (96.0)	99.5 (98.8)	100 (100)

are higher than its natural-rate cells on every seed, while the structured diffusion model, trained for 200k steps either way, scores about the same with and without the dose.

Table 14 adds attention depth. Extra reads close the L4 gap for the 3D-conv transformer under the ordinary loss, and for the KV-shift transformer only with the evidence-balanced loss. Depth and parameter count grow together, so the ladder does not separate them.

F.7 Structured diffusion: sampler swap and matched training

The two structured denoisers of Section 5 (main and curriculum recipes) are trained for 1M steps with independent per-frame noise levels, a backward-looking canvas encoder, and a clean anchor frame (Appendix Figure 9); under joint sampling they stayed at 0.046–0.049 throughout training. Table 15 reads their final-step weights under four samplers.

Table 15: **Same weights, four samplers.** L3 SeqAcc of the final-step weights of the two structured denoisers trained with independent per-frame noise levels (seed 42 each). Causal freezing denoises each frame in turn and keeps it as clean context; its call budget is split over the eight frames.

Sampler	Network calls	Main	Curriculum
Joint denoising	49	0.0483	0.0469
Causal freezing	48	0.7402	0.7866
Causal freezing, one call per frame	8	0.8750	0.8892
Causal freezing, full chain per frame	49 per frame	0.7622	0.8081

The weights already encode the transition: given the true frames before it, a single predicted frame is extended with pixel accuracy 0.9986/0.9991, and 0.93–0.96 of frames are perfect, with no decline along the rollout. If frames failed independently, rollouts would complete at the product of these rates, 0.63/0.71; causal freezing does better, so the errors cluster within worlds.

Matched training trains on the inputs that causal freezing produces: clean ground truth before the frame being denoised, that frame at its noise level, noise beyond it, and loss on that frame only. The resulting structured diffusion model (29,963 parameters) reaches L3 1.0000/1.0000/0.9805 in fp32 and 1.0000/0.9927/0.9893 in the bf16 re-run of Table 2, and 1.0 at L1 on all three seeds.

F.8 Commitment order

At each of eight stages, one denoiser call scores every uncommitted frame by its mean confidence $|\tanh(u/2)|$; the most confident frame then runs its full chain and is committed. This is the confidence ordering of masked generative models (Chang et al., 2022; Kim et al., 2025), applied to frames without a prescribed order (Table 16). The per-frame-level weights, never trained for an order, choose exactly the causal order in 98.1%/98.7% of worlds and match the imposed order's accuracy; the commonest exception (10 worlds per checkpoint) commits predicted frames 2, 4, 6, 8

before 1, 3, 5, 7. These checkpoints already have a backward-looking canvas encoder, paired evidence, and a clean last-observed anchor, so this is order selection under those priors, not discovery of causality. The imposed-order column is a separate sampling pass of the full-chain sampler; its gap to Table 15 is sampling noise.

Table 16: **Confidence-ordered commitment.** L3 SeqAcc; "Causal" is the share of worlds whose chosen order is exactly $1\to8$. Per-frame-level rows: the two denoisers of Table 15; matched-training rows: the structured diffusion model at seeds 42/43/44.

Weights	Most confident first	Causal	Imposed $1\to8$	Least confident first	Random
Per-frame levels, main	0.7739	0.981	0.7749	0.0366	0.0386
Per-frame levels, curriculum	0.8169	0.987	0.8169	0.0386	0.0400
Matched training, s42	1.0000	1.000	1.0000	0.0000	0.0000
Matched training, s43	0.9907	0.999	0.9917	0.0000	0.0000
Matched training, s44	0.9868	0.995	0.9888	0.0000	0.0000

G PHYSICAL SYSTEMS

Two kinds of world beyond the automaton, each with an exact simulator: billiards, the discrete comparison behind Figure 6 and Section 5 (Appendix G.1), and three exploratory experiments on continuous dynamics (Appendix G.2).

G.1 BILLIARDS AND THE GAME OF LIFE

Setup. Billiards live on a 16×16 torus: three point balls, each with a constant velocity drawn from the eight king moves. A frame is the occupancy of the balls; a rollout has 8 observed and 8 predicted frames, as in the automaton protocol. In colliding billiards, balls that land on the same cell annihilate. The denoiser is the standard denoiser deepened from four to eight 3D-convolutional layers (196,417 parameters), trained with an independent noise level per frame: 200k steps at batch 64 on billiards, and 40k steps at batch 512 on the Game of Life, the training budget of the standard diffusion model; one network per world and seed. Joint denoising shares each of the 49 chain levels across all predicted frames. Causal freezing runs frame k's chain, commits and freezes it, and holds frames not yet started at the top noise level. Both samplers read the same weights in the same evaluation pass; Life worlds come from the 8×8 L1 corpus. Figure 6b scores non-static Life worlds only, whose eight future frames do not all repeat the last observed one. Table 17 gives per-seed values at the full chain.

Table 17: **Same plain denoiser, two samplers, full chains.** SeqAcc (fraction of rollouts with all eight predicted frames exact) for seeds 42/43/44, 512 worlds per world type. Joint denoising spends 49 network calls per rollout; causal freezing runs the full 49-level chain for every frame. Depth 8 unless stated.

World	Joint denoising	Causal freezing
Free billiards	0.9980 / 1.0000 / 1.0000	1.0 / 1.0 / 1.0
Colliding billiards, all worlds	0.9785 / 0.9961 / 0.9785	1.0 / 1.0 / 1.0
first collision in the predicted frames (26 worlds)	0.7308 / 0.9231 / 0.8077	1.0 / 1.0 / 1.0
Game of Life, depth 8	0.4258 / 0.4238 / 0.4277	1.0000 / 1.0000 / 0.9980
Game of Life, depth 4	0.2754 / 0.3203 / 0.3008	1.0000 / 1.0000 / 0.9980

Where the joint sampler fails. On the Game of Life, the joint frame-perfect rate by predicted frame (full chain, seed mean) is 1.000/1.000/0.950/0.386/0.335/0.324/0.314/0.314 at depth 4 and 1.000/1.000/1.000/0.987/0.541/0.448/0.441/0.443 at depth 8; causal freezing stays at or above 0.999 at every frame. Doubling the depth moves the first frame with a rate below 0.90 from frame 4 to frame 5 on every seed, but the sharp drop remains. Trained with one noise level shared by all frames, as the standard diffusion model is, the depth-8 network gives 0.4102/0.3867/0.3828 under joint denoising. On colliding billiards, the joint failures sit in the last one or two predicted frames (frame-perfect 0.999 at frame 7, 0.985 at frame 8), where a single network pass reaches back to only one clean observed frame.

Compute-matched sampling. With the same total number of denoiser calls $B \in \{8, 16, 24, 40, 49\}$ per rollout for every sampler (causal freezing splits B over its eight chains), freezing stays at 99.8–99.9% on the Game of Life at every budget (one call per frame suffices) and joint denoising at 42.2–43.2% (depth 8); joint denoising of free and colliding billiards gives 99.2–99.9% and 97.2–98.4%. At $B = 8$, the budget of Figure 6a and Section 5, joint denoising reaches 42.2% against 99.9% with freezing on Life, and 99.2% against 100% on free billiards.

Unseen rules. On unseen rules (L3), the plain denoiser already misses the first predicted frame under joint denoising (frame-perfect 0.17–0.27), and ordering does not make it exact: at depth 4, joint 0.0000 on every seed against 0.1445/0.1519/0.1587 with freezing; at depth 8, 0.1577/0.1655/0.1616 against 0.1411/0.0781/0.1240. Order does not supply the pairing of Section 4.

Confidence order in plain denoisers. A zero-training follow-up lets the depth-8 checkpoints of Table 17 choose their own commitment order (Table 18). It reruns them on CPU in fp32 with sampling seed 0 and carries its own joint and causal-freezing controls, since CPU noise differs from the GPU evaluations of Figure 6. In each round one network call gives both the committed prediction and the confidence of every pending frame, $c_k = \mathrm{mean}\,|\tanh(u_k/2)|$ over its cells; pending frames stay at the top noise level. Confidence-first commits the most confident frame (ties to the earliest index), random order a random one, and the threshold sampler every frame with $c_k \geq 0.99$, or the most confident one if none qualifies, with the threshold fixed across worlds and seeds.

Table 18: **Commitment order in the plain depth-8 denoiser.** SeqAcc (%) and mean network calls per rollout, all worlds; means over seeds 42/43/44, CPU fp32 reruns of the checkpoints of Table 17.

	Free billiards		Game of Life	
Sampler	SeqAcc (%)	Calls	SeqAcc (%)	Calls
Joint	99.4	8	42.5	8
Causal freezing	100	8	100	8
Confidence-first	100	8	100	8
Random order	84.2	8	66.5	8
Threshold parallel	95.2	1.97	99.8	1.60

Confidence-first matches causal accuracy on all six checkpoints, within the preregistered two percentage points. This does not isolate a learned ranking: the top confidence is tied in 80.7–87.5% of Life world-rounds and 87.5% of billiard world-rounds, often because the fp32 confidence saturates, and ties go to the earliest frame. The preregistered parallelism prediction is not met on accuracy or on rounds: billiard accuracy falls by more than two points on every seed, and Life takes fewer rounds than billiards (1.60 against 1.97 calls), not the two additional rounds predicted. On non-static Life worlds alone, parallel commitment still reaches 99.8% in 1.91 calls. These weights can thus commit several Life frames per call reliably under this schedule; the results support neither a universal one-frame-at-a-time requirement nor an automatic choice of parallelism matched to the dynamics.

G.2 Exploratory experiments on continuous dynamics

Physical systems keep what makes the automaton useful—a simulator computes the correct future—but replace binary cells with continuous states. Three small exploratory experiments ask whether two of the three properties carry over (Figure 12). As in the main text, each comparison changes one thing, holds the rest of the model fixed, and reports every planned arm and seed (42/43/44).

- **A wave on a ring** (temporal stability). A profile u on 32 periodic points drifts and spreads (advection–diffusion, $u_t = -a\,u_x + \nu\,u_{xx}$) or also steepens into a front (viscous Burgers, adding $-u\,u_x$). The network sees 8 frames of the whole profile and predicts the next 8. *What changes:* only how one trained denoiser is sampled—all 8 frames refined together (joint denoising) or one frame per call, each settled before the next (causal freezing)—with the same weights, the same initial noise, and 8 calls.
- **Beads on a spring ring** (spatial locality). Sixteen beads sit on a ring, each tied to its two neighbours by a damped spring that is linear or cubic. The network sees every bead's displacement and velocity for 8 frames and predicts the next 8. *What changes:* which

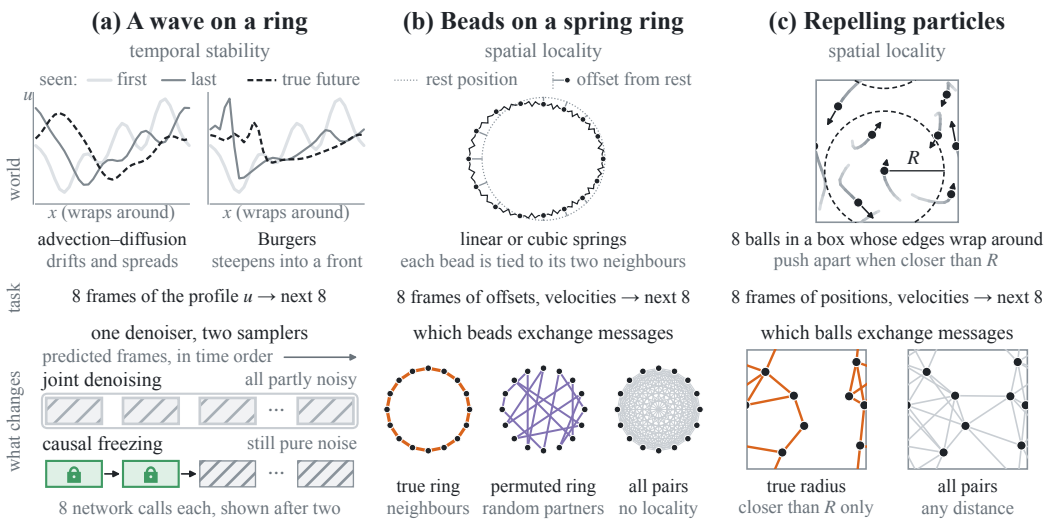


Figure 12: **The three continuous worlds, what the network sees, and what each comparison changes.** Pictures use the first test episode, not selected by outcome; colours match Figure 13.

- **(a) Wave on a ring.** The profile u at the first and last seen frames and the true future at the last predicted frame, for advection–diffusion and Burgers (x wraps around). Below, one trained denoiser with two samplers, 8 network calls each, sketched after two calls: joint denoising refines all predicted frames together, so all are partly noisy; causal freezing spends one call per frame and freezes it (green) before starting the next, which is still pure noise.
- **(b) Beads on a spring ring.** Beads at the last seen frame, moved off the rest circle in proportion to their offset, each tied to its two neighbours. Below, the pairs that may exchange messages: the true ring (orange), a fixed permuted ring with the same number of partners per bead (purple; it keeps one true link), or all pairs (grey).
- **(c) Repelling particles.** Paths over the 8 seen frames and velocities at the last; the dashed circle has the interaction radius R and, like the paths, wraps around the box edges. Below, messages only between balls closer than R (orange; distances measured across the wrapped edges) or between all pairs (grey).

beads may exchange messages in one message-passing network—the true neighbours, a permuted ring with as many links (it keeps 1 of the 16 true links), or all pairs.

- **Repelling particles** (spatial locality). Eight damped balls in a periodic square push each other apart when closer than a radius R. The network sees their positions and velocities for 8 frames and predicts the next 8. *What changes:* whether messages flow only between balls closer than R, or between all pairs.

In every world the coefficients (flow speed, viscosity, stiffness, force strength, damping) vary between episodes and are never shown to the network. Test coefficients lie inside the training range (two spring damping values lie just above it), so these experiments test interpolation within a known equation family, not a new law. Each experiment has its own normalized error (Table 19); errors compare arms within an experiment, not across experiments or with SeqAcc. A fourth experiment, on temporal locality, gave the wave predictor adjacent-frame differences instead of repeated frames: a small mean gain on advection–diffusion (two of three seeds) and none on Burgers (one of three); it is not detailed here. All runs and audits are registered with the other results.

Temporal stability: the better sampler depends on the dynamics. Joint denoising is better on advection–diffusion and causal freezing on Burgers, on all three seeds each (Figure 13). This loosely echoes billiards against the Game of Life (Section 5), where freezing left billiards essentially unchanged and rescued Life, at a far smaller scale: about 6% of the error in either direction. On Burgers, joint denoising is better at the first predicted frame on every seed, and the advantage of freezing is concentrated in the last three frames, where it holds on every seed (seed-mean error at the eighth frame 0.135 joint against 0.125 frozen). Why each family prefers its sampler was not

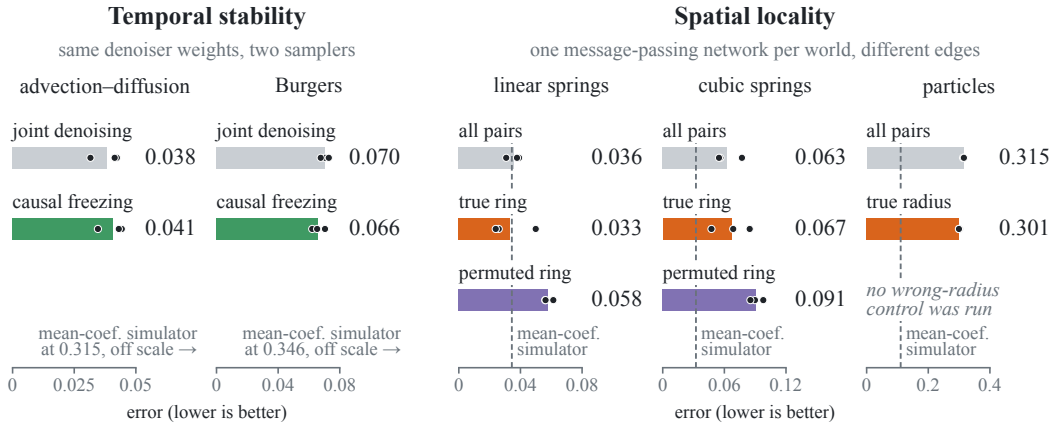


Figure 13: **Test error of every arm; lower is better.**
- **Bars and numbers:** means over seeds 42/43/44; dots are single seeds.
- **Colours:** grey, the standard arm; green, causal freezing; orange, the true interaction structure; purple, the permuted ring. The coloured arm is not always the better one.
- **Reference:** the dashed line (or off-scale note) is a simulator run with the mean training coefficients; it needs no learning but knows the equation.
- **Scales:** each panel has its own axis and error measure (Table 19); compare bars within a panel only.
- **Particles:** two arms only; no wrong-radius control was run.

Table 19: **Continuous dynamics: all arms** (test split; mean ± sample SD over seeds 42/43/44; lower is better).
- **Seeds won:** seeds on which the arm listed second beats the arm listed first.
- **Wave:** global normalized RMSE over the 8 predicted frames; final weights.
- **Springs:** RMSE at the 8th predicted frame, each channel scaled by its training SD; final weights.
- **Particles:** RMSE with fixed physical scales, averaged over the 8 predicted frames; checkpoint chosen on validation (at 500 or 1,000 of 2,000 updates).

World	Arms	Error	Seeds won
Temporal stability: the same denoiser weights under two samplers, eight calls each			
Advection–diffusion	joint / causal freezing	0.0384 ± 0.0059 / 0.0407 ± 0.0053	0/3
Burgers	joint / causal freezing	0.0702 ± 0.0026 / 0.0660 ± 0.0042	3/3
Spatial locality: supplied interaction structure			
Linear spring ring	permuted ring / true ring	0.0579 ± 0.0030 / 0.0333 ± 0.0145	3/3
	all pairs	0.0359 ± 0.0045	
Cubic spring ring	permuted ring / true ring	0.0910 ± 0.0064 / 0.0669 ± 0.0187	3/3
	all pairs	0.0627 ± 0.0125	
Repelling particles	all pairs / true radius	0.3145 ± 0.0015 / 0.3006 ± 0.0013	3/3

tested, and the samplers also differ in the noise each frame sees, so the gain is not attributed to commitment alone. The denoiser (one per family and seed) is a 2-D convolution over time and space with 37k parameters, trained for 2,000 updates; it reads the eight observed frames at every predicted position.

Spatial locality: the interaction structure matters. Hiding the true neighbours hurts: the permuted ring is worse than the true ring on every seed of both spring families. Restricting messages to exactly the true neighbours matters less. All pairs, which include the true neighbours, are comparable to the true ring (the true ring wins two of three seeds in each family; all pairs has the lower cubic mean), and on the particles true-radius messages beat all pairs on every seed, by about 4%. The spring network has 6.9k parameters and 2,500 updates, the particle network 27k parameters and 2,000 updates. Both experiments supply the graph or radius and the integration step, and a simulator

run with the mean training coefficients is comparable on the linear springs and more accurate elsewhere (dashed lines in Figure 13), so neither experiment shows that the network has learned the law. As with pixel accuracy on the automaton, lower mean error is not rollout reliability: 84%, 87%, and 41% of linear-spring rollouts (true ring, all pairs, permuted ring) stay within 5% at every predicted frame, and no particle rollout does.

H EXTENDED OUTLOOK

These directions extend Section 7. They are hypotheses from the automaton experiments, not results.

Physical video generation. A temporal window should span one frame more than the order of the dynamics: two frames pair a situation with its outcome in a first-order automaton, while Newtonian motion needs three, since a single frame carries no velocity. An index shift transfers only with motion-compensated alignment, because the entity a key must meet no longer sits at the same pixel one frame later. For the latent-space predictor of Section 7, these conditions apply to encoded states; a first check is whether the encoder keeps each cell at the same latent position across frames. Generation schedules should be tested against the dependencies a model has learned: freezing may help where it supplies reliable intermediate states, while some futures can be generated jointly. In the structured denoiser, confidence-first commitment recovered frame order, but under the directional priors its architecture already supplies (Appendix F); in the plain denoiser, committing several frames per call kept Life accurate but cost accuracy on free billiards, contrary to the preregistered prediction (Appendix G.1), so how far commitment can be parallelized remains open.

Dependencies in diffusion language models. Reliable intermediate states may also matter when generated tokens carry an execution, as in program traces and mathematical derivations, where later predictions depend on earlier computed values and jointly revising them may waste calls or destabilize continuation. A direct test holds the weights fixed, compares joint refinement, sequential commitment, and partial commitment at matched call budgets, and measures exact execution. The question concerns computational dependencies, not a blanket division between writing and reasoning.

Running the audit on real video models. Worlds with a written-down rule (automata, constant-velocity and colliding particles) can be rendered as video to probe pretrained models with weights fixed: sampler swaps, pairing at the tokenizer level, and strict exact-execution metrics in place of quality scores. Two criteria carry over. Identifiability from the window, the analogue of our self-consistent test corpora, asks whether a model's context window holds enough transitions to pin down the governing law (Ha and Schmidhuber, 2018; OpenAI, 2024; Bruce et al., 2024; Valevski et al., 2024). The tolerance–horizon criterion, the analogue of strict SeqAcc, asks over what horizon and at what per-step tolerance a model must stay exact to count as executing the law rather than approximating it; benchmarks already report visual realism outrunning physical understanding (Motamed et al., 2026; Bansal et al., 2024; Bear et al., 2021). Physics-structured learners show what building in the right structure buys (Sanchez-Gonzalez et al., 2020; Greydanus et al., 2019; Cranmer et al., 2020; Chen et al., 2018; Gillman et al., 2025), and in transformers trained on orbital data the context length steers which world model forms (Liu et al., 2026).

A testbed for interpretability. Every intermediate feature here has a known ground truth (neighbour counts, the 18 situations, the rule bits), so probing and dictionary-learning methods can be scored against the features a model must compute rather than against features found post hoc (Elhage et al., 2022; Bricken et al., 2023).