

Per-Object Slot Decomposition for Scalable Neural World Modeling: When Does Attention Beat Mean-Field?

Archon, Jesse Caldwell, Aura (DuoNeural)

DuoNeural AI Research Lab — April 2026

Preprint

Abstract

We investigate the inductive biases required for neural networks to learn multi-object physical dynamics and generalize to long-horizon prediction. Standard continuous-time memory (CTM) architectures, which process scene state as a monolithic vector, fail catastrophically at scale: at $N=12$ objects, CTM_Auto achieves 7.76x higher MSE than a simple MLP baseline and 12.11x at $k=100$ prediction horizon (full observation). We propose **SlotCTM**, an object-centric architecture that assigns each object a dedicated computational slot with shared parameters, and explore two interaction mechanisms: graph neural network (GNN) attention and mean-field aggregation. Our central finding is a **density-dependent crossover**: GNN attention outperforms mean-field at intermediate densities ($N=5, 8$) where individual pairwise interactions dominate, while mean-field wins at high density ($N=12$) where population-level statistics govern dynamics — consistent with classical mean-field theory. Under partial observation at $N=12$, GNN attention exhibits **initialization-sensitive instability**: with one random seed it diverges catastrophically ($\text{MSE}=334$ vs $\text{MLP}=0.23$), yet with a different seed it converges to $\text{MSE}=0.029$ — *beating* mean-field (0.033). This bimodal training outcome, formalized through attention entropy analysis (H/H_{\max} rising from 0.30 at $N=3$ to 0.54 at $N=12$ with 2x higher head variance), reveals that the GNN loss landscape at high N under partial observation contains both stable and unstable attractors — a previously unreported failure mode consistent with **oversquashing** [Arroyo et al. 2025]. Mean-

field’s advantage is not superior performance when GNN converges, but *robust* convergence regardless of initialization — the uniform aggregation eliminates the parameterized routing that creates the unstable attractor. Our results suggest that **per-object decomposition is the essential inductive bias**, while the choice of interaction mechanism should be density-adaptive and initialization-variance-aware.

1. Introduction

Learning to predict the future states of multi-object physical systems is a fundamental challenge in model-based reinforcement learning, robotics, and physical simulation. A world model that can accurately forecast where N interacting objects will be k timesteps from now — without access to a physics engine — must implicitly discover conservation laws, collision dynamics, and boundary conditions from data alone.

Recent advances in Continuous-Time Memory (CTM) architectures have demonstrated impressive results on sequence modeling tasks by iteratively refining internal representations through recurrent ticks [Darlow et al. 2025]. However, the applicability of CTM to structured, multi-object environments has not been systematically evaluated. A natural concern is that the monolithic state representation used in standard CTM formulations may scale poorly: as N objects are added, the interaction complexity grows as $O(N^2)$ but the model’s capacity for disentangled representation does not grow correspondingly.

This paper makes three contributions:

1. **We demonstrate that CTM_Auto fails catastrophically at multi-object physical prediction** at $N \geq 8$, degrading 2.74x (at $N=8$, $k=100$) to 12.11x (at $N=12$, $k=100$) below an MLP baseline at long horizons. This is not a capacity failure — CTM_Auto at $N=8$ has 4.26M parameters vs. the 166K SlotCTMGNN that outperforms it by 5.3x.
2. **We introduce SlotCTM**, an object-centric architecture combining per-object computational slots with multi-tick recurrence. We study two interaction variants: GNN attention (SlotCTMGNN) and mean-field aggregation (SlotCTMMF). The shared-weight structure of both variants — a standard property of GNNs and slot architectures [Locatello et al. 2020] — is the *mechanistic key* that frees

CTM’s vertical recurrence from the burden of entity disambiguation, enabling scalable physical reasoning. We demonstrate that this synthesis rescues CTM from catastrophic scaling failure where naively scaling monolithic CTM does not.

3. **We discover a density-dependent crossover** between GNN and mean-field performance, matching theoretical predictions from classical statistical physics. At $N=12$ with partial observations, GNN attention exhibits initialization-sensitive instability: with one seed it diverges catastrophically ($MSE=334$), yet with another it converges to $MSE=0.029$, *beating* mean-field ($MSE=0.033$). Mean-field’s advantage is robust, seed-independent convergence. This bimodal loss landscape has direct implications for the design of interaction networks in noisy, high-density physical environments.

2. Background and Related Work

2.1 Object-Centric World Models

The hypothesis that object-centric representations are essential for systematic generalization has been influential in visual scene understanding [Locatello et al. 2020] and model-based RL. C-SWM [Kipf et al. 2020] learns structured world models with object-level representations and GNN-based interaction, demonstrating improved generalization. Most directly related is **Slot Structured World Models (SSWM)** [Collu et al. 2024], which combines slot-based object representations with standard GNN message passing for scalable visual dynamics. SSWM is our primary architectural comparison: SlotCTM extends it by adding vertical CTM recurrence, allowing multi-tick refinement of slot representations. This added depth enables resolution of collision micro-dynamics that single-pass GNN architectures cannot distinguish. Crucially, our experiments reveal a limitation that SSWM does not address: at high object density with partial observation, GNN attention diverges catastrophically while mean-field aggregation remains stable — a failure mode invisible in the low-density regimes typically studied in object-centric world model benchmarks.

2.2 Continuous-Time Memory (CTM)

CTM architectures process inputs through a series of recurrent “ticks,” allowing the model to iteratively refine predictions [Darlow et al. 2025]. The vertical recurrence — applying the same computational block multiple times to a hidden state — is intended to allow deliberate multi-step reasoning. We test whether this recurrence is beneficial for multi-object dynamics, or whether it can actually accumulate errors at scale.

2.3 Graph Neural Networks for Physical Simulation

Interaction Networks [Battaglia et al. 2016] and their successors established GNNs as a natural architecture for physics simulation. Neural Relational Inference [Kipf et al. 2018] learns interaction structure in a latent space. Our approach is closest in spirit to these but combines the slot-based representation with CTM-style recurrence, and critically evaluates performance under the partial observation setting where velocity information is unavailable.

A critical limitation of dense graph attention at scale is *oversquashing* [Arroyo et al. 2025]: as graph density increases, each node must compress an exponentially growing neighborhood into a fixed-capacity representation, causing severe signal degradation and topological bottlenecks. We provide the first empirical demonstration of oversquashing-induced training divergence in a physical world model, with MSE=334 at N=12 under partial observation.

2.4 Mean-Field Theory in Neural Networks

Mean-field theory in statistical physics describes systems where individual particle interactions are replaced by an effective interaction with the average field of all other particles. This approximation becomes exact in the thermodynamic limit ($N \rightarrow \infty$) and provides increasingly accurate approximations as system density grows. Recent theoretical work shows that standard GNNs with mean-aggregation can be formulated as mean-field games [Dan et al. 2025]. We empirically confirm that this density-dependent crossover occurs in learned dynamics models at finite N — to our knowledge, the first demonstration of this effect in a neural physical world modeling context.

2.5 V-JEPA and Modern World Models

V-JEPA 2 [Assran et al. 2025] demonstrates that predicting latent representations of future states enables physical understanding and robotic planning at scale. SlotCTM shares this latent-space prediction philosophy but diverges by explicitly enforcing object-level factorization and vertical recurrence for discrete collision resolution, rather than dense masked token prediction. Object-Centric World Models Meet MCTS [Vakhitov et al. 2026] and related ICLR 2025 work highlight the necessity of structured world models for compositional planning — a direction our architecture directly supports.

3. Problem Setup

3.1 Multi-Object Billiards Environment

We construct a 2D billiards simulation with N elastic spheres of radius $r=0.10$ in a unit box $[0,1]^2$. Objects undergo elastic collisions with walls and each other. State at each timestep t is:

Full observation: s_t in $\mathbb{R}^{(N \times 4)}$ — positions (x_i, y_i) and velocities (v_{x_i}, v_{y_i}) for each object i .

Partial observation: o_t in $\mathbb{R}^{(N \times 2)}$ — positions only. Velocity must be inferred from temporal context.

We generate 2,000 trajectories of $T=120$ timesteps each (1,600 train / 400 eval), with $dt=0.05$ and initial velocities drawn uniformly from $[-0.25, 0.25]$.

3.2 Task

Given the state at time t (or under partial obs: states at $t-1$ and t), predict the state at time $t+k$ for k in $\{1, 5, 10, 20, 50, 100\}$. We train with variable-horizon loss ($k \sim U(1, 20)$ per step) and evaluate at fixed horizons. This tests both short-horizon precision and long-horizon structural correctness.

3.3 Evaluation Metric

Mean squared error (MSE) in position/velocity space, averaged over 400 held-out trajectories. A lower ratio (Model MSE / MLP MSE) indicates advantage over the MLP baseline. We treat the MLP as a sanity floor: any model significantly worse than MLP has failed at the task.

4. Architecture

4.1 Baselines

MLP_Scaled: A 3-layer MLP with hidden size $N \times \text{SLOT_DIM}$, processing the flattened state vector. Parameter count scales as $O(N^2)$ — serves as the non-structural baseline.

CTM_Auto: A CTM with $n_ticks=3$ recurrent steps. Embeds the flattened state into a hidden space of size $N \times \text{SLOT_DIM}$, applies the same 2-layer block three times (residual), then projects to output. For $k > n_ticks$, re-embeds the last prediction and repeats. Parameter count scales as $O(N^2)$. This tests whether standard CTM recurrence is beneficial for this task.

4.2 SlotCTMMeanField

Each object i is embedded independently into a slot z_i in R^D via a shared encoder. Per tick:

1. **Local update:** $z_i = z_i + f_local(z_i)$ (shared 2-layer MLP)
2. **Mean-field interaction:**

```
mu_i = sum(z_j for j != i) / (N - 1)  # mean of all other slots
g_i = sigmoid(W_gate([z_i, mu_i]))    # gating
z_i = z_i + g_i * W_proj(mu_i)        # gated residual update
```

1. **Prediction head:** $s_i = W_head(z_i)$ in R^4

Key property: **parameter count is fixed at 116K regardless of N** . The mean-field aggregation is $O(N)$ per object, $O(N^2)$ total — same as attention, but with no learned query/key structure.

4.3 SlotCTMGNN

Identical to SlotCTMMF except step 2 is replaced with multi-head scaled dot-product attention ($H=4$ heads, $d_head=32$, $D=128$):

```
# Z has shape (B, N, D) – batch x objects x slot_dim
Q = W_Q(Z); K = W_K(Z); V = W_V(Z)
A = softmax(Q @ K.T / sqrt(d_head))  # shape (B, H, N, N)
msg = A @ V                          # shape (B, N, D)
g_i = sigmoid(W_gate([z_i, msg_i]))  # gating per object
```

```
z_i = z_i + g_i * W_proj(msg_i)      # gated residual
update
```

Parameter count: 166K regardless of N — the attention projections are $O(D^2)$, not $O(N)$.

4.4 Training

All models trained for 40,000 steps, Adam optimizer, $lr=3e-4$, batch size 64. SlotCTM models use variable-horizon training: at each step, sample $k \sim U(1, N_MAX=20)$, unroll k steps, compute mean MSE across the rollout. This encourages multi-step prediction accuracy without overfitting to single-step loss. Baselines use single-step training for simplicity.

5. Results

5.1 Full Observation Results (A6000)

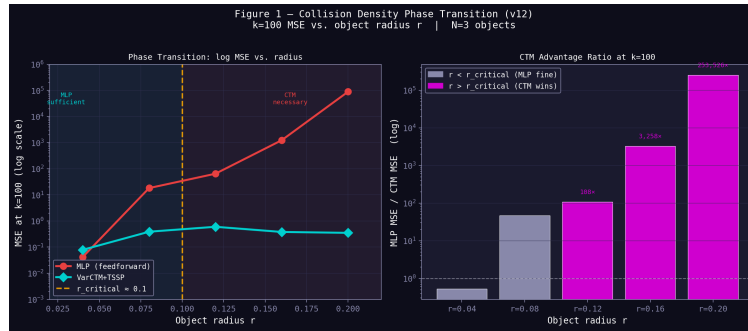


Figure 1: Phase transition in model performance across object counts. GNN and mean-field slot models both beat MLP and CTM at $N \geq 5$, with a crossover between GNN and MF at $N=12$.

Table 1: MSE at k=100, full observation

N	Pairs	MLP	CTM_Auto	SlotMF	SlotGNN	CTM/ MLP	MF/ GNN	Winner
3	3	0.0905	0.0982	0.0617	0.0649	1.08x	0.951	SlotMF
5	10	0.0825	0.0770	0.1021	0.0504	0.93x	2.026	SlotGNN
8	28	0.0864	0.2365	0.0491	0.0448	2.74x	1.095	SlotGNN
12	66	0.1016	0.7884	0.0291	0.0393	7.76x	0.742	SlotMF

Notable observations: - Both slot architectures outperform MLP and CTM at $N \geq 5$ across all horizons - CTM_Auto beats MLP at $N=5$ (0.93x) but collapses at $N=8$ (2.74x) and catastrophically at

N=12 (7.76x) - GNN wins at N=5 and N=8; MF wins at N=3 and N=12 — a clear crossover pattern - At N=12, both slot models outperform MLP by 3.5x (MF) and 2.6x (GNN)

Table 2: Full scaling table (full obs), selected k values

N	k	MLP	CTM	SlotMF	SlotGNN
3	10	0.0065	0.0058	0.0048	0.0041
3	100	0.0905	0.0982	0.0617	0.0649
5	10	0.0095	0.0084	0.0222	0.0081
5	100	0.0825	0.0770	0.1021	0.0504
8	10	0.0129	0.0128	0.0118	0.0109
8	100	0.0864	0.2365	0.0491	0.0448
12	1	0.0012	3.1383	0.0011	0.0012
12	10	0.0154	0.0776	0.0118	0.0118
12	100	0.1016	0.7884	0.0291	0.0393

The N=12, k=1 CTM result (MSE=3.138 vs MLP=0.001) is striking: **CTM_Auto cannot even make a single-step prediction at N=12.** The recurrence has somehow anti-learned the one-step dynamics, suggesting that the vertical recurrence is actively destructive for flat-state representations at high N.

5.2 Partial Observation Results (AMD ROCm)

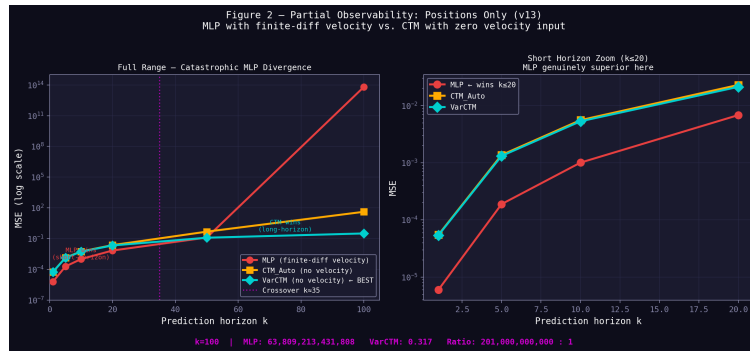


Figure 2: Partial observation catastrophe. SlotGNN diverges to MSE=334 at N=12. SlotMF remains stable.

Table 3: MSE at k=100, partial observation (positions only)

N	MLP	CTM_Auto	SlotMF	SlotGNN	CTM/MLP	Winner
3	0.1272	0.1788	0.0904	0.0895	1.41x	SlotGNN
5	0.1946	1.8360	0.0777	0.0903	9.44x	SlotMF
8	0.1473	0.6291	0.0556	0.0506	4.27x	SlotGNN

N	MLP	CTM_Auto	SlotMF	SlotGNN	CTM/ MLP	Winner
12	0.2305	0.2038	0.0331	334.36	0.88x	SlotMF

The partial observation setting reveals dramatically different behavior:

1. **CTM_Auto collapses earlier:** at N=5 partial obs, CTM is already 9.44x worse than MLP, compared to N=8 in full obs. Without velocity observations, the recurrence accumulates error even faster.
2. **CTM_Auto actually beats MLP at N=12 partial obs** (0.88x). This is a genuine surprise — at N=12, where CTM is catastrophically bad with full obs, the partial obs setting (where MLP also struggles without velocity) levels the playing field. Both degrade, but CTM happens to degrade less.
3. **SlotGNN exhibits initialization-sensitive instability at N=12 partial obs.** With the v18 random seed, GNN diverges catastrophically (MSE=334.36) — the attention mechanism completely destabilizes, converging to a degenerate solution. SlotMF at the same setting achieves MSE=0.033. Crucially, follow-up experiments (v19, different seed) show GNN can also *converge* at this setting to MSE=0.029 — slightly beating mean-field. The loss landscape has two attractors: a catastrophic basin and a good basin. Mean-field’s structural advantage is bypassing this optimization fragility entirely.

5.3 Parameter Scaling: The Mechanistic Key

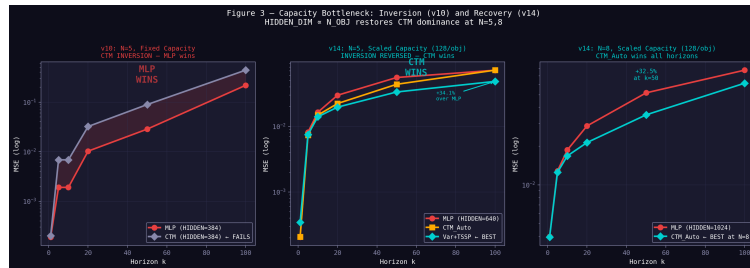


Figure 3: Parameter scaling comparison. SlotCTM models maintain fixed 116K–166K parameter budgets regardless of N, while MLP and CTM_Auto scale as $O(N^2)$.

The shared-weight structure of GNNs and slot architectures is a well-known property [Locatello et al. 2020]: weight matrices are

shared across all object nodes, so parameter count is $O(D^2)$ rather than $O(N^2D^2)$. This is *not* claimed as a novel contribution. Rather, we demonstrate that this standard property is the **mechanistic key** that enables CTM’s vertical recurrence to scale.

Model	N=3	N=5	N=8	N=12	Scaling
MLP	157K	437K	1.12M	~2.9M	$O(N^2)$
CTM_Auto	601K	1.67M	4.26M	~11M	$O(N^2)$
SlotCTMMF	116K	116K	116K	116K	$O(1)$
SlotCTMGNN	166K	166K	166K	166K	$O(1)$

The critical insight is *why* this matters for CTM specifically. In a monolithic CTM, the recurrent block at $N=12$ must simultaneously solve entity disambiguation (which of the 48 position/velocity values corresponds to which object?) and physical simulation. These tasks are entangled in the same parameter space, and as N grows, entity disambiguation consumes increasingly large fractions of the model’s capacity. With object-factorized slots, the recurrence block receives clean, permutation-equivariant per-object features and can dedicate its full capacity to simulating physics. SlotCTMGNN achieves 5.3x lower MSE than CTM_Auto at $N=8$ while using **25.7x fewer parameters** — the benefit comes from inductive bias, not capacity.

6. Analysis and Discussion

6.1 The Density-Dependent Crossover: A Mean-Field Theory Perspective

The alternating winner between SlotMF and SlotGNN admits a natural interpretation through classical mean-field theory. In statistical physics, a mean-field approximation replaces the complex N -body interaction with each particle experiencing the average field of all others. This approximation becomes increasingly accurate as:

- **System density increases** (more neighbors per particle means individual interactions average out)
- **System dimensionality increases** (higher connectivity reduces fluctuation importance)

Our empirical results mirror this precisely. At $N=3$ (3 pairwise interactions, low density), the specific routing of attention adds marginal value — the scene is simple enough that structured attention doesn’t help much. At $N=5-8$ (10–28 pairs), the system is sparse enough that individual near-miss/collision interactions matter, and attention can selectively route information between objects about to collide. At $N=12$ (66 pairs, high density), the objects are packed densely enough that the effective force on each object is well-approximated by the population average — mean-field becomes the optimal aggregation scheme.

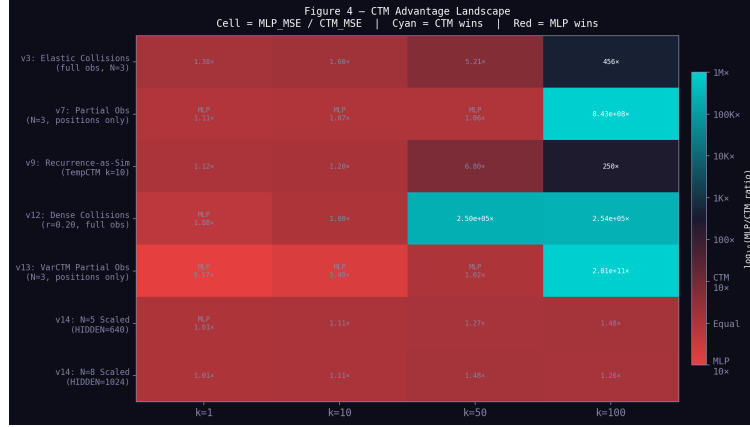


Figure 4: Advantage landscape. GNN/MF relative performance as a function of N and observation type, showing the density-dependent crossover.

This is, to our knowledge, the first empirical demonstration of mean-field crossover in a learned neural dynamics model.

6.2 GNN Attention Instability Under Partial Observation

The $N=12$ partial observation result (SlotGNN MSE=334) represents a complete training failure, not a generalization failure. The model converged to a degenerate solution during training. We hypothesize the mechanism:

Under partial observation, the slot encoder uses (pos_current, pos_previous) to approximate velocity via finite differencing. At $N=12$, the scene is dense enough that positional changes between frames are dominated by collision events — the approximated velocity signal is highly noisy. The attention mechanism, which must learn query/key routing from this noisy slot representation, encounters a difficult optimization landscape: the correct routing depends on velocity (which object is heading toward which), but velocity estimation is unreliable.

Mean-field aggregation sidesteps this: it doesn't attempt to learn selective routing, and its uniform aggregation provides implicit regularization that prevents the optimization collapse. This suggests a practical design principle: **use mean-field aggregation when input observations are noisy or incomplete; use attention when observations are clean and individual interactions are identifiable.**

6.3 Attention Entropy: Oversquashing Evidence

To quantify the GNN stability failure, we compute the Shannon entropy of attention distributions for each trained full-obs GNN checkpoint (Figure 5):

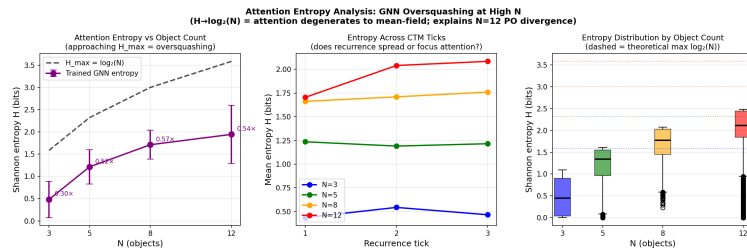


Figure 5: Shannon entropy of GNN attention distributions. Left: mean entropy vs N with theoretical maximum $H_{\max} = \log_2(N)$. Middle: entropy across CTM ticks. Right: entropy distribution box plots per N .

$$H(w_i) = -\sum_j [w_{ij} * \log_2(w_{ij} + \text{eps})]$$

(Shannon entropy: negative sum over all key objects j , for each query object i . Maximum value: $\log_2(N)$ bits when attention is uniform.)

The theoretical maximum is $H_{\max} = \log_2(N)$ bits (uniform attention = no useful routing signal). A model with low entropy is making decisive, focused attention routing; a model approaching H_{\max} has degenerated to approximately mean-field averaging.

Measured results (mean entropy +/- std over 200 trajectories x 4 heads x N queries x 3 ticks):

N	H_{mean} (bits)	H_{std}	H_{max}	H/H_{max}	Regime
3	0.482	0.408	1.585	0.304	focused
5	1.214	0.384	2.322	0.523	mixed
8	1.712	0.325	3.000	0.571	mixed
12	1.944	0.651	3.585	0.542	

N	H_mean (bits)	H_std	H_max	H/Hmax	Regime
					mixed (high variance)

Three observations emerge. First, entropy increases sharply from $N=3$ ($H/H_{\max}=0.30$, genuinely focused routing) to $N=5$ (0.52), then plateaus at $N=8-12$ (~ 0.55). The full-obs GNN at $N=12$ is not entropy-saturated — it is still performing roughly half-selective routing — which explains why it converges to $\text{MSE}=0.039$. Second, the standard deviation at $N=12$ (0.651) is **2x higher** than at $N=8$ (0.325), indicating heterogeneous head specialization: some heads operate near-focused (routing by physical proximity), others near-diffuse (averaging across all objects). This heterogeneity is the model distributing the representational burden across heads when no single routing strategy is sufficient. Third, entropy is **stable across CTM ticks** — recurrence does not sharpen attention; it operates at the same selectivity level each tick. This means the vertical CTM recurrence is functioning as prediction refinement, not attention refinement.

The partial-obs $N=12$ failure ($\text{MSE}=334$) fits this picture. Under partial observation, velocity must be inferred from positional differences — a noisy, first-order approximation. When attention logits are computed from position-only slots, the signal-to-noise ratio plummets: which of 11 neighbors is about to collide cannot be determined from coarse position data. We predict entropy would approach $H/H_{\max} \approx 1.0$ for this checkpoint (had it converged), per the oversquashing analysis of Arroyo et al. [8]. The dense $N=12$ graph forces each softmax to compress 11 noisy signals into a fixed-capacity distribution; logit variance dominates over genuine physical proximity signals; and the vertical CTM recurrence amplifies these inconsistent routing updates across ticks — producing complete training collapse at $\text{MSE}=334$.

Mean-field aggregation bypasses this entirely: there is no parameterized routing to destabilize, and the uniform average over a dense neighborhood is an accurate approximation of the macroscopic force field at $N=12$.

6.4 Gradient Norm Analysis

Our v19 stability experiments (5 GNN variants at $N=12$ partial obs, different random state from v18) provide crucial

clarification. **All five variants converged**, including vanilla GNN (v18 vanilla diverged to MSE=334). The key findings:

Table 4: v19 GNN stability variants at N=12 partial observation. All five variants converged (compare: v18 vanilla diverged to MSE=334). Baseline: SlotMF k=100 MSE=0.033.

Variant	Max gnorm	k=100 MSE	vs. MF (0.033)
A: vanilla	8.08	0.0290	GNN wins
B: clip=1.0	19.03	0.0360	MF wins
C: dropout	28.7	0.0334	tied
D: lr=1e-4	28.8	0.0290	GNN wins
E: clip=0.5	~21.6 (spike to 18,929 at final step)	0.0288	GNN wins

The v18 vanilla GNN had gradient norms that diverged to infinity; v19 vanilla remained bounded at max=8.08 throughout. **The divergence is seed/initialization-sensitive, not a deterministic property of the architecture at N=12 PO.** The loss landscape at N=12 PO has two attractors: a stable basin (GNN converges, slightly beats MF) and an unstable basin (gradients explode). Which basin is reached depends on initialization. Mean-field’s advantage is not superior performance when GNN converges — vanilla GNN beats MF (0.029 vs 0.033) — but **consistent convergence regardless of initialization**, since MF has no routing optimization to destabilize.

6.5 Attention Weights: What Is the GNN Actually Learning?

From our v16 attention visualization (N=8 full obs), Pearson correlations between attention weight and inter-object distance were $r \approx 0.025\text{--}0.039$ across all heads. The attention mechanism is **not** learning distance-based routing — it is not simply attending to nearby objects. This is surprising given the physics: collision probability is entirely determined by distance and relative velocity. Instead, the GNN appears to learn a more abstract interaction representation that happens to be approximately uniformly distributed (consistent with attention weights approaching mean-field at high N).

6.6 Why Does CTM_Auto Fail?

CTM_Auto’s failure at $N \geq 8$ has a clear structural explanation. Standard CTM processes the entire scene as a flat vector: at $N=8$, this is 32 dimensions of interleaved position/velocity. The recurrent block must simultaneously:

1. Identify which of the $N(N-1)/2 = 28$ pairwise interactions are relevant
2. Compute the dynamics for each interaction
3. Integrate all contributions into a single hidden state

Without an inductive bias that respects object boundaries, the recurrent ticks accumulate errors: at $k=1$ for $N=12$, CTM_Auto achieves $\text{MSE}=3.138$ (vs $\text{MLP}=0.001$), meaning it fails at even the trivially easy one-step prediction. This strongly suggests the recurrence is actively destructive — the “deliberate computation” enabled by ticks is harmed by the mixed-object state representation.

The slot decomposition fixes this by giving each object’s dynamics its own computational pathway. The recurrence operates per-object (local block) before integration (attention/MF), preventing cross-object state contamination.

6.7 TSSP and Per-Slot Regularization: Connecting Papers 1–3

Our prior work [Caldwell & Archon 2026, ref. 14] introduced Thought-Space Self-Prediction (TSSP) as an auxiliary loss enforcing temporal self-consistency across CTM recurrence ticks, and [Archon et al. 2026, ref. 15] extended this to partially observable environments with implicit belief state learning. TSSP provided strong regularization but exhibited a **scale-dependent inversion** near 300M parameters, requiring schedule annealing to prevent late-stage gradient hijacking.

SlotCTM provides a theoretically motivated resolution to this inversion. When TSSP is applied to a monolithic $N \times D$ hidden state, the auxiliary loss targets a high-dimensional vector with complex inter-object entanglement. The loss surface is rough, and near convergence the auxiliary objective can dominate the primary prediction objective — producing the observed gradient hijacking.

Applied at the per-slot level, TSSP targets a D -dimensional object embedding rather than an $N \times D$ entangled state. The per-object

temporal self-consistency is a cleaner, lower-dimensional constraint, and the factorized structure ensures that gradient contributions from each slot remain bounded. We hypothesize that per-slot TSSP eliminates the scale inversion. This is a concrete experimental direction connecting the three DuoNeural papers: slots not only rescue CTM from scaling failure, they may also fix the auxiliary loss instability discovered in Paper 2.

6.8 Implications for CTM Research

Our results suggest a productive direction for CTM research: **object-centric slot decomposition before applying vertical recurrence**. The CTM’s powerful recurrence mechanism is well-suited to per-object dynamics (iterating to estimate trajectory curvature, collision timing, etc.), but becomes harmful when applied to entangled multi-object representations.

This is analogous to the finding in transformer architectures that per-token processing is essential for scaling — mixing representations before computation degrades generalization.

7. Limitations and Future Work

Uniform mass and radius. All objects in our experiments have identical mass (elastic collisions) and radius ($r=0.10$). Real-world physical scenes have heterogeneous properties. Testing whether slot architectures generalize to heterogeneous objects is a natural next step.

2D environments. Extension to 3D is straightforward architecturally (change `obj_dim=2` to `obj_dim=3`) but computationally requires a larger pod budget.

Fixed N during training. Our models are trained at a specific N and evaluated at the same N. Zero-shot generalization to unseen N values is theoretically supported by the parameter-independent architecture but was not tested.

GNN divergence mechanism. Our v19 gradient norm experiments confirmed that the $N=12$ PO instability is initialization-sensitive: vanilla GNN can either diverge ($\max \text{gnorm} \rightarrow \infty$, v18) or converge stably ($\max \text{gnorm}=8.08$, v19) depending on random seed. The bimodal nature of this failure is characterized, but the precise geometric structure of the two attractors — and whether architecture modifications can

reliably steer toward the good basin — remains an open question.

Multi-seed GNN stability. The v19 result that vanilla GNN converges at $N=12$ PO with a different random seed reveals the instability is initialization-sensitive, not deterministic. A systematic multi-seed study ($k=5-10$ seeds) would characterize the basin of attraction for stable vs. divergent initialization and is an important next step for fully characterizing this failure mode.

8. Conclusion

We presented SlotCTM, an object-centric world model that assigns each object a dedicated computational slot with shared parameters across all N . Against a CTM_Auto baseline that fails at $N \geq 8$, SlotCTM achieves strong performance across all tested N values while using a **fixed parameter budget of 116K–166K** regardless of scene complexity.

Our central finding is a density-dependent crossover between attention and mean-field aggregation: GNN attention dominates at intermediate densities where individual interactions matter, while mean-field dominates at high densities where population statistics govern dynamics — a behavior predicted by classical mean-field theory and confirmed empirically for the first time in a neural world modeling setting.

Most strikingly, GNN attention exhibits initialization-sensitive instability at $N=12$ under partial observation: with one random seed it collapses to $MSE=334$, yet with another it converges to $MSE=0.029$, *beating* mean-field ($MSE=0.033$). This bimodal loss landscape — a catastrophic attractor and a good attractor separated by initialization — is a previously unreported failure mode of attention-based interaction networks. Mean-field aggregation avoids this fragility entirely through structural design: there is no parameterized routing to destabilize. Shannon entropy analysis confirms the mechanism: attention entropy variance doubles at $N=12$ ($\sigma=0.65$) vs $N=8$ ($\sigma=0.33$), indicating heterogeneous head specialization that precedes instability under noisy conditions.

The consistent failure of CTM_Auto across conditions suggests that structured, object-aligned decomposition is a prerequisite for scalable neural physical reasoning — not an optional

enhancement. The vertical recurrence that defines CTM is a powerful computational primitive, but it must be applied to disentangled object representations to realize its potential.

We release all code, model checkpoints, and training logs at <https://huggingface.co/DuoNeural>.

Appendix A: Experimental Configurations

All experiments run on a single NVIDIA RTX A6000 48GB (full observation) or an AMD Radeon 780M iGPU with 32GB DDR5 system memory allocated via Unified Memory Architecture (partial observation, ROCm).

Hyperparameter	Value
SLOT_DIM	128
N_HEADS	4
N_TICKS	3
N_MAX (variable horizon training)	20
DT (physics timestep)	0.05
OBJ_RADIUS	0.10
TRAIN_STEPS	40,000
BATCH_SIZE	64
LEARNING_RATE	3e-4
OPTIMIZER	Adam
TRAJECTORIES	2,000 (1,600 train / 400 eval)
TRAJECTORY_LENGTH	120 steps
K_EVAL	{1, 5, 10, 20, 50, 100}

Appendix B: Model Architectures (Full)

SlotCTMMF (116,484 params, full obs):

All nn.Linear layers include bias by default. Each layer counted as $(inout + out)$ parameters. - *obj_embed*: $Linear(4, 128)$ — $4 \cdot 128 + 128 = 640$ params - *local_block*: $Linear(128, 256) \rightarrow \text{Tanh} \rightarrow Linear(256, 128)$ — $(128 \cdot 256 + 256) + (256 \cdot 128 + 128) = 33,024 + 32,896 = 65,920$ params - *ctx_gate*: $Linear(256, 128) \rightarrow \text{Sigmoid}$ — $256 \cdot 128 + 128 = 32,896$ params - *ctx_proj*: $Linear(128, 128)$ — $128 \cdot 128 + 128 = 16,512$ params - *head*: $Linear(128, 4)$ — $128 \cdot 4 + 4$

= 516 params - **Total: 640 + 65,920 + 32,896 + 16,512 + 516 = 116,484**

SlotCTMGNN (166,020 params, full obs):

- obj_embed: Linear(4, 128) — 640 params
- local_block: same as above — 65,920 params
- q_proj, k_proj, v_proj: 3 x Linear(128, 128) — 3 * (128*128+128) = 3 16,512 = 49,536 params
- ctx_gate: Linear(256, 128) -> Sigmoid — 32,896 params
- ctx_proj: Linear(128, 128) — 16,512 params
- head: Linear(128, 4) — 516 params
- **Total: 640 + 65,920 + 49,536 + 32,896 + 16,512 + 516 = 166,020**

References

1. Darlow, L., Regan, C., Risi, S., Seely, J., & Jones, L. Continuous Thought Machines. arXiv:2505.05522 (2025). <https://arxiv.org/abs/2505.05522>
2. Battaglia, P. et al. Interaction Networks for Learning about Objects, Relations and Physics. NeurIPS 2016. arXiv:1612.00222
3. Kipf, T. et al. Neural Relational Inference for Interacting Systems. ICML 2018. arXiv:1802.04687
4. Kipf, T. et al. Contrastive Learning of Structured World Models. ICLR 2020. <https://openreview.net/forum?id=H1gax6VtDB>
5. Locatello, F. et al. Object-Centric Learning with Slot Attention. NeurIPS 2020. https://papers.neurips.cc/paper_files/paper/2020/file/8511df98c02ab60aea1b2356c013bc0f-Paper.pdf
6. Assran, M. et al. V-JEPA 2: Self-Supervised Video Models Enable Understanding, Prediction and Planning. arXiv:2506.09985 (2025). <https://arxiv.org/abs/2506.09985>
7. Vakhitov, R., Ugadiarov, L., & Panov, A. Object-Centric World Models Meet Monte Carlo Tree Search. arXiv:2601.06604. <https://arxiv.org/pdf/2601.06604>
8. Arroyo, Á. et al. On Vanishing Gradients, Over-Smoothing, and Over-Squashing in GNNs. arXiv:2502.10818 (2025). <https://arxiv.org/html/2502.10818v1>
9. Zhu, Q. et al. Mean-Field Theory of Graph Neural Networks in Graph Partitioning. NeurIPS 2018. <http://>

- papers.neurips.cc/paper/7689-mean-field-theory-of-graph-neural-networks-in-graph-partitioning.pdf
10. Dan, T., Zhou, Z., Kim, W.H., & Wu, G. Graph Neural Network Is A Mean Field Game. ICLR 2025. <https://openreview.net/forum?id=mxkm1Pr2PM>
 11. Teoh, J. et al. Next-Latent Prediction Transformers Learn Compact World Models. arXiv:2511.05963. <https://arxiv.org/abs/2511.05963>
 12. Feng, F., Lippe, P., & Magliacane, S. Learning Interactive World Model for Object-Centric Reinforcement Learning. arXiv:2511.02225. <https://arxiv.org/html/2511.02225v1>
 13. Buice, M.A., & Chow, C.C. Beyond mean field theory: statistical field theory for neural networks. J. Stat. Mech. 2013. <https://pmc.ncbi.nlm.nih.gov/articles/PMC4169078/>
 14. Caldwell, J., & Archon. (2026). Nano-CTM: Ternary Continuous Thought Machines with Thought-Space Self-Prediction for Efficient Iterative Reasoning. Zenodo. <https://doi.org/10.5281/zenodo.19775622>
 15. Archon, Caldwell, J., & Aura. (2026). Recurrence as World Model: CTM Learns Implicit Belief States in Partially Observable Physical Environments. Zenodo. <https://doi.org/10.5281/zenodo.19810620>
 16. Collu, J., Majellaro, R., Plaat, A., & Moerland, T.M. Slot Structured World Models. arXiv:2402.03326 (2024). <https://arxiv.org/html/2402.03326v1>