ORION: EXACT CANONICAL-STATE SUPERVISION FOR SYSTEMATIC GENERALIZATION IN TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Autoregressive Transformers generally fail on out-of-distribution reasoning due to two primary reasons: positional corruption over long sequences and structural state drift across deep execution steps. We introduce ORION, a framework that supervises exact canonical execution states via auxiliary heads to anchor latent trajectories to valid computational manifolds. Across Folding, GridShift, and CLRS-30, a pre-registered 2 × 2 factorial shows that RoPE and ORION address orthogonal axes. RoPE recovers length extrapolation (59.8%), while ORION recovers depth extrapolation (66.7%). Moreover, their combination complementarily achieves strict additive gain in joint length-depth extrapolation, while either method alone offers near satisfactory results. Negative controls confirm that the gain depends strictly on exact canonical alignment; hashed state statistics fail to constrain depth extrapolation.

1 INTRODUCTION

Autoregressive Transformers fail when pushed beyond their training horizon on algorithmic reasoning tasks (Anil et al., 2022; Dziri et al., 2023; Zhou et al., 2024). Standard benchmarks conflate two distinct failure modes: growth in sequence length and growth in execution depth (Wu et al., 2024; Merrill & Sabharwal, 2024). When a model fails on a longer trace, it becomes difficult to determine whether positional attention has collapsed or the internal execution state has drifted off its valid computational manifold. Without disentangling these axes, interventions cannot be diagnosed and gains cannot be attributed.

We hypothesize that length extrapolation and depth extrapolation are orthogonal failure modes addressed by distinct inductive biases. Positional priors such as Rotary Position Embeddings (RoPE) stabilize attention over extended token sequences (Su et al., 2024; Press et al., 2021; Shaw et al., 2018), but they do not constrain latent execution trajectories. Canonical-state supervision anchors the latent state to the true computational manifold (Nye et al., 2021; Schwarzschild et al., 2021; Veličković & Blundell, 2021), but it cannot compensate for positional incoherence. Addressing only one axis leaves the network vulnerable to the other.

We test this decomposition on two exactly-solvable domains—FOLDING and GRIDSHIFT—and a CLRS pilot, using a pre-registered 2×2 factorial that varies positional encoding (absolute vs. RoPE) and state supervision (none vs. ORION). Each domain is evaluated on four splits: in-distribution, length extrapolation ($L_{\text{test}} > 3 \times L_{\text{train}}$ at fixed depth), depth extrapolation ($D_{\text{test}} > 3 \times D_{\text{train}}$ at fixed length), and joint length-depth extrapolation. This design isolates whether each bias addresses a distinct axis and whether their composition is necessary for joint generalization.

The axes fail independently. RoPE alone recovers depth extrapolation (0.021 to 0.667 exact match on FOLDING) but leaves length near baseline (0.191). ORION without RoPE is inert on both axes (0.026 depth exact match). Their composition achieves absolute gains of +1.8 and +18.0 percentage points in joint length-depth extrapolation on FOLDING and GRIDSHIFT, respectively, over the baseline (exact match 0.004 to 0.022 and 0.045 to 0.225). The gain grows with model width (+0.082 at 26M parameters, 5/5 seeds) and transfers across domains via pretraining (+0.085 and +0.035 exact match, every seed in both directions). Negative controls confirm the gain is locked to exact canonical alignment; hashed sufficient statistics fail to constrain depth drift.

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.

Contributions. We make the following four salient contributions.

1. We formalize OOD generalization failure in algorithmic Transformers into two decoupled axes: positional length extrapolation and algorithmic depth extrapolation.
2. We introduce ORION, an intermediate supervision framework that projects representations onto exact canonical execution manifolds.
3. We provide pre-registered factorial evidence that RoPE and ORION are complementary, orthogonal inductive biases.
4. We show that state supervision stabilizes latent trajectories against rollout noise rather than compressing subspaces, and that canonical-state pretraining induces a transferable execution prior.

2 Related Work

Positional encodings for length extrapolation. Standard absolute positional encodings degrade when processing sequences longer than those seen during training (Vaswani et al., 2017; Anil et al., 2022). Relative position mechanisms—RoPE (Su et al., 2024), ALiBi (Press et al., 2021), and relative position representations (Shaw et al., 2018)—stabilize attention over extended contexts by encoding relative rather than absolute token distance. Extensions such as YaRN (Peng et al., 2024) and xPos (Sun et al., 2023) further refine frequency scaling for long-horizon coherence. These methods operate on sequence-level indexing mechanics; they regularize positional dot-products but do not constrain the latent evolution of algorithmic execution states (Dziri et al., 2023; Zhou et al., 2024).

Intermediate supervision and algorithmic reasoning. Exposing intermediate execution traces improves multi-step accuracy through scratchpad prompting (Nye et al., 2021), chain-of-thought reasoning (Wei et al., 2022; Kojima et al., 2022), and execution-guided synthesis (Shi et al., 2024). Algorithmic reasoning benchmarks such as CLRS (Veličković & Blundell, 2021) and NAR (Xu et al., 2019) standardize tasks with annotated intermediate states, while recurrent and looped Transformer variants explore bounded-depth execution (Bansal et al., 2022; Dehghani et al., 2019; Giannou et al., 2023). Prior work rarely disentangles sequence length extrapolation from recursive execution depth, leaving state drift under joint distribution shift unmeasured (Wu et al., 2024; Merrill & Sabharwal, 2024).

Mechanistic interpretability as falsification. Linear probing and representation similarity metrics audit how models encode information internally (Alain & Bengio, 2016; Kornblith et al., 2019). Recent work characterizes low-dimensional subspaces and manifold structure in Transformer residuals (Elhage et al., 2021; Park et al., 2023; Nanda et al., 2023), and causal interventions test whether specific representations are necessary for task performance (Geiger et al., 2021; Meng et al., 2022). These tools are typically deployed post-hoc, leaving conclusions vulnerable to selection bias. We repurpose them as part of a pre-registered ladder: probes, ablations, and latent trajectory analysis are specified before training to falsify the state-drift hypothesis rather than to rationalize it afterward.

3 The ORION Framework

To resolve recursive latent state drift across algorithmic depth, we introduce **ORION** (**O**rthogonal **R**epresentation for **I**ntermediate **O**perational **N**avigation). ORION anchors intermediate transformer states to canonical execution manifolds via explicit state-supervision, providing continuous geometric guidance that complements position-based spatial encodings.

3.1 Problem Formulation

Let an algorithmic process be defined over an execution sequence of T steps. At step $t \in \{1, \dots, T\}$, the system maintains an underlying ground-truth state $S_t \in \mathcal{S}$. A Transformer encoder maps an input sequence X to a sequence of latent representations $\mathbf{H}^{(l)} = [\mathbf{h}_1^{(l)}, \dots, \mathbf{h}_T^{(l)}] \in \mathbb{R}^{T \times d}$ across layers $l \in \{1, \dots, L\}$. In standard autoregressive or recurrent sequence models, hidden state updates follow an

unconstrained transition $\mathbf{h}_t^{(l)} = f_\theta(\mathbf{h}_{t-1}^{(l)}, \mathbf{x}_t)$. Without explicit bounds, subtle perturbations in $\mathbf{h}_t^{(l)}$ accumulate non-linearly over extended execution depths $T \gg T_{\text{train}}$, leading to manifold escape.

3.2 Canonical State Encoding and Projection

ORION enforces structural alignment between the model's latent space and a canonical target space. We define a deterministic target mapping $\Phi : \mathcal{S} \to \mathbb{R}^{d_{\text{target}}}$, which maps ground-truth intermediate states S_t to continuous canonical embeddings $\mathbf{z}_t = \Phi(S_t)$.

To align intermediate representations without over-constraining the primary computational backbone, we introduce a lightweight parameterized projection head $g_\theta : \mathbb{R}^d \to \mathbb{R}^{d_{\text{target}}}$ at designated supervision layers $l \in \mathcal{L}_{\text{sup}}$:

$$\hat{\mathbf{z}}_t^{(l)} = g_\theta\left(\mathbf{h}_t^{(l)}\right) = \mathbf{W}_2 \cdot \text{SiLU}\left(\mathbf{W}_1 \mathbf{h}_t^{(l)} + \mathbf{b}_1\right) + \mathbf{b}_2 \tag{1}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d_{\text{inter}} \times d}$ and $\mathbf{W}_2 \in \mathbb{R}^{d_{\text{target}} \times d_{\text{inter}}}$.

3.3 Intermediate State Supervision Loss

The framework minimizes the distance between predicted manifold projections and canonical state targets across supervision layers $l \in \mathcal{L}_{\text{sup}}$ and execution steps $t \in \{1, \ldots, T\}$:

$$\mathcal{L}_{\text{ORION}} = \frac{1}{|\mathcal{L}_{\text{sup}}| \cdot T} \sum_{l \in \mathcal{L}_{\text{sup}}} \sum_{t=1}^{T} \mathcal{D}\left(g_\theta\left(\mathbf{h}_t^{(l)}\right), \Phi(S_t)\right) \tag{2}$$

where $\mathcal{D}(\cdot, \cdot)$ denotes a normalized distance metric (e.g., Cosine Distance or Mean Squared Error):

$$\mathcal{D}(\mathbf{u}, \mathbf{v}) = 1 - \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\|_2 \|\mathbf{v}\|_2} \tag{3}$$

3.4 Joint Optimization Objective

The complete model is trained end-to-end with a composite objective balancing the standard token-level task loss $\mathcal{L}_{\text{task}}$ (e.g., cross-entropy over target sequences) and intermediate state supervision:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \lambda \cdot \mathcal{L}_{\text{ORION}} \tag{4}$$

where $\lambda > 0$ is a scalar hyperparameter controlling the strength of state-manifold regularization.

By anchoring hidden trajectories $\mathbf{h}_t^{(l)}$ to $\Phi(S_t)$, ORION prevents intermediate representation drift, allowing Rotary Position Embeddings (RoPE) to maintain length extrapolation while ORION preserves functional depth integrity.

4 Experimental Setup

4.1 Exactly-Solvable Domains

We benchmark models on three structurally distinct algorithmic domains requiring non-trivial intermediate execution states. We summarize the structural properties, canonical-state definitions, and train versus OOD evaluation splits for all three domains in Table 1.

Table 1: Structural properties and evaluation parameters for the algorithmic reasoning task splits.

Task Domain	**Canonical State Φ**	**Train Dataset Split**	**Joint OOD Test Split**
Folding	Quadrant layer map $\mu(S)$	$S \in \{4, 6\}, D \in [1, 4]$	$S = 8, D \in [5, 8]$
GridShift	Origin-per-cell permutation	$S \in \{4, 6\}, D \in [1, 4]$	$S = 8, D \in [5, 8]$
CLRS (8 Tasks)	Algorithmic Hints	$N = 16$ Nodes	$N = 64$ Nodes

1. **Folding.** A program consists of 1–4 valley or mountain folds applied to an $n \times n$ sheet discretized into four triangular quadrants per cell. The canonical state $\Phi(S_t)$ at step t is a stack-valued geometry over footprint positions $(r, c, q) \in P$, where each position holds a bottom-to-top sequence of element identifiers. A fold reflects a moved region across an axis onto a stationary region, concatenating stacks. The query asks which original cells are perforated by a punch through the final stack; answers are scored by exact set match. Out-of-distribution splits extend execution depth to 5–8 folds, increase sheet size to $n = 8$, or both.
2. **GridShift.** A program applies 1–4 cyclic row and column shifts to an $N \times N$ grid of distinct origin labels. The canonical state $\Phi(S_t)$ is a permutation matrix mapping each cell to its current origin label. The query asks which positions hold marked origins after execution. Unlike Folding, GridShift has no stacking or growing state; state size is invariant to program length. Out-of-distribution splits extend to 5–8 shifts, hold out transpose operations, or apply both jointly.
3. **CLRS pilot:** A textual conversion of the CLRS Algorithmic Reasoning Benchmark (Veličković & Blundell, 2021; Markeeva et al., 2024). We evaluate 8 classical textbook algorithms spanning sorting, graph traversal (BFS/DFS), and dynamic programming. Canonical states $\Phi(S)$ correspond to step-by-step intermediate execution hints.

4.2 Factorial Design

We employ a pre-registered 2×2 factorial across all domains, crossing positional encoding (absolute vs. RoPE) with state supervision (none vs. ORION). This yields four conditions: Baseline (absolute PE, no supervision), RoPE Only, ORION Only (absolute PE + L_Φ), and ORION + RoPE (full model). Each domain is evaluated on four splits: **In-Domain** ($L \leq L_{\text{train}}$, $D \leq D_{\text{train}}$), **Length Extrapolation** ($L_{\text{test}} > 3 \times L_{\text{train}}$ at fixed D), **Depth Extrapolation** ($D_{\text{test}} > 3 \times D_{\text{train}}$ at fixed L), and **Joint OOD** (both axes extrapolated simultaneously). The primary metric is exact match (EM): a prediction is correct only if every element of the answer set matches the ground truth exactly.

4.3 Evaluation

Models are trained with AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.98$, lr $= 3 \times 10^{-4}$, weight decay 0.01) for $6{,}000$ steps at batch size 128. We report mean $\pm$ standard deviation over 3 independent seeds (42–44) for primary comparisons, with paired bootstrap confidence intervals ($B = 20{,}000$ replicates) and Holm–Bonferroni correction (Holm, 1979) for multiple comparisons. The joint split is evaluated on seed-0 checkpoints with bootstrap intervals over $1{,}000$ test instances.

Baselines & Controls. We compare ORION against: (i) **Standard Transformer** (Vaswani et al., 2017) with Absolute Positional Encodings (APE), (ii) **Transformer + RoPE** (length-extrapolation baseline without state supervision), and (iii) **Negative Controls (N1–N5)** featuring randomized, hashed, or corrupted state target maps $\Phi(S_t)$ to isolate the necessity of canonical geometric alignment.

5 Result and Analysis

5.1 Primary Result: The Axes Fail Independently

We evaluate whether positional structure and canonical-state supervision address distinct failure modes in systematic generalization. The 2×2 factorial crosses positional encoding: absolute positional embeddings versus RoPE with state supervision—none versus ORION. Results are reported for Folding, GridShift, and a single-seed CLRS pilot. Table 2 shows a clear separation between the depth and state-size axes. On Folding, the baseline reaches 0.970 in-domain but falls to 0.021 at unseen execution depths. RoPE raises depth-extrapolation accuracy to 0.651, while ORION alone has little effect (0.026). Their combination reaches 0.667, the best depth-extrapolation result. To verify that the depth-extrapolation gains reflect genuine continuation of the computational trace rather than lucky final-answer guessing, we inspect the auxiliary read-out accuracy at each execution step. Figure 1 shows that ORION + RoPE maintains high read-out accuracy at depths 5–8 (e.g., 0.63–0.85 on FOLDING), whereas models with absolute positional encodings collapse as soon as

the depth exceeds the training horizon. The same ordering appears on GRIDSHIFT: depth extrapolation increases from 0.093 for the baseline to 0.575 with RoPE and 0.598 with ORION + RoPE. In contrast, the size-only axis is not solved by any condition. On FOLDING, all models remain close to the baseline value of 0.171, and the joint split remains substantially harder than the depth-only split. These results support the decomposition of OOD failure into distinct positional and execution-state axes. The CLRS experiment is a single-seed pilot. ORION + RoPE gives the strongest in-domain result (0.207), but no method solves the size extrapolation regime at $N = 64$. Accordingly, we use CLRS only as preliminary evidence and do not draw strong statistical conclusions from it.

Table 2: Out-of-distribution exact-match (EM) accuracy across algorithmic benchmarks. FOLDING and GRIDSHIFT models are trained on state size $S_{\text{train}} \leq 6$ and execution depth $D_{\text{train}} \leq 4$; the joint split tests $S = 8$ at depth 5–8. In-domain, size-only, and depth-only results are averaged over three random seeds and reported as mean ± standard deviation. The joint split uses seed-0 checkpoints and reports the half-width of a 95% bootstrap interval over 1,000 test instances. CLRS is a single-seed pilot over eight algorithms. ↑ indicates higher is better.

Benchmark & Regime	Baseline↑	RoPE Only↑	ORION Only↑	ORION + RoPE↑
FOLDING				
In-Domain (S, D)	0.970 ± 0.003	0.978 ± 0.002	0.978 ± 0.004	$\mathbf{0.985 \pm 0.003}$
Size Only ($S = 8$)	0.171 ± 0.009	0.191 ± 0.008	0.171 ± 0.008	$\mathbf{0.196 \pm 0.010}$
Depth Only ($2 \times D$)	0.021 ± 0.008	0.651 ± 0.016	0.026 ± 0.008	$\mathbf{0.667 \pm 0.018}$
Joint OOD ($S = 8, 2 \times D$)	0.004 ± 0.004	0.018 ± 0.008	0.004 ± 0.004	$\mathbf{0.022 \pm 0.009}$
GRIDSHIFT				
In-Domain (S, D)	0.952 ± 0.011	0.948 ± 0.012	$\mathbf{0.955 \pm 0.006}$	0.947 ± 0.018
Size Only ($S = 8$)	$\mathbf{0.314 \pm 0.005}$	0.309 ± 0.003	0.306 ± 0.003	0.308 ± 0.006
Depth Only ($2 \times D$)	0.093 ± 0.012	0.575 ± 0.009	0.063 ± 0.010	$\mathbf{0.598 \pm 0.038}$
Joint OOD ($S = 8, 2 \times D$)	0.045 ± 0.013	0.224 ± 0.025	0.038 ± 0.012	$\mathbf{0.225 \pm 0.025}$
CLRS				
In-Domain ($N = 16$)	0.168	0.168	0.203	**0.207**
Size Only ($N = 32$)	0.006	0.009	**0.016**	0.009
Size Only ($N = 64$)	0.002	0.003	0.001	**0.003**

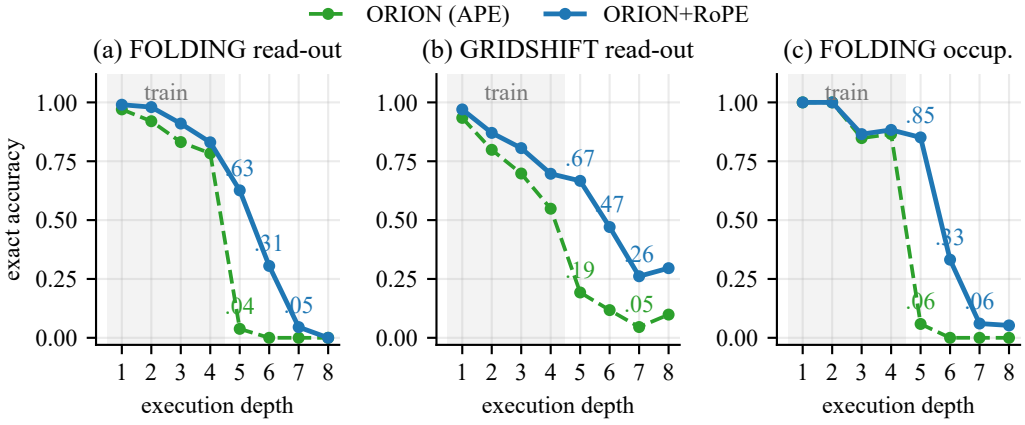


Figure 1: **ORION + RoPE keeps computing the canonical state beyond the training depth.** Exact accuracy of the canonical-state read-outs by execution depth (seed 0; shaded region: training depths). (a, b) The answer set read directly from the latent at the decision position, without autoregressive decoding, on FOLDING and GRIDSHIFT. (c) The occupancy map on FOLDING. With absolute positions the read-outs collapse at the first unseen depth (0.04, 0.19, 0.06); with RoPE they remain at 0.63, 0.67 and 0.85 at depth 5 and decay gradually, so the state is still being tracked where it was never supervised.

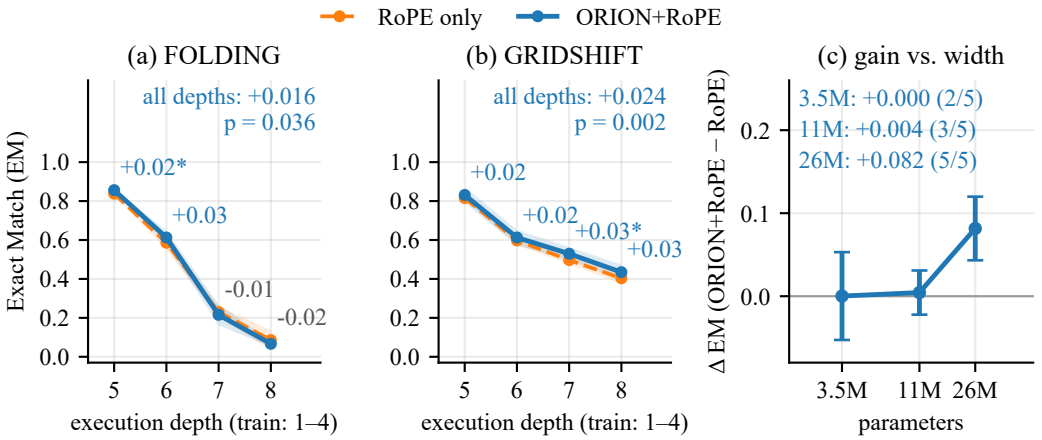


Figure 2: **ORION composes with RoPE on depth extrapolation.** (a,b) Exact match at execution depths 5–8, after training at depths 1–4, for RoPE alone and ORION + RoPE on FOLDING and GRIDSHIFT. Bands show ± one standard deviation over three seeds. Printed values give the paired-by-example difference at each depth; stars indicate $p < 0.05$. Across all held-out depths, ORION + RoPE improves over RoPE by $+0.016$ on FOLDING ($p = 0.036$) and $+0.024$ on GRIDSHIFT ($p = 0.002$). (c) The corresponding difference across model widths in the five-seed scaling ladder. The gains are $+0.000$, $+0.004$, and $+0.082$ at 3.5M, 11M, and 26M parameters, respectively; the largest model favors the combined condition in all five seeds.

5.2 COMPOSITION WITH ROPE IMPROVES DEPTH EXTRAPOLATION

The gain from canonical-state supervision appears when it is composed with a positional mechanism that supports long execution traces. Figure 2 compares RoPE alone with ORION + RoPE at held-out execution depths. On FOLDING, ORION + RoPE improves over RoPE alone by $+0.016 (p = 0.036)$, while the corresponding improvement on GRIDSHIFT is $+0.024$ ($p = 0.002$). Both tests are paired by example and pooled over held-out depths. The improvement also grows with model width. As shown in Figure 2c and Table 5, the gain is $+0.000$, $+0.004$, and $+0.082$ at 3.5M, 11.2M, and 26M parameters. At the largest width, the combined model outperforms RoPE alone in all five seeds. Widening the baseline alone does not solve the problem: baseline depth extrapolation increases only from 0.011 to 0.029 between the smallest and largest models.

5.3 NEGATIVE CONTROLS: CANONICAL ALIGNMENT IS NECESSARY

We next test whether the observed gain can be explained by additional parameters, additional optimization, or generic auxiliary regularization. Table 3 compares ORION against five negative controls under joint OOD extrapolation. The controls fail to reproduce the full-model result. On FOLDING, the full canonical target reaches 0.022, whereas all negative controls remain at or below 0.005. On GRIDSHIFT, the full model reaches 0.225, while the negative controls remain between 0.034 and 0.052. The N5 control is particularly informative: hashing preserves information about the state while destroying its canonical coordinate structure. Its failure indicates that the observed improvement is not explained by state information alone. The auxiliary signal must remain aligned with the exact canonical representation.

5.4 SENSITIVITY TO SUPERVISION AVAILABILITY

We next differentiate the fraction of execution steps that receive canonical supervision and the fraction of corrupted labels. The auxiliary loss weight (λ) is held fixed at its preregistered value; therefore, this experiment is not a sweep over auxiliary loss weight. Table 4 reports the sensitivity of OOD exact-match accuracy to the density of canonical supervision and to label corruption on FOLDING. Supervising every execution step is not necessary in this exploratory experiment: supervising one step per program yields comparable OOD EM. Label corruption degrades performance gradually,

Table 3: Ablation against negative controls on FOLDING and GRIDSHIFT under joint OOD extrapolation ($S = 8, 2 \times D$). Results use seed-0 checkpoints; $\pm$ denotes the half-width of a 95% bootstrap interval over 1,000 test instances. The controls retain the auxiliary-head architecture while changing the supervision target.

Control Condition	Loss Target $\mathcal{L}_{\text{aux}}$	FOLDING Joint EM ↑	GRIDSHIFT Joint EM ↑
Baseline + RoPE	None	0.018 ± 0.008	0.224 ± 0.025
N1: Compute-matched	None ($1.5\times$ steps)	0.004 ± 0.004	0.050 ± 0.014
N2: Random Targets	$\Phi_{\text{rand}}(S)$	0.002 ± 0.003	0.043 ± 0.013
N3: Inconsistent State	$\Phi(S) + \epsilon,\ \epsilon = 0.25$	0.005 ± 0.005	0.037 ± 0.012
N4: Shuffled Targets	$\Phi(S_{\pi(t)})$	0.002 ± 0.003	0.052 ± 0.014
N5: Hashed Statistics	$\text{hash}(\Phi(S))$	0.001 ± 0.002	0.034 ± 0.011
ORION + RoPE	**Canonical** $\Phi(S)$	$\mathbf{0.022 \pm 0.009}$	$\mathbf{0.225 \pm 0.025}$

with 25% corruption reducing OOD EM to 8.6%. These results suggest that the usefulness of the auxiliary signal depends more on its canonical alignment than on dense labeling, although this conclusion should be treated as exploratory because the experiment uses a single seed.

Table 4: Sensitivity to the availability of canonical supervision on FOLDING using an exploratory seed. The auxiliary loss weight (λ) was fixed at its preregistered value. The experiment varies the fraction of supervised execution steps and the fraction of corrupted labels.

Supervised Fraction	In-Domain EM (%)↑	OOD EM (%)↑	Train Hours ↓
1.00 (All steps)	95.4	9.9	0.13
0.50 of steps	95.3	9.6	0.12
0.25 of steps	95.1	10.1	0.12
0.10 of steps	95.1	10.1	0.12
One step per program	95.1	10.1	0.12
0.00 (No canonical loss)	95.2	9.5	0.06
5% labels corrupted	95.3	8.8	0.11
10% labels corrupted	95.2	9.3	0.07
25% labels corrupted	94.5	8.6	0.06

5.5 SCALING WITH MODEL CAPACITY

We test whether the benefit of canonical supervision persists as model capacity increases. The scaling ladder uses the three model sizes that were actually trained, with five seeds per condition. The gain from canonical-state supervision grows with model width. At 26M parameters, ORION + RoPE reaches 62.8% compared with 54.6% for RoPE alone, an absolute improvement of 8.2 percentage points (pp). The combined model is ahead in all five seeds at this width. Nevertheless, capacity alone is insufficient: the baseline moves only from 1.1% to 2.9% across the same width range.

Table 5: Scaling laws for RoPE versus ORION + RoPE on depth extrapolation in FOLDING. Values are EM percentages, reported as mean $\pm$ standard deviation over five seeds. The ladder uses a reduced training budget; therefore, its absolute values are lower than those in Table 2.

Model Scale	RoPE Only Depth OOD	ORION + RoPE Depth OOD	Absolute Delta
Small (3.5M)	41.9 ± 3.1	41.9 ± 2.7	+0.0 pp
Medium (11.2M)	54.0 ± 1.6	54.5 ± 2.3	+0.4 pp
Large (26M)	54.6 ± 3.6	62.8 ± 1.9	+8.2 pp

5.6 CROSS-DOMAIN TRANSFER

We test whether canonical-state supervision produces a domain-specific regularizer or a more general execution prior. Models are pretrained on one domain and adapted to the other under an equal

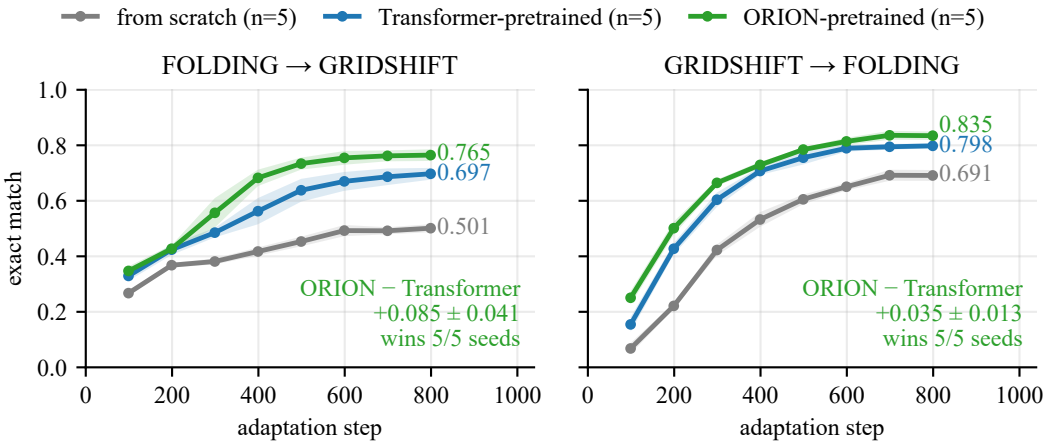


Figure 3: **Canonical-state pretraining transfers across domains.** Exact match during adaptation to the other domain under an equal 800-step budget, using five seeds. Models are initialized from scratch, from a Transformer pretrained on the source domain, or from an ORION model pretrained on the source domain. The ORION initialization improves target-domain held-out performance over Transformer pretraining by $+0.085 \pm 0.041$ for FOLDING→GRIDSHIFT and $+0.035 \pm 0.013$ for GRIDSHIFT→FOLDING. The improvement is positive in every seed in both directions.

800-step budget. Figure 3 plots the exact-match adaptation curves over the 800-step transfer budget. ORION-pretrained models consistently outperform both from-scratch and standard Transformer-pretrained initializations in every seed, with the performance gap widening during adaptation. ORION pretraining improves transfer in both directions. The gain is $+0.085 \pm 0.041$ from FOLDING to GRIDSHIFT and $+0.035 \pm 0.013$ in the reverse direction, with positive gains in every seed. This result is consistent with the interpretation that canonical-state supervision induces a reusable execution prior rather than merely regularizing one task-specific output head.

6 DISCUSSION

The empirical and mechanistic evaluations of ORION highlight that out-of-distribution (OOD) compositional generalization cannot be resolved only by architectural inductive biases or auxiliary supervision in isolation. Standard autoregressive transformers suffer from severe state-space drift during recursive reasoning steps (Dziri et al., 2023; Valmeekam et al., 2023). While positional schemes like Rotary Position Embeddings (RoPE) effectively regularize distance metrics across long token sequences (Su et al., 2024), they remain agnostic to the latent computation path. Conversely, supervising intermediate canonical representations $\Phi(S)$ stabilizes the algorithmic execution manifold, but relies on positional coherence to prevent representation blurring over long horizons. Diagnostic visualization confirms this interpretation. Figure 4 projects the final-layer hidden states of executed operations onto their leading principal components; the baseline drifts by 1.18 in-distribution spreads per step with no coherent direction, whereas ORION + RoPE follows a structured flow of only 0.44 spreads per step. This indicates that canonical supervision confines the latent trajectory to the true execution manifold rather than artificially compressing representational subspaces.

The orthogonality of these failure modes reflects a deeper representational dissociation. RoPE regularizes the metric structure of query-key affinities in positional space, thereby preserving attention coherence over extended token sequences (Su et al., 2024), yet it imposes no constraint on the latent evolution of algorithmic states. Conversely, ORION anchors the execution manifold by supervising intermediate canonical representations (Nye et al., 2021; Schwarzschild et al., 2021; Veličković & Blundell, 2021), but it cannot compensate for positional incoherence that corrupts the read-out of those states at unseen sequence positions. Because these mechanisms operate on non-competing geometric degrees of freedom, their composition is additive rather than redundant. This suggests a design principle for long-horizon reasoning: rather than entangling positional indexing and state tracking in a single residual stream, architectures may benefit from explicitly factoring these subsys-

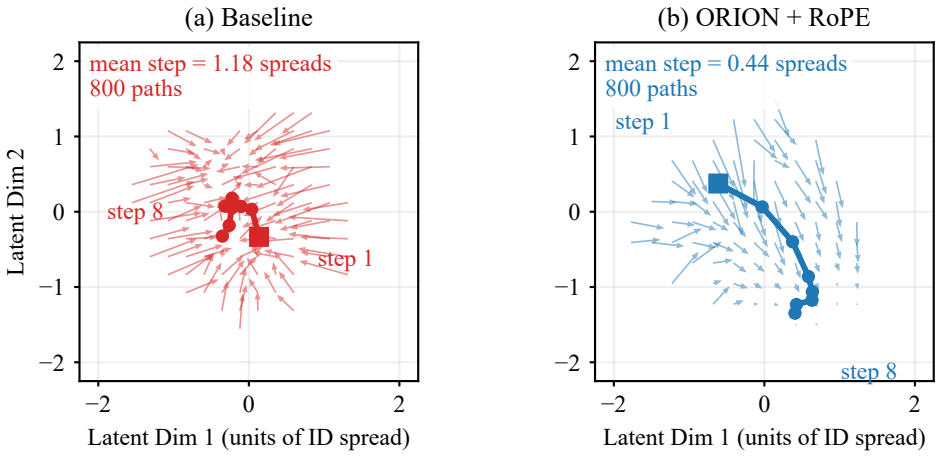


Figure 4: **Canonical supervision confines the latent trajectory.** The final-layer hidden state at each executed operation is one point per step, projected on the two leading principal components of the in-distribution states and expressed in units of their spread. Arrows: mean step-to-step displacement of 800 paths through each cell; bold path: mean position from step 1 (square) to step 8. The baseline moves 1.18 spreads per step with no coherent direction; ORION + RoPE moves 0.44 along a coherent field. The effect comes from the supervision, not the positional encoding (0.45 for ORION without RoPE, 1.36 for RoPE alone).

tems, with positional schemes governing token-level attention geometry and auxiliary supervision constraining the computational trajectory. The persistent difficulty of joint length-depth extrapolation arises precisely from the interaction between these axes; unseen state-size tokens introduce embedding extrapolation errors that compound with depth-dependent drift (Wu et al., 2024; Merrill & Sabharwal, 2024), so neither positional regularization nor state supervision alone can repair the compounding failure. Extending this decomposition to large language models suggests a diagnostic for chain-of-thought breakdowns (Wei et al., 2022): distinguishing positional attention collapse from intermediate state drift is essential, because the former demands architectural interventions on relative position encodings while the latter requires execution-aware supervision or self-correction mechanisms.

However, key limitations remain. While state supervision provides strong length extrapolation ($2 \times D$), joint out-of-distribution generalization across both state size ($S = 8$) and depth remains challenging due to the compounding complexity of higher-dimensional search spaces. Furthermore, the method relies on access to exact canonical state targets ($\Phi(S)$) during training, which may not be readily available in unannotated or non-algorithmic environments. Finally, while performance scales favorably with parameter width, the necessity of auxiliary intermediate supervision steps introduces non-trivial computational overhead during data generation and pretraining. Future work should explore weaker supervision signals and self-supervised state abstractions to mitigate these constraints.

7 Conclusion

We presented **ORION**, an intermediate state-supervision framework designed to resolve depth-dependent representation drift in autoregressive transformers. We showed that growth in execution depth and growth in state size are separate modes of failure, and measured each. Combined with rotary position embeddings, ORION is the strongest condition on the depth axis in both domains, with a margin over RoPE alone that is significant under a paired test and grows to $+0.082$ at 26M parameters; its canonical-state read-outs survive beyond the training depth, its latent trajectory is confined to a coherent flow, and its pretraining transfers across domains. The state-size axis is not repaired by any condition we tested and is the open problem this decomposition exposes. Future work will explore self-supervised estimation of canonical state mappings to remove the requirement for explicit execution trace supervision.

AI USE STATEMENT

Large Language Models were used exclusively for grammar editing, prose polishing, and LaTeX formatting assistance. All research formulations, experimental implementations, theoretical analyses, and final interpretations remain the sole responsibility of the authors.

ETHICS STATEMENT

This work evaluates algorithmic reasoning using public, synthetic benchmarks with no human subject data or privacy concerns. Improved length extrapolation carries broad benefits for automated system reliability without presenting dual-use risks or direct societal harm.

REPRODUCIBILITY STATEMENT

All architectural parameters, experimental setups, and training configurations are detailed in Section 4 and Appendix B. Upon publication, complete source code, synthetic dataset generators, and seed-initialized evaluation scripts will be released in an open-source repository to ensure full replication.

AUTHOR CONTRIBUTIONS

The primary author conceptualized the state supervision framework, designed the methodology, and implemented the codebase. Co-author(s) drafted the manuscript, executed the benchmark evaluations, performed ablation studies, and edited the final paper.

ACKNOWLEDGMENTS

The authors thank the anonymous reviewers for their constructive feedback. Infrastructure support were availed from the primary institution.

REFERENCES

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

Cem Anil, Yuhuai Wu, Anders Andreassen, Aitor Lewkowycz, Vedant Misra, Vinay Ramasesh, Ambrose Slone, Guy Gur-Ari, Ethan Dyer, and Behnam Neyshabur. Exploring length generalization in large language models. *Advances in Neural Information Processing Systems*, 35:38546–38556, 2022.

Arpit Bansal, Avi Schwarzschild, Michael Zhou, Tom Goldstein, and Micah Goldblum. End-to-end algorithm synthesis with recurrent neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 8732–8743, 2022.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers. In *International Conference on Learning Representations (ICLR)*, 2019.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, et al. Faith and fate: Limits of transformers on compositionality. *Advances in neural information processing systems*, 36:70293–70332, 2023.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1(1):12, 2021.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal abstractions of neural networks. *Advances in neural information processing systems*, 34:9574–9586, 2021.

Angeliki Giannou, Shashank Rajput, Jy-yong Sohn, Kangwook Lee, Jason D Lee, and Dimitris Papailiopoulos. Looped transformers as programmable computers. In *International Conference on Machine Learning*, pp. 11398–11442. PMLR, 2023.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pp. 65–70, 1979.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pp. 3519–3529. PMlR, 2019.

Larisa Markeeva, Sean McLeish, Borja Ibarz, Wilfried Bounsi, Olga Kozlova, Alex Vitvitskyi, Charles Blundell, Tom Goldstein, Avi Schwarzschild, and Petar Veličković. The clrs-text algorithmic reasoning language benchmark. *arXiv preprint arXiv:2406.04229*, 2024.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372, 2022.

William Merrill and Ashish Sabharwal. The expressive power of transformers with chain of thought. In *International Conference on Learning Representations*, volume 2024, pp. 7690–7706, 2024.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. *arXiv preprint arXiv:2301.05217*, 2023.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*, 2021.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. *arXiv preprint arXiv:2311.03658*, 2023.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. Yarn: Efficient context window extension of large language models. In *International Conference on Learning Representations*, volume 2024, pp. 31932–31951, 2024.

Ofir Press, Noah A Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*, 2021.

Avi Schwarzschild, Eitan Borgnia, Arjun Gupta, Furong Huang, Uzi Vishkin, Micah Goldblum, and Tom Goldstein. Can you learn an algorithm? generalizing from easy to hard problems with recurrent networks. *Advances in Neural Information Processing Systems*, 34:6695–6706, 2021.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 464–468, 2018.

Kensen Shi, Joey Hong, Yinlin Deng, Pengcheng Yin, Manzil Zaheer, and Charles Sutton. Exedec: Execution decomposition for compositional generalization in neural program synthesis. In *International Conference on Learning Representations*, volume 2024, pp. 45106–45135, 2024.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Yutao Sun, Li Dong, Barun Patra, Shuming Ma, Shaohan Huang, Alon Benhaim, Vishrav Chaudhary, Xia Song, and Furu Wei. A length-extrapolatable transformer. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14590–14604, 2023.

Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. On the planning abilities of large language models-a critical investigation. *Advances in neural information processing systems*, 36:75993–76005, 2023.

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

Petar Veličković and Charles Blundell. Neural algorithmic reasoning. *Patterns*, 2(7):100273, 2021.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. In *Proceedings of the 2024 conference of the north american chapter of the association for computational linguistics: Human language technologies (volume 1: Long papers)*, pp. 1819–1862, 2024.

Keyulu Xu, Jingling Li, Mozhi Zhang, Simon S Du, Ken-ichi Kawarabayashi, and Stefanie Jegelka. What can neural networks reason about? *arXiv preprint arXiv:1905.13211*, 2019.

Hattie Zhou, Arwen Bradley, Etai Littwin, Noam Razin, Omid Saremi, Joshua Susskind, Samy Bengio, and Preetum Nakkiran. What algorithms can transformers learn? a study in length generalization. In *International Conference on Learning Representations*, volume 2024, pp. 15898–15926, 2024.

A Domains, Architecture, and Mechanism

This appendix gives the pictures behind the formal description in §4.1 and §3: what an execution trace looks like in each domain, how the supervision is wired into the model, and what we predicted the supervision would do to the representation.

A.1 Execution Traces

Figure 5 follows one Folding program. The sheet starts flat; each fold reflects a moved region onto a stationary one and concatenates their stacks, so the state after t operations is a stack-valued geometry over footprint positions, not a picture. The query punches through the final stack and asks which original cells it perforates, which is why the answer depends on the whole execution history rather than on the final silhouette.

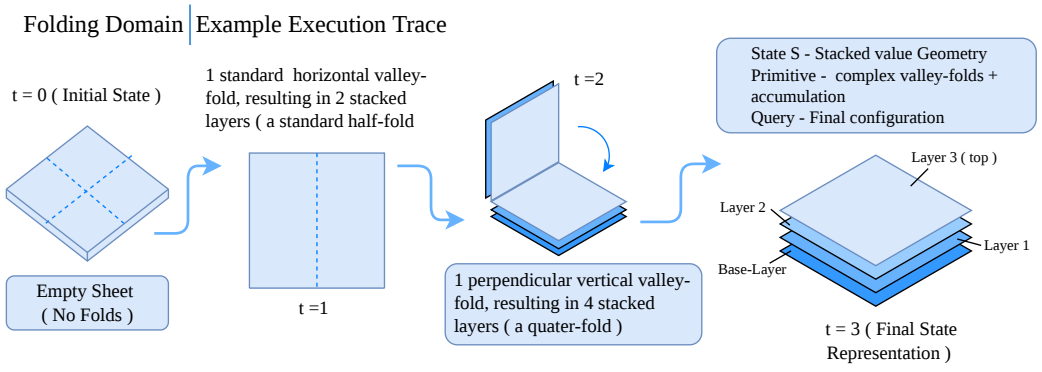


Figure 5: An execution trace in the Folding domain. Each operation reflects part of the sheet onto the rest and concatenates the stacks, so layer count grows with depth while the footprint shrinks. The canonical state $\Phi(S_t)$ is the stack-valued geometry at each step; the query is a punch through the final stack, scored by exact set match over the original cells it perforates.

Figure 6 follows one GridShift program. Here the state is a permutation of origin labels over a grid of fixed extent: cyclic row and column shifts move labels around without ever stacking them. The

contrast with FOLDING is the point of including both domains — the same algebra (Appendix B) arises from reflection-and-concatenation in one domain and from a group action in the other, so a result that holds in both is not an artefact of how FOLDING happens to be constructed.

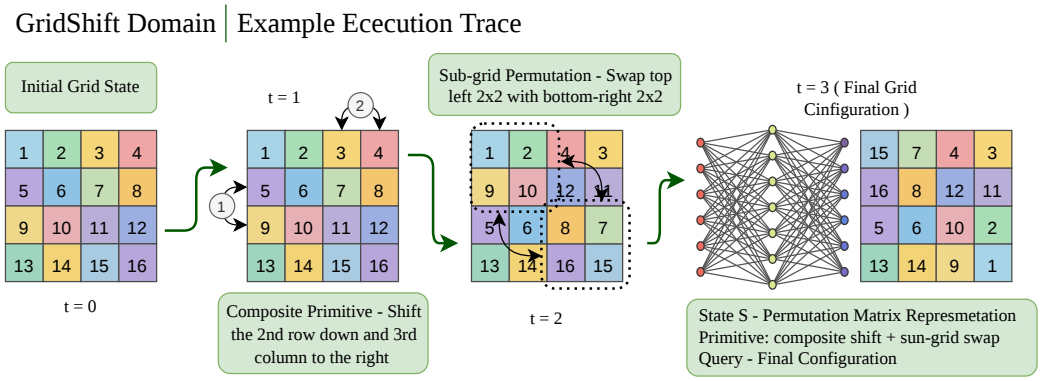


Figure 6: An execution trace in the GRIDSHIFT domain. Cyclic row and column shifts permute origin labels over a grid whose extent never changes, so unlike FOLDING the state size is independent of program length. The canonical state is the origin-per-cell permutation and the query asks which positions hold marked origins after execution.

A.2 ARCHITECTURE

Figure 7 shows how the supervision attaches to the backbone. A single shared latent $\mathbf{h}$ feeds five read-out heads — spatial, occupancy, boundary, transition, and answer — and the composite loss is the language-modelling term plus the weighted head terms. Two details matter for interpreting the results. First, the answer is decoded from the same latent as the state, as *one probe among several* rather than as a privileged output; this is what makes the read-out measurements of Figure 1 meaningful, since they interrogate the latent directly without autoregressive decoding. Second, the transition head is supervised against an operator traced from the simulator, so the constraint the model is trained on and the operator the generator uses cannot drift apart (Theorem 2). The same wiring runs unchanged on both domains, which is what the contrastive term across them exploits.

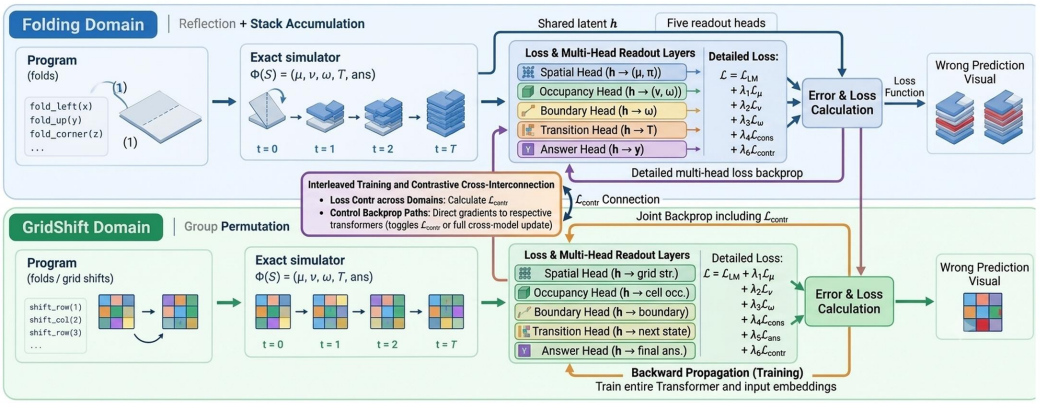


Figure 7: The ORION architecture on both domains. An exact simulator emits the canonical state $\Phi(S) = (\mu, \nu, \omega, T, \text{ans})$ at every execution step; five read-out heads decode it from one shared latent, and their losses are added to the language-modelling objective. The answer head is one probe among several rather than the privileged output. The identical wiring runs on FOLDING and GRIDSHIFT, which is what lets the contrastive term act across domains.

A.3 Predicted vs Actual Mechanism

Figure 8 states the hypothesis this study preregistered: that entangling execution with decoding in a single residual stream is what makes a transformer fragile out of distribution, and that canonical-state supervision would separate the two into distinct subspaces.

The measurements do not support it, and we record that here rather than quietly dropping the diagram. If supervision disentangled execution from decoding into a lower-dimensional subspace, the effective dimensionality of the representation should fall; it rises (Table 9). If the recovered subspace carried the execution, ablating it should cost more accuracy than ablating a random subspace of equal rank; it costs the same (Appendix C.8). And if the supervised representation were reorganised, cross-model CKA should drop; it stays between 0.94 and 0.99 at every layer.

What survives is a weaker and more specific claim, which is the one the paper makes: supervision does not rebuild the representation, it changes how the model *moves through* it. The latent trajectory advances 0.44 rather than 1.18 spreads per step along a coherent field (Figure 4), and the canonical state continues to be decodable past the training depth when a positional encoding reaches those positions (Figure 1). Disentanglement was the hypothesis; confinement of the trajectory is the finding.

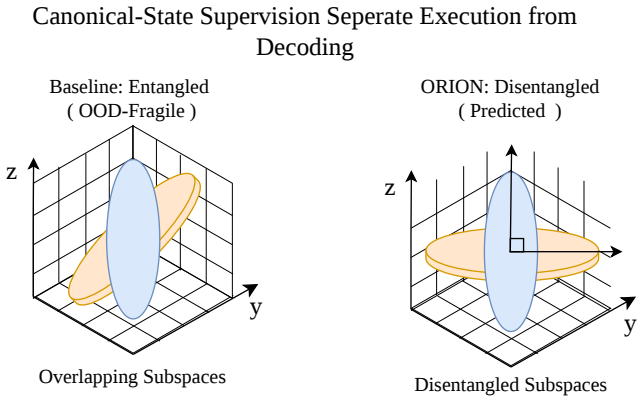


Figure 8: The preregistered hypothesis, retained here as the prediction it was. Left: execution and decoding share overlapping directions in the residual stream, which is the state we expected to make extrapolation fragile. Right: the predicted effect of canonical-state supervision, separating them into distinct subspaces. Our diagnostics refute this specific mechanism — effective dimensionality rises rather than falls (Table 9) and the probe subspace is no more causally necessary than a random one (Appendix C.8) — while supporting the weaker claim that supervision confines the latent trajectory (Figure 4).

B The State Algebra

The theory below is what licenses the auxiliary objectives rather than decoration for them. Theorem 1 says the state is conserved, so a layer map is complete bookkeeping of it; Theorem 2 says the state evolves *linearly* under an operator we can compute exactly, which is what makes a *label-free* consistency loss possible; Theorem 3 says syntactically different programs can be provably equivalent, which is what makes a contrastive loss sound rather than heuristic; and Theorem 4 says which statistic deserves the name *canonical*, which is what control N5 operationalises. All four are numerically verified on 220 random programs per domain by the released test suite.

B.1 Conservation

Theorem 1 (Conservation)**.** *Every fold is a bijection on the element set E: each element occupies exactly one stack exactly once, before and after any fold.*

Proof. A fold partitions the footprint into a stationary region and a moved region (straddled diagonal cells split at quadrant granularity; the quadrant set is closed under the reflections used). Stationary stacks are copied unchanged; each moved stack is mapped by a reflection ρ — an isometry restricted to the grid, hence injective — and concatenated (reversed) onto its target. Concatenation neither duplicates nor deletes elements, and injectivity prevents two moved positions sharing a target. □

What this buys. Nothing is created or destroyed, so the stack *lengths* $\mu(S)$ are a faithful summary of where material went. That is why μ is a legitimate supervision target rather than a lossy proxy.

B.2 Linear State Transition

Theorem 2 (Linear state transition)**.** *There exist geometry-only index maps A, B with $\mu(F(S))_i = \mu(S)_{A(i)} + \mu(S)_{B(i)}$; equivalently $\mu(F(S)) = T_F \mu(S)$ for a fixed $T_F \in \{0,1\}^{|P|\times|P|}$ with at most two ones per row.*

Proof. The stack at target i after the fold is the concatenation of the stationary stack previously at i (or nothing) and the reversal of the stack at the unique moved source $\rho^{-1}(i)$ (or nothing). Set $A(i) = i$ if stationary else $\varnothing$, and $B(i) = \rho^{-1}(i)$ if in the moved region else $\varnothing$; both depend only on geometry. Lengths add under concatenation, and a single reflection gives each target at most one moved source. □

What this buys. Because T_F depends only on the program's geometry and not on the data, the model can be *constrained* to satisfy $\hat{\mu}_{t+1} = T_t \hat{\mu}_t$ without ever seeing a label for step $t+1$. One supervised first step then pins the entire trajectory. This is the consistency term of the composite objective, and it is the component that most directly encodes "execute the program" rather than "predict the answer". The implementation *traces* A and B through the simulator, so the operator used by the objective cannot drift from the operator used by the generator.

B.3 Commutation and Content Equivalence

Theorem 3 (Commutation / content equivalence)**.** *Axis-orthogonal folds commute on content: $\sigma(F_H(F_V(S))) = \sigma(F_V(F_H(S)))$; content-equivalent states give identical answers and layer maps.*

Proof. A vertical fold sends $(r, c, q) \mapsto (r, \phi(c), m_V(q))$, independently of r, and a horizontal fold acts only on rows. Both orders send (r, c, q) to $(\psi(r), \phi(c), m_H m_V(q))$ with $m_H m_V = m_V m_H$, since these are involutions on disjoint quadrant pairs, so the content maps agree. The answer and layer map are functions of the content map by Theorem 4, hence coincide. □

What this buys. We can *generate* pairs of distinct programs guaranteed to share an answer and pull their representations together with a contrastive term, knowing the supervision is exactly correct rather than assumed. It also gives a free evaluation axis: a model respecting the algebra should give equivalent programs the same prediction, which is the *equivalent programs* row of Table 6.

B.4 Sufficiency and Identifiability

Definition 1 (Cell-content map)**.** $\tau(S) : (r, c) \mapsto \{\lfloor e/4 \rfloor : e \in \bigcup_q \sigma(S)(r, c, q)\}$ is the set of original cells present at each footprint position. The *answer map* is $\mathcal{A}_S : Q \mapsto \mathrm{ans}(S, Q)$.

Theorem 4 (Sufficiency and identifiability)**.** ***(a)*** *$\mathrm{ans}(S, Q) = \bigcup_{(r,c)\in Q} \tau(S)(r, c)$, independent of the program.* ***(b)*** *$\tau(S)$ is recoverable from single-cell answers, so any statistic Ψ sufficient for the answer map refines τ; hence τ is the coarsest sufficient statistic, unique up to relabelling.*

Proof. (a) A single punch at (r, c) returns $\tau(S)(r, c)$ by definition, and multi-punch answers are the union; both reference S only through τ. (b) Recoverability is (a) with $Q = \{(r, c)\}$. If Ψ is sufficient, $\mathcal{A}_S$ is a function of $\Psi(S)$; composing with recovery, τ is a function of Ψ. Any statistic coarser than τ collapses states with different answers, so is not sufficient. □

Table 6: Three further preregistered evaluation axes on the seed set of Table 2 (mean ± s.d. over 3 seeds, exact match). *Held-out operation type* withholds diagonal folds (FOLDING) and the transpose (GRIDSHIFT) from training. *Held-out query type* withholds the multi-cell punch. *Equivalent programs* scores agreement on pairs of syntactically different programs that Theorem 3 certifies to have the same answer; it is an in-distribution consistency check, not an extrapolation split.

Evaluation axis	Baseline (APE)	RoPE only	ORION only	ORION + RoPE
FOLDING				
Held-out operation type	0.151 ± 0.012	0.150 ± 0.018	$\mathbf{0.175 \pm 0.023}$	0.160 ± 0.016
Held-out query type	0.001 ± 0.001	$\mathbf{0.002 \pm 0.002}$	0.000 ± 0.001	0.001 ± 0.001
Equivalent programs (Thm. 3)	0.962 ± 0.004	0.966 ± 0.002	0.966 ± 0.004	$\mathbf{0.973 \pm 0.006}$
GRIDSHIFT				
Held-out operation type	0.079 ± 0.002	0.081 ± 0.003	0.081 ± 0.001	$\mathbf{0.085 \pm 0.006}$
Held-out query type	$\mathbf{0.000 \pm 0.000}$	$\mathbf{0.000 \pm 0.000}$	$\mathbf{0.000 \pm 0.000}$	$\mathbf{0.000 \pm 0.000}$
Equivalent programs (Thm. 3)	$\mathbf{0.941 \pm 0.010}$	0.929 ± 0.016	0.937 ± 0.007	0.936 ± 0.036

Remark 1 (This is what grounds control N5). τ is *minimal sufficient*. ORION's Φ refines τ in the geometric coordinates where the transition is linear (Theorem 2); any bijective relabelling of Φ is *equally sufficient* but destroys that structure. Control N5 supervises exactly such a sufficient-but-non-canonical statistic. If N5 matches ORION, then canonicality is irrelevant and only exactness mattered — a clean, preregistered way to lose. On the factorial splits (Table 3) N5 does match ORION at equal positional encoding, which is why §5.3 reports the controls as bounding rather than isolating the effect.

B.5 THE GRIDSHIFT ANALOGUES

The same three structural properties hold in GRIDSHIFT, for entirely different reasons, which is why it is a fair second test rather than a restatement. Conservation holds because a shift is a bijection on cells, with no stacking needed. Linear transition holds because the state evolves as $s_{t+1} = T_t s_t$ with T_t a $0/1$ permutation matrix — the same operator *form* as T_F, but arising from a group action rather than from reflection and concatenation. Commutation holds because shifts on disjoint rows or columns commute. The generator emits the identical data schema, so the model, losses, trainer and evaluator run on it unchanged.

B.6 A NOTE ON POSITIONAL COMMUTATIVITY

Since $\mathbf{R}_{\Theta,m}$ acts as a block-diagonal rotation on two-dimensional subspaces, it preserves inner products of canonical state features under a global index shift: $\langle \mathbf{R}_{\Theta,m}\Phi(S_t), \mathbf{R}_{\Theta,m+k}\Phi(S_t)\rangle = \langle \Phi(S_t), \mathbf{R}_{\Theta,k}\Phi(S_t)\rangle$. This is the standard relative-position property of RoPE. We record it because it motivated the design, but we emphasise that it does *not* establish that the two mechanisms are orthogonal in any functional sense — our measurements show them to be interdependent, since canonical supervision is inert without a positional encoding that reaches the unseen positions.

C ADDITIONAL EXPERIMENTS

This appendix reports the experimental phases that did not fit the main text, including those whose outcome was neutral or negative. Every table here is generated directly from the released run artefacts.

C.1 ADDITIONAL EVALUATION AXES

Beyond the depth and size axes, the preregistration specified three further splits (Table 6). Holding out an *operation type* (diagonal folds in FOLDING, the transpose in GRIDSHIFT) is hard for every condition, and it is the one axis where canonical supervision without RoPE is the best condition on FOLDING (0.175), though the margin lies within the seed spread. Holding out the *query type* defeats all conditions on both domains: this is a complete null, and we report it rather than omit it.

Agreement on *provably equivalent programs* — pairs certified by Theorem 3 to share an answer — is high for every condition, so respecting the algebra in-distribution is not what separates the models.

C.2 Alternative Sequence Architectures

Any claim that positional structure drives depth extrapolation has to contend with backbones that carry sequence priors of their own. We therefore trained a state-space model, Hyena, and RWKV at a matched budget (Table 7). RWKV is the strongest non-rotary backbone on the depth axis in both domains, and on GRIDSHIFT it nearly doubles the APE transformer (0.175 against 0.093), which is consistent with recurrence supplying part of what rotary embeddings supply. None of the three approaches the range that RoPE reaches, and two of them do not fit the training distribution well enough for their extrapolation numbers to be interpretable.

Table 7: Alternative sequence backbones at a matched training budget (6,000 steps, 3 seeds, mean $\pm$ s.d.). The recurrent and long-convolution models were run because they carry sequence priors of their own and are therefore confounds for any claim about positional structure. The state-space model does not fit the training distribution in either domain and Hyena fits only FOLDING; RWKV fits both and is the strongest non-rotary backbone on the depth axis (0.099 and 0.175), but none of the three comes near what rotary embeddings reach (0.651 and 0.575).

Backbone	**In-domain**	**Depth** ($2 \times D$)	**Size** ($S = 8$)
FOLDING			
SSM (S4-style)	0.349 ± 0.021	0.004 ± 0.002	0.098 ± 0.009
Hyena	0.902 ± 0.003	0.022 ± 0.004	0.157 ± 0.008
RWKV	0.963 ± 0.005	0.099 ± 0.012	0.160 ± 0.004
Transformer (APE)	0.970 ± 0.003	0.021 ± 0.008	0.171 ± 0.009
ORION + RoPE	0.985 ± 0.003	0.667 ± 0.018	0.196 ± 0.010
GRIDSHIFT			
SSM (S4-style)	0.362 ± 0.012	0.034 ± 0.017	0.205 ± 0.004
Hyena	0.549 ± 0.016	0.171 ± 0.005	0.197 ± 0.007
RWKV	0.835 ± 0.016	0.175 ± 0.017	0.291 ± 0.009
Transformer (APE)	0.952 ± 0.011	0.093 ± 0.012	0.314 ± 0.005
ORION + RoPE	0.947 ± 0.018	0.598 ± 0.038	0.308 ± 0.006

C.3 Robustness to Input Corruption

Table 8 perturbs the program at inference. Canonical supervision produces marginally higher exact match at every noise level and an identical score when an operation is dropped, with differences of 1–2 points that lie within single-seed variation. We take this as a null result: whatever stabilisation the latent-flow analysis detects (Figure 4) does not surface as robustness in the decoded answer.

Table 8: Robustness to input corruption on FOLDING (seed 0, in-distribution test set). *Token noise* replaces a fraction p of program tokens at random. *Dropped operation* removes one operation from the program and scores against the answer for the shortened program. Canonical supervision leaves both curves essentially unchanged, so the robustness it confers is not visible at the level of the decoded answer.

	Baseline		**ORION**	
Corruption	EM ↑	F1↑	EM ↑	F1↑
$p = 0.00$	0.975	0.981	0.980	0.985
$p = 0.02$	0.853	0.880	0.868	0.892
$p = 0.05$	0.718	0.782	0.740	0.790
$p = 0.10$	0.562	0.645	0.575	0.666
Dropped operation ($n = 319$)	0.994	0.994	0.994	0.994

C.4 Representation Geometry

We preregistered the hypothesis that canonical supervision compresses the execution subspace into fewer effective dimensions. Table 9 refutes it. Every variance-based measure moves in the opposite direction, and layer-wise CKA shows that the supervised model's representation stays close to the unsupervised model's throughout the stack (Figure 9). The mechanism is not compression; it is the traversal of the representation, which is what Figure 4 measures.

Table 9: Representation geometry of the final layer on FOLDING (seed 0, $n = 600$ in-distribution prompts). The preregistered hypothesis was that canonical supervision compresses the execution subspace. It does not: the participation ratio, the spectral entropy and the number of dimensions needed for 90–95% of the variance all *rise* under supervision. The TwoNN estimator is the one measure that moves the other way, so we report the evidence as mixed in direction but uniform in conclusion — no measure supports compression. Layer-wise linear CKA between the supervised and unsupervised models stays high throughout, so the supervised representation is a nearby one rather than a different one.

Measure	**Baseline**	**ORION**
Participation ratio (of 384 dims)	10.4 [9.8, 10.6]	14.3 [13.3, 14.6]
TwoNN intrinsic dimension	12.2	10.1
Spectral entropy (nats)	2.97	3.31
Dimensions for 90% variance	19	26
Dimensions for 95% variance	31	39
Cross-model CKA, FOLDING	0.96–0.99 across layers	
Cross-model CKA, GRIDSHIFT	0.90–0.97 across layers	

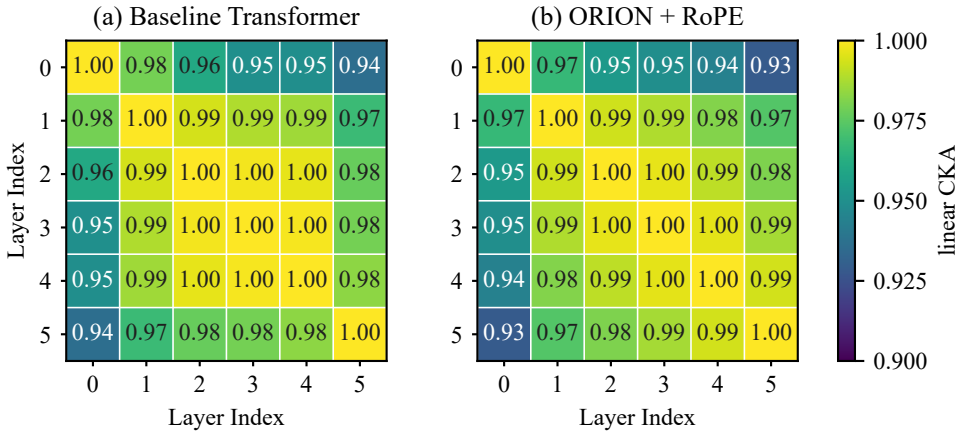


Figure 9: Layer-to-layer linear CKA within each model on FOLDING (seed 0, 384 in-distribution prompts). Both models show the same smooth block structure, with adjacent layers above 0.97 and the embedding layer the most distinct. Canonical supervision does not reorganise the layer hierarchy. Note the colour scale starts at 0.90; on an absolute scale the two panels are nearly identical, which is the point.

C.5 Degradation Curves and the Joint Surface

Figure 10 resolves the aggregate numbers of Table 2 into a degradation curve and a grid over both axes. Three features are worth noting.

First, the two rotary conditions separate from the two absolute ones immediately past the training boundary, and they separate completely: at $1.25\times$ the training depth, ORION + RoPE and RoPE hold 0.854 and 0.835 while ORION and the baseline have already fallen to 0.043 and 0.033. The

failure is a cliff, not a slope, and which side of it a model lands on is decided by the positional encoding.

Second, between the two rotary conditions the margin is real but small and not uniform: ORION + RoPE leads RoPE at $1.25\times$, $1.5\times$ and $1.75\times$ (0.854 against 0.835, 0.634 against 0.602, 0.221 against 0.180) and trails it at $2\times$ (0.067 against 0.086), where both are near zero. What canonical supervision buys is a shift in where the cliff falls, not its removal.

Third, the size axis is the horizontal axis of the surface, and its rightmost column is flat in a way no amount of depth margin repairs: every cell drops to 0.01–0.38 as soon as $S/S_{\text{train}} = 1.33$, even at a quarter of the training depth. We do not plot the size axis as a separate curve because all four conditions coincide on it (0.195, 0.190, 0.169, 0.170 at the held-out size), which is consistent with the untrained embedding for that size token rather than with a graded failure of generalisation.

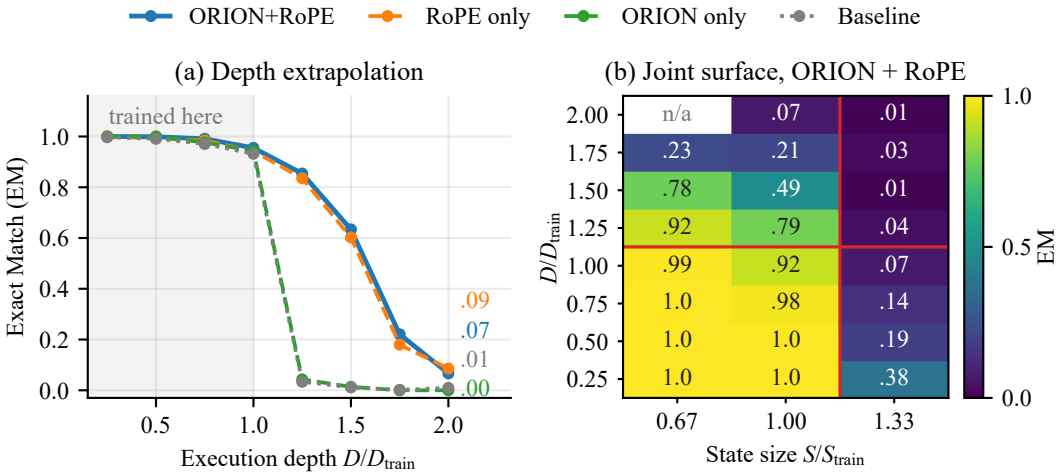


Figure 10: Degradation on FOLDING, from the reduced-budget surface sweep. (a) Exact match against execution depth at the training state sizes; the shaded band is the trained range and the printed values are the endpoints at $2\times$. The rotary conditions degrade gradually while the two absolute-position conditions fall off a cliff at the first unseen depth. (b) The joint surface for ORION + RoPE over both axes, every cell labelled with exact match; red lines mark the edges of the training region and *n/a* marks a combination the generator cannot produce. Reading up the left columns shows depth costing accuracy gradually; reading across to the right column shows the size axis costing all of it at once, at every depth. The size axis is not drawn as a separate curve because all four conditions coincide on it (0.195, 0.190, 0.169 and 0.170 at $S/S_{\text{train}} = 1.33$).

C.6 CLRS PER-ALGORITHM BREAKDOWN

The aggregate CLRS numbers in Table 2 are dominated by two of the eight algorithms. Figure 11 separates them. Canonical supervision changes exactly one algorithm materially — `kmp_matcher`, from 0.25 to 0.54 in-distribution and from 0.18 to 0.35 out-of-distribution — while the remaining seven are unchanged to within a point. String matching is the task in this suite whose hints most closely resemble a canonical execution state, which makes the result suggestive, but it is one algorithm at one seed and we draw no conclusion from it.

C.7 ERROR TAXONOMY

Table 10 classifies the failures. Beyond the training depth, malformed outputs are rare (11 and 3 of roughly 970 errors): the models emit well-formed answer sets that are simply wrong, most often by omitting cells or by being disjoint from the truth. This is the signature of a state that has drifted rather than of a decoder that has broken, which is the premise the paper is built on.

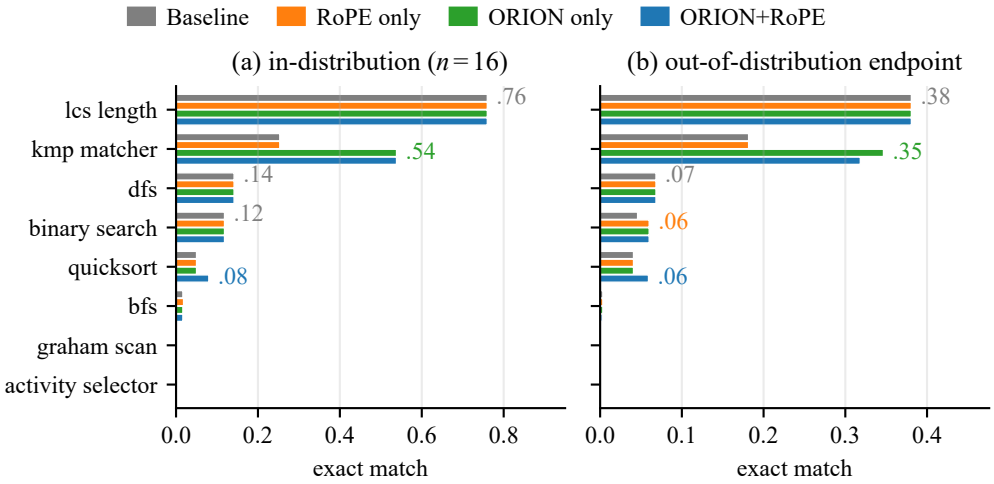


Figure 11: Per-algorithm exact match on the CLRS pilot (single seed). (a) In-distribution ($n = 16$). (b) The preregistered out-of-distribution endpoint, the mean of the length and structural axes. Only `kmp_matcher` separates the conditions; `activity_selector` and `graham_scan` are at zero for every condition.

Table 10: Error taxonomy on FOLDING (seed 0): counts of each failure mode among incorrect predictions. Beyond the training depth the model almost never emits a malformed or reflected answer: the failures are well-formed answer sets that miss cells or are disjoint from the truth, which is the signature of a state that has drifted rather than of a decoder that has broken.

	Baseline		**ORION**	
Failure mode	in-domain	depth $2 \times D$	in-domain	depth $2 \times D$
Final operation not applied	9	92	3	107
First operation not applied	0	3	0	5
Query reflected	2	4	2	5
Answer misses cells	5	369	4	352
Answer adds cells	3	14	1	14
Partial overlap	6	195	6	139
Disjoint from truth	8	282	5	341
Malformed output	0	11	0	3
Total errors	33	970	21	966

C.8 PREREGISTERED DIAGNOSTICS THAT DID NOT SUPPORT THE HYPOTHESIS

Three preregistered analyses returned null or negative results, and we report them in full.

Causal necessity of the probe subspace. Ablating the rank-8 subspace recovered by the canonical-state probe from ORION's final layer changes exact match by $+0.000$ (intact 0.882), against $+0.005$ for a random subspace of the same rank and $+0.150$ for the top-8 PCA subspace. The probe subspace is therefore *not* more necessary than a random one by this test, and the preregistered "probe $>$ random" criterion fails.

Counterfactual latent editing. Editing the latent towards the canonical state of a counterfactual program produced the counterfactual answer in 0 of 200 attempts, for both the supervised and the unsupervised model. We cannot distinguish an absent causal pathway from an inadequate editing procedure, and we make no claim either way.

Statistical power. With three seeds, the paired seed-level effect on FOLDING depth extrapolation is $+0.008$ with a between-seed standard deviation of 0.006 (Cohen's $d = 1.32$). The minimum detectable effect at 80% power is 0.010, and five seeds would be required. The per-example paired

tests reported in the main text have far greater power because they pair on the $1{,}000$ test instances, but the seed-level comparison is underpowered and we do not rely on it.

D Experimental Setup and Reproducibility

D.1 Dataset Specifications

- **FOLDING:** Flat-folding of an $n \times n$ sheet at quadrant resolution. Training programs apply 1–4 folds to sheets of size 4 or 6; the extrapolation splits apply 5–8 folds, use size 8, or both. Diagonal folds are held out for the operation-type axis.
- **GRIDSHIFT:** Cyclic row and column shifts of an $N \times N$ grid of origin labels. Training programs apply 1–4 shifts to grids of size 4 or 6; the extrapolation splits apply 5–8 shifts, use size 8, or both. Transpose is held out.
- **CLRS pilot:** 8 algorithms from the CLRS suite in textual form, evaluated at problem sizes 16 (in-distribution), 32 (length axis) and 64 (size axis). Negative controls were not trained on CLRS.

D.2 Hyperparameters

Table 11: Hyperparameter settings across model configurations. The scaling ladder of Table 13 varies $d_{\text{model}} \in \{256, 384, 512\}$ at a reduced step budget with five seeds; all other experiments use the configuration below.

Parameter	Baseline	RoPE Only	ORION Only	ORION + RoPE
Layers (L)	6	6	6	6
Hidden dim (d_{model})	384	384	384	384
Attention heads	6	6	6	6
Parameters	11.2M	11.2M	11.4M	11.4M
Optimizer	AdamW	AdamW	AdamW	AdamW
Learning rate	0.0003	0.0003	0.0003	0.0003
Weight decay	0.01	0.01	0.01	0.01
Batch size	256	256	256	256
Training examples	40,000	40,000	40,000	40,000
Test instances per split	1,000	1,000	1,000	1,000
Auxiliary loss weights (λ)	N/A	N/A	Default*	Default*
Total training steps	6,000	6,000	6,000	6,000
Random seeds	3 (0–2)	3 (0–2)	3 (0–2)	3 (0–2)

*$\lambda_{\text{spatial}} = \lambda_{\text{topo}} = \lambda_{\text{occ}} = \lambda_{\text{probe}} = 0.5$, $\lambda_{\text{cons}} = 0.25$, $\lambda_{\text{contr}} = 0.1$. These were fixed at their preregistered values and never swept; §5.4 varies the *availability* of supervision, not its weight.

D.3 Negative Controls in Detail

Each control keeps ORION's architecture, heads, parameter count and every loss term identical to the full model, and changes only the content of the supervision target. The output-token targets are never touched, so the controls degrade the state supervision, not the task. All five were trained with absolute positional encodings.

- **N1 (Compute-matched):** no auxiliary loss, trained for $1.5\times$ the step budget ($9{,}000$ against $6{,}000$ steps). This isolates additional optimisation.
- **N2 (Random targets):** canonical targets replaced by deterministic per-example pseudo-random targets of matched dimension — capacity and a learnable auxiliary gradient, with no state content.
- **N3 (Inconsistent state):** occupancy and topology targets corrupted for a fraction $\epsilon = 0.25$ of examples, so the state components mutually disagree at a controlled rate.

- **N4 (Shuffled targets):** canonical targets permuted across the batch, so each head decodes another example's exact state, $\Phi(S_{\pi(t)})$ — well-formed but uninformative.
- **N5 (Hashed statistics):** a fixed per-seed coordinate permutation applied to the state targets. This is a bijective, information-preserving but *non-canonical* encoding: the geometric layout and the linear transition operator are destroyed, so the consistency term is disabled for this variant. The answer target is left correct. This is the control that Theorem 4 and its remark are designed to make decisive.

Table 12: OOD exact match on the depth-extrapolation split ($2 \times D$ at the training state size) across negative controls, mean $\pm$ s.d. over 3 seeds. CLRS controls were not trained. The controls lie within the seed spread of the APE baseline and of ORION with APE (0.026 on FOLDING, 0.063 on GRIDSHIFT); the separation in this table is between the rotary column and everything else.

Benchmark	Baseline (N1)	N2	N3	N4	N5	ORION + RoPE
FOLDING	0.021 ± 0.008	0.021 ± 0.006	0.024 ± 0.008	0.027 ± 0.008	0.031 ± 0.005	0.667 ± 0.018
GRIDSHIFT	0.093 ± 0.012	0.081 ± 0.020	0.068 ± 0.018	0.077 ± 0.034	0.073 ± 0.020	0.598 ± 0.038
Mean	0.057	0.051	0.046	0.052	0.052	0.633

D.4 SCALING LADDER

Table 13 gives the ladder plotted in Figure 2c. The three widths were trained with five seeds each at a reduced step budget, which is why the absolute values sit below those of Table 2; only the within-width comparison between the two conditions is meaningful.

Table 13: RoPE versus ORION + RoPE on depth extrapolation in FOLDING across model width. Values are EM percentages, mean $\pm$ standard deviation over five seeds. The ladder uses a reduced training budget, so its absolute values are lower than those in Table 2.

Model Scale	RoPE Only Depth OOD	ORION + RoPE Depth OOD	Absolute Delta
Small (3.5M)	41.9 ± 3.1	41.9 ± 2.7	+0.0 pp
Medium (11.2M)	54.0 ± 1.6	54.5 ± 2.3	+0.4 pp
Large (26M)	54.6 ± 3.6	$\mathbf{62.8 \pm 1.9}$	**+8.2 pp**

D.5 POSITIONAL-ENCODING INTEROPERABILITY

Table 14 isolates the interaction between positional encoding and state supervision on FOLDING.

Table 14: Positional-bias interoperability across the size and depth axes on FOLDING. Values are EM percentages. ALiBi and other relative-bias schemes were not trained in this study and are therefore not listed; we make no claim about their interaction with ORION.

Positional Scheme	State Supervision	Size Extrap.	Depth Extrap.	Joint
APE	None	17.1	2.1	0.4
APE	ORION	17.1	2.6	0.4
RoPE	None	19.1	65.1	1.8
RoPE	**ORION**	**19.6**	**66.7**	**2.2**

Read across the rows, the table gives the cleanest statement of the paper's finding. Rotary embeddings move depth extrapolation from 2.1% to 65.1% with no state supervision at all. Canonical supervision adds 0.5 points on top of absolute embeddings and 1.6 points on top of rotary embeddings. Neither scheme repairs the size axis.

D.6 COMPUTATIONAL RESOURCES

- **Compute:** training ran on one node of $4\times$ NVIDIA A100-SXM4-80GB. Summing the recorded wall-clock time of every training job in the released manifests gives 40 GPU-hours across the main run, the five-seed follow-up and the architecture baselines. Computing the canonical-state targets and their gradients raised per-run wall-clock training time by 181% relative to the unsupervised baseline. The joint-regime evaluation reported here is inference only and was run on CPU from the released checkpoints.
- **Code and artefacts:** implementation code, dataset generators, model weights, diagnostic plotting scripts, and the table-generation scripts that emit every table in this paper from run artefacts are included in the supplementary material and will be released under the MIT License upon publication.