Fluid Reasoning Representations

Dmitrii Kharlapenko[1,2*] **Terry Jingchen Zhang**[1,2,3*] **Arth Singh**[1,4,5*]
Alessandro Stolfo[2] **Arthur Conmy** **Mrinmaya Sachan**[2] **Zhijing Jin**[1,6]
[1]Jinesis Lab, University of Toronto &Vector Institute
[2]ETH Zurich [3]University of Oxford
[4]AIM Intelligence [5]Georgia Institute of Technology
[6]Max Planck Institute for Intelligent Systems, Tübingen, Germany
[*]Equal contribution

Abstract

Frontier language models are progressing rapidly on complex tasks involving abstract concepts through extended thinking at test time, yet we lack a mechanistic account of how such thinking changes their internal representations. We introduce *Fluid Reasoning Representations* (FRRs), a representation-level account of how LLMs organize action and predicate concepts during self-generated reasoning, and test them on obfuscated planning, symbolic, and mathematical tasks. Across base, instruction-tuned, and extended-thinking LLMs, representations of the same action or predicate become more similar across wordings and move toward the corresponding unobfuscated concepts over the reasoning trace. Causal probes show that these representations affect behavior: cross-naming steering improves held-out accuracy beyond Gaussian and shuffled controls, symbolic patching retains more action information than shuffled patching, and subtracting refined directions degrades accuracy; together, these results suggest that extended thinking strengthens a representation dynamic already present at lower magnitude in base and instruction-tuned LLMs. Our codebase is open-sourced here.

1 Introduction

Test-time thinking has become a central feature for frontier large language models (LLMs) to tackle complex tasks involving abstract concepts. Scaling inference-time computation can improve LLM performance, in some settings more effectively than increasing model size (Snell et al., 2024). Yet we still lack a mechanistic account of what this process changes inside the model (Venhoff et al., 2025; Xu et al., 2025). Prior work shows that task and function information can be encoded, localized, and manipulated in activation space (Todd et al., 2024; Hendel et al., 2023; Park et al., 2025). This representation-level view raises a sharper question: does extended thinking mainly allocate additional computation to search and template reuse, or does it reshape internal representations used by later reasoning steps?

We address this problem by studying model representations on obfuscated reasoning tasks. These tasks keep the underlying problem structure fixed while systematically replacing task-relevant words, making it possible to test whether representations of the same operation become more consistent across wordings and useful for downstream behavior. We study this setting across planning, symbolic, and mathematical reasoning tasks and multiple open-weight LLMs, including QwQ, Qwen2.5, DeepSeek-R1-Distill-Qwen, Nemotron, and Seed-OSS variants; controlled planning domains serve as the primary causal testbed. The detailed task construction, representation extraction, and intervention setup are given in Section 3; the introduction focuses on the central question of whether test-time thinking changes the representations that support reasoning.

We introduce **Fluid Reasoning Representations** (FRRs), a representation-level account of how LLMs organize action and predicate concepts during self-generated reasoning. FRRs describe a dynamic in which representations of the same action or predicate become more similar across wordings and move toward the corresponding unobfuscated concepts. We test this account by asking three questions: whether equivalent concepts become more consistent across different wordings, whether they partially align with the corresponding unobfuscated concepts, and whether interventions on these representations change downstream reasoning behavior.

Contributions. This paper contributes a mechanistic account of thinking behavior in LLMs by defining FRRs and showing how they can be measured in obfuscated reasoning traces. Empirically, action and predicate representations become more similar across wordings over the reasoning trace and move toward corresponding unobfuscated con-

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 13823–13841
October 24-29, 2026 ©2026 Association for Computational Linguistics

cepts. Causal probes show that these representations affect behavior: cross-naming steering improves held-out accuracy beyond Gaussian and shuffled controls, symbolic patching retains more action information than shuffled patching, and subtracting refined directions degrades accuracy. Base and instruction-tuned LLMs exhibit the same dynamic at lower magnitude, suggesting that extended thinking strengthens an existing representational dynamic rather than creating it from scratch; alignment with unobfuscated concepts still does not guarantee functional interchangeability.

2 Related Work

Representation-level accounts of LLM behavior. Prior work shows that LLM behavior can often be understood through activation-space directions or features[1] (Nanda et al., 2023; Olah, 2024), and that task/function information can be localized, manipulated, or constructed in context (Todd et al., 2024; Hendel et al., 2023; Park et al., 2025). These studies make representation analysis useful for causal questions, but usually focus on fixed tasks, prompt-level task vectors, or short-context adaptation. We instead ask how self-generated test-time thinking changes action and predicate representations over the course of a trace, and whether those changing representations matter for later behavior.

Thinking behavior and internal mechanisms. Recent reasoning interpretability work studies behavioral and representational structure in extended thinking traces, including uncertainty, backtracking, verification, and reasoning-specific internal features (Venhoff et al., 2025; Bogdan et al., 2025; Lee et al., 2025; Ward et al., 2025). This work helps explain what appears in traces and where some internal features are located. FRRs target a complementary question: whether the thinking process reshapes task representations that the model later uses, which we test by comparing representations across timestamps and intervening on them.

Obfuscated reasoning. Obfuscated reasoning tasks keep the underlying problem structure fixed while replacing task-relevant words, exposing how much LLMs rely on familiar lexical cues (Valmeekam et al., 2023, 2024). We use this setting as a controlled identification strategy: vary surface namings while preserving the same task, then test whether representations for corresponding actions and predicates become more consistent, align with clean-domain concepts, and causally affect behavior. This differs from robustness evaluation because the goal is not only to measure failure under obfuscation, but to isolate the internal adaptation that makes some obfuscated reasoning succeed.

[1] https://transformer-circuits.pub/2021/framework/index.html

3 Methodology

Dataset. We use BlocksWorld, a classic planning domain (IPC, 1998), as the base environment: each puzzle specifies initial and goal block arrangements, and the agent must use *pick-up*, *put-down*, *stack*, and *unstack* under simple physical constraints. We generate and verify puzzles with PlanBench (Valmeekam et al., 2023) (prompt in Section C). To create the obfuscated setting, Mystery BlocksWorld replaces every action and predicate with a semantically unrelated word under a complete **naming**, preserving logical structure while breaking familiar surface semantics (prompt in Section D). We generated 14 variants beyond the original, yielding 15 obfuscations of the same domain; a **puzzle** is a unique initial-goal state pair, and our analysis uses 300 four-block puzzles mapped across all 15 namings.

Baseline evaluations. QwQ-32B is the strongest open-weight LLM in our evaluation, solving regular BlocksWorld at 96% and Mystery BlocksWorld at 35%; successful mystery solutions typically require 15–20k-token traces (Table 1). A reasoning-behavior breakdown following Venhoff et al. (2025) appears in Section E.

Naming variation. QwQ-32B accuracy varies across namings (0.05–0.47): coherent alternative domains and reversible-sounding operations are hardest, while abstract or mismatched replacements are easier (full results in Section K). *Mystery naming 3* uses random strings; QwQ-32B often recognizes BlocksWorld and maps symbols quickly in this condition (about 2k tokens vs. 15–20k elsewhere), consistent with training-set exposure, so we exclude naming 3 from representational analyses because it largely bypasses the semantic adaptation process.

3.1 Representation Collection

Representation construction. Following Park et al. (2025), we extract action and predicate vectors from LLM activations during BlocksWorld and Mystery BlocksWorld reasoning. For traces $\mathcal{B}$, phrase a, naming N, layer L, and timestamp T,

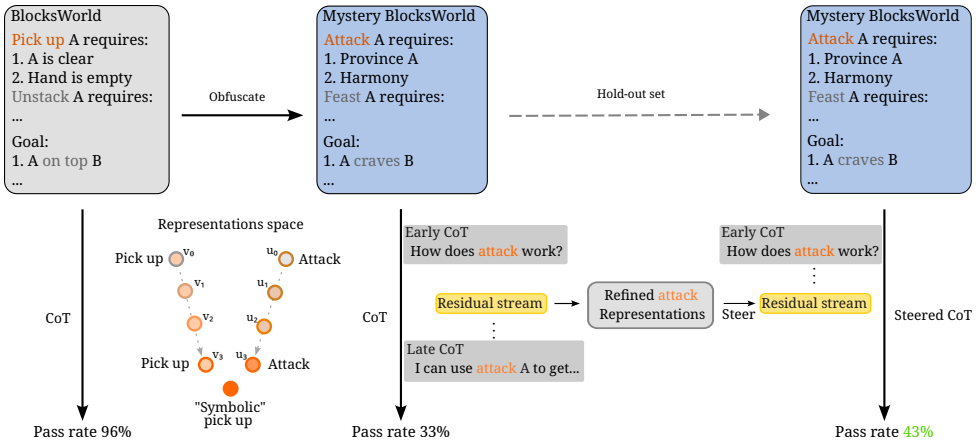


Figure 1: **Pipeline overview.** We extract action/predicate representations from obfuscated reasoning traces and test them with steering interventions.

Table 1: BlocksWorld and **Naming 1** Mystery BlocksWorld performance on 300 puzzles. "Tokens" is average thinking-trace length; "Accuracy Preserved" is Mystery/Standard. * indicates unparsable formatting.

LLM	BlocksWorld		Mystery		Accuracy Preserved
	Acc	Tokens	Acc	Tokens	
Regular LLMs					
GPT-4.1 (step-by-step)	0.92	556	0.18	3837	**20%**
Qwen2.5-32B-base	0.21	71	0.00	1390	**0%**
Qwen2.5-32B-instruct (step-by-step)	0.38	353	0.00	1479	**0%**
Llama-3.3-70B-instruct (step-by-step)	0.40	760	0.02	1142	**5%**
LLMs with Extended Test-Time Thinking					
DeepSeek-R1-Distill-Qwen-32B	0.81	2387	0.08	8500	**10%**
DeepSeek-R1-Distill-Llama-70B	0.66	2674	0.10	10636	**15%**
Nemotron-49B-instruct	0.48*	1162	0.19	9200	**40%**
QwQ-32B	**0.96**	3633	**0.35**	16186	**36%**

we scan the 100-token window before T, collect token spans encoding a plus one preceding token, average hidden states within each span and then across spans, and repeat every 200 tokens at all layers:

$$\mathbf{r}_a^{N,L,T} = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \left(\frac{1}{|s|} \sum_{i \in s} \mathbf{h}_i^L \right), \quad (1)$$

where $\mathcal{S}$ is the set of matched spans. Within each naming, we follow Venhoff et al. (2025) and center over the action or predicate set $\mathcal{A}$:

$$\tilde{\mathbf{r}}_a^{N,L,T} = \mathbf{r}_a^{N,L,T} - \frac{1}{|\mathcal{A}|} \sum_{a' \in \mathcal{A}} \mathbf{r}_{a'}^{N,L,T}, \quad (2)$$

which removes the per-naming common-mode component. We then average centered vectors across namings to isolate what is shared across surface forms:

$$\bar{\mathbf{r}}_a^{L,T} = \frac{1}{|\mathcal{N}|} \sum_{N \in \mathcal{N}} \tilde{\mathbf{r}}_a^{N,L,T}, \quad (3)$$

where $\mathcal{N}$ is the set of namings. We validate this construction with PCA, clean-domain similarity, positive steering, and negative steering in Sections 4.2 and 5.

3.2 Fluid Reasoning Representations

An LLM exhibits *Fluid Reasoning Representations* for action set $\mathcal{A}$ if, for some layer L, $\{\mathbf{r}_a^{N,L,T}\}_{a,N,T}$ satisfies three criteria as T increases:

C1 *Cross-naming alignment.* The cosine similarity between $\tilde{\mathbf{r}}_a^{N,L,T}$ and $\bar{\mathbf{r}}_a^{L,T}$ increases with T, while the similarity between $\tilde{\mathbf{r}}_a^{N,L,T}$ and $\bar{\mathbf{r}}_{a'}^{L,T}$ for $a' \neq a$ does *not* (and ideally decreases). [Tested: Figure 4.]

C2 *Clean-domain similarity.* The cross-naming vector $\bar{\mathbf{r}}_a^{L,T}$ approaches the centered representation of the same action or predicate computed on the unobfuscated domain, despite no token overlap. [Tested: Figure 4.]

C3 *Causal contribution.* Injecting $\bar{\mathbf{r}}_a^{L,T}$ into early reasoning improves held-out accuracy more

than (a) Gaussian noise of matched norm, (b) shuffled assignments of actions to vectors, and (c) in-naming vectors $\tilde{\mathbf{r}}_a^{N,L,T}$; subtracting the refined direction degrades accuracy beyond a shuffled control of equal magnitude. [Tested: Sections 5.1 and 5.3; Table 6.]

Relation to prior work. Unlike explicit in-context remappings in synthetic graph tasks (Park et al., 2025), our setting requires inference from ∼15–20k self-generated thinking tokens and adds clean-domain alignment (C2) plus causal tests (C3).

4 Representational Studies

In this section, we first test whether QwQ-32B (Qwen Team, 2025) refines action and predicate representations during extended Mystery BlocksWorld reasoning. We operationalize *Fluid Reasoning Representations* as increasing alignment between in-naming representations and their counterparts averaged across namings, plus causal effects under intervention.

4.1 Representational Convergence

We extract in-naming representations from Mystery naming 1 at 2k, 4k, 7k, and 10k tokens and compare them with representations from all other namings. Figure 2 shows that corresponding actions become more similar across namings during reasoning, plateauing around 7k tokens, while different-action similarities remain lower; Figure 3 shows the same pattern in PCA space, with equivalent actions clustering across namings especially in deeper layers. The residual similarity among different actions is mainly driven by "stack" and "unstack" being closer to each other than to "pick up" or "put down". This convergence also clarifies the cost of reasoning. QwQ-32B solves Standard BlocksWorld with ∼3.6k tokens, but successful Mystery solutions require 15–20k tokens (Table 1). The first ∼7k tokens align with the convergence phase in Figure 2, suggesting that representational alignment, not raw token count alone, is the relevant cost.

4.2 Representation Similarity

We next compare naming-specific representations with cross-naming averages and clean-domain representations. Similarity to the corresponding cross-naming average increases during reasoning and plateaus around 7k tokens, while different-action similarities become increasingly negative (Figure 4). Clean BlocksWorld representations also

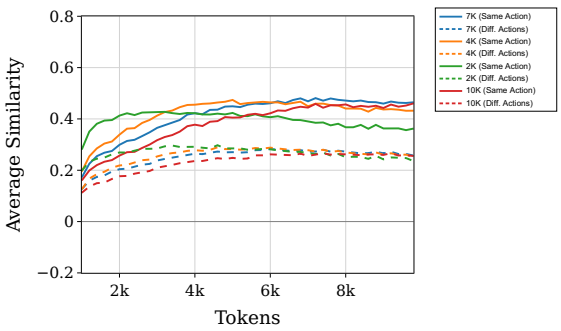


Figure 2: **Action and predicate representations converge across obfuscated namings over the reasoning trace.** Cross-naming cosine similarity grows over tokens and plateaus near 7k; same-action pairs separate from different-action pairs.

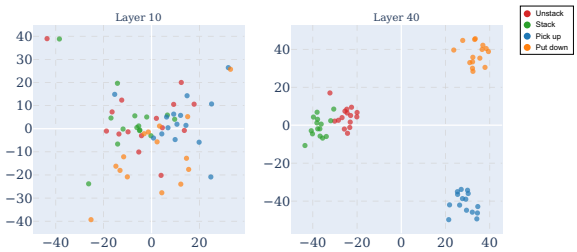


Figure 3: **Layer-wise PCA at 7k tokens shows clean action-type clustering.** Equivalent actions cluster across obfuscated namings, and the separation sharpens with depth.

move toward the same cross-naming vectors over time, linking the obfuscated-trace dynamic to the model's representation of the unobfuscated task. Together, these comparisons give the cross-naming average a concrete interpretation: it tracks a role-specific direction that preserves action identity while abstracting over surface labels. The clean-domain comparison anchors the same direction to the unobfuscated action concept, linking the obfuscated representations to the task structure that the model must use to plan. We therefore treat the cross-naming average as an empirical construction of shared role information.

4.3 Qwen2.5 Trace-Processing Controls

We tested whether the FRR phenomenon is specific to post-training for extended test-time thinking by feeding the *same* QwQ-generated traces through Qwen2.5-32B-instruct and Qwen2.5-32B-base,[2] then extracting representations identically. Both Qwen2.5 variants exhibit qualitatively simi-

[2] https://huggingface.co/Qwen/Qwen2.5-32B-Instruct

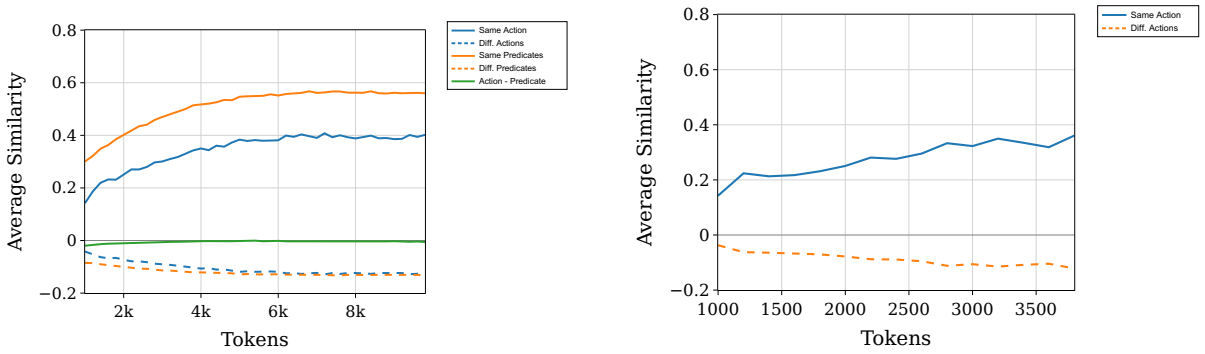


Figure 4: **Both Mystery and clean BlocksWorld traces drift toward the same cross-naming averages.** (Left) Centered Mystery representations approach their cross-naming average over the trace, while different-action pairs separate. (Right) Clean BlocksWorld representations also move toward the cross-naming average, linking obfuscated adaptation to the corresponding unobfuscated task concepts; predicates are omitted because their tokens are harder to identify in clean traces.

lar but weaker convergence dynamics (Figure 5), with the base LLM adapting more slowly, consistent with the fact that it is being asked to process traces it would not itself generate. This comparison places FRRs on a broader in-context representational adaptation substrate (Park et al., 2025) that is already present in base and instruction-tuned Qwen2.5 LLMs. The contrast also matters: models trained to sustain extended test-time thinking generate the self-consistent traces that drive alignment past the 7k-token plateau (Table 1) and make the dynamic more visible under intervention. We therefore frame FRRs as a representational dynamic that is **amplified and operationally exploited** by extended test-time thinking rather than created from scratch.

4.4 Cross-Model/Cross-Domain Analysis

We also study the convergence dynamic on three additional open-weight LLMs: DeepSeek-R1-Distill-Qwen-32B (DeepSeek-AI, 2025), Nemotron-49B-instruct,[3] and Seed-OSS-36B-instruct (ByteDance-Seed-Team, 2025). All three reproduce both the increasing in-naming/cross-naming alignment of criterion **C1** (Section G) and the layer-wise PCA separation of action types (Figure 8). Across our 20 namings, which span coherent alternative domains (gardening, legal, cooking), mystical framings, and nonsense strings (Section J), baseline accuracy ranges 0.05–0.65 with naming, but C1 holds for every non-degenerate naming and the cross-naming construction in Equation (3) generalizes to namings withheld from steering-vector construction (Table 5). These results keep the empirical scope focused on structured planning while extending beyond a single naming or LLM.

Finally, to test whether the same representational dynamic holds beyond BlocksWorld, we ran the representation pipeline of Section 3.1 on a second LLM (Nemotron-49B-instruct) and two additional families: *Mystery Logistics* (PDDL Logistics with action and predicate vocabulary obfuscated under five surface variants) and *GSM8K-Renamed* (math word problems with nouns and operation verbs replaced under four variants, numerals and person names preserved). We collect 200 traces per cell at each LLM's default steering layer (47 for QwQ-32B, 60 for Nemotron-49B-instruct) and report C1 as the mean cross-naming cosine over actions and predicates within each family (Table 2). Three conclusions follow from Table 2: C1 remains positive and usually strong in every (LLM, family) cell despite zero token overlap across namings, so FRRs transfer from BlocksWorld to PDDL Logistics and GSM8K-style math reasoning rather than reflecting a BlocksWorld-only artifact; the LLM ranking is domain dependent, with QwQ-32B giving the clearest BlocksWorld alignment while Seed-OSS-36B-instruct and DeepSeek-R1-Distill-Qwen-32B lead on Logistics and GSM8K; and the Qwen2.5-32B trajectory shows that post-training reshapes a pre-existing substrate, since Qwen2.5-32B-base and Qwen2.5-32B-instruct already align broadly across domains whereas QwQ-32B specializes the dynamic for BlocksWorld and loses cross-domain alignment. Cross-domain mean-residual and C2

[3] https://huggingface.co/nvidia/Llama-3_3-Nemotron-Super-49B-v1

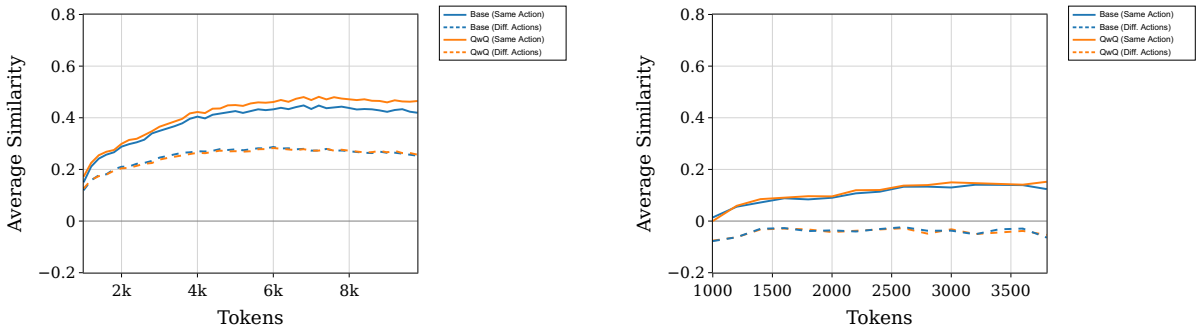


Figure 5: **Qwen2.5 base and instruct exhibit weaker but qualitatively similar FRR dynamics when fed QwQ-generated traces** (7k tokens). (Left) Cross-naming similarity vs. QwQ. (Right) Clean-domain similarity vs. QwQ. Both controls reproduce the trend at lower magnitude, indicating extended-thinking post-training amplifies the dynamic rather than creates it.

analyses support the same picture: the LLMs share a generic structural subspace across families, but cosine alignment alone does not guarantee later functional interchangeability.

These cross-domain results sharpen the scope of the claim. Logistics preserves planning-style state transitions while changing the object and predicate inventory; GSM8K-Renamed preserves arithmetic structure while replacing familiar lexical cues. Positive C1 in both settings shows that the measured alignment extends beyond the specific BlocksWorld vocabulary and actions. The domain-dependent ranking shows that the strength of this alignment depends on both the task family and the LLM, so we use C1 as a diagnostic for consistent role representations within a task family.

LLM	Family	$\lvert\mathcal{N}\rvert$	$C1_a$	$C1_p$
QwQ-32B	BlocksWorld	14	**0.94**	**0.93**
Seed-OSS-36B-instruct	BlocksWorld	3	0.83	0.82
Qwen2.5-32B-instruct	BlocksWorld	3	0.83	0.82
DeepSeek-R1-Distill-Qwen-32B	BlocksWorld	3	0.80	0.79
Qwen2.5-32B-base	BlocksWorld	3	0.79	0.75
Nemotron-49B-instruct	BlocksWorld	3	0.67	0.73
Qwen2.5-32B-base	Logistics	3	**0.84**	0.81
Qwen2.5-32B-instruct	Logistics	3	0.81	**0.83**
Seed-OSS-36B-instruct	Logistics	3	0.77	0.80
DeepSeek-R1-Distill-Qwen-32B	Logistics	3	0.74	0.77
QwQ-32B	Logistics	3	0.68	0.55
Nemotron-49B-instruct	Logistics	3	0.61	0.63
Seed-OSS-36B-instruct	GSM8K	2	**0.86**	**0.83**
Qwen2.5-32B-base	GSM8K	2	0.84	0.81
DeepSeek-R1-Distill-Qwen-32B	GSM8K	2	0.82	0.81
Qwen2.5-32B-instruct	GSM8K	2	0.82	0.79
Nemotron-49B-instruct	GSM8K	2	0.66	–
QwQ-32B	GSM8K	2	0.65	–

Table 2: **C1 across (LLM, family) cells.** $\lvert\mathcal{N}\rvert$ is the number of obfuscation namings; $C1_a$ and $C1_p$ are cross-naming cosine alignment for actions and predicates. "–" indicates unmeasured predicate alignment; best $C1_a$ per family is bolded.

5 Causal Validation

The representational results suggest that refined vectors encode actionable puzzle structure. We test this with positive steering, symbolic patching, negative steering, and clean-centroid replacement. Together, these probes form a **causal evidence chain**. Positive steering asks whether a refined vector can help an early trace; symbolic patching tests whether the vector must match the underlying action or predicate; negative steering tests whether the original trace relies on the refined direction; and replacement steering tests how far clean-domain and obfuscated representations can substitute for one another.

5.1 Positive Steering

We inject late-stage action and predicate representations into early reasoning traces and measure held-out accuracy. Gains from cross-naming vectors test whether the representation abstracts beyond a specific lexical naming.

Experimental Setup. Our steering procedure selects a steering layer L, token window $[t_{\text{start}}, t_{\text{end}})$, and steering scale s. We collect three types of steering vectors at layer L from the 40 correctly solved puzzles: **(1)** centered **in-naming** representations $\tilde{\mathbf{r}}_a^{N,L,T}$ for all actions and predicates, **(2) cross-naming** representations $\bar{\mathbf{r}}_a^{L,T}$ averaged across all namings, and **(3)** random Gaussian vectors $\mathbf{v}_{\text{rand}}[a]$ scaled to match the norm of **in-naming** representations. We extract prefixes of t_{end} tokens from a hold-out set of 100 different 4-block problem roll-

outs as our intervention dataset. For each prefix, we identify token indices i corresponding to action or predicate a, obtain hidden states $\mathbf{h}_i^L$ at layer L, and apply the following norm-preserving intervention:

$$\mathbf{h'}_i^L = s \cdot \mathbf{h}_i^L + (1-s) \cdot \mathbf{v}_{\text{type}}[a], \tag{4}$$

$$\mathbf{h}_i^L \leftarrow \mathbf{h'}_i^L \cdot \frac{\|\mathbf{h}_i^L\|_2}{\|\mathbf{h'}_i^L\|_2}, \tag{5}$$

where $\mathbf{v}_{\text{type}}[a] \in \{\tilde{\mathbf{r}}_a^{N,L,T}, \bar{\mathbf{r}}_a^{L,T}, \mathbf{v}_{\text{rand}}[a]\}$ depending on the experiment condition. This procedure adds the refined representation while preserving activation magnitude. We measure accuracy improvement on steered puzzles compared to non-steered baseline. We selected scale $s = \frac{2}{3}$ after a sweep on layer 20 using **in-naming** representations (Section L.2; see also Figure 11). The steering window is $[1500, 2500]$.

Results. Figure 6 shows accuracy change after positive steering, averaged across namings (excluding naming 3), as a function of intervention layer L, with full per-condition statistics in Table 6. The layer-resolved view directly addresses a concern that our gains might come from disrupting misleading surface lexical priors rather than from injecting useful structure: random Gaussian gains are confined to early layers ($L \leq 10$); from layer 20 onward, where layer-wise PCA shows the onset of action-type separation across namings (Section F), random-noise steering stops helping (layer 20 random Gaussian: -0.36%, p=0.627, Table 6), while structured injections continue to yield significant gains (layer 40 cross-naming: $+1.43\%$, p=0.021). If the cross-naming advantage stemmed from generic surface-token disruption, both random noise and shuffled in-naming permutations would match it at deep layers; they do not (Section M). Mean improvements at significant cells are 1.4–1.8%; per-naming maxima reach +10% on coherent-domain namings where surface priors are strongest (Section K), but heterogeneity across namings and a layer-by-naming interaction keep the cross-naming-averaged effect bounded (per-naming optima in Table 5). The experiment is designed as a causal probe, so the relevant signal is the **structured/random asymmetry**: **cross-naming** $>$ **in-naming** $\gg$ **random**, consistent with criterion **C3** of Fluid Reasoning Representations.

5.2 Symbolic Patching

To test whether adapted representations support abstraction across surface forms, we conduct a patching experiment that replaces naming-specific representations with cross-naming "symbolic" representations and tests whether the LLM can operate effectively without access to the original naming-specific encodings.

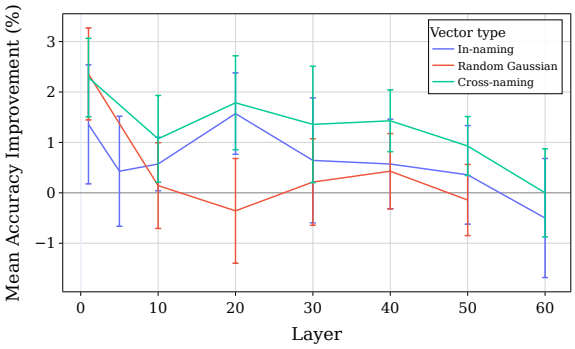


Figure 6: **Positive steering across layers.** Accuracy change under in-naming, cross-naming, and Gaussian steering; error bars show standard error.

Symbolic Representation Construction. We construct symbolic representations to be minimally out-of-distribution while capturing abstract structural information. We collect centered **cross-naming** representations $\bar{\mathbf{r}}_a^{L,T}$ for each action and predicate, compute the overall mean $\bar{\mathbf{r}}_{\text{mean}}^{L,T}$ across all actions (and separately for predicates), and construct symbolic representations as:

$$\mathbf{r}_{\text{symbolic}}[a] = \bar{\mathbf{r}}_{\text{mean}}^{L,T} + s \cdot \bar{\mathbf{r}}_a^{L,T} \tag{6}$$

where s is a mixing scale and $\bar{\mathbf{r}}_a^{L,T}$ is the centered cross-naming representation of action a.

Experimental Design. Since the LLM maintained reasonable accuracy even when all actions were replaced with a single vector, we use a comparative approach: **(1)** Symbolic Patching replaces residual stream activations for action/predicate tokens with corresponding symbolic representations, and **(2)** Shuffled Patching uses randomly permuted symbolic representations as control. We patch token window $[2000, 4000]$ on all layers up to a selected end layer, then measure accuracy difference $\text{Acc}_{\text{symbolic}} - \text{Acc}_{\text{shuffled}}$. Across all tested mixing scales and end layers, **matched symbolic representations** consistently outperform shuffled ones (Figure 11, in the appendix), supporting criterion **C3**: the cross-naming construction carries information that the shuffled control destroys, even when both are inserted at the same layers and magnitudes. Because both conditions patch the same positions with

the same vector construction, the accuracy gap reflects whether the inserted vector is assigned to the correct action or predicate.

5.3 Negative Steering

To further test whether refined in-naming representations are functionally used, we conduct negative steering by subtracting converged in-naming representations from LLM activations. Since steering interventions can easily degrade performance, we use shuffled representations as a comparative baseline. Performance degradation beyond the shuffled control demonstrates that the adaptations encode operationally important structural information rather than arbitrary activation patterns.

Experimental Design. We perform interventions across token window $[2000, 4000]$ on multiple layers, subtracting centered naming representations extracted from the 4k timestamp (chosen because representations are near convergence while still at our window's end, see Figure 2). We use shuffled in-naming representations as control rather than Gaussian noise, because shuffling preserves the per-naming activation statistics and only breaks the action–vector correspondence; this is exactly the null we need to rule out for criterion **C3**.

Result. Subtracting refined in-naming representations degrades held-out accuracy by $2.9\% \pm 1.06\%$ (start layer 10, end layer 30) relative to a shuffled control of equal magnitude, and by $2.3\% \pm 0.99\%$ at end layer 20 (full per-layer results in Section I). Because the shuffled control matches activation magnitude, per-naming statistics, and intervention layers exactly, the gap is not attributable to general intervention disruption. Together with positive steering, negative steering shows the same representation directions from both sides: **injecting** refined directions can improve held-out accuracy, while **subtracting** matched directions selectively degrades it.

5.4 Replacement Steering

The preceding interventions use naming-specific or cross-naming vectors extracted from obfuscated traces. As a stricter causal probe, we test whether the centered clean-domain centroid for an action is itself read by downstream layers as a coherent substitute for the obfuscated representation. At each LLM's default steering layer, we install a forward hook that replaces the residual stream activation at obfuscated phrase positions with the corresponding clean-naming centroid, using positional alignment between obfuscated and clean phrase orderings. We compare against baseline and ablation controls, scoring per-token log-likelihood of each obfuscated trace's own continuation tokens over 30 traces per cell.

On Nemotron-49B-instruct, clean-centroid replacement is less disruptive than zeroing for coherent Mystery BlocksWorld namings, but random-letter naming erases the advantage and Logistics does not show the same substitution effect. We reran the probe on Qwen2.5-32B-instruct, Qwen2.5-32B-base, and the two highest-C1 LLMs outside QwQ-32B; across these four LLMs, clean-centroid replacement is consistently more disruptive than zeroing on BlocksWorld, while the two probes are statistically indistinguishable on Logistics. Replacement steering therefore supports a narrow functional reading: **geometric alignment** of obfuscated and clean centroids does *not* automatically imply **functional interchangeability**, and the clean centroid acts as a coherent substitute only in the Nemotron BlocksWorld cell. The cross-naming-vs-Gaussian asymmetry of Section 5.1 remains the cleanest C3 evidence overall.

6 Conclusion

We introduced *Fluid Reasoning Representations* as a mechanistic account of how LLMs organize abstract task concepts during extended test-time thinking. Across obfuscated reasoning tasks and multiple open-weight models, action and predicate representations become increasingly tied to their latent roles rather than their surface forms, and interventions on these representations affect downstream reasoning. The evidence suggests that test-time thinking is not only a longer search process: it can also construct **task-specific symbol meanings** in activation space. Comparisons with base and instruction-tuned models further show that they already exhibit the same dynamic at lower magnitude, so extended test-time thinking amplifies and operationally exploits a **pre-existing representational substrate** rather than creating it from scratch. FRRs therefore offer a concrete view on the internal dynamics behind thinking behavior at test time in complex problem-solving.

Limitations

Our evidence is concentrated on structured reasoning settings where action and predicate phrases can

be aligned across obfuscated and clean variants. The interventions are controlled probes over fixed token windows and should be interpreted as evidence about representational dynamics rather than as general-purpose accuracy interventions. Cross-domain analyses include Logistics and GSM8K-style renaming, but the strongest functional evidence remains in the BlocksWorld setting; broader domains and model families are left for future work.

Acknowledgements

This work was supported in part by the German Federal Ministry of Education and Research (BMBF): Tübingen AI Center, FKZ: 01IS18039B; by the Machine Learning Cluster of Excellence, EXC number 2064/1 – Project number 390727645; by Coefficient Giving; by the SwissAI Initiative; by Schmidt Sciences; by the Frontier Model Forum and AI Safety Fund; by the Canadian AI Safety Institute Research Program at CIFAR; by the Canadian AI Safety Institute Research Program at CIFAR through a Catalyst Award; by the Survival and Flourishing Fund; by the Cooperative AI Foundation; and by the Division of Engineering Science at the University of Toronto. Resources used in preparing this research project were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute.

References

Paul C. Bogdan, Uzay Macar, Neel Nanda, and Arthur Conmy. 2025. Thought anchors: Which llm reasoning steps matter? *Preprint*, arXiv:2506.19143.

ByteDance-Seed-Team. 2025. Seed-oss open-source models. https://github.com/ByteDance-Seed/seed-oss.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Jaden Fiotto-Kaufman, Alexander R Loftus, Eric Todd, Jannik Brinkmann, Caden Juang, Koyena Pal, Can Rager, Aaron Mueller, Samuel Marks, Arnab Sen Sharma, Francesca Lucchetti, Michael Ripa, Adam Belfki, Nikhil Prakash, Dmitrii Troitskii, Carla Brodley, Arjun Guha, Jonathan Bell, Byron Wallace, and David Bau. 2024. Nnsight and ndif: Democratizing access to open-weight foundation model internals. *Preprint*, arXiv:2407.14561.

Roee Hendel, Mor Geva, and Amir Globerson. 2023. In-context learning creates task vectors. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9318–9333, Singapore. Association for Computational Linguistics.

IPC. 1998. International planning competition. https://www.icaps-conference.org/competitions. Accessed: August 30, 2026.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Andrew Lee, Lihao Sun, Chris Wendler, Fernanda Viégas, and Martin Wattenberg. 2025. The geometry of self-verification in a task-specific reasoning model. *Preprint*, arXiv:2504.14379.

Neel Nanda and Joseph Bloom. 2022. Transformerlens.

Neel Nanda, Andrew Lee, and Martin Wattenberg. 2023. Emergent linear representations in world models of self-supervised sequence models. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 16–30, Singapore. Association for Computational Linguistics.

Christopher Olah. 2024. What is a linear representation? what is a multidimensional feature? *Transformer Circuits Thread*.

Core Francisco Park, Andrew Lee, Ekdeep Singh Lubana, Yongyi Yang, Maya Okawa, Kento Nishi, Martin Wattenberg, and Hidenori Tanaka. 2025. Iclr: In-context learning of representations. *Preprint*, arXiv:2501.00070.

Qwen Team. 2025. Qwq-32b: Embracing the power of reinforcement learning.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *Preprint*, arXiv:2408.03314.

Eric Todd, Millicent Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. 2024. Function vectors in large language models. In *The Twelfth International Conference on Learning Representations*.

Karthik Valmeekam, Matthew Marquez, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. 2023. Planbench: An extensible benchmark for evaluating large language models on planning and reasoning about change. *Preprint*, arXiv:2206.10498.

Karthik Valmeekam, Kaya Stechly, and Subbarao Kambhampati. 2024. Llms still can't plan; can lrms? a preliminary evaluation of openai's o1 on planbench. *Preprint*, arXiv:2409.13373.

Constantin Venhoff, Iván Arcuschin, Philip Torr, Arthur Conmy, and Neel Nanda. 2025. Understanding reasoning in thinking language models via steering vectors. *Preprint*, arXiv:2506.18167.

Jake Ward, Chuqiao Lin, Constantin Venhoff, and Neel Nanda. 2025. Reasoning-finetuning repurposes latent representations in base models. *Preprint*, arXiv:2507.12638.

Fengli Xu, Qianyue Hao, Zefang Zong, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, Tianjian Ouyang, Fanjin Meng, Chenyang Shao, Yuwei Yan, Qinglong Yang, Yiwen Song, Sijian Ren, Xinyuan Hu, Yu Li, Jie Feng, Chen Gao, and Yong Li. 2025. Towards large reasoning models: A survey of reinforced reasoning with large language models. *Preprint*, arXiv:2501.09686.

A Potential Risks

This work studies controlled representational interventions for mechanistic analysis, not deployment-oriented model-control methods. The main potential risk is that steering-style interventions could be over-generalized as practical model-control techniques. We mitigate this by using synthetic obfuscated reasoning tasks, reporting aggregate behavioral effects, and framing the interventions as diagnostic probes of whether task representations matter for later reasoning.

Scientific artifacts and licenses. We use public or cited scientific artifacts, including BlocksWorld and PlanBench task formats, GSM8K-style examples, public model checkpoints or APIs, and inference and interpretability software. PlanBench/LLMs-Planning, GSM8K, TransformerLens, and NNSight are MIT-licensed; vLLM, QwQ-32B, Qwen2.5-32B base/instruct, and Seed-OSS-36B base/instruct are Apache-2.0-licensed. DeepSeek-R1-Distill-Qwen-32B and DeepSeek-R1-Distill-Llama-70B are released under the MIT License, with the model cards also noting Apache-2.0 terms for the Qwen-derived model and Llama 3.3 terms for the Llama-derived model. Llama-3.3-70B-Instruct is used under the Llama 3.3 Community License, Nemotron-49B-instruct under the NVIDIA Open Model License with Llama 3.3 terms, and GPT-4.1 through the OpenAI API terms. The corresponding creators are cited in the main text, related work, and implementation details. We use all artifacts only for academic research evaluation and mechanistic analysis, strictly follow their intended use and license terms, and do not redistribute model weights or package the artifacts as deployable systems.

Data content and documentation. The data consists of synthetic planning states, symbolic action and predicate vocabularies, GSM8K-style math-word-problem variants, model-generated traces, and aggregate evaluation results. The tasks are in English and synthetic symbolic domains; they do not represent demographic groups and do not contain private user data or newly collected human-subject data. The generated task text and obfuscation vocabularies use synthetic symbols rather than real-person names, and we checked them by construction and manual inspection for identifying or offensive content.

Data statistics. The main text and appendix report the relevant dataset and evaluation statistics. Mystery BlocksWorld uses 300 four-block puzzles and 15 primary obfuscation namings for QwQ-32B analyses, with naming 3 excluded from representation analyses because it bypasses the intended adaptation process. Cross-domain C1 analyses use Logistics and GSM8K-Renamed variants with the number of namings and traces reported in Table 2. Steering vectors are constructed from 40 correctly solved puzzles and evaluated on 100 held-out four-block puzzles per naming; additional hyperparameters and splits are reported in Section H.

Computational experiments. All computational experiments are inference-only: we do not train or fine-tune LLMs. Model families and sizes are reported in the main text and model names where applicable. Sections H and L report representation extraction windows, timestamps, layers, steering windows, steering scales, decoding settings, package versions, and implementation details. Exact total GPU-hour accounting is not used to support any claim in the paper; the inference and intervention stack is vLLM v0.7.3 with PyTorch forward hooks.

Descriptive statistics. We report accuracies, token counts, cosine similarities, mean steering improvements, standard errors, and paired significance tests where appropriate. Figures and tables specify whether values are means, held-out accuracies, cosine similarities, or best per-family/cell measurements.

Human subjects. This work does not use human subjects, crowdworkers, annotators, or newly col-

lected human-subject data.

B AI Assistant Use

AI assistants were used for language polishing, LaTeX editing, and code-writing support. The authors checked the final manuscript, experimental claims, numerical results, and citations, and AI assistants are not listed as authors.

C BlocksWorld Prompt Example

BlocksWorld system prompt

```
I am playing with a set of blocks
    where I need to arrange the
    blocks into stacks. Here are the
     actions I can do

Pick up a block
Unstack a block from on top of
    another block
Put down a block
Stack a block on top of another
    block

I have the following restrictions on
     my actions:
I can only pick up or unstack one
    block at a time.
I can only pick up or unstack a
    block if my hand is empty.
I can only pick up a block if the
    block is on the table and the
    block is clear.
A block is clear if the block has no
     other blocks on top of it and
    if the block is not picked up.
I can only unstack a block from on
    top of another block if the
    block I am unstacking was really
     on top of the other block.
I can only unstack a block from on
    top of another block if the
    block I am unstacking is clear.
Once I pick up or unstack a block, I
     am holding the block.
I can only put down a block that I
    am holding.
I can only stack a block on top of
    another block if I am holding
    the block being stacked.
I can only stack a block on top of
    another block if the block onto
    which I am stacking the block is
     clear.
Once I put down or stack a block, my
     hand becomes empty.
Once you stack a block on top of a
    second block, the second block
    is no longer clear.

Here is an example problem:

[STATEMENT]

As initial conditions I have that,
    Block B is clear, Block C is
    clear, the hand is empty, Block
    C is on top of Block A, Block A
    is on the table, Block B is on
    the table.
My goal is to have that Block A is
    on top of Block C and Block B is
     on top of Block A

My plan is as follows:

[PLAN]
unstack Block C from on top of Block
     A
put down Block C
pick up Block A
stack Block A on top of Block C
pick up Block B
stack Block B on top of Block A
[PLAN END]
```

D Mystery Prompt Example

Mystery BlocksWorld system prompt (naming 1)

```
I am playing with a set of objects.
    Here are the actions I can do:
   Attack object
   Feast object from another object
   Succumb object
   Overcome object from another
       object

I have the following restrictions on
     my actions:
    To perform Attack action, the
        following facts need to be
        true: Province object,
        Planet object, Harmony.
    Once Attack action is performed
        the following facts will be
        true: Pain object.
    Once Attack action is performed
        the following facts will be
        false: Province object,
        Planet object, Harmony.
    To perform Succumb action, the
        following facts need to be
        true: Pain object.
    Once Succumb action is performed
         the following facts will be
         true: Province object,
        Planet object, Harmony.
    Once Succumb action is performed
         the following facts will be
         false: Pain object.
    To perform Overcome action, the
        following needs to be true:
        Province other object, Pain
        object.
    Once Overcome action is
        performed the following will
         be true: Harmony, Province
        object, Object Craves other
        object.
```

```
    Once Overcome action is
        performed the following will
         be false: Province other
        object, Pain object.
    To perform Feast action, the
        following needs to be true:
        Object Craves other object,
        Province object, Harmony.
    Once Feast action is performed
        the following will be true:
        Pain object, Province other
        object.
    Once Feast action is performed
        the following will be false
        :, Object Craves other
        object, Province object,
        Harmony.

Here is an example problem:
[STATEMENT]
As initial conditions I have that,
    province Block B, province Block
     C, harmony, Block C craves
    Block A, planet Block A, planet
    Block B.
My goal is to have that Block A
    craves Block C and Block B
    craves Block A.
My plan is as follows:
[PLAN]
feast Block C from Block A
succumb Block C
attack Block A
overcome Block A from Block C
attack Block B
overcome Block B from Block A
[PLAN END]
```

E Behavior Analysis

Through manual inspection of DeepSeek-R1-Distill-Qwen-32B and QwQ-32B traces, we identified recurring behavioral patterns in Mystery BlocksWorld solving. LLMs begin with **comparative analysis**, examining initial and goal states to identify conflicting predicates. They then alternate between **recursive search** (working backwards from goals to identify required actions) and **exploration** (experimenting with actions to discover achievable states). These exploratory behaviors occupy the first half of reasoning traces. The second phase involves **plan formulation**, where LLMs construct action sequences and verify validity, iteratively rebuilding when conflicts arise. The final phase consists of **plan verification**, where LLMs validate solutions before committing to answers.

F Layer-wise PCA

fig. 8 contains layer-wise PCA for action representations of different LLMs. The clustering patterns become more pronounced in deeper layers, with clear separation between action types emerging around layers 20–30.

G Cross-LLM Similarity Analysis

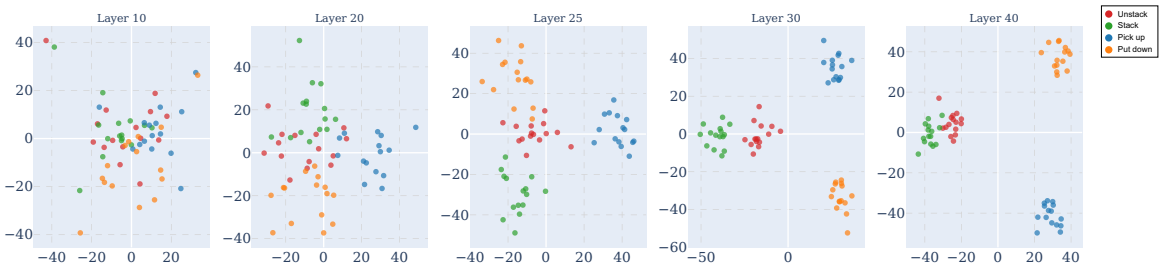
To validate that representational convergence is not unique to QwQ-32B, we analyzed action and predicate representations across multiple LLMs. Figures 9a, 9b, 10a and 10b show how centered action and predicate representations from different timestamps converge toward cross-naming average representations extracted at 7k tokens, analogous to the analysis in Section 4.2.

H Hyperparameters and Experimental Configuration

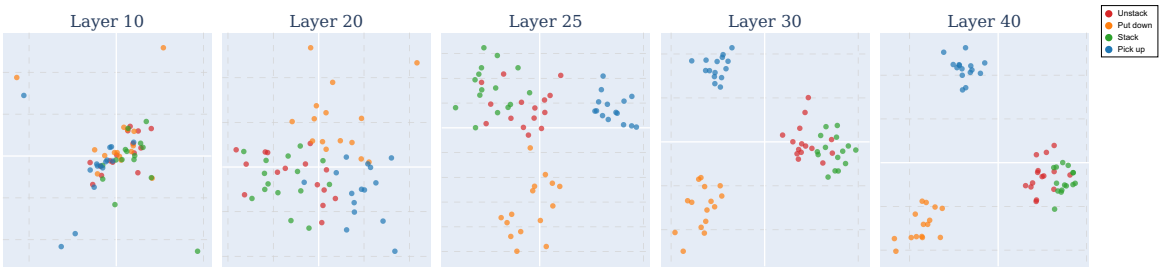
This section summarizes the hyperparameters and experimental settings used throughout our analysis.

H.1 Representation Collection

- **Phrase-matching window**: 100 tokens before each timestamp.
- **Extraction stride and reported timestamps**: Hidden states are sampled every 200 tokens; reported analyses use 2k, 4k, 7k, and 10k.
- **Batch size** (b): Number of reasoning traces per batch for representation extraction. We use 40 correctly solved traces for steering-vector construction and reserve 100 held-out puzzles for steering evaluation.
- **Layers analyzed**: All layers from 0 to LLM depth
- **Number of puzzles for representation collection**: 40 correctly solved puzzles for steering vectors, full dataset for general analysis
- **Layer selection for representation analysis**: We selected layer 40 because action representations have largely separated by this point and change only modestly in later layers. We also performed the analysis at layer 30 and observed no qualitatively different conclusions.

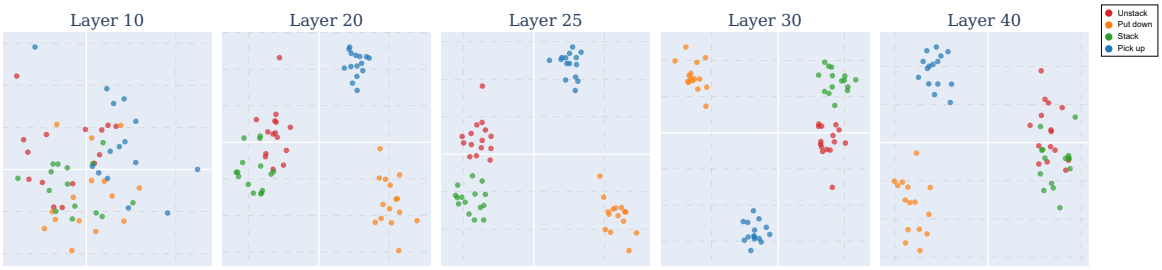


(a) QwQ

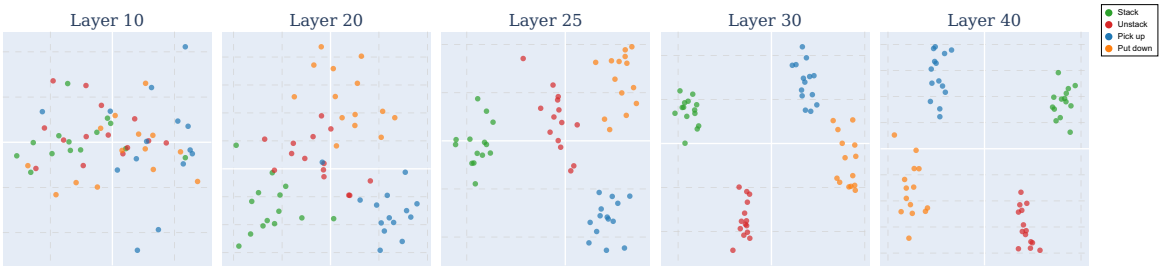


(b) DeepSeek-R1-Distill-Qwen-32B

Figure 7: Layer-wise action PCA at 7k tokens.

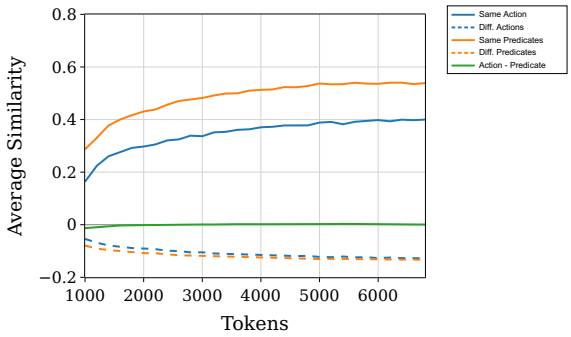


(a) Nemotron-49B-instruct

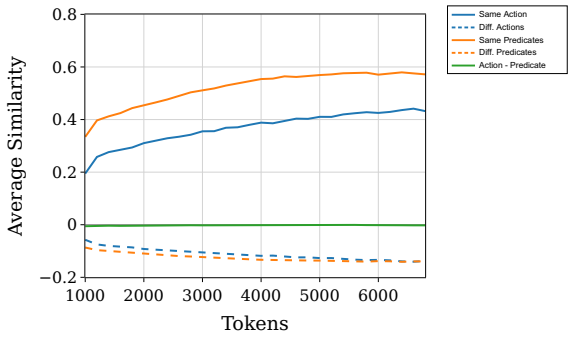


(b) Seed-OSS-36B-instruct

Figure 8: Layer-wise action PCA at 7k tokens.

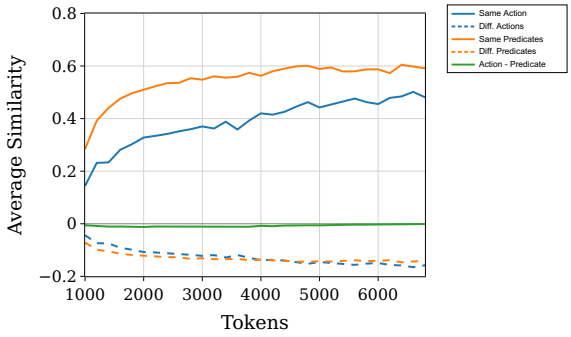


(a) DeepSeek-R1-Distill-Qwen-32B.

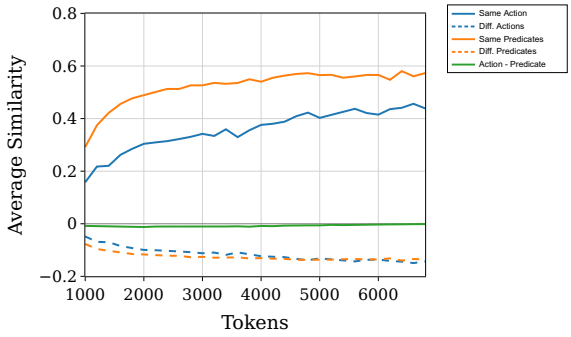


(b) Nemotron-49B-instruct.

Figure 9: Similarities with cross-naming average representations.



(a) Seed-OSS-36B-instruct.



(b) Seed-OSS-36B-base (ByteDance-Seed-Team, 2025).

Figure 10: Similarities with cross-naming average representations.

H.2 Positive Steering

- **Steering scale** (s): $\frac{2}{3}$ (selected after sweep, see Figure 12a)
- **Steering window**: $[t_{\text{start}}, t_{\text{end}}) = [1500, 2500]$ tokens
- **Layers tested**: 1, 5, 10, 20, 30, 40, 50, 60
- **Intervention dataset**: 100 held-out 4-block problem rollouts
- **Steering vector extraction timestamp**: 7k tokens (for layer-wise analysis)

H.3 Symbolic Patching

- **Patching window**: $[2000, 4000]$ tokens
- **End layers tested**: Multiple layers up to selected end layer
- **Scaling factors** (s): $\{10, 20\}$
- **Symbolic representation construction**: $\mathbf{r}_{\text{symbolic}}[a] = \mathbf{r}_{\text{mean}} + s \cdot \mathbf{r}_a$
- **Control condition**: Shuffled symbolic representations (random permutation)

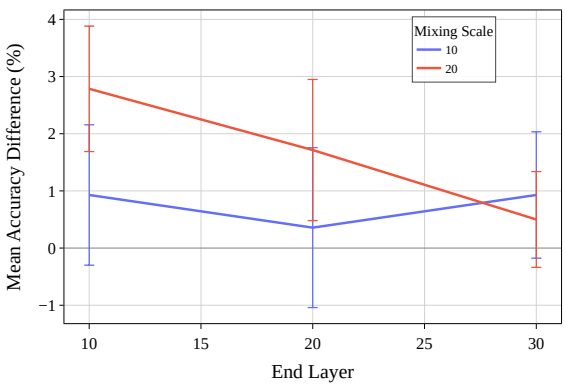


Figure 11: **Symbolic patching.** Mean $\text{Acc}_{\text{symbolic}} - \text{Acc}_{\text{shuffled}}$ across scales and end layers; error bars show standard error.

H.4 Negative Steering

- **Intervention window**: $[2000, 4000]$ tokens
- **Representation extraction timestamp**: 4k tokens
- **Start layer**: 10
- **End layers tested**: 20, 30
- **Control condition**: Shuffled centered naming representations

H.5 LLM Inference

- **Decoding strategy**: Greedy decoding
- **Maximum sequence length**: 24,576 tokens
- **Temperature**: 0 (greedy)
- **Implementation**: vLLM v0.7.3 with PyTorch forward hooks

H.6 Dataset Configuration

- **Number of puzzles**: 300 four-block BlocksWorld puzzles
- **Number of mystery namings**: 15 (primary experiments), 20 (including additional variants)
- **Naming 3 exclusion**: Excluded from representational analyses (recognized as BlocksWorld)
- **Train/test split**: 40 correctly solved puzzles for steering vector extraction, 100 held-out puzzles for steering evaluation

I Negative Steering

To further validate the causal role of FRRs, we conduct an ablation experiment testing whether disrupting representational adaptations decreases accuracy. Since steering interventions can easily degrade performance through general disruption rather than targeted ablation, we use a comparative approach.

We perform interventions across token window $[2000, 4000]$ on multiple layers, subtracting centered in-naming representations extracted from the 4k timestamp. We use shuffled in-naming representations as the control because they preserve activation statistics while breaking action-vector correspondence. Starting at layer 10, end layer 20 degrades accuracy by $2.3\% \pm 0.99\%$ relative to the shuffled control, and end layer 30 degrades accuracy by $2.9\% \pm 1.06\%$.

J Mystery BlocksWorld Naming Variants

K Mystery Performance Analysis

Table 5 reveals several patterns supporting our hypothesis that semantic coherence impedes abstraction. Namings with coherent alternative domains (Mystery 10: gardening, Mystery 11: legal proceedings, Mystery 16: reversible operations) achieve

Table 3: Action Mappings Across Mystery Namings

Naming	pick up	put down	stack	unstack
Mystery 1	attack	succumb	overcome	feast
Mystery 2	illuminate	silence	distill	divest
Mystery 3	tltezi	jchntg	deesdu	xavirm
Mystery 4	swim	fire	deduct	respond
Mystery 5	whisper	calculate	orbit	navigate
Mystery 6	decode	hibernate	thunder	quench
Mystery 7	explore	ripen	weave	bloom
Mystery 8	harvest	ignite	carve	suspend
Mystery 9	construct	demolish	reinforce	collapse
Mystery 10	plant	harvest	nurture	prune
Mystery 11	prosecute	acquit	testify	appeal
Mystery 12	broadcast	receive	encrypt	decode
Mystery 13	whisper	banish	entangle	unmask
Mystery 14	question	resolve	interweave	liberate
Mystery 15	summon	dismiss	fold	unravel
Additional Naming Variants				
Mystery 16	open	close	connect	disconnect
Mystery 17	chop	serve	season	taste
Mystery 18	release	grasp	separate	combine
Mystery 19	transcend	sublimate	actualize	deconstruct
Mystery 20	flixate	grample	chonder	sprill

the lowest base accuracies, while abstract or semantically incoherent combinations (Mystery 2, Mystery 13) enable superior performance.

The steering improvement data shows notable heterogeneity across namings. The maximum improvements reported here represent the best performance across layers 20, 30, 40, and 50, as different namings exhibit optimal responsiveness at different depths. Some namings (Mystery 5, Mystery 11) show minimal or no improvement from in-naming steering, while others (Mystery 1, Mystery 10) demonstrate substantial gains. This suggests that certain semantic structures are more amenable to representational refinement than others.

L Implementation Details

L.1 Steering engine

We implement steering using PyTorch forward hooks on top of vLLM v0.7.3 (Kwon et al., 2023), which provides substantial performance improvements, reducing experiment runtimes from several hours to tens of minutes compared to alternatives like TransformerLens (Nanda and Bloom, 2022) or NNSight (Fiotto-Kaufman et al., 2024). However, this approach has tradeoffs: we must be mindful of cache recomputations, since they recompute representations without steering interventions, and vLLM's optimizations introduce some numerical instability during extended reasoning traces. To address this instability, we run experiments across multiple naming variants. All experiments use greedy decoding with maximum sequence length of 24,576 tokens.

L.2 Hyperparameters

We perform positive steering as described in Section 5.1 on layer 20 using **in-naming** representations to determine the optimal steering scale. Figure 12a shows that scales $\frac{2}{3}$ and $\frac{4}{5}$ have similar effects. While improvement from $\frac{4}{5}$ is slightly higher, we chose $\frac{2}{3}$ since it has a more stable effect on all namings.

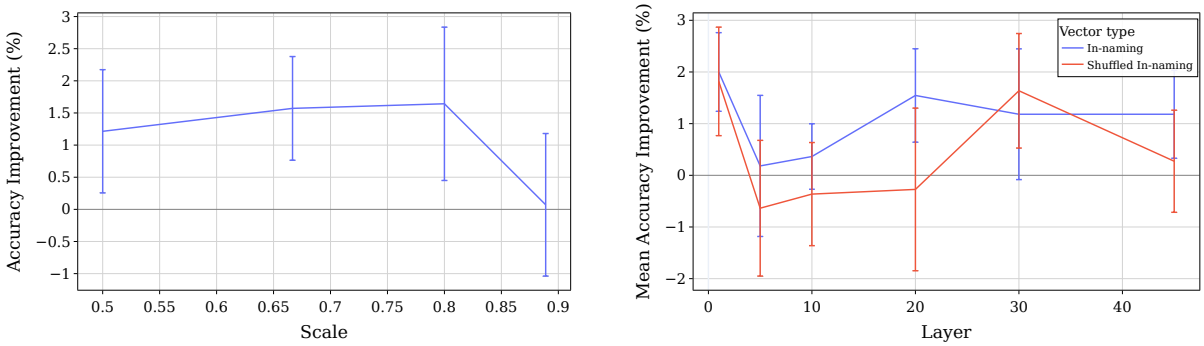
M Shuffled In-Naming Steering Controls

Setup. In addition to positive steering (Section 5.1), we tested a *shuffled in-naming* control. For each naming, we applied a single consistent permutation of centered in-naming vectors across actions/predicates (e.g., `pick up`→`stack`, `stack`→`put down`, etc.). This preserves per-naming distributional statistics while breaking the action–representation alignment. Interventions used the same window $[1500, 2500]$, scale $s = \frac{2}{3}$, and norm-preserving update rule as before, run on a reduced subset of layers and namings.

Findings and interpretation. Figure 12b shows a three-phase trend: (i) early layers (≤ 5) improve relative to baseline, consistent with disruption of surface semantics; (ii) middle layers ($\sim$5–20) degrade performance, likely breaking emerging abstractions; and (iii) later layers ($\geq$30) again show improvements, comparable to unshuffled *in-naming*. Early/mid results align with our story: disruption

Table 4: Predicate Mappings Across Mystery Namings

Naming	ontable	clear	handempty	holding	on
Mystery 1	planet	province	harmony	craves	pain
Mystery 2	aura	essence	nexus	harmonizes	pulse
Mystery 3	oxtslo	adohre	jqlyol	gszswg	ivbmyg
Mystery 4	fever	marble	craving	mines	shadow
Mystery 5	crystal	fountain	autumn	illuminates	legend
Mystery 6	prism	hollow	zenith	echoes	emblem
Mystery 7	fossil	dialect	equinox	fractures	symphony
Mystery 8	nebula	labyrinth	mirage	captivates	cascade
Mystery 9	eclipse	vintage	paradox	resonates	twilight
Mystery 10	crystal	puzzle	vortex	whispers	cipher
Mystery 11	nebula	molecule	anthem	silhouettes	voltage
Mystery 12	horizon	compass	solstice	orbits	quantum
Mystery 13	tethered	unburdened	hollow	shrouds	consuming
Mystery 14	echoing	sovereign	potential	obscures	contemplating
Mystery 15	suspended	timeless	interval	transcends	enveloping
Additional Naming Variants					
Mystery 16	paired	single	balanced	matches	mirrors
Mystery 17	plated	fresh	kitchen	simmering	marinated
Mystery 18	floating	occupied	crowded	repels	avoids
Mystery 19	phenomenal	unmediated	dialectical	instantiates	necessitates
Mystery 20	morkled	thristy	plimmish	vexates	quorbles



(a) Layer-20 scale sweep.

(b) Shuffled in-naming control.

Figure 12: Additional steering controls and sweeps.

helps initially, but permutations harm once action-specific representations form. Late-layer gains suggest action vectors contain shared structural components, so even mismatched but in-manifold vectors can occasionally assist; the runs therefore serve as a targeted control for action-vector correspondence under the same steering rule.

N Statistical Analysis of Steering Effects

We conducted statistical tests to validate that steering with refined representations improves accuracy over baseline. Our analysis uses one-sample t-tests treating each mystery naming as an independent observation.

N.1 Test Methodology

Data Structure. Our experiments evaluate steering across 14 mystery namings (excluding naming 3) with approximately 100 puzzles per naming (indices 200–300). Each mystery naming provides accuracy measurements under multiple conditions: baseline (no steering), in-naming steering, cross-naming steering, and random Gaussian steering at various layers.

Statistical Test. We employ one-sample t-tests to assess whether mean accuracy improvements across mystery namings significantly exceed zero. Each mystery naming serves as an independent observation, with the null hypothesis $H_0 : \mu_{\text{improvement}} = 0$. We use one-tailed tests since we hypothesize positive improvements. The test statistic is:

$$t = \frac{\bar{\Delta}}{\text{SE}(\Delta)} = \frac{\bar{\Delta}}{s_\Delta/\sqrt{n}} \tag{7}$$

where $\bar{\Delta}$ is the mean improvement across $n =$

Table 5: Mystery BlocksWorld performance and maximum steering improvement by naming. "–" indicates variants not used in the steering sweep.

Naming Variant	Base Acc.	In-Naming Steering	Cross-Naming Steering	Semantic Description
Mystery 1	0.33	+0.10	+0.11	Mixed violent/consumption metaphors
Mystery 2	0.47	+0.05	+0.05	Abstract mystical/spiritual terms
Mystery 3	0.65	–	–	Random strings
Mystery 4	0.25	+0.03	+0.03	Mixed physical actions
Mystery 5	0.24	-0.01	+0.01	Communication/navigation metaphors
Mystery 6	0.26	+0.05	+0.07	Technical/elemental operations
Mystery 7	0.19	+0.02	+0.03	Nature/growth cycle
Mystery 8	0.11	+0.02	+0.04	Agriculture/crafting metaphors
Mystery 9	0.25	+0.02	+0.01	Construction/destruction cycle
Mystery 10	0.05	+0.09	+0.05	Coherent gardening domain
Mystery 11	0.14	+0.00	+0.02	Legal proceedings domain
Mystery 12	0.16	+0.06	+0.02	Communication technology
Mystery 13	0.48	+0.06	+0.06	Dark mystical operations
Mystery 14	0.24	+0.02	+0.02	Abstract philosophical inquiry
Mystery 15	0.34	+0.04	+0.04	Mystical summoning/manipulation
Additional Variants				
Mystery 16	0.05	–	–	Reversible operations (open/close)
Mystery 17	0.27	–	–	Coherent cooking domain
Mystery 18	0.07	–	–	Physical manipulation verbs
Mystery 19	0.33	–	–	Abstract philosophical concepts
Mystery 20	0.29	–	–	Complete nonsense words

14 mystery namings, s_Δ is the sample standard deviation, and $\mathrm{SE}(\Delta)$ is the standard error.

N.2 Results

Table 6 shows that several steering conditions produce statistically significant improvements over baseline. Layer 20 in-naming ($p = 0.042$), layer 20 cross-naming ($p = 0.044$), and layer 40 cross-naming ($p = 0.021$) all achieve significance at $\alpha = 0.05$, with mean improvements ranging from 1.4% to 1.8%.

Random Gaussian steering at layer 20 shows negative mean improvement (-0.36%, $p = 0.627$), confirming that improvements from structured representations are not artifacts of random perturbations. Cross-naming representations show numerically higher improvements than in-naming representations at all tested layers, with the strongest and most consistent effects observed at layer 40 where cross-naming achieves 1.43% improvement ($p = 0.021$) compared to 0.57% for in-naming ($p = 0.274$).

N.3 Discussion

Our statistical analysis provides evidence that steering with refined representations improves accuracy over baseline, with the strongest effects observed at layers 20 and 40 for cross-naming steering. The significant improvements at multiple layers, combined with the lack of improvement from random Gaussian controls, support our hypothesis that representational adaptations during reasoning contain meaningful structural information that causally contributes to problem-solving performance.

The moderate effect sizes (1.4–1.8% for significant conditions) and variability across mystery namings reflect the challenging nature of using steering with long reasoning rollouts.

Table 6: Steering accuracy improvements across 14 mystery namings. SE = standard error; * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$, ns = not significant.

Condition	**Mean Δ**	**SE**	**t-statistic**	**p-value**	**Sig.**
Layer 20 In-naming	1.57%	0.84%	1.878	0.042	*
Layer 20 Cross-naming	1.79%	0.97%	1.846	0.044	*
Layer 20 Random Gaussian	-0.36%	1.08%	-0.332	0.627	ns
Layer 30 In-naming	0.64%	1.29%	0.500	0.313	ns
Layer 30 Cross-naming	1.36%	1.20%	1.133	0.139	ns
Layer 40 In-naming	0.57%	0.92%	0.618	0.274	ns
Layer 40 Cross-naming	1.43%	0.64%	2.249	0.021	*
Layer 50 In-naming	0.36%	1.01%	0.352	0.365	ns
Layer 50 Cross-naming	0.93%	0.61%	1.531	0.075	ns