LATENT TRAJECTORY STEERING UNLOCKS TRAPPED CAPACITY IN RECURSIVE REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Recursive reasoning models such as HRM and TRM achieve strong performance on structured reasoning tasks with far fewer parameters than LLMs, yet their inference dynamics are not well understood. We analyze their latent trajectories from a dynamical-systems perspective and find distinct stable basins for successful and failed rollouts, with high-stretching finite-time Lyapunov exponent (FTLE) ridges concentrated near transitions between them. Based on this observation, we propose a training-free inference method that identifies these ridge locations along the unperturbed trajectory, applies controlled perturbations to induce basin crossings, and aggregates the resulting trajectories through Gibbs-weighted selection. Without updating model parameters or using ground-truth labels, our method improves the accuracy of TRM in Sudoku-Extreme from 87.4% to 98.43% and in Maze from 85.3% to 91.0%, with consistent gains in HRM. These results identify dynamical trapping as a major failure mode in recursive reasoning: redirecting a failed latent trajectory across a high-FTLE boundary can recover the correct solution without retraining the model. Compared to existing alternatives, such as PTRM, our method locates a more precise point of intervention in the latent reasoning trajectory, such that much fewer rollouts and augmentation are needed.

1 INTRODUCTION

Recent advances in large language model (LLM) reasoning span inference-time scaling, such as Chain-of-Thought and search-based verification (Wei et al., 2022), training-time behavior optimization via reinforcement learning and supervised fine-tuning (Jaech et al., 2024; Guo et al., 2025; Yang et al., 2025), and training-time scaling through larger models and more data. However, these approaches remain brittle on tasks requiring long-range propagation of constraints and compositional generalization, such as Sudoku, Maze, and ARC-AGI, where reasoning must maintain global consistency over many dependent decisions. A key limitation is that token-level reasoning exposes intermediate computation to cascading errors and often relies on human-annotated reasoning traces (Zhu et al., 2025; Deng et al., 2024; Hao et al., 2025). In contrast, *latent reasoning* performs multi-step computation through internal iterations in a continuous state space, offering a path toward complex structured reasoning with substantially smaller parameter budgets.

Two representative models in this direction are the Hierarchical Reasoning Model (HRM) (Wang et al., 2025a; 2026) and the Tiny Recursive Model (TRM) (Jolicoeur-Martineau, 2025). HRM employs two recurrent modules operating at different temporal scales, while TRM shows that a smaller shared network with recursive updates can achieve even stronger generalization. These results suggest that weight-tied iterative computation is a central mechanism for achieving effective reasoning depth in compact models.

This iterative computation naturally defines a discrete dynamical system in latent space. At each reasoning step, HRM/TRM repeatedly applies a shared update operator, which can be written abstractly as $z_{k+1} = f_\theta(z_k; x)$. Under this view, recursive reasoning corresponds to the evolution of a latent state toward stable structures such as fixed points or attractors. Studying the dynamical properties of this evolution is therefore essential for understanding not only whether these models solve a task, but also how and why their reasoning trajectories succeed or fail.

Despite their empirical success and natural formulation as dynamical systems, the internal dynamics of these models remain insufficiently characterized. Recent dynamical-systems analyses (Ren

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.

& Liu, 2026) show that HRM trajectories often collapse to fixed points, whether they are correct or spurious. Huang et al. (2026) further show that training can widen correct basins. For TRM, Sghaier et al. (2026) show that deterministic recursion can converge to suboptimal solutions without an escape mechanism, and that stochastic perturbations can explore alternative solution basins. Together, these results identify basin trapping as a recurring failure mode in recursive inference and motivate explicitly targeting the geometry of basin transitions.

Existing diagnostic methods characterize recursive dynamics through fixed-point residuals and perturbation-based analyses of attractors (Ren & Liu, 2026; Huang et al., 2026; Fein-Ashley & Rashidinejad, 2026), while dimensionality reduction provides a complementary view of latent-state geometry (Ning & Rangaraju, 2025). We introduce the finite-time Lyapunov exponent (FTLE) to localize instability along the recursive trajectory and identify candidate transition regions between attractor basins. Lyapunov-based methods have been used to analyze neural dynamics (Engelken et al., 2023; Vogt et al., 2024; Zheng & Shlizerman, 2025); FTLE localizes this analysis to finite horizons, and recent work has shown that high-stretching FTLE structures can reveal critical regions in neural networks (Storm et al., 2024; Kuehn & Wöhrer, 2026; Vock & Meisel, 2026; Liu et al., 2025; Muruganantham et al., 2026). Complementarily, Lagrangian coherent structure (LCS) theory suggests that separatrices between basins of attraction can manifest as repelling LCSs, which are often diagnosed by ridges of the FTLE field (Haller, 2011; Shadden et al., 2005; Beneitez et al., 2020; Haller & Sapsis, 2011).

Building on this view, we propose a *training-free* guided steering method for pretrained HRM/TRM checkpoints: compute FTLE along the unperturbed inference trajectory, identify locally unstable regions, and apply targeted perturbations at those steps to encourage escape from spurious basins and transition toward favorable attractors. By targeting a high-FTLE region instead of perturbing throughout recursive inference, our method concentrates computation where basin switching is most likely. This substantially reduces the rollout budget: we use $K = 50$ rollouts on both tasks, compared with $K = 100$ for PTRM, while also reducing the ACT budget from 64 to 16 on Sudoku-Extreme.

Experiments on Sudoku-Extreme and Maze demonstrate substantial accuracy gains for HRM/TRM without retraining (Fig. 1). In addition, on ARC-AGI-2, TRM accuracy improves from 6.7% to 8.3%. These results indicate that FTLE-guided steering can unlock latent capacity already encoded in the trained model.

In summary, our contributions are fourfold: (i) We formalize recursive inference in HRM/TRM as a latent dynamical system and introduce FTLE as a diagnostic for locally unstable regions near basin boundaries. (ii) We introduce a training-free, FTLE-guided perturbation method that steers trained HRM/TRM across LCS-like basin boundaries to improve their performance by releasing the untapped capacity. (iii) We provide mechanistic evidence that FTLE peaks align with unstable transition regions between successful and failed recursive trajectories. (iv) We validate the method on HRM and TRM across Sudoku-Extreme, Maze-Hard, Pencil Puzzle Bench(PPBench) and ARC-AGI-2 datasets throughexperiments, ablations and visualizations.

2 Background and Related Work

2.1 Latent and Recursive Reasoning Models

Latent reasoning replaces explicit token-level traces with internal computation, including implicit, continuous, and compressed chain-of-thought representations (Zhu et al., 2025; Deng et al., 2024; Hao et al., 2025; Cheng & Van Durme, 2024; Shen et al., 2025; Geiping et al., 2025). Within this methods, HRM and TRM achieve strong performance on reasoning tasks with less parameters (Wang et al., 2025a; Jolicoeur-Martineau, 2025). Their combination of parameter efficiency and strong reasoning ability motivates our study of their inference dynamics.

For our purposes, these architectural details can be summarized by a unified discrete-time latent dynamical system. Let s_t denote the recurrent latent state at recursive step t. The inference process can be written as

$$s_{t+1} = F_\theta(s_t; x), \tag{1}$$

$$\hat{y}_t = D_\theta(s_t), \tag{2}$$

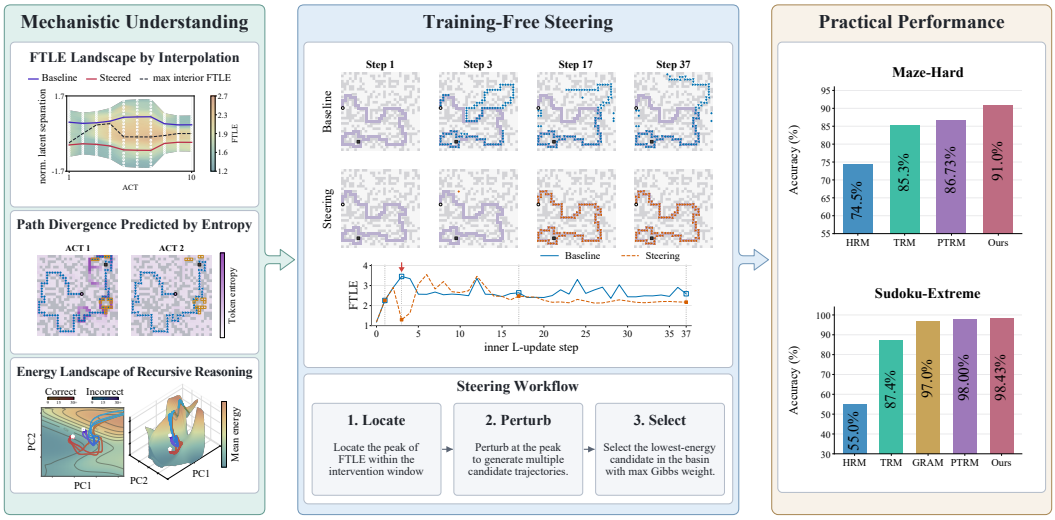


Figure 1: Overview of the proposed FTLE-guided, training-free inference framework. Perturbations steer failed HRM/TRM trajectories from spurious basins toward correct solutions; energy and entropy diagnostics reveal attractor separation and early uncertainty. The interpolation between correct and spurious reasoning trajectories characterizes the landscape. Without retraining, our method (based on TRM) improves Sudoku-Extreme and Maze-Hard accuracy over baselines.

where F_θ is the input-conditioned transition map determined by model parameters θ, and D_θ reads out token predictions or candidate solutions from the latent state. Our FTLE, basin-boundary, and attractor-steering analysis are conducted under this abstract framework; the original HRM/TRM update equations and their relation to this notation are provided in Appendix C.

2.2 Dynamical Views of Reasoning Models

Recent work has examined recursive reasoning through a perspective of dynamical system. Ren & Liu (2026) identify fixed-point violations and grokking-like transitions in HRM, showing that trajectories may enter incorrect attractors and either occasionally escape or remain trapped. Huang et al. (2026) formalize *attractor learning*: test-time scaling arises from learning task-conditioned attractors, with fixed-point residuals as convergence diagnostics. Jolicoeur-Martineau (2025) introduce deterministic recursive refinement in TRM. Subsequent PTRM work shows that this recursion can converge to suboptimal solutions without an escape mechanism and uses stochastic perturbations to explore alternative solution basins (Sghaier et al., 2026). Studies of looped Transformers reveal cyclic fixed-point behavior and stabilizing attention patterns (Geiping et al., 2025). Inference-time methods increasingly treat reasoning as selecting among candidates (Wei et al., 2022; Jaech et al., 2024; Guo et al., 2025) Probabilistic Tiny Recursive Model (PTRM)(Sghaier et al., 2026) and Generative Recursive Reasoning Models (GRAM)(Baek et al., 2026) take privilege of noise to improve the performance of reasoning models. Awadhiya (2026) adopt an associative memory view, using Gibbs weighting to select robust answers from wider attractor basins.

2.3 FTLE and LCS Diagnostics

The finite-time Lyapunov exponent (FTLE) quantifies exponential amplification of infinitesimal perturbations over finite horizons. In fluid dynamics, FTLE ridges identify Lagrangian coherent structures (LCSs) which organize trajectory patterns and act as transport barriers (Haller, 2015). Haller's variational theory establishes repelling LCSs as surfaces extremizing normal repulsion (Haller, 2011). In multi-attractor systems, basin boundaries can be understood as LCSs: Beneitez et al. (2020) show FTLE ridges coincide with basin boundaries in high-dimensional dissipative systems. Haller & Sapsis (2011) establish duality between largest and smallest FTLEs for repelling and attracting structures. Storm et al. (2024) extend FTLE to deep neural networks, finding that ridges in input space separate class regions and correlate with uncertainty and adversarial susceptibility.

Motivated by evidence that FTLE ridges can trace basin and neural class boundaries, we hypothesize that they may indicate boundaries between correct and spurious attractors in HRM/TRM. Our method identifies candidate separatrices via high FTLE, perturbs latent states to cross them, and selects outputs through an energy-based rollout aggregation.

3 METHODOLOGY

We propose a training-free inference intervention for steering the recursive latent dynamics of pretrained HRM/TRM models. Building on the latent dynamical abstraction in Section 2, our goal is to redirect failed rollouts away from spurious attractor basins without updating model parameters or using ground-truth labels. The method consists of three stages: *FTLE localization*, *stochastic basin steering*, and *Gibbs-weighted basin selection*.

3.1 LCS-INSPIRED FTLE LOCALIZATION

We use the *finite-time Lyapunov exponent (FTLE)* to identify where a recursive trajectory is most sensitive to perturbation (Shadden et al., 2005; Haller & Yuan, 2000). Given a latent state s_t and horizon T, we compose the update maps as

$$\Phi_t^{t+T} = F_{t+T-1} \circ \cdots \circ F_t, \tag{3}$$

and define the local Cauchy–Green tensor

$$C_t(s_t) = \left(D\Phi_t^{t+T}(s_t)\right)^\top D\Phi_t^{t+T}(s_t), \tag{4}$$

where $D\Phi_t^{t+T}(s_t)$ is the Jacobian of the composed map at s_t. The maximal FTLE is

$$\lambda_T(s_t) = \frac{1}{T} \log \sqrt{\lambda_{\max}\big(C_t(s_t)\big)}. \tag{5}$$

Thus, $\lambda_T(s_t)$ measures the finite-horizon amplification rate of infinitesimal perturbations. We estimate it along high-dimensional latent trajectories using Jacobian-vector and vector-Jacobian products, following standard Lyapunov diagnostics (Ott, 1993; Cvitanovic et al., 2005; Hadjighasem et al., 2017); pseudo code is provided in Appendix C.

Inspired by LCS theory, we treat high-FTLE regions as candidate separatrices between basins (Haller, 2011). Because HRM/TRM dynamics are discrete and high-dimensional, FTLE is used only for localization, with perturbations applied near these regions to promote basin crossing.

3.2 FTLE-GUIDED STOCHASTIC BASIN STEERING

Inspired by noise-induced transitions in stochastic dynamical systems (Freidlin & Wentzell, 2012; Börner et al., 2024; Santolin et al., 2025), we inject localized stochastic perturbations at high-FTLE ridges, treated as candidate LCS-like separatrices. Since FTLE measures the local sensitivity of the dynamics to perturbations, steering through high FTLE regions offers a more effective mechanism for basin transitions.

We first run the baseline trajectory, select the highest-FTLE position $t^\star$ within a correction window W, replay the model to $t^\star$, and generate K rollouts by injecting independently sampled Gaussian noise from $t^\star$ through the end of the current recurrent cycle:

$$t^\star = \arg\max_{t \le W} \lambda_T(s_t), \tag{6}$$

$$\tilde{z}_{L,k}^{l+1} = F_L(\tilde{z}_{L,k}^l + \epsilon_{L,k}^l; \tilde{z}_{H,k}, x), \qquad \tilde{z}_{H,k}^+ = F_H(\tilde{z}_{H,k} + \epsilon_{H,k}, \tilde{z}_{L,k}^{\text{end}}). \tag{7}$$

Here, $\epsilon_{L,k}^l$ denotes sustained low-level noise from the selected low step to the end of the current recurrent cycle, while $\epsilon_{H,k}$ denotes a high-level perturbation injected once before the high-level update. This produces candidate rollouts $\{(\hat{y}_k, E_k)\}_{k=1}^K$ without modifying model parameters or accessing labels. The full procedure is given in Appendix C.

3.3 Gibbs-Weighted Basin Selection

Perturbed rollouts may converge to different predictions and differ substantially in trajectory quality. A reliable selection mechanism should incorporate complementary evidence from structural validity, trajectory stability, and transient exploration dynamics. We therefore construct empirical answer basins by grouping rollouts, $B_y = i : \hat{y}_i = y$, and assign each rollout a reasoning trajectory energy. For rollout (i), we define

$$E_i = \underbrace{\boldsymbol{\omega}_v^\top \widetilde{\mathbf{e}}_i^{\,v}}_{\text{task validity}} + \underbrace{\boldsymbol{\omega}_s^\top \widetilde{\mathbf{e}}_i^{\,s}}_{\text{trajectory stability}} + \underbrace{\boldsymbol{\omega}_x^\top \widetilde{\mathbf{e}}_i^{\,x}}_{\text{basin exploration}} . \tag{8}$$

Validity features measure structural consistency (e.g., Sudoku row/column/box violations; Maze connectivity, graph consistency, path-length compactness); stability features measure dynamical convergence (e.g., terminal latent residuals, late-step prediction changes); and uncertainty features measure how sharply a rollout commits to its final prediction. Full task-specific definitions and weights are given in Appendix D.

The trajectory energies are then converted into Gibbs weights $\pi_i = \frac{\exp(-\beta E_i)}{\sum_{j=1}^{K} \exp(-\beta E_j)}$, where β controls the selection sharpness. We aggregate rollout weights at the basin level, $M(B_y) = \sum_{i \in B_y} \pi_i$, $B^\star = \arg\max_{B_y} M(B_y)$, and select the minimum-energy representative from the winning basin, $i^\star = \arg\min_{i \in B^\star} E_i$, $\hat{y}^\star = \hat{y}_{i^\star}$. This yields a soft basin-level voting rule that combines repeated support with trajectory quality. When all rollouts have equal energy, it reduces to standard self-consistency voting (Wang et al., 2023); otherwise, higher-quality rollouts contribute more strongly to the final basin decision.

4 Mechanistic Analysis

4.1 Latent Attractor Landscape and Failure Modes

Having established the methodology, we now turn to the latent dynamics underlying recursive reasoning. We analyze these dynamics through FTLE to explain why training-free steering can improve inference.

Prior work has shown that recurrent or implicit neural computation can be organized around stable fixed points and attractor-like structures (Ceni et al., 2020; Schöne et al., 2025), while looped and test-time recurrent architectures use iterative hidden-state computation for reasoning and sequence modeling (Sun et al., 2025; Saunshi et al., 2025). These perspectives motivate studying inference as a trajectory through a learned recurrent state space (Fein-Ashley & Rashidinejad, 2026; Wang et al., 2025b). Across Maze-TRM checkpoints, Figure 2 show that training gradually shifts the dynamics from early local sensitivity toward stronger long-horizon contraction. This suggests that training shapes the geometry of the recurrent state space in which inference trajectories settle.

This perspective also explains why convergence is not a reliable indicator of correctness. Consistent with observations of trajectories (Ren & Liu, 2026), we observe several failure modes: rollouts may oscillate, enter high-error plateaus, or converge to incorrect attractors. FTLE complements these diagnostics by localizing regions of heightened dynamical sensitivity where perturbations may redirect failed trajectories into alternative basins.



Figure 2: Latent dynamics of HRM/TRM. Cumulative FTLE trends indicate attraction toward stable regions. Checkpoint comparisons show that training progressively organizes the latent landscape.

4.2 FTLE Ridges as LCS-like Basin Boundaries

FTLE is a classical diagnostic for repelling Lagrangian coherent structures (LCSs), whose ridges mark finite-time separation of nearby trajectories (Haller, 2015); however, high FTLE alone is not a proof of an LCS, since other non-hyperbolic effects can also create ridges (Hadjighasem et al., 2017). Following Haller's variational criteria (Haller, 2011), we therefore treat high-FTLE points only as *candidate LCS-like basin boundaries*: for each selected point, we inspect a local two-dimensional latent slice for spectral separation and ridge-normal alignment.

Figure 3 provides two complementary diagnostics. In the local latent slice, the selected high-FTLE region exhibits projected geometric signatures consistent with a repelling cross-section. In the trajectory-space slice, obtained by interpolating between correct and incorrect rollouts at matched ACT steps, the FTLE maximum lies in the interior between the two trajectories rather than at either endpoint. These observations are consistent with a locally high-stretching region separating trajectories that move toward different attractor basins, motivating our use of such ridges as candidate LCS-like basin boundaries.

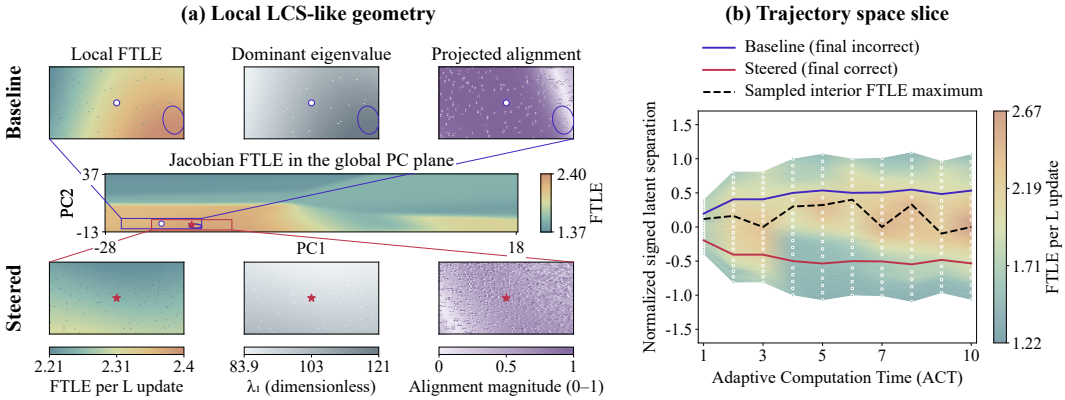


Figure 3: Complementary evidence for an LCS-like basin boundary. (a) A local latent landscape exhibits an FTLE ridge with projected Haller-style diagnostics. (b) A slice across the trajectory space reveals a persistent interior FTLE maximum between correct and incorrect rollouts.

4.3 Gibbs Selection in the Energy Landscape

After perturbation, the sampled rollouts can be interpreted as local probes of neighboring answer basins in the learned latent landscape. Rather than selecting a single rollout, we assign each rollout a label-free trajectory energy that combines model-internal stability signals with task-rule violations. Gibbs weighting then aggregates probability mass at the decoded-answer level, favoring basins that are reached repeatedly by low-energy, rule-consistent trajectories.

Stablemax entropy is used only as an auxiliary uncertainty signal. It identifies token locations where the model remains locally uncertain among competing path hypotheses. As shown in Figure 4, perturbation redistributes early uncertainty toward an alternative path region before the rollout settles. This motivates using entropy as a part of the energy.

Figure 5 illustrates the resulting selection geometry on Maze. Under the proposed label-free energy, corrected trajectories tend to occupy lower-energy regions due to structural constraints and stable terminal dynamics. Failed trajectories are assigned higher energy when they correspond to disconnected paths, graph inconsistencies, or unstable terminal states. Thus, the Gibbs selection stage combines basin-level support with rule-based filtering over the perturbed ensemble.

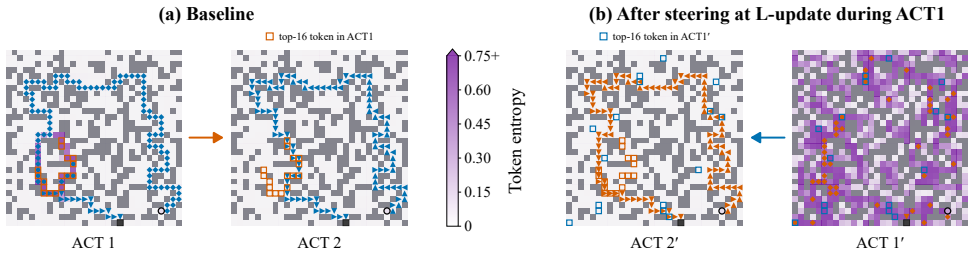


Figure 4: Stablemax entropy before and after FTLE-guided steering at ACT steps 1 and 2. High-entropy tokens indicate locally ambiguous path regions. After perturbation, uncertainty is temporarily elevated and shifts toward an alternative candidate path before the rollout stabilizes and reaches the correct solution.

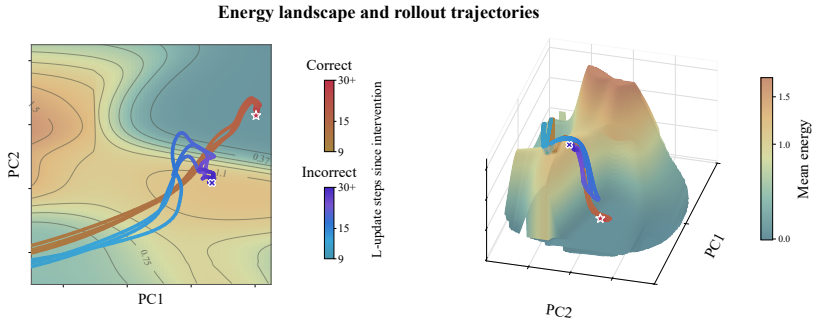


Figure 5: Energy-based basin selection on Maze. The label-free trajectory energy assigns lower scores to structurally valid and dynamically stable rollouts, while Gibbs aggregation favors answer basins supported by multiple low-energy rollouts.

5 Experiments

We evaluate FTLE-guided steering on Maze-hard, Sudoku-Extreme, Pencil Puzzle Bench and ARC-AGI-2 datasets. For each input, our training-free inference computes the baseline FTLE, samples perturbed rollouts from the selected high-FTLE position, and returns the answer chosen by Gibbs-weighted basin aggregation.

5.1 Main Results

Table 1: Comparison with recursive reasoning baselines on Sudoku-Extreme and Maze-Hard. Reported HRM, TRM, GRAM, and PTRM results are shown for reference. On Sudoku-Extreme, the reported PTRM result uses $K = 100$ rollouts and 64 ACT steps, whereas ours uses $K = 50$ and 16 ACT steps. On Maze-Hard, the reported PTRM result uses $K = 100$ rollouts and 16 ACT steps, whereas ours uses $K = 50$ and 16 ACT steps.

Method	Sudoku	Maze
HRM (Wang et al., 2025a)	55.0%	74.5%
TRM (Jolicoeur-Martineau, 2025)	87.4%	85.3%
GRAM (Baek et al., 2026)	97.0%	—
PTRM (Sghaier et al., 2026)	$98.00 \pm 0.13\%$ (98.75% reported)	84.2% (86.73% reported)
Ours (TRM)	**98.43 ± 0.07%**	**91.0%**
Ours (HRM)	$62.48 \pm 0.91\%$	79.4%

Table 1 places our method against prior recursive reasoning models. On Sudoku-Extreme, our result is close to PTRM (98.43% vs. 98.75%) while using fewer rollouts. On Maze-Hard, our method outperforms PTRM by a clear margin (91.00% vs. 86.73%) while using half as many rollouts, demonstrating a clear advantage for targeted FTLE-guided steering under the reported settings.

Table 2: Exact accuracy at representative ACT budgets. The HRM and TRM baselines are reproduced on the same hardware as the corresponding steering runs. Complete ACT sweeps are provided in Appendix F.4.

	Sudoku-Extreme		Maze-Hard		
Method	10 ACT	16 ACT	3 ACT	5 ACT	16 ACT
TRM	85.15 ± 0.25	89.19 ± 0.33	85.20	85.00	85.10
TRM + Steering	$\mathbf{97.90 \pm 0.10}$	$\mathbf{98.43 \pm 0.07}$	**90.40**	**91.00**	**91.00**
HRM	51.77 ± 0.96	55.25 ± 0.85	73.30	73.90	74.40
HRM + Steering	$\mathbf{57.23 \pm 0.77}$	$\mathbf{62.48 \pm 0.91}$	**77.70**	**80.00**	**79.40**

The left panel of table 2 reports Sudoku-Extreme exact accuracy under fixed ACT budgets, including three-seed HRM results and the available TRM runs. The right panel of table 2 reports the corresponding Maze sweep for both architectures. Across tasks and budgets, steering improves over the deterministic rollout without parameter updates.

Table 3 reports the performance improvement on PPBench and ARC-AGI-2. Our method improves the accuracy from 53.1% to 71.4% at 16 ACT steps, a gain of 18.3 percentage points over deterministic decoding and matching PTRM best-Q@100 (Sghaier et al., 2026). Importantly, we use the same TRM checkpoint, reproduce both PTRM and our method, and align the number of rollouts at $K = 100$ in this task.

The ARC-AGI-2 results are reported under matched settings with 100 augmentations and 25 rollouts per augmentation. At 16 ACT steps, FTLE steering matches the augmented TRM baseline and PTRM at pass@1 (4.2%), while improving pass@2 from 4.2% to 5.0% and pass@5 from 6.7% to 8.3%. The unchanged pass@1 but improved pass@5 indicates that FTLE-guided perturbation expands the set of useful candidate basins, although selecting the best candidate remains challenging.

Table 3: PPBench exact accuracy and ARC-AGI-2 accuracy at representative ACT budgets. PPBench uses the golden subset. For ARC-AGI-2, the TRM baseline includes input augmentation, and results are evaluated over all test tasks. PTRM and FTLE steering use same settings. Complete ACT sweeps are provided in Appendix F.4.

	PPBench			ARC-AGI-2			
	Exact accuracy			pass@2		pass@5	
Method	ACT 5	ACT 10	ACT 16	ACT 10	ACT 16	ACT 10	ACT 16
TRM baseline	53.1	53.1	53.1	4.2	4.2	5.8	6.7
PTRM	**69.4**	**71.4**	69.4	4.2	4.2	6.7	6.7
FTLE Steering (Ours)	67.3	**71.4**	**71.4**	5.0	5.0	**8.3**	**8.3**

5.2 Ablation on Intervention and Selection

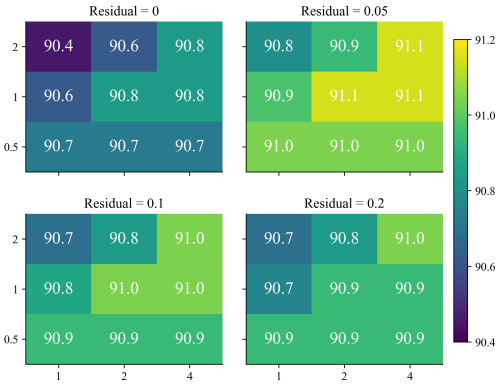
Ablation studies isolate the two core components of the method. Table 4 shows that perturbing at a low-FTLE position in the later trajectory segment yields only 88.3% performance, whereas a randomly selected fixed step and the first step achieve 88.4% and 89.6%, respectively. Notably, perturbation at the maximum FTLE point performs best, reaching 90.1%.Table 5 shows that $\sigma_L = 8, \sigma_H = 4$ yields 91.0% versus 85.2%, 89.00% and 90.1% for other scales. Appendix Table 6 further shows that the structured Maze rule energy is essential for selecting among the sampled basins: removing it reduces steering accuracy to below the baseline, while adding stablemax entropy provides an additional gain when used as a gated auxiliary signal. Figure 6 confirms that basin selection remains robust across a range of trajectory-energy weights, with accuracy staying above 90% under both local and global weight perturbations.

Table 4: Intervention-point ablation on TRM-Maze. All methods use 1,000 Maze examples, $K = 30$, $\sigma_L = 8$, $\sigma_H = 4$, and a 10-ACT rollout budget.

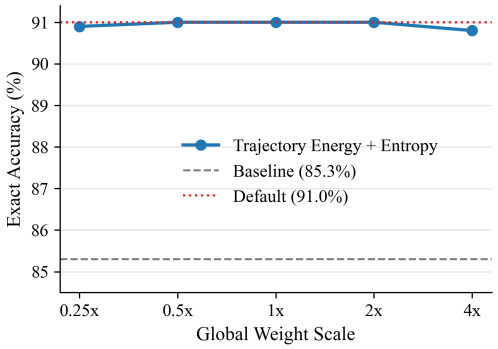
Method	Baseline	Steering
FTLE-min	85.30%	88.30%
Fixed step	85.30%	88.40%
First step	85.30%	89.60%
FTLE-max step	85.30%	90.10%

Table 5: Ablation of perturbation magnitudes on TRM-Maze using Trajectory Energy with Entropy. All results are evaluated on 1,000 Maze test examples with $K = 50$ rollouts.

σ_L	σ_H	Baseline	Steering
0.02	0.01	85.30%	85.20%
2	1	85.30%	90.10%
8	4	85.30%	**91.00%**
16	8	85.30%	89.00%



(a) Local weight perturbations



(b) Global weight scaling

Figure 6: Robustness of Gibbs re-selection under local and global Trajectory Energy variations. In (a), the horizontal axis varies the path-graph structure weight and the vertical axis varies the predicted path-cell count weight, with separate panels for different residual-stability weights. In (b), all structural energy weights are scaled together. Accuracy remains high across most configurations, indicating that the method is not narrowly tuned to a single hyperparameter setting.

6 Conclusion and Discussion

We presented a training-free intervention framework for recursive reasoning from a dynamical-systems perspective. We cast HRM/TRM inference as an iterative latent map, employ FTLE to localize candidate LCS-like boundaries between attractor basins, and apply targeted perturbations followed by Gibbs-weighted basin selection. We evaluate the method in Sudoku-Extreme, Maze-Hard, PPBench, and ARC-AGI-2. Across the evaluated tasks and settings, it improves accuracy without parameter updates, with gains for both HRM and TRM where both architectures are evaluated. These results support the view that some inference failures arise from capture by spurious basins and may be corrected through targeted test-time intervention.

These findings further suggest that the effectiveness of FTLE-guided steering comes from exploiting the dynamical structure of recursive inference. This use of dynamical structure reduces the required rollouts and augmentations on the current benchmarks, and may become increasingly important in higher-dimensional tasks and more complex recursive reasoning architectures, where unguided perturbation strategies are unlikely to scale. Future work should evaluate this scalability on harder benchmarks such as ARC-AGI-3, and extend the framework to small-parameter recursive reasoners and other recursive reasoning architectures.

AI Use Statement

In this work, we used generative AI tools for **language polishing and editing of the manuscript** and **assistance with drafting and revising manuscript sections**. We have not used generative AI tools for **research ideation, experimental design, data collection, or result analysis**, and **synthetic data generation and mathematical and theoretical development (including formulating claims,**

proof assistance, and hypothesis refinement) are not applicable to this work. Additionally, we used generative AI tools for **assistance with LaTeX formatting and reference organization**. We have reviewed all AI-assisted work. Specifically, all AI-assisted text was manually verified for factual accuracy against original sources, and any citations suggested by AI tools were independently confirmed through manual literature search. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Kanishk Awadhiya. Reasoning as attractor dynamics: Latent memory retrieval via gibbs-weighted energy minimization, 2026. URL https://arxiv.org/abs/2606.24543.

Junyeob Baek, Mingyu Jo, Minsu Kim, Mengye Ren, Yoshua Bengio, and Sungjin Ahn. Generative recursive reasoning, 2026. URL https://arxiv.org/abs/2605.19376.

Miguel Beneitez, Yohann Duguet, Philipp Schlatter, and Dan S. Henningson. Edge manifold as a lagrangian coherent structure in a high-dimensional state space. *Physical Review Research*, 2(3), Aug 2020. ISSN 2643-1564. doi: 10.1103/physrevresearch.2.033258. URL http://dx.doi.org/10.1103/physrevresearch.2.033258.

Reyk Börner, Ryan Deeley, Raphael Römer, Tobias Grafke, Valerio Lucarini, and Ulrike Feudel. Saddle avoidance of noise-induced transitions in multiscale systems. *Phys. Rev. Res.*, 6:L042053, Dec 2024. doi: 10.1103/PhysRevResearch.6.L042053. URL https://link.aps.org/doi/10.1103/PhysRevResearch.6.L042053.

Andrea Ceni, Peter Ashwin, and Lorenzo Livi. Interpreting recurrent neural networks behaviour via excitable network attractors. *Cognitive Computation*, 12(2):330–356, 2020. doi: 10.1007/s12559-019-09634-2. URL https://doi.org/10.1007/s12559-019-09634-2.

Jeffrey Cheng and Benjamin Van Durme. Compressed chain of thought: Efficient reasoning through dense representations, 2024. URL https://arxiv.org/abs/2412.13171.

Predrag Cvitanovic, Roberto Artuso, Ronnie Mainieri, Gregor Tanner, Gábor Vattay, Niall Whelan, and Andreas Wirzba. Chaos: classical and quantum. *ChaosBook. org (Niels Bohr Institute, Copenhagen 2005)*, 69:25, 2005.

Yuntian Deng, Yejin Choi, and Stuart Shieber. From explicit cot to implicit cot: Learning to internalize cot step by step, 2024. URL https://arxiv.org/abs/2405.14838.

Rainer Engelken, Fred Wolf, and L. F. Abbott. Lyapunov spectra of chaotic recurrent neural networks. *Phys. Rev. Res.*, 5:043044, Oct 2023. doi: 10.1103/PhysRevResearch.5.043044. URL https://link.aps.org/doi/10.1103/PhysRevResearch.5.043044.

Mohammad Farazmand and George Haller. Erratum and addendum to "a variational theory of hyperbolic lagrangian coherent structures" [physica d 240 (2011) 574–598]. *Physica D: Nonlinear Phenomena*, 241(4):439–441, 2012. ISSN 0167-2789. doi: https://doi.org/10.1016/j.physd.2011.09.013. URL https://www.sciencedirect.com/science/article/pii/S0167278911002600.

Jacob Fein-Ashley and Paria Rashidinejad. Solve the loop: Attractor models for language and reasoning, 2026. URL https://arxiv.org/abs/2605.12466.

Mark I. Freidlin and Alexander D. Wentzell. *Random Perturbations of Dynamical Systems*, volume 260 of *Grundlehren der mathematischen Wissenschaften*. Springer, Berlin, Heidelberg, 3 edition, 2012. ISBN 978-3-642-25846-6. doi: 10.1007/978-3-642-25847-3.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach, 2025. URL https://arxiv.org/abs/2502.05171.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025. doi: 10.1038/s41586-025-09422-z. URL https://doi.org/10.1038/s41586-025-09422-z.

Alireza Hadjighasem, Mohammad Farazmand, Daniel Blazevski, Gary Froyland, and George Haller. A critical comparison of lagrangian methods for coherent structure detection. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(5):053104, 05 2017. ISSN 1054-1500. doi: 10.1063/1.4982720. URL https://doi.org/10.1063/1.4982720.

George Haller. A variational theory of hyperbolic lagrangian coherent structures. *Physica D: Nonlinear Phenomena*, 240(7):574–598, 2011. ISSN 0167-2789. doi: https://doi.org/10.1016/j.physd.2010.11.010. URL https://www.sciencedirect.com/science/article/pii/S0167278910003143.

George Haller. Lagrangian coherent structures. *Annual Review of Fluid Mechanics*, 47 (Volume 47, 2015):137–162, 2015. ISSN 1545-4479. doi: https://doi.org/10.1146/annurev-fluid-010313-141322. URL https://www.annualreviews.org/content/journals/10.1146/annurev-fluid-010313-141322.

George Haller and Themistoklis Sapsis. Lagrangian coherent structures and the smallest finite-time lyapunov exponent. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 21(2), May 2011. ISSN 1089-7682. doi: 10.1063/1.3579597. URL http://dx.doi.org/10.1063/1.3579597.

George Haller and Guocheng Yuan. Lagrangian coherent structures and mixing in two-dimensional turbulence. *Physica D: Nonlinear Phenomena*, 147(3):352–370, 2000. ISSN 0167-2789. doi: https://doi.org/10.1016/S0167-2789(00)00142-1. URL https://www.sciencedirect.com/science/article/pii/S0167278900001421.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason E. Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. In *Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=Itxz7S4Ip3.

Benhao Huang, Zhengyang Geng, and J Zico Kolter. Equilibrium reasoners: Learning attractors enables scalable reasoning. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=lh95PnOlpM.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024. URL https://arxiv.org/abs/2412.16720.

Alexia Jolicoeur-Martineau. Less is more: Recursive reasoning with tiny networks, 2025. URL https://arxiv.org/abs/2510.04871.

Christian Kuehn and Tobias Wöhrer. Tracking finite-time lyapunov exponents to robustify neural odes. *Neurocomputing*, 699:134371, 2026. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2026.134371. URL https://www.sciencedirect.com/science/article/pii/S0925231226017698.

Shuhong Liu, Nozomi Akashi, Qingyao Huang, Yasuo Kuniyoshi, and Kohei Nakajima. Exploiting chaotic dynamics as deep neural networks. *Phys. Rev. Res.*, 7:033031, Jul 2025. doi: 10.1103/l8dq-mphg. URL https://link.aps.org/doi/10.1103/l8dq-mphg.

Yazhini Muruganantham, Andrei Velichko, and Samidurai Rajendran. Local lyapunov analysis via micro-ensembles: finite-time lyapunov exponent estimation and knn-based predictive comparison in complex-valued bam neural networks. *International Journal of Systems Science*, pp. 1–20, May 2026. ISSN 1464-5319. doi: 10.1080/00207721.2026.2672682. URL http://dx.doi.org/10.1080/00207721.2026.2672682.

Alex Ning and Vainateya Rangaraju. Visualizing llm latent space geometry through dimensionality reduction, 2025. URL https://arxiv.org/abs/2511.21594.

Edward Ott. *Chaos in Dynamical Systems*. Cambridge University Press, 1993.

Zirui Ren and Ziming Liu. Are your reasoning models reasoning or guessing? a mechanistic analysis of hierarchical reasoning models, 2026. URL https://arxiv.org/abs/2601.10679.

Davide Santolin, Nahuel Freitas, Massimiliano Esposito, and Gianmaria Falasco. Bridging freidlin-wentzell large deviations theory and stochastic thermodynamics. *Phys. Rev. E*, 111:024106, Feb 2025. doi: 10.1103/PhysRevE.111.024106. URL https://link.aps.org/doi/10.1103/PhysRevE.111.024106.

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. Reasoning with latent thoughts: On the power of looped transformers, 2025. URL https://arxiv.org/abs/2502.17416.

Mark Schöne, Babak Rahmani, Heiner Kremer, Fabian Falck, Hitesh Ballani, and Jannes Gladrow. Implicit language models are RNNs: Balancing parallelization and expressivity. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 53439–53466. PMLR, 2025. URL https://proceedings.mlr.press/v267/schone25a.html.

Amin Sghaier, Ali Parviz, and Alexia Jolicoeur-Martineau. Probabilistic tiny recursive model, 2026. URL https://arxiv.org/abs/2605.19943.

Shawn C. Shadden, Francois Lekien, and Jerrold E. Marsden. Definition and properties of lagrangian coherent structures from finite-time lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena*, 212(3):271–304, 2005. ISSN 0167-2789. doi: https://doi.org/10.1016/j.physd.2005.10.007. URL https://www.sciencedirect.com/science/article/pii/S0167278905004446.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. Codi: Compressing chain-of-thought into continuous space via self-distillation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 677–693. Association for Computational Linguistics, Nov 2025. ISBN 979-8-89176-332-6". doi: 10.18653/v1/2025.emnlp-main.36. URL https://aclanthology.org/2025.emnlp-main.36/.

L. Storm, H. Linander, J. Bec, K. Gustavsson, and B. Mehlig. Finite-time lyapunov exponents of deep neural networks. *Phys. Rev. Lett.*, 132:057301, Feb 2024. doi: 10.1103/PhysRevLett.132.057301. URL https://link.aps.org/doi/10.1103/PhysRevLett.132.057301.

Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, Tatsunori Hashimoto, and Carlos Guestrin. Learning to (Learn at test time): RNNs with expressive hidden states. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 57503–57522. PMLR, 2025. URL https://proceedings.mlr.press/v267/sun25h.html.

Simon Vock and Christian Meisel. Critical dynamics governs deep learning, 2026. URL https://arxiv.org/abs/2507.08527.

Ryan Vogt, Yang Zheng, and Eli Shlizerman. Lyapunov-guided representation of recurrent neural network performance. *Neural Computing and Applications*, 36(34):21211–21226, 2024. doi: 10.1007/s00521-024-09824-6. URL https://doi.org/10.1007/s00521-024-09824-6.

Guan Wang, Jin Li, Yuhao Sun, Xing Chen, Changling Liu, Yue Wu, Meng Lu, Sen Song, and Yasin Abbasi Yadkori. Hierarchical reasoning model, 2025a. URL https://arxiv.org/abs/2506.21734.

Guan Wang, Changling Liu, Chenyu Wang, Cai Zhou, Yuhao Sun, Yifei Wu, Shuai Zhen, Luca Scimeca, and Yasin Abbasi Yadkori. Hrm-text: Efficient pretraining beyond scaling, 2026. URL https://arxiv.org/abs/2605.20613.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Zhilin Wang, Yafu Li, Jianhao Yan, Yu Cheng, and Yue Zhang. Unveiling attractor cycles in large language models: A dynamical systems view of successive paraphrasing. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12740–12755. Association for Computational Linguistics, Jul 2025b. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.624. URL https://aclanthology.org/2025.acl-long.624/.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837. Curran Associates, Inc., 2022. doi: 10.52202/068431-1800. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Caleb Zheng and Eli Shlizerman. Hyperpruning: Efficient search through pruned variants of recurrent neural networks leveraging lyapunov spectrum, 2025. URL https://arxiv.org/abs/2506.07975.

Rui-Jie Zhu, Tianhao Peng, Tianhao Cheng, Xingwei Qu, Jinfa Huang, Dawei Zhu, Hao Wang, Kaiwen Xue, Xuanliang Zhang, Yong Shan, et al. A survey on latent reasoning, 2025. URL https://arxiv.org/abs/2507.06203.

A HRM/TRM Recursive Definitions and Dynamical Formalization

This appendix reviews the recursive structures of HRM and TRM as defined in their original papers, and explains how we unify them into a discrete-time latent dynamical system without modifying the original model definitions. Sections A.1 and A.2 follow the original update forms; Section A.3 introduces the unified notation $s_{t+1} = F_\theta(s_t; x)$ as an analytical abstraction for FTLE, LCS-like boundary, and basin steering analysis.

A.1 HRM Recursive Definition

HRM models reasoning as a coupled recursive process operating at two different temporal scales. The original paper defines four learnable components: an input network $f_I(\cdot;\theta_I)$, a low-level recurrent module $f_L(\cdot;\theta_L)$, a high-level recurrent module $f_H(\cdot;\theta_H)$, and an output network $f_O(\cdot;\theta_O)$. Given an input x, the working representation is first obtained as

$$\tilde{x} = f_I(x;\theta_I). \tag{9}$$

HRM maintains two latent states z_L^i and z_H^i. Over a total of $N \times T$ low-level time steps, the low-level module updates at every step, while the high-level module updates only at the end of each T-step cycle. The original recurrence is:

$$z_L^i = f_L(z_L^{i-1}, z_H^{i-1}, \tilde{x}; \theta_L), \tag{10}$$

$$z_H^i = \begin{cases} f_H(z_H^{i-1}, z_L^i; \theta_H), & \text{if } i \equiv 0 \pmod{T}, \\ z_H^{i-1}, & \text{otherwise.} \end{cases} \tag{11}$$

After N complete high-level cycles, the output is read from the high-level state:

$$\hat{y} = f_O(z_H^{NT}; \theta_O). \tag{12}$$

Thus, a single forward pass of HRM is not a single feedforward mapping, but a recursive computation alternating between fast low-level updates and slow high-level updates. The low-level module performs local computations under a fixed high-level context, while the high-level module periodically absorbs low-level results and changes the context for subsequent low-level recursion. Our FTLE diagnosis for HRM primarily targets the low-level z_L transitions, as this part provides the densest recursive trajectory.

A.2 TRM Recursive Definition

TRM simplifies HRM. The original paper shows that TRM no longer relies on two independent recurrent modules, complex hierarchical structures, or fixed-point theorems. Instead, it uses a single small network to recursively update the latent reasoning state z and progressively refine the current answer representation y.

Given the input embedding x, current answer representation y, and reasoning latent state z, one *latent recursion* of TRM consists of n low-level reasoning updates followed by one answer update:

$$z_{k+1} = f_\theta(z_k, y, x), \qquad k = 0, 1, \ldots, n-1, \tag{13}$$

$$y' = g_\theta(y, z_n), \tag{14}$$

where f_θ updates the reasoning latent state based on the input and current answer, and g_θ refines the answer representation based on the accumulated reasoning state. This process can be viewed as a composite mapping from (y, z_0) to (y', z_n):

$$(y', z_n) = \mathcal{R}_\theta(y, z_0; x), \tag{15}$$

where $\mathcal{R}_\theta = g_\theta \circ f_\theta^{(n)}$ denotes one complete latent recursion.

TRM's *deep recursion* repeats the above process T times to form one supervision step:

$$(y^{(1)}, z^{(1)}) = \mathcal{R}_\theta(y^{(0)}, z^{(0)}; x), \tag{16}$$

$$(y^{(2)}, z^{(2)}) = \mathcal{R}_\theta(y^{(1)}, z^{(1)}; x), \tag{17}$$

$$\vdots \tag{18}$$

$$(y^{(T)}, z^{(T)}) = \mathcal{R}_\theta(y^{(T-1)}, z^{(T-1)}; x), \tag{19}$$

where the first $T-1$ recursions are detached from the computation graph after the forward pass and do not participate in gradient backpropagation; the final recursion retains gradients for supervision signal generation. The final prediction is read from $y^{(T)}$ by the output head:

$$\hat{y} = f_O(y^{(T)}; \theta_O). \tag{20}$$

The original hyperparameters are set to $n = 6$ and $T = 3$. To align with HRM notation, y can be viewed as the current embedded answer (analogous to the high-level solution representation z_H in HRM), and z as the reasoning latent state (analogous to the low-level state z_L). We adopt this correspondence throughout our analysis.

A.3 Unified Dynamical Abstraction

Based on the original recursive definitions above, we unify the inference processes of HRM and TRM as a discrete-time latent dynamical system:

$$s_{t+1} = F_\theta(s_t; x), \qquad \hat{y}_t = D_\theta(s_t). \tag{21}$$

Here s_t denotes the complete latent state at recursive position t. For HRM, s_t contains (z_L^i, z_H^i); for TRM, s_t contains the current answer representation y and the reasoning latent state z. F_θ denotes the local transition map induced by the original model's recursive updates, and D_θ denotes the output readout function.

This unified formulation is our analytical abstraction rather than a verbatim restatement of the original equations. It places different recursive architectures into a common dynamical-systems framework, enabling the definition of layer-wise FTLE, local perturbations, and basin-level selection used in the main text.

B Variational Diagnostics for Repelling LCS Candidates

This appendix describes how we connect the high-FTLE intervention points used in our method to the variational theory of hyperbolic Lagrangian Coherent Structures (LCSs). Our goal is not to claim that a scalar FTLE maximum along a single reasoning trajectory is, by itself, a mathematically complete LCS. Instead, we use the maximal-FTLE layer as a candidate time slice and then perform a local and consistency check in the latent state space. This check asks whether the candidate point satisfies the finite-time geometric conditions expected of a repelling LCS.

B.1 Definition of finite-time Lyapunov exponents and candidate Lagrangian coherent structures

For a fixed puzzle input x, the recursive model defines an input-conditioned discrete-time dynamical system

$$s_{t+1} = F_\theta(s_t; x), \qquad \hat{y}_t = D_\theta(s_t), \tag{22}$$

where s_t denotes the latent recurrent state and D_θ is the readout head. For a finite horizon T, we define the local flow map

$$\Phi_t^{t+T}(s_t) = F_\theta^T(s_t; x) = s_{t+T}. \tag{23}$$

The linearized evolution of an infinitesimal perturbation δs_t is

$$\delta s_{t+T} = D\Phi_t^{t+T}(s_t)\, \delta s_t. \tag{24}$$

Following the standard LCS construction, we define the finite-time Cauchy–Green strain tensor

$$\mathbf{C}_t^{t+T}(s_t) = \left[D\Phi_t^{t+T}(s_t)\right]^\top D\Phi_t^{t+T}(s_t). \tag{25}$$

Let its eigenpairs be

$$\mathbf{C}_t^{t+T}(s_t)\xi_i = \lambda_i \xi_i, \qquad 0 < \lambda_1 \le \cdots \le \lambda_{n-1} \le \lambda_n. \tag{26}$$

The largest finite-time Lyapunov exponent is then

$$\sigma_T(s_t) = \frac{1}{2T} \log \lambda_n(s_t). \tag{27}$$

In the main algorithm, we first identify the most unstable layer along the unperturbed reasoning trajectory:

$$t^\star = \arg\max_t \sigma_T(s_t). \tag{28}$$

The role of $t^\star$ is to identify where to search for a local repelling LCS candidate. The layer $t^\star$ is not itself an LCS; rather, the candidate LCS is a local codimension-one hypersurface in the latent state space at that layer.

B.2 Variational criterion from Haller's theorem

We recall the relevant part of Haller's variational theory of hyperbolic LCSs. Consider a compact material hypersurface $\mathcal{M}(t) \subset U$ over a finite interval $[t_0, t_0 + T]$. Let $\mathbf{C}_{t_0}^{t_0+T}$ be the Cauchy–Green tensor associated with the flow map, with eigenvalues

$$0 < \lambda_1 \le \cdots \le \lambda_{n-1} \le \lambda_n \tag{29}$$

and corresponding orthonormal eigenvectors $\xi_1, \ldots, \xi_n$.

Theorem 7 of Haller (2011), specialized to repelling LCSs. A compact material hypersurface $\mathcal{M}(t)$ is a repelling weak LCS over $[t_0, t_0 + T]$ if and only if, for all $x_0 \in \mathcal{M}(t_0)$,

$$\lambda_{n-1}(x_0, t_0, T) \neq \lambda_n(x_0, t_0, T) > 1, \tag{30}$$

$$\xi_n(x_0, t_0, T) \perp T_{x_0}\mathcal{M}(t_0), \tag{31}$$

$$\langle \nabla\lambda_n(x_0, t_0, T), \xi_n(x_0, t_0, T)\rangle = 0. \tag{32}$$

Moreover, $\mathcal{M}(t)$ is a repelling LCS if and only if it is a repelling weak LCS and the matrix $\mathbf{L}(x_0, t_0, T)$ defined in (Haller, 2011; Farazmand & Haller, 2012) is positive definite for all $x_0 \in \mathcal{M}(t_0)$.

In coordinates aligned with the Cauchy–Green eigenbasis, Haller's matrix takes the form

$$\mathbf{L} = \begin{pmatrix} A & 2\frac{\lambda_n-\lambda_1}{\lambda_1\lambda_n}\langle \xi_1, \nabla_{\xi_n}\xi_n\rangle & \cdots & 2\frac{\lambda_n-\lambda_{n-1}}{\lambda_{n-1}\lambda_n}\langle \xi_{n-1}, \nabla_{\xi_n}\xi_n\rangle \\ 2\frac{\lambda_n-\lambda_1}{\lambda_1\lambda_n}\langle \xi_1, \nabla_{\xi_n}\xi_n\rangle & 2\frac{\lambda_n-\lambda_1}{\lambda_1\lambda_n} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 2\frac{\lambda_n-\lambda_{n-1}}{\lambda_{n-1}\lambda_n}\langle \xi_{n-1}, \nabla_{\xi_n}\xi_n\rangle & 0 & \cdots & 2\frac{\lambda_n-\lambda_{n-1}}{\lambda_{n-1}\lambda_n} \end{pmatrix}, \tag{33}$$

where

$$A = \nabla^2\mathbf{C}^{-1}[\xi_n, \xi_n, \xi_n, \xi_n]. \tag{34}$$

Equivalently, Haller gives the coordinate-free expression

$$A = -\frac{1}{\lambda_n^2}\left\langle \xi_n, \nabla^2\lambda_n\,\xi_n\right\rangle + 2\sum_{q=1}^{n-1}\frac{\lambda_n - \lambda_q}{\lambda_q\lambda_n}\left\langle \xi_q, \nabla_{\xi_n}\xi_n\right\rangle^2. \tag{35}$$

Positive definiteness of $\mathbf{L}$ corresponds to a nondegenerate local maximum of the normal repulsion rate among nearby material.

For a repelling LCS, the weak-LCS conditions must be supplemented by positive definiteness of the second-variation matrix $\mathbf{L}$, with the corrected expression given by Haller (Haller, 2011; Farazmand & Haller, 2012). In a genuinely two-dimensional flow, the corrected condition reduces to

$$\left\langle \xi_2, \nabla^2\lambda_2\,\xi_2\right\rangle < 0. \tag{36}$$

Our PCA slice is embedded in a high-dimensional discrete latent map and is not assumed to define an invariant two-dimensional material flow. We therefore use the variational theory as geometric motivation rather than as a theorem-level pass/fail test.

B.3 Projected Diagnostics and Interpretation

Haller's theory clarifies why FTLE ridges should be interpreted with care. A ridge of the largest FTLE field is not automatically a Lagrangian coherent structure: it may reflect strong finite-time stretching. In the exact variational theory, a compact hypersurface must satisfy eigenvalue separation, normal alignment of the dominant Cauchy–Green eigenvector, and stationarity of the largest eigenvalue along that direction, together with the corrected second-variation condition. Our projected diagnostics do not constitute a direct verification of all these conditions.

For each representative failure-to-success correction, we examine the local projected geometry at the FTLE-selected latent time $t^\star$. The objective is not to prove the existence of a global LCS, but to determine whether the selected high-FTLE neighborhood exhibits geometric organization consistent with an LCS-inspired repelling boundary. The diagnostic procedure is:

1. Compute the finite-horizon transition map $\Phi_{t^\star}^{t^\star+T}$ used by the FTLE estimator.
2. Estimate the largest Cauchy–Green eigenvalue and its dominant eigenvector using Jacobian-vector and vector-Jacobian products.
3. Construct a local two-dimensional PCA section around the selected latent state and recompute the FTLE and dominant Cauchy–Green stretching value on this section.
4. Compare the projected dominant stretching direction with the local FTLE level-set tangent through $|\langle \xi_1, \tau\rangle|$.

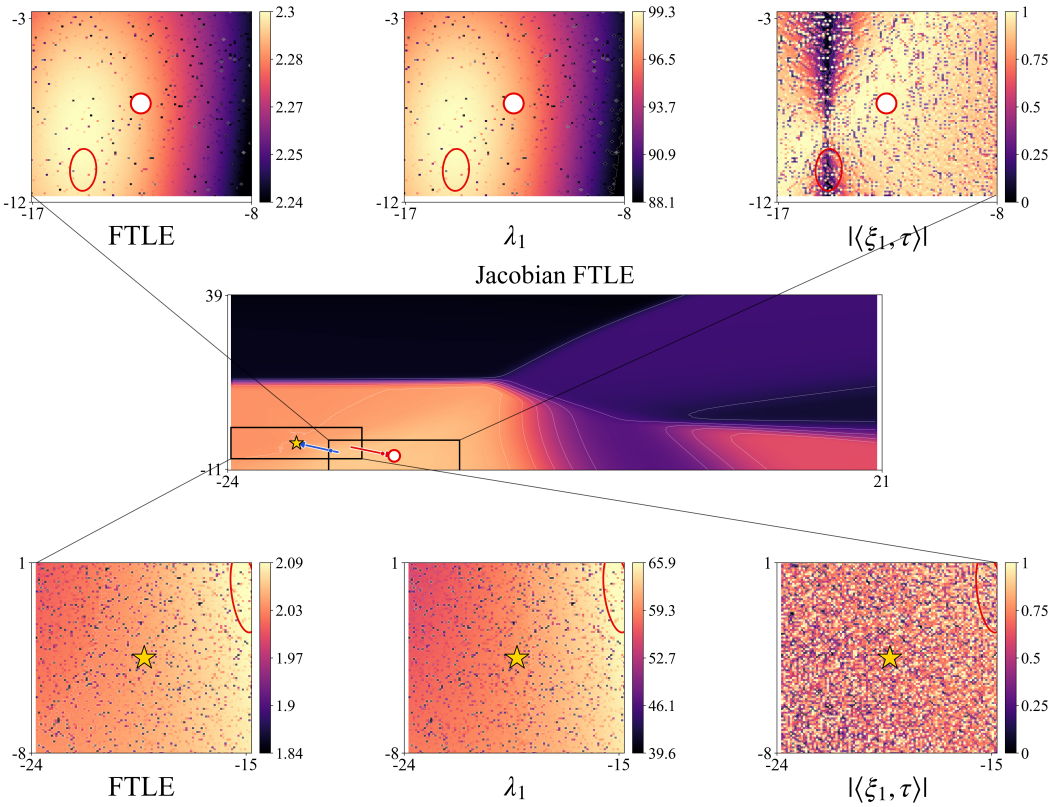
Here, τ denotes the local tangent to an FTLE level set in the two-dimensional visualization plane. It should not be identified with the tangent of an exact material LCS. Small values of $|\langle \xi_1, \tau \rangle|$ indicate that the projected dominant stretching direction is locally normal to the FTLE contours.

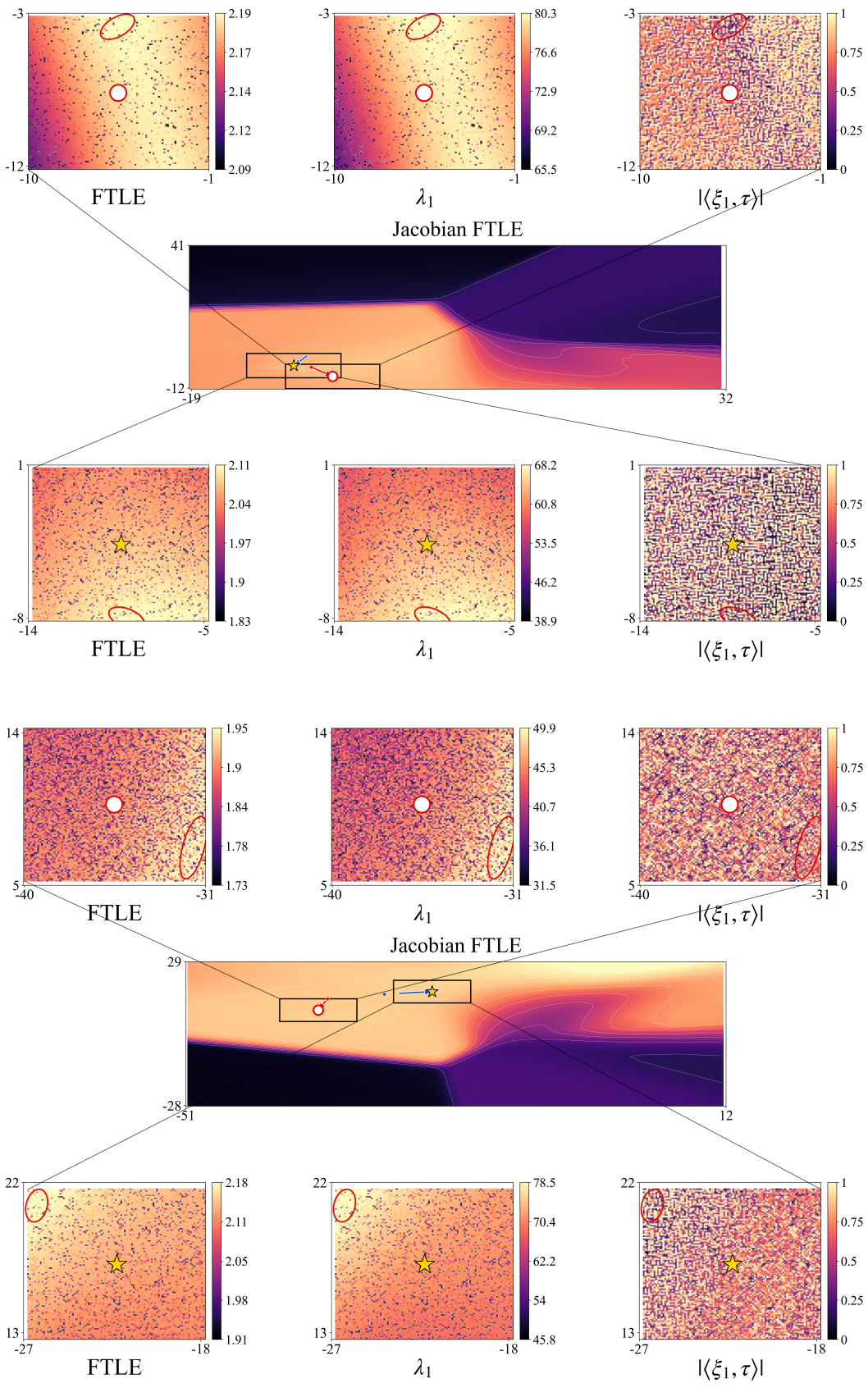
It must be mentioned that these projected first-order diagnostics do not by themselves establish a repelling LCS, whose full verification would additionally require the corrected second-variation matrix to be positive definite. Because this condition is difficult to evaluate in the full latent space, we interpret the reported quantities more narrowly as evidence of local, weak-LCS-like repelling geometry consistent with a boundary between trajectories leading to different computational outcomes.

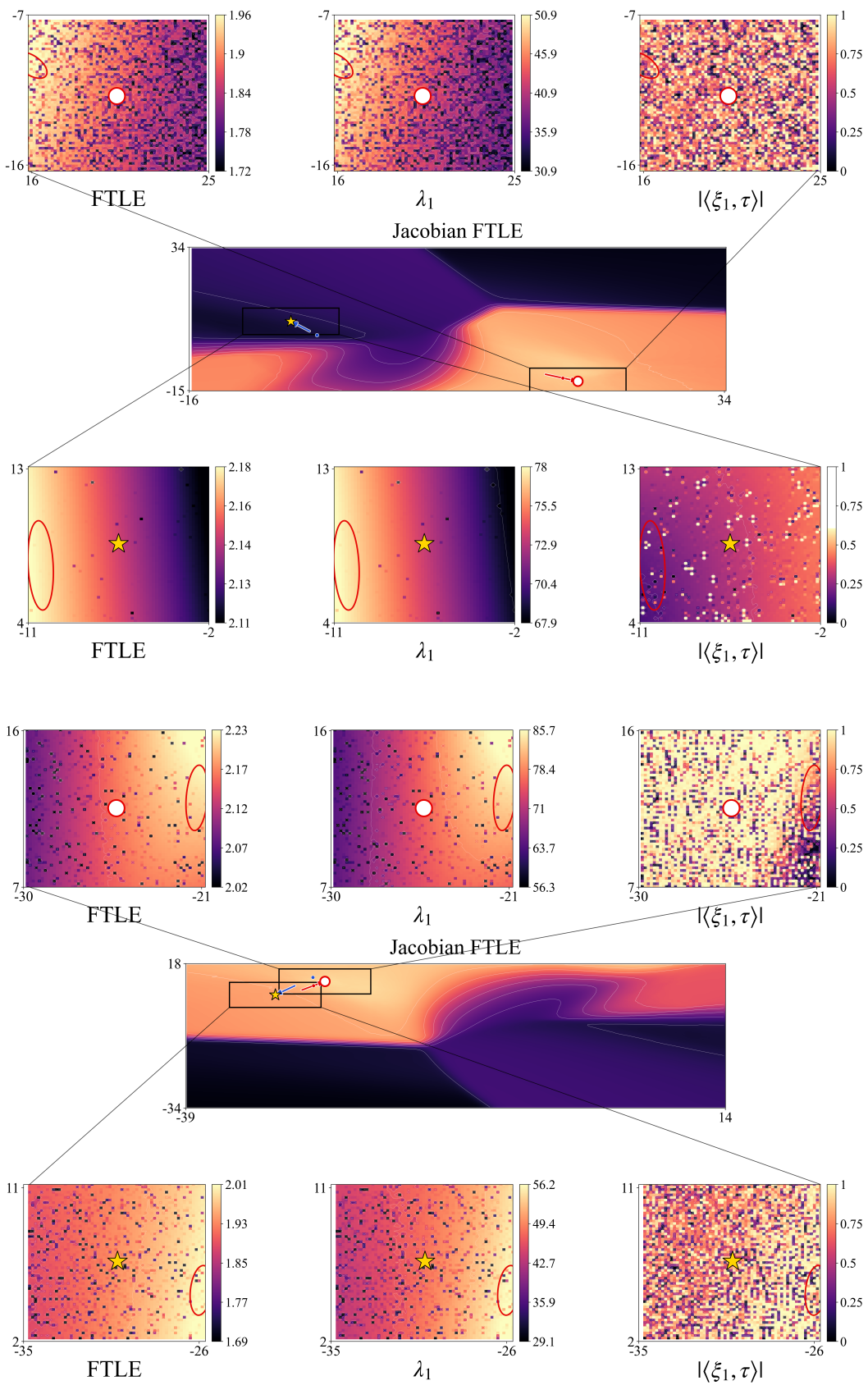
Building on these diagnostics, we further examine the local projected geometry around FTLE-selected intervention points in representative corrected examples. In each figure, the middle panel shows the Jacobian-FTLE landscape in a two-dimensional PCA projection. The red circle, white dot, and yellow star indicate the relevant baseline and steered latent states at the phase corresponding to the baseline maximum-FTLE point. A coherent high-FTLE structure between these projected states is consistent with the interpretation of the selected region as an LCS-inspired boundary in the visualization plane.

The local panels report the FTLE field, the dominant Cauchy–Green stretching value λ_1, and the projected alignment measure $|\langle \xi_1, \tau \rangle|$. In the implementation and figure labels, the eigenvalues are stored in descending order, so λ_1 and ξ_1 denote the largest eigenvalue and its associated eigenvector; these correspond to λ_n and ξ_n under the ascending theoretical convention used above.

Across examples, the selected neighborhoods exhibit elevated finite-time stretching and coherent projected directional structure. These observations provide local geometric information beyond the scalar FTLE value measured along the original reasoning trajectory. They remain projected diagnostics rather than a proof of a global LCS in the full latent space.




FTLE
λ_1
$|\langle \xi_1, \tau \rangle|$
Jacobian FTLE
FTLE
λ_1
$|\langle \xi_1, \tau \rangle|$
Jacobian FTLE
FTLE
λ_1
$|\langle \xi_1, \tau \rangle|$


FTLE
λ_1
$|\langle \xi_1, \tau \rangle|$
Jacobian FTLE
FTLE
λ_1
$|\langle \xi_1, \tau \rangle|$
Jacobian FTLE
FTLE
λ_1
$|\langle \xi_1, \tau \rangle|$

C Algorithmic Pseudocode

C.1 Jacobian FTLE Power Iteration

We treat the recursive inference process of HRM/TRM as a discrete-time nonlinear dynamical system. For a given input x, the model maintains low-level and high-level latent variables (z_L, z_H) at each recursive phase. In TRM, the low-level update can be written as

$$z_L^{t+1} = F_t(z_L^t; x, z_H^t), \tag{37}$$

where F_t denotes the model's low-level recurrent transition, conditioned on the current high-level state and input embedding. In HRM, the analogous low-level transition has the same role, while the high-level transition periodically updates the slow state from the completed low-level computation. Our steering intervention primarily targets local instability in the low-level latent state, so by default we compute the FTLE profile over valid low-level latent transitions and use its peak to locate candidate intervention points.

For a discrete phase point $t = (a, h, l)$, let the local map be F_t, or for a window of length T, let the composed map be

$$\Phi_{t,T} = F_{t+T-1} \circ \cdots \circ F_t. \tag{38}$$

The Jacobian FTLE at this point is defined as

$$\lambda_T(z_t) = \frac{1}{T} \log \sigma_{\max} \left(D\Phi_{t,T}(z_t) \right), \tag{39}$$

where $D\Phi_{t,T}(z_t)$ is the Jacobian of the composed latent map with respect to the current latent state, and $\sigma_{\max}$ denotes the maximum singular value. Equivalently,

$$\lambda_T(z_t) = \frac{1}{T} \log \max_{\|v\|_2 = \epsilon} \frac{\|D\Phi_{t,T}(z_t)v\|_2}{\epsilon}. \tag{40}$$

Here ϵ is used only for numerical normalization; since the computation uses Jacobian-vector products, the estimate corresponds to the perturbation growth rate under local linearization, rather than retraining the model or modifying parameters.

To avoid explicitly constructing the high-dimensional Jacobian, we use power iteration to estimate the maximum singular value. Given a random initial direction v_0, we first normalize it to $\|v_0\|_2 = \epsilon$. At each iteration, we compute via forward-mode automatic differentiation

$$\delta_i = D\Phi_{t,T}(z_t)v_i, \tag{41}$$

and obtain the local growth rate

$$\ell_i = \frac{1}{T} \log \frac{\|\delta_i\|_2}{\epsilon}. \tag{42}$$

We then set $u_i = \delta_i / \|\delta_i\|_2$, and compute via reverse-mode automatic differentiation

$$\tilde{v}_{i+1} = D\Phi_{t,T}(z_t)^\top u_i, \tag{43}$$

and renormalize to

$$v_{i+1} = \epsilon \frac{\tilde{v}_{i+1}}{\|\tilde{v}_{i+1}\|_2}. \tag{44}$$

This process is equivalent to power iteration on $D\Phi^\top D\Phi$, but only requires jvp and vjp operations, avoiding storage of the full Jacobian. If any iteration produces zero norm or non-finite values, we resample the random direction to ensure numerical stability. The final FTLE is taken as the mean of the last 20% of the power iteration values ℓ_i, reducing the transient bias from the initial random direction. The FTLE estimation pseudocode is as follows:

Algorithm 1 Jacobian FTLE by JVP/VJP power iteration

Require: Latent state z, local/composed map Φ, perturbation scale ϵ, horizon T, number of power iterations M
Ensure: FTLE estimate λ

Sample $v \sim \mathcal{N}(0, I)$
$v \leftarrow \epsilon \cdot v / \|v\|_2$
for $i = 1$ to M **do**
  $\delta \leftarrow \text{JVP}(\Phi, z, v)$ $\{\delta = D\Phi(z)v\}$
  $d \leftarrow \|\delta\|_2$
  $\ell_i \leftarrow (1/T) \cdot \log(\max(d, \text{tiny})/\epsilon)$
  if d is finite and $d > 0$ **then**
    $u \leftarrow \delta / d$
    $g \leftarrow \text{VJP}(\Phi, z, u)$ $\{g = D\Phi(z)^\top u\}$
    if $\|g\|_2$ is finite and positive **then**
      $v \leftarrow \epsilon \cdot g / \|g\|_2$
    else
      Resample v with $\|v\|_2 = \epsilon$
    end if
  else
    Resample v with $\|v\|_2 = \epsilon$
  end if
end for
$\lambda \leftarrow$ mean of the last 20% of $\{\ell_i\}$
return λ

C.2 FTLE-GUIDED PERTURBATION AND STEERING

In steering inference, each latent phase point is identified by a triple (a, h, l), with flattened index

$$\text{flat}(a, h, l) = a \cdot H_{\text{cycles}} L_{\text{cycles}} + h \cdot L_{\text{cycles}} + l. \tag{45}$$

We select the intervention target as the point with maximum FTLE within an early window of valid low-level phase points:

$$t^\star = \arg\max_{t \le W} \lambda_T(z_t). \tag{46}$$

Ties are resolved by choosing the earlier flat index. After obtaining $t^\star = (a^\star, h^\star, l^\star)$, we replay the baseline dynamics to that phase, inject Gaussian perturbation, and select the final answer by Gibbs basin aggregation.

Algorithm 2 FTLE-guided perturbation inference

Require: Model f_θ, input x, maximum ACT steps T_0, FTLE window W, number of rollouts K, noise scales σ_L, σ_H, inverse temperature β
Ensure: Selected prediction $\hat{y}$

```
Run an unperturbed rollout of f_θ(x) for T_0 ACT steps.
for each low-level latent state z_L^{a,h,l} in the first W latent steps do
    Compute λ^{a,h,l} ← FTLE(F_L, z_L^{a,h,l}) using Algorithm 1.
end for
(a*, h*, l*) ← arg max_{(a,h,l)≤W} λ^{a,h,l}
Re-execute the model to recover target states (z_L*, z_H*) at (a*, h*, l*).
for k = 1 to K do
    Sample normalized Gaussian directions ξ^r_{L,k} and ξ_{H,k}, where r = 0, ..., L_end − l* indexes
    the remaining low-level steps in the current recurrent cycle.
    Set Δz^r_{L,k} ← σ_L ‖z_L*‖_2 ξ^r_{L,k} and Δz_{H,k} ← σ_H ‖z_H*‖_2 ξ_{H,k}.
    Roll out f_θ(x) from the initial carry.
    for each ACT/high cycle (a, h) do
        for each low-level step l do
            if a = a*, h = h*, and l ≥ l* then
                Inject low-level perturbation z_L ← z_L + Δz^{l−l*}_{L,k}.
            end if
            Update z_L ← F_L(z_L; z_H, x).
        end for
        if a = a* and h = h* then
            Inject high-level perturbation z_H ← z_H + Δz_{H,k}.
        end if
        Update z_H ← F_H(z_H, z_L).
    end for
    Decode rollout prediction ŷ_k ← DECODE(z_H).
    Compute label-free diagnostics for confidence, residual, instability, and task-rule consistency.
end for
Compute robust-normalized rollout energies E_k.
Assign Gibbs weights π_k ∝ exp(−βE_k).
Group rollouts by identical prediction ŷ_k into answer basins.
Select B* ← arg max_B Σ_{k:ŷ_k∈B} π_k.
return representative prediction ŷ from B*.
```

In the main TRM Maze experiments, we use 50 FTLE power iterations, and selects the intervention target using one-step Jacobian FTLE; the early correction window contains 36 valid low-level latent transitions for TRM and 12 for HRM. The Jacobian FTLE visualization used for PCA landscapes follows the same JVP/VJP power iteration principle, but is implemented in a batched version: at the specified phase slice (h, l), we replay saved randomly initialized trajectories and compute Jacobian FTLE for up to 2048 latent states, using a small batch size to control memory usage. This implementation ensures that the FTLE landscapes in the figures are mathematically consistent with the instability measure used in steering inference.

D Rollout Energy and Gibbs Basin Selection Details

For each input instance, steering inference produces K candidate rollouts. We assign each rollout i a label-free energy E_i, used only for test-time re-selection and never trained against ground-truth labels. Let $x_m(i)$ denote the raw value of diagnostic component m for rollout i, including model-internal stability signals and task-rule signals. Each component is robustly normalized within the Krollout set:

$$z_m(i) = \frac{x_m(i) - \mathrm{median}(x_m)}{1.4826 \cdot \mathrm{MAD}(x_m) + \epsilon}, \tag{47}$$

and the combined energy is

$$E_i = \sum_m w_m \, z_m(i), \tag{48}$$

where w_m is the weight for component m. The stability components include logit negative log-likelihood, terminal latent residual, and late-step prediction instability. The rule components use task structure: Sudoku penalizes row, column, and box conflicts; Maze penalizes connectivity and path-structure violations.

After computing E_i, we convert energies into Gibbs-style rollout weights:

$$\pi_i = \frac{\exp(-\beta E_i)}{\sum_{j=1}^{K} \exp(-\beta E_j)}. \tag{49}$$

Rollouts are then grouped by identical final predictions into answer basins B. The basin mass is

$$M(B) = \sum_{i:\hat{y}_i \in B} \pi_i, \tag{50}$$

and the selected basin is

$$B^\star = \arg\max_B M(B). \tag{51}$$

Within $B^\star$, we use the lowest-energy rollout as the representative output; ties are broken by lower minimum energy and then by larger basin size. This basin-level selection rewards answers that are both low-energy and repeatedly reached by the perturbed dynamics.

D.1 Maze-Energy Components

For Maze, the task-rule component decomposes into the following terms.

Connection failure. This binary term checks whether the predicted path connects the maze start and goal:

$$x_{\text{conn}}(i) = \begin{cases} 0, & \text{if the predicted path reaches the goal,} \\ 1, & \text{otherwise.} \end{cases} \tag{52}$$

This term penalizes rollouts that fail the most basic maze connectivity constraint.

Graph badness. We convert the predicted path tokens into an induced grid graph containing the start, goal, and predicted path cells. The graph-badness term penalizes disconnected components, isolated cells, dangling path segments, and excessive branching. In implementation, disconnected components receive a penalty proportional to the number of extra components; start and goal nodes are encouraged to have degree one; interior path nodes are penalized for degree zero, degree one, or degree greater than two. This term favors predictions that form a single simple path-like structure.

Blue-path count. This term measures the number of predicted path cells. It discourages degenerate solutions that mark too many cells as part of the path, even if they satisfy weak connectivity. It therefore acts as a compactness prior over candidate maze paths.

Residual magnitude. The residual term uses the mean residual over the final recurrent steps of the rollout. It favors rollouts whose latent dynamics are closer to a settled recurrent state after perturbation. This term is model-internal and does not use labels.

Logit negative log-likelihood. This term measures the model's token-level predictive confidence for the selected prediction. In the final maze setting, its weight is set to zero, but we include it as an optional component for completeness.

Prediction instability. This term measures the instability of the predicted answer across late recurrent steps. Lower instability indicates that the candidate answer is more dynamically stable. As with the logit term, its weight is zero in the final maze configuration but it remains available as an optional energy component.

Stablemax token entropy. For Maze rollouts, we compute an auxiliary uncertainty signal from the token logits at the final ACT step used by the selector. Let $z_{i,u,c}$ denote the logit of rollout i, token position u, and class $c \in \{1, \dots, C\}$, with $C = 6$ for Maze. We first apply the stablemax transform

$$s(z) = \begin{cases} z + 1, & z \geq 0, \\ (1 - z)^{-1}, & z < 0, \end{cases} \tag{53}$$

and normalize it into a probability vector

$$p^{\text{stable}}_{i,u,c} = \frac{s(z_{i,u,c})}{\sum_{c'=1}^{C} s(z_{i,u,c'})}. \tag{54}$$

The stablemax entropy of token u is then

$$H^{\text{stable}}_{i,u} = -\sum_{c=1}^{C} p^{\text{stable}}_{i,u,c} \log p^{\text{stable}}_{i,u,c}. \tag{55}$$

We aggregate this over maze tokens to obtain the rollout-level entropy

$$e^{\text{ent}}_i = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} H^{\text{stable}}_{i,u}, \tag{56}$$

where $\mathcal{U}$ denotes the set of valid maze grid positions.

The entropy term is robustly normalized within the K-rollout set,

$$\tilde{e}^{\text{ent}}_i = \frac{e^{\text{ent}}_i - \text{median}_j(e^{\text{ent}}_j)}{\text{MAD}_j(e^{\text{ent}}_j) + \epsilon}, \tag{57}$$

and used as a weak adjustment to the Maze energy,

$$E^{\text{maze+ent}}_i = E^{\text{maze}}_i + w_{\text{ent}} \tilde{e}^{\text{ent}}_i. \tag{58}$$

In our entropy-assisted setting, $w_{\text{ent}} < 0$, so entropy acts as a mild preference for rollouts that remain locally uncertain among otherwise structurally plausible basins. We do not allow this term to override the maze constraints directly: an entropy-preferred basin is accepted only if it passes basin-mass and halt-confidence gates; otherwise the selector falls back to the original Trajectory Energy basin.

In the main TRM-Maze setting, the active energy weights are

$$(w_{\text{conn}}, w_{\text{graph}}, w_{\text{blue}}, w_{\text{res}}) = (1.0, 2.0, 1.0, 0.1), \tag{59}$$

with logit and instability weights set to zero. The base Trajectory Energy selector sets $w_{\text{ent}} = 0$; Trajectory Energy with Entropy activates this term only through the gated fallback rule above.

E Experimental Settings and Hyperparameters

All experiments use the released pretrained HRM/TRM checkpoints without any parameter updates. The deterministic baseline is the original model rollout under the same ACT budget as the steered run. For each test input, we first run the deterministic trajectory, compute the FTLE profile along the recurrent latent states, and select the maximal low-level FTLE position as the intervention point. Perturbed rollouts are then generated by injecting Gaussian noise from this selected point while keeping the input and model parameters fixed.

For the main Maze experiments, we evaluate all 1,000 Maze-Hard test examples with $K = 50$ rollouts, ACT budget 16, Gibbs inverse temperature $\beta = 2.0$, and perturbation scales $(\sigma_L, \sigma_H) = (8, 4)$. The trajectory energy weights are fixed to connection 1.0, graph 2.0, blue-path count 1.0, and terminal residual 0.1, with logit and instability weights set to zero. The entropy-assisted selector computes stablemax entropy over the six Maze token classes at the final ACT step, averages over valid grid tokens, robustly normalizes it within the rollout set, and applies a weak candidate weight -0.02.

ForSudoku-Extreme, we evaluate 10,000 test examples over three random seeds with $K = 50$ rollouts, ACT budget 16, Gibbs inverse temperature $\beta = 2.0$, and perturbation scales $(\sigma_L, \sigma_H) = (0.02, 0.02)$. The trajectory energy weights are fixed to robustly normalized logit negative log-likelihood 1.0, terminal residual 0.1.

For PPBench, we evaluate the 25,000 TRM checkpoint on the PTRM-compatible golden split. The available split contains 61 instances from six puzzle types; following the five-category aggregate reported by PTRM, Table 9 reports results on the 49 Sudoku, Lightup, Nurikabe, Heyawake, and Tapa instances, excluding Shakashaka. All results use a single seed. The deterministic TRM baseline uses one rollout without latent noise, while PTRM uses $K = 100$ noisy rollouts with best-Q selection. Our FTLE steering uses $K = 100$, 10 FTLE power iterations, and selects the maximum-FTLE intervention point within the first 36 low-level transitions. Perturbations are applied to z_L through the remaining low-level cycle and once to z_H, with normalized Gaussian scales $(\sigma_L, \sigma_H) = (4, 2)$. Final answers are selected by Gibbs basin aggregation with $\beta = 4$, using robustly normalized logit negative log-likelihood, terminal residual, and prediction instability with weights $(1.5, 1.0, 0.25)$. Ground-truth labels are used only for final exact-match evaluation.

For ARC-AGI-2, we evaluate the released TRM checkpoint on all 120 evaluation tasks with $A = 100$ fixed input augmentations per task. FTLE steering selects the intervention coordinate from the unaugmented baseline trajectory within the first $W = 20$ low-level transitions, $K = 25$ stochastic rollouts per augmentation, and perturbation scales $(\sigma_L, \sigma_H) = (1.0, 0.5)$. Candidates are selected by Gibbs basin mass with $\beta = 2$, using a label-free energy based on softmax entropy, terminal residual, and late prediction instability; selected basins are aggregated across augmentations by vote count with basin mass as a tie-breaker. The matched PTRM reproduction uses the same checkpoint, tasks, augmentations, K, and ACT budgets, with high-level Gaussian noise $\sigma = 0.2$ and Q-based selection as in PTRM. Ground-truth outputs are used only for final metric computation.

F Additional Experimental Results

F.1 Additional TRM-Maze Ablations

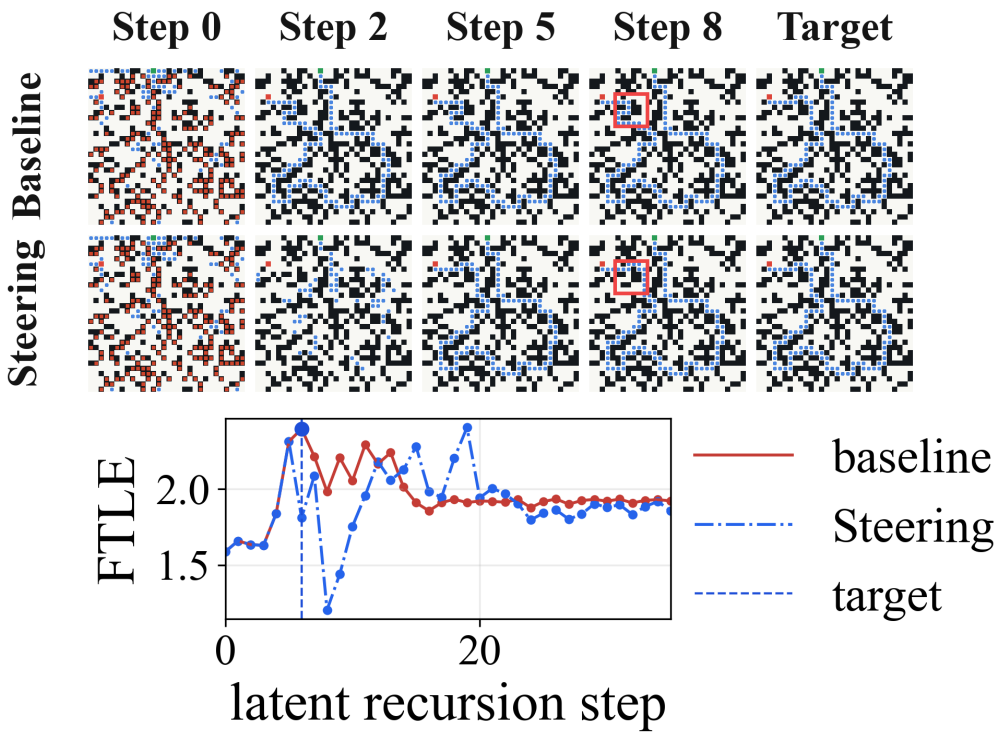
Table 6: Module ablation of TRM-Maze basin selection using the completed $K = 50$ FTLE-guided rollouts on 1,000 test examples. "Trajectory Energy with Entropy" denotes the entropy-assisted variant accepted only under basin-mass and halt-confidence gates.

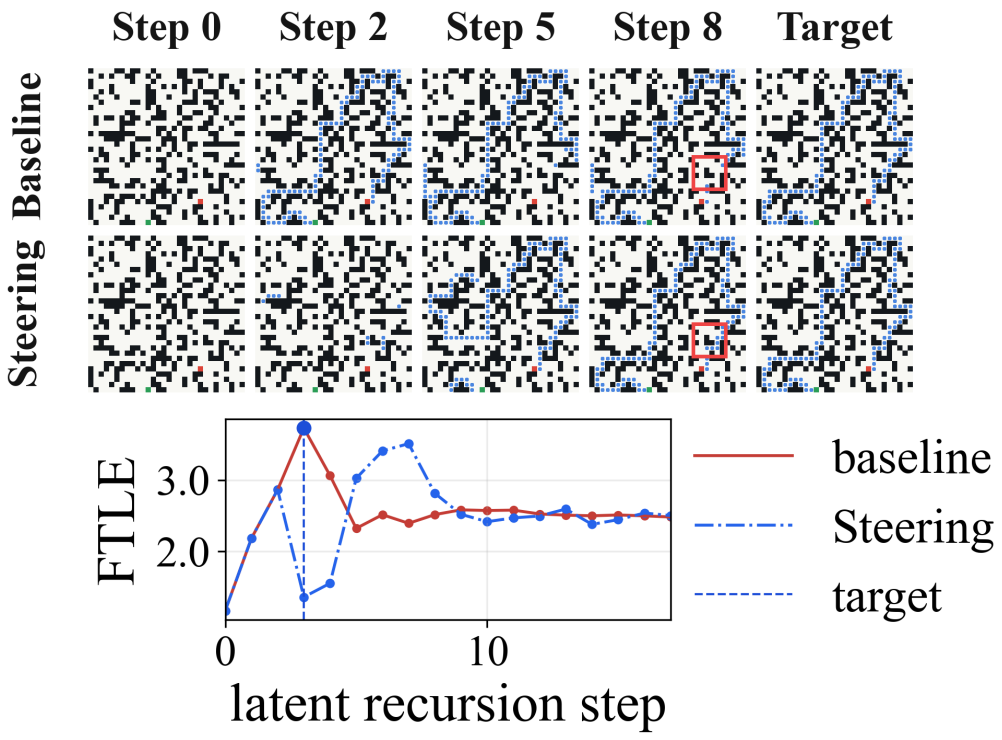
Selection variant	Active components	Exact	Acc.
Baseline TRM	none	853/1000	85.3%
No rule energy	residual only	853/1000	85.3%
No stability energy	rule only	905/1000	90.5%
No graph term	connection + blue + residual	895/1000	89.5%
No blue-count term	connection + graph + residual	859/1000	85.9%
Trajectory Energy	rule + residual	905/1000	90.5%
Trajectory Energy + entropy	rule + residual + entropy	910/1000	91.0%

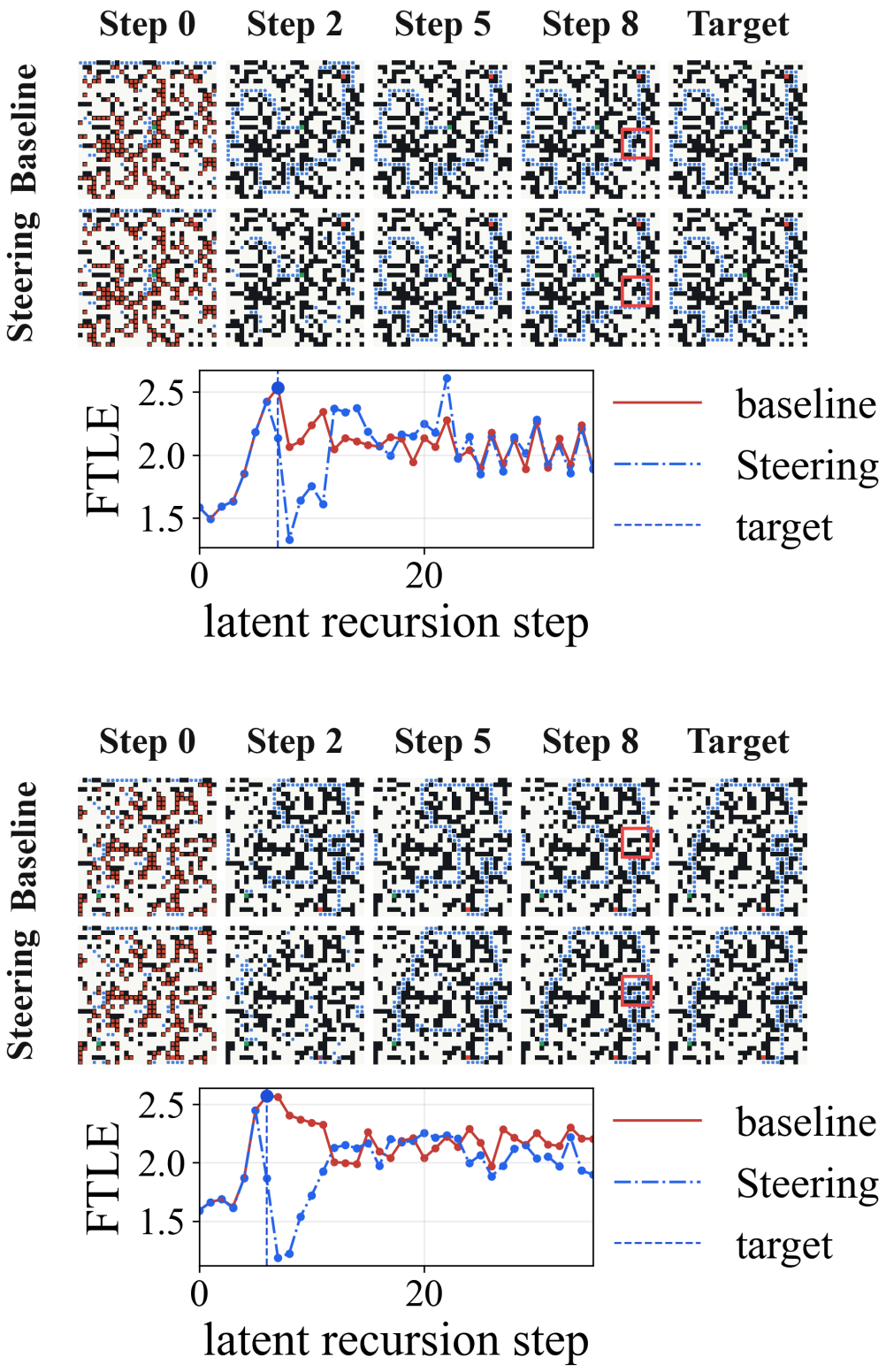
F.2 FTLE-Guided Corrections on Maze Samples

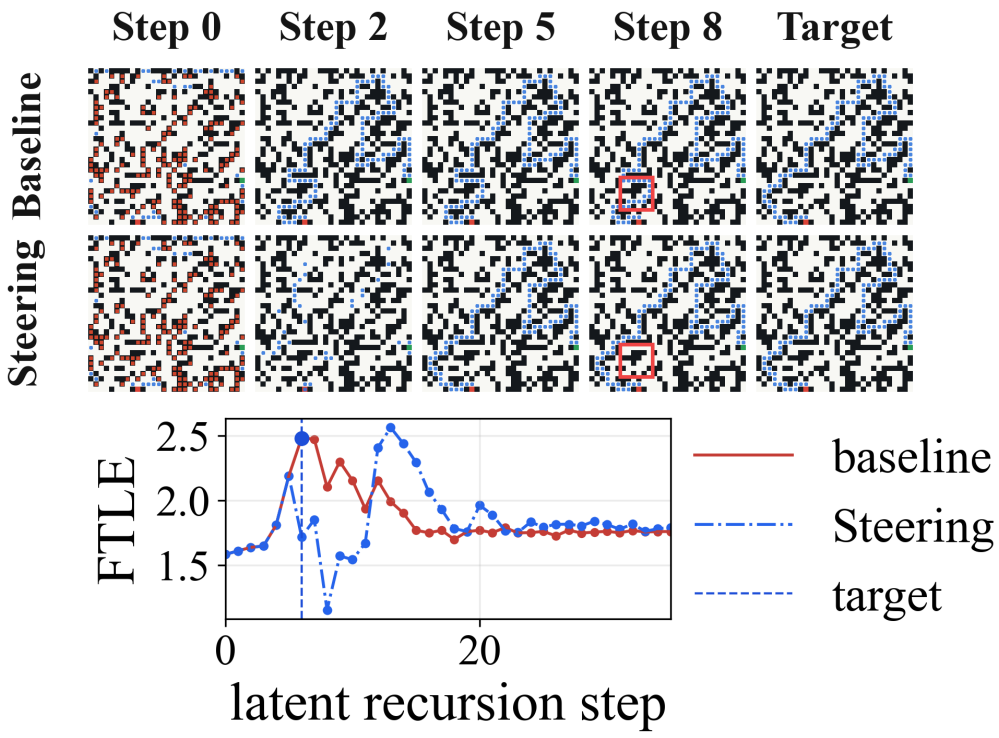
We present additional qualitative results for FTLE-guided trajectory corrections on the Maze task. Each sample corresponds to a baseline-failed case in which the unperturbed rollout settles on an incorrect decoded path, while FTLE-guided steering redirects the rollout toward a correct solution.

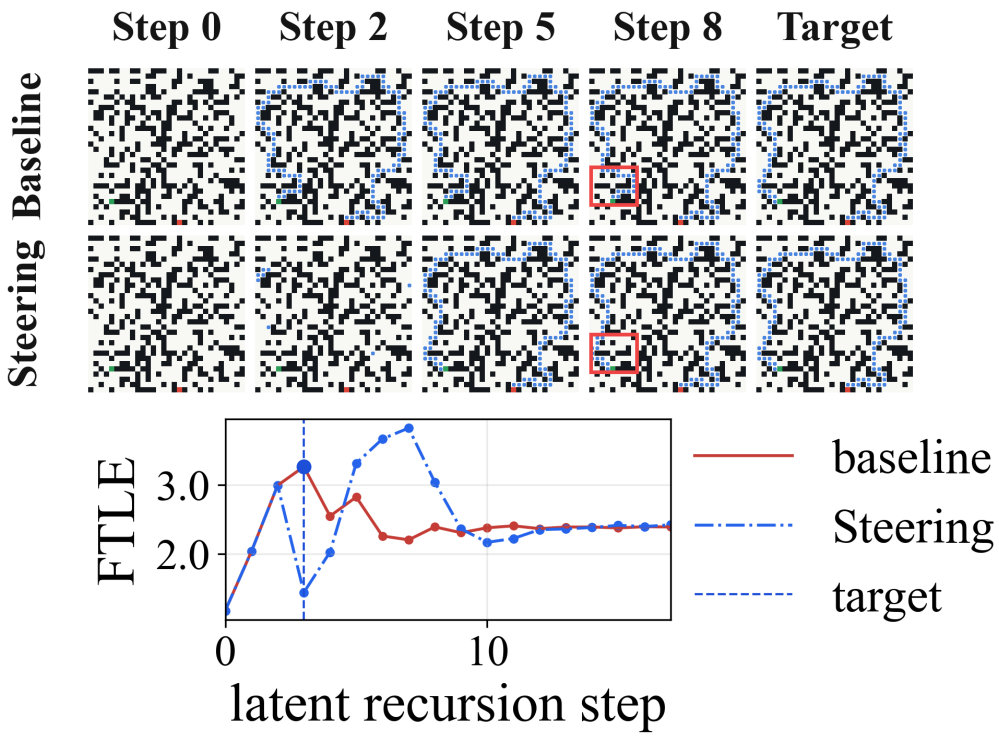
For each sample, the upper two panels show the predicted maze paths from the baseline rollout and the corrected rollout, respectively; the lower panel shows the FTLE curve along the latent trajectory, with the vertical dashed line marking the intervention point. These examples illustrate that perturbing near an early high-FTLE point can change the decoded path and subsequently lead to a target-consistent rollout. The accompanying FTLE profiles provide qualitative evidence that the intervention moves the trajectory away from the original failed branch and toward a different local regime, rather than merely extending the same deterministic rollout.

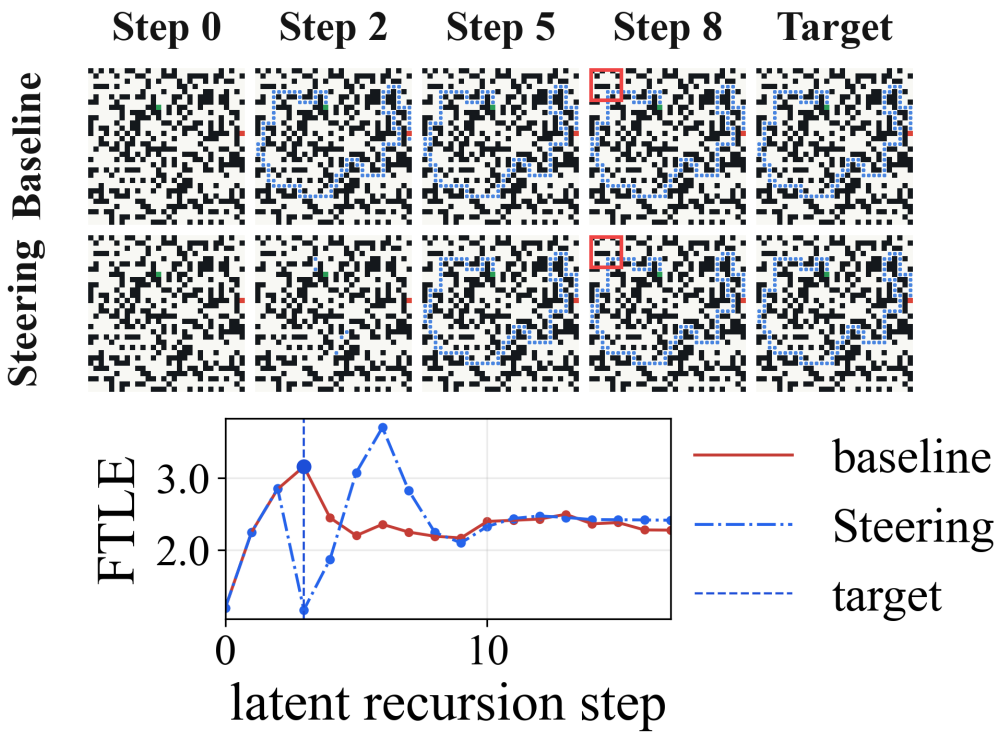

Step 0
Step 2
Step 5
Step 8
Target
Baseline
Steering
FTLE
2.0
1.5
0
20
latent recursion step
baseline
Steering
target

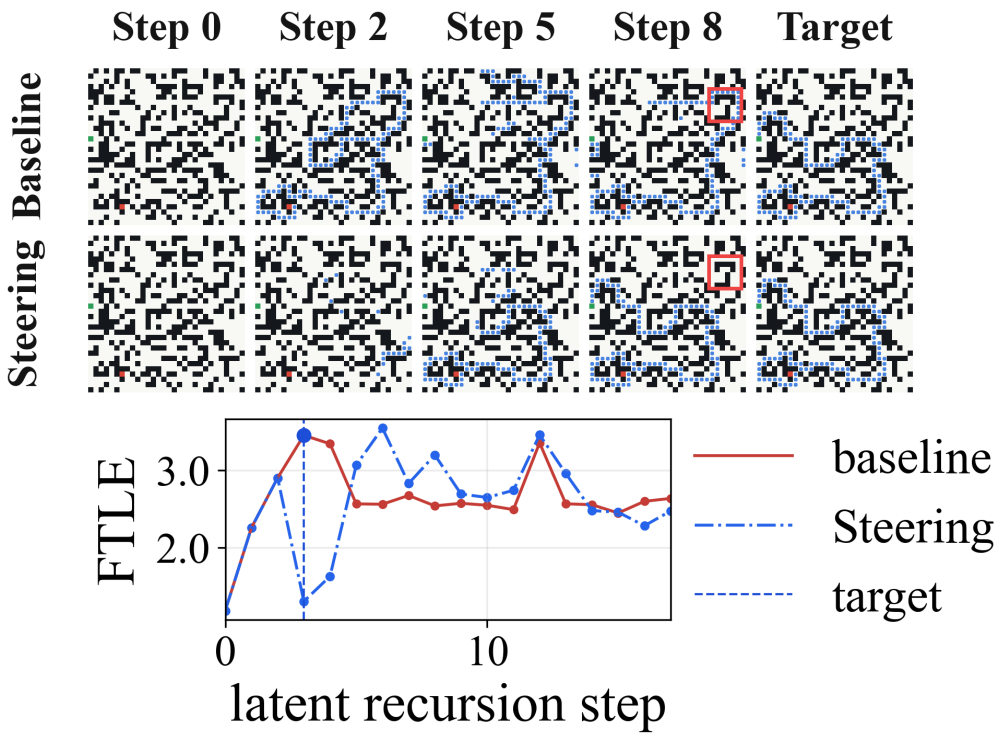

Step 0
Step 2
Step 5
Step 8
Target
Baseline
Steering
FTLE
3.0
2.0
0
10
latent recursion step
baseline
Steering
target

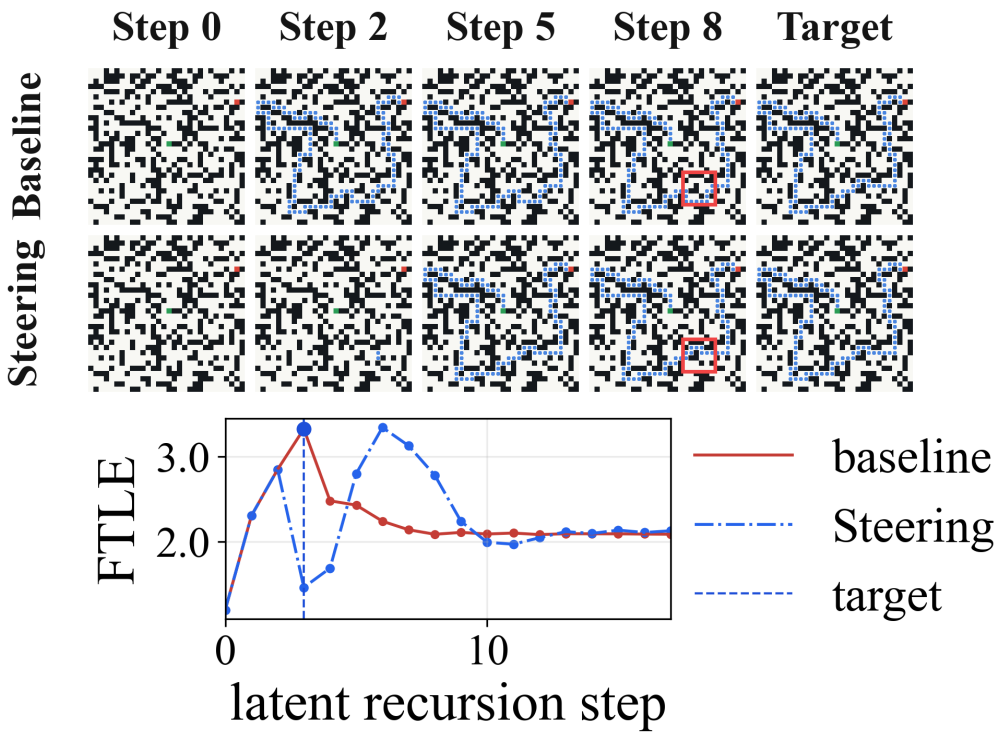

Step 0
Step 2
Step 5
Step 8
Target
Baseline
Steering
FTLE
2.5
2.0
1.5
0
20
latent recursion step
baseline
Steering
target
Step 0
Step 2
Step 5
Step 8
Target
Baseline
Steering
FTLE
2.5
2.0
1.5
0
20
latent recursion step
baseline
Steering
target


Step 0
Step 2
Step 5
Step 8
Target
Baseline
Steering
FTLE
2.5
2.0
1.5
0
20
latent recursion step
baseline
Steering
target


Step 0
Step 2
Step 5
Step 8
Target
Baseline
Steering
FTLE
3.0
2.0
0
10
latent recursion step
baseline
Steering
target


Step 0
Step 2
Step 5
Step 8
Target
Baseline
Steering
FTLE
3.0
2.0
0
10
latent recursion step
baseline
Steering
target

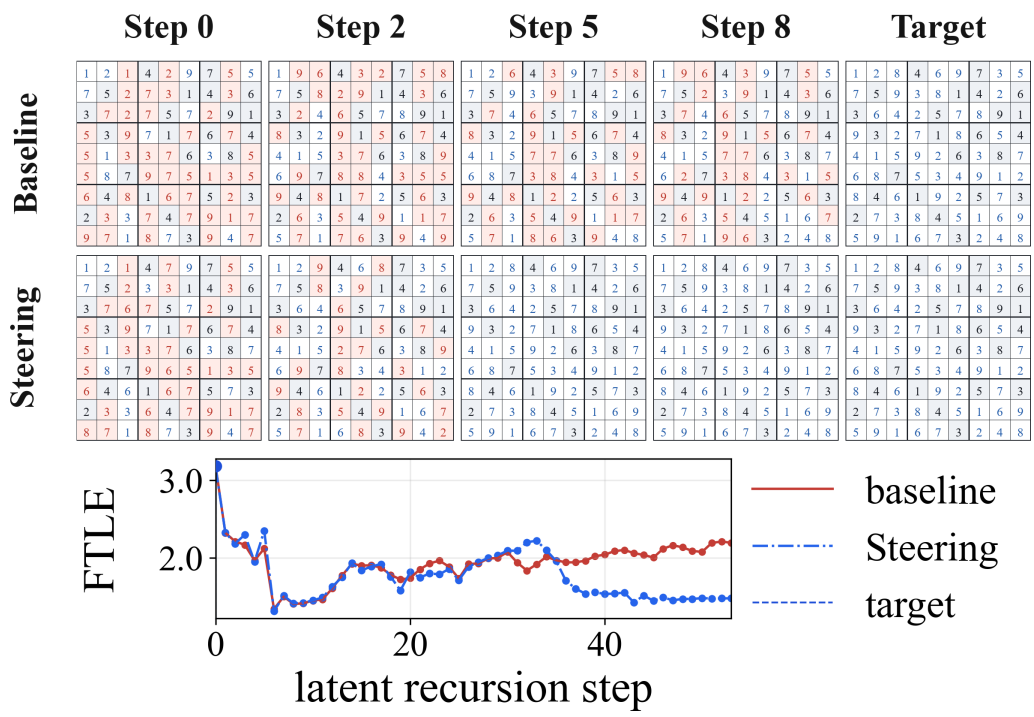

Step 0
Step 2
Step 5
Step 8
Target
Baseline
Steering
FTLE
3.0
2.0
0
10
latent recursion step
baseline
Steering
target

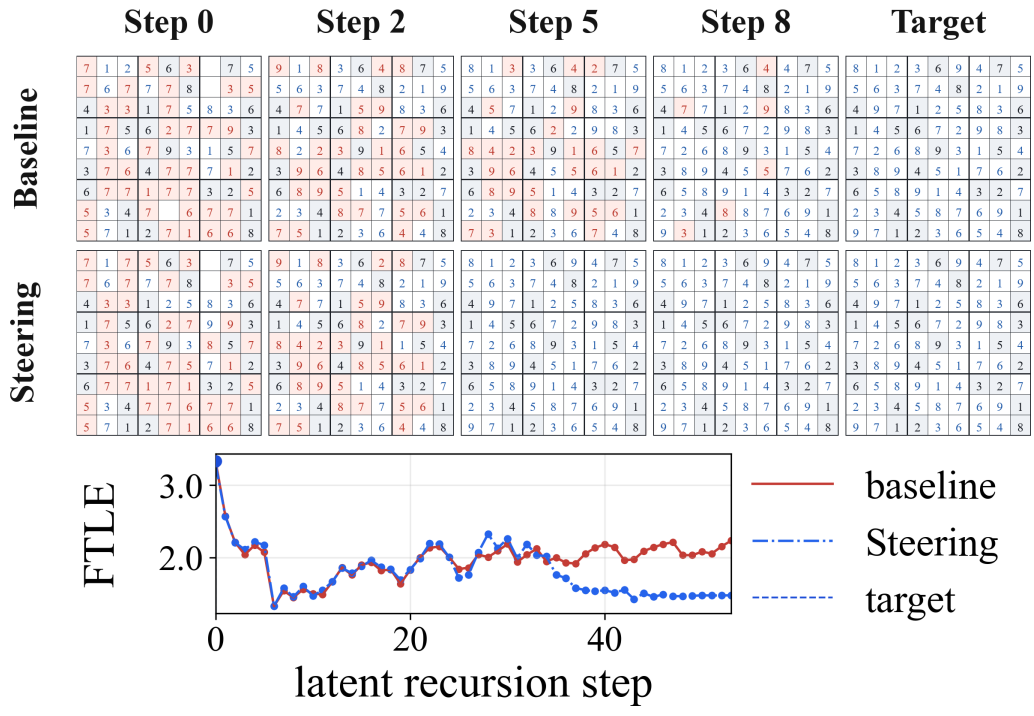


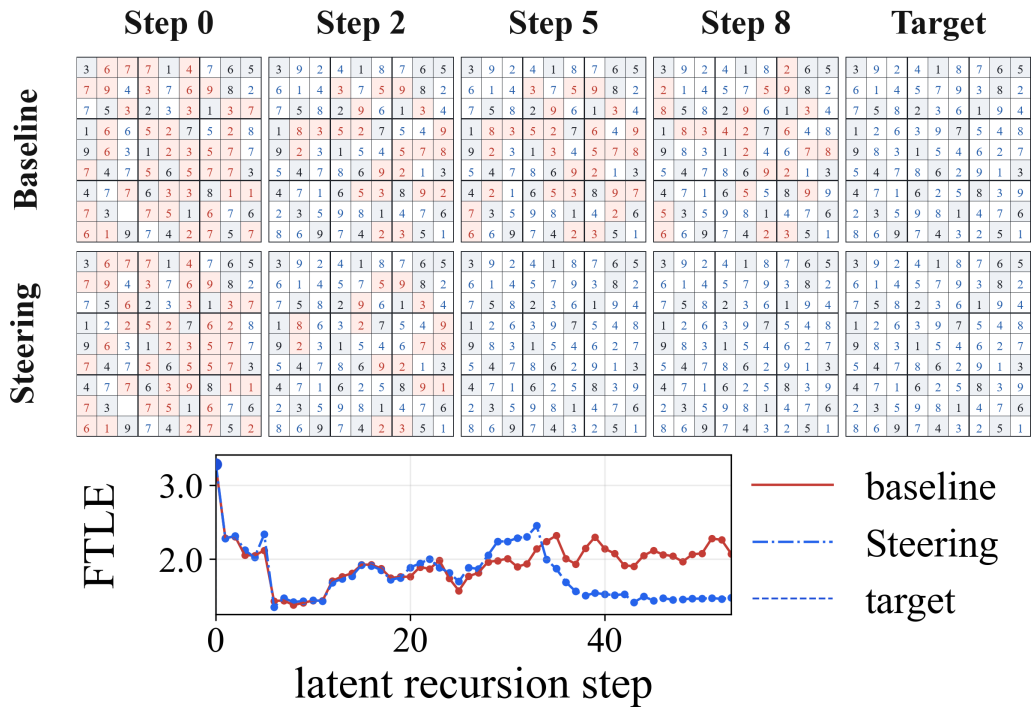
F.3 FTLE-Guided Corrections on Sudoku Samples

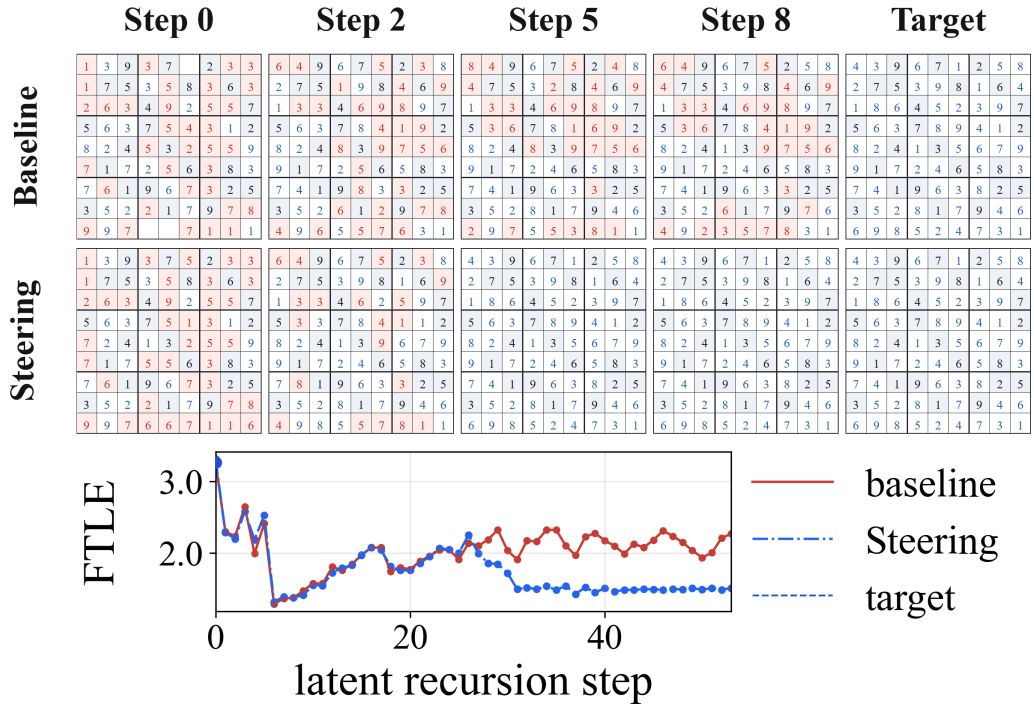
We provide additional qualitative results for FTLE-guided trajectory corrections on the Sudoku-Extreme task. For these samples, the intervention is applied at the highest FTLE step (usually first latent step) and the trajectory is locally most sensitive to perturbation. Even a small perturbation at this point can redirect the rollout toward a correct decoded grid.

For each sample, the upper two panels show the predicted Sudoku grids from the baseline rollout and the corrected rollout; the lower panel shows the FTLE curve, with the intervention marked at the first step. The post-intervention FTLE profiles are consistent with a transition away from the original failed branch, while the decoded sudoku grids show that the perturbed rollout reaches a valid solution.


Step 0
Step 2
Step 5
Step 8
Target
Baseline
Steering
FTLE
3.0
2.0
0
20
40
latent recursion step
baseline
Steering
target


Step 0
Step 2
Step 5
Step 8
Target
Baseline
Steering
FTLE
3.0
2.0
0
20
40
latent recursion step
baseline
Steering
target





F.4 Detailed Results Across Tasks and ACT Budgets

This subsection reports comprehensive tabulated results across the evaluated ACT budgets for Sudoku-Extreme, Maze-Hard, PPBench, and ARC-AGI-2. Unlike the compact comparisons presented in the main text, the tables below retain lower-budget regimes and intermediate ACT settings, including settings in which steering yields no improvement over the corresponding baseline. Results

Table 7: Sudoku exact accuracy under fixed ACT budgets. Results are reported as mean ± standard deviation over three random seeds when available.

Method	10 ACT steps	16 ACT steps
TRM	$85.15 \pm 0.25\%$	$89.19 \pm 0.33\%$
TRM + Steering	$97.90 \pm 0.10\%$	$98.43 \pm 0.07\%$
HRM	$51.77 \pm 0.96\%$	$55.25 \pm 0.85\%$
HRM + Steering	$57.23 \pm 0.77\%$	$62.48 \pm 0.91\%$

Table 8: Maze exact accuracy under different ACT budgets using Trajectory Energy selection. The Maze test set contains 1,000 examples, so single-run accuracies are reported.

Method	1 step	2 steps	3 steps	5 steps	10 steps	16 steps
TRM	74.60%	85.10%	85.20%	85.00%	85.10%	85.10%
TRM + Steering	0.40%	**87.30%**	**90.40%**	**91.00%**	**91.00%**	**91.00%**
HRM	16.40%	70.70%	73.30%	73.90%	74.40%	74.40%
HRM + Steering	0.30%	35.20%	**77.70%**	**80.00%**	**79.40%**	**79.40%**

for Sudoku-Extreme are averaged over three random seeds, whereas Maze-Hard and PPBench follow their respective fixed-test-set protocols. Results for ARC-AGI-2 are evaluated on all 120 tasks under matched augmentation and stochastic-rollout budgets.

F.5 Trajectory-space FTLE Slices on Additional Samples

We further examine the trajectory-space geometry on five representative Sudoku-Extreme test cases. For each case, we compare a successful rollout with an unsuccessful rollout and construct a latent slice by interpolating between the two trajectories at matched reasoning steps. FTLE is then evaluated on this interpolated slice, so that the resulting heat map probes the region separating the two observed inference outcomes rather than a hand-chosen two-dimensional projection.

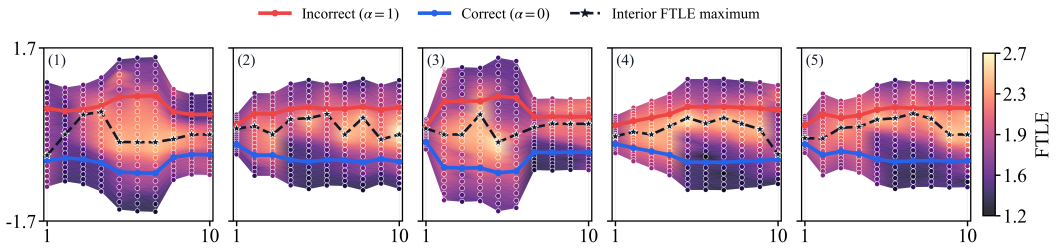


Figure 7: Five Sudoku trajectory-space slices. Interior FTLE maxima persist between correct and incorrect rollouts across ACT steps, consistent with a reproducible candidate LCS-like boundary pattern.

Figure 7 shows that the strongest FTLE responses appear in the interior region between the correct and incorrect trajectories, rather than only at either endpoint. Across the five cases, these high-FTLE regions form ridge-like structures that persist over multiple reasoning steps. This pattern is consistent with our central interpretation: successful and failed rollouts evolve toward different attractor-like regimes, and the high-FTLE ridge between them marks a locally sensitive region where nearby trajectories can separate.

These examples are not intended as an additional hyperparameter study. Instead, they provide a qualitative reproducibility check for the candidate-boundary hypothesis in the Sudoku setting. The repeated appearance of an interior FTLE ridge supports the view that the steering method acts near a meaningful geometric structure in trajectory space, rather than exploiting a single idiosyncratic sample.

Table 9: PPBench exact accuracy on the 49-example, golden subset under fixed ACT budgets. All methods use the same TRM checkpoint; PTRM and FTLE steering use $K = 100$ stochastic rollouts. At 48 steps, FTLE steering matches PTRM without parameter updates.

Method	ACT 1	ACT 2	ACT 3	ACT 5	ACT 10	ACT 16
TRM	30.6%	42.9%	46.9%	53.1%	53.1%	53.1%
PTRM (best-Q@100)	32.7%	63.3%	65.3%	69.4%	71.4%	69.4%
TRM + FTLE Steering (Ours)	4.1%	40.8%	53.1%	67.3%	**71.4%**	**71.4%**

Table 10: ARC-AGI-2 exact solved-task accuracy across ACT budgets under matched augmentation and rollout budgets. All methods use the released TRM checkpoint and all 120 evaluation tasks with 100 input augmentations. PTRM and FTLE steering use $K = 25$ stochastic rollouts per augmentation. PTRM uses its native Q-head selector with $\sigma = 0.2$; FTLE steering uses energy-based basin aggregation with $(\sigma_L, \sigma_H) = (1.0, 0.5)$.

ACT	Method	pass@1	pass@2	pass@5
1	TRM + Augmentation	2.5%	3.3%	5.0%
1	PTRM	3.3%	3.3%	5.8%
1	TRM + FTLE Steering (ours)	2.5%	2.5%	5.0%
2	TRM + Augmentation	4.2%	5.0%	6.7%
2	PTRM	4.2%	5.0%	5.8%
2	TRM + FTLE Steering (ours)	5.0%	5.8%	7.5%
3	TRM + Augmentation	4.2%	5.0%	5.8%
3	PTRM	4.2%	5.8%	6.7%
3	TRM + FTLE Steering (ours)	5.0%	5.0%	7.5%
5	TRM + Augmentation	4.2%	4.2%	6.7%
5	PTRM	4.2%	5.8%	6.7%
5	TRM + FTLE Steering (ours)	4.2%	5.0%	**8.3%**
10	TRM + Augmentation	4.2%	4.2%	5.8%
10	PTRM	4.2%	4.2%	6.7%
10	TRM + FTLE Steering (ours)	4.2%	5.0%	**8.3%**
16	TRM + Augmentation	4.2%	4.2%	6.7%
16	PTRM	4.2%	4.2%	6.7%
16	TRM + FTLE Steering (ours)	4.2%	5.0%	**8.3%**