LOOPREG: REPRESENTATION SUPERVISION ENABLES TEST-TIME RECURRENT DEPTH IN DIFFUSION TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reusing Transformer blocks offers a way to increase diffusion-model computation at test time, but naive recurrence can degrade generation beyond its training depth. We introduce LoopREG, a representation-supervised recurrent diffusion Transformer that repeatedly applies a shared block at the state supervised by a pretrained visual encoder. Alignment is applied after the final recurrent visit during training; inference extends the same learned operator to additional visits. On ImageNet 256×256, LoopREG reduces SiT-B/2 FID from the local REG baseline's 15.08 to 4.91 at 400K training steps, a 67.41% reduction. Within a checkpoint trained with two visits, one extra inference visit improves FID from 14.71 to 5.34 with unchanged weights. The benefit persists on SiT-L/2 and SiT-XL/2. Controlled interventions show that useful recurrence follows the location of representation supervision, while REPA alignment produces a smaller extrapolation benefit. These results make representation-supervised recurrence a practical route to improving diffusion generation through additional inference depth.

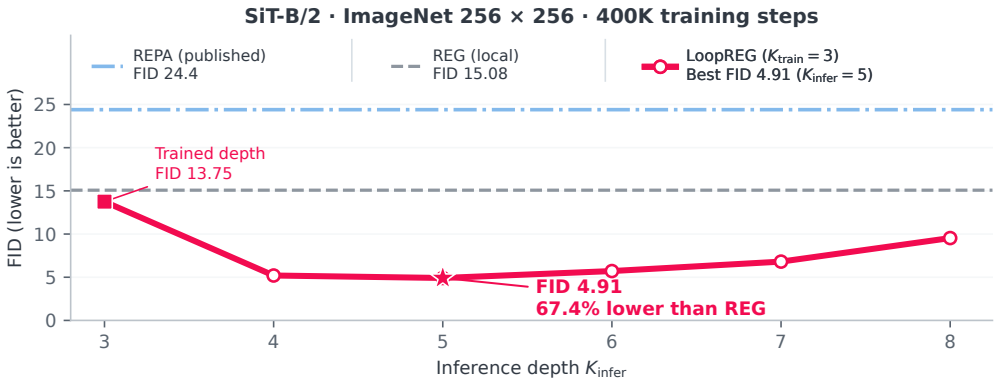

Figure 1: **LoopREG reduces REG's FID by 67.41% on SiT-B/2.** At 400K training steps, horizontal references show published REPA (24.40; Wu et al., 2025) and local REG (15.08). LoopREG ($K_{\text{train}} = 3$) reaches FID 4.91 at $K_{\text{infer}} = 5$ (star). Figure 3 gives full sweeps. Local evaluations: EMA, 50K samples, 250 denoiser calls, CFG=1.

1 INTRODUCTION

Diffusion Transformers such as DiT and SiT devote a fixed stack of blocks to each denoiser evaluation (Peebles & Xie, 2023; Ma et al., 2024). Increasing the number of sampling steps adds computation across diffusion time. Shared recurrence offers a complementary possibility: spend more computation refining the hidden state within each evaluation, while reusing the same parameters. For this to improve generation, a learned block must remain useful when applied beyond its training horizon.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

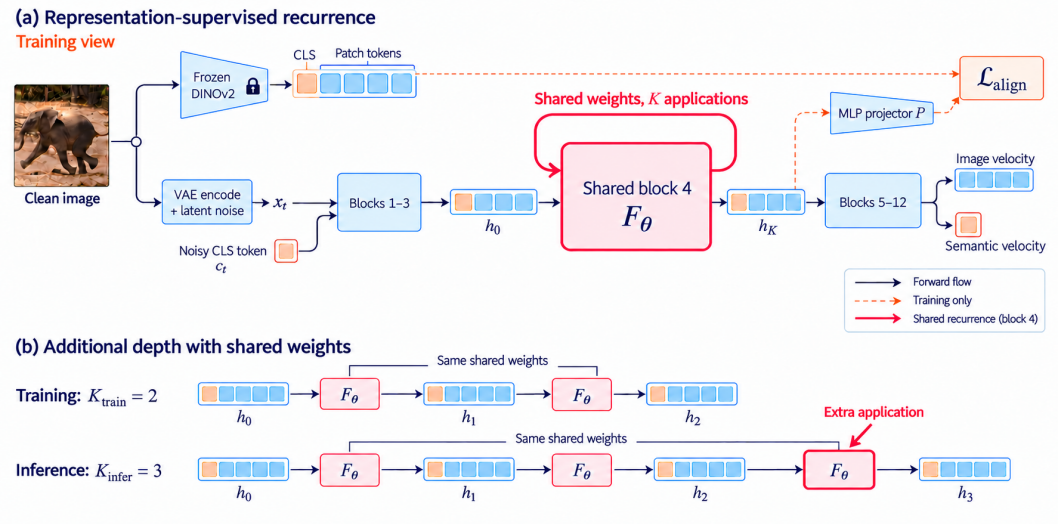

Figure 2: **LoopREG: representation-supervised shared recurrence.** SiT-B/2 is shown. (a) One clean image supplies both the VAE latent, noised after encoding, and frozen DINOv2 targets. Orange denotes CLS/semantic tokens; blue denotes image/patch tokens. Alignment reads the final recurrent state through a projector. (b) Shared weights are reused beyond the training horizon, illustrated by $K_{\text{train}} = 2$ and $K_{\text{infer}} = 3$. Sampling starts from noise.

What makes recurrent computation reusable beyond its training horizon? Our experiments identify representation supervision as an enabling ingredient. Naively repeating a block in plain SiT yields at most a marginal matched-depth improvement and degrades sharply with an extra inference visit. The same recurrent construction becomes useful under REPA alignment and substantially more effective under REG's representation entanglement (Yu et al., 2025; Wu et al., 2025). This contrast connects the value of extra computation to how the recurrent state is supervised.

We introduce **LoopREG**, a recurrent diffusion Transformer built on REG. LoopREG reuses one Transformer block at the representation-supervised state, applies alignment after the final training-time visit, and extends that block's trajectory at inference (Figure 2). It retains REG's joint image and semantic-token denoising objective. Inference adds visits with unchanged weights and a fixed diffusion sampling schedule.

The resulting generation gains are substantial. On ImageNet 256×256 at 400K training steps, LoopREG-B/2 reaches FID 4.91, compared with 15.08 for the local REG baseline: a 67.41% reduction (Figure 1). Within a LoopREG checkpoint trained with two visits, a third visit reduces FID from 14.71 to 5.34. Useful extra depth also persists on SiT-L/2 and SiT-XL/2. On XL/2, extra inference depth reaches the matched-depth 400K quality at 220K training steps.

The key design choice is *where* to recur. With supervision fixed at block four, extra recurrence helps at the supervised state and fails at the other tested sites. Moving supervision and recurrence together restores the benefit at other depths. A compact analysis relates this placement to direct supervision of the recurrent endpoint and interprets the finite useful inference horizon.

Our main contributions are:

1. **LoopREG.** We introduce representation-supervised shared recurrence for diffusion Transformers, enabling additional inference depth with the same trained parameters.
2. **Substantial generation gains.** LoopREG lowers SiT-B/2 FID from 15.08 to 4.91, with useful recurrent-depth extrapolation across B/2, L/2, and XL/2. Training-horizon sweeps characterize the resulting quality–depth curves.
3. **A supervision-based design principle.** Plain SiT/REPA/REG controls and site/depth interventions show that representation supervision and its location determine whether extra recurrent computation is useful in the tested settings.

2 Related Work

Diffusion and representation supervision. DiT replaces the denoising backbone with a Transformer, and SiT studies flow and diffusion within a stochastic-interpolant framework (Peebles & Xie, 2023; Ma et al., 2024). REPA aligns projected intermediate states with clean pretrained visual representations (Yu et al., 2025). REG jointly denoises image latents and a semantic token while retaining representation alignment (Wu et al., 2025). SRA instead uses the model's own EMA representations for alignment (Jiang et al., 2026). We study a further capability of representation supervision: enabling a recurrent operator to improve generation beyond its training horizon. The supervision-site interventions identify where this capability emerges.

Looped Transformers and latent computation. Universal Transformers combine shared depth with adaptive computation (Dehghani et al., 2019). Theoretical studies show that looped Transformers can implement programs (Giannou et al., 2023), learn iterative data-fitting procedures (Yang et al., 2024), and realize multistep gradient descent in controlled in-context-learning settings (Gatmiry et al., 2024). Recurrent-depth language models explore scaling latent computation beyond a fixed forward pass (Geiping et al., 2025). We investigate when repeated computation improves a diffusion denoiser, using representation supervision as the central intervention.

Elastic depth and visual generation. LoopFormer conditions recurrent updates on time and step size and trains shortcut-consistent trajectories (Jeddi et al., 2026); PonderNet learns a stopping distribution (Banino et al., 2021). ELT explores looped visual generation with intra-loop self-distillation and reports gains beyond training depth (Goyal et al., 2026). We instead investigate external representation supervision and its location, connecting recurrent-depth extrapolation to supervised-state dynamics. Our RandomLoop control samples terminal horizons from finite uniform distributions; heavy-tailed random horizons are also used in recurrent-depth language-model training (Geiping et al., 2025).

Dynamics and inference-time reuse. Deep equilibrium models learn implicitly defined fixed points (Bai et al., 2019). Recurrent-dynamics work distinguishes regimes in which extra iterations help, preserve, or degrade predictions (Viakhirev et al., 2026); training-free looping uses damped updates to mitigate failures of naive reapplication (Chen et al., 2026). LoopREG trains the shared operator explicitly and reuses its learned update at inference. Internal Guidance combines intermediate and final denoising predictions to guide sampling (Zhou et al., 2026). LoopREG varies the recurrent depth of a representation-supervised block within each denoiser call. Its generation benefit depends on the supervision location and the inference horizon.

3 LoopREG

LoopREG adds shared recurrence to a representation-supervised diffusion Transformer. Its defining choice is to repeat the block whose output receives representation supervision, so that the supervised state is also the state refined by extra inference computation.

3.1 Representation-Supervised Shared Recurrence

Image and representation inputs. Let x_0 be the VAE latent of a clean training image, y its class label, and $z = (z_{\mathrm{cls}}, z_{\mathrm{patch}})$ the frozen DINOv2 features of the same image. Following REG, we form an image state and a semantic-token state at a common time $t \sim \mathcal{U}[0, 1]$:

$$x_t = (1 - t)x_0 + t\epsilon, \qquad c_t = (1 - t)z_{\mathrm{cls}} + t\epsilon_c, \tag{1}$$

where ϵ and ϵ_c are independent Gaussian noise. The Transformer processes image tokens and the semantic token together, conditioned on t and y, and predicts both velocities. Figure 2 shows how the clean image supplies the latent and the alignment targets.

Reusing the supervised block. Write the blocks before the recurrent site as H, the shared block as F_θ, and the remaining blocks and output projection as G_ϕ. At a fixed sampling state $u_t = (x_t, c_t)$,

$$h_0 = H(u_t; t, y), \quad h_{k+1} = F_\theta(h_k; t, y), \quad (\hat{v}^x_K, \hat{v}^c_K) = G_\phi(h_K; t, y). \tag{2}$$

The depth K counts all applications of the shared block, including its ordinary first visit. Thus $K = 1$ executes it once and $K = 3$ executes the same module three times. Each visit receives the preceding visit's state with the same time and class conditioning. LoopREG uses no recurrent-index embedding or per-visit parameters.

The supervision depth d counts Transformer blocks in the original backbone. We co-locate the recurrent block at $\ell = d$ and read the representation after its *final* recurrent visit. A nonlinear projector P maps that endpoint to the clean teacher features. Alignment is evaluated once per forward pass, so the training objective supervises the endpoint of the learned trajectory. The main B/2 model uses $d = \ell = 4$, and L/2 and XL/2 use $d = \ell = 8$. Block labels here are one-based.

3.2 Training and Inference

Training objective. Let M count teacher tokens, including the CLS token, and let bars denote unit normalization along the feature dimension. LoopREG retains REG's negative-cosine alignment and joint velocity losses:

$$\mathcal{L}_{\mathrm{rep}}(h_K, z) = -\frac{1}{M}\sum_{i=1}^{M} \langle \overline{P(h_K)_i}, \bar{z}_i \rangle, \tag{3}$$

$$\mathcal{L} = \mathbb{E}[\mathrm{MSE}(\hat{v}_K^x, \epsilon - x_0) + 0.03\,\mathrm{MSE}(\hat{v}_K^c, \epsilon_c - z_{\mathrm{cls}}) + 0.5\mathcal{L}_{\mathrm{rep}}(h_K, z)]\,. \tag{4}$$

MSE averages over the coordinates of each prediction; the expectation averages over training examples, noise, and time. Fixed-depth training uses $K = K_{\mathrm{train}}$ throughout. Since every visit shares F_θ, gradients update one operator over the states induced by its recurrent trajectory. RandomLoop instead samples a terminal horizon per optimization step, shared across distributed ranks, to train multiple usable endpoints.

Extending depth at inference. Generation initializes both x and c from noise. Each denoiser call evaluates Equation (2) at a chosen $K = K_{\mathrm{infer}}$, and the sampler updates both states using the predicted velocities. The frozen visual teacher and clean representations are used only during training. Changing K_{infer} changes the number of shared-block applications within a call, while preserving all learned parameters and the diffusion time grid. For example, a model trained with $K_{\mathrm{train}} = 2$ can be evaluated with $K_{\mathrm{infer}} = 3$ to add one recurrent refinement at every sampling step.

Computation. For a backbone with D blocks and a recurrent core of q blocks, the number of block executions per denoiser call is

$$B(K) = D + (K - 1)q. \tag{5}$$

The main model uses $q = 1$, so one extra visit adds one block execution. For B/2, the trained $K_{\mathrm{train}} = 2$ call executes 13 blocks and $K_{\mathrm{infer}} = 3$ executes 14. Shared recurrence adds computation without adding copies of the block's weights. We report block executions alongside generation metrics; measured runtime also depends on the sampler and implementation.

Controlled variants. Plain LoopSiT uses the same recurrence with image-velocity regression alone. LoopREPA adds patch-token alignment with coefficient 0.5 and has no REG semantic-token path or semantic loss. Independent-copy controls replace the shared operator with separately trained blocks. These variants distinguish the effects of representation supervision, REG's joint representation objective, and parameter reuse.

3.3 Interpreting Supervised Recurrence

Co-location directly supervises the recurrent endpoint. If the loop is before supervision, let $T_{\ell \to d}$ map its endpoint to the supervised layer. The alignment gradient is

$$\nabla_{h_K} \mathcal{L}_{\mathrm{rep}} = J_T(h_K)^\top \nabla_{T(h_K)} \ell_{\mathrm{rep}}(P(T(h_K)), z). \tag{6}$$

At co-location, $T = I$: the representation objective reads the recurrent endpoint directly. Earlier loops receive that gradient through intervening blocks, while a later block with distinct parameters lies outside the alignment branch. This structure provides a reason to co-locate recurrence and supervision. Section 4.3 tests the design by moving the two sites independently and jointly.

A shared operator learns from its own recurrent trajectory. Supervising the endpoint also trains the transitions that reach it. Let $g_k = \nabla_{h_k}\mathcal{L}$ and let θ contain the shared block's parameters. Backpropagation through $K = K_{\text{train}}$ visits gives

$$g_k = J_h F_\theta(h_k)^\top g_{k+1}, \qquad \nabla_\theta \mathcal{L} = \sum_{k=0}^{K-1} J_\theta F_\theta(h_k)^\top g_{k+1}. \tag{7}$$

Time and class conditioning are held fixed within this recurrence. The same parameters receive updates at each visited state, including states produced by their own earlier applications. Co-location makes representation alignment part of the endpoint signal g_K. These two choices work together: alignment shapes the state at which recurrence ends, while sharing trains one transition over the path leading there. An unshared stack assigns these transitions to different parameter sets. This explains the design rationale for a reusable refinement operator, with its useful test-time horizon determined by the measured trajectory.

Useful refinement can end before equilibrium. Additional visits trace a longer hidden-state trajectory. Even a stable trajectory can pass through a task-preferred state before its fixed point. For the recurrence $h_{k+1} = h_\infty + \lambda(h_k - h_\infty)$, $0 < \lambda < 1$, and a preferred state $h_g = h_\infty + r(h_0 - h_\infty)$, $0 < r < 1$,

$$\|h_k - h_g\|^2 = \|h_0 - h_\infty\|^2(\lambda^k - r)^2. \tag{8}$$

The error first decreases and then increases as the trajectory passes h_g. This finite-horizon interpretation is consistent with LoopREG's observed FID minima beyond training and its later degradation. Appendix B gives the assumptions, formal statements, and derivations. We select inference depths from the measured generation curves.

4 Experiments

Protocol. We evaluate class-conditional ImageNet at 256×256, using checkpoints trained for 400K optimization steps with global batch 256. Main sweeps use EMA weights, 50K generated samples, 250 denoiser calls with an Euler–Maruyama SDE sampler, and CFG=1. We use the local ImageNet reference and OpenAI Inception evaluator. Appendix A gives the full sampling settings and provenance.

4.1 Generation Gains from Test-Time Depth

LoopREG delivers large gains over REG and its trained endpoint. For LoopREG-B/2 trained at $K_{\text{train}} = 2$, FID improves from 14.71 to 5.34 and remains below the matched value at $K_{\text{infer}} = 4$. The relative reduction at $K_{\text{infer}} = 3$ is 63.67%. Increasing the training horizon shifts the minimum: $K_{\text{train}} = 3$ reaches FID 4.91 at $K_{\text{infer}} = 5$, while $K_{\text{train}} = 4$ reaches 5.19 at $K_{\text{infer}} = 6$. Relative to the local REG baseline at 15.08, the best result reduces FID by 67.41%. Figure 3 shows how all three recurrent horizons cross below the baseline before reaching their finite optima. Within-checkpoint sweeps use unchanged weights. Across training horizons, equal optimization steps involve different training compute, as quantified by Equation (5).

Representation supervision unlocks recurrent extrapolation. Plain LoopSiT at matched depth slightly improves on the local SiT baseline (FID 35.25 to 34.45), but an additional visit worsens FID to 46.41. LoopREPA improves by 23.64% from $K_{\text{infer}} = 2$ to $K_{\text{infer}} = 3$, despite having no REG semantic token, then degrades at $K_{\text{infer}} = 4$. Alignment alone therefore enables useful extrapolation in this control, while REG substantially increases the gain and useful recurrent horizon.

4.2 Scaling to Larger Backbones

The extrapolation benefit persists as the backbone grows. At 400K steps, LoopREG-L/2 improves from FID 4.23 at $K_{\text{infer}} = 2$ to 3.21 at $K_{\text{infer}} = 3$. LoopREG-XL/2 uses 28 blocks, supervision at depth eight, and shared block eight. It improves from 3.3823 to 2.8376, a 16.10% reduction, with sFID also improving. Table 1 places these results beside the published SiT and REG baselines at each scale. Together with the B/2 results, these experiments establish useful recurrent extrapolation across three model scales.

Model	Steps	K_{train}	K_{infer}	FID ↓	sFID ↓	IS ↑	P ↑	R ↑
SiT-B/2 (local)	400K	1	1	35.25	6.65	42.14	0.530	0.650
REPA-B/2*	400K	–	–	24.40	6.40	59.90	0.590	0.650
REG-B/2 (local)	400K	1	1	15.08	6.73	95.53	0.649	0.640
LoopSiT-B/2	400K	2	2	**34.45**	**6.43**	**43.01**	**0.533**	**0.639**
LoopSiT-B/2	400K	2	3	46.41	10.01	34.28	0.464	0.618
LoopREPA-B/2	400K	2	2	21.93	**6.37**	66.58	0.601	**0.651**
LoopREPA-B/2	400K	2	3	**16.74**	6.68	**84.67**	**0.629**	0.615
LoopREG-B/2	400K	2	2	14.71	6.76	97.13	0.650	**0.631**
LoopREG-B/2	400K	2	3	**5.34**	**5.80**	**185.98**	**0.796**	0.508
LoopREG-B/2	400K	3	3	13.75	6.69	100.82	0.658	**0.634**
LoopREG-B/2	400K	3	5	**4.91**	**5.40**	**217.61**	**0.851**	0.452
SiT-L/2*	400K	–	–	18.80	5.29	72.00	0.640	0.640
REPA-L/2*	400K	–	–	10.00	5.20	109.20	0.690	0.650
REG-L/2*	400K	–	–	4.60	5.21	167.60	0.750	0.630
LoopREG-L/2	400K	2	2	4.23	5.09	170.36	0.752	**0.642**
LoopREG-L/2	400K	2	3	**3.21**	**4.44**	**262.66**	**0.859**	0.520
SiT-XL/2*	7M	–	–	8.30	6.32	131.70	0.680	0.670
REPA-XL/2*	4M	–	–	5.90	5.73	157.80	0.700	0.690
REG-XL/2*	400K	–	–	3.40	4.87	184.10	0.760	0.640
LoopREG-XL/2	400K	2	2	3.38	4.85	184.71	0.758	**0.635**
LoopREG-XL/2	400K	2	3	**2.84**	**4.64**	**254.70**	**0.839**	0.549

Table 1: **Generation quality across model scales.** Horizontal rules separate backbones; spacing separates method families. Shaded rows are baselines; an asterisk denotes published results from Wu et al. (2025), Table 9 (SiT/REPA attributed there to Ma et al. (2024); Yu et al. (2025)). Training steps are shown explicitly; the published REPA-XL/2 reference uses 4M steps. All other rows are local evaluations. Bold marks the best displayed metric within each multi-row checkpoint group; best depths are selected within the reported sweeps. P/R: precision/recall. Appendix A gives provenance, including the L/2 server summary.

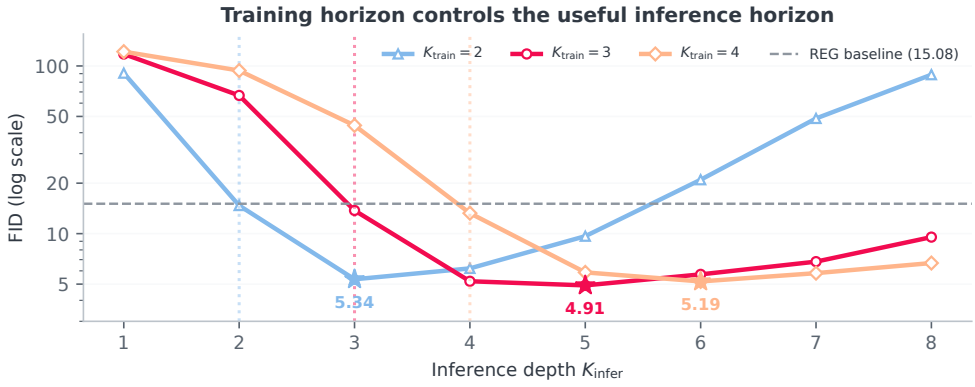

Figure 3: **Training horizon shifts the useful inference horizon.** Dotted lines mark training horizons; stars and labels mark the best observed FID. The dashed horizontal line shows the local REG baseline. All three training horizons produce large improvements over this baseline, with optima moving to later recurrent depths. The logarithmic vertical axis preserves the severe under-roll failures.

One extra block application yields a large quality return. Table 2 isolates the added recurrent depth within each checkpoint. Moving from $K_{\text{infer}} = 2$ to 3 adds one shared-block execution per denoiser call across all three backbones, with unchanged weights and the same 250-call sampler. The relative increase in block executions becomes smaller for deeper backbones, while each model

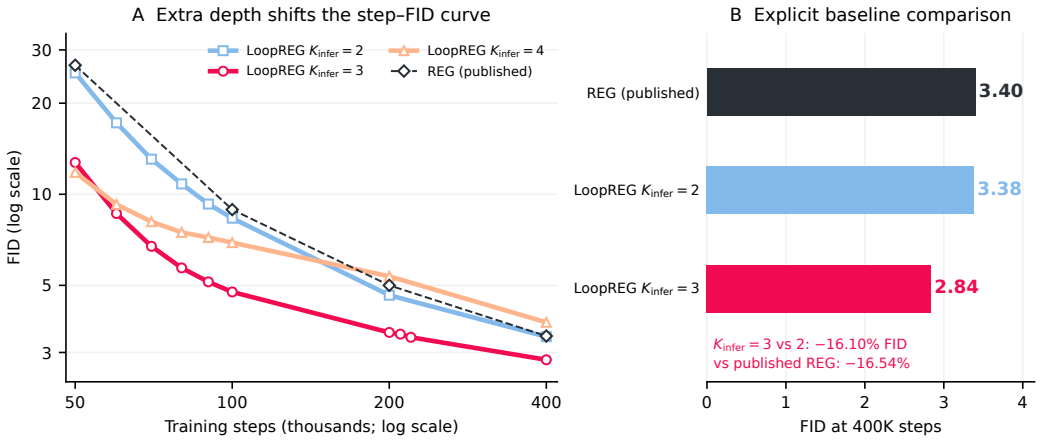

Figure 4: **Extra recurrent computation improves the optimization-step–FID curve.** Left: fixed $K_{\text{infer}} = 2, 3, 4$ curves from 50K steps along one XL training run, with published REG reference points. Right: 400K results; $K_{\text{infer}} = 2, 3$ share one checkpoint, while REG is a published reference. Lines connect evaluated checkpoints; 210K/220K include $K_{\text{infer}} = 3$ only.

retains a generation benefit. This gives LoopREG a direct way to spend additional computation inside the denoiser, complementary to increasing the number of sampling steps. Block counts measure executed Transformer depth; their runtime cost also depends on the backbone and implementation.

Backbone	Blocks/call	Added blocks (%)	FID (K_{infer}: 2 → 3)	FID reduction (%)
SiT-B/2	13→14	7.69	14.71→5.34	**63.67**
SiT-L/2	25→26	4.00	4.23→3.21	**24.13**
SiT-XL/2	29→30	3.45	3.38→2.84	**16.10**

Table 2: **Return from one extra shared-block application.** All models train with $K_{\text{train}} = 2$ for 400K steps. Added block executions are relative to matched $K_{\text{infer}} = 2$; FID reductions compare the same checkpoint at $K_{\text{infer}} = 2$ and 3.

Reaching lower FID with fewer optimization steps. Along the same XL training trajectory, 220K at $K_{\text{infer}} = 3$ reaches FID 3.3675, compared with 3.3823 at 400K and $K_{\text{infer}} = 2$: 45% fewer optimization steps at a slightly lower FID. Against published REG-XL thresholds (Wu et al., 2025), the earliest evaluated crossings use 40K/60K/100K/220K steps versus 50K/100K/200K/400K (Figure 4). These are optimization-step comparisons with additional inference recurrence; measured runtime savings require a separate assessment. Appendix F reports the full checkpoint grid and metric comparisons.

Fidelity and coverage. The 400K XL $K_{\text{infer}} = 2$ to 3 improvement accompanies higher precision and lower recall (0.6347 to 0.5490). For the 220K $K_{\text{infer}} = 3$ versus 400K $K_{\text{infer}} = 2$ comparison, sFID improves from 4.8470 to 4.5372, while recall falls to 0.5180. At 60K–100K, $K_{\text{infer}} = 3$ improves FID but slightly worsens sFID. Precision and recall in Table 1 make this fidelity–coverage change explicit. Training with random horizons offers an alternative when retaining coverage is a priority (Section 4.4).

4.3 Where Should LoopREG Recur?

Useful extrapolation depends on the supervised state. We first hold REG supervision at $d = 4$ and move the recurrent block. At matched $K_{\text{infer}} = 2$, all four tested sites have similar FID, but the unseen $K_{\text{infer}} = 3$ response separates them sharply (Figure 5 and Table 4). B4 improves, while B2, B6, and B8 degrade. Thus matched-depth quality alone does not predict useful extrapolation.

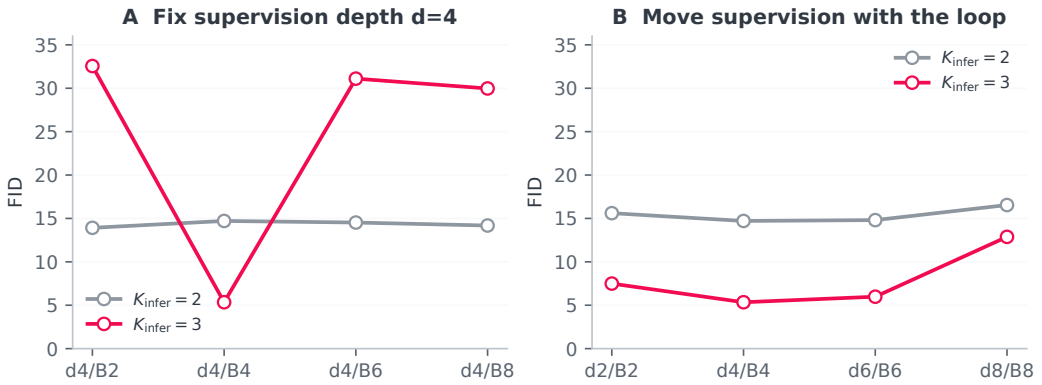

Figure 5: **Useful recurrence follows representation supervision in the tested interventions.** Left: fixed supervision at depth four, varied loop block. Right: supervision and recurrence moved together. Both panels reuse the d4/B4 anchor. Each model trains at $K_{\text{train}} = 2$.

We then move supervision with the loop. Co-located d2/B2, d6/B6, and d8/B8 each improve at $K_{\text{infer}} = 3$, with a weaker effect at depth eight. The benefit moves with supervision, supporting the co-located design in Section 3.3. The two groups share the same d4/B4 anchor.

Representation geometry tracks useful recurrent refinement. We probe the recurrent states of the two LoopREG-B/2 checkpoints on 1,024 forward-noised ImageNet images at five diffusion times. Patch CKA compares mean hidden patch features with mean DINOv2 patch features, using a fixed 256-image subset. Spatial correlation compares pairwise patch-token similarities, using 64 sampled positions across all 1,024 images. These complementary measurements describe semantic organization and spatial structure along the recurrent trajectory.

Figure 6 displays every tested depth together with generation FID. For $K_{\text{train}} = 2$, patch CKA and spatial correlation both peak at $K_{\text{infer}} = 3$, where FID is lowest. Training at $K_{\text{train}} = 3$ moves these geometry peaks to $K_{\text{infer}} = 5$, again matching the best generation depth. The extended trajectory therefore contains states with stronger teacher-referenced geometry than the state at which training ends. This correspondence connects the generation gains to refinement of the representation selected by LoopREG's supervision.

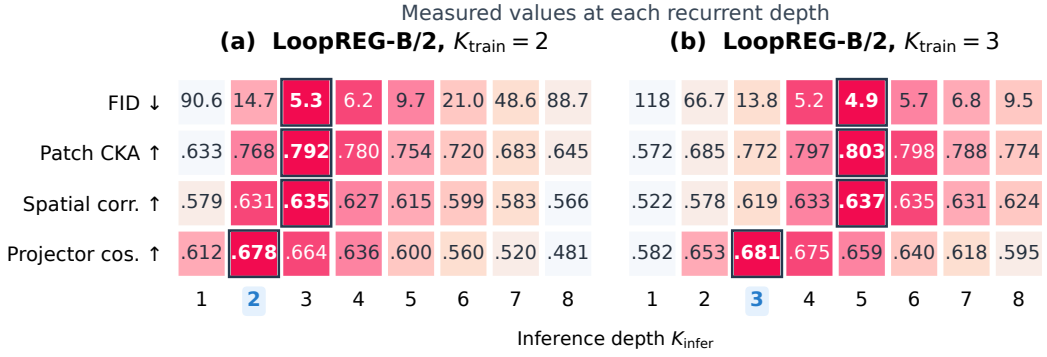

Figure 6: **Useful extra depth improves representation geometry.** Cells show measured values; darker shading indicates a better rank *within each row*, and outlines mark that row's optimum. Blue depth labels mark K_{train}. Patch CKA, spatial correlation, and FID prefer the same extra depth for each LoopREG checkpoint. Native projector cosine instead peaks at the trained depth. Representation measurements use the final N=1,024 probe (CKA subset 256), averaged over five diffusion times; FID uses the separate 50K generation evaluation.

The trained projection and extended trajectory play different roles. Native patch-projector cosine reaches its maximum at the trained depth in both checkpoints. The projector thus favors the supervised endpoint, while the best generation states occur farther along the trajectory. This distinction fits the finite-horizon interpretation in Section 3.3: a block can continue to improve task-relevant geometry after reaching the endpoint used by its training objective. Appendix G provides all five models and individual diffusion times, including the plain-recurrence control. Paired output-depth interventions in Appendix C further identify a contribution from REG's semantic-state updates.

4.4 Choosing a Depth Policy

Fixed training horizons give LoopREG its strongest peak FID. They also concentrate generation around a narrower depth range: at its best B/2 depth, the $K_{\text{train}} = 3$ model has precision 0.851 and recall 0.452, versus the REG baseline's 0.649/0.640. RandomLoop broadens the useful range and preserves more coverage. Training with $K \in \{1, 2, 3\}$ yields a best observed FID of 11.68 with recall 0.640. This offers a choice between peak fidelity and flexibility in the number of inference visits.

A timestep-dependent policy adds another operating point. Applying $K_{\text{infer}} = 3$ for $t \geq 0.6$ and $K_{\text{infer}} = 2$ for $t < 0.6$ reaches FID 7.25, between the two fixed-depth results, with recall 0.566. It reduces total block executions by 4.17% relative to fixed $K_{\text{infer}} = 3$. Appendix D reports the complete random-depth curves, compute accounting, and unshared controls.

5 Conclusion

LoopREG makes additional test-time depth useful by repeatedly applying a shared Transformer block at a representation-supervised state. It reduces SiT-B/2 FID from 15.08 to 4.91 at 400K training steps and extends recurrent-depth gains to L/2 and XL/2. The supervision-site interventions identify a transferable design choice within these models: useful recurrence follows the supervised representation. These results establish a practical way to improve diffusion generation by extending learned recurrent computation.

Scope and future directions. The experiments cover class-conditional ImageNet, SiT backbones, and DINOv2 supervision. Useful depth has a finite optimum, and improved FID often accompanies lower recall. Extending the approach to other generation settings and selecting depths without a generation sweep are natural next steps.

Reproducibility Statement

We provide the original training, sampling, and evaluation code, experiment configurations, and scripts for regenerating the quantitative figures and tables in the anonymous supplementary material. The supplement documents the required data and pretrained assets and gives commands for training from scratch and evaluation. Appendix A records the experimental settings; Appendices H and G provide final-checkpoint sweeps and representation probes. The explanatory derivations and additional diagnostics appear in Appendices B, C, and E.

AI Assistance Disclosure

Generative AI tools assisted with manuscript drafting and revision, literature retrieval, code auditing, result interpretation, development of the dynamical-systems framework and proofs, experiment planning, architecture-figure generation and editing, and implementation of data-processing, visualization, and evaluation scripts. Numerical results in this manuscript are drawn from the project's recorded experimental evidence.

References

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. Deep equilibrium models. In *Advances in Neural Information Processing Systems*, 2019. URL https://arxiv.org/abs/1909.01377.

Andrea Banino, Jan Balaguer, and Charles Blundell. Pondernet: Learning to ponder, 2021. URL https://arxiv.org/abs/2107.05407.

Lizhang Chen, Jonathan Li, Chen Liang, Ni Lao, and Qiang Liu. Training-free looped transformers, 2026. URL https://arxiv.org/abs/2605.23872.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1807.03819.

Khashayar Gatmiry, Nikunj Saunshi, Sashank J. Reddi, Stefanie Jegelka, and Sanjiv Kumar. Can looped transformers learn to implement multi-step gradient descent for in-context learning? In *International Conference on Machine Learning*, 2024. URL https://proceedings.mlr.press/v235/gatmiry24b.html.

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. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2502.05171.

Angeliki Giannou, Shashank Rajput, Jy-Yong Sohn, Kangwook Lee, Jason D. Lee, and Dimitris Papailiopoulos. Looped transformers as programmable computers. In *International Conference on Machine Learning*, 2023. URL https://proceedings.mlr.press/v202/giannou23a.html.

Sahil Goyal, Swayam Agrawal, Gautham Govind Anil, Prateek Jain, Sujoy Paul, and Aditya Kusupati. Elt: Elastic looped transformers for visual generation, 2026. URL https://arxiv.org/abs/2604.09168.

Ahmadreza Jeddi, Marco Ciccone, and Babak Taati. Loopformer: Elastic-depth looped transformers for latent reasoning via shortcut modulation. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2602.11451.

Dengyang Jiang, Mengmeng Wang, Liuzhuozheng Li, Lei Zhang, Haoyu Wang, Wei Wei, Guang Dai, Yanning Zhang, and Jingdong Wang. No other representation component is needed: Diffusion transformers can provide representation guidance by themselves. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2505.02831.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning*, 2019. URL https://arxiv.org/abs/1905.00414.

Nanye Ma, Mark Goldstein, Michael S. Albergo, Nicholas M. Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, 2024. URL https://arxiv.org/abs/2401.08740.

Maxime Oquab, Timothee Darcet, Theo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023. URL https://arxiv.org/abs/2304.07193.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE/CVF International Conference on Computer Vision*, 2023. URL https://arxiv.org/abs/2212.09748.

Ivan Viakhirev, Kirill Borodin, Amirah Almutairi, Serguei Barannikov, Maxim Abramov, and Grach Mkrtchian. Think shallow, solve deep: Controlling recurrent dynamics for reliable test-time depth, 2026. URL https://arxiv.org/abs/2608.18222.

Ge Wu, Shen Zhang, Ruijing Shi, Shanghua Gao, Zhenyuan Chen, Lei Wang, Zhaowei Chen, Hongcheng Gao, Yao Tang, Jian Yang, Ming-Ming Cheng, and Xiang Li. Representation entanglement for generation: Training diffusion transformers is much easier than you think. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2507.01467.

Liu Yang, Kangwook Lee, Robert Nowak, and Dimitris Papailiopoulos. Looped transformers are better at learning learning algorithms. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2311.12424.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.06940.

Xingyu Zhou, Qifan Li, Xiaobin Hu, Hai Chen, and Shuhang Gu. Guiding a diffusion transformer with the internal dynamics of itself. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. URL https://arxiv.org/abs/2512.24176.

A Experimental Setup and Evidence Provenance

We study class-conditional ImageNet generation at 256×256. Generation metrics and representation probes are retained in separate tables with their checkpoint, recurrent-depth, and evaluation settings. All numerical tables and inline results are generated from these records. The final 1,024-image DINO probe supplies the representation analysis; output-depth interventions and smaller-sample depth-selection diagnostics use their own evaluation protocols.

Training. SiT-B/2 has 12 blocks of width 768 with 12 heads; SiT-L/2 has 24 blocks of width 1024 with 16 heads. Image latents have four channels and spatial size 32×32, with latent patch size two. The semantic teacher is DINOv2 ViT-B/14 (Oquab et al., 2023). Training uses global batch 256, AdamW with learning rate 10^{-4}, betas $(0.9, 0.999)$, epsilon 10^{-8}, no weight decay, gradient clipping at one, class dropout 0.1, FP16 mixed precision, and TF32. The default launcher uses eight GPUs and checkpoints every 10K steps. RandomLoop's horizon draw is a deterministic function of the experiment seed and optimization step and is synchronized across ranks; equal configured weights implement a uniform distribution over the specified choices. Horizon sampling has no learned halting mechanism. Nominally equal step counts do not match training FLOPs across horizons or supervision methods.

Sampling. The sampler constructs 250 time points from 1 to 0.04 and appends zero. It applies 249 stochastic Euler–Maruyama updates and a final deterministic drift update, totaling 250 network calls at CFG=1. REG jointly evolves image and semantic-token states. Network inputs/weights are FP32 with TF32 enabled; velocities and state updates are promoted to FP64. The VAE decoder is evaluated separately. Generated images are clamped, multiplied by 255, and converted to uint8 by truncation. The evaluator reads a local ImageNet reference and the local OpenAI TensorFlow Inception graph. FID, sFID, IS, precision, and recall are imported as reported, without recomputing them from unavailable image batches.

The configured global seed is zero and the rank seed is $\mathrm{seed} \times \mathrm{world_size} + \mathrm{rank}$. Raw artifacts show different world sizes or per-GPU batches across runs; therefore these are not matched-noise paired comparisons. The source's resume branch skips the sampler for completed batches and does not advance the random-number stream through their internal SDE noise draws. This prevents a blanket claim of bit-identical resumed and uninterrupted generation. The available artifacts do not establish which final cells, if any, were affected. The paired output-depth experiments below instead use fresh full runs and record their complete random-number traces.

Late evidence and precision. The final N=1,024 mechanism bundle and unshared-x3 K3/K6 metrics supersede earlier development results or missing-data notes. The representation analysis in Appendix G uses this final bundle. The separate velocity probe remains N=256 because no larger velocity probe is supplied. The L/2 main-result curve is supplied as the experiment owner's final server-summary Markdown, at four decimal places for FID; it is parsed automatically, but its original metric/configuration JSON is not yet part of this evidence set. The verified launcher and snapshot checkpoint-status metadata are corroborating implementation/status evidence, not a replacement for those metric artifacts.

Baseline and selection scope. Local REG-B/2 is the comparison baseline; proximity to the published REG result is a sanity check, not proof of evaluator identity. No independently evaluated local non-loop REPA baseline or local non-loop L/2 baseline is inserted. Best inference depths are descriptive minima over observed sweeps. Appendix H presents full inference-depth sweeps at the final core-model checkpoints; the supplementary CSV retains all ingested depths and checkpoints, including unfavorable outcomes. No standard errors or confidence intervals are synthesized from a single seed.

Independent output-depth evaluations. Appendix C adds paired 50K evaluations at generation seeds 2 and 3 for both train-two and train-three B/2 checkpoints. GPU/batch configuration is fixed within each checkpoint study. Fresh full runs reproduce the matched and one-step-extrapolated endpoints and evaluate mixed output depths. The mixed policies require two network evaluations per state update. They are distinct records from the historical depth sweeps and do not resolve missing provenance for other historical checkpoints.

Code availability and reproduction. The anonymous code supplement contains the original model and loss implementations, training and EMA sampling scripts, the local Inception evaluator, and configurations for shared, random-horizon, unshared, and supervision-site experiments. It also includes the final N=1,024 representation records, control results, and scripts that regenerate quantitative figures and tables from these records. The accompanying guide specifies environment installation, dataset layout, public pretrained assets, and commands for training and evaluation. ImageNet data and pretrained DINOv2/VAE assets are obtained separately; trained LoopREG checkpoint binaries are not bundled, and the supplied trainers produce the checkpoints used for reruns. Portability changes and a correction to recorded inference-depth metadata are documented; the model, loss, and sampler computations are retained.

B Dynamical Interpretation and Derivations

LoopREG has two interacting iteration scales: recurrent applications evolve the hidden state within a denoiser call, while diffusion steps evolve the image and semantic states across calls. We analyze where supervision enters the first trajectory, why its useful horizon can be finite, and how its outputs affect the second trajectory. The position interventions appear in Section 4.3; the output-path interventions appear in Appendix C.

B.1 Supervision Locality

Proposition 1 (Gradient at the recurrent endpoint)**.** *Let d be the supervision depth and ℓ the loop site. For $\ell \le d$, write $T = T_{\ell \to d}$ for the intervening transformation and $\mathcal{L}_{\mathrm{rep}} = \ell_{\mathrm{rep}}(P(T(h_K)), z)$. At differentiable points,*

$$\nabla_{h_K} \mathcal{L}_{\mathrm{rep}} = J_T(h_K)^\top J_P(T(h_K))^\top \nabla_1 \ell_{\mathrm{rep}}(P(T(h_K)), z). \tag{9}$$

At $\ell = d$, $T = I$. If $\ell > d$ and the loop parameters are distinct from upstream parameters, this alignment loss has zero derivative with respect to the loop parameters.

Co-location makes the projector read the recurrent endpoint directly. Earlier recurrence receives alignment gradients through the intervening layers; later recurrence is trained by downstream denoising losses but lies outside the alignment branch. This gradient structure motivates the hypothesis that co-location favors useful extrapolation. Moving supervision and recurrence separately and jointly tests that hypothesis, rather than selecting a block from matched-depth quality alone.

B.2 Finite Useful Horizons

Proposition 2 (Task-optimal transient state)**.** *Consider $h_{k+1} = h_\infty + \lambda(h_k - h_\infty)$, with $0 < \lambda < 1$ and $h_0 \neq h_\infty$. For a task-preferred state $h_g = h_\infty + r(h_0 - h_\infty)$, $0 < r < 1$, the increments shrink geometrically, while $Q(k) = \|h_k - h_g\|^2$ has continuous minimum at*

$$k^\star_{\mathrm{cont}} = \frac{\log r}{\log \lambda}. \tag{10}$$

An integer minimizer is one of the two bracketing integers, selected by comparing Q. Beyond the crossing, Q increases toward a positive limit.

Stability and task quality can therefore prefer different stopping times. The proposition characterizes a stable scalar recurrence; the learned Transformer's finite FID optimum is an empirical observation. It motivates studying the whole recurrent trajectory rather than treating smaller state increments as a stopping criterion.

For a family satisfying $\lambda_{K_{\mathrm{train}}}^{K_{\mathrm{train}}} = c$ and a shared task-preferred progress r_g, with $0 < c < 1$ and $0 < r_g < 1$, substitution gives

$$k^\star_{\mathrm{cont}} = K_{\mathrm{train}} \frac{\log r_g}{\log c}. \tag{11}$$

The continuous optimum lies beyond training exactly when $r_g < c$; Appendix B derives the stricter condition for an extra integer application to help. Under the shared-progress assumptions, a longer

training horizon distributes latent progress over smaller updates and moves the optimum to a later depth. The observed $2 \to 3$, $3 \to 5$, and $4 \to 6$ horizons are consistent with this direction. The constants specify the assumed family; selecting real-model depths requires estimating their task dependence. The derivations and a schematic follow below; Appendix E evaluates candidate depth-selection models.

B.3 Semantic Feedback across Sampling Steps

For REG, the recurrent computation returns image and semantic velocities $(\hat{v}_K^x, \hat{v}_K^c)$. Under the linear interpolant, CFG=1, and squared diffusion amplitude $g(t)^2 = 2t$, the implemented reverse-time drifts at $t > 0$ are

$$b_K^x = x + (2-t)\hat{v}_K^x(x, c, t, y), \qquad b_K^c = c + (2-t)\hat{v}_K^c(x, c, t, y). \tag{12}$$

An Euler–Maruyama step updates each state with its corresponding drift and additive noise; the time increment Δt_j is negative.

Proposition 3 (Delayed image effect of a semantic update)**.** *Start two updates from the same (x_j, c_j) with identical noise increments. Hold the image velocity fixed and change the semantic velocity by δv_j^c. Then*

$$\delta x_{j+1} = 0, \qquad \delta c_{j+1} = \Delta t_j (2 - t_j)\delta v_j^c. \tag{13}$$

If the next image update uses the same function b^x in both trajectories and the same noise,

$$\delta x_{j+2} = \Delta t_{j+1}\big[b^x(x_{j+1}, c_{j+1} + \delta c_{j+1}, t_{j+1}, y) - b^x(x_{j+1}, c_{j+1}, t_{j+1}, y)\big]. \tag{14}$$

Changing only the final semantic output leaves the final image unchanged.

Thus an extra-depth semantic update can affect images through subsequent evaluations of the image vector field. This identifies a pathway to intervene on: evaluate two depths at the same current joint state, and choose the image and semantic outputs separately. Our directional hypothesis is that, when the image output already uses extra depth, the extra-depth semantic output further improves generation. The structure determines the intervention; the sign of its FID effect is tested across two training horizons in Appendix C.

B.4 Derivations

Proof of Proposition 1. For $s = T(h_K)$ and $a = P(s)$, the total differential is $d\mathcal{L}_{\text{rep}} = \nabla_1 \ell_{\text{rep}}(a, z)^\top J_P(s) J_T(h_K)\, dh_K$. Transposing gives Equation (9). At co-location $J_T = I$. If alignment is evaluated before the recurrent block, its computational graph has no dependency on the distinct parameters of that later block, so that partial derivative is zero. This claim presumes those parameters are not also tied to upstream modules; the audited local-block implementation satisfies this condition. Other downstream losses may still provide nonzero gradients.

For a shared core, set $J_j = \partial F_\theta(h_j)/\partial h_j$ and $A_j = \partial F_\theta(h_j)/\partial \theta$. Holding the prefix state fixed with respect to the core parameters, define adjoints $g_K = \nabla_{h_K}\mathcal{L}$ and $g_j = J_j^\top g_{j+1}$. Then

$$\nabla_\theta \mathcal{L} = \sum_{j=0}^{K-1} A_j^\top g_{j+1}. \tag{15}$$

This is the ordinary backpropagation-through-time identity. The shared operator aggregates gradients over states reached by its own trajectory. Contributions can reinforce or cancel; an unshared stack instead updates separate parameter sets.

Proof of Proposition 2. Induction gives $h_k = h_\infty + \lambda^k(h_0 - h_\infty)$. Consequently,

$$\|h_{k+1} - h_k\| = (1-\lambda)\lambda^k \|h_0 - h_\infty\|, \tag{16}$$

$$Q(k) = \|h_0 - h_\infty\|^2 (\lambda^k - r)^2. \tag{17}$$

For continuous $k \geq 0$, λ^k decreases strictly from one to zero and crosses r once, at $\log(r)/\log(\lambda)$. Before the crossing, the error strictly decreases; afterwards it strictly increases. A discrete minimizer

must be one of the bracketing integers. For $m = \lfloor k^\star_{\text{cont}} \rfloor$, choose m if $r \geq (\lambda^m + \lambda^{m+1})/2$, and $m + 1$ otherwise, with a tie at equality. Thus rounding $k^\star_{\text{cont}}$ to the nearest integer is not generally correct because the trajectory is nonlinear in depth. The limiting error is $r^2\|h_0 - h_\infty\|^2 > 0$.

The same derivation applies to a vector trajectory restricted to a single stable eigenmode when the preferred state lies on that trajectory. General multidimensional dynamics need not be unimodal, normal, or contractive. Neither shrinking observed increments nor a U-shaped FID curve establishes a spectral-radius bound for a Transformer.

Latent discretization. Taking logarithms of $\lambda_{K_{\text{train}}}^{K_{\text{train}}} = c$ yields $\log \lambda_{K_{\text{train}}} = \log c / K_{\text{train}}$. Substitution into the continuous optimum gives Equation (11). For $r_g < c$, both logarithms are negative and $\log r_g / \log c > 1$, giving a continuous optimum beyond the training horizon. The through-origin descriptive fit to the observed horizon pairs has coefficient 1.5517. This summarizes the observed horizon shift under the latent-progress model; estimating its variability requires additional training configurations.

When does one extra application improve the task error? Write $C = \|h_0 - h_\infty\|^2 > 0$ and $\lambda = \lambda_{K_{\text{train}}}$, so $\lambda^{K_{\text{train}}} = c$. Subtracting the two task errors gives

$$Q(K_{\text{train}} + 1) - Q(K_{\text{train}}) = Cc(\lambda - 1)\big[c(1 + \lambda) - 2r_g\big]. \tag{18}$$

Since $c > 0$ and $\lambda < 1$, one extra integer application improves Q if and only if

$$r_g < \frac{c(1 + \lambda)}{2}. \tag{19}$$

Equality gives a tie. This is stricter than $r_g < c$, which only places the continuous optimum beyond training. For example, $\lambda = 1/2$, $K_{\text{train}} = 2$ and $r_g = 0.24$ place the continuous optimum after two, yet the discrete optimum remains two. The criterion identifies the missing task-dependent quantity: the desired progress r_g must be estimated independently of the observed best depth before this model can select an inference horizon.

Proof of Proposition 3. For a state $u \in \{x, c\}$, the linear interpolant gives

$$\text{score}_K^u = \frac{-(1 - t)\hat{v}_K^u - u}{t}, \qquad b_K^u = \hat{v}_K^u - \tfrac{1}{2}(2t)\,\text{score}_K^u = u + (2 - t)\hat{v}_K^u. \tag{20}$$

The sampler evaluates this expression at positive times, including $t = 0.04$ for its final update to zero. Its stochastic steps are

$$x_{j+1} = x_j + \Delta t_j b^x(x_j, c_j, t_j, y) + \sqrt{2t_j|\Delta t_j|}\,\xi_j^x, \tag{21}$$

$$c_{j+1} = c_j + \Delta t_j b^c(x_j, c_j, t_j, y) + \sqrt{2t_j|\Delta t_j|}\,\xi_j^c. \tag{22}$$

Take two steps from the same current state with the same two noise tensors. Their image drifts are identical, giving $\delta x_{j+1} = 0$. Subtracting the semantic updates cancels the noise and the current state, giving Equation (13). At the next step, subtracting the image updates again cancels the common image state and noise; the difference of the image drift evaluations gives Equation (14). The final sampler call returns only the image update, so its semantic output cannot affect the returned image.

For a differentiable next-step image drift, a semantic intervention of strength α yields the local sensitivity

$$\left.\frac{dx_{j+2}}{d\alpha}\right|_{\alpha=0} = \Delta t_{j+1}\,\frac{\partial b^x}{\partial c}\,\Delta t_j(2 - t_j)\delta v_j^c. \tag{23}$$

The Jacobian is evaluated along the unperturbed trajectory. This expression identifies the feedback channel; the FID direction is an empirical property tested by the complete sampling intervention.

Teacher-forced error and generated trajectories. The velocity probe measures risk on forward-noised data, whereas the sampler visits model-induced joint image/semantic states. The terminal image depends on the directions of successive drift changes and on subsequent sensitivity to both states. Consequently, the observed reduction in FID can accompany an increase in teacher-forced NMSE. The projector is a nonlinear MLP; its local Jacobian and the downstream network can be sensitive to different hidden-state directions.

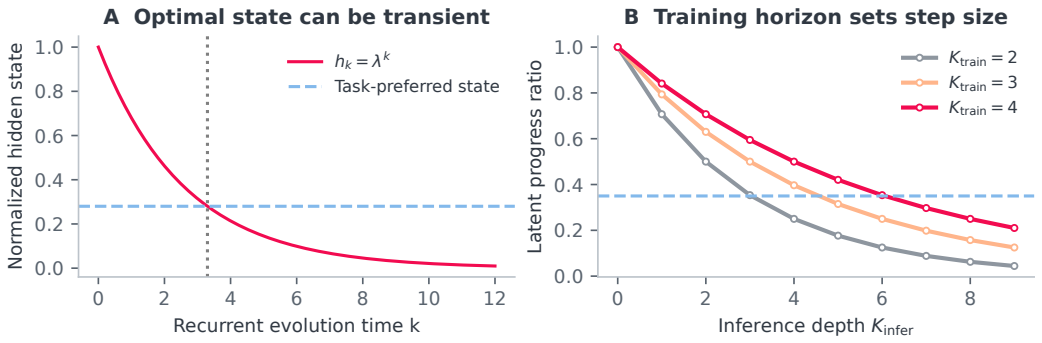

Figure 7: **Transient optima and horizon-dependent latent progress.** A contractive trajectory can pass through a task-preferred state before reaching its fixed point. Under a fixed total-progress assumption, a larger training horizon corresponds to smaller recurrent steps and a later optimal inference horizon.

C Paired Output-Depth Interventions

Experimental question and replication. We use shared Block4 SiT-B/2 REG checkpoints trained for 400K steps at $K_{\text{train}} = m \in \{2, 3\}$. Mixed policies evaluate depths m and $m + 1$ at the same current (x, c, t, y) of that condition's trajectory; equal-depth policies evaluate the network once. We select image velocity from depth K_x and semantic velocity from depth K_c. Conditions A/B/C/D use $(m, m), (m + 1, m), (m, m + 1), (m + 1, m + 1)$ throughout sampling. Each selected output updates its corresponding state, so later inputs depend on both policies.

For train-two, the four conditions and primary directional comparison $\text{FID}(B) > \text{FID}(D)$ were specified before a seed-1 screening run with 5K images per condition. Seeds 2 and 3 then evaluated all four conditions at 50K each with the checkpoint, depths, and comparison fixed. Train-three was designated as a cross-horizon replication with depths three and four before its evaluation. Its two 50K seeds use the same factorial design. All 16 final evaluations are reported; screening samples are excluded from the table and means. A/D reproduce each checkpoint's native matched and one-step-extrapolated policies. This intervention tests a fixed extra step, rather than searching for the best depth of either checkpoint.

Sampling and common random numbers. All conditions use EMA, CFG=1, the original 250-update Euler–Maruyama time grid, FP32 network evaluation with TF32, FP64 state arithmetic, and the same VAE, quantization, ImageNet reference, and Inception evaluator. Train-two uses six H100 GPUs and train-three uses four, each with batch 64. Rank seeds are $sW+r$, where s is the generation seed, W the world size, and r the rank. Within each checkpoint and seed, initial image and semantic noise, class labels, and every SDE noise increment are shared across all four conditions. Each condition is a fresh complete sampling run. Different world sizes make cross-horizon comparisons unpaired, with partially overlapping rank seeds. The two checkpoints represent different training horizons, not repeated training seeds.

Outputs, trajectories, and compute. For a mixed policy, the two network calls use identical current inputs before either state is advanced. Thus B→D keeps the image-update function at depth $m + 1$, but changes future semantic inputs to that function. A→C similarly isolates the semantic-output policy at image depth m. Changing image depth under a fixed semantic policy gives A→B and C→D. The mixed policies require 500 network calls because both depths are evaluated separately; the native joint endpoints require 250. These interventions identify output-path effects rather than provide a compute-matched alternative sampler.

Implementation checks. Routed A/D match the native sampler's image latents at their respective depths with identical inputs and random states. Mixed outputs match direct model outputs evaluated at those same inputs. Replacing only the final semantic output leaves the final image unchanged, as

Condition	K_x	K_c	FID ↓	sFID ↓	IS ↑	Prec. ↑	Rec. ↑	Calls
$K_{\text{train}} = 2$, *generation seed 2; 50,000 images per condition*								
A	2	2	14.709	6.702	97.933	0.6473	**0.6326**	250
B	3	2	7.533	6.236	142.720	0.7455	0.5599	500
C	2	3	9.362	6.097	132.561	0.7021	0.5974	500
D	3	3	**5.339**	**5.783**	**185.375**	**0.7934**	0.5143	250
$K_{\text{train}} = 2$, *generation seed 3; 50,000 images per condition*								
A	2	2	14.567	6.632	98.236	0.6482	**0.6381**	250
B	3	2	7.398	6.207	144.263	0.7472	0.5660	500
C	2	3	9.175	6.067	133.129	0.7047	0.5971	500
D	3	3	**5.296**	**5.758**	**187.057**	**0.7970**	0.5130	250
$K_{\text{train}} = 3$, *generation seed 2; 50,000 images per condition*								
A	3	3	13.924	6.552	100.534	0.6555	**0.6365**	250
B	4	3	7.438	5.480	139.488	0.7519	0.5747	500
C	3	4	9.279	6.155	132.510	0.7111	0.6005	500
D	4	4	**5.235**	**5.295**	**176.790**	**0.8011**	0.5296	250
$K_{\text{train}} = 3$, *generation seed 3; 50,000 images per condition*								
A	3	3	13.848	6.612	100.628	0.6583	**0.6339**	250
B	4	3	7.519	5.540	139.223	0.7535	0.5700	500
C	3	4	9.303	6.193	130.924	0.7167	0.5902	500
D	4	4	**5.339**	**5.351**	**175.520**	**0.8038**	0.5294	250

Table 3: All paired 50K output-depth evaluations. Bold marks the best metric within each checkpoint and generation seed. Calls counts denoiser evaluations per image: A/D use 250 and B/C use 500, all with 250 state updates.

Proposition 3 requires. Model forwards consume no random numbers, and the recorded native SDE draws preserve pairing.

FID, interaction, and coverage. Both checkpoints and both seeds give D<B<C<A in FID. For train-two, B→D improves FID by 29.1% and 28.4%; train-three gives 29.6% and 29.0%. Joint refinement improves sFID, IS, and precision while lowering recall in all four groups. The FID-scale interaction $J_{11} - J_{10} - J_{01} + J_{00}$ is positive in each group: the joint FID reduction is smaller than the sum of isolated reductions. This nonlinear metric contrast does not assign additive percentages of the gain to separate mechanisms. Means average two separate 50K metrics within each checkpoint; they are not pooled 100K FIDs or estimates of training-seed uncertainty.

D Site Interventions, Depth Policies, and Controls

D.1 Site and Supervision Depth

Table 4 supplements Figure 5 with both inference depths and recall for every tested site/depth pair.

D.2 Training and Inference Depth Policies

Training over different horizons controls the range of useful inference depths. LoopREG-x3 at $K_{\text{infer}} = 5$ has precision 0.851 and recall 0.452, compared with the REG baseline's 0.649 and 0.640. Lower FID therefore coexists with reduced measured coverage. RandomLoop broadens the region of useful depths and retains recall closer to the baseline, at a worse peak FID. For example, Random $\{1, 2, 3\}$ attains its best observed FID of 11.68 at $K_{\text{infer}} = 5$ with recall 0.640. Fixed depth favors peak FID; random depth favors a broader useful range. Some random-policy curves continue improving at the evaluated boundary.

A timestep-dependent depth schedule gives another operating point. Using $K_{\text{infer}} = 2$ for $t < 0.6$ and $K_{\text{infer}} = 3$ for $t \geq 0.6$ yields FID 7.25 and recall 0.566, between fixed $K_{\text{infer}} = 2$ and fixed $K_{\text{infer}} = 3$. Its mean recurrent depth is 2.416; relative to fixed $K_{\text{infer}} = 3$, recurrent-core executions

Sup. depth	Loop block	FID $K_{\text{infer}} = 2$	FID $K_{\text{infer}} = 3$	FID $K_{\text{infer}} = 4$	Gain (%)	R at $K_{\text{infer}} = 3$
Fixed supervision at depth four						
4	2	**13.92**	32.56	58.85	-133.93	0.676
4	4	14.71	**5.34**	6.20	63.67	0.508
4	6	**14.53**	31.11	69.58	-114.13	0.611
4	8	**14.18**	29.97	59.55	-111.33	0.618
Supervision and recurrence moved together						
2	2	15.61	**7.49**	10.25	51.99	0.544
4	4	14.71	**5.34**	6.20	63.67	0.508
6	6	14.81	**5.98**	7.78	59.61	0.528
8	8	16.54	**12.88**	28.32	22.16	0.541

Table 4: **Site/depth interventions.** Bold marks the lowest FID within each row. Gain is the relative FID reduction from $K_{\text{infer}} = 2$ to $K_{\text{infer}} = 3$; negative values indicate degradation. The last column reports recall at the extrapolated depth. The d4/B4 anchor is repeated across groups. At d8/B8, FID improves while sFID worsens (full metrics in the supplementary CSV).

decrease by 19.47%, and total block executions by only 4.17%. Neither quantity is a measured wall-clock or full-model FLOP saving.

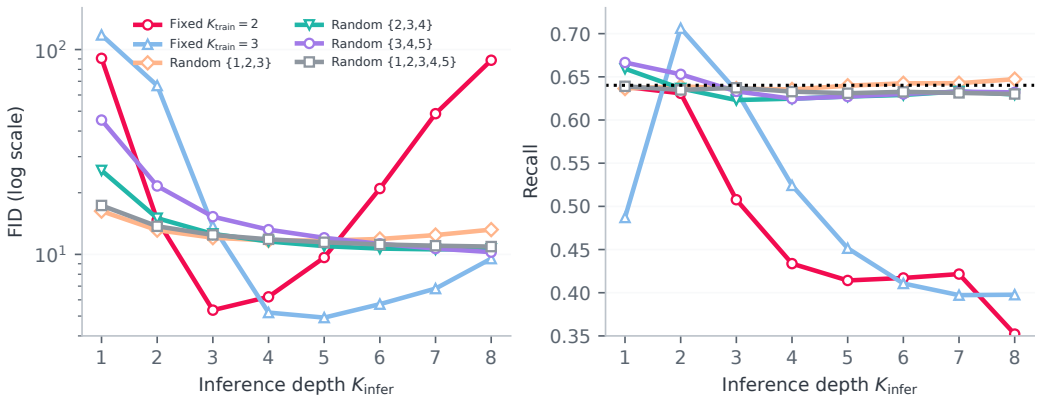

Figure 8: Fixed and random training horizons. The full observed depth range exposes under-roll failures, finite optima, and recall differences. The dotted recall line is the local REG baseline. Random $\{3, 4, 5\}$ and $\{1, 2, 3, 4, 5\}$ are still improving in FID at $K_{\text{infer}} = 8$, the evaluated boundary.

D.3 Shared and Periodic Unshared Operators

For an unshared pair, $H = F_b \circ F_a$ and $K_{\text{infer}} = 2m$ evaluates H^m; for a triplet, $H = F_c \circ F_b \circ F_a$ and $K_{\text{infer}} = 3m$. Inference cycles through the trained copies without adding weights. The matched pair attains FID 14.15, slightly better than shared train-two's 14.71. It improves to 12.27 on its second cycle before deteriorating sharply. The triplet worsens from 13.32 to 40.91 on its first extra cycle. Thus sharing does not explain a matched-depth advantage; its main benefit here is greater useful extrapolation.

D.4 Teacher-Forced Velocity Error

For LoopREG, velocity NMSE is lowest at the training horizon, whereas generation FID is lowest at a larger depth. The extra-depth benefit therefore cannot be reduced to improvement in this measured one-step teacher-forced error.

K_{train} / Policy	K_{infer}	Blocks	FID ↓	sFID ↓	IS ↑	P ↑	R ↑
2 / fixed	1	12	151.9461	50.3472	6.8845	0.1480	0.1539
2 / fixed	2	13	**14.1520**	**6.6645**	**99.3560**	**0.6541**	**0.6383**
2 / periodic	2	13	14.1520	**6.6645**	99.3560	**0.6541**	**0.6383**
2 / periodic	4	15	**12.2748**	8.2114	**137.2766**	0.6315	0.5338
2 / periodic	6	17	61.1025	19.6300	39.9223	0.2887	0.4964
2 / periodic	8	19	147.5555	52.2028	10.3376	0.0813	0.2113
3 / periodic	3	14	**13.3211**	**6.5114**	**103.6684**	**0.6622**	**0.6294**
3 / periodic	6	17	40.9129	23.2540	53.8996	0.4237	0.5638

Table 5: **Unshared controls at 400K.** Every evaluated depth is included, grouped by training horizon and inference policy. The fixed and periodic pair share their matched-depth anchor. Bold marks metric optima within each group.

Model	K_{train}	NMSE K_{infer}	FID K_{infer}	NMSE train	NMSE FID-best
LoopSiT-x2	2	2	2	0.46074	–
LoopREG-x2	2	2	3	0.45706	0.45898
LoopREG-x3	3	3	5	0.45684	0.46031

Table 6: The separate N=256 teacher-forced velocity probe, averaged over five noise times. The FID-best horizon differs from the NMSE-best horizon for both LoopREG checkpoints. "–" means that the two horizons coincide.

E Depth-Selection Diagnostics

The main method selects inference depth from measured sweeps. We additionally tested two candidate forecasting rules on the existing train-two, train-three, and train-four checkpoints. Their historical endpoint behavior was known during development; predictions were frozen before new evaluation samples were generated. These diagnostics use 5K or 4,096 images per metric, separately from the main 50K results.

Early training curves. Exponential fits $E_q(s) = E_{q,\infty} + A_q e^{-\gamma_q s}$ model FID's mean and covariance contributions from observations at 50K, 100K, and 150K steps. The fitted rate must be positive and the limit nonnegative; invalid fits abstain. The grid contains three training horizons and six inference depths, $K_{\text{infer}} \in \{2, \ldots, 7\}$. Calibration uses seed 1701; 200K and 400K evaluations use seed 2901 with paired streams and 5K samples per cell. All three rules select the lowest-FID

Forecast	MAE, 200K	MAE, 400K	Depth hits, 400K	Selected $(K_{\text{train}}, K_{\text{infer}})$
Component exponential	0.638	3.262	1/3	(3,5)
Total-FID exponential	0.652	3.299	1/3	(3,5)
Latest observation (150K)	2.237	5.258	1/3	(3,5)

Table 7: Early-curve forecasts on 5K-sample FID. MAE averages 18 candidates. Depth hits count correct inference-depth selections across three trained models.

pair $(K_{\text{train}}, K_{\text{infer}}) = (3, 5)$, but their per-model selections $4, 5, 7$ match only one of the observed optima $3, 5, 6$. Exponential fitting improves numerical forecasts without improving depth selection over the 150K ranking. At 400K, 16 of 18 measured FIDs fall below their fitted limits. This rule requires early trained checkpoints and does not select training depth before training.

Local sampling responses. The second rule interpolates both image and semantic velocities as $\hat{v}_\alpha = (1 - \alpha)\hat{v}_k + \alpha\hat{v}_{k+1}$. Complete sampling at $\alpha \in \{0, 1/16, 1/8, 1/4\}$ produces terminal Inception features after decoding and quantization. Linear and quadratic fits extrapolate these features to $\alpha = 1$, using 4,096 paired samples at calibration seed 5101. Predictions are fixed before independent-seed endpoint evaluation at seed 5201. The eight edges are $2 \to 3, 3 \to 4$ for train-two, $3 \to 4, 4 \to 5, 5 \to 6$ for train-three, and $4 \to 5, 5 \to 6, 6 \to 7$ for train-four. Endpoint evalua-

Method	Direction hits	Depth hits	Change MAE	Mean selection loss
Quadratic feature extrapolation	3/8	0/3	8550.500	8.744
Quadratic, half step	3/8	0/3	82100.946	8.744
Linear feature extrapolation	3/8	0/3	242.900	8.744
Zero change	0/8	0/3	3.560	8.744
$K_{\text{train}} + 1$	–	1/3	–	0.405
Observed calibration anchors	–	3/3	–	0.000

Table 8: Local feature extrapolation on 4,096-sample FID. Direction hits compare signs over eight edges. Selection loss is measured FID at the selected depth minus the best measured FID; depths are chosen by accumulated predicted changes, with smaller-depth tie-breaking.

tions again favor depths $3, 5, 6$. All three nonzero extrapolators predict deterioration on every edge, missing the five beneficial edges and selecting the trained depths. Their mean selection loss is 8.744 FID. The $K_{\text{infer}} = K_{\text{train}} + 1$ rule performs better; ranking observed calibration anchors recovers all three optima using its multiple-depth measurements. The tested local feature approximations therefore do not supply a predictive stopping rule.

Earlier representation diagnostics. A development score combining gradient compatibility, geometry gain, and perturbation response selects co-located depth two, whereas depth four yields the best measured FID. All four reported semantic-direction perturbation ratios exceed one. A median geometry-peak rule predicts horizons $3, 4, 6$, compared with observed $3, 5, 6$. A separate frozen REG-baseline score selects its already supervised block four while its mean extra-step geometry gains remain negative at every site. These results distinguish supervision placement and recurrent training from prediction of the optimal depth. Their development records and the conditional diagonal-Gaussian reference calculation are retained in the code supplement.

F XL Training Milestones

Table 9 summarizes the 50K, 100K, 200K, and 400K checkpoints, together with the 220K checkpoint used in the main-text comparison with published REG. Each row identifies the evaluated inference-depth set; a dash marks a checkpoint evaluated at only one depth. Gain is relative to the same checkpoint at $K_{\text{infer}} = 2$, and recall is reported at its best observed FID. Published REG

Steps	Tested K_{infer}	FID at 2	Best K_{infer}	Best FID	Gain (%)	Recall
50K	2,3,4,5	25.1962	4	**11.8154**	53.11	0.3407
100K	2,3,4,5	8.3360	3	**4.7552**	42.96	0.4682
200K	2,3,4,5	4.6383	3	**3.4942**	24.67	0.5010
220K	3	–	–	3.3675	–	0.5180
400K	1,2,3,4,5	3.3823	3	**2.8376**	16.10	0.5490

Table 9: **Representative XL training checkpoints.** Best depths are minima over the listed sets. The 220K checkpoint supplies a single-depth observation. All metrics at the final 400K checkpoint appear in Appendix H; intermediate checkpoints remain in the supplementary CSV.

results are reference values from Table 9 of Wu et al. (2025); the comparison uses no local non-loop REG-XL run. Checkpoint crossings reported in the main text refer to the earliest evaluated observations in the full record, rather than exact crossing times.

G DINO Representation Trajectories

The final probe evaluates 1,024 forward-noised real images at $t \in \{.1, .3, .5, .7, .9\}$. Unbiased linear CKA (Kornblith et al., 2019) compares mean hidden patch features with either mean DINO patch features or DINO CLS features. The launcher caps each CKA calculation at a deterministic 256-image subset; this cap is specified by the launcher rather than the saved per-model metadata. "CLS CKA" names the teacher target, not the model's semantic-token state. Spatial correlations compare

off-diagonal normalized patch Gram entries, while native cosine compares normalized projected features with DINO targets. Both use all 1,024 images and 64 sampled patch positions.

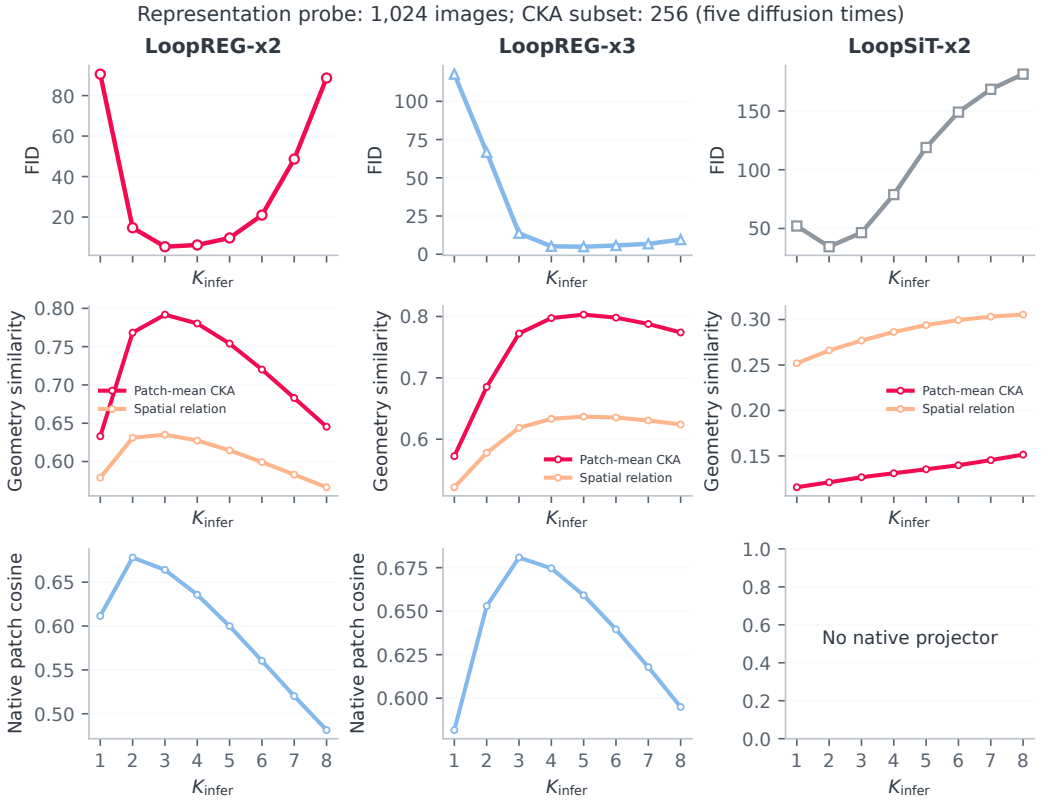

Figure 9: **Representation geometry and native alignment along recurrent depth.** FID, patch-mean CKA/spatial correlation, and native projector cosine use aligned depth axes. Geometry is averaged over five noise times from the final 1,024-image probe (CKA subset: 256). These are teacher-forced hidden trajectories; FID comes from separate full-generation evaluations. Vertical scales differ across panels.

For LoopREG trained at $K_{\text{train}} = 2$, patch-mean CKA and spatial correlation peak at $K_{\text{infer}} = 3$, matching the FID minimum; native projector cosine peaks at $K_{\text{infer}} = 2$. For $K_{\text{train}} = 3$, all three geometry metrics peak at $K_{\text{infer}} = 5$, while native cosine peaks at $K_{\text{infer}} = 3$. The correspondence depends on the metric and model: teacher-CLS CKA for train-two peaks at four, and LoopSiT's geometry similarity increases even as FID deteriorates. Relative hidden-state increments also do not establish global contraction. The supplementary CSV retains all 200 per-time and 40 averaged records across five models, including baseline hidden-state probes; those baseline probes are not generation results at additional depths.

H FINAL-CHECKPOINT GENERATION SWEEPS

The following tables give every evaluated inference depth at the 400K checkpoint for the core SiT/REPA/REG comparisons and both larger scales. This selection is by model and checkpoint, not by metric value. Site interventions, random-depth policies, and unshared controls are summarized in Appendix D. The supplementary `results_master.csv` retains all 250 historical generation observations, including intermediate checkpoints and additional controls, with scripts for regenerating their full tables.

"Blocks" counts Transformer-block executions, not exact FLOPs. Bold marks each metric's optimum within a model and checkpoint; FID and sFID are minimized, and IS, precision, and recall are maximized. Displayed values are rounded; canonical records retain source precision.

H.1 SiT-B/2: SUPERVISION AND TRAINING HORIZON

K_{infer}	Blocks	FID ↓	sFID ↓	IS ↑	P ↑	R ↑
LoopSiT-B/2, B4, $K_{\text{train}} = 2$						
1	12	52.0813	11.3411	28.9425	0.3676	**0.6608**
2	13	**34.4524**	**6.4313**	**43.0118**	**0.5334**	0.6387
3	14	46.4096	10.0098	34.2806	0.4637	0.6182
4	15	78.8033	19.1662	20.7103	0.3153	0.5667
5	16	118.9361	31.6541	12.4613	0.1959	0.4317
6	17	149.0437	42.9569	8.6622	0.1284	0.2810
7	18	168.4506	53.1676	6.6430	0.0944	0.1592
8	19	181.3879	63.7429	5.4062	0.0754	0.0928
LoopREPA-B/2, d=4, B4, $K_{\text{train}} = 2$						
1	12	88.8098	20.6534	14.7986	0.2660	0.6509
2	13	21.9274	**6.3655**	66.5780	0.6013	**0.6515**
3	14	**16.7427**	6.6802	**84.6709**	**0.6285**	0.6147
4	15	28.3614	9.0526	61.8826	0.4946	0.6118
5	16	59.4920	14.8863	30.6378	0.2966	0.5953
6	17	101.3782	27.0815	15.0184	0.1545	0.4585
7	18	134.6486	45.3181	9.2247	0.0925	0.2664
8	19	158.8770	64.4720	6.6918	0.0646	0.1125
LoopREG-B/2, d=4, B4, $K_{\text{train}} = 2$						
1	12	90.5972	16.9459	14.6801	0.2443	**0.6385**
2	13	14.7094	6.7629	97.1320	0.6496	0.6309
3	14	**5.3434**	**5.8021**	185.9796	0.7958	0.5077
4	15	6.2050	6.6180	**207.7940**	**0.8078**	0.4337
5	16	9.6552	9.4030	174.6141	0.7287	0.4142
6	17	20.9611	15.3694	108.8510	0.5695	0.4171
7	18	48.6352	25.9636	50.0054	0.3735	0.4216
8	19	88.6875	40.5806	22.4223	0.2294	0.3521
LoopREG-B/2, d=4, B4, $K_{\text{train}} = 3$						
1	12	117.7844	33.5290	10.0884	0.1792	0.4871
2	13	66.7103	16.5472	23.7143	0.3315	**0.7063**
3	14	13.7500	6.6929	100.8156	0.6578	0.6337
4	15	5.2002	**5.4001**	178.3913	0.8023	0.5243
5	16	**4.9136**	5.4032	217.6060	0.8506	0.4515
6	17	5.7158	5.9812	**224.6076**	**0.8513**	0.4107
7	18	6.8081	7.1051	209.2107	0.8195	0.3972
8	19	9.5347	8.9836	173.4586	0.7508	0.3977

Continued on the next page

K_{infer}	Blocks	FID ↓	sFID ↓	IS ↑	P ↑	R ↑
LoopREG-B/2, d=4, B4, $K_{\text{train}} = 4$						
1	12	121.6970	35.4991	9.3515	0.1827	0.4151
2	13	93.8612	23.3516	14.4801	0.2425	0.6437
3	14	44.1952	11.7995	39.1530	0.4310	**0.7128**
4	15	13.2435	6.5524	102.6734	0.6605	0.6335
5	16	5.8749	5.3505	161.3391	0.7849	0.5518
6	17	**5.1874**	**5.2995**	193.9240	0.8394	0.4845
7	18	5.8128	5.6755	**206.5647**	**0.8531**	0.4439
8	19	6.6785	6.2577	204.0657	0.8496	0.4178

H.2 SiT-L/2 and SiT-XL/2

K_{infer}	Blocks	FID ↓	sFID ↓	IS ↑	P ↑	R ↑
LoopREG-L/2, d=8, B8, $K_{\text{train}} = 2$						
1	24	59.4139	12.6041	26.0546	0.3214	**0.7664**
2	25	4.2286	5.0924	170.3622	0.7516	0.6420
3	26	**3.2082**	**4.4353**	262.6577	0.8593	0.5201
4	27	5.4971	4.5898	**291.8665**	**0.8827**	0.4438
5	28	7.0317	4.9216	267.8294	0.8450	0.4206
LoopREG-XL/2, d=8, B8, $K_{\text{train}} = 2$						
1	28	52.4630	10.6201	31.5308	0.3699	**0.7668**
2	29	3.3823	4.8470	184.7094	0.7584	0.6347
3	30	**2.8376**	**4.6353**	**254.6991**	**0.8386**	0.5490
4	31	3.7734	5.1102	239.4408	0.8179	0.5240
5	32	7.8614	7.0541	164.5297	0.7154	0.5457