AN END-TO-END LATENT-ROLLOUT APPROACH FOR PUSHING FEW-STEP IMAGENET-256 GENERATION TO FID 1.11 WITHOUT FRÉCHET LOSSES

Anonymous authors
Paper under double-blind review

ABSTRACT

Iterative generation poses a joint optimization problem across steps, as intermediate predictions shape subsequent computations and ultimately determine the final output distribution. Few-step generators distilled from pretrained diffusion and flow-matching models make such optimization computationally practical end to end. We build on this opportunity with a distill-then-refine approach that uses teacher imitation to establish a strong initialization for a few-step rollout in latent space, then shifts to end-to-end refinement of the complete latent rollout against real data. We introduce FiST (Flow-in-Stage Transformer), an architecture that composes learned latent-state transitions in a few stages using a shared Transformer, with optional cross-stage hidden communication. Distillation applies teacher-forced regression to selected states along teacher trajectories; refinement replaces this supervision with adversarial and auxiliary classification objectives on the final latent output. A trainable discriminator module operates on semantically rich features extracted from clean real and generated latents by a frozen SiT backbone pretrained with REPA. All training takes place in latent space, without image decoding. During refinement, FiST consumes its own intermediate predictions, and endpoint gradients pass through every generation stage. For class-conditional generation on ImageNet at 256×256, our approach achieves FID 1.11 (IS 282) with three stages and FID 1.15 (IS 280) with two. These results demonstrate competitive few-step generation through learned distribution-level supervision, without explicit Fréchet-distance minimization. Ablations characterize how distillation, pretrained checkpoint choices, refinement supervision, and cross-stage hidden communication affect generation quality.

1 INTRODUCTION

Diffusion and flow-matching models achieve high-quality image synthesis through iterative sampling, but repeated network evaluations make generation computationally expensive (Ho et al., 2020; Lipman et al., 2023). Distillation accelerates sampling by transferring the behavior of these pretrained models into few-step generators (Salimans & Ho, 2022). Beyond sampling acceleration, distillation can provide a strong initialization that facilitates stable distribution-level optimization against real data through an adversarial objective (Lin et al., 2025a). Treating the complete few-step rollout as a differentiable generator allows its intermediate computations to be optimized jointly through supervision on the final output. Our central question is *whether end-to-end refinement of a distilled few-step rollout against real data can close the gap to many-step generation.*

We introduce **FiST (Flow-in-Stage Transformer)**, an architecture that factorizes the noise-to-data mapping into a composition of learned latent-state transitions across a small number of stages. A shared Transformer implements these transitions, mapping the current latent state to the next one at each stage; the final stage produces the generator's latent output. The basic rollout passes latent states between stages, while an augmented rollout additionally carries hidden features through a cross-stage hidden router. We use a *distill-then-refine* procedure to train both rollout forms entirely in latent space, separating distillation-based initialization from subsequent end-to-end refinement. Distillation applies teacher-forced regression to selected states along teacher trajectories, including intermediate states and the final latent output.

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.

The refinement phase replaces the regression supervision with adversarial and auxiliary classification objectives driven by real data. We feed latents of real images and the generator's latent outputs into a frozen SiT backbone pretrained with REPA (Ma et al., 2024; Yu et al., 2025). This backbone provides a semantically rich feature space shared by real and generated samples, while a trainable discriminator module uses two output heads to assess sample realism and class consistency, respectively. The objectives therefore supervise the final latent output, leaving intermediate stage states free to adapt during refinement. The discriminator processes clean latents directly, avoiding image decoding during optimization. By learning to distinguish real from generated samples, the discriminator provides adaptive supervision for distribution-level refinement without explicit Fréchet-distance minimization or dependence on a specific evaluation metric.

During refinement, FiST is executed from noise using its own intermediate predictions as inputs to subsequent stages, matching its inference computation. For each generator update, we retain the full gradient path from the endpoint objectives through the discriminator and every generation stage, including the hidden router in the augmented rollout. The discriminator's frozen feature backbone is differentiated with respect to the generated latent. The shared Transformer therefore receives gradient contributions from its use at every stage, each accounting for how that stage's prediction affects the final output through the remaining stages.

We evaluate our approach on ImageNet 256×256 for class-conditional image generation. FiST achieves **FID 1.11** (IS 282) with three stages and **FID 1.15** (IS 280) with two. Both achieve lower reported FID than two-step FACM (1.32) and one-step W-Flow (1.29) (Peng et al., 2026; Han et al., 2026), and approach the reported FID of leading many-step models such as GenFirst (0.97) (Zheng et al., 2026) and SFD (1.04) (Pan et al., 2025). Ablations examine distillation duration, teacher classifier-free guidance, auxiliary classification, cross-stage hidden communication, pretrained checkpoints, and stage count. Training curves of FID and IS further characterize the refinement dynamics. These studies show that longer distillation and REPA-based discriminator features improve FID significantly after refinement, with additional gains from auxiliary classification and cross-stage hidden communication. Teacher guidance also affects the FID–IS tradeoff.

Our contributions are summarized as follows:

- We introduce FiST, a Transformer architecture for staged latent rollouts that realizes the noise-to-data mapping through a short sequence of learned latent-state transitions, with a basic rollout and an augmented variant supporting cross-stage hidden communication.
- We develop an effective distill-then-refine procedure that initializes FiST by regressing to selected teacher-trajectory states under teacher forcing, then refines the complete latent rollout end to end using real-data adversarial and auxiliary classification supervision in a frozen REPA feature space.
- We evaluate our end-to-end latent-rollout approach for class-conditional image generation on ImageNet at 256×256, demonstrating competitive two- and three-stage performance and characterizing the effects of architecture, distillation, pretrained checkpoints, and refinement supervision.

2 Related Work

Distillation trains few-step generators using supervision from pretrained diffusion and flow-matching models. Progressive distillation repeatedly halves the number of steps in a deterministic diffusion sampler (Salimans & Ho, 2022). Consistency models support one- and few-step generation and can be trained either by distillation from pretrained models or directly from data (Song et al., 2023). Adversarial objectives further connect fast generation with distribution-level supervision. ADD combines diffusion-teacher supervision with an adversarial loss for few-step generation (Sauer et al., 2024b). CTM combines a consistency distillation loss with denoising score matching and adversarial losses (Kim et al., 2024). DMD introduces score-based distribution matching (Yin et al., 2024b), and DMD2 augments it with real-data adversarial training (Yin et al., 2024a). APT separates consistency-based initialization from subsequent adversarial post-training of a one-step generator (Lin et al., 2025a), while D2O directly fine-tunes a diffusion-pretrained generator with an adversarial objective (Zheng & Yang, 2025). We train FiST with a distill-then-refine procedure: teacher-forced regression to selected teacher-trajectory states initializes the generator, and subsequent end-to-end refinement replaces regression with real-data adversarial and auxiliary classification objectives.

Pretrained representations provide a basis for distribution-level supervision. Projected GANs train discriminator heads on fixed image features (Sauer et al., 2021). LADD performs adversarial distillation in latent space, using frozen diffusion-teacher features extracted from re-noised generated latents (Sauer et al., 2024a). Representation alignment offers an externally guided way to shape the features used in generative modeling: REPA aligns diffusion Transformer representations with pretrained visual features (Yu et al., 2025), and GAT aligns a trainable Transformer discriminator with frozen DINOv2 features (Hyun et al., 2026). W-Flow trains one-step generators through Sinkhorn-derived distribution updates in pretrained feature spaces (Han et al., 2026). FD-loss directly minimizes Fréchet discrepancies in pretrained feature spaces (Yang et al., 2026), while AdvFD augments fixed-representation matching with an adversarially learned representation (Gao et al., 2026). Our refinement procedure uses a frozen SiT backbone pretrained with REPA as a latent feature extractor, with trainable adversarial and auxiliary classification heads but no Fréchet-distance losses. The backbone processes clean real and generated latents directly, avoiding image decoding.

Optimizing a multi-step generator also depends on which parts of the sampling process remain in the gradient graph. DRaFT differentiates a reward through an entire diffusion sampling chain, while DRaFT-K restricts backpropagation to the last few denoising steps (Clark et al., 2024). In few-step distillation, DMD2 trains its multi-step generator on its own intermediate outputs, simulating inference (Yin et al., 2024a), and TDM matches teacher and student trajectory distributions while restricting backpropagation to one ODE step (Luo et al., 2025). During refinement, we optimize FiST across its complete staged rollout using endpoint adversarial and auxiliary classification losses. Gradients pass through the discriminator's frozen feature backbone and every generation stage of the generator, making earlier stages responsive to their effect on the final output.

3 Approach

3.1 FiST Architecture

Let $z = E(x)$ be the latent encoding of an image x produced by a pretrained VAE encoder E, and let c denote its class label. FiST factorizes the noise-to-data mapping into a composition of learned latent-state transitions implemented by a shared Transformer F_θ, with each forward pass constituting a *generation stage*. For Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$ and a fixed stage count K, the basic rollout is:

$$z_0 = \epsilon, \qquad z_s = F_\theta(z_{s-1}; e_s, c), \qquad s = 1, \dots, K, \tag{1}$$

where e_s is a learned embedding for stage s and z_K is the final latent output. We construct F_θ from a pretrained SiT (Ma et al., 2024) by replacing its timestep embedder with a stage embedding table. We also reinitialize its final prediction layer, as FiST predicts the next-stage state rather than velocity. By default, FiST reuses backbone weights from a SiT checkpoint pretrained with REPA (Yu et al., 2025).

The basic rollout passes latent states between stages. An augmented rollout additionally retains layer-wise hidden features for subsequent stages through a cross-stage hidden router. Let H_s collect the hidden features retained from stage s, and let $\mathcal{H}_{<s} = \{H_j\}_{j<s}$ be the memory available to that stage. With router parameters ϕ and $\Theta = (\theta, \phi)$, the augmented computation is

$$(z_s, H_s) = F_\Theta(z_{s-1}, \mathcal{H}_{<s}; e_s, c), \qquad \mathcal{H}_{<1} = \varnothing. \tag{2}$$

Executing either rollout for K stages from $z_0 = \epsilon$ defines the complete differentiable generator $G_\Theta(\epsilon, c) = z_K$, where $\Theta = \theta$ for the basic rollout and $\Theta = (\theta, \phi)$ for the augmented rollout. We develop a lightweight implementation of the router with negligible overhead in both parameter count and computation. At each Transformer layer, attention operates along the stage axis at corresponding patch positions, with the current stage providing the query and the current and earlier stages providing the keys. Because attention spans only a few stages, we use a single head with low-dimensional queries and keys. Earlier-stage hidden features serve as values after learned channel-wise gating, avoiding a costly value projection. The resulting message is added residually to the current hidden features. Appendix A provides further details on parameter initialization, routing, and overhead.

3.2 Distillation-Based Initialization

For each noise–label pair (ϵ, c), a frozen teacher generates a latent trajectory from noise to a latent endpoint. We simply divide the sampling time interval into K equal parts and select the corresponding

trajectory states $\{\bar{z}_s\}_{s=0}^{K}$, with $\bar{z}_0 = \epsilon$ and $\bar{z}_K$ the final latent. FiST is then trained through teacher-forced regression to these selected states:

$$(\hat{z}_s, H_s) = F_\Theta(\bar{z}_{s-1}, \mathcal{H}_{<s}; e_s, c), \qquad s = 1, \dots, K,$$
$$\mathcal{L}_{\text{dist}}(\Theta) = \mathbb{E}_{\epsilon,c}\left[\sum_{s=1}^{K} \text{MSE}(\hat{z}_s, \bar{z}_s)\right]. \tag{3}$$

The loss supervises FiST's predictions of the selected intermediate states and the final latent output. Each stage receives the preceding teacher state $\bar{z}_{s-1}$ as its latent input; its prediction $\hat{z}_s$ is regressed to $\bar{z}_s$ and does not become the next stage's input. In the augmented rollout, $\mathcal{H}_{<s}$ contains FiST-computed hidden features from earlier teacher-forced stages, preserving differentiable cross-stage hidden communication; the basic rollout omits the hidden-memory inputs and outputs. End-to-end refinement of the complete latent rollout starts from the distilled FiST checkpoint. Appendix B provides implementation details and the distillation algorithm.

3.3 End-to-End Refinement

Starting from the distilled checkpoint, we execute FiST from noise using its own intermediate predictions as inputs to subsequent stages. The rollout matches the inference computation and produces the final latents $z^- = G_\Theta(\epsilon, c)$, while real image latents $z^+ = E(x)$ serve as samples from the target distribution. Refinement replaces teacher-forced regression with adversarial and auxiliary classification objectives on this latent endpoint. The teacher and its selected trajectory states are no longer used.

Latent-space discriminator. We extract features from both real image latents z^+ and generated endpoints z^- using a frozen SiT backbone pretrained with REPA (Yu et al., 2025). The backbone processes clean latents at $t = 0$, providing semantically rich multi-layer hidden features. Let $\Phi(z, \tilde{c})$ collect the feature maps from selected layers, where $\tilde{c}$ equals c or the unconditional label according to the conditioning dropout used in backbone pretraining. This exposes the discriminator's trainable module to both conditional and unconditional features during training, with both objectives using the same features from a single backbone pass. The module $\mathcal{D}_\omega$ consists of a convolutional aggregation network followed by two output heads that produce an adversarial logit d_ω and a vector of class logits q_ω:

$$\big(d_\omega(z, \tilde{c}), q_\omega(z, \tilde{c})\big) = \mathcal{D}_\omega\big(\Phi(z, \tilde{c})\big). \tag{4}$$

Here ω denotes all trainable parameters, while the backbone parameters remain frozen. The original class label c remains the classification target. Appendix C details the selected feature layers, conditioning dropout, and trainable module architecture.

Refinement objectives. We use a logistic adversarial objective with a non-saturating generator loss (Goodfellow et al., 2014), together with auxiliary classification supervision (Odena et al., 2017). Write $\text{sp}(a) = \log(1 + \exp(a))$ for softplus, sg for stop-gradient, and CE for cross-entropy over class logits. Expectations include the randomness of conditioning dropout. For the discriminator update, we minimize

$$\mathcal{L}_{\text{D}}(\omega; \Theta) = \mathbb{E}_{z^+,c,\tilde{c}}\big[\text{sp}\big(-d_\omega(z^+, \tilde{c})\big) + \lambda_{\text{cls}}^{\text{D}}\,\text{CE}\big(q_\omega(z^+, \tilde{c}), c\big)\big] + \mathbb{E}_{\epsilon,c,\tilde{c}}\big[\text{sp}\big(d_\omega(\text{sg}[z^-], \tilde{c})\big)\big]. \tag{5}$$

The adversarial term trains the discriminator to distinguish real from generated samples, while the classification term is applied only to real samples. For the generator update, both objectives act on the final latent output:

$$\mathcal{L}_{\text{G}}(\Theta; \omega) = \mathbb{E}_{\epsilon,c,\tilde{c}}\big[\text{sp}\big(-d_\omega(G_\Theta(\epsilon, c), \tilde{c})\big) + \lambda_{\text{cls}}^{\text{G}}\,\text{CE}\big(q_\omega(G_\Theta(\epsilon, c), \tilde{c}), c\big)\big]. \tag{6}$$

The two loss terms encourage sample realism and consistency with the requested class, respectively. We use $\lambda_{\text{cls}}^{\text{D}} = 0.7$ and $\lambda_{\text{cls}}^{\text{G}} = 0.5$; Appendix C explains how the loss weights are set to balance initial gradient magnitudes. These losses require neither paired real targets for generated samples nor regression to teacher outputs. In other words, given a noise–label pair (ϵ, c), the generated endpoint is not required to reproduce a particular real image or teacher prediction. Instead, the two objectives supervise the output distribution through a discriminator trained on real and generated samples.

End-to-end updates. Each refinement iteration takes one discriminator update followed by one generator update. During the generator update, ω is held fixed and we retain the full gradient path through $\mathcal{D}_\omega$, the frozen feature backbone Φ, and every generation stage. Let $\ell_{\mathrm{G}}(z, c, \tilde{c}; \omega)$ denote the per-sample loss inside the expectation in Eq. (6), evaluated at generated latent z. The generator gradient is

$$\nabla_\Theta \mathcal{L}_{\mathrm{G}} = \mathbb{E}_{\epsilon,c,\tilde{c}}\left[\left(\frac{\partial G_\Theta(\epsilon, c)}{\partial \Theta}\right)^{\mathsf{T}} \nabla_z \ell_{\mathrm{G}}(z, c, \tilde{c}; \omega)\big|_{z=G_\Theta(\epsilon,c)}\right]. \tag{7}$$

Computing the endpoint gradient requires differentiating Φ with respect to its latent input while keeping the backbone parameters fixed. The rollout Jacobian accounts for all intermediate latent-state dependencies and, in the augmented rollout, cross-stage hidden-memory dependencies.

To make each stage's contribution to the shared-parameter gradient explicit, consider the basic rollout and write $F_\theta^s(z) = F_\theta(z; e_s, c)$. Let $J_s = \partial F_\theta^s / \partial z_{s-1}$ be its input Jacobian and $A_s = \partial F_\theta^s / \partial \theta$ its parameter Jacobian with the stage input held fixed, both evaluated along the current rollout. For one sampled endpoint loss, the state gradients v_s and parameter gradient satisfy

$$v_K = \nabla_{z_K} \ell_{\mathrm{G}}, \qquad v_{s-1} = J_s^{\mathsf{T}} v_s, \quad s = K, \ldots, 1, \qquad \nabla_\theta \ell_{\mathrm{G}} = \sum_{s=1}^{K} A_s^{\mathsf{T}} v_s. \tag{8}$$

The shared Transformer therefore receives a contribution from every stage invocation, with each v_s accounting for how that stage's prediction influences the final output through the remaining stages. Later stages take the generator's own predictions as inputs, and end-to-end updates jointly optimize the computations that produce and process these predictions. For the augmented rollout, the same recursion applies after including the hidden memory in the stage state and using the full parameter set Θ. This additionally accounts for the effect of earlier hidden features on all later stages that access them through the router.

With supervision applied only at the endpoint, intermediate states can depart from teacher states as the complete latent rollout is optimized for final sample quality. All refinement objectives are evaluated in latent space, without image decoding or Fréchet-distance losses. Appendix C provides the refinement algorithm and details gradient propagation through both rollout forms.

4 Experiments

4.1 Experimental Setup

We study class-conditional image generation on ImageNet at 256×256 resolution. Images are center-cropped and resized, then encoded using the pretrained SD-VAE (Rombach et al., 2022); no additional data augmentation is applied. Our default configuration uses the officially released SiT-XL/2 checkpoint trained with REPA (Yu et al., 2025) as the teacher and as the source of pretrained weights for FiST and the discriminator backbone.

FiST is trained for 40K distillation steps followed by 20K refinement iterations, with a batch size of 1024 throughout. Distillation uses a learning rate of 10^{-4}. Refinement alternates one discriminator update with one generator update, using respective learning rates of 4×10^{-5} and 10^{-5} following the two-time-scale update rule (TTUR) (Heusel et al., 2017). Both training phases use AdamW and cosine learning-rate decay with a 1K-step warmup. Detailed hyperparameters are provided in Appendix D.

Evaluation follows the ADM protocol (Dhariwal & Nichol, 2021) using 50K generated images with randomly sampled class labels. We report FID, IS, sFID, precision, and recall at the lowest-FID FiST checkpoint within 20K refinement iterations. FiST uses neither inference-time classifier-free guidance nor an exponential moving average of generator parameters. We report FiST's stage count, parameter count, and inference FLOPs in Table 1.

4.2 Comparison with Existing Methods

Model size and inference cost. Table 1 summarizes FiST's model size and inference cost. Sharing Transformer weights across stages allows longer rollouts with little change in parameter count,

Table 1: **Model size and inference cost of FiST.** Augmented FiST includes the cross-stage hidden router. Parameter counts and FLOPs are approximate. Inference costs exclude VAE decoding. For reference, SiT-XL/2 has approximately 675M parameters and costs 118.7 GFLOPs per evaluation. SiT-equivalent compute per stage is the average per-stage cost relative to one such evaluation.

Type	Stages	Parameters (M)	GFLOPs per image	Avg. GFLOPs per stage	SiT-equivalent compute per stage
Basic	3	673	356.1	118.7	$1.00\times$
Augmented	2	677	239.8	119.9	$1.01\times$
Augmented	3	677	359.7	119.9	$1.01\times$

Table 2: **System-level comparison on class-conditional ImageNet** 256×256**.** Baseline results follow the cited works and official releases. NFE denotes nominal network evaluations; Stg. denotes FiST stages (see Table 1 for costs). SD: SD-VAE; E2E: VAE trained jointly with the generator; Sem: SD-VAE + SemVAE; RAE: representation autoencoder. SIM: SigLIP + Inception + MAE.

Method	Space	NFE / Stg.	FID↓	IS↑	sFID↓	Prec.↑	Rec.↑
Many-step generation without Fréchet losses							
DiT-XL/2 (Peebles & Xie, 2023)	SD	250×2	2.27	278.2	4.60	0.83	0.57
SiT-XL/2 (Ma et al., 2024)	SD	250×2	2.06	277.5	4.49	0.83	0.59
SiT-XL/2 + REPA (Yu et al., 2025)	SD	250×2	1.42	305.7	4.70	0.80	0.65
SiT-XL/1 + REPA-E (Leng et al., 2025)	E2E	250×2	1.12	302.9	4.09	0.79	0.66
RAE + DiT^{DH}-XL (Zheng et al., 2025)	RAE	50×2	1.13	262.6	–	0.78	0.67
UCGM-S (REPA-E) (Sun et al., 2025)	E2E	40×2	1.06	–	–	–	–
SFD-XXL (Euler) (Pan et al., 2025)	Sem	200×2	1.04	–	–	–	–
GenFirst (EiT) (Zheng et al., 2026)	E2E	250×2	0.97	283.4	–	0.79	0.67
One- and few-step generation without Fréchet losses							
MeanFlow-XL/2 (Geng et al., 2025a;b)	SD	1	3.43	247.5	–	–	–
iMF-XL/2 (Geng et al., 2025b)	SD	1	1.72	282.0	–	–	–
iMF-XL/2 (Geng et al., 2025b)	SD	2	1.54	–	–	–	–
AFM-XL/2 (Lin et al., 2025b)	SD	2	2.11	273.8	4.33	0.82	0.55
AFM-XL/2, 112 layers (Lin et al., 2025b)	SD	1	1.94	292.2	4.54	0.79	0.56
Drifting-L/2 (Deng et al., 2026)	SD	1	1.54	258.9	–	–	–
W-Flow-XL/2 (Han et al., 2026)	SD	1	1.29	265.4	–	–	–
UCGM-T (REPA-E) (Sun et al., 2025)	E2E	2	1.39	–	–	–	–
FACM-XL (Peng et al., 2026)	VA	2	1.32	292.0	–	–	–
FreeFlow-XL/2 (Tong et al., 2026)	SD	1	1.45	–	–	–	–
One-step generation with Fréchet losses							
iMF-XL/2 + FD (Inception) (Yang et al., 2026)	SD	1	0.72	295.0	–	0.76	0.68
iMF-XL/2 + FD (SIM) (Yang et al., 2026)	SD	1	0.76	301.3	–	0.77	0.67
JiT-H + AdvFD (Gao et al., 2026)	Pixel	1	0.72	–	–	–	–
FiST: distill-then-refine without Fréchet losses							
FiST (basic)	SD	3	1.16	282.6	4.19	0.76	0.68
FiST (augmented)	SD	2	1.15	280.1	4.18	0.76	0.69
FiST (augmented)	SD	3	**1.11**	282.0	4.17	0.76	0.69

while computation grows approximately linearly with stage count. For the three-stage configuration, cross-stage hidden communication adds approximately 4M parameters and 3.6 GFLOPs per image, increasing model size by about 0.6% and computation by about 1.0%. An augmented FiST stage therefore costs approximately one SiT-XL/2 evaluation, allowing the stage counts in Table 2 to be interpreted in terms of SiT-equivalent compute.

Quantitative comparison. Table 2 compares basic and augmented FiST with existing image generators. The three-stage basic rollout achieves FID 1.16 with IS 282.6. The augmented rollout achieves FID 1.15 with IS 280.1 using two stages and FID 1.11 with IS 282.0 using three. All three configurations obtain lower reported FID than the listed one- and few-step methods trained without



Figure 1: **Uncurated samples from three-stage augmented FiST on ImageNet** 256×256**.** Samples are generated from randomly drawn Gaussian noise and class labels.

Fréchet losses, including W-Flow, UCGM-T and FACM. These results are also comparable to those of many-step models such as REPA-E (FID 1.12) and RAE (FID 1.13), which use 250×2 and 50×2 network evaluations, respectively. Several many-step models and methods trained with Fréchet losses report lower FID. Nevertheless, FiST demonstrates that competitive FID can be achieved with a short latent rollout through adversarial distribution learning and auxiliary classification, without explicit Fréchet-distance minimization. Our discriminator adapts its supervision to discrepancies between real and generated samples, offering a flexible route to distribution-level optimization that does not depend on a differentiable formulation of the evaluation metric.

The additional metrics provide a broader view of generation quality. FiST's sFID scores of 4.17–4.19 are lower than those of the listed DiT, SiT, REPA, and AFM configurations, but slightly higher than REPA-E's 4.09. All three FiST configurations achieve precision 0.76, with recall 0.68 for the basic rollout and 0.69 for both augmented configurations. Although adversarial training can suffer from mode collapse, the augmented FiST configurations achieve the highest reported recall among the listed methods. This suggests that distillation-based initialization followed by end-to-end adversarial refinement can attain low FID together with broad distribution coverage. The lower precision relative to several baselines nevertheless reflects a remaining tradeoff between coverage and fidelity.

Table 3: **Effects of distillation duration and teacher guidance.** Results use three-stage augmented FiST. Teacher guidance uses REPA's best-FID and qualitative-sampling scales, 1.8 and 4.0, with guidance intervals $[0, 0.7]$ and $[0, 1]$, respectively (Yu et al., 2025). Arrows indicate before $\rightarrow$ after refinement; metrics are measured at the lowest-FID checkpoint within 20K refinement iterations.

Distillation duration (CFG 1.8, $[0, 0.7]$)

Steps	FID↓	IS↑
4K	60.58 → 1.69	35.7 → 239.7
10K	8.76 → 1.19	172.1 → 271.7
40K	5.23 → 1.11	205.7 → 282.0

Teacher guidance (40K distillation)

CFG	Interval	FID↓	IS↑
1.8	$[0, 0.7]$	5.23 → 1.11	205.7 → 282.0
4.0	$[0, 1]$	10.41 → 1.23	448.8 → 302.5

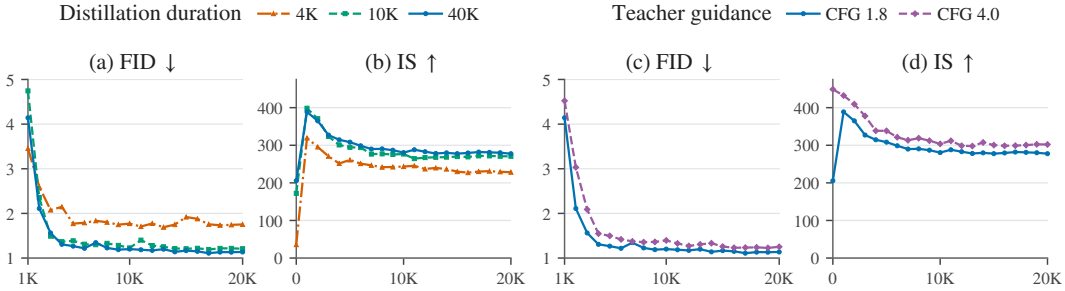

Figure 2: **Refinement dynamics under different distillation and teacher guidance settings.** Experimental settings match Table 3; horizontal axes show refinement iterations. FID curves begin at 1K to prevent large pre-refinement values from obscuring later differences.

Qualitative results. Figure 1 shows uncurated samples spanning animals, everyday objects, vehicles, food, and scenes, with rich texture detail and a natural photographic appearance. Most samples depict recognizable objects with coherent structures and plausible scene layouts. This visual quality is obtained from REPA-pretrained weights with 40K distillation steps and 20K refinement iterations, using ImageNet as the sole image dataset for refinement.

4.3 Distillation and Refinement Dynamics

We study how distillation duration and teacher guidance affect refinement of three-stage augmented FiST. Table 3 summarizes outcomes, while Figure 2 tracks their evolution during refinement.

Distillation duration. We compare 4K, 10K, and 40K distillation steps under a common learning-rate schedule, each followed by the same 20K-iteration refinement procedure. Refinement improves all three checkpoints, reducing FID from 60.58 to a best of 1.69 even after only 4K distillation steps. Longer distillation lowers the best FID to 1.19 at 10K and 1.11 at 40K, with diminishing gains. The FID gaps persist late in refinement, suggesting that the preceding distillation still affects quality under a fixed refinement budget.

Teacher guidance. With 40K distillation steps, we compare teacher CFG 1.8 over $[0, 0.7]$ against CFG 4.0 over $[0, 1]$. Stronger guidance yields higher IS but worse FID before and after refinement: its best-FID checkpoint reaches FID 1.23 with IS 302.5, versus FID 1.11 and IS 282.0 for CFG 1.8. Teacher guidance thus affects refinement even after teacher supervision ends. For CFG 1.8, Figure 2 shows an early IS peak followed by a decline while FID generally improves. IS therefore peaks well before FID reaches its minimum. The early IS rise may reflect auxiliary classification strengthening class-discriminative cues; see Section 4.4 for analysis.

4.4 Component Ablations

Auxiliary classification. Adding auxiliary classification improves the best FID from 1.22 to 1.11 and the corresponding IS from 277.0 to 282.0 (Table 4). The training curves reveal different early

Table 4: **Effects of auxiliary classification and pretrained checkpoints.** Results use three-stage augmented FiST. Left: FID and IS at the lowest-FID checkpoint within 20K refinement iterations. Right: FID with different pretrained weights for distillation (teacher and FiST initialization) and the discriminator's frozen backbone. SiT and REPA denote vanilla and REPA-trained SiT-XL/2 checkpoints, respectively. †Refinement does not improve the distilled generator's FID.

Auxiliary classification

Refinement objectives	FID↓	IS↑
Adversarial only	1.22	277.0
Adversarial + classification	1.11	282.0

Pretrained checkpoints

Distillation	Discriminator	FID↓
SiT	SiT	$> 8^{\dagger}$
SiT	REPA	1.21
REPA	REPA	1.11

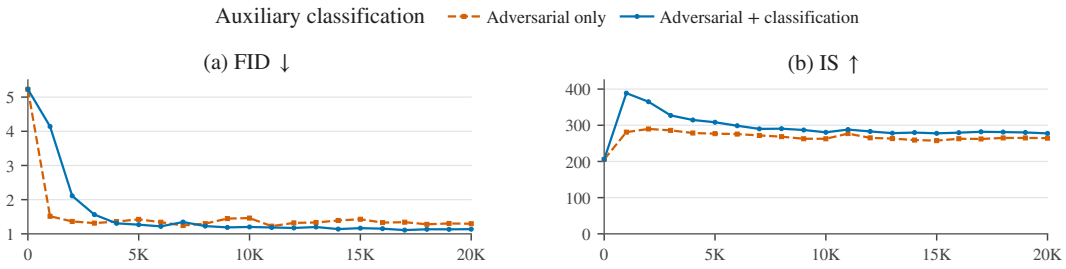

Figure 3: **Auxiliary classification during refinement.** Horizontal axes show refinement iterations; iteration 0 denotes the distilled checkpoint before refinement.

behavior: adversarial-only refinement reduces FID faster, whereas classification produces a larger initial IS rise (Figure 3). Later in refinement, classification yields both lower FID and higher IS, indicating that its benefit extends beyond the transient IS increase.

Pretrained checkpoints. Table 4 compares vanilla SiT and REPA checkpoints. With a vanilla SiT checkpoint used for both distillation (as the teacher and to initialize FiST) and the discriminator's frozen backbone, subsequent refinement does not improve the distilled generator's FID, which stays above 8. Pairing a SiT checkpoint for distillation with a REPA checkpoint for the discriminator yields FID 1.21, while using REPA checkpoints for both yields 1.11. These results highlight the importance of the discriminator's pretrained features, with a further gain from REPA-based distillation.

Cross-stage hidden communication. At three stages, the augmented rollout improves FID from 1.16 to 1.11 over the basic rollout (Table 2). This modest gain requires approximately 1% additional compute per stage (Table 1), making hidden communication an inexpensive architectural addition.

Stage count. We also experiment with one- and four-stage FiST, obtaining FID 1.47 and 1.12, respectively. The higher FID with one stage supports the benefit of multiple learned transitions, while the fourth stage adds computation without improving on the three-stage result. Together with the results above, these findings support two- and three-stage configurations as effective choices for balancing generation quality and inference cost.

5 Conclusion

We presented FiST and a distill-then-refine approach for optimizing few-step image generation end to end in latent space. Teacher imitation establishes staged latent rollouts, after which adversarial and auxiliary classification objectives refine them against real data. On class-conditional ImageNet 256×256, FiST achieves FID 1.15 with two stages and 1.11 with three, without image decoding during training or explicit Fréchet-distance minimization. Ablations highlight the importance of distillation and pretrained checkpoints, with further gains from auxiliary classification and cross-stage hidden communication. These results support short differentiable rollouts initialized by distillation as a practical basis for end-to-end optimization through adaptive distribution-level supervision.

AI USE STATEMENT

In this work, we used generative AI tools to help refine mathematical notation and clarify methodological descriptions, and to review the wording and evidential support of claims about experimental results. These tools also assisted with literature search, manuscript organization and revision, language editing, reference formatting, and code for plotting experimental results. We reviewed and revised the AI-assisted content, checking technical descriptions and mathematical expressions against the underlying method, reported results against the experimental logs, and references against their sources. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies efficient class-conditional image generation using ImageNet and publicly available pretrained models. The resulting generator may inherit biases from the dataset and pretrained representations, and its outputs could be misused to create misleading visual content. Our experiments evaluate generation quality and efficiency but do not assess fairness, privacy leakage, or safeguards against misuse. These considerations should be addressed before downstream deployment, and generated content should be clearly identified as synthetic. Use and distribution of the underlying datasets and models should respect their applicable licenses and usage terms.

REPRODUCIBILITY STATEMENT

Appendix A details FiST's architecture and initialization, while Appendix B specifies teacher-trajectory construction, state selection, and distillation. Appendix C describes the discriminator implementation. Algorithms 1 and 2 summarize the two training procedures. Section 4.1 documents data preprocessing, pretrained model sources, training schedules, and evaluation protocols, with detailed hyperparameters provided in Appendix D.

REFERENCES

Kevin Clark, Paul Vicol, Kevin Swersky, and David J. Fleet. Directly fine-tuning diffusion models on differentiable rewards. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2309.17400.

Mingyang Deng, He Li, Tianhong Li, Yilun Du, and Kaiming He. Generative modeling via drifting, 2026. URL https://arxiv.org/abs/2602.04770.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems*, volume 34, pp. 8780–8794, 2021. URL https://arxiv.org/abs/2105.05233.

Mingju Gao, Jingkai Zhou, Kun Gai, Changqian Yu, and Hao Tang. AdvFD: Boosting visual generation via adversarial Fréchet distance loss, 2026. URL https://arxiv.org/abs/2608.11205.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling, 2025a. URL https://arxiv.org/abs/2505.13447.

Zhengyang Geng, Yiyang Lu, Zongze Wu, Eli Shechtman, J. Zico Kolter, and Kaiming He. Improved mean flows: On the challenges of fastforward generative models, 2025b. URL https://arxiv.org/abs/2512.02012.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, 2014. URL https://arxiv.org/abs/1406.2661.

Jiaqi Han, Puheng Li, Qiushan Guo, Renyuan Xu, Stefano Ermon, and Emmanuel J. Candès. One-step generative modeling via Wasserstein gradient flows, 2026. URL https://arxiv.org/abs/2605.11755.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://arxiv.org/abs/1706.08500.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022. URL https://arxiv.org/abs/2207.12598.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://arxiv.org/abs/2006.11239.

Sangeek Hyun, MinKyu Lee, and Jae-Pil Heo. Scalable GANs with transformers. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2509.24935.

Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ODE trajectory of diffusion. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.02279.

Xingjian Leng, Jaskirat Singh, Yunzhong Hou, Zhenchang Xing, Saining Xie, and Liang Zheng. REPA-E: Unlocking VAE for end-to-end tuning with latent diffusion transformers, 2025. URL https://arxiv.org/abs/2504.10483.

Shanchuan Lin, Xin Xia, Yuxi Ren, Ceyuan Yang, Xuefeng Xiao, and Lu Jiang. Diffusion adversarial post-training for one-step video generation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, 2025a. URL https://arxiv.org/abs/2501.08316.

Shanchuan Lin, Ceyuan Yang, Zhijie Lin, Hao Chen, and Haoqi Fan. Adversarial flow models, 2025b. URL https://arxiv.org/abs/2511.22475.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.02747.

Yihong Luo, Tianyang Hu, Jiacheng Sun, Yujun Cai, and Jing Tang. Learning few-step diffusion models by trajectory distribution matching, 2025. URL https://arxiv.org/abs/2503.06674.

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 *Computer Vision – ECCV 2024*, 2024. URL https://arxiv.org/abs/2401.08740.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *The Sixth International Conference on Learning Representations*, 2018. URL https://arxiv.org/abs/1802.05957.

Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier GANs. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2642–2651, 2017. URL https://proceedings.mlr.press/v70/odena17a.html.

Yueming Pan, Ruoyu Feng, Qi Dai, Yuqi Wang, Wenfeng Lin, Mingyu Guo, Chong Luo, and Nanning Zheng. Semantics lead the way: Harmonizing semantic and texture modeling with asynchronous latent diffusion, 2025. URL https://arxiv.org/abs/2512.04926.

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

Yansong Peng, Kai Zhu, Yu Liu, Pingyu Wu, Hebei Li, Xiaoyan Sun, and Feng Wu. FACM: Flow-anchored consistency models. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2507.03738.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022. URL https://arxiv.org/abs/2112.10752.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *The Tenth International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2202.00512.

Axel Sauer, Kashyap Chitta, Jens Müller, and Andreas Geiger. Projected GANs converge faster. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://arxiv.org/abs/2111.01007.

Axel Sauer, Frederic Boesel, Tim Dockhorn, Andreas Blattmann, Patrick Esser, and Robin Rombach. Fast high-resolution image synthesis with latent adversarial diffusion distillation, 2024a. URL https://arxiv.org/abs/2403.12015.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *Proceedings of the European Conference on Computer Vision*, 2024b. URL https://arxiv.org/abs/2311.17042.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, 2023. URL https://arxiv.org/abs/2303.01469.

Peng Sun, Yi Jiang, and Tao Lin. Unified continuous generative models, 2025. URL https://arxiv.org/abs/2505.07447.

Shangyuan Tong, Nanye Ma, Saining Xie, and Tommi Jaakkola. Flow map distillation without data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 33973–33984, 2026. URL https://arxiv.org/abs/2511.19428.

Jiawei Yang, Zhengyang Geng, Xuan Ju, Yonglong Tian, and Yue Wang. Representation Fréchet loss for visual generation, 2026. URL https://arxiv.org/abs/2604.28190.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Frédo Durand, and William T. Freeman. Improved distribution matching distillation for fast image synthesis. In *Advances in Neural Information Processing Systems*, volume 37, 2024a. URL https://arxiv.org/abs/2405.14867.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Frédo Durand, William T. Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6613–6623, 2024b. URL https://arxiv.org/abs/2311.18828.

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 *The Thirteenth International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.06940.

Bowen Zheng and Tianming Yang. Revisiting diffusion models: From generative pre-training to one-step generation. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. URL https://arxiv.org/abs/2506.09376.

Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders, 2025. URL https://arxiv.org/abs/2510.11690.

Guangting Zheng, Yiyuan Zhang, Tao Yang, Yunpeng Chen, Rui Zhu, Jiajun Deng, and Yanyong Zhang. GenFirst: Generation before reconstruction for stable end-to-end latent generative modeling, 2026. URL https://arxiv.org/abs/2608.29335.

A FiST Architecture Details

Parameter initialization. FiST retains SiT's pretrained patch embedding, class-conditioning pathway, and Transformer blocks. The timestep embedder is replaced by a learned stage embedding table $E_{\text{stage}} \in \mathbb{R}^{K\times D}$, where D is the Transformer hidden width; the selected embedding $e_s = E_{\text{stage}}[s]$ enters the existing conditioning pathway. The stage embeddings and final prediction layer are initialized from scratch, as are the router parameters in the augmented rollout.

Cross-stage hidden router. The router is designed to add cross-stage information flow at negligible overhead. Let $h_{s,\ell,n} \in \mathbb{R}^D$ denote the hidden feature at stage s, layer ℓ, and patch position n. For $s \geq 2$, the router applies single-head causal attention at each Transformer layer, operating across stages at corresponding patch positions. Spatial interactions between patch positions remain part of the Transformer's within-stage computation. The query and key projections are affine, with output width $r = 64$; no normalization is applied before or after these projections. Their weights $W_\ell^Q, W_\ell^K \in \mathbb{R}^{r\times D}$ and biases $b_\ell^Q, b_\ell^K \in \mathbb{R}^r$ are shared across stages and patch positions. The projected queries and keys determine scaled dot-product attention weights over the current and earlier stages:

$$\begin{aligned} q_{s,\ell,n} &= W_\ell^Q h_{s,\ell,n} + b_\ell^Q, \qquad k_{j,\ell,n} = W_\ell^K h_{j,\ell,n} + b_\ell^K, \\ \alpha_{s\leftarrow j,\ell,n} &= \frac{\exp\left(q_{s,\ell,n}^\mathsf{T} k_{j,\ell,n}/\sqrt{r}\right)}{\sum_{i=1}^{s}\exp\left(q_{s,\ell,n}^\mathsf{T} k_{i,\ell,n}/\sqrt{r}\right)}, \quad j = 1,\dots,s. \end{aligned} \tag{9}$$

A learned channel-wise gate $g_{s\leftarrow j,\ell} \in \mathbb{R}^D$ is associated with each directed stage pair and layer, and shared across patch positions. The router aggregates gated hidden features from earlier stages and adds the resulting message residually:

$$\begin{aligned} m_{s,\ell,n} &= \sum_{j=1}^{s-1} \alpha_{s\leftarrow j,\ell,n}\left(g_{s\leftarrow j,\ell} \odot h_{j,\ell,n}\right), \\ \widetilde{h}_{s,\ell,n} &= h_{s,\ell,n} + m_{s,\ell,n}, \end{aligned} \tag{10}$$

where $\odot$ denotes element-wise multiplication. The routed features $\widetilde{h}_{s,\ell,n}$ are passed to Transformer block ℓ, while H_s retains the features $h_{s,\ell,n}$ before routing for use by later stages. The current-stage key participates in the softmax, but only earlier stages contribute values; the remaining weights are not renormalized. No value or output projection is used, and the first stage receives no message.

Parameter and computational overhead. For L Transformer layers and $K \geq 2$ stages, the router parameter count is

$$P_{\text{router}} = L\left[2r(D+1) + D\frac{K(K-1)}{2}\right], \tag{11}$$

where the first term counts the query and key projection weights and biases, and the second counts the channel-wise gates. With $L = 28$, $D = 1152$, and $r = 64$, the router adds 4.16M and 4.23M parameters for two and three stages, respectively, approximately 0.6% of the 675M parameters in SiT-XL/2. The added computation is dominated by the width-r projections; attention and gated aggregation operate only along the short stage axis at each patch position.

B Distillation Details

Teacher sampling. The frozen teacher is a SiT-XL/2 model pretrained with REPA (Yu et al., 2025), whose checkpoint also provides the pretrained weights for FiST's backbone. Each trajectory is generated with $N = 20$ Heun integration steps on a uniform time grid, using classifier-free guidance (Ho & Salimans, 2022). Guidance takes the form $v_\varnothing + w(v_c - v_\varnothing)$, where v_c and $v_\varnothing$ denote conditional and unconditional velocity predictions. Following the guidance settings used for REPA's best reported FID, the teacher sampler uses $w = 1.8$ for $t \in [0, 0.7]$ and $w = 1$ elsewhere.

Algorithm 1 Distillation-Based Initialization of FiST

Require: Stage count K, frozen teacher, initial FiST parameters Θ

for each distillation iteration **do**
  Sample a minibatch of noise–label pairs (ϵ, c).
  Generate teacher trajectories and select $\{\bar{z}_s\}_{s=0}^{K}$ as detached targets.
  Set $\mathcal{H}_{<1} = \varnothing$ and $\mathcal{L} = 0$.
  for $s = 1, \ldots, K$ **do**
    $(\hat{z}_s, H_s) \leftarrow F_\Theta(\bar{z}_{s-1}, \mathcal{H}_{<s}; e_s, c)$.
    $\mathcal{L} \leftarrow \mathcal{L} + \mathrm{MSE}(\hat{z}_s, \bar{z}_s)$.
    $\mathcal{H}_{<s+1} \leftarrow \mathcal{H}_{<s} \cup \{H_s\}$.
  end for
  Update Θ using $\nabla_\Theta \mathcal{L}$.
end for
return Θ

State selection. The selected states $\{\bar{z}_s\}_{s=0}^{K}$ correspond to $K+1$ evenly spaced times spanning the teacher's sampling interval $[0, 1]$. When a selection time falls between two sampled times, its state is obtained by linear interpolation between the corresponding teacher states. The initial noise $\bar{z}_0 = \epsilon$ and final teacher endpoint $\bar{z}_K$ are retained exactly.

Loss reduction and optimization. Algorithm 1 summarizes distillation using batched operations. Each MSE averages over the minibatch and latent coordinates; stage losses are summed before a single optimizer update. For the augmented rollout, hidden features remain attached to the computation graph, allowing later losses to backpropagate through earlier stages. The basic rollout omits the hidden-memory inputs, outputs, and updates.

C Refinement Details

Discriminator implementation. The frozen SiT-XL/2 backbone is evaluated through layer 16, with conditioning dropout probability 0.1, matching its pretraining setting. Feature extraction uses each backbone's native clean-data time input: $t = 0$ for REPA-pretrained SiT and $t = 1$ for vanilla SiT. Let $\Phi_\ell(z, \tilde{c})$ denote a feature map reshaped from the token sequence at selected layer $\ell \in \{4, 8, 12, 16\}$, with 1152 channels and spatial resolution 16×16. Layer-specific 1×1 projections reduce these maps to 512 channels after stacking:

$$u_0 = \sigma(\mathrm{GN}_0(\mathrm{Proj}_0([\Phi_\ell(z, \tilde{c})]_\ell))), \tag{12}$$

where $[\cdot]_\ell$ denotes stacking along the selected-layer axis, σ denotes SiLU, and GroupNorm is applied separately to each feature map throughout the trainable module.

Three residual blocks combine spatial processing within each map with mixing across the selected layers:

$$\begin{aligned} \tilde{u}_b &= \sigma\big(\mathrm{GN}_{b,1}\big(\mathrm{Conv}_{b,1}(u_b)\big)\big), \\ \Delta u_b &= \mathrm{Expand}_b\big(\mathrm{Mix}_b\big(\sigma\big(\mathrm{GN}_{b,\mathrm{mix}}\big(\mathrm{Squeeze}_b(\tilde{u}_b)\big)\big)\big)\big), \\ u_{b+1} &= \sigma\big(\mathrm{Skip}_b(u_b) + \mathrm{GN}_{b,2}\big(\mathrm{Conv}_{b,2}(\tilde{u}_b + \Delta u_b)\big)\big), \quad b = 0, 1, 2. \end{aligned} \tag{13}$$

Here $\mathrm{Conv}_{b,1}$ and $\mathrm{Conv}_{b,2}$ are layer-specific 3×3 convolutions, implemented as grouped convolutions with one group per selected layer. $\mathrm{Squeeze}_b$ and Expand_b are layer-specific 1×1 projections that reduce the channel width to one quarter and restore it, respectively. Mix_b is a dense 1×1 projection over the joint layer–channel axis at each spatial position. Skip_b is the identity when dimensions match; otherwise, it applies average pooling and a layer-specific 1×1 projection. The three blocks have overall spatial strides $(1, 2, 2)$ and output channel widths $(512, 768, 1024)$, yielding spatial resolutions $(16 \times 16, 8 \times 8, 4 \times 4)$, respectively.

The layer axis is retained until final pooling. A layer-specific 1×1 projection, GroupNorm, and SiLU precede averaging over the layer and spatial axes:

$$\begin{aligned} f &= \mathrm{Mean}_{\ell,h,w}\big[\sigma\big(\mathrm{GN}_{\mathrm{out}}\big(\mathrm{Proj}_{\mathrm{out}}(u_3)\big)\big)\big], \\ \big(d_\omega(z, \tilde{c}), q_\omega(z, \tilde{c})\big) &= \big(\mathrm{Head}_{\mathrm{adv}}(f), \mathrm{Head}_{\mathrm{cls}}(f)\big). \end{aligned} \tag{14}$$

Algorithm 2 End-to-End Refinement of FiST

Require: Distilled generator G_Θ, frozen encoder E, frozen feature extractor Φ, trainable discriminator module $\mathcal{D}_\omega$

1: **for** each refinement iteration **do**
2: Sample images x with labels c, encode $z^+ = E(x)$, and sample Gaussian noise ϵ.
3: Generate detached endpoints $z^- = \mathrm{sg}[G_\Theta(\epsilon, c)]$.
4: Compute $\mathcal{L}_\mathrm{D}$ using Eq. (5) and update ω using $\nabla_\omega \mathcal{L}_\mathrm{D}$; hold Θ fixed.
5: Sample new Gaussian noise ϵ, retaining the labels c.
6: Generate endpoints $z^- = G_\Theta(\epsilon, c)$ with gradient tracking enabled.
7: Compute $\mathcal{L}_\mathrm{G}$ using Eq. (6) and update Θ using $\nabla_\Theta \mathcal{L}_\mathrm{G}$; hold the updated ω fixed.
8: **end for**
9: **return** Θ

The two linear heads produce one adversarial logit and 1000 class logits. Spectral normalization (Miyato et al., 2018) is applied to all trainable convolutional projections.

Loss reduction and weights. Each loss term is a batch mean; the discriminator adversarial loss sums the real and generated means. For equally sized minibatches, near-zero adversarial logits, and near-uniform class probabilities over $C = 1000$ classes, balancing initial gradient norms at the head outputs gives

$$\lambda_\mathrm{cls}^\mathrm{D} \approx \frac{1}{\sqrt{2(1-1/C)}} \approx 0.7, \qquad \lambda_\mathrm{cls}^\mathrm{G} \approx \frac{1}{2\sqrt{1-1/C}} \approx 0.5. \tag{15}$$

At initialization, the per-sample adversarial and classification gradient norms at the head outputs are approximately $1/2$ and $\sqrt{1-1/C}$, respectively. The discriminator includes separate real and generated adversarial logits, giving a combined gradient norm $\sqrt{2}$ times that of the generator. The rounded classification weights remain fixed throughout training.

Alternating updates. Algorithm 2 implements the updates using Eqs. (5) and (6).

Gradient propagation with hidden memory. For the augmented rollout, combine the latent state and hidden memory into

$$\xi_s = (z_s, \mathcal{H}_{<s+1}), \qquad \xi_s = \widetilde{F}_\Theta^s(\xi_{s-1}; c), \qquad \xi_0 = (\epsilon, \varnothing), \tag{16}$$

where $\widetilde{F}_\Theta^s$ applies F_Θ at stage s and appends the returned hidden features to memory. For one sampled endpoint loss, let $v_s = \nabla_{\xi_s} \ell_\mathrm{G}$. The gradient recursion becomes

$$\begin{aligned} v_K &= (\nabla_{z_K} \ell_\mathrm{G}, 0), \\ v_{s-1} &= \left(\frac{\partial \widetilde{F}_\Theta^s(\xi_{s-1}; c)}{\partial \xi_{s-1}} \right)^\mathsf{T} v_s, \qquad s = K, \dots, 1, \\ \nabla_\Theta \ell_\mathrm{G} &= \sum_{s=1}^{K} \left(\frac{\partial \widetilde{F}_\Theta^s(\xi_{s-1}; c)}{\partial \Theta} \right)^\mathsf{T} v_s. \end{aligned} \tag{17}$$

The parameter derivatives hold the stage input ξ_{s-1} fixed. The terminal memory gradient is zero because the loss acts directly only on z_K; earlier hidden features receive gradients through their influence on subsequent stages. This accounts for gradient propagation through both latent-state transitions and hidden features passed through the cross-stage router. Omitting the memory recovers Eq. (8).

D Experimental Setup Details

Pretrained checkpoints. We use the EMA version (`sd-vae-ft-ema`) of the publicly available SD-VAE. The REPA checkpoint is the officially released SiT-XL/2 model trained for 4M iterations (Yu et al., 2025). The vanilla SiT-XL/2 checkpoint used in the pretrained-checkpoint ablations comes from the official SiT release (Ma et al., 2024).

Table 5: **Training hyperparameters.** Each refinement iteration consists of one discriminator update followed by one generator update. G and D denote the generator and discriminator, respectively.

Setting	Distillation	Refinement
Training duration	40K steps	20K iterations
Batch size	1024	1024
Learning rate	10^{-4}	10^{-5} (G), 4×10^{-5} (D)
Optimizer	AdamW	AdamW
AdamW betas	$(0.9, 0.99)$	$(0.9, 0.99)$
Weight decay	0	0
Learning-rate schedule	Cosine decay	Cosine decay
Warmup steps	1K	1K
Minimum learning-rate ratio	0.1	0.1
Maximum gradient norm	1.0	1.0

Training configuration. Table 5 summarizes the training hyperparameters. Warmup and cosine decay are applied separately in the two training phases. The minimum learning rate is 0.1 times the corresponding base learning rate. Settings varied in individual ablations are specified alongside their results. Teacher sampling and state selection are described in Appendix B, while discriminator implementation and loss-weight calibration are detailed in Appendix C.

E ADDITIONAL QUALITATIVE SAMPLES

We present uncurated samples from 20 selected ImageNet classes. All samples are generated by three-stage augmented FiST at the checkpoint with FID 1.11 and IS 282.0.



Figure 4: **Additional uncurated samples.** Class 246: Great Dane.



Figure 5: **Additional uncurated samples.** Class 287: lynx.



Figure 6: **Additional uncurated samples.** Class 353: gazelle.



Figure 7: **Additional uncurated samples.** Class 388: giant panda.



Figure 8: **Additional uncurated samples.** Class 393: anemone fish.



Figure 9: **Additional uncurated samples.** Class 407: ambulance.



Figure 10: **Additional uncurated samples.** Class 425: barn.



Figure 11: **Additional uncurated samples.** Class 483: castle.



Figure 12: **Additional uncurated samples.** Class 497: church.



Figure 13: **Additional uncurated samples.** Class 586: half track.



Figure 14: **Additional uncurated samples.** Class 625: lifeboat.



Figure 15: **Additional uncurated samples.** Class 630: Loafer.



Figure 16: **Additional uncurated samples.** Class 717: pickup.



Figure 17: **Additional uncurated samples.** Class 793: shower cap.



Figure 18: **Additional uncurated samples.** Class 871: trimaran.



Figure 19: **Additional uncurated samples.** Class 873: triumphal arch.



Figure 20: **Additional uncurated samples.** Class 888: viaduct.



Figure 21: **Additional uncurated samples.** Class 903: wig.



Figure 22: **Additional uncurated samples.** Class 949: strawberry.



Figure 23: **Additional uncurated samples.** Class 958: hay.