CrossFlow: One-Step Generation Across Latent and Pixel Spaces

Xiyuan Wang[1], Xiao Zhang[2], Yang Li[2], Ruoxi Jiang[3], Zhao Zhong[2], Liefeng Bo[2], Muhan Zhang[1]*
[1]Institute for Artificial Intelligence, Peking University [2]Tencent [3]Fudan University
wangxiyuan@pku.edu.cn muhan@pku.edu.cn

Abstract

Most diffusion and flow-matching generators define the prior, probability path, and prediction target in the same representation space. Latent diffusion improves efficiency by moving this path into an autoencoder latent space, but the final sample is still produced by a separately trained decoder. This separation creates a mismatch: the generator is optimized for latent-space prediction, while final quality depends on how the decoder handles generated latents that may differ from clean encoder outputs. We introduce CrossFlow, a cross-space flow formulation that maps noisy latent inputs directly to pixel-space images. The key technical step is a velocity-free one-step objective: the latent trajectory defines the training path, but the supervised prediction is an image rather than a latent displacement. This lets one model act both as a one-step latent-to-pixel generator and as a decoder replacement for latent diffusion pipelines. On class-conditional ImageNet-1k at 256×256, CrossFlow-XL achieves 1.62 FID with one function evaluation. Ablations show that the latent encoder and pixel-space perceptual and adversarial losses are important for fidelity. These results indicate that cross-space flow objectives can combine the efficiency of latent representations with direct pixel-space supervision, without requiring a separate decoder at inference. Code is available at https://github.com/Xi-yuanWang/crossflow.

1 Introduction

Latent diffusion has become a dominant framework for image generation. Diffusion models [1; 2] and flow-matching methods [3; 4] provide training objectives, and Transformer backbones [5; 6] have made these objectives scalable. Large latent diffusion systems such as Stable Diffusion 3 [7], FLUX [8], and SANA [9] reduce the computation by running the generative trajectory in the compressed latent space of an autoencoder. This design has become standard because it avoids the cost of denoising directly in high-dimensional pixel space.

The same design also leaves an unresolved interface problem. In a latent diffusion model, the generator is trained to predict latents, while the decoder is trained to reconstruct images from latents produced by an encoder. However, at inference time, the decoder receives generated latents rather than clean encoder outputs. If these generated latents are distribution-shifted, the decoder can amplify errors. The generator also cannot be trained directly with image-level perceptual or adversarial losses unless its output is first passed through the separate decoder. Thus, latent diffusion gains efficiency by decoupling generation from decoding, but this decoupling weakens direct pixel-level supervision.

This paper asks a simple question: must the noise space and the output space be the same? A useful image generator could start from a latent prior, preserve the computational advantages of latent representations, and still predict the image itself. Such a model would remove the explicit decoder

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

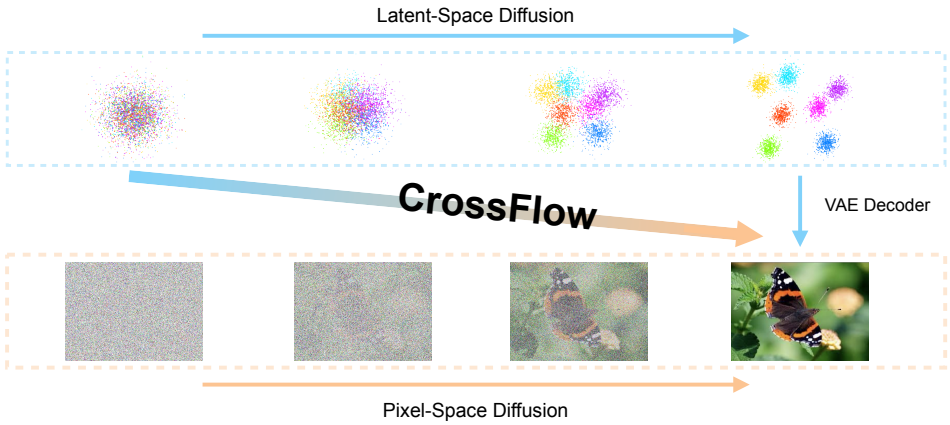


Figure 1: **CrossFlow generation paradigm.** Latent diffusion uses a two-stage pipeline: iterative denoising in latent space followed by VAE decoding. Pixel-space diffusion performs iterative denoising directly in the image domain. CrossFlow uses a single model to map a noisy latent prior directly to a pixel-space image, unifying one-step generation and latent-to-pixel decoding.

stage, train directly on generated images, and expose the model during training to the kind of noisy or imperfect latents that appear during generation.

We propose *CrossFlow*, a cross-space one-step generation framework. As shown in Figure 1, CrossFlow maps a noisy latent input directly to a pixel-space image in a single function evaluation. This setting is different from standard flow matching, where the model predicts a velocity to update the current state, both of which must live in the same space. A latent-to-pixel model cannot update a latent variable with a pixel-space velocity. CrossFlow resolves this mismatch by deriving objectives in which the latent-space velocity is eliminated from the supervised target. The latent path still defines how clean latents are corrupted, but the network output is pixel.

We instantiate CrossFlow with a fixed pretrained image encoder and a Vision Transformer generator. During training, an image is encoded into a latent, corrupted along a latent trajectory, and passed to the generator, which directly predicts pixels. Because the output is already an image, we can use reconstruction, perceptual, and adversarial losses in the same training objective. The resulting model has two uses: it is a one-step latent-to-pixel generator, and it can replace the decoder in a latent diffusion pipeline whose latents are produced by a generator rather than by a clean encoder. On ImageNet-1k at 256×256, CrossFlow-B, CrossFlow-L, and CrossFlow-XL achieve 2.54, 1.69, and 1.62 FID, respectively, with one function evaluation. CrossFlow-XL is competitive with strong multi-step latent diffusion baselines while operating in a different regime: it maps latent noise directly to pixels and does not use a separate decoder at inference. Ablations show that both the latent representation and pixel-space auxiliary losses matter. CrossFlow also improves FID when used as a decoder for latents produced by a DiT generator in the VA-VAE latent space.

Our contributions are:

- We introduce CrossFlow, a one-step generation framework in which the input prior and output image may live in different representation spaces.
- We implement CrossFlow as a ViT-based latent-to-pixel generator with direct pixel-space perceptual and adversarial refinement.
- We show strong one-step ImageNet 256×256 generation results and demonstrate that the same model can replace a VAE decoder for generated latents.

2 Related Work

CrossFlow is related to diffusion and flow-based generation, latent diffusion and autoencoding, and generative models that allow the input and output spaces to differ. The distinction from most prior work is architectural as well as objective-level: CrossFlow keeps the generative input in a compact latent space but supervises the model directly in pixel space.

2.1 Diffusion, Flow Matching, and Few-Step Generation

Diffusion models learn to reverse a noise-corruption process. DDPMs [1] and score-based models [2] established this paradigm, while DDIM [10] introduced deterministic sampling for faster inference. Flow matching [4; 3] instead learns a velocity field that transports noise to data along a prescribed path. Transformer architectures such as DiT [5] and SiT [6] improved scalability and have become common backbones for high-quality image generation. Many methods reduce the number of sampling steps. Consistency models [11] learn mappings that remain consistent along probability-flow ODE trajectories, enabling one-step or few-step generation, and improved consistency training [12] stabilizes this objective. Progressive distillation [13], distribution matching distillation [14], MeanFlow [15], and improved MeanFlow [16] also compress or reformulate iterative sampling into fewer evaluations. CrossFlow shares the goal of efficient generation, but changes the supervised space. These methods typically define the trajectory, velocity, and prediction target in the smae space; CrossFlow takes a noisy latent-space input while predicting a pixel-space image.

2.2 Latent Diffusion and Autoencoders

Latent diffusion models [17] reduce generation cost by running diffusion in the latent space of a pretrained autoencoder rather than directly in pixel space [18]. This strategy underlies many modern systems, including SDXL [19], Stable Diffusion 3 [7], FLUX [8], and SANA [9]. The autoencoder is therefore a core part of the system, and methods such as VQ-VAE [20], VQ-GAN [21], and DC-AE [22] study the trade-off among compression, reconstruction fidelity, and efficiency. A related line of work improves compatibility between autoencoders and latent generators. LSGM [23] proposed score-based generative modeling in latent space with joint optimization; REPA-E [24] tuned VAEs with latent diffusion Transformers through representation alignment; DSD [25] interpreted diffusion as self-distillation for end-to-end latent diffusion; and unified-latent VAEs [26; 27] co-optimize the tokenizer and diffusion model. These methods address the mismatch between tokenization and generation, but they generally retain separate encoder, generator, and decoder modules. CrossFlow explores a complementary decomposition: the encoder may remain fixed, while the generator and decoder are merged into one model trained on noisy latent inputs and pixel-space targets.

2.3 Generation Across Different Spaces

Natural images have strong spatial and semantic structure, so their intrinsic dimension can be lower than their ambient pixel dimension. GANs map low-dimensional noise directly to images [28; 29; 30], but standard normalizing flows [31; 32] require invertible transformations and therefore impose dimension matching. Augmented normalizing flows [33] and injective flows [34] relax this restriction in different ways. Standard diffusion and flow-matching models [10; 1; 2; 4] usually avoid the issue by defining the full trajectory inside a single space. CrossFlow targets the latent-to-pixel case. It does not require an invertible map and does not decode a generated latent through a separately trained module. Instead, the latent path is used only to define noisy inputs and time derivatives, while the supervised prediction is an image.

3 Method

Diffusion models are typically confined to a single operating space. As a consequence, they lack native support for cross-space generation, such as directly mapping from noisy latents to pixels. This restriction is particularly pronounced in latent diffusion models (LDMs), which necessitate an auxiliary decoder to project the generated latents back into the pixel domain. We propose *CrossFlow* to bridge this gap by extending flow-based supervision to cross-space configurations. By optimizing the model to map a latent prior directly to a sample in the image space, CrossFlow eliminates the need for sequential latent generation and separate decoding.

3.1 Background: Latent Diffusion Models (LDM)

Consider a data sample $x \sim p_{\text{data}}$ and Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$. The LDM first maps the data into a latent representation via an encoder $z = \mathcal{E}(x)$, and subsequently constructs a noisy latent trajectory:

$$z_t = \alpha(t)z + \beta(t)\epsilon, \tag{1}$$

where $t \in [0, 1]$ parameterizes the noise schedule. The coefficient functions $\alpha(t)$ and $\beta(t)$ satisfy the boundary conditions $\alpha(0) = 1$, $\beta(0) = 0$, $\alpha(1) = 0$, and $\beta(1) = 1$. Generative modeling in an LDM proceeds by solving the probability-flow ordinary differential equation (PF-ODE):

$$z_0 = z_1 + \int_1^0 v(z_\tau, \tau)\, \mathrm{d}\tau, \tag{2}$$

where $v(z_\tau, \tau)$ denotes the marginal velocity of the latent state z_τ. The resulting clean latent z_0 is subsequently mapped back to the pixel space using a decoder, yielding $x_0 = \mathcal{D}(z_0)$. To estimate the intractable marginal velocity $v(z_t, t)$, we leverage the tractable conditional velocity $v(z_t, t, z)$:

$$v(z_t, t, z) = \dot{\alpha}(t)z + \dot{\beta}(t)\epsilon = \left(\dot{\alpha}(t) - \frac{\dot{\beta}(t)}{\beta(t)}\alpha(t)\right) z + \frac{\dot{\beta}(t)}{\beta(t)} z_t, \tag{3}$$

where dots indicate time derivatives with respect to t. The diffusion model F_θ is optimized via score-matching regression:

$$\mathcal{L}_v = \mathbb{E}_{x,\epsilon,t} \left\| F_\theta(z_t, t) - v(z_t, t, z) \right\|_2^2. \tag{4}$$

At convergence, $F_\theta(z_t, t)$ recovers the marginal velocity $v(z_t, t)$ via the conditional expectation:

$$v(z_t, t) = \mathbb{E}_{z \sim p(\cdot|z_t)} \left[v(z_t, t, z) \right]. \tag{5}$$

During inference, Eq. (2) is integrated numerically to transport z_t toward the clean latent z_0. This formulation strictly mandates that the model output $F_\theta(z_t, t)$ resides in the same mathematical space as z_t, fundamentally preventing standard LDMs from executing end-to-end latent-to-pixel generation.

One-step generative models relax the integration overhead by learning a direct mapping from the latent prior z_1 to the clean target z_0 within a single evaluation. However, because their underlying derivations remain tied to intra-space velocity fields, they cannot be readily extended to cross-space generation. To overcome this limitation, we introduce a unified framework for cross-space flow supervision.

3.2 A Unified Framework for Cross-Space Flow Supervision

In principle, one-step models map between arbitrary points along a probability path without explicit numerical integration. To analyze these methodologies systematically, we first introduce a unified formulation that encompasses a broad family of single-step generative paradigms. For $0 \le r \le t \le 1$ and arbitrary coefficient functions $\lambda(t, r)$ and $\gamma(t, r)$, we define the target function:

$$\mathcal{Z}(z_t, t, r; \lambda, \gamma) = \lambda(t, r) z_t + \frac{1}{\gamma(t, r)} \int_t^r v(z_\tau, \tau)\, \mathrm{d}\tau. \tag{6}$$

This formulation highlights how varying the parameterization of the ODE flow recovers different architectures. In the first regime, where $\lambda(t, r) = \gamma(t, r) = 1$, $\mathcal{Z}$ corresponds to the exact solution of the PF-ODE. Setting $r = 0$ recovers consistency models [11; 12], whereas specifying a flexible destination $r \in [0, 1]$ yields consistency trajectory models [35]. This ODE-based parameterization is also closely linked to shortcut models [36]. In a second regime, where $\lambda = 0$, $\mathcal{Z}$ simplifies to a time-averaged velocity integral. Specifically, choosing $\gamma(t, r) = r - t$, $\alpha(t) = 1 - t$, and $\beta(t) = t$ instantiates MeanFlow [15] and Improved MeanFlow [16].

Despite its generality, this unified representation reveals why existing methods are incompatible with cross-space generation: their training objectives depend inherently on the latent-space velocity field. For instance, few-step models like Improved MeanFlow [16] supervise the network by matching a linear combination of the model output and its time derivative to a local latent-velocity target. Such formulations rely entirely on the structural homogeneity between the input and output spaces. To lift this constraint, we introduce two design criteria.

Criterion 1: Eliminating Latent-Velocity Dependencies. To enable cross-space supervision, the objective function must be structurally decoupled from latent-space velocities. Direct supervision via velocity terms prevents parameterizing a generator that maps latents to pixels.

Following standard conventions [15], we differentiate Eq. (6) with respect to t. Simplifying the expression via Eq. (3) and Eq. (5) yields:

$$\partial_t \gamma(t,r)\mathcal{Z}(z_t,t,r;\lambda,\gamma) + \gamma(t,r)\frac{\mathrm{d}\mathcal{Z}}{\mathrm{d}t} = \partial_t(\gamma(t,r)\lambda(t,r))\left(\alpha(t) - \dot{\alpha}(t)\frac{\beta(t)}{\dot{\beta}(t)}\right)\mathbb{E}_{z_0\sim p(\cdot|z_t)}[z_0] + \left(\partial_t(\gamma(t,r)\lambda(t,r))\frac{\beta(t)}{\dot{\beta}(t)} + \gamma(t,r)\lambda(t,r) - 1\right)v(z_t,t). \tag{7}$$

To cancel the problematic marginal velocity term $v(z_t,t)$ on the right-hand side, we enforce the following structural constraint:

$$\partial_t(\gamma(t,r)\lambda(t,r))\frac{\beta(t)}{\dot{\beta}(t)} + \gamma(t,r)\lambda(t,r) - 1 = 0 \implies \gamma(t,r)\lambda(t,r) = 1 - \frac{\phi(r)}{\beta(t)}, \tag{8}$$

where $\phi(r)$ is a function dependent solely on the destination timestep r. This provides our first necessary condition for cross-space supervision.

Criterion 2: Endpoint Boundary Conditions at $r = 0$. We additionally require the framework to yield a valid image at the destination boundary $r = 0$, which mirrors the endpoint of the underlying PF-ODE:

$$\mathcal{Z}(z_t,t,0;\lambda,\gamma) = z_t + \int_t^0 v(z_\tau,\tau)\,\mathrm{d}\tau. \tag{9}$$

Aligning Eq. (9) with the general formulation in Eq. (6) mandates the boundary conditions $\lambda(t,0) = 1$ and $\gamma(t,0) = 1$.

Training Objective. Synthesizing the generalized trajectory target in Eq. (7) with our design criteria yields a flexible family of cross-space optimization objectives. We replace the latent-space representation $\mathcal{Z}$ in Eq. (7) with a pixel-space neural network $F_\theta(z_t,t,r)$ and optimize it to reconstruct the paired pixel-space target x_0. This yields the general cross-space loss function:

$$\mathcal{L}_{\text{General}} = \mathbb{E}_{x_0,\epsilon,t,r}\left\|\partial_t\gamma(t,r)\,F_\theta(z_t,t,r) + \gamma(t,r)\frac{\mathrm{d}F_\theta(z_t,t,r)}{\mathrm{d}t} - \frac{\phi(r)\dot{\beta}(t)}{\beta(t)^2}\left(\alpha(t) - \dot{\alpha}(t)\frac{\beta(t)}{\dot{\beta}(t)}\right)x_0\right\|_2^2, \tag{10}$$

where $\frac{\mathrm{d}F_\theta}{\mathrm{d}t}$ denotes the total derivative evaluated along the latent diffusion trajectory. This objective is governed by four schedule functions. The pair $\alpha(t)$ and $\beta(t)$ determines the latent diffusion path, while $\gamma(t,r)$ and $\phi(r)$ isolate the specific cross-space statistics learned by the network. To satisfy Criterion 2, we enforce $\gamma(t,0) = 1$ and $\phi(0) = 0$. Together, these boundary conditions guarantee that evaluating $F_\theta(\epsilon,1,0)$ with a standard normal prior $\epsilon \sim \mathcal{N}(0,I)$ maps directly to a valid pixel-space sample.

For our primary empirical evaluations, we adopt a simple and intuitive default instantiation: a linear noise schedule given by $\alpha(t) = 1 - t$ and $\beta(t) = t$, alongside $\phi(r) = r$ (satisfying $\phi(0) = 0$) and $\lambda(t,r) = 1$. Substituting these choices into Eq. (8) uniquely determines $\gamma(t,r) = 1 - \frac{r}{t}$. Plugging these schedules into Eq. (10) yields the concrete CrossFlow objective utilized in our experiments:

$$\mathcal{L}_{\text{CF}} = \mathbb{E}_{x_0,\epsilon,t,r}\left\|\frac{r}{t^2}\cdot\left(F_\theta(z_t,t,r) + \frac{t(t-r)}{r}\frac{\mathrm{d}F_\theta(z_t,t,r)}{\mathrm{d}t} - x_0\right)\right\|_2^2. \tag{11}$$

Crucially, F_θ is never optimized to predict a latent-space displacement vector. Instead, it natively outputs a clean image while its trajectory dynamics along the latent space are structurally bound by the cross-space flow loss. We explore alternative valid configurations of α, β, γ, and ϕ across synthetic toy domains in Appendix C.

3.3 Implementation Details

Computing the time derivative. Following Improved MeanFlow [16], the total time derivative $\frac{\mathrm{d}F_\theta}{\mathrm{d}t}$ is evaluated along the latent diffusion trajectory via a Jacobian-Vector Product (JVP): $\frac{\mathrm{d}F_\theta}{\mathrm{d}t} = \text{JVP}\Big(F_\theta(z_t,t,r),,\big(v(z_t,t,z),1,0\big)\Big)$, where the first tuple defines the function inputs and the second

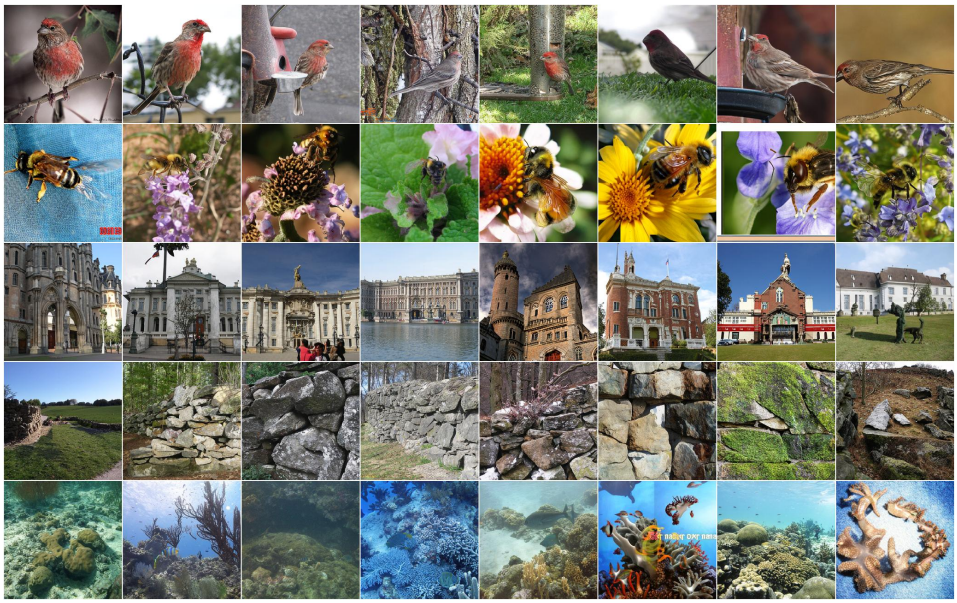


Figure 2: **Uncurated 1-NFE class-conditional samples on ImageNet** 256×256**.** Samples are generated by CrossFlow-XL from latent noise directly to pixels. Rows correspond to class 12 (house finch, linnet, *Carpodacus mexicanus*), class 309 (bee), class 698 (palace), and class 973 (coral reef), respectively.

tuple represents the corresponding tangent directions. To optimize computational efficiency, we execute this step using forward-mode automatic differentiation (`torch.func.jvp`), which incurs a marginal overhead of approximately twice the runtime of a standard forward pass. Following the convention established by MeanFlow [15], the JVP term is detached from the computational graph during backpropagation.

Two-Stage Training Pipeline. In addition to the generator F_θ, our framework relies on an encoder $\mathcal{E}$ to project input images into the latent domain. Although the cross-space objective theoretically supports end-to-end joint optimization of F_θ and $\mathcal{E}$, simultaneous training exhibits optimization instabilities in practice. To mitigate this issue, we adopt a decoupled, two-stage training pipeline. In the first stage, the encoder $\mathcal{E}$ is trained within a standard autoencoder framework using a combination of reconstruction and adversarial objectives; the corresponding decoder is subsequently discarded. In the second stage, the weights of $\mathcal{E}$ are frozen, and the cross-space generator F_θ is trained from scratch. This modular formulation allows CrossFlow to seamlessly integrate high-performance, pretrained structural encoders. By default, we employ the encoder from VA-VAE [37]. We defer the development of stable single-stage joint training strategies to future work.

Loss function. Because the CrossFlow objective directly aligns the network outputs and their time derivatives with the pixel-space target x_0, the framework can natively leverage advanced structural and perceptual image metrics to guide generation. Mirroring standard practice in autoencoder training, our total optimization objective augments the pixel-wise $L1$ loss with an adversarial GAN loss and perceptual loss. Notably, incorporating the analytical weight $\frac{r}{t^2}$ into the perceptual loss induces training instability. We therefore omit this weighting factor during perceptual loss computation.

4 Experiments

We evaluate CrossFlow on ImageNet-1k [38] at resolution 256×256. We follow the standard class-conditional protocol and report Fréchet Inception Distance (FID) [39] on 50,000 generated samples. The main evaluation is one-function-evaluation generation (1-NFE): the model receives a sample from the latent prior and outputs pixels directly. Unless otherwise stated, we use the linear schedule $\alpha(t) = 1 - t$, $\beta(t) = t$.

Table 1: **System-level comparison on ImageNet** 256×256 **generation.** FID is evaluated on 50,000 generated samples. $\times 2$ in NFE indicates classifier-free guidance. Methods are grouped by sampling regime and generation space; CrossFlow is a one-step latent-to-pixel generator.

Method	NFE	Space	Params	FID ↓	**Method**	NFE	Space	Params	FID ↓
Multi-step latent-space diffusion/flow					*1-NFE pixel-space GAN*				
DiT-XL/2 [5]	250×2	latent	675M	2.27	BigGAN-deep [29]	1	pixel	56M	6.95
SiT-XL/2 [6]	250×2	latent	675M	2.06	StyleGAN-XL [30]	1	pixel	166M	2.30
SiT-XL/2 + REPA [43]	250×2	latent	675M	**1.42**	GigaGAN [44]	1	pixel	569M	3.45
Multi-step pixel-space diffusion/flow					*1-NFE pixel-space diffusion/flow*				
ADM-G [18]	250×2	pixel	554M	4.59	EPG-L/16 [45]	1	pixel	540M	8.82
SiD, UViT [46]	1000×2	pixel	2.5B	2.44	pMF-B/16 [47]	1	pixel	118M	3.12
VDM++ [48]	256×2	pixel	2.5B	2.12	pMF-L/16 [47]	1	pixel	410M	2.52
SiD2 [49]	512×2	pixel	–	1.38	pMF-H/16 [47]	1	pixel	956M	2.22
PixelDiT-XL/16 [50]	100×2	pixel	797M	**1.61**					
1-NFE latent-space diffusion/flow					***1-NFE latent-to-pixel (ours)***				
iCT-XL/2 [12]	1	latent	675M	34.24	CrossFlow-B	1	latent→pixel	95M	2.54
Shortcut-XL/2 [36]	1	latent	676M	10.60	CrossFlow-L	1	latent→pixel	315M	1.69
MeanFlow-XL/2 [15]	1	latent	676M	3.43	CrossFlow-XL	1	latent→pixel	461M	**1.62**
iMF-XL/2 [16]	1	latent	659M	1.72					



Figure 3: **Visualization of** $F_\theta(z_t, t, 0)$ **as** t **varies from 0 (left, clean latent) to 1 (right, pure noise).** Each row represents a different semantic category. The interpolation preserves recognizable structure while progressively introducing stochastic variation and noise.

We implement F_θ with a Vision Transformer (ViT) backbone [40]. Following ViT conventions, we evaluate three variants: **CrossFlow-B** with 12 layers and hidden dimension 768, **CrossFlow-L** with 24 layers and hidden dimension 1024, and **CrossFlow-XL** with 28 layers and hidden dimension 1152. All models are trained using the Muon optimizer [41; 42], with a learning rate=8e-4 and epochs=160.

4.1 Main Results: 1-NFE FID

We first evaluate one-step image generation. Table 1 compares CrossFlow against prior ImageNet 256×256 results, across different representation spaces and sampling regimes.

Overall, CrossFlow achieves strong results with a single function evaluation. CrossFlow-XL achieves 1.62 FID, a competitive performance with multi-step latent diffusion methods such as DiT-XL/2 (2.27 FID with 250 steps) and SiT-XL/2 (2.06 FID with 250 steps). Among the one-step methods, our CrossFlow improves over the one-step latent-space method iMF-XL/2 (1.72 FID), while avoiding a separate decoder at inference. We provide uncrated samples in Figure 2 for a qualitative comparison.

We also demonstrate the scalability of the CrossFlow model: the FID improves from 2.54 (CrossFlow-B) to 1.62 (CrossFlow-XL) as we increase the parameters from 95M to 461M.

Table 2: **Ablation studies.** Unless otherwise specified, all models use CrossFlow-B, the VA-VAE encoder, the full loss configuration, the Muon optimizer, r-only time conditioning, no class-label dropout, and 160 training epochs.

Ablation	FID ↓	Ablation	FID ↓
Reference setting			
Full CrossFlow-B	**2.54**		
Encoder choice			
Encoder: SD-VAE [17], 16 latent channels	4.08	Encoder: FLUX.2 [51], 128 latent channels	3.81
Auxiliary losses and objective variants			
Loss: $\phi(r) = 0$	8.70	Loss: JVP with estimated marginal velocity	7.06
Loss: without GAN loss	8.86	Loss: without perceptual loss	30.21
Loss: without both perceptual and GAN losses	50.65		
Training and conditioning choices			
Optimizer: Muon → AdamW	2.80	Time conditioning: $r \to (r, t)$	2.73
Class conditioning: drop class labels by 10%	2.76		

4.2 Ablation Studies

We ablate our primary design choices using the CrossFlow-B architecture. As shown in Table 2, these experiments are categorized into encoder selection, auxiliary losses and objective variants, and training or conditioning configurations. Unless otherwise specified, our baseline reference setting utilizes the VA-VAE encoder, the full CrossFlow objective (including both perceptual and GAN losses), the Muon optimizer, r-only time conditioning, and no class-label dropout.

Encoder Selection. Our framework leverages the latent space of a pretrained VAE. In this suite, we compare our default VA-VAE [37] (32 latent channels) against SD-VAE [17] (16 latent channels) and FLUX.2 [51] (128 latent channels). The results indicate that the VA-VAE latent space yields the strongest performance, achieving an FID of 2.54, compared to 3.81 for FLUX.2 and 4.08 for SD-VAE. This trend aligns directly with the established gFID rankings of these VAEs on ImageNet-1k. These findings suggest that the specific structure of the latent representation could contribute to direct, single-step latent-to-pixel generation—though this experiment does not entirely isolate individual contributing factors, such as latent channel capacity.

Loss and Objective Variants. We next evaluate the impact of individual training objectives. Removing the adversarial (GAN) loss degrades the FID from 2.54 to 8.86, while omitting the perceptual loss causes a much more severe drop to 30.21. Excluding both auxiliary losses yields a poor FID of 50.65. The cross-space flow objective provides the necessary structural training signal for single-step generation, whereas the perceptual and adversarial losses act as vital regularizers that dramatically boost high-frequency sample fidelity.

Furthermore, setting $\phi(r) = 0$ results in an FID of 8.70, and utilizing a JVP (Jacobian-Vector Product) formulation with estimated marginal velocity yields 7.06.

Crucially, these results do not imply that the core CrossFlow loss is secondary. While perceptual and GAN losses are standard staples in image VAE training [21], a traditional VAE decoder cannot natively generate samples directly from noise. The cross-space flow objective provides the necessary structural training signal for single-step generation, whereas the perceptual and adversarial losses act as vital regularizers that dramatically boost high-frequency sample fidelity. We explore the sensitivity of the GAN loss weight further in Figure 4 and provide comprehensive details in Appendix B.5.

Training and Conditioning Choices. The remaining ablations demonstrate that while our default optimization choices are beneficial, they are less critical to overall performance than the core loss configuration. Swapping the Muon optimizer for AdamW increases the FID to 2.80. Transitioning from r-only time conditioning to full (r, t) conditioning yields an FID of 2.73, while introducing a 10% class-label dropout rate results in an FID of 2.76.

4.3 Ablation Studies

We ablate the main design choices using CrossFlow-B. Table 2 groups the experiments into encoder choice, objective and auxiliary losses, and training or conditioning choices. Unless otherwise specified, the reference setting uses the VA-VAE encoder, the CrossFlow objective with perceptual and GAN losses, the Muon optimizer, r-only time conditioning, and no class-label dropout. This organization separates representation choices from loss choices and secondary training choices, making the relative effect of each design axis easier to read.

Encoder ablation:. Our model leverages the encoder from a pretrained VAE. Here, we compare the performance between our default VA-VAE [37] with SD-VAE [17] and FLUX.2 [51]. The encoder ablation shows that the VA-VAE latent space gives the best result among the tested encoders. FLUX.2 reaches 3.81 FID and SD-VAE reaches 4.08 FID, compared with 2.54 FID for VA-VAE. This is consistent with the ranking of GFID of those VAE in ImageNet-1k. This suggests that the latent representation matters for direct one-step latent-to-pixel generation, although the experiment does not isolate a single cause, such as channel count alone.

Loss ablation: We first ablate the choice of training loss, Removing the GAN loss degrades FID from 2.54 to 8.86, while removing the perceptual loss causes a larger degradation to 30.21. Removing both auxiliary losses gives 50.65 FID. These results should not be interpreted as saying that the CrossFlow loss is incidental. Perceptual and GAN losses are common in image VAE training [21], but a standard VAE decoder cannot generate from noise. The consistency-loss baseline also shows that the cross-space flow objective is needed for one-step generation. Overall, the CrossFlow objective supplies the structural training signal, while perceptual and adversarial losses substantially improve sample fidelity. Figure 4 further shows that the GAN loss weight matters; Appendix B.5 provides details.

The remaining ablations show that the default training choices are meaningful but less dominant than the objective and loss configuration. Replacing Muon with AdamW gives 2.80 FID, using (r, t) time conditioning gives 2.73 FID, and dropping class labels with 10% probability gives 2.76 FID.

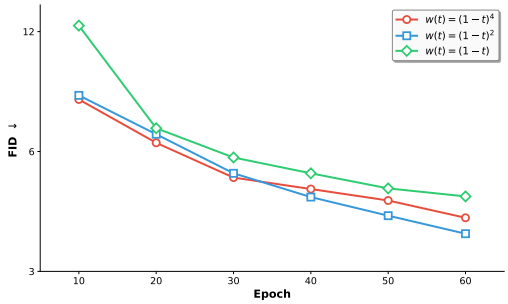


Figure 4: **Ablation of the GAN weight scheduler $w(t)$.** Higher GAN weights lead to higher FID because of partial mode collapse, while overly small weights on high-noise latents slow initial convergence. The optimized schedule balances stability and convergence speed.

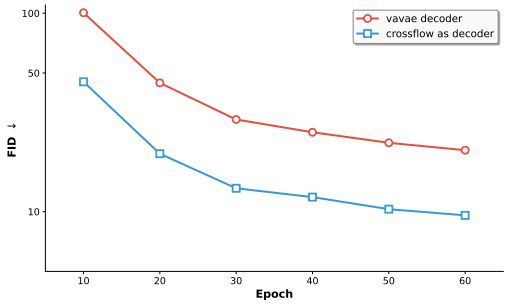


Figure 5: **CrossFlow as a VAE decoder.** We evaluate CrossFlow as a VAE decoder for latent diffusion and report FID over training epochs for a LightningDiT-B/1 generator in the VA-VAE latent space. CrossFlow consistently improves generation quality over the VA-VAE decoder.

4.4 Performance as a VAE Decoder

Finally, we evaluate CrossFlow as a decoder in a VAE-style pipeline. This setting tests whether the same latent-to-pixel model can serve not only as a generator, but also as a high-quality decoder for latents produced by a latent generator. We train LightningDiT-B/1 in the VA-VAE latent space [37]. Figure 5 compares the same DiT generator with different decoders as training progresses. CrossFlow used as a VAE decoder improves FID throughout training, supporting the claim that the model can decode generator-produced latents more effectively than the default VA-VAE decoder in this setting.

5 Conclusion

We introduced *CrossFlow*, a cross-space generative framework that maps latent noise directly to pixels. The central idea is to keep the efficiency of a latent input path while supervising the model with an image-valued one-step objective. Instantiated with a fixed encoder and a Transformer generator, CrossFlow unifies one-step generation and latent-to-pixel decoding in a single model. This design removes the separate pretrained decoder at inference, exposes generated images directly to perceptual and adversarial refinement losses, and improves decoding quality when input latents are produced by a latent generator. On ImageNet-1k at 256×256, CrossFlow-XL achieves 1.62 FID with a single function evaluation, and ablations show that the latent representation and pixel-space auxiliary losses are important for high-fidelity generation.

Acknowledgments and Disclosure of Funding

This work is supported by the Beijing Natural Science Foundation (Z260022) and National Natural Science Foundation of China (62561160152).

References

[1] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *NeurIPS*, 2020.

[2] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *ICLR*, 2021.

[3] X. Liu, C. Gong, and Q. Liu, "Flow straight and fast: Learning to generate and transfer data with rectified flow," *ICLR*, 2023.

[4] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, and M. Nickel, "Flow matching for generative modeling," *ICLR*, 2023.

[5] W. Peebles and S. Xie, "Scalable diffusion models with transformers," *ICCV*, 2023.

[6] N. Ma, M. Goldstein, M. S. Albergo, N. M. Boffi, E. Vanden-Eijnden, and S. Xie, "Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers," *ECCV*, 2024.

[7] P. Esser, S. K. andreas Blattmann, R. Entezari, J. Müller, H. Saini, Y. Levi, D. Lorenz, A. Sauer, F. Boesel, and et al, "Scaling rectified flow transformers for high-resolution image synthesis," *ICML*, 2024.

[8] B. F. Labs, "Flux," https://github.com/black-forest-labs/flux, 2024.

[9] E. Xie, J. Chen, J. Chen, H. Cai, Y. Lin, Z. Zhang, M. Li, Y. Lu, and S. Han, "Sana: Efficient high-resolution image synthesis with linear diffusion transformers," *ICLR*, 2025.

[10] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *ICLR*, 2021.

[11] Y. Song, P. Dhariwal, M. Chen, and I. Sutskever, "Consistency models," *ICML*, 2023.

[12] Y. Song and P. Dhariwal, "Improved techniques for training consistency models," *ICML*, 2024.

[13] T. Salimans and J. Ho, "Progressive distillation for fast sampling of diffusion models," *ICLR*, 2022.

[14] T. Yin, M. Gharbi, R. Zhang, E. Shechtman, F. Durand, W. T. Freeman, and T. Park, "One-step diffusion with distribution matching distillation," *CVPR*, 2024.

[15] Z. Geng, M. Deng, X. Bai, J. Z. Kolter, and K. He, "Mean flows for one-step generative modeling," *NeurIPS*, 2025.

[16] Z. Geng, Y. Lu, Z. Wu, E. Shechtman, J. Z. Kolter, and K. He, "Improved mean flows: On the challenges of fastforward generative models, 2025b," 2026.

[17] R. R. andreas Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," *CVPR*, 2022.

[18] P. Dhariwal and A. Q. Nichol, "Diffusion models beat gans on image synthesis," in *NeurIPS*, 2021.

[19] D. Podell, Z. English, K. L. andreas Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach, "Sdxl: Improving latent diffusion models for high-resolution image synthesis," *ICLR*, 2023.

[20] A. V. D. Oord, O. Vinyals, and et al, "Neural discrete representation learning," *NeurIPS*, 2017.

[21] P. Esser, R. Rombach, and B. Ommer, "Taming transformers for high-resolution image synthesis," *CVPR*, 2021.

[22] J. Chen, H. Cai, J. Chen, E. Xie, S. Yang, H. Tang, M. Li, Y. Lu, and S. Han, "Deep compression autoencoder for efficient high-resolution diffusion models," *ICLR*, 2025.

[23] A. Vahdat, K. Kreis, and J. Kautz, "Score-based generative modeling in latent space," 2021.

[24] X. Leng, J. Singh, Y. Hou, Z. Xing, S. Xie, and L. Zheng, "Repa-e: Unlocking vae for end-to-end tuning with latent diffusion transformers," *ICCV*, 2025.

[25] X. Wang and M. Zhang, "Diffusion as self-distillation: End-to-end latent diffusion in one model," *CoRR*, vol. abs/2511.14716, 2025.

[26] J. Heek, E. Hoogeboom, T. Mensink, and T. Salimans, "Unified latents (ul): How to train your latents," *arXiv:2602.17270*, 2026.

[27] S. Duggal, X. Bai, Z. Wu, R. Zhang, E. Shechtman, A. Torralba, P. Isola, and W. T. Freeman, "End-to-end training for unified tokenization and latent denoising," *CoRR*, vol. abs/2603.22283, 2026.

[28] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative adversarial nets," *NeurIPS*, 2014.

[29] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," in *ICLR*, 2019.

[30] A. Sauer, K. Schwarz, and A. Geiger, "StyleGAN-XL: Scaling StyleGAN to large diverse datasets," in *SIGGRAPH*, 2022.

[31] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," *ICML*, 2015.

[32] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real nvp," *ICLR*, 2017.

[33] C. Huang, L. Dinh, and A. C. Courville, "Augmented normalizing flows: Bridging the gap between generative flows and latent variable models," *ICML*, 2020.

[34] P. Sorrenson, F. Roth, K. Dreczkowski, V. Stimper, and F. Noé, "Lifting architectural constraints of injective flows," *ICLR*, 2024.

[35] D. Kim, C. Lai, W. Liao, N. Murata, Y. Takida, T. Uesaka, Y. He, Y. Mitsufuji, and S. Ermon, "Consistency trajectory models: Learning probability flow ODE trajectory of diffusion," in *ICLR*, 2024.

[36] K. Frans, D. Hafner, S. Levine, and P. Abbeel, "One step diffusion via shortcut models," in *ICLR*, 2025.

[37] J. Yao, B. Yang, and X. Wang, "Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models," in *CVPR*, 2025.

[38] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. IEEE, 2009, pp. 248–255.

[39] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in neural information processing systems*, vol. 30, 2017.

[40] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *ICLR*, 2021.

[41] K. Jordan, Y. Jin, V. Boza, Y. Jiacheng, F. Cesista, L. Newhouse, and J. Bernstein, "Muon: An optimizer for hidden layers in neural networks," 2024. [Online]. Available: https://kellerjordan.github.io/posts/muon/

[42] J. Liu, J. Su, X. Yao, Z. Jiang, G. Lai, Y. Du, Y. Qin, W. Xu, E. Lu, J. Yan, Y. Chen, H. Zheng, Y. Liu, S. Liu, B. Yin, W. He, H. Zhu, Y. Wang, J. Wang, M. Dong, Z. Zhang, Y. Kang, H. Zhang, X. Xu, Y. Zhang, Y. Wu, X. Zhou, and Z. Yang, "Muon is scalable for LLM training," *Arxiv abs/2502.16982*, 2025.

[43] S. Yu, S. Kwak, H. Jang, J. Jeong, J. Huang, J. Shin, and S. Xie, "Representation alignment for generation: Training diffusion transformers is easier than you think," *ICLR*, 2025.

[44] M. Kang, J.-Y. Zhu, R. Zhang, J. Park, E. Shechtman, S. Paris, and T. Park, "Scaling up GANs for text-to-image synthesis," in *CVPR*, 2023.

[45] J. Lei, K. Liu, J. Berner, H. Yu, H. Zheng, J. Wu, and X. Chu, "There is no VAE: End-to-end pixel-space generative modeling via self-supervised pre-training," in *ICLR*, 2026.

[46] E. Hoogeboom, J. Heek, and T. Salimans, "Simple diffusion: End-to-end diffusion for high resolution images," in *ICML*, 2023.

[47] Y. Lu, S. Lu, Q. Sun, H. Zhao, Z. Jiang, X. Wang, T. Li, Z. Geng, and K. He, "One-step latent-free image generation with pixel mean flows," *arXiv:2601.22158*, 2026.

[48] D. P. Kingma and R. Gao, "Understanding diffusion objectives as the ELBO with simple data augmentation," in *NeurIPS*, 2023.

[49] E. Hoogeboom, T. Mensink, J. Heek, K. Lamerigts, R. Gao, and T. Salimans, "Simpler diffusion (SiD2): 1.5 FID on ImageNet512 with pixel-space diffusion," in *CVPR*, 2025.

[50] Y. Yu, W. Xiong, W. Nie, Y. Sheng, S. Liu, and J. Luo, "PixelDiT: Pixel diffusion transformers for image generation," *arXiv preprint arXiv:2511.20645*, 2025.

[51] B. F. Labs, "FLUX.2: Frontier Visual Intelligence," https://bfl.ai/blog/flux-2, 2025.

[52] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *NeurIPS*, 2019.

[53] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-art natural language processing," in *EMNLP*, 2020.

[54] R. Wightman, "Pytorch image models," 2019.

[55] A. Obukhov, M. Seitzer, P.-W. Wu, S. Zhydenko, J. Kyl, and E. Y.-J. Lin, "High-fidelity performance metrics for generative models in pytorch," 2020. [Online]. Available: https://github.com/toshas/torch-fidelity

[56] Q. Sun, Z. Jiang, H. Zhao, and K. He, "Is noise conditioning necessary for denoising generative models?" in *ICML*, 2025.

[57] B. Zheng, N. Ma, S. Tong, and S. Xie, "Diffusion transformers with representation autoencoders," *ICLR*, 2026.

[58] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, V. Khalidov, M. Szafraniec, S. E. Yi, M. Ramamonjisoa, F. Massa, D. Haziza, L. Wehrstedt, J. Wang, T. Darcet, T. Moutakanni, L. Sentana, C. Roberts, A. Vedaldi, J. Tolan, J. Brandt, C. Couprie, J. Mairal, H. Jégou, P. Labatut, and P. Bojanowski, "Dinov3," *Arxiv abs/2508.10104*, 2025.

A Broader Impact

This work studies a general method for efficient image generation and does not introduce a domain-specific deployment. Potential benefits include lower inference cost and simpler latent-to-pixel generation pipelines, which may make high-quality synthesis more accessible for research and creative applications. At the same time, the method inherits standard risks of image generation systems, including the creation of misleading or harmful synthetic content and the reproduction of biases present in the training data. We therefore expect deployment to require the same safeguards as other generative image models, such as data curation, provenance or watermarking mechanisms, content filtering, and application-specific human oversight.

B Implementation Details

This appendix provides implementation details that are useful for reproduction but not required to understand the main method. The main text summarizes the model design and the role of the auxiliary losses.

B.1 Compute and Software

All experiments are conducted on 8 Linux nodes, each with 8 NVIDIA H20 GPUs. Our implementation is based on PyTorch [52], Transformers [53], and timm [54]. We use torch-fidelity [55] for FID computation. Training uses PyTorch Distributed Data Parallel with a per-GPU batch size of 32. We use the same fixed hyperparameter set across experiments rather than dense hyperparameter tuning.

B.2 Architecture and Conditioning

We use the VA-VAE encoder [37] as the fixed image encoder and a standard ViT backbone [40] as the CrossFlow generator. On ImageNet 256×256, the generator outputs 16×16 image patches, giving 256 image patch tokens. We additionally use 16 context tokens for class and time conditioning.

The theoretical formulation permits conditioning on both t and r. In the implementation, we condition only on r. This choice follows the observation that a model can infer the noise level t from the noisy latent input [56], and it keeps the input format the same for reconstruction and one-step generation. Table 2 reports the corresponding ablation: using both (r, t) gives 2.73 FID, while the default r-only conditioning gives 2.54 FID.

Table 3: **Training setup for CrossFlow variants on ImageNet** 256×256**.** All variants use the same training schedule and differ mainly in depth, hidden dimension, and parameter count.

Hyperparameter	CrossFlow-B	CrossFlow-L	CrossFlow-XL
Depth (layers)	12	24	28
Hidden dimension	768	1024	1152
# parameters	95M	315M	461M
Patch size	16	16	16
Epochs	160	160	160
Batch size	2048	2048	2048
Optimizer	Muon	Muon	Muon
Learning rate	8e-4	8e-4	8e-4
LR warmup	5 epochs	5 epochs	5 epochs
EMA decay	selected from 0.999/0.9995/0.9999		
Weight decay	1e-6	1e-6	1e-6

B.3 Time Sampling

For the time distribution, we start from a uniform distribution on $[0, 1]$ and apply the shifted-time transformation used in RAE [57],

$$t \mapsto \frac{\alpha t}{1 + (\alpha - 1)t}, \qquad \alpha = \sqrt{\text{latent dimension}/4096}. \tag{12}$$

This transformation adjusts the effective time distribution for different latent dimensionalities. With 50% probability, we set $t = r$, in which case the CrossFlow loss reduces to the reconstruction endpoint. With the remaining 50% probability, we sample two shifted times independently and assign them to t and r according to the time ordering used by the training objective.

B.4 Auxiliary Pixel-Space Losses

To improve perceptual quality, we combine the CrossFlow objective with perceptual and GAN losses following common VAE training practice [21]. The perceptual loss is applied to the same scaled prediction and target used in the reconstruction-compatible objective:

$$\mathcal{L}_{\text{perc}} = \text{PerceptualLoss}\left(\hat{x} + \frac{\big(\beta(t) - \beta(r)\big)\beta(t)}{\dot{\beta}(t)\beta(r)}\,\frac{d\hat{x}}{dt},\, \Big[\alpha(t) - \frac{\beta(t)}{\dot{\beta}(t)}\dot{\alpha}(t)\Big]x_0\right). \tag{13}$$

For adversarial training, we use a hinge GAN objective. The generator loss is

$$\mathbb{E}_{t\sim T, z_t\sim p_t} - w(t)D(G(z_t, r)), \tag{14}$$

and the discriminator loss is

$$\mathbb{E}_{t\sim T, z_t\sim p_t, x\sim p} \max(1 - D(G(z_t, r)), 0) + \max(1 + D(x), 0). \tag{15}$$

Here $w(t)$ controls the relative adversarial strength across time.

For the GAN discriminator architecture, we use the same architecture as dinoGAN in RAE [57]. For the perceptual loss, we map images to DINOv3-B features [58] and compute a Huber loss between the feature tensors.

B.5 GAN Weighting and Diagnostics

Because CrossFlow directly outputs pixels, adversarial training can be applied in the same space where image quality is evaluated. However, we observe that a uniform GAN weight can overemphasize low-signal regions of the trajectory. Figure 6 illustrates the resulting collapse: when the vanilla GAN loss is used, low-time samples can ignore the class label and noise and collapse to similar images.

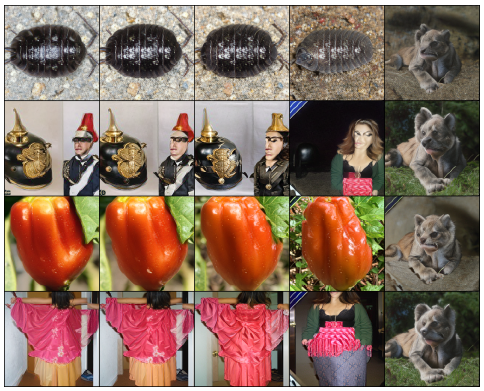


Figure 6: Illustration of GAN collapse under a vanilla adversarial loss. Rows show different tuples of clean image, latent z_1, class label, and noise; columns show decreasing time values. The model remains reasonable in the high-signal region but can collapse at low time values.

Figure 7 compares the gradient norms of the GAN loss and the CrossFlow loss across time. The GAN gradient remains relatively stable, whereas the CrossFlow gradient varies with time, motivating a time-dependent adversarial weight.

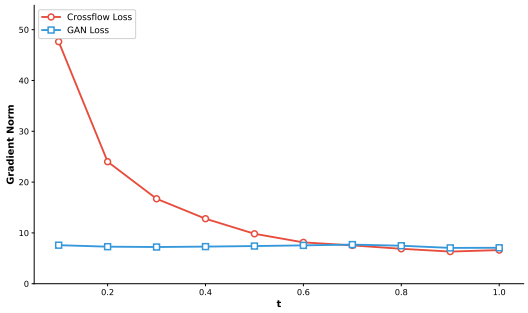


Figure 7: Gradient norms of the GAN loss and CrossFlow loss across time, averaged over 204,800 training samples. The mismatch in relative gradient magnitude motivates weighting the adversarial loss by time.

We therefore weight the generator loss by $w(t)$. We experiment with $w(t) = 1 - t$, $(1 - t)^2$, and $(1 - t)^4$, and use $(1 - t)^2$ in the main experiments. Figure 4 shows the resulting FID trajectories: too small a weight leads to weak adversarial refinement, while too large a weight can cause collapse.

C Additional Details on the CrossFlow Design Space

This appendix gives the details behind Sec. **??**. The purpose of the toy experiment is diagnostic: it checks which mathematical choices give a usable one-step map from noisy latents to images. It is not used as a main benchmark.

The CrossFlow objective Eq. (10) contains several time-dependent coefficients. The noise schedule α, β defines the latent probability path, while ϕ, λ, γ determine the one-step statistic and its induced training residual. The natural design order is:

$$\text{choose } \phi(r) \quad \Longrightarrow \quad \text{choose } \lambda(t, r) \quad \Longrightarrow \quad \text{derive } \gamma(t, r).$$

Indeed, the velocity-cancellation condition Eq. (8) gives

$$\gamma(t, r)\lambda(t, r) = 1 - \frac{\phi(r)}{\beta(t)}. \tag{16}$$

Therefore, once ϕ and λ are chosen, the original coefficient γ is determined by

$$\gamma(t, r) = \frac{1 - \phi(r)/\beta(t)}{\lambda(t, r)}. \tag{17}$$

When $\phi(r) \neq 0$, we can divide the residual by $\phi(r)$. We denote the divided coefficient by

$$\tilde{\gamma}(t, r) := \frac{\gamma(t, r)}{\phi(r)} = \frac{1}{\lambda(t, r)} \left(\frac{1}{\phi(r)} - \frac{1}{\beta(t)} \right). \tag{18}$$

Thus, after Observation 1, the practical design space is most clearly discussed in terms of $\tilde{\gamma}$, while the original coefficient is always recovered as

$$\gamma(t, r) = \phi(r)\tilde{\gamma}(t, r). \tag{19}$$

We use a Swiss-roll toy problem as a diagnostic. All variants use the same MLP, the same fixed encoder, the same detached directional derivative, the same paired-data conditional velocity, and 800 training steps. Lower sliced Wasserstein distance (SWD) is better. The toy implementation uses the reverse time convention, where $t = 0$ is noise and $t = 1$ is the clean latent; for readability, the formulas below are written in the paper convention.

Observation 1: choose $\phi(r) \neq 0$ so the loss contains the data term. The first question is whether we can set $\phi(r)$ to the constant zero function. If $\phi(r) = 0$, Eq. (16) gives

$$\gamma(t, r)\lambda(t, r) = 1. \tag{20}$$

Table 4: Swiss-roll diagnostic for the main design choices. In O2 and O3, the variants are written in terms of the divided coefficient $\tilde{\gamma} = \gamma/\phi$. The original coefficient is $\gamma = \phi\tilde{\gamma}$. The cosine rows use $\alpha^2 + \beta^2 = 1$; all other rows use the linear schedule. Lower SWD is better.

Observation	Variant	SWD ↓
O1	$\phi(r) = \beta(r)$, $\lambda = 1$, hence $\gamma(t,r) = 1 - \frac{\beta(r)}{\beta(t)}$ and $\tilde{\gamma}(t,r) = \frac{1}{\beta(r)} - \frac{1}{\beta(t)}$	0.0087
O1	$\phi = 0$, no boundary condition	0.4483
O1	$\phi = 0$, clean boundary only	0.3938
O2	$\tilde{\gamma}(t,r) = \frac{1}{\beta(t)}$, never zero	0.2775
O2	$\tilde{\gamma}(t,r) = \frac{1}{\beta(r)} - \frac{1}{\beta(t)}$, diagonal zero $\tilde{\gamma}(t,t) = 0$	0.0087
O2	$\tilde{\gamma}_\delta(t,r) = \frac{1}{\beta(r+0.1)} - \frac{1}{\beta(t)}$, shifted zero $\tilde{\gamma}_\delta(t,t-0.1) = 0$	0.0169
O2	$\tilde{\gamma}_\delta(t,r) = \frac{1}{\beta(r+0.2)} - \frac{1}{\beta(t)}$, shifted zero $\tilde{\gamma}_\delta(t,t-0.2) = 0$	0.0455
O3	linear schedule, inverse-β $\tilde{\gamma}$, matched	0.0087
O3	linear schedule, cosine-matched $\tilde{\gamma}(t,r) = \cot\frac{\pi r}{2} - \cot\frac{\pi t}{2}$, not matched	0.3829
O3	cosine schedule, inverse-β $\tilde{\gamma}$, not matched	0.0125
O3	cosine schedule, $\tilde{\gamma}(t,r) = \cot\frac{\pi r}{2} - \cot\frac{\pi t}{2}$, matched	0.0094

In this case, the image-valued data term disappears from the residual:

$$\mathcal{L}_{\phi=0} = \mathbb{E}\left\|\partial_t\gamma(t,r)F_\theta(z_t,t,r) + \gamma(t,r)\frac{\mathrm{d}F_\theta}{\mathrm{d}t}\right\|_2^2. \tag{21}$$

For example, with $\gamma(t,r) = 1$, the loss only asks the prediction to remain constant along the latent trajectory. It does not specify which image should be predicted. Therefore, the $\phi = 0$ branch needs an additional boundary loss, similar to consistency models. The toy experiment confirms this behavior: with no boundary, the model is essentially unanchored; with only a clean boundary, generation from noise is still poor. In contrast, the branch $\phi(r) \neq 0$ performs much better because the residual itself contains the image-valued data term.

When $\phi(r) = 0$, Eq. (16) gives $\gamma(t,r)\lambda(t,r) = 1$, and the one-space statistic becomes

$$Z(z_t,t,r;\lambda,\gamma) = \lambda(t,r)\left(z_t + \int_t^r v(z_\tau,\tau)\,\mathrm{d}\tau\right), \tag{22}$$

which is a scaled consistency-model statistic. This branch asks the prediction to remain consistent along the probability-flow trajectory. Its cross-space loss is

$$\mathbb{E}_{x_0,\epsilon,t,r}\left\|\partial_t\gamma(t,r)F_\theta(z_t,t,r) + \gamma(t,r)\frac{\mathrm{d}F_\theta}{\mathrm{d}t}\right\|_2^2. \tag{23}$$

For $\gamma(t,r) = 1$, this reduces to the standard consistency-style constraint

$$\mathbb{E}_{x_0,\epsilon,t,r}\left\|\frac{\mathrm{d}F_\theta}{\mathrm{d}t}\right\|_2^2. \tag{24}$$

The constraint keeps the image prediction unchanged along the trajectory, but it does not identify which image should be predicted. A trivial constant output can satisfy it. This branch therefore needs an additional reconstruction boundary,

$$\mathbb{E}_{x_0\sim p_{\mathrm{data}}}\left\|F_\theta(\mathcal{E}(x_0),0,0) - x_0\right\|_2^2. \tag{25}$$

This makes the branch a minimal cross-space consistency baseline: the flow loss enforces invariance, while the boundary loss defines the invariant prediction.

There is also a boundary subtlety. Two-point consistency models in one space use the noisy-state boundary $Z(z_t,t,t) = \gamma(t,t)z_t$. In cross-space generation, this boundary has no direct analogue because z_t is latent-valued but the desired output is image-valued. We keep this branch as a meaningful baseline, but not as the preferred CrossFlow objective.

We therefore use the branch $\phi(r) \neq 0$. Since ϕ depends only on r, dividing the residual by $\phi(r)$ does not introduce additional t-derivative terms. Using $\tilde{\gamma} = \gamma/\phi$, the divided residual can be written as

$$\mathcal{L}_{\phi\neq0} = \mathbb{E}\left\|\partial_t\tilde{\gamma}(t,r)F_\theta(z_t,t,r) + \tilde{\gamma}(t,r)\frac{\mathrm{d}F_\theta}{\mathrm{d}t} - \frac{\dot{\beta}(t)}{\beta(t)^2}\left(\alpha(t) - \dot{\alpha}(t)\frac{\beta(t)}{\dot{\beta}(t)}\right)x_0\right\|_2^2. \tag{26}$$

Multiplying the whole residual by -1 gives an equivalent squared loss; Eq. (26) uses the sign convention induced by $\tilde{\gamma} = \gamma/\phi$. After this simplification, the remaining design questions are how to choose λ so that the resulting $\tilde{\gamma}$ has useful zeros, and how to match $\tilde{\gamma}$ to the noise schedule.

Observation 2: choose λ so the resulting $\tilde{\gamma}$ has diagonal reconstruction anchors. The second question is how to choose $\lambda(t, r)$ after fixing a nonzero $\phi(r)$. From Eq. (18),

$$\tilde{\gamma}(t, r) = \frac{1}{\lambda(t, r)} \left(\frac{1}{\phi(r)} - \frac{1}{\beta(t)} \right). \tag{27}$$

Since $\tilde{\gamma}(t, r)$ multiplies $\mathrm{d}F_\theta/\mathrm{d}t$ in Eq. (26), the derivative term vanishes wherever $\tilde{\gamma}(t, r) = 0$. At such points, Eq. (26) reduces to

$$\mathbb{E} \left\| \partial_t \tilde{\gamma}(t, r) F_\theta(z_t, t, r) - \frac{\dot{\beta}(t)}{\beta(t)^2} \left(\alpha(t) - \dot{\alpha}(t) \frac{\beta(t)}{\dot{\beta}(t)} \right) x_0 \right\|_2^2. \tag{28}$$

Thus, a zero of $\tilde{\gamma}$ turns the CrossFlow residual into a reconstruction-type loss.

The most natural choice is to place this zero on the diagonal $r = t$. If $\lambda(t, r)$ is finite and nonzero, then Eq. (27) gives

$$\tilde{\gamma}(t, t) = 0 \quad \Longleftrightarrow \quad \phi(t) = \beta(t). \tag{29}$$

Therefore, the simplest diagonal-anchor construction is

$$\phi(r) = \beta(r), \qquad \lambda(t, r) = 1. \tag{30}$$

Substituting Eq. (30) into Eq. (17) gives the original statistic coefficient

$$\gamma(t, r) = 1 - \frac{\beta(r)}{\beta(t)}, \tag{31}$$

and the divided coefficient

$$\tilde{\gamma}(t, r) = \frac{\gamma(t, r)}{\phi(r)} = \frac{1}{\beta(r)} - \frac{1}{\beta(t)}. \tag{32}$$

The toy results show why the diagonal zero matters. A simple nonzero choice, such as $\tilde{\gamma}(t, r) = 1/\beta(t)$, performs poorly because the derivative term is always present and the objective never becomes a pure reconstruction anchor. Shifted-zero variants also create reconstruction anchors, but the anchors are moved away from the natural pair (t, t). Their performance degrades as the shift grows. We therefore use the diagonal condition

$$\tilde{\gamma}(t, t) = 0, \tag{33}$$

which gives a direct reconstruction anchor at every noise level and keeps the objective easy to interpret.

Observation 3: choose ϕ and λ jointly with the noise schedule so $\tilde{\gamma}$ has the correct scale. Eq. (28) shows that, when $\tilde{\gamma}(t, t) = 0$, the diagonal residual becomes a reconstruction loss. However, it reconstructs x_0 itself only if the coefficient of F_θ matches the coefficient of x_0. This requires

$$\partial_t \tilde{\gamma}(t, t) = \frac{\dot{\beta}(t)}{\beta(t)^2} \left(\alpha(t) - \dot{\alpha}(t) \frac{\beta(t)}{\dot{\beta}(t)} \right). \tag{34}$$

Otherwise, the model is asked to fit a scaled version of x_0, which makes the supervision less direct.

A convenient construction is to choose a scalar function g and set

$$\tilde{\gamma}(t, r) = g(r) - g(t), \tag{35}$$

where

$$g'(t) = -\frac{\dot{\beta}(t)}{\beta(t)^2} \left(\alpha(t) - \dot{\alpha}(t) \frac{\beta(t)}{\dot{\beta}(t)} \right). \tag{36}$$

Then $\tilde{\gamma}(t, t) = 0$, and

$$\partial_t \tilde{\gamma}(t, t) = -g'(t) = \frac{\dot{\beta}(t)}{\beta(t)^2} \left(\alpha(t) - \dot{\alpha}(t) \frac{\beta(t)}{\dot{\beta}(t)} \right), \tag{37}$$

so Eq. (34) is satisfied.

This desired $\tilde{\gamma}$ can be implemented in the original design order. First choose

$$\phi(r) = \frac{1}{g(r)}. \tag{38}$$

Then choose

$$\lambda(t, r) = \frac{g(r) - \frac{1}{\beta(t)}}{g(r) - g(t)}. \tag{39}$$

Substituting Eqs. (38) and (39) into Eq. (18) gives

$$\begin{aligned} \tilde{\gamma}(t, r) &= \frac{1}{\lambda(t, r)} \left(\frac{1}{\phi(r)} - \frac{1}{\beta(t)} \right) \\ &= \frac{g(r) - g(t)}{g(r) - \frac{1}{\beta(t)}} \left(g(r) - \frac{1}{\beta(t)} \right) \\ &= g(r) - g(t). \end{aligned} \tag{40}$$

The original coefficient is then

$$\gamma(t, r) = \phi(r)\tilde{\gamma}(t, r) = 1 - \frac{g(t)}{g(r)}. \tag{41}$$

For the linear schedule used in our main experiments, Eq. (36) gives

$$g(t) = \frac{1}{\beta(t)}. \tag{42}$$

Therefore,

$$\phi(r) = \beta(r), \qquad \lambda(t, r) = 1, \qquad \gamma(t, r) = 1 - \frac{\beta(r)}{\beta(t)}, \qquad \tilde{\gamma}(t, r) = \frac{1}{\beta(r)} - \frac{1}{\beta(t)}. \tag{43}$$

This recovers the simple CrossFlow construction. In practice, we clip the endpoint singularity for numerical stability.

For schedules that do not satisfy $\alpha + \beta = 1$, inverse-β is no longer automatically schedule-matched. For example, for the cosine schedule

$$\alpha(t) = \cos \frac{\pi t}{2}, \qquad \beta(t) = \sin \frac{\pi t}{2}, \qquad \alpha(t)^2 + \beta(t)^2 = 1, \tag{44}$$

Eq. (36) gives

$$g(t) = \cot \frac{\pi t}{2}. \tag{45}$$

A schedule-matched construction is therefore

$$\phi(r) = \tan \frac{\pi r}{2}, \qquad \lambda(t, r) = \frac{\cot \frac{\pi r}{2} - \frac{1}{\sin(\pi t/2)}}{\cot \frac{\pi r}{2} - \cot \frac{\pi t}{2}}, \tag{46}$$

which yields

$$\tilde{\gamma}_{\cos}(t, r) = \cot \frac{\pi r}{2} - \cot \frac{\pi t}{2}. \tag{47}$$

In the Swiss-roll diagnostic, this schedule-matched cosine choice improves over raw inverse-β and gives the best cosine-schedule result.

Overall, the design rule is simple. First choose $\phi(r) \neq 0$ so that the residual contains an image-valued target. Then choose $\lambda(t, r)$ to satisfy the desired endpoint behavior and to place reconstruction anchors at useful locations. Finally, γ is determined by Eq. (17), and the divided residual is governed by $\tilde{\gamma} = \gamma/\phi$. For the linear schedule, this logic gives $\phi(r) = \beta(r)$, $\lambda(t, r) = 1$, $\gamma(t, r) = 1 - \beta(r)/\beta(t)$, and $\tilde{\gamma}(t, r) = 1/\beta(r) - 1/\beta(t)$, with mild endpoint clipping for numerical stability.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the paper's scope as a cross-space flow objective for one-step latent-to-pixel generation and decoder replacement. The claims are supported by the method derivation and ImageNet 256×256 experiments, and the introduction explicitly frames the results as complementing, rather than dominating, existing latent diffusion systems.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses practical limitations and failure modes in the Discussion, including instability of the consistency-style objective, the need to detach the Jacobian-related term with imperfect velocity estimates, and possible GAN collapse at low noise levels. The introduction also limits the scope of the empirical claim to evidence that cross-space flow objectives are promising complements to latent diffusion.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The paper presents derivations for the cross-space objective in the Method section, including the interpolation assumptions, boundary conditions, and the ordinary differential equation used to remove explicit noise dependence from the target. The work does not rely on separate formal theorems beyond these derivations.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The Experiments section specifies the dataset, resolution, metric, sample count, model variants, architecture family, encoder usage, training objective components, and key training hyperparameters. These details provide the main information needed to understand and reproduce the reported claims.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The code repository is linked in the abstract. The paper provides experimental details, but does not establish that the repository includes all data-access and execution instructions needed to faithfully reproduce every main result.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The Experiments section and hyperparameter table specify ImageNet 256×256 evaluation, FID on 50,000 generated samples, model sizes, ViT backbone, fixed pretrained encoder input, epochs, batch size, optimizer, learning rate, warmup, EMA decay candidates, and weight decay.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: The experiments report point estimates such as FID but do not report error bars, as the experiment is expensive, and most previous work did not provide FID error bar.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper provides model sizes and training hyperparameters, hardware type, memory, runtime, or total compute required for the reported experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work is a generative modeling study using standard benchmark data and does not describe human-subject experiments, private data collection, or other procedures that would indicate a deviation from the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: The paper discusses pure methodological benefits such as one-step generation and decoder replacement.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper uses a widely-used imagenet dataset, which is well tested and does not pose such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The paper cites the main existing assets and prior methods it uses or compares against, including ImageNet and pretrained autoencoder-related work.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [No]

Justification: The code repository is linked in the abstract, but the manuscript does not establish that complete asset documentation is provided alongside the code.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing experiments or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing experiments or research with human subjects, so IRB or equivalent approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method described in the paper is a cross-space flow objective for image generation and does not use LLMs as an important, original, or non-standard component of the research methodology.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.