Few-Step Diffusion Sampling Through Instance-Aware Discretizations

Supplementary Material

Contents

A. Synthetic experiment details

This section details the synthetic experiments that motivate our instance-specific framework. We begin by describing the setup of our 2D toy example and the different timestep optimization strategies under comparison. Then we provide additional qualitative comparison contributing to the motivation of our instance-level approach.

A.1. Synthetic data and noise schedule configuration

To better capture the distributional feature of high dimensional image data, we adhere to the synthetic data distribution used in [1]. Based on this, we make the following modifications.

First, for better observation of the transition trajectory from prior distribution to data distribution, we convert the variance exploding (VE) noise schedule $\alpha_t = 1, \sigma_t = t$ to flow matching Optimal Transport(OT) noise schedule $\alpha_t = 1 - t, \sigma_t = t$. This ensures that the variance of prior and data distribution are on the same order of magnitude. Specifically, we make the following adaptation.

$$t^{\text{OT}} = \frac{t^{\text{VE}}}{1 + t^{\text{VE}}}, \quad \mathbf{x}_t^{\text{OT}} = \frac{1}{1 + t^{\text{VE}}} \mathbf{x}_t^{\text{VE}}. \tag{1}$$

The equivalence and transition between noise schedules are mathematically guaranteed, for readers interested, please refer to Proposition 1 in [2] or Lemma 2 in [3]. Subsequently, we convert the epsilon prediction ϵ_θ to velocity prediction v_θ (this can also be applied to data prediction $\mathbf{x}_\theta$, we also refer the interested readers to [4] for an in depth look):

$$\begin{aligned} v_\theta(\mathbf{x}_t^{\text{OT}}, t) &= \epsilon_\theta(\mathbf{x}_t^{\text{VE}}, t_{\text{VE}}) - \mathbf{x}_0 \\ &= \epsilon_\theta(\mathbf{x}_t^{\text{VE}}, t_{\text{VE}}) - \frac{\mathbf{x}_t^{\text{OT}} - t^{\text{OT}} \cdot \epsilon_\theta(\mathbf{x}_t^{\text{VE}}, t_{\text{VE}})}{1 - t^{\text{OT}}} \\ &= \frac{\epsilon_\theta(\mathbf{x}_t^{\text{VE}}, t_{\text{VE}}) - \mathbf{x}_t^{\text{OT}}}{1 - t^{\text{OT}}}. \end{aligned} \tag{2}$$

This transition provides the following benefits: the magnitude of the transition is preserved, qualitative comparison between trajectories becomes feasible, which provides insights for the design of our training dynamics.

A.2. Comparison experiment setup

Building upon the optimal transport noise schedule, we provide the detailed settings of (a), (b), (c), (d) in the main plots. Specifically, we keep the training and sampling set for (b) (c) (d) identical. Given the NFE budget of 3,

(a) The timestep is uniformly discretized $\{\tau_i = \frac{i}{N}(T - t_0) + t_0\}_{i=0}^{3}$, serving as a baseline method.

(b) The timestep is optimized through **??** and shared during sampling. As,

$$\arg\min_{\xi} \mathbb{E}_{\mathbf{x}_T \sim \mathcal{N}(0, \sigma_T^2 \mathbf{I})} \left[\mathrm{d}(\Psi(\mathbf{x}_T, \psi), \Psi(\mathbf{x}_T, \xi)) \right]. \tag{3}$$

(c) For each prior point, we conduct an instance-level optimization problem. During sampling, we assign each point the corresponding optimized timestep. Thus the optimization can be reframed as:

$$\arg\min_{\{\xi^{\mathbf{x}_T}\}} \mathbb{E}_{\mathbf{x}_T \sim \mathcal{N}(0, \sigma_T^2 \mathbf{I})} \left[\mathrm{d}(\Psi(\mathbf{x}_T, \psi), \Psi(\mathbf{x}_T, \xi^{\mathbf{x}_T})) \right]. \tag{4}$$

(d) The timestep is optimized, the network design is simpler compared to high dimensional case, with 2 layers of FFN along with Relu activation, and apply sigmoid to normalize the output. Thus $\phi(\cdot) : \mathbb{R}^2 \to \mathbb{R}^N$. (Here 2 is the data dimension, N is the number of step.)

$$\arg\min_{\phi} \mathbb{E}_{\mathbf{x}_T \sim \mathcal{N}(0, \sigma_T^2 \mathbf{I})} \left[\mathrm{d}(\Psi(\mathbf{x}_T, \psi), \Psi(\mathbf{x}_T, \xi^{\phi})) \right]. \tag{5}$$

Uniform schedule represents (a), globally optimized represents (b) and instance-level condition represents (d).

A.3. Results

Quantitative results. We conduct a comprehensive quantitative evaluation of three timestep scheduling strategies—uniform, globally optimized, and instance-level optimized—across different step budgets. As shown in Table 1,

Table 1. Comparison of different timestep scheduling strategies across MSE, KL divergence, and Wasserstein distance under varying step numbers.

Steps	MSE ↓			KL Divergence ↓			Wasserstein ↓		
	Uniform	Global	**Instance**	Uniform	Global	**Instance**	Uniform	Global	**Instance**
3	0.0588	0.0245	0.0158	5.1574	1.4452	0.6091	0.2284	0.1466	0.0925
4	0.0213	0.0223	0.0115	1.1362	1.3032	0.6065	0.1361	0.1403	0.0872
5	0.0238	0.0122	0.0093	1.2130	0.5819	0.3463	0.1496	0.1039	0.0771
6	0.0147	0.0094	0.0076	0.6775	0.3521	0.2434	0.1139	0.0869	0.0716
7	0.0121	0.0072	0.0066	0.4180	0.2246	0.2194	0.1038	0.0787	0.0722
8	0.0098	0.0052	0.0045	0.2871	0.1143	0.1304	0.0953	0.0665	0.0595
9	0.0078	0.0046	0.0033	0.2197	0.1257	0.1026	0.0826	0.0622	0.0510
10	0.0065	0.0035	0.0031	0.1630	0.0977	0.0819	0.0791	0.0564	0.0452

we assess each method using three metrics: KL divergence and Wasserstein distance to measure distribution-level fidelity, and mean squared error (MSE) to capture per-instance reconstruction accuracy. The results reveal that instance-level scheduling outperforms uniform and globally optimized method across all metrics and step counts, especially in low-NFE settings. Notably, instance-specific schedules achieve lower divergence and error with fewer steps, highlighting the benefits of dynamically adapting the timestep schedule to each sample.

Qualitative results. As illustrated in Fig. 1, we qualitatively compare the searched timesteps across different regions. This comparison reveals that while using a uniformly optimized timestep improves overall sample correctness, the instance-specific design provides greater flexibility. This allows for more tailored trajectories that better align with the ground truth sampled data points.

Intermediary Supervision vs. Global error supervision

Our qualitative analysis presented in Fig. 1, also offer insights regarding the choice of supervision signal for learning adaptive solver parameters. Different strategies exist in prior work: methods such as AMED [5] and Bespoke Solvers [6] utilize intermediary loss terms that compare states along the sampling trajectory. In contrast, approaches like LD3 [7] and Bespoke Non-stationary Solvers compute the distance metric only at the final endpoint $\mathbf{x}_0$, effectively supervising based on the global truncation error. The strong performance achieved with global error supervision supported by arguments in LD3 [7] and theoretical validation in AYS [8], may be attributed to its robustness. Specifically, when there is a substantial NFE gap between student and teacher solvers, their intermediate trajectories can diverge significantly, potentially making intermediary supervision signals less reliable or even misleading. Global error supervision, by focusing only on the final outcome, may provide the optimization process with a larger effective search space and more flexibility in

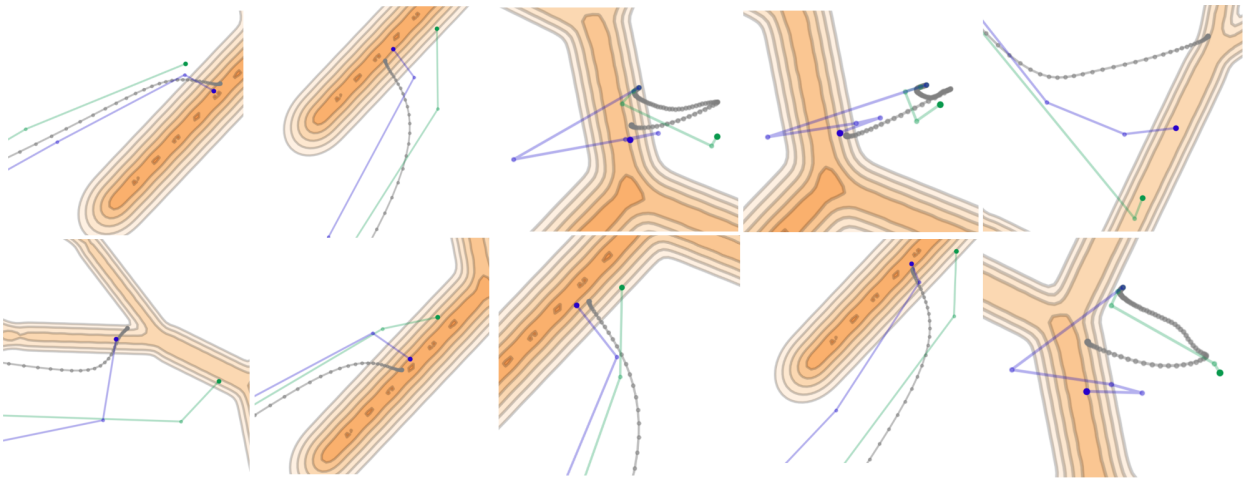


Figure 1. Qualitative comparison of sampling trajectories: Ground Truth (gray, 100 NFE), Globally Optimized Timesteps (green), and Instance-Specific Timesteps (purple). Orange coutour represents the data manifold.

determining the parameterization for the entire path.

B. Implementation and architectural specifics

B.1. Prior conditioning network architecture

Here we present a detailed description of our prior conditioning network.

Our instance-aware parameter prediction network takes a processed representation of the initial noise $\mathbf{x}_T$, combined with an embedding of any available conditional guidance $\mathbf{c}$, as its input. The initial noise $\mathbf{x}_T \sim \mathcal{N}(0, \sigma_T^2\mathbf{I})$ is first normalized to ensure unit variance, agnostic to noise schedule. We then apply Singular Value Decomposition (SVD), $\text{SVD}(\mathbf{x}_T) \rightarrow U\Sigma V^T$ for feature rearrangement. The resulting components—the singular vectors U, V^T and singular values Σ—are individually processed through FFNs with ReLU activation, and their outputs are subsequently concatenated to form the noise representation, $\text{Rep}(\mathbf{x}_T)$.

Conditional guidance $\mathbf{c}$ is transformed into a suitable embedding, $Emb(\mathbf{c})$, before being combined with $\text{Rep}(\mathbf{x}_T)$. For class labels, the one-hot encoded vector is scaled by a factor of $1/\sqrt{\text{dim}_{\text{label}}}$ to ensure unit variance, following recommendations in [9]. For text-based conditioning, exemplified by architectures such as FLUX.1-dev DiT which may utilize dual text embeddings, each text embedding is passed through Linear layers. These processed text features are then combined (e.g., via concatenation or summation) to form the unified $Emb(\mathbf{c})$. Finally, the representations $\text{Rep}(\mathbf{x}_T)$ and $Emb(\mathbf{c})$ are concatenated to serve as the complete input to our parameter prediction network. For video latent structures with $[f, C, H, W]$, directly handling the latent requires heavy computational overhead compared to images, thus we first pool the video latent on the frame dimension f to ensure computational efficiency. The overall architectural design illustrated in Fig. 2.

B.2. Mitigating exposure bias via shift factors

A common challenge in diffusion models is *exposure bias*. During training, the network (ϵ_θ) is exposed to noisy states $\mathbf{x}_t$ derived directly from clean data $\mathbf{x}_0$ (i.e., $\mathbf{x}_t = \alpha_t\mathbf{x}_0 + \sigma_t\epsilon$). During sampling, the network processes states generated iteratively from an initial noise $\mathbf{x}_T$, leading to states that can drift from the distribution seen in training. This discrepancy degrades performance, especially with few sampling steps. This mismatch problem is noted in prior literature [10–12]. Our learnable shift and scale factors are designed to mitigate such effects. We detail three noise schedules and their Signal-to-Noise Ratios (SNRs) at the starting point of sampling process, $t = T$.

EDM-VE. The Elucidating DPM (EDM) framework utilizes a Variance Exploding (VE) schedule where $\alpha_t = 1$ and $\sigma_t = t$. For the maximum time $T_{max} = 80.0$ (where sampling begins), the SNR (defined as α_T^2/σ_T^2) is $1/80^2 = 1.5625 \times 10^{-4}$.

Stable Diffusion-VP. This Variance Preserving (VP) schedule is an adaptation of the DDPM linear schedule. With $\beta(t)$ representing the noise variance schedule in continuous time, the schedule parameters are $\alpha_t = \exp(-\frac{1}{2}\int_0^t \beta(s)ds)$ and $\sigma_t = \sqrt{1-\alpha_t^2}$. Using the specific continuous $\beta(t) = (\sqrt{0.00085} \cdot (1-t) + \sqrt{0.012} \cdot t)^2$ (where t is normalized to $[0, 1]$, and sampling starts at $t = 1$), the resulting SNR is 4.7×10^{-3}.

Flow Matching-OT. The noise schedule is $\alpha_t = 1 - t, \sigma_t = t$. While the full implementation(training) of Flux [13] is not available, the sampling implementation suggests a starting timestamps of $T = 1.0$.

To alleviate exposure bias problem in few step sampling, we make the following adaptation: given the function evaluation at current state $\epsilon_\theta(\mathbf{x}_{\tau_n}, \tau_n)$, we introduce the shift factors in the following form.

$$\begin{aligned} \hat{\epsilon}_\theta(\mathbf{x}_n, \tau_n, \Delta\tau_n, \gamma_n) &:= \gamma_n \cdot \epsilon_\theta(\mathbf{x}_n, \tau_n + \Delta\tau_n), \\ \xi^\phi = \{\tau_n, \Delta\tau_n, \gamma_n\}_{n=1}^N &= \phi(\mathbf{x}_T, \mathbf{c}). \end{aligned} \tag{6}$$

Observations. ?? in the paper demonstrates that the learnable shift and scale factors yield a substantially greater impact when applied to models utilizing the EDM-VE schedule (*e.g.*, on FFHQ dataset) and DDPM-VP in LDM (*e.g.*, on LSUN-bedroom), compared to FLUX.1-dev. In FLUX.1-dev, as NFE increases, the effect of shift factors becomes more negligible and even negative compared to the other two pretrained model. We attribute this disparity primarily to the more pronounced exposure bias inherent in EDM-VE and VP (Stable Diffusion) schedule, which provides a larger scope for improvement through our shift factor design.

B.3. Training configuration and sampling efficiency

The optimization of a set of hyperparameters is known to be challenging, often exhibiting instability and necessitating meticulous design to hit optimal configurations [7]. We alleviate this by designing an instance level network, to further improve and stabilize the training procedure of our prior conditioning network, we adhere to the following settings.

Training configuration. We pre-generate a fixed teacher dataset of noise data pairs, similar to LD3 [7]. Empirically, a larger pre-generated dataset improves our model's performance. We test among Dpm_Solver, Uni_PC and iPNDM and select the best based on FID as our teacher trajectory. We empirically find that iPNDM gives the most promising teacher data. To save memory, we save the random generator state instead of raw gaussian noise. Building upon efficiency consideration, we adopt the Analytical First Step (AFS) [14], which analytically approximates the initial update without invoking the denoising network, reducing the total NFE by one. Additionally, to stabilize optimization under limited NFE budgets, we employ Exponential Moving Average (EMA) on

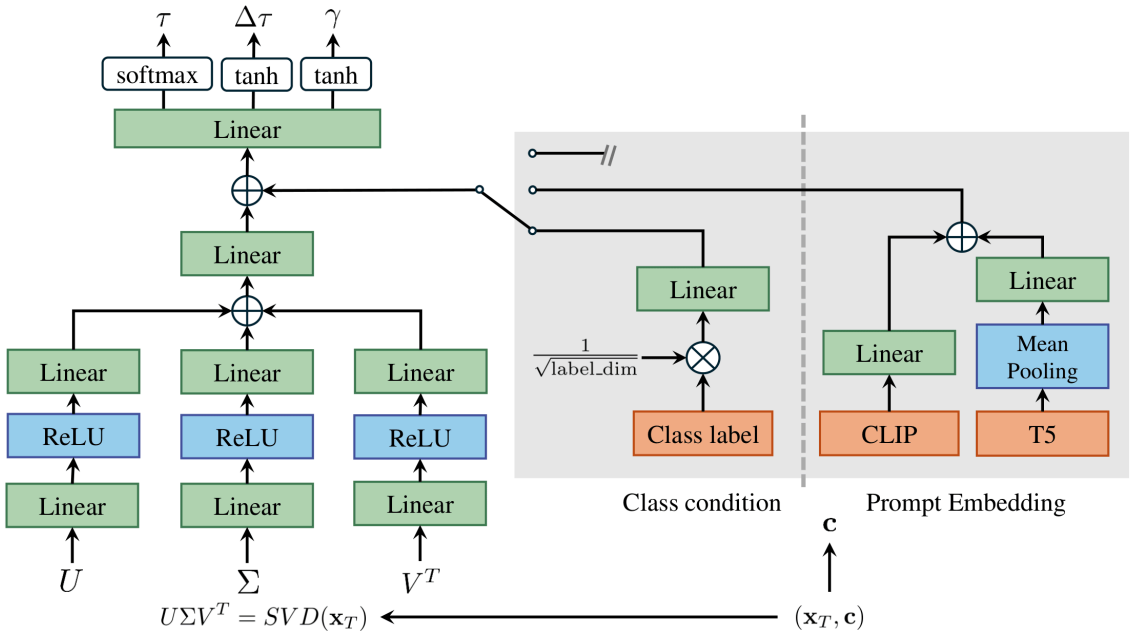


Figure 2. Architectural design of the proposed lightweight prior conditioning network. When conditional information is available, class indices are first scaled by a factor of $\frac{1}{\sqrt{\text{label\_dim}}}$ and then processed through a linear layer. For prompt embeddings (FLUX.1-dev), T5 embeddings undergo mean pooling to reduce dimensionality before being concatenated with CLIP embeddings.

ϕ, comparing configurations with and without EMA (20%) and selecting the best-performing variant based on validation results. We present the hyperparameter setting as in Table 2.

Efficiency analysis. Table 3 presents an efficiency analysis of our instance-aware parameter network. Training times reported are evaluated on NVIDIA A100 GPUs. The Sampling Overhead (%) column quantifies the ratio of our prior network's inference time to the total sampling time; this specific overhead is evaluated for an NFE=5 setting. Parameter Overhead (%) is calculated as the ratio of our prior network's parameters to those of the base diffusion model. The sampling overhead for FLUX.1-dev is 2.3%, and CIFAR10 is 2.5% and ImageNet64 is 1.9%.

Hyperparameter	CIFAR10	FFHQ	AFHQv2	ImageNet64	LSUN	FLUX.1-dev	LTX-Video
Learning Rate	0.05	0.05	0.05	0.05	0.05	0.001	0.001
Batch Size	16	16	16	8	4	4	4
GPUs (A100)	1	1	1	1	4	4	4
Nimgs/videos	10k	10k	10k	10k	10k	10k	5k
b_γ	0.05	0.05	0.05	0.05	0.05	0.01	0.01
$b_{\Delta\tau}$	0.05	0.05	0.05	0.05	0.05	0.01	0.01

Table 2. Hyperparameter settings.

Models	Training Time	Sampling Overhead (%)	Parameter Overhead (%)
CIFAR10 (32×32)	8 min	2.5%	0.43%
ImageNet64 (64×64)	40 min	1.9%	0.32%
FLUX.1-dev (512×512)	8 h†	2.3%	0.21%

Table 3. Efficiency Analysis. †: for FLUX.1-dev, we train ϕ for 2 hours on 4 A100 GPUs with a batch size of 4.

C. Additional experimental results

Here we first provide ablation experiments regarding the design components of our instance-specific paradigm. Then we provide the full experimental results across various NFE (3-8) settings, and additional qualitative results.

C.1. Ablations on design components.

Ablation instance level framework. We first ablate the design components in our INDIS framework: Singular value decomposition of the noise, discretization shifted factors and the instance-network itself. As illustrated in Table 4. It's observed that both shifted factors and instance-level design are crucial for the final results, with instance-level discretization influencing the majority of the performance.

	NFE					
Ablation (on CIFAR-10)	3	4	5	6	7	8
w/o instance level	12.05	14.44	5.36	3.83	3.84	3.20
w/o shifted factors	11.12	8.35	4.64	3.30	2.92	3.24
w/o SVD	9.60	4.31	**3.27**	**2.65**	2.77	2.62
full	**9.26**	**4.14**	3.31	2.81	**2.60**	**2.34**

Table 4. Ablation study on the components of our method. Metrics are FID ↓.

Ablation on solver choices. The improvements of instance-aware approach is agnostic to solver choices (as illustrated

in Table 5, all solver choices reach comparable few-step generation results.), while the selection of iPNDM is that it serves as a good foundation for quality improvement.

Solver choices	NFE					
	3	4	5	6	7	8
Uni_PC	68.25	43.92	24.01	13.12	6.63	4.41
Uni_PC (+INDIS)	10.88	5.88	3.84	3.57	**2.59**	2.48
DPM-solver++	68.43	46.59	24.99	12.16	6.88	4.62
DPM-solver++ (+INDIS)	9.92	6.52	4.62	3.55	3.17	2.92
iPNDM	57.39	29.78	17.35	9.95	7.61	5.41
iPNDM (+INDIS)	**9.26**	**4.14**	**3.31**	**2.81**	2.60	**2.34**

Table 5. Ablation study on different solver choices, with and without our method (INDIS), on CIFAR-10.

Ablation on teacher steps. We first ablate our teacher-steps design choices, ranging from 10-30. The results (Table 6) demonstrate a direct correlation between teacher precision and student performance. We select the 30-step iPNDM solver as the default setting in our pixel space diffusion models and latent space LSUN-Bedroom.

Teacher NFEs.	3	4	5	6	7	8
10	11.57	8.34	5.52	4.01	4.45	3.38
20	9.73	6.59	3.48	3.11	2.81	2.55
30	**9.26**	**4.14**	**3.31**	**2.81**	**2.60**	**2.34**

Table 6. Ablations on teacher solver steps.

C.2. Results on pixel space DPMs

Here we present the full results of pixel space DPMs across NFEs, as illustrated in Tables 7 to 10.

C.3. Results on latent space DPMs

Here we present the full results of latent space DPMs and flow matching models across NFEs, as illustrated in Tables 12 and 13

C.4. Results on Video Domains

Besides VBench [15], we also extend our instance-aware discretization strategy to VMBench [16], to further test the robustness of our method as illustrated in Table 14.

C.5. Comparison with Solver Distillation

Here we provide additional comparison with solver distillation approaches, including the efficient parallel gradients method EPD [17], and an SDE learning based variant AdaSDE [18], the result is illustrated in Table 11.

D. Qualitative Comparison

Here we present the qualitative comparison with teacher on FLUX (Fig. 3). Standard comparison against global heuristics and globally optimized baselines on LTX-Video (Figs. 4 and 5), FLUX.1-dev (512x512) (Figs. 6 and 7), LSUN-Bedroom (256x256) (Figs. 12a and 13a), CIFAR10 (32x32) (Fig. 11a), ImageNet (64x64) (Fig. 10a), FFHQ (64x64) (Fig. 8a) and AFHQv2 (64x64) (Fig. 9a).

Figure 3. Qualitative comparison against teacher and global baseline.

Method	NFE=3	NFE=4	NFE=5	NFE=6	NFE=7	NFE=8
best heu.	57.39	29.78	17.35	9.95	7.61	5.41
DMN	77.69	26.35	12.93	8.09	5.42	5.90
AMED	18.49	17.18	7.59	7.04	4.36	5.56
GITS	25.98	10.11	6.77	4.29	3.43	2.70
LD3	16.52	9.31	5.32	3.35	3.37	2.65
INDIS	**9.26**	**4.14**	**3.31**	**2.81**	**2.60**	**2.34**

Table 7. FID results on CIFAR10.

Method	NFE=3	NFE=4	NFE=5	NFE=6	NFE=7	NFE=8
best heu.	72.29	29.35	17.52	11.44	8.76	6.86
DMN	178.09	31.30	20.93	12.12	10.17	11.00
AMED	26.87	26.89	12.49	9.97	6.64	7.86
GITS	26.41	13.59	8.85	6.39	5.36	4.91
LD3	23.86	14.15	8.56	5.97	4.69	3.97
INDIS	**17.72**	**8.92**	**6.91**	**4.72**	**3.90**	**3.31**

Table 8. FID results on FFHQ.

Method	NFE=3	NFE=4	NFE=5	NFE=6	NFE=7	NFE=8
best heu.	40.24	15.35	9.01	6.26	4.73	3.83
DMN	178.76	33.15	26.11	16.01	13.03	10.12
AMED	31.82	18.99	7.34	8.19	4.39	5.55
GITS	24.17	12.20	8.72	6.10	5.48	4.03
LD3	17.94	9.33	6.09	3.63	2.77	2.63
INDIS	**10.15**	**4.80**	**3.48**	**2.77**	**2.37**	**2.35**

Table 9. FID results on AFHQv2.

Method	NFE=3	NFE=4	NFE=5	NFE=6	NFE=7	NFE=8
best heu.	44.93	21.32	15.53	10.27	8.64	6.60
DMN	33.72	15.24	10.47	6.74	5.39	4.98
AMED	28.06	32.69	10.74	10.63	6.66	7.71
GITS	26.41	16.41	9.85	8.39	6.44	5.64
LD3	27.82	17.03	11.55	7.53	5.63	5.40
INDIS	**18.96**	**10.95**	**7.28**	**5.82**	**4.94**	**4.49**

Table 10. FID results on ImageNet64.

	CIFAR10 32×32			**FFHQ 64×64**			**ImageNet64 (class-cond.)**			**LSUN 256×256 (latent)**		
Method	**NFE=3**	**NFE=5**	**NFE=7**	**NFE=3**	**NFE=5**	**NFE=7**	**NFE=3**	**NFE=5**	**NFE=7**	**NFE=3**	**NFE=5**	**NFE=7**
EPD	10.40	4.33	2.82	21.74	7.84	4.81	**18.28**	**6.35**	5.26	13.21	7.52	5.97
AdaSDE	12.62	4.18	2.88	23.80	8.05	5.11	18.51	6.90	5.26	18.03	6.96	5.16
INDIS	**9.26**	**3.31**	**2.60**	**17.72**	**6.91**	**3.90**	18.96	7.28	**4.94**	**12.44**	**4.99**	**3.81**

Table 11. Comparison with solver distillation. FID at NFE=3,5,7 on CIFAR10, FFHQ, class-conditional ImageNet64, and LSUN 256×256 (latent-space).

Method	NFE=3	NFE=4	NFE=5	NFE=6	NFE=7	NFE=8
best heu.	41.99	11.93	6.38	5.08	4.39	4.88
DMN	28.11	11.82	6.15	4.71	5.16	4.55
AMED	58.21	15.67	13.20	8.92	7.00	4.19
GITS	44.78	21.67	17.29	11.52	9.59	8.82
LD3	14.62	8.48	5.93	4.52	4.31	4.22
INDIS	**12.44**	**6.55**	**4.99**	**3.84**	**3.81**	**3.66**

Table 12. FID results on LSUN-Bedroom.

Metrics	Method	NFE=3	NFE=4	NFE=5	NFE=6	NFE=7
FID(↓)	RDS	64.50	30.36	30.12	23.16	**22.58**
	GOD	56.82	29.33	28.52	23.32	22.77
	INDIS	**44.35**	**26.07**	**24.89**	**22.93**	22.70
CLIP(↑)	RDS	23.29	27.77	29.66	30.42	30.76
	GOD	24.41	28.24	29.70	30.55	30.80
	INDIS	**26.33**	**28.70**	**30.01**	**30.67**	**30.86**
CMMD(↓)	RDS	1.75	1.10	0.86	0.82	0.89
	GOD	1.72	1.01	0.79	0.77	0.75
	INDIS	**1.69**	**0.98**	**0.75**	**0.73**	**0.73**

Table 13. Full comparison on FLUX.1-dev(MS-COCO). For each column, we bold the best performing method. We found AFS to be negative on this large scale pretrained model, thus reporting the result w/o AFS.

Prompt set	**Method**	**Aes.**	**Img.**	**Subj.**	**PAS**	**TCS**	**CAS**
VBench	RDS	0.579	0.597	0.963	2.162	95.643	49.615
	GOD	0.583	0.603	0.963	3.155	96.672	49.776
	INDIS	**0.593**	**0.613**	**0.964**	**3.368**	**98.337**	**49.852**
VMBench	RDS	0.552	0.601	0.955	2.956	96.053	48.981
	GOD	0.561	0.607	0.956	3.220	96.892	49.693
	INDIS	**0.583**	**0.611**	**0.961**	**3.347**	**97.187**	**50.138**

Table 14. Performance on VBench and VMBench prompt subsets.

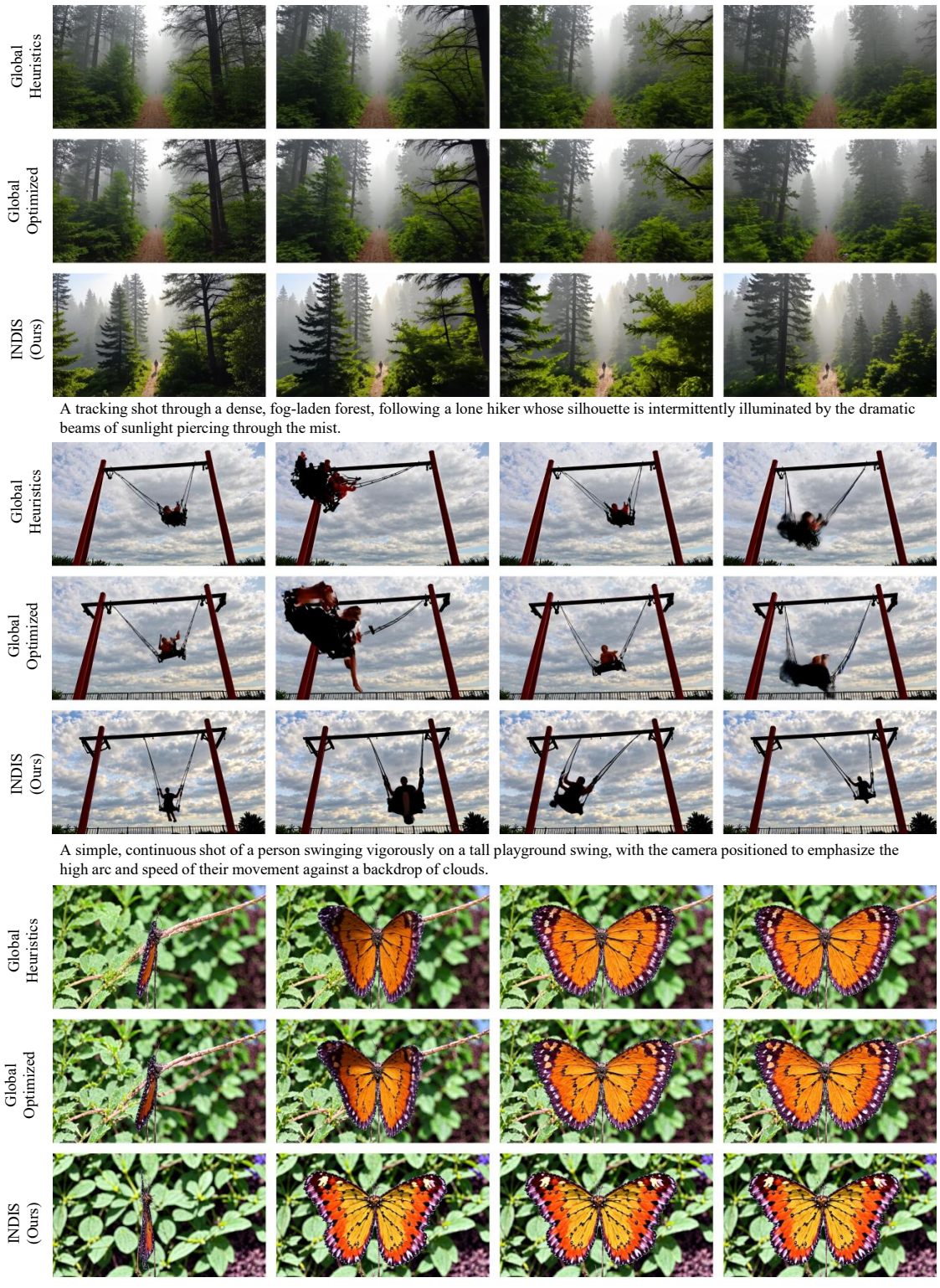


Figure 4. Qualitative Comparison (NFE=5) between global heuristics, global optimized and instance level INDIS methods (1/2) on LTX-Video

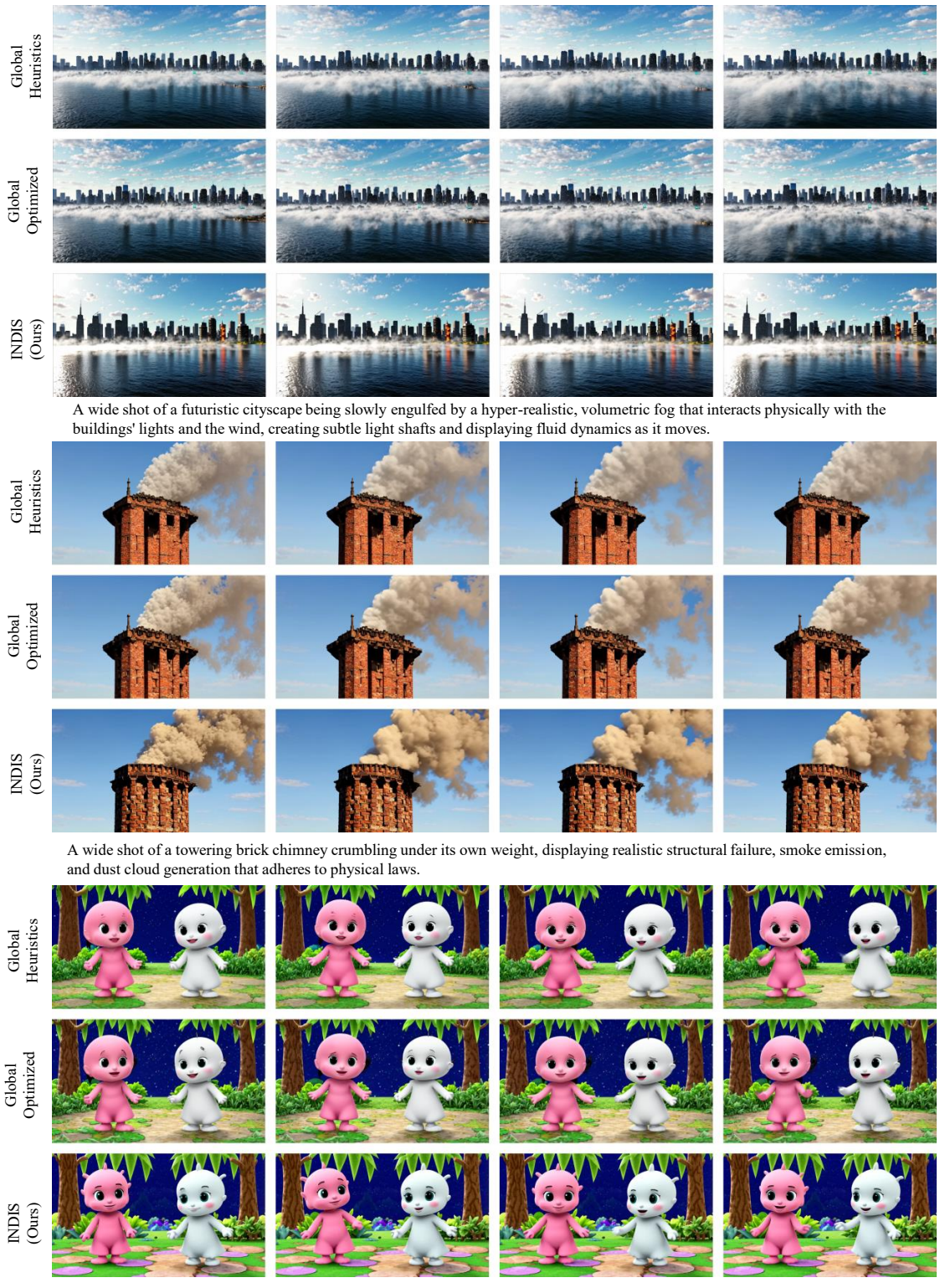
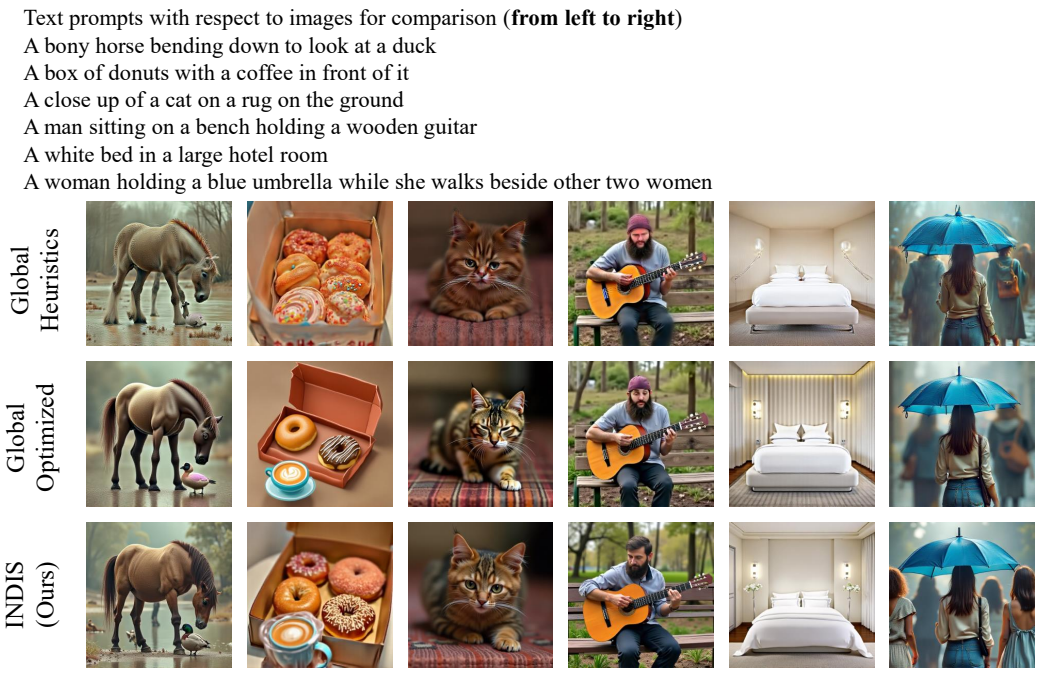
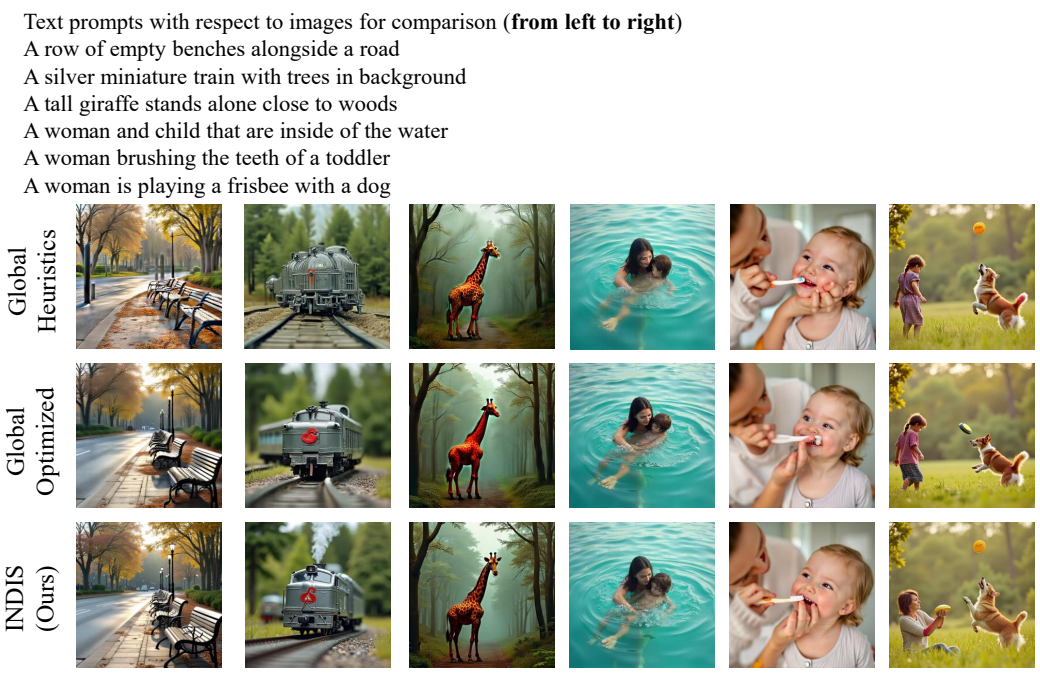


Figure 5. Qualitative Comparison (NFE=5) between global heuristics, global optimized and instance level INDIS methods (2/2) on LTX-Video



(a)



(b)

Figure 6. Qualitative Comparison (NFE=7) between global heuristics, global optimized and instance level INDIS methods (1/2) on FLUX.1-dev

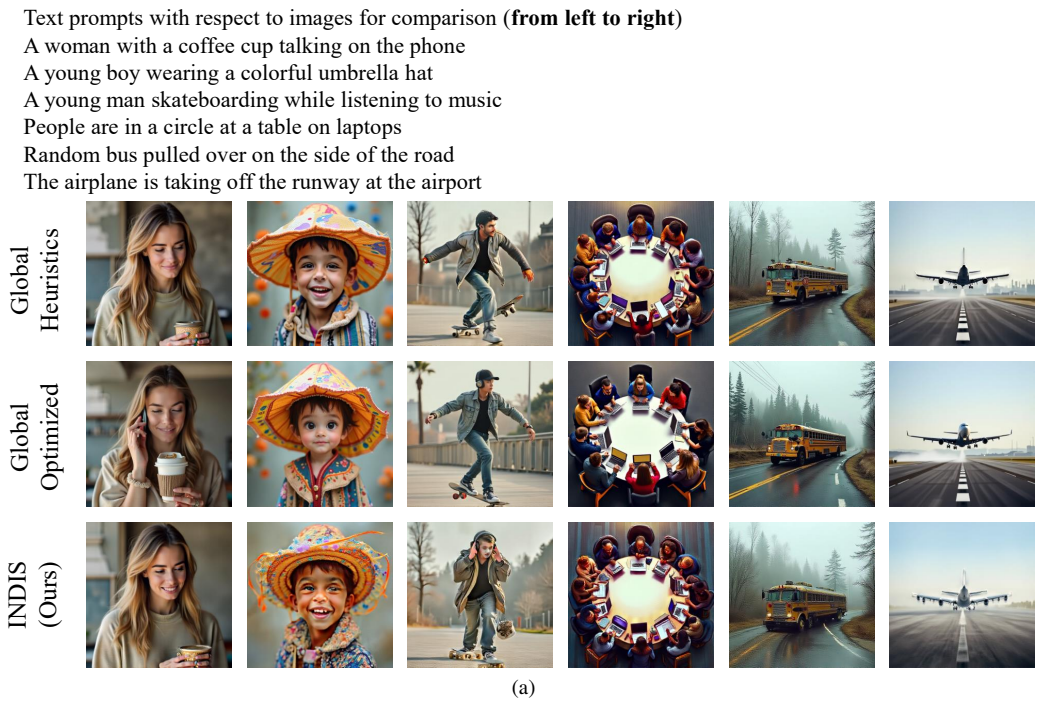


Figure 7. Qualitative Comparison (NFE=7) between global heuristics, global optimized and instance level INDIS methods (2/2) on FLUX.1-dev

(a) Selected best heuristics.

(b) INDIS

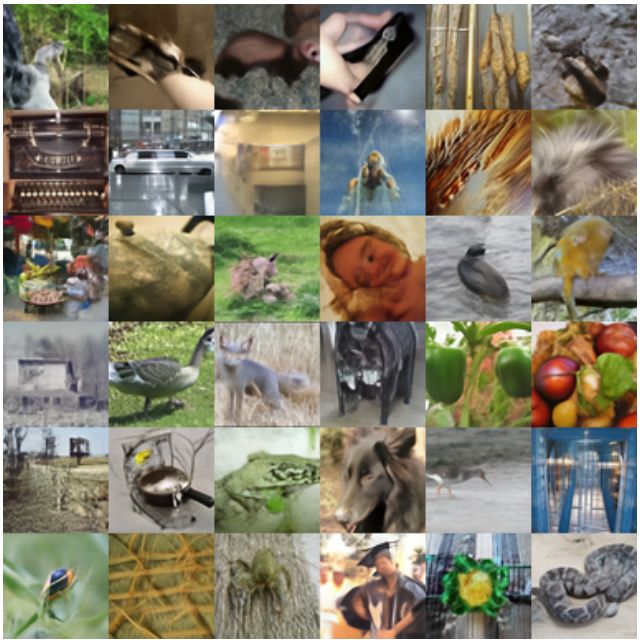
Figure 8. Qualitative comparison on FFHQ64x64 datasets with NFE=3 settings.

(a) Selected best heuristics.

(b) INDIS

Figure 9. Qualitative comparison on AFHQv2 64x64 datasets with NFE=3 settings.

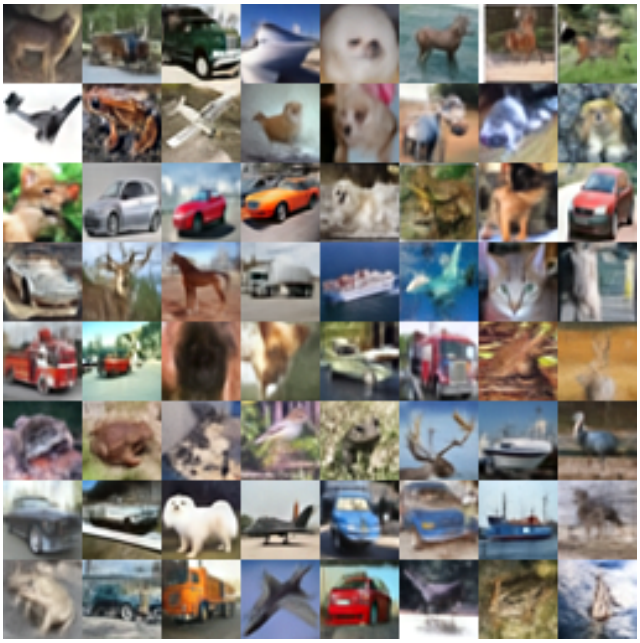
(a) Selected best heuristics.



(b) INDIS

Figure 10. Qualitative comparison on ImageNet 64x64 datasets with NFE=3 settings.

(a) Selected best heuristics.



(b) INDIS

Figure 11. Qualitative comparison on CIFAR10 32x32 datasets with NFE=3 settings.

(a) Selected best heuristics.

(b) INDIS

Figure 12. Qualitative comparison on latent space LSUN-Bedroom 256x256 datasets with NFE=3 settings.

(a) Selected best heuristics.

(b) INDIS

Figure 13. Qualitative comparison on latent space LSUN-Bedroom 256x256 datasets with NFE=4 settings.

References

[1] Tero Karras, Miika Aittala, Tuomas Kynkäänniemi, Jaakko Lehtinen, Timo Aila, and Samuli Laine. Guiding a diffusion model with a bad version of itself. In *Proc. NeurIPS*, 2024. 1

[2] Sangyun Lee, Zinan Lin, and Giulia Fanti. Improving the training of rectified flows. In *Proc. NeurIPS*, 2024. 1

[3] Ashwini Pokle, Matthew J. Muckley, Ricky T. Q. Chen, and Brian Karrer. Training-free linear image inverses via flows. *TMLR*, 2024. ISSN 2835-8856. 1

[4] Diederik P Kingma and Ruiqi Gao. Understanding diffusion objectives as the ELBO with simple data augmentation. In *Proc. NeurIPS*, 2023. 1

[5] Zhenyu Zhou, Defang Chen, Can Wang, and Chun Chen. Fast ode-based sampling for diffusion models in around 5 steps. In *Proc. CVPR*, 2024. 2

[6] Neta Shaul, Juan Perez, Ricky T. Q. Chen, Ali Thabet, Albert Pumarola, and Yaron Lipman. Bespoke solvers for generative flow models. In *Proc. ICLR*, 2024. 2

[7] Vinh Tong, Dung Trung Hoang, Anji Liu, Guy Van den Broeck, and Mathias Niepert. Learning to discretize denoising diffusion ODEs. In *Proc. ICLR*, 2025. 2, 3

[8] Amirmojtaba Sabour, Sanja Fidler, and Karsten Kreis. Align your steps: Optimizing sampling schedules in diffusion models. In *Proc. ICML*, 2024. 2

[9] Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. Analyzing and improving the training dynamics of diffusion models. In *Proc. CVPR*, 2024. 3

[10] Mang Ning, Mingxiao Li, Jianlin Su, Albert Ali Salah, and Itir Onal Ertugrul. Elucidating the exposure bias in diffusion models. In *Proc. ICLR*, 2024. 3

[11] Shanchuan Lin, Bingchen Liu, Jiashi Li, and Xiao Yang. Common diffusion noise schedules and sample steps are flawed. In *Proc. CVPR*, 2024.

[12] Mingxiao Li, Tingyu Qu, Ruicong Yao, Wei Sun, and Marie-Francine Moens. Alleviating exposure bias in diffusion models through sampling with shifted time steps. In *Proc. ICLR*, 2024. 3

[13] Black Forest Labs. Flux. https://github.com/black-forest-labs/flux, 2024. 3

[14] Tim Dockhorn, Arash Vahdat, and Karsten Kreis. Genie: Higher-order denoising diffusion solvers. In *Proc. NeurIPS*, 2022. 3

[15] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *Proc. CVPR*, 2024. 5

[16] Xinran Ling, Chen Zhu, Meiqi Wu, Hangyu Li, Xiaokun Feng, Cundian Yang, Aiming Hao, Jiashu Zhu, Jiahong Wu, and Xiangxiang Chu. Vmbench: A benchmark for perception-aligned video motion generation. *arXiv preprint arXiv:2503.10076*, 2025. 5

[17] Beier Zhu, Ruoyu Wang, Tong Zhao, Hanwang Zhang, and Chi Zhang. Distilling parallel gradients for fast ode solvers of diffusion models. In *Proc. ICCV*, 2025. 5

[18] Ruoyu Wang, Beier Zhu, Junzhi Li, Liangyu Yuan, and Chi Zhang. Adaptive stochastic coefficients for accelerating diffusion sampling. In *Proc. NeurIPS*, 2025. 5