arXiv:2610.14136v2 [cs.AI] 14 Oct 2026

CORRECTION TRAINING FOR DIFFUSION MODELS

Donghoon Ahn[1], **Runqian Wang**[1], **Alexei A. Efros**[1], **Taesung Park**[2], **Yash Belhe**[2]
[1]University of California, Berkeley [2]Reve

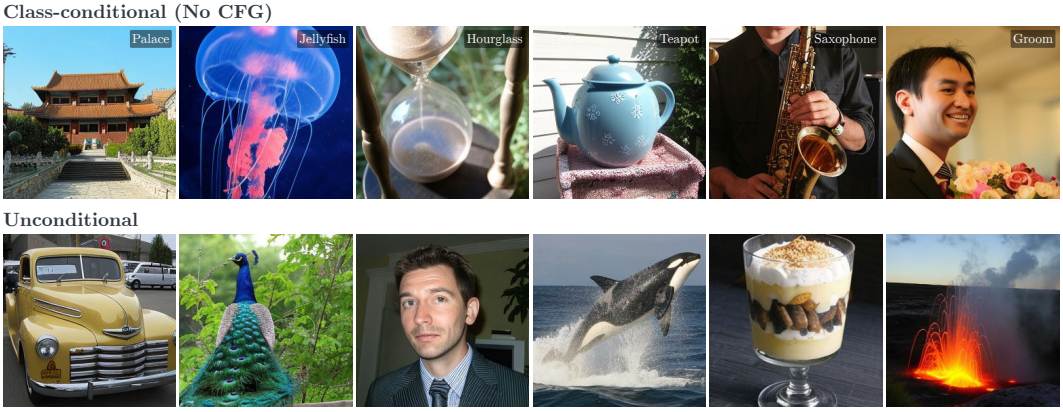


Figure 1: **ImageNet samples with correction.** *Without the use of guidance*, our correction method can achieve high-quality class-conditional (top) and *unconditional* (bottom) image samples.

ABSTRACT

Iterative sampling in diffusion models during inference can lead to error accumulation due to exposure bias. The denoiser is trained on real training images interpolated with noise, but at inference its inputs are constructed from its own imperfect predictions from the earlier steps, something it has never seen at training time. This might be one reason diffusion models often struggle without additional help, such as CFG. In this paper, we introduce a correction step for diffusion models which attempts to reduce this training-inference gap. The corrector takes the base model's own prediction as input and is trained against the same ground-truth target as the base model. This makes correction an explicit learning task, with training inputs provided by the base model. During sampling, at each denoising step, we iteratively refine the denoiser's clean-sample prediction. This allows us to correct predictions at intermediate denoising steps, preventing early artifacts from persisting into the final sample. We find that applying correction beyond the single step supervised during training is key to the significant quality gains. This simple framework leads to substantial improvement in sample quality while largely preserving diversity. We extensively evaluate our method across widely used architectures, including DiT, SiT, and JiT. Our method substantially improves upon previously reported FID scores for *unguided generation* and *unconditional generation*. Our method achieves performance competitive with classifier-free guidance (CFG), and combining correction with CFG further improves diversity and reduces oversaturation typical of CFG. Finally, we show that the additional inference cost of correction can be amortized by distilling the refined predictions back into the base diffusion model, retaining the quality gains without additional inference-time overhead.

1 INTRODUCTION

Diffusion models generate images through a sequence of denoising predictions (Ho et al., 2020; Song et al., 2020). At each sampling step, a neural network estimates clean data from a noisy input, and this estimate is used to update the sample. But one curious property of diffusion models is the

Project page: `https://sunovivid.github.io/correction-training/`

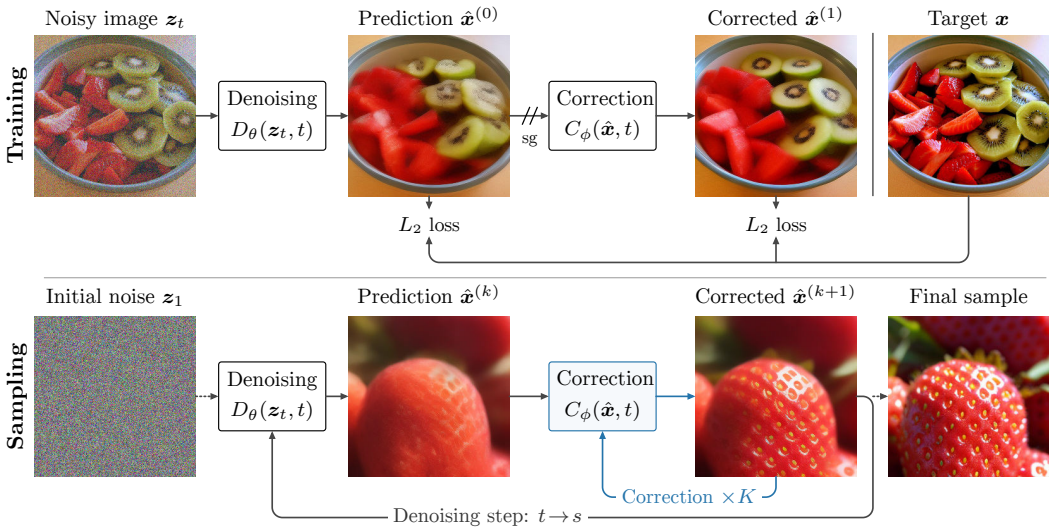


Figure 2: **Denoising and prediction correction.** During training, the denoiser and corrector predict the same clean target. Stop-gradient blocks the correction loss through the denoiser's prediction. During sampling, each denoising prediction is refined K times before the next sampling step.

training-inference gap arising from what is known as exposure bias (Ning et al., 2023; 2024; Li et al., 2024a; Yu & Zhan, 2025). The denoiser is trained on real training images interpolated with noise. At inference time, however, it receives noisy inputs that depend on *its own imperfect predictions* from earlier steps. These inputs can differ from the noisy training inputs, creating a mismatch between training and inference. As noise decreases along the denoising trajectory, prediction errors can accumulate, often producing visual artifacts and incoherent structures in the generated images. A common way to improve sample quality is classifier-free guidance (CFG) (Ho & Salimans, 2022), which generates better quality samples, but at the cost of reduced diversity and oversaturation.

In this paper, we take a more direct approach: correcting the *model's own prediction*. We introduce a *corrector* that takes the base model's prediction as input and is trained against the same ground-truth target, as illustrated in Fig. 2. This makes correction an explicit learning task, with training inputs provided by the base model. During sampling, at each denoising step, we therefore iteratively correct the denoiser's clean-sample prediction. Although the corrector is trained only on predictions from the base denoiser, we find that it can be applied repeatedly to its own outputs at inference time without additional training. This iterative correction further improves sample quality, producing sharper images with more coherent structures. We hypothesize that repeated correction helps compensate for errors accumulated over earlier denoising steps, reducing the denoiser's training–inference gap and contributing to the observed improvement (see Sec. 4.2 for details).

We evaluate our method across class-conditional, unconditional, and text-to-image generation, using established models including DiT (Peebles & Xie, 2023), SiT (Ma et al., 2024), REPA (Yu et al., 2024), and JiT (Li & He, 2026). Correction improves unguided sample quality across these settings with a relatively small amount of additional training. The gain remains at matched numbers of function evaluations (NFEs), suggesting that introducing correction steps can make more effective use of inference compute than denoising alone. In unconditional generation, correction also provides substantial improvements without any guidance signal. Comparisons with CFG on DiT and SiT show better quality at matched diversity, as measured by FID (Heusel et al., 2017) and recall (Kynkäänniemi et al., 2019). Qualitatively, correction shows less oversaturation and greater visual realism than CFG. Moreover, adding correction to CFG reduces the stereotypical "AI-generated" look of its samples. We present conditional and unconditional samples using correction in Fig. 1.

Finally, we show that the corrector can be made as efficient as standard diffusion models while retaining its quality advantage. We distill corrected predictions back into the base diffusion model. The student learns to match predictions obtained through repeated correction and requires only one forward pass per sampling step. The resulting model retains much of the quality improvement while preserving the base model's architecture, parameter count, and inference cost. This allows the benefits of correction to be incorporated into the base model without additional correction passes

Algorithm 1 Training loss

```
# D: denoiser; C: corrector
# weight: correction loss weight
t = sample_time(x)  # per image
e = randn_like(x)
z = (1 - t) * x + t * e

pred = D(z, t)
refined = C(stopgrad(pred), t)

loss = mse(pred, x)
loss += weight * mse(refined, x)
```

Algorithm 2 Sampling with correction

```
# D: denoiser; C: corrector
# K: corrections per step
z = randn(shape)
for t, s in zip(times, times[1:]):
    pred = D(z, t)
    #  correction steps
    for _ in range(K):
        pred = C(pred, t)
    v = (z - pred) / t
    z = z + (s - t) * v
sample = z
```

during generation. We hope that the effectiveness, simplicity, and efficiency of corrector can help to bridge the long-standing gap between unconditional and conditional generation.

2 Learning to Correct Denoising Predictions

2.1 Background: Denoising and Sampling

Diffusion models learn to recover data from noisy observations (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020; Song et al., 2020). Flow matching (Lipman et al., 2022; Liu et al., 2022; Albergo et al., 2025) extends the diffusion paradigm by using a linear interpolation between noise and image. Specifically, given a clean sample $\boldsymbol{x} \sim p_{\text{data}}$ and independent noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$, the training input is

$$\boldsymbol{z}_t = (1-t)\boldsymbol{x} + t\boldsymbol{\epsilon}, \qquad t \in [0,1], \tag{1}$$

where $t = 0$ denotes data and $t = 1$ denotes noise. An $\boldsymbol{x}$-prediction network D_θ estimates the clean sample from $(\boldsymbol{z}_t, t)$. Its denoising objective is

$$\mathcal{L}_{\text{denoise}} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{\epsilon},t}\left[\|D_\theta(\boldsymbol{z}_t, t) - \boldsymbol{x}\|_2^2\right]. \tag{2}$$

At each sampling step, the denoiser predicts the clean image, and the sampler uses this prediction to construct the next noisy state. For example, an Euler step from t to $s < t$ gives

$$\boldsymbol{z}_s = \frac{s}{t}\boldsymbol{z}_t + \left(1 - \frac{s}{t}\right)\hat{\boldsymbol{x}}^{(0)}, \qquad \hat{\boldsymbol{x}}^{(0)} = D_\theta(\boldsymbol{z}_t, t). \tag{3}$$

The updated state $\boldsymbol{z}_s$ then becomes the denoiser's next input. During training, however, inputs are constructed by interpolating real images with independent noise (Eq. 1). Sampling inputs therefore depend on the model's own earlier predictions and may differ from the inputs encountered during training, creating a training–inference gap (Ning et al., 2023; 2024; Li et al., 2024a).

2.2 Prediction Error During Sampling

Fig. 3 (a) illustrates how structural errors can persist along the sampling trajectory. Early predictions contain incoherent object structures. Later denoising steps add texture and facial details such as teeth, eyes, and a tongue while largely preserving the incorrect object structure. The clean targets used for training, however, are real images with coherent object structures. The denoiser learns to recover these structures from noise, but is not explicitly trained to further refine its own predictions at the same noise level.

One widely used way to obtain sharper, more coherent samples is classifier-free guidance (CFG) (Ho & Salimans, 2022). CFG strengthens conditioning by amplifying the difference between conditional and unconditional predictions. However, it can reduce sample diversity and produce oversaturated images.

In this paper, we make prediction correction an explicit learning task: we train a corrector C_ϕ that takes the denoiser's own prediction as input. During sampling, we apply this learned mapping repeatedly to improve sample quality. As we show in Sec. 4, repeated correction can improve generation even when it increases reconstruction error on training inputs. We investigate this behavior and its relationship to the training–inference gap.

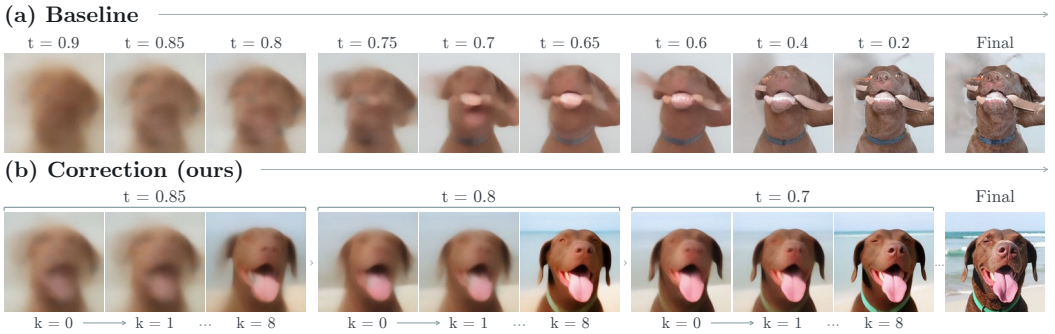


Figure 3: **Prediction correction improves structural coherence during sampling.** We show clean-sample predictions $\hat{\boldsymbol{x}}$ along denoising trajectories without (a) and with (b) correction. In (a), early structural errors persist as later denoising steps add details. In (b), iterative correction revises the prediction at each t, producing more coherent object structures before the next denoising step. Each triplet shows this refinement within one denoising step. Additional visualizations are shown in Appendix D.2 (Fig. 34).

2.3 Learning to Correct Model Predictions

Standard denoising training learns the mapping $\boldsymbol{z}_t \to \boldsymbol{x}$ by reconstructing images corrupted by a prescribed noise process. Given a noisy observation $\boldsymbol{z}_t$, the denoiser produces an initial prediction $\hat{\boldsymbol{x}}^{(0)}$, which may still contain inaccuracies. We introduce *prediction correction* as a reconstruction task that uses this prediction as its input. The corrector takes $\hat{\boldsymbol{x}}^{(0)}$ and predicts the same clean target $\boldsymbol{x}$, learning the mapping $\hat{\boldsymbol{x}}^{(0)} \to \boldsymbol{x}$ (Fig. 2).

When $\hat{\boldsymbol{x}}^{(0)}$ already captures much of the structure in $\boldsymbol{x}$, the initial denoising pass has already performed much of the reconstruction. The corrector can then build on this estimate and focus on the remaining inaccuracies. We refine $\hat{\boldsymbol{x}}^{(0)}$ by applying corrector C_ϕ again at the original time t: $\hat{\boldsymbol{x}}^{(1)} = C_\phi(\hat{\boldsymbol{x}}^{(0)}, t)$. The corrected prediction directly estimates the clean sample $\boldsymbol{x}$. We train it with

$$\mathcal{L}_{\text{corr}} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{\epsilon},t}\left[\|C_\phi(\text{sg}(\hat{\boldsymbol{x}}^{(0)}), t) - \boldsymbol{x}\|_2^2\right], \tag{4}$$

where sg is the stop-gradient operation.

Starting from a pretrained denoiser, we jointly optimize $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{denoise}} + \lambda_{\text{corr}}\mathcal{L}_{\text{corr}}$, where λ_{corr} weights the correction task. The denoising term supervises the initial prediction, while the correction term supervises its refinement. The two tasks are trained jointly, with corrector inputs generated online by the current model. Algorithm 1 summarizes the loss computation in Python-style pseudocode.

2.4 Iterative Correction at Inference

Training supervises only a single correction step, from the initial denoising prediction $\hat{\boldsymbol{x}}^{(0)}$ to the refined prediction $\hat{\boldsymbol{x}}^{(1)}$. At inference, however, the model can repeatedly refine its own outputs, further improving sample quality. The output $\hat{\boldsymbol{x}}^{(k)}$ at the k-th correction iteration is another estimate of the same clean image. This estimate may still contain inaccuracies, motivating further applications of the learned correction.

Starting from the denoising prediction, we apply K correction passes:

$$\begin{aligned} \hat{\boldsymbol{x}}^{(0)} &= D_\theta(\boldsymbol{z}_t, t), \\ \hat{\boldsymbol{x}}^{(k+1)} &= C_\phi(\hat{\boldsymbol{x}}^{(k)}, t), \qquad k = 0, \ldots, K-1. \end{aligned} \tag{5}$$

Fig. 3 illustrates how successive corrections refine intermediate predictions throughout the sampling trajectory. This provides a way to spend additional inference compute on improving the predictions at each time. At matched total inference cost, we find this more effective than simply increasing the number of denoising steps.

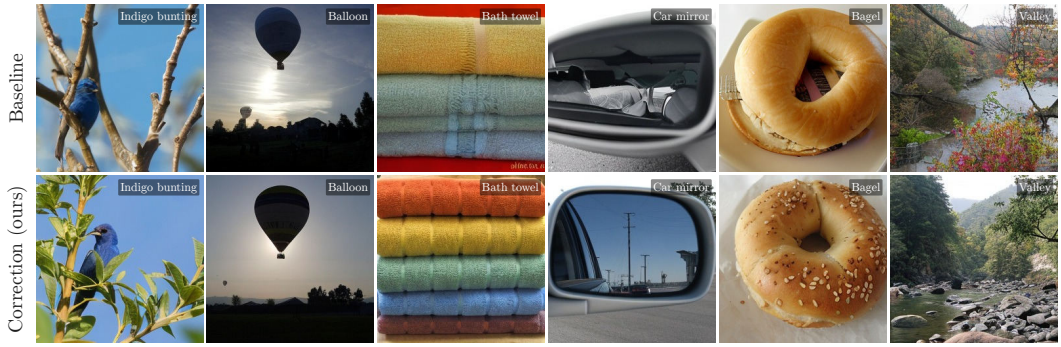


Figure 4: **Correction improves class-conditional sample quality.** We compare baseline (top) and correction (bottom), starting from the same noise in each column. Correction gives more coherent and detailed images for different classes and noises compared against the baseline.

Table 1: **Correction improves unguided class-conditional generation.** We compare FID-50K, precision, and recall on ImageNet-256 class-conditional image generation across 4 base models. We report consistent improvement on FID and precision over unguided baseline and it is competitive with CFG. All methods are evaluated at approximately 250 NFEs. See Tab. 13 for evaluation details.

DiT-XL/2	**FID** ↓	**Prec.** ↑	**Rec.** ↑
Unguided (baseline)	9.58	0.668	**0.676**
Corrective sampling	**2.07**	**0.811**	0.603
CFG (reference)	2.51	0.818	0.581

SiT$_{\text{REPA}}$-XL/2	**FID** ↓	**Prec.** ↑	**Rec.** ↑
Unguided (baseline)	5.53	0.708	**0.739**
Corrective sampling	**1.31**	**0.797**	0.703
CFG (reference)	1.35	0.803	0.699

SiT-XL/2	**FID** ↓	**Prec.** ↑	**Rec.** ↑
Unguided (baseline)	8.55	0.687	**0.669**
Corrective sampling	**1.92**	**0.812**	0.608
CFG (reference)	2.11	0.820	0.595

JiT-L/16	**FID** ↓	**Prec.** ↑	**Rec.** ↑
Unguided (baseline)	14.86	0.607	**0.672**
Corrective sampling	**2.96**	**0.766**	0.601
CFG (reference)	2.35	0.780	0.606

3 Results

We evaluate correction on class-conditional and unconditional generation (Sec. 3.1) and text-to-image generation (Sec. 3.2). We also demonstrate that the base and correction models can share most parameters and that additional correction passes can be eliminated with distillation (Sec. 3.3). We detail our experimental settings in Appendix B.

3.1 ImageNet Generation

We evaluate our method on widely used transformer-based architectures, including DiT (Peebles & Xie, 2023), SiT (Ma et al., 2024), and JiT (Li & He, 2026). We initialize both the base denoiser and a separate corrector from publicly released checkpoints and jointly fine-tune them following Algorithm 1. Unless otherwise specified, we follow each baseline's training recipe and evaluation protocol. Number of function evaluations (NFE) under correction measures both denoiser steps and correction steps. We approximately match the total NFE by reducing the number of denoising steps as the number of correction iterations K increases.

Class-conditional generation. We present class conditional image samples in Fig. 4. Correction produces sharp, realistic, and diverse images, all *without applying any guidance*. The resulting samples exhibit realism competitive with that achieved by guidance-based methods such as classifier-free guidance (Ho & Salimans, 2022), perturbation guidance (Ahn et al., 2024) and autoguidance (Karras et al., 2024). See Appendix E.1 for more samples.

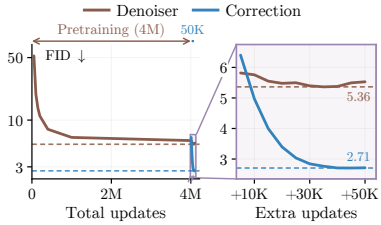
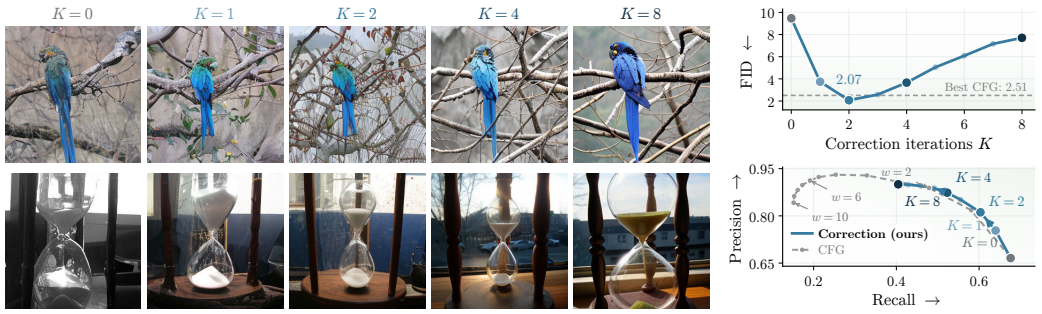


Figure 5: **Efficiency of correction training.** FID converges within relatively few corrector training steps.



(a) Effect of changing K (b) Generation quality and diversity

Figure 6: **Iterative correction provides a controllable refinement trajectory on ImageNet.** Left: Evolvement of the same samples over more correction steps. The main content remains the same while quality improves. Right: FID and precision–recall across K. Correction gives a controllable tradeoff between coverage and quality and lower minimum FID.



Figure 7: **Correction improves *unconditional* sample quality on ImageNet.** Without correction, generated samples exhibit artifacts and structural inconsistencies (top), while corrective sampling produces diverse and high-quality images (bottom). The samples in each column share the same initial noise.

In Tab. 1, we evaluate generation using FID, precision, and recall (Heusel et al., 2017; Kynkäänniemi et al., 2019). Correction *significantly reduces FID at matched NFE* (DiT: 9.58 → **2.07**, SiT: 8.55 → **1.92**, and SiT_{REPA}: 5.53 → **1.31**.). This suggests that correction is a better allocation of inference compute than simply increasing denoising steps. These gains require little additional training: for SiT_{REPA}, 50K updates amount to just 1.25% of its 4M pretraining updates. Fig. 5 compares FID convergence with correction versus continued denoising. Fig. 6 shows how the number of correction iterations K affects the final samples and their fidelity–diversity trade-off. Correction effectively suppresses artifacts and improves object coherence, yielding lower FID and higher precision at the cost of some recall. We further analyze the power spectra of the generated samples in Appendix D.3.

Unconditional generation. Unconditional generation remains a challenging task since no class or text conditioning is available to constrain the outputs (Li et al., 2024b). We show that for unconditional generation, our prediction correction significantly improves image quality and coherence, all without requiring guidance. We present qualitative comparisons of corrected and uncorrected samples in Fig. 7. Correction removes structural inconsistencies and artifacts from the baseline samples while maintaining the main subject.

Quantitatively, we compare correction and autoguidance (Karras et al., 2024) in Tab. 2 on ImageNet unconditional generation. Correction with $K = 4$ substantially improves FID over the baseline and the best evaluated autoguidance scale ($w \in \{1, 2, \ldots, 10\}$, with the released pretrained model as the guide), while larger K trades recall for higher precision (Fig. 8).

Diversity preservation. Although comparison with CFG is not our main goal, our method addresses two key limitations of CFG: reduced diversity and oversaturation. Fig. 9 shows samples from the same ImageNet class as we vary the CFG scale w and the number of correction iterations K. At higher CFG scales, the samples become less diverse, converging on canonical close-up por-

Method	FID ↓	Prec. ↑	Rec. ↑
Baseline	24.82	0.571	**0.699**
Correction ($K = 1$)	17.80	0.620	0.679
Correction ($K = 4$)	**5.37**	**0.782**	0.595
Autoguidance	9.37	0.699	0.630

Table 2: **Correction lowers unconditional FID without guidance.** 50K training updates, 250 NFE, autoguidance $w = 7$.

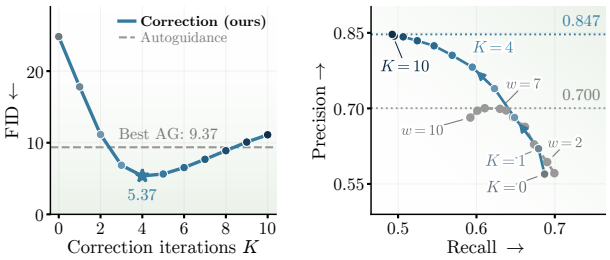


Figure 8: **Correction reaches lower FID and higher peak precision than autoguidance.** Dashed: best autoguidance FID. Dotted: each method's observed peak precision.

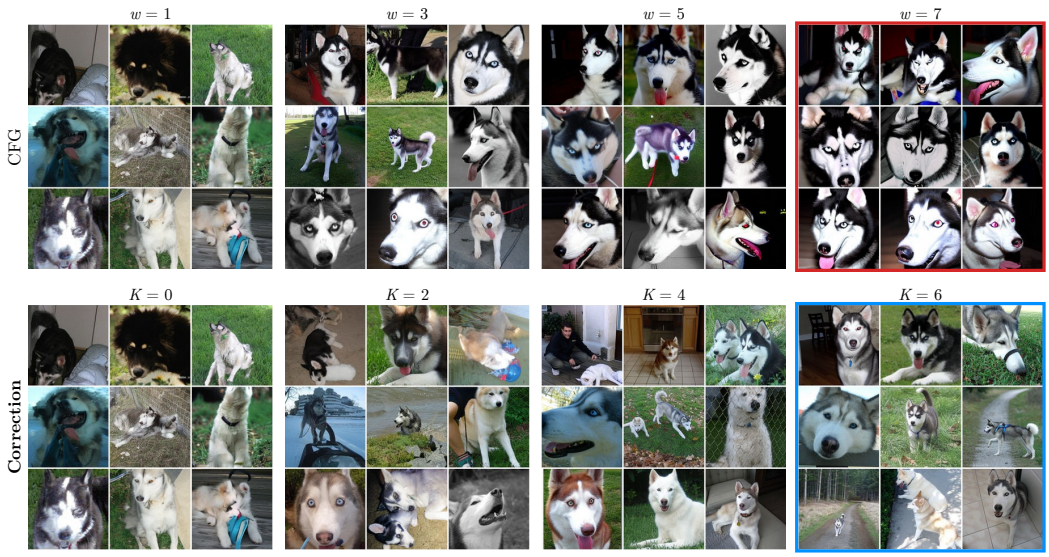


Figure 9: **Correction preserves more diversity and produces less oversaturation than CFG.** Samples from SiT_{REPA}-XL/2 with increasing correction depth K (bottom) or CFG scale w (top), at approximately matched NFE, same initial noise.

traits, while their colors become increasingly saturated. With correction, the dogs appear in more varied poses, framings, and backgrounds while retaining sharp details and realism. Additional class-wise comparisons are shown in Figs. 26 and 27 in the appendix.

3.2 Text-to-Image Generation

To assess our method on a more complex, practically relevant task, we evaluate text-to-image generation using NanoGen (Leng et al., 2026), a unified training framework for diffusion transformers. We use a pretrained DiT (Peebles & Xie, 2023) with a DDT head (Wang et al., 2026), operating in the latent space of the FLUX.2 VAE (Labs, 2025). We use the JourneyDB (Sun et al., 2023), Short-Caption, and Long-Caption subsets of the BLIP-3o pretraining mixture (Chen et al., 2025), with aspect-preserving preprocessing to 256×256. For the quantitative comparison, we jointly fine-tune the denoiser and corrector for 50K steps, adding a relatively small training cost compared with pretraining the base model.

Fig. 10 illustrates changes in object structure without guidance: correction separates the violin bodies, clarifies the guitar and car geometry, and organizes cluttered interiors. Quantitatively, Tab. 3 shows higher GenEval (Ghosh et al., 2023) and DPG-Bench (Hu et al., 2024) scores with correction at approximately matched NFE, first without CFG and then with correction applied before CFG. Both comparisons use the same-checkpoint baseline and DiffusionBench's native eval-



Figure 10: **Correction improves object structure in *text-to-image* generation.** Compared with no correction (top), correction (bottom) separates the violin bodies, clarifies the guitar and car geometry, and organizes cluttered interiors. All samples are without CFG; full prompts are in Appendix B.3.

Without CFG				*With CFG*			
Method	NFE	GenEval ↑	DPG-Bench ↑	Method	NFE	GenEval ↑	DPG-Bench ↑
Baseline	250	28.30	70.71	Baseline	250	68.08	83.38
Correction ($K = 6$)	245	**50.09**	**76.82**	**Correction** ($K = 2$)	246	**70.48**	**83.69**

Table 3: **Correction improves text-to-image alignment with and without CFG.** We observe consistent improvements on GenEval and DPG-Bench by applying correction over the baseline, both with and without CFG, evaluated at approximately matched NFE. See Table 13 for detailed settings.

uation pipeline; sampling protocols are provided in Table 13, and additional guidance scales in Appendix B.3.

Fig. 11 compares different combinations of CFG and correction. Stronger CFG alone can improve conditioning adherence and sample quality, but often at the cost of reduced diversity and oversaturated or stylized appearances. Lowering the guidance scale alleviates these effects, but may leave samples under-refined. We find that *applying correction jointly with CFG alleviates the diversity issue* while preserving the quality refinement. We observe that CFG alone produces visually similar samples, while combining CFG and correction leads to higher diversity (Fig. 11).

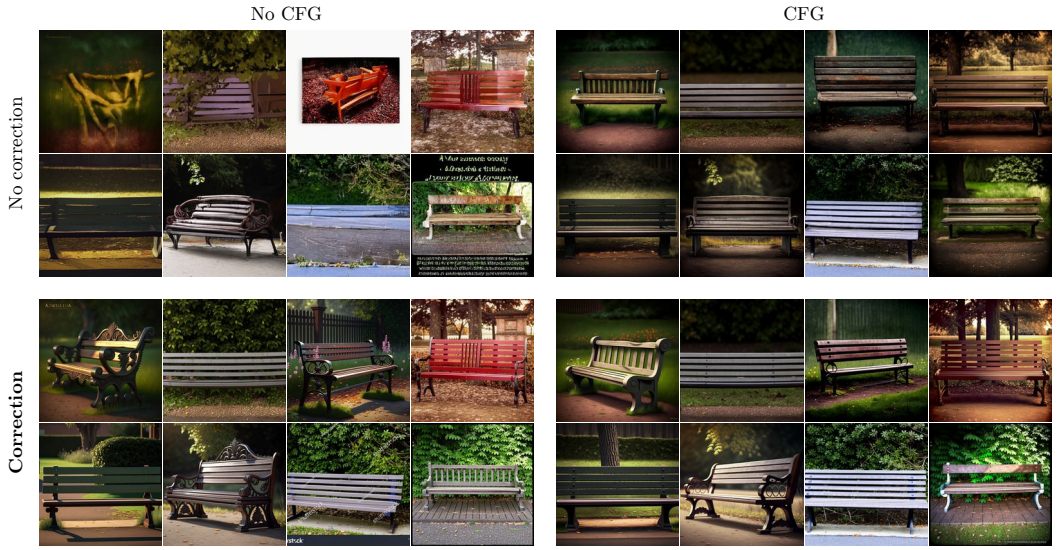


Figure 11: **Correction improves the diversity of CFG samples while preserving quality.** Columns compare no guidance ($w = 1$) and CFG ($w = 6$); rows compare no correction ($K = 0$) and correction ($K = 6$). All settings use the same eight seeds and number of denoising steps. Corrected samples show more varied bench shapes and viewpoints, with less oversaturation and a more natural appearance. Prompt: "a photo of a bench".

Unconditional generation on web images. Using the same model, we also present *unconditional* generation results in Appendix E.6. Surprisingly, correction improves realism while preserving both the diversity of web images and the overall structure of the corresponding uncorrected samples.

Human evaluation. To assess sample quality and diversity through human judgments, we conduct a preference study (Tab. 4) using 80 GenEval single-object prompts on the Rapidata platform[1]. Raters answer "Which image has better visual quality?" for individual images and "Which group is more diverse? Ignore image quality." for four-image groups. Raters prefer CFG + Correction over the baseline using DiffusionBench's default CFG scale (Leng et al., 2026) in more than 60% of responses. Evaluation details are provided in Appendix C.3.

Method	Quality ↑	Diversity ↑
Baseline	39.25%	39.75%
Correction	**60.75%**	**60.25%**

Table 4: **Correction is preferred for quality and diversity.** Released CFG ($w = 6$) versus CFG followed by correction ($w = 4$, $K = 8$).

3.3 Making correction efficient

Parameter-efficient correction through weight sharing. The denoiser and corrector can share a backbone while retaining separate lightweight input embedders, which contribute only a small fraction of the total parameters. On SiT-XL/2, this configuration achieves performance comparable to separate networks with roughly half the total parameters (Tab. 5). In the shared configuration, the backbone receives gradients from both the denoising and correction losses.

We also compare against a larger single denoiser with roughly matched parameter count, and find that iterative correction yields substantially larger gains than simply doubling the denoiser depth (Sec. 4.1, Tab. 7).

Model	Params	FID ↓	K	NFE
Baseline	0.68B	8.55	0	250
Separate corrector	1.35B	**1.92**	2	249
Shared corrector	0.68B	4.47	1	250
Shared corrector	0.68B	2.14	3	248

Table 5: **A shared backbone retains similar correction quality.** Using shared layers from SiT-XL/2 on ImageNet with 100K steps leads to similar FID, suggesting that correction's advantage does not primarily come from using more parameters.

Model	Params	FID ↓	K	Steps	NFE ↓
Base	0.68B	8.62	–	250	250
Teacher	1.35B	**1.92**	2	83	249
Student	0.68B	2.04	–	250	250
Student	0.68B	2.26	–	83	**83**

Table 6: **Distillation achieves similar FID under much smaller NFE.** Distilling a SiT-XL/2 correction model into the base model leads to a similar FID performance with 3 times less NFE on ImageNet.

Distilling correction into the base model. Weight sharing reduces parameter overhead, but each correction iteration still requires an additional forward pass. We therefore distill corrected predictions into the original denoiser. The student is initialized from the teacher's base denoiser and trained to match its predictions after k^* correction iterations:

$$\mathcal{L}_{\text{distill}} = w_{\text{distill}}(t)\|D_{\theta'}(\boldsymbol{z}_t, t) - \hat{x}^{(k^*)}\|_2^2, \tag{6}$$

where k^* controls the number of teacher correction iterations. Tab. 6 reports results for class-conditional ImageNet-256 generation using SiT-XL/2 with $k^* = 2$. The student achieves comparable FID while reducing NFE by approximately 67%. We use $w_{\text{distill}}(t) = 1/t^2$ for $t > 0$, corresponding to velocity-space MSE.

4 Discussion

In this section, we investigate where the quality gains from correction might come from. We first compare a denoiser–corrector pair with a single deeper denoiser containing the same total number of transformer layers (Sec. 4.1). We then empirically examine the behavior of iterative correction and its relationship to the training–inference gap (Sec. 4.2). Finally, we examine how the number of corrections seen during training and denoiser updates affect the quality gains from iterative correction (Sec. 4.3).

[1] https://www.rapidata.ai/

4.1 PARAMETER-MATCHED BASELINES

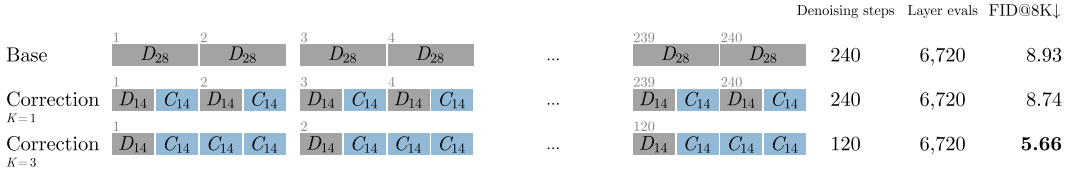


Figure 12: **Matched parameters.** Repeated correction outperforms a deeper denoiser at a comparable parameter count and matched inference budget.

A single correction achieves similar FID to a monolithic denoiser. A natural question is whether a denoiser–corrector pair with a single correction ($K = 1$) can outperform a monolithic denoiser with the same total number of transformer layers (Fig. 12, first and second rows). We also use the same number of denoising steps for both configurations. More specifically, we compare a 28-layer denoiser (D_{28}) with a 14-layer denoiser and a 14-layer corrector ($D_{14} + C_{14}$), both trained from scratch for 80K updates using LightningDiT (Yao et al., 2025) on ImageNet 256×256 (Russakovsky et al., 2015). The two configurations have approximately the same parameter count (675M and 681M, respectively). With a single correction at each denoising step ($K = 1$), the denoiser–corrector pair achieves similar FID to the monolithic denoiser (8.74 versus 8.93). Thus, splitting the model into a denoiser and a corrector gives little quality gain when trained from scratch.

Iterative correction outperforms a monolithic denoiser. We then reuse the same corrector for $K = 3$ iterations at each denoising step (Fig. 12, third row). This lowers FID to 5.66, while halving the number of denoising steps keeps the total transformer-layer evaluation budget unchanged (Tab. 7). These results suggest that *iterative* correction makes effective use of inference compute.

Even when we match the effective depth within each denoising step, the denoiser–corrector pair still performs better. The shallow $D_7 + C_7$ pair with $K = 3$ achieves lower FID than D_{28} (4.21 versus 4.88), using the same number of denoising steps and roughly half as many parameters (Tab. 7. See Fig. 31 in Appendix). These results are from training from scratch. When starting from a pretrained model, as in our main experiments in Sec. 3, a single correction can outperform a deeper monolithic denoiser and converges much faster (Fig. 30 in the Appendix).

Model	Params (M)	Sampling	Steps	Layer evals	FID@8K↓	FID@50K↓
Matched parameters						
D_{28}	675	Unguided	240	6,720	8.93	–
D_{28}	675	CFG ($w = 5$)	120	6,720	6.17	–
D_{14} + C_{14}	681	$K = 1$	240	6,720	8.74	–
D_{14} + C_{14}	681	$K = 3$	120	6,720	5.66	–
D_{14} + C_{14}	681	$K = 4$	96	6,720	**5.56**	–
Matched effective depth						
D_{28}	675	Unguided	240	6,720	8.19	4.88
D_7 + C_7	346	$K = 3$	240	6,720	**7.51**	**4.21**

Table 7: **Iterative correction outperforms matched baselines.** Although a same-layer denoiser-corrector pair achieves similar performance with a large monolithic denoiser with a single correction (D_{14} + C_{14}, $K = 1$), iterative correction (D_{14} + C_{14}, $K = 3$) outperforms the monolithic baseline. Even the smaller $D_7 + C_7$ pair outperforms the larger monolithic baseline with $K = 3$. We select the best CFG scale from $w \in \{1, \ldots, 10\}$.

4.2 HOW ITERATIVE CORRECTION CHANGES PREDICTIONS AND SAMPLING

Iterative correction sharpens the denoiser's predictions. The corrector is trained to reconstruct the clean target from the denoiser's prediction. Its output is another clean-image prediction, allowing the same mapping to be applied repeatedly at a fixed noise level. We examine how this repetition changes the predictions. Fig. 13 shows this visually. We use noisy data $\boldsymbol{z}_t = (1 - t)\boldsymbol{x} + t\boldsymbol{\epsilon}$ as input to JiT-L/16 and show its predictions before and after correction. As the correction count K increases, object structures become increasingly coherent, producing realistic-looking, globally

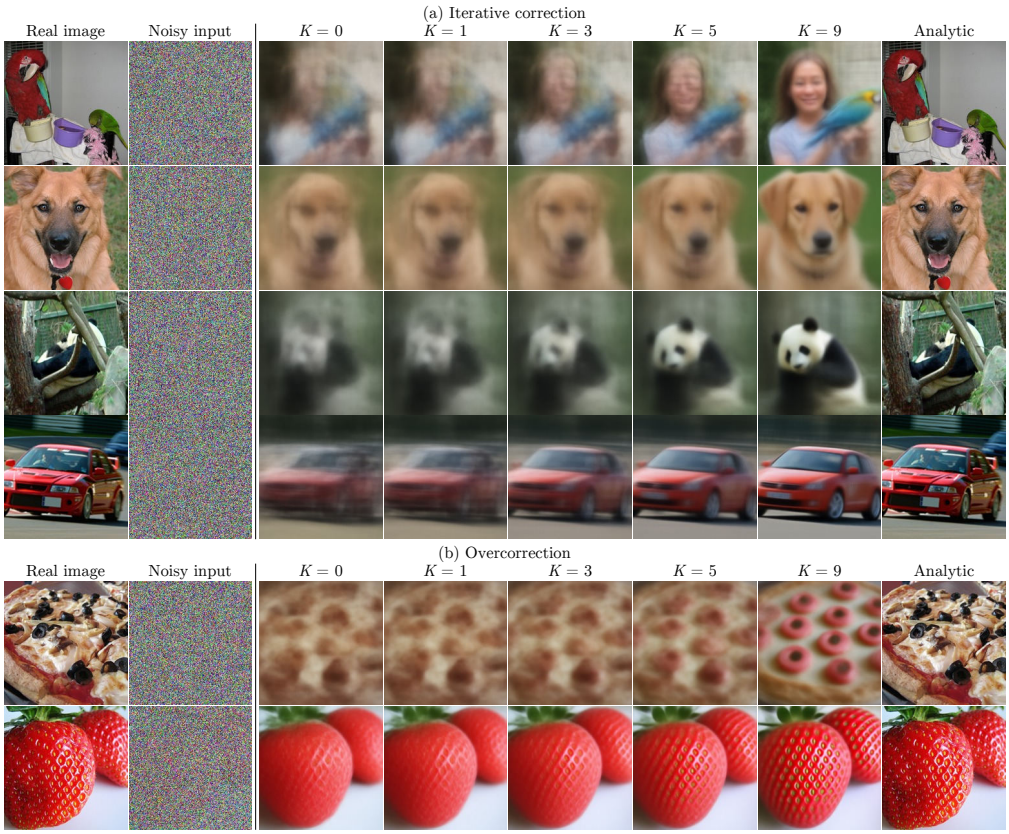


Figure 13: **Iterative correction sharpens predictions.** We show the effect of iterative correction on JiT-L/16. **(a)** The base denoiser's predictions ($K = 0$) are often incoherent and contain artifacts. Repeated correction resolves ambiguous structures and makes faces, object contours, and other details more coherent. **(b)** In extreme cases, correction appears to push this coherence too far, turning ambiguous structures in the base prediction into overly regular geometric shapes. Further selected examples appear in Appendix D.2. Analytic target is computed using Eq. 10 in Appendix.

coherent images at $K = 9$. For example, in (a), the blurry, "average-like" predictions of dogs, pandas, and cars become distinct, well-formed objects. In (b), the corrector sometimes makes overly coherent structures, such as disk-like pizza toppings or strawberry seeds arranged in a regular grid.

In our from-scratch experiments, a single correction yields similar FID to a deeper, parameter-matched denoiser (Sec. 4.1). However, repeated application of the same corrector substantially improves sample quality at a matched inference budget. Our intuition is that splitting the computation into denoising and correction exposes a refinement operation that can be reused. The denoiser first maps the noisy input into a clean-image prediction, and the corrector learns to refine this prediction toward the original clean target. A single correction may provide refinement similar to that achieved by additional denoiser layers. Repeated correction, however, reapplies this learned mapping to its own outputs, beyond the single iteration used during training. This may reinforce coherent structures, sometimes excessively, producing overly regular patterns such as the disks and grids.

Does iterative correction reduce denoising error? Iterative correction sharpens predictions, but the corrected outputs can differ substantially from the original clean image (Fig. 13). Thus, we analyze whether these visual improvements also reduce denoising error. We denote the denoising error after correction by

$$\mathcal{L}^{(k)}_{\text{denoise}} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{\epsilon},t}\left[\|\hat{\boldsymbol{x}}^{(k)} - \boldsymbol{x}\|_2^2\right], \tag{7}$$

where k is the number of correction iterations.

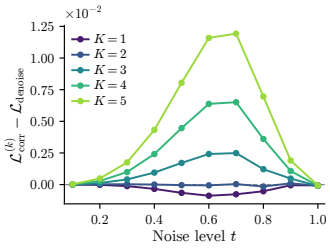


Figure 14: **Iterative correction can increase denoising error.**

We compare this error with the base denoiser's error (Eq. 2) by measuring $\mathcal{L}^{(k)}_{\text{denoise}} - \mathcal{L}_{\text{denoise}}$. Fig. 14 shows positive differences for $k \geq 3$, meaning that the

denoising error *increases* after correction. Thus, even as predictions become sharper and more coherent, iterative correction can move them farther on average from the optimal denoising prediction.

Iterative correction reduces the training–inference gap. How can iterative correction improve sample quality even when it increases denoising error? Denoising error is measured on noisy inputs drawn from the *training distribution*, whereas intermediate states encountered during sampling can follow a different distribution (Song et al., 2025). Each corrected prediction is used to construct the next noisy state, which becomes the denoiser's input. Thus, we hypothesize that correction may improve generation by bringing these states closer to the training distribution. We measure the training–inference gap using a classifier two-sample test (Lopez-Paz & Oquab, 2016). We train classifiers to distinguish intermediate states $\boldsymbol{z}_t$ from the training distribution, $\boldsymbol{z}_t = (1-t)\boldsymbol{x} + t\boldsymbol{\epsilon}$, from those along denoising trajectories at inference. Specifically, we experiment with SiT (Ma et al., 2024) on 100K intermediate states, and report binary cross-entropy (BCE) and AUROC on held-out datasets. A smaller training-inference gap should make the two distributions harder to distinguish, bringing AUROC closer to 0.5 (random chance).

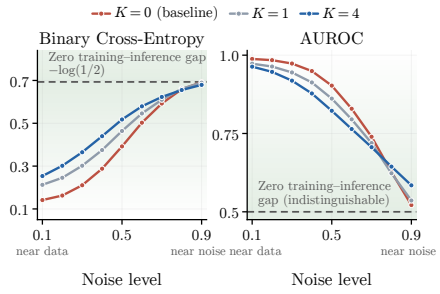


Figure 15: **Measuring the training–inference gap with classifier two-sample tests.** We train a classifier to distinguish samples from the training distribution and the inference distribution, and report AUROC and binary cross-entropy (BCE). Overall, correction makes classification harder, indicating a smaller gap between training and inference.

As shown by the BCE in Fig. 15, the classifier can distinguish training and inference states, with classification becoming easier as the noise level decreases. This trend is consistent with errors accumulating during sampling due to exposure bias (Ning et al., 2024; 2023). Notably, a single correction (gray lines) yields higher BCE and lower AUROC, suggesting that correction *reduces the training–inference gap*. Iterative correction generally reduces the gap further (blue lines), which may contribute to better sample quality even when reconstruction error on training inputs increases.

4.3 How training affects iterative correction

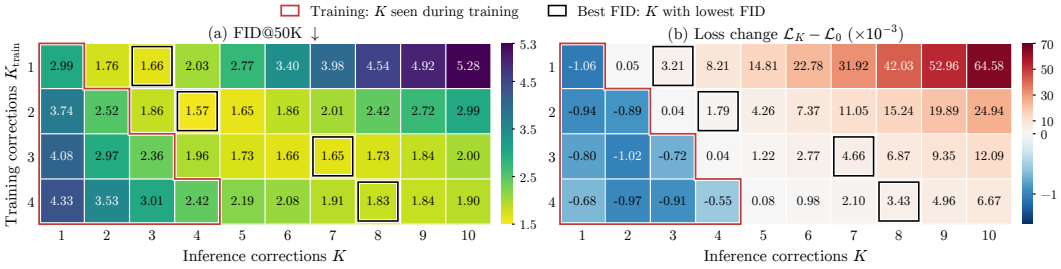


Figure 16: **Correction beyond the supervised training depth improves generation (lower FID).** Supervising more correction steps shifts the best-performing tested inference correction count K toward larger values, often approximately twice the maximum supervised depth. This suggests that "extrapolating" the learned correction beyond its training inputs may play an important role in the observed gains.

Extrapolation is key to substantial quality gains. We denote the number of correction iterations applied during training by K_{train}. In our default setting ($K_{\text{train}} = 1$), the corrector sees only the denoiser's initial prediction during training. We also train the corrector to predict the target after multiple correction iterations ($K_{\text{train}} > 1$). Increasing K_{train} shifts the best-performing inference correction count K toward larger values, while the best achieved FID remains similar (Fig. 16, left). Notably, **the best performance comes from applying correction more times than supervised during training**. The right panel shows that increasing K_{train} lowers denoising error below the base denoiser's only for correction iterations included in training. For $K > K_{\text{train}}$, the error remains above this baseline, even at the correction count with the best FID.

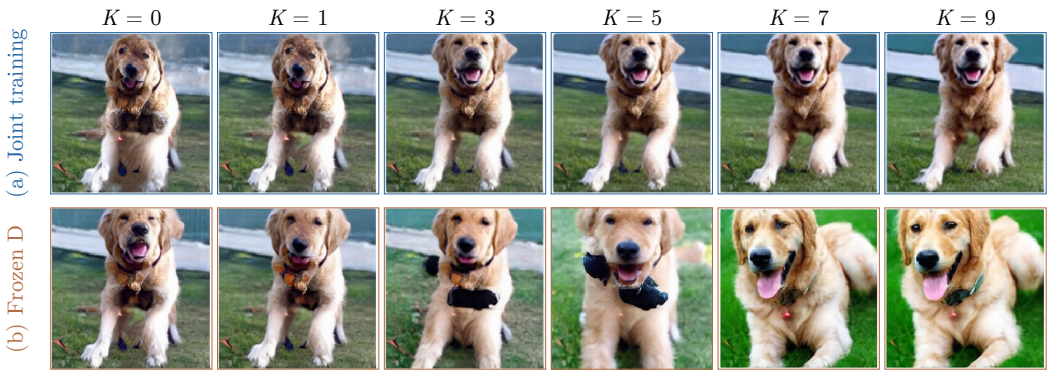


Figure 17: **A frozen denoiser can support high quality refinement, but iterative correction can also oversaturate samples and produce abrupt changes in the sample prediction.**

Our intuition is that the corrector learns a mapping toward the optimal denoising prediction. Supervising more correction iterations may simply spread this correction over more steps, so more inference iterations are needed to obtain similar gains from *extrapolating* the learned mapping beyond the iterations used during training.

Denoiser updates affect iterative correction. To examine how denoiser updates affect iterative correction, we compare (a) jointly training the denoiser and corrector, our default setting in Algorithm 1, with (b) freezing the denoiser and training only the corrector under the same correction framework. Interestingly, as shown in Fig. 17, samples in (b) improve with correction but change more abruptly in structure and appearance, with slight oversaturation typical of CFG. Fig. 18 shows similar FID for both settings at $K = 5$. With further correction, FID continues to decrease with joint training but increases again with a frozen denoiser despite visually improved samples, similar to the trend at higher CFG scales. One possible interpretation is that, in (b), the corrector specializes in the errors of a particular denoiser checkpoint, making it less robust to iterative application. In (a), the denoiser continues to evolve during training, exposing the corrector to a broader range of errors. This may help the corrector generalize to its own outputs and remain effective across different correction counts K.

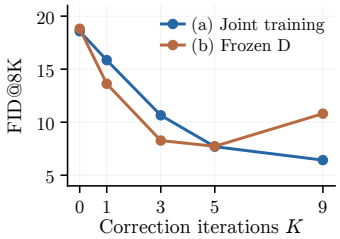


Figure 18: **Frozen-denoiser correction is competitive at small K but deteriorates earlier.**

5 Related work

Stacked generalization (Wolpert, 1992) trains a higher-level predictor on held-out predictions of base models to improve generalization. In diffusion models, self-conditioning (Chen et al., 2022) uses previous clean-data predictions as additional denoiser inputs, and recent concurrent works extend this idea to iterative refinement in language modeling and structured reasoning (Yoo et al., 2026; Helbling et al., 2026). We train an explicit corrector on denoiser predictions and reuse it across correction iterations, without requiring the original noisy input. Idempotent Generative Networks (Shocher et al., 2024) also learn iterative mappings through an explicit idempotency objective. Our method retains the denoising objective and applies correction at each noise level to refine predictions before the sampling update. The most relevant work to ours is Schiff et al. (2026), which trains a corrector on generated samples and applies it repeatedly during sampling in discrete diffusion for language modeling.

6 Conclusion

We introduce correction training, which learns to correct a diffusion model's predictions using the original clean-data targets. Despite being trained for a single correction, the corrector supports repeated refinement at inference. Experiments on class-conditional and unconditional ImageNet

generation and text-to-image generation demonstrate substantial improvements in sample quality. Correction can achieve competitive quality with guidance while better preserving diversity. Correction can also be distilled to remove inference overhead. Our analysis offers empirical insights into why correction works, but a theoretical understanding of its effectiveness and iterative behavior remains an open question for future research.

ACKNOWLEDGMENTS

We would like to thank Konpat Preechakul for valuable insights and feedback, as well as Michael Gharbi for insightful discussions, and Nicolas Dufour for providing detailed feedback on the manuscript. We are grateful to A. Sophia Koepke, Anne Harrington, Amil Dravid, Jorge Diaz Chao, Isabella Yu, Yutong Bai, Yong-Hyun Park, Sangyoon Lee, Sihyun Yu, Susung Hong, Joonghyuk Shin, Gyuseong Lee, and Sanghyun Lee for helpful discussions. This work was supported, in part, by NSF IIS-2403305 and ONR MURI grants.

REFERENCES

Donghoon Ahn, Hyoungwon Cho, Jaewon Min, Wooseok Jang, Jungwoo Kim, SeonHwa Kim, Hyun Hee Park, Kyong Hwan Jin, and Seungryong Kim. Self-rectifying diffusion sampling with perturbed-attention guidance. In *European Conference on Computer Vision*, pp. 1–17. Springer, 2024.

Michael Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025.

Jiuhai Chen, Zhiyang Xu, Xichen Pan, Yushi Hu, Can Qin, Tom Goldstein, Lifu Huang, Tianyi Zhou, Saining Xie, Silvio Savarese, et al. Blip3-o: A family of fully open unified multimodal models-architecture, training and dataset. *arXiv preprint arXiv:2505.09568*, 2025.

Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*, 2022.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36: 52132–52152, 2023.

Alec Helbling, Andrey Bryutkin, Mauro Martino, Nima Dehmamy, and Hendrik Strobelt. Flow reasoning models: Scaling reasoning through iterative self-refinement. *arXiv preprint arXiv:2606.29150*, 2026.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. Ella: Equip diffusion models with llm for enhanced semantic alignment. *arXiv preprint arXiv:2403.05135*, 2024.

Tero Karras, Miika Aittala, Tuomas Kynkäänniemi, Jaakko Lehtinen, Timo Aila, and Samuli Laine. Guiding a diffusion model with a bad version of itself. *Advances in Neural Information Processing Systems*, 37:52996–53021, 2024.

Yuya Kobayashi, Masato Ishii, Yuhta Takida, Takashi Shibuya, and Yuki Mitsufuji. Spectral prior for reducing exposure bias in diffusion models. *arXiv preprint arXiv:2607.22091*, 2026.

Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. *Advances in neural information processing systems*, 32, 2019.

Black Forest Labs. FLUX.2: Frontier Visual Intelligence. https://bfl.ai/blog/flux-2, 2025.

Xingjian Leng, Jaskirat Singh, Zhanhao Liang, Ethan Smith, Martin Bell, Aninda Saha, Yuhui Yuan, and Liang Zheng. Diffusionbench: On holistic evaluation of diffusion transformers. *arXiv preprint arXiv:2606.24888*, 2026.

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 *International Conference on Learning Representations*, volume 2024, pp. 16816–16838, 2024a.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 36115–36125, 2026.

Tianhong Li, Dina Katabi, and Kaiming He. Return of unconditional generation: A self-supervised representation generation method. *Advances in Neural Information Processing Systems*, 37: 125441–125468, 2024b.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

David Lopez-Paz and Maxime Oquab. Revisiting classifier two-sample tests. *arXiv preprint arXiv:1610.06545*, 2016.

Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, pp. 23–40. Springer, 2024.

Mang Ning, Enver Sangineto, Angelo Porrello, Simone Calderara, and Rita Cucchiara. Input perturbation reduces exposure bias in diffusion models. *arXiv preprint arXiv:2301.11706*, 2023.

Mang Ning, Mingxiao Li, Jianlin Su, Albert Ali Salah, and Itir Onal Ertugrul. Elucidating the exposure bias in diffusion models. In *International Conference on Learning Representations*, volume 2024, pp. 15167–15189, 2024.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Seyedmorteza Sadat, Otmar Hilliges, and Romann M Weber. Eliminating oversaturation and artifacts of high guidance scales in diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2024.

Yair Schiff, Omer Belhasin, Roy Uziel, Guanghan Wang, Marianne Arriola, Gilad Turok, Ran Zilberstein, Michael Elad, and Volodymyr Kuleshov. Learn from your mistakes: Self-correcting masked diffusion models. *arXiv preprint arXiv:2602.11590*, 2026.

Assaf Shocher, Amil Dravid, Yossi Gandelsman, Inbar Mosseri, Michael Rubinstein, and Alexei Efros. Idempotent generative network. In *International Conference on Learning Representations*, volume 2024, pp. 35440–35460, 2024.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. pmlr, 2015.

Kiwhan Song, Jaeyeon Kim, Sitan Chen, Yilun Du, Sham Kakade, and Vincent Sitzmann. Selective underfitting in diffusion models. *arXiv preprint arXiv:2510.01378*, 2025.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Keqiang Sun, Junting Pan, Yuying Ge, Hao Li, Haodong Duan, Xiaoshi Wu, Renrui Zhang, Aojun Zhou, Zipeng Qin, Yi Wang, et al. Journeydb: A benchmark for generative image understanding. *Advances in neural information processing systems*, 36:49659–49678, 2023.

Shuai Wang, Zhi Tian, Weilin Huang, and Limin Wang. Ddt: Decoupled diffusion transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 40633–40642, 2026.

David H Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15703–15712. IEEE, 2025.

Jaehoon Yoo, Wonjung Kim, Floor Eijkelboom, Chanhyuk Lee, Nicholas M Boffi, Seunghoon Hong, and Jinwoo Kim. Self-conditioned flow map language models via fixed-point flows. *arXiv preprint arXiv:2607.00714*, 2026.

Meng Yu and Kun Zhan. Frequency regulation for exposure bias mitigation in diffusion models. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 10370–10378, 2025.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. *arXiv preprint arXiv:2410.06940*, 2024.

APPENDIX CONTENTS

A Design choices and alternative realizations **18**

A.1 Corrector interface 18

A.2 Training configuration 20

A.3 Inference configuration 20

B Implementation Details **23**

B.1 Training configurations 23

B.2 Corrector parameterization 25

B.3 Figure and Table Details 27

C Additional Results **28**

C.1 Comparing Correction and CFG 28

C.2 Combining CFG and Correction 33

C.3 Human evaluation 33

D Discussion **35**

D.1 Training and capacity controls 35

D.2 Iterative correction visualizations 36

D.3 Power spectrum analysis 41

E Additional Samples **41**

E.1 Effect of Correction Depth 41

E.2 SiT: Random-Class Generation 43

E.3 SiT_{REPA}: Random-Class Generation 44

E.4 JiT: Random-Class Generation 45

E.5 Unconditional Generation 46

E.6 Unconditional Generation from a Text-to-Image Model 47

A Design choices and alternative realizations

A practical realization of correction training. In this section, we examine the design choices in our framework. The default configuration used in the main paper is a simple implementation compatible with pretrained models: we use a separate corrector initialized from the base denoiser, omit the noisy input $\boldsymbol{z}_t$, retain the timestep-conditioning pathway to preserve the original architecture, and apply stop-gradient to prevent the correction loss from updating the denoiser. Many other combinations of these design choices are possible within our correction training framework. Instead of exhaustively evaluating all combinations, we vary one component at a time from the default configuration to understand how each choice affects performance.

A.1 Corrector interface

Access to the noisy state $\boldsymbol{z}_t$ from corrector. Prior approaches (Chen et al., 2022) to learning from model predictions often also provide the original input. In our diffusion setting, this corresponds to giving the corrector access to $\boldsymbol{z}_t$ in addition to the denoiser's prediction. Tab. 8 shows that, on JiT, correction without $\boldsymbol{z}_t$ achieves comparable FID to correction with $\boldsymbol{z}_t$, while requiring no architectural changes to the denoiser.

The SiT comparison in Fig. 19 further shows how this choice affects the precision–recall trade-off. Here, p-only denotes conditioning on the proposal $\hat{\boldsymbol{x}}$ alone, whereas $p + \boldsymbol{z}_t$ provides both the proposal and the noisy input. Proposal-only correction achieves higher precision at slightly lower recall. One possible explanation is that, without access to the original noisy input, the corrector is less constrained by it and can modify the proposal more aggressively.

Variant	K=0	K=1	K=2	K=3	K=4	K=5
Proposal only ($\hat{x}$)	24.63	17.96	14.07	**13.35**	14.27	16.06
With $\boldsymbol{z}_t$ conditioning	24.72	18.22	14.84	13.79	14.67	16.58

Table 8: **Proposal-only correction performs comparably to $\boldsymbol{z}_t$-conditioned correction.** Unguided FID50K on JiT-B/16 at approximately 250 NFE.

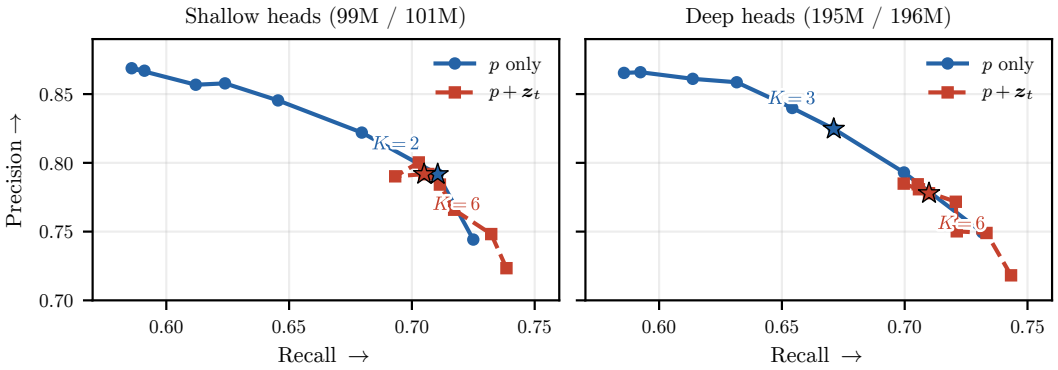


Figure 19: **Proposal-only correction reaches higher peak precision.** SiT on ImageNet with approximately capacity-matched correction heads, sweeping $K = 1$–8. Stars mark each arm's lowest-FID setting.

Timestep conditioning. By default, we provide the true timestep to reuse the pretrained model's timestep-conditioning pathway. However, Tab. 9 shows that using the true timestep or a constant time embedding yields similar final FID in our residual-correction setting. This finding may not generalize to other settings, including correctors trained from scratch: the pretrained pathway associates t with the noise level of the original noisy input, which does not directly describe the errors in the clean-sample prediction received by the corrector.

Prediction-space versus hidden-state refinement. We compare correction in output space (pixels or latents) with refinement of the network's hidden states. Output-space correction benefits from

Corrector time input	FID ↓	Precision ↑	Recall ↑
True timestep	2.825	0.763	0.721
Constant time embedding	2.895	0.745	0.734

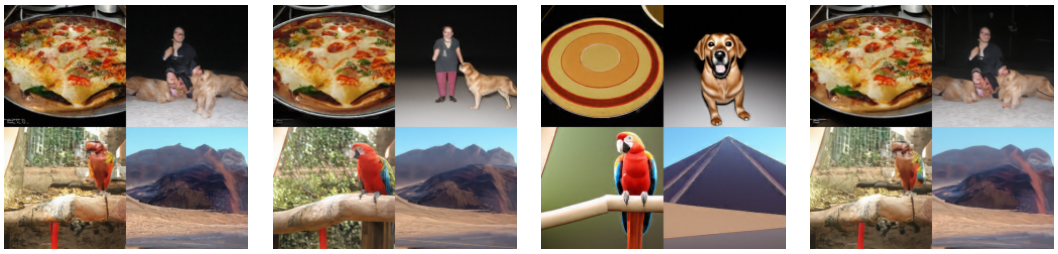
Table 9: **Explicit time conditioning has little effect on FID in this setting.** We compare true and fixed timestep inputs to the corrector on SiT-REPA at $K = 1$, using FID-50K at 250 NFE. The denoiser uses the true timestep in both cases.

repeated application, whereas the hidden-state variants do not achieve comparable gains (Tab. 10). Increasing the number of correction iterations during training does not improve sample quality in either setting. Fig. 20 presents qualitative comparisons at $K = 5$.

On the other hand, removing stop-gradient from the base denoiser's prediction achieves a lower best FID, but retaining it yields more consistent gains at larger K. Without stop-gradient, samples become visually simpler under repeated correction, as shown in Fig. 20(b).

Stop-gradient	K_{train}	Inference-time iterations K					
		0	1	2	3	4	5
Correcting model predictions $\hat{x}$							
✓	1	24.63	17.96	14.07	13.35	14.27	16.06
✗	1	29.08	17.38	12.88	19.25	24.64	28.92
✓	1–5	24.66	21.02	21.68	22.51	23.30	24.67
Iterating over hidden states h							
✓	1	24.63	24.78	35.73	102.84	261.61	339.93
✗	1	24.19	24.63	39.85	104.94	233.81	326.03
✓	1–5	24.66	24.99	25.01	25.12	25.35	25.47

Table 10: **Output-space iteration enables effective iterative correction.** Component ablations on JiT-B/16, evaluated by unguided FID50K at approximately 250 NFE.



baseline (no correction) (a) ours (b) no stop-grad (c) multi-iter train



(d) hidden space (e) hidden, no stop-grad (f) hidden, multi-iter

Figure 20: **Output-space correction remains effective under repeated application.** Qualitative comparisons of the component ablations in Tab. 10 at $K = 5$, using the same initial noise and class labels.

A.2 Training configuration

Stop-gradient versus end-to-end optimization. We further evaluate the effect of stop-gradient on additional pretrained models (Fig. 21). Unlike the earlier comparison in Table 10, the variants achieve similar best FID here. Full stop-gradient (our default) nevertheless leads to less degradation at larger correction counts K.

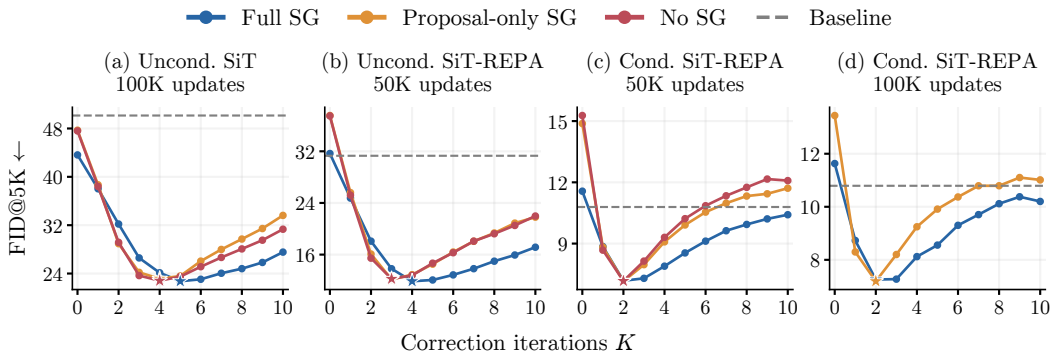


Figure 21: **Full stop-gradient degrades more slowly at large correction counts in these comparisons.** FID@5K versus K, using EMA weights, the full correction interval, and at most 250 NFE. SG denotes stop-gradient; full SG blocks both proposal and base-combination paths.

A.3 Inference configuration

Correction count K. Increasing K improves FID up to a setting-dependent optimum, after which further correction can degrade it (Tab. 10; Fig. 21).

Correction interval. Restricting JiT correction to early steps and reallocating the saved calls to denoising improves FID at a fixed sampling budget (Tab. 11).

	End r_{end}				
Start r_{start}	0.8	0.6	0.4	0.2	0.0
1.0	7.70	**6.79**	7.72	9.05	11.85[full]
0.8	—	11.16	11.12	12.59	13.76
0.6	—	—	14.54	15.09	17.94
0.4	—	—	—	17.10	20.05
0.2	—	—	—	—	19.96

[full]Full trajectory ($1 \to 0$).

Table 11: **Allocating correction to early steps yields the lowest FID at a fixed sampling budget.** JiT-L/16 with $K = 5$, evaluated by FID8K. Rows and columns specify the start and end noise levels, decreasing from noise to data.

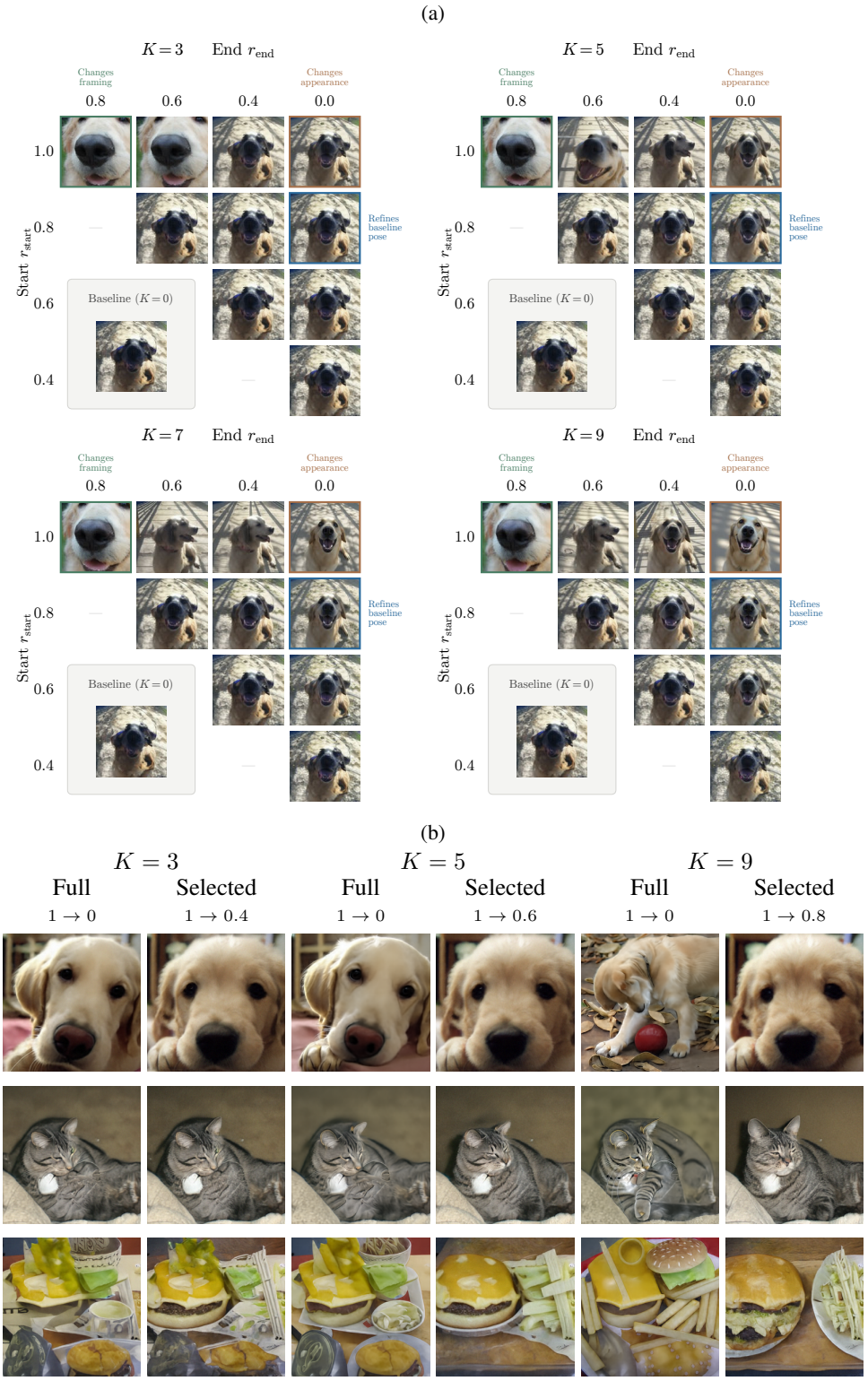


Figure 22: **Skipping early correction refines the baseline pose.** Gray boxes show the matched $K = 0$ baseline; green, blue and orange mark early-only, skip-early and full-range correction, respectively.

Restricting correction to early steps changes the generated structure. Matched-noise comparisons of full-trajectory correction and the interval selected by FID8K for each K.

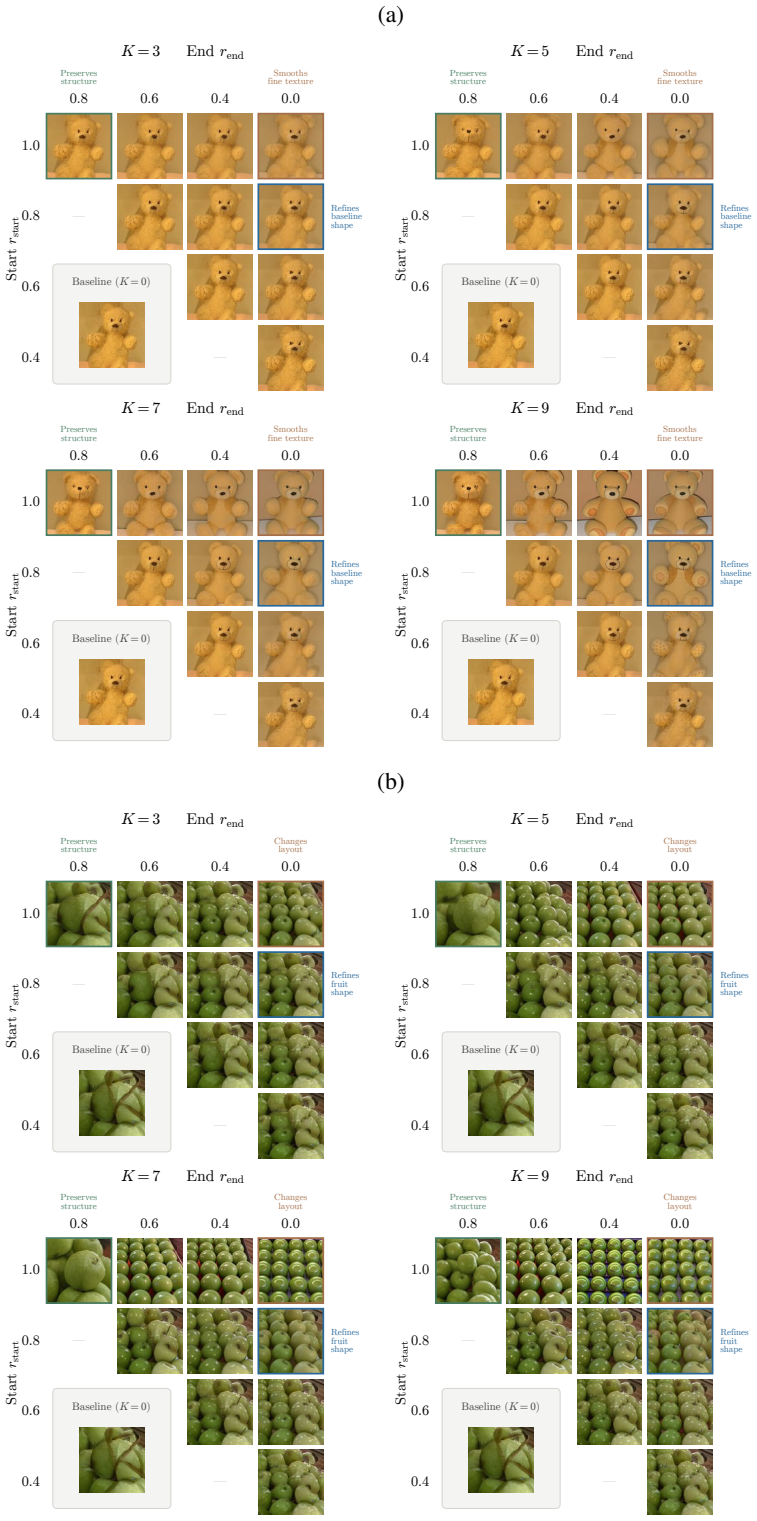


Figure 23: **(a) Full-range correction changes fine texture and shape. (b) Full-range correction changes object layout and appearance.** Gray boxes show the matched $K = 0$ baseline; green, blue and orange mark early-only, skip-early and full-range correction, respectively.

B Implementation Details

B.1 Training configurations

Common backbone settings. We retain each pretrained model's backbone and original conditioning. The separate corrector uses an independent copy of the denoiser backbone; latent-space models retain their original VAE, while JiT operates directly on RGB images. Training and quantitative sampling settings are summarized in Tables 12 and 13.

Table 12: **Iterative refinement uses correctors trained with a single correction pass.** Training settings for the main ImageNet and quantitative text-to-image comparisons. We follow each model's original training recipe as closely as possible. Updates count additional training, not pretraining; D and C denote separate denoiser and corrector networks. Sampling settings are reported separately.

Setting	DiT-XL/2	SiT-XL/2	SiT_{REPA}-XL/2[†]	JiT-L/16	T2I[‡] (Sec. 3.2)
Data and initialization					
Training data	ImageNet	ImageNet	ImageNet	ImageNet	BLIP3o mixture
Image resolution			256×256		
Denoising space	SD-VAE latents	SD-VAE latents	SD-VAE latents	RGB pixels	FLUX.2-VAE latents
Denoiser init.	Released	Released	Released	Released	202K base
Corrector init.			Independent copy of the pretrained denoiser		
Additional updates	100K	100K	50K	100K	50K
Optimization					
Global batch size	256	256	256	1,024	1,024
Optimizer			AdamW; weight decay 0		
Adam (β_1, β_2)	$(0.9, 0.999)$	$(0.9, 0.999)$	$(0.9, 0.999)$	$(0.9, 0.95)$	$(0.9, 0.95)$
Denoiser LR	10^{-4}	10^{-4}	10^{-4}	5×10^{-5}	Linear decay
Corrector LR	10^{-4}	10^{-4}	10^{-4}	5×10^{-5}	10^{-4}
Warmup	None	None	100 updates	5 epochs	No new warmup
LR after warmup	Constant	Constant	Constant	Constant	D: linear; C: const.
EMA decay	0.9999	0.9999	0.9999	0.9999/0.9996	0.9995
Training objective and correction					
Prediction (D/C)	ϵ/ϵ	v/v	v/v	x/x	v/v
Corrector image input	Detached clean-sample prediction; original timestep and conditioning retained				
Output combination	Gated ϵ	Gated v	Gated v	Direct x	Gated v
Gate initialization	0	0	0	—	0
Correction passes	$K_{\text{train}} = 1$				
Loss weights $(D : C)$	$1 : 1$; correction loss has no gradient path into the separate denoiser				

[†] SiT_{REPA} is initialized from the publicly released SiT-XL/2 checkpoint trained with the REPA loss for 4M iterations; the REPA loss is not used during additional training.

[‡] T2I denotes text-to-image generation (Sec. 3.2). We use the DiffusionBench codebase and its T2I checkpoint, trained in the FLUX.2 VAE latent space using velocity prediction.

Table 13: **The main quantitative comparisons use approximately 250 network evaluations per image.** Sampling settings for Tables 1 and 3. K is the number of corrections per active step; intervals use noise time t (1: noise, 0: data). NFE includes correction and both CFG branches. Guided T2I applies correction before CFG.

Method	CFG w	K	CFG interval	Correction interval	Steps	NFE
DiT-XL/2; DDPM						
Baseline	1	0	—	—	250	250
CFG	1.5	0	$1 \to 0$	—	125	250
Correction	1	2	—	$1 \to 0$	83	249
SiT-XL/2; Euler–Maruyama SDE						
Baseline	1	0	—	—	250	250
CFG	1.5	0	$1 \to 0$	—	125	250
Correction	1	2	—	$1 \to 0$	83	249
SiT$_{\mathrm{REPA}}$-XL/2; Euler–Maruyama SDE						
Baseline	1	0	—	—	250	250
CFG	1.8	0	$0.7 \to 0$	—	148	250
Correction	1	5	—	$0.7 \to 0.5$	123	248
JiT-L/16; Heun ODE (final Euler step)						
Baseline	1	0	—	—	125	249
CFG	2.4	0	$0 < t < 0.9$	—	63	250
Correction	1	9	—	$1 \to 0.8$	45	242
T2I (DiT + DDT); Euler ODE						
Baseline	1	0	—	—	250	250
Correction	1	6	—	$1 \to 0$	35	245
CFG	4	0	$1 \to 0$	—	125	250
CFG + Correction	4	2	$1 \to 0$	$1 \to 0$	41	246

Table 14: **Unconditional ImageNet settings.** Sampling settings for Table 2, using SiT$_{\mathrm{REPA}}$-XL/2.

Method	**Setting**	**Train steps**	**Denoising steps**	**NFE**
Baseline	–	50K	250	250
Correction	$K = 1$	50K	125	250
Correction	$K = 4$	50K	50	250
Autoguidance	$w = 7$	50K	125	250

With CFG ($w = 2$)			*With CFG* ($w = 6$)		
Method	GenEval ↑	DPG-Bench ↑	Method	GenEval ↑	DPG-Bench ↑
Baseline	54.57	80.11	Baseline	71.70	84.44
Correction ($K = 2$)	61.17	80.82	**Correction** ($K = 2$)	**72.65**	**84.55**
Correction ($K = 4$)	63.61	**81.11**	**Correction** ($K = 4$)	71.52	84.38
Correction ($K = 6$)	**63.74**	80.79	**Correction** ($K = 6$)	69.53	83.42

Table 15: **Correction's alignment gains depend on guidance strength and iteration count.** Additional guidance scales use the same checkpoint as Table 3, but fix 50 Euler steps rather than NFE. Correction is applied before CFG. Bold marks the best score within each block.

B.2 CORRECTOR PARAMETERIZATION

Algorithm 1 uses x-prediction. For an ϵ- or v-predicting denoiser, we recommend the following two practical options.

Direct clean-sample prediction. The simplest starting point is to convert the denoiser's output into a clean-sample proposal and train the corrector to predict a refined clean sample:

$$\hat{\boldsymbol{x}}^{(1)} = C_{\boldsymbol{\phi}}\Big(\mathrm{sg}(\hat{\boldsymbol{x}}^{(0)}), t\Big)\,.$$

The result can then be converted into the representation required by the sampler using $\boldsymbol{z}_t$, t, and the noise schedule. These conversions are performed outside the corrector network; $\boldsymbol{z}_t$ need not be supplied as an additional network input. The training loss can still be computed in velocity or noise space.

Why convert the proposal to x rather than retain the denoiser's original output space? In our small-scale experiments with a single correction ($K = 1$) and without additional noisy-state conditioning, x-space correction performs best among the tested same-space input–output choices, including with ϵ- and v-predicting denoisers. This makes clean-sample input and output a useful starting point, especially when training the corrector from scratch. These experiments do not establish a universally optimal parameterization since it's small-scale run on $K = 1$. Directly correcting noise or velocity remains an option.

Reusing pretrained output representations. When initializing the corrector from a pretrained denoiser, another consideration is how much of its learned representation to retain. For example, SiT (Ma et al., 2024) is trained to predict velocity. Changing its output to x requires adapting the pretrained output representation. Direct x-prediction can still work and converged quickly in our experiments, but retaining the original output parameterization offers another way to preserve the representation of pretrained models. This motivated our use of clean-sample input with native ϵ- or v-output in the main pretrained-model experiments.

Retaining the native output also raises a practical question: how should it be combined with a clean-sample input? An exact conversion between x, ϵ, and v requires $\boldsymbol{z}_t$ or another component in addition to the quantity being converted. Providing $\boldsymbol{z}_t$ as an additional condition is possible, but its empirical benefits were mixed (App. A.1), so we omit it by default. Instead, we retain the base prediction in its native output space and linearly combine it with the corrector's native-space output through a learned gate. This allows us to use a clean-sample input while retaining the pretrained output space, without supplying $\boldsymbol{z}_t$ to the network or requiring an explicit conversion inside it.

Let $\boldsymbol{q}$ denote the target in the chosen output space and $\hat{\boldsymbol{q}}^{(0)}$ the base prediction. Our full stop-gradient formulation is:

$$\hat{\boldsymbol{q}}^{(1)} = \mathrm{sg}(\hat{\boldsymbol{q}}^{(0)}) + \boldsymbol{\gamma} \odot \left[C^{q}_{\boldsymbol{\phi}}\Big(\mathrm{sg}(\hat{\boldsymbol{x}}^{(0)}), t\Big) - \mathrm{sg}(\hat{\boldsymbol{q}}^{(0)})\right], \tag{8}$$

$$\mathcal{L}_{\mathrm{corr}} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{\epsilon},t}\left[\lambda_q(t)\left\|\hat{\boldsymbol{q}}^{(1)} - \boldsymbol{q}\right\|_2^2\right], \tag{9}$$

where $\boldsymbol{q}$ is the target in the chosen output space, such as v for SiT or ϵ for DiT, and $\lambda_q(t)$ preserves the model's loss weighting. The unconstrained gate is initialized to zero, preserving the base prediction at initialization. It is learned per channel, except for DiT and plain SiT, which use a scalar shared across channels. For better performance, one can extend it to timestep-dependent gating or 2D tensor gates, but we keep the (per-channel) scalar for simplicity.

Correction with a fixed base component. Gating is not the only way to combine different input and output parameterizations. We also tested a formulation that combines the corrector's output with a fixed component of the base prediction. Under our convention $v = \epsilon - x$, an x-space prediction is combined with the base noise prediction, while an ϵ-space prediction is combined with the base clean-sample prediction:

$$\hat{\boldsymbol{x}}^{(1)} = \begin{cases} \hat{\boldsymbol{x}}, \\ \hat{\boldsymbol{\epsilon}} - \mathrm{sg}(\hat{\boldsymbol{v}}^{(0)}), \\ \mathrm{sg}(\hat{\boldsymbol{\epsilon}}^{(0)}) - \hat{\boldsymbol{v}}, \end{cases} \qquad \hat{\boldsymbol{\epsilon}}^{(1)} = \begin{cases} \hat{\boldsymbol{x}} + \mathrm{sg}(\hat{\boldsymbol{v}}^{(0)}), \\ \hat{\boldsymbol{\epsilon}}, \\ \mathrm{sg}(\hat{\boldsymbol{x}}^{(0)}) + \hat{\boldsymbol{v}}, \end{cases} \qquad \hat{\boldsymbol{v}}^{(1)} = \begin{cases} \mathrm{sg}(\hat{\boldsymbol{\epsilon}}^{(0)}) - \hat{\boldsymbol{x}}, \\ \hat{\boldsymbol{\epsilon}} - \mathrm{sg}(\hat{\boldsymbol{x}}^{(0)}), \\ \hat{\boldsymbol{v}}. \end{cases}$$

These expressions illustrate alternative fixed-base-component formulations for each output space. Our small-scale experiments used the velocity-space formulation on the right and minimized velocity-space MSE.

Here $\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{\epsilon}}$, and $\hat{\boldsymbol{v}}$ denote the corrector's predictions. The input can independently be chosen from the base clean, noise, or velocity predictions. All combinations minimize velocity-space MSE without a learned gate. For x- and ϵ-space predictions, this objective also requires compensating for errors in the retained base component, so these predictions are not directly supervised against the clean sample or noise alone.

By explicitly using the decomposition $v = \epsilon - x$, this formulation may offer a more principled way to connect different input and output parameterizations than a learned blend. Our exploration was limited by time constraints, however: we tested it only in small-scale experiments with a single correction and did not extensively investigate repeated correction or larger-scale settings.

Gradient paths. The correction loss can reach the denoiser through the corrector's input $\hat{\boldsymbol{x}}^{(0)}$ and, when used, the retained base prediction $\hat{\boldsymbol{q}}^{(0)}$. Our default, **Full stop-gradient (SG)**, detaches both (Eq. 8). For separate networks, the denoiser is then updated only by its denoising loss; it is not frozen. We also compare **Proposal-only SG**, which detaches only the corrector's input, and **No SG**, which detaches neither. Full SG achieves competitive FID and degrades less at large correction counts in our comparisons (App. A.2).

Full-target versus residual prediction. Independently of the input and output spaces, the network can predict either the full target or a residual relative to the base prediction. This choice applies both to direct clean-sample prediction and to native-space correction with gating. Without gating, residual prediction takes the form:

$$\hat{\boldsymbol{q}}^{(1)} = \mathrm{sg}(\hat{\boldsymbol{q}}^{(0)}) + R^q_{\boldsymbol{\phi}}\Big(\mathrm{sg}(\hat{\boldsymbol{x}}^{(0)}), t\Big)\,.$$

With gating, the gate can instead scale the predicted residual. Thus, residual prediction is not an alternative to gating; the two can be used together. The correction objective remains unchanged. Zero-initializing the residual output projection preserves the base prediction at initialization. Although we do not present a dedicated comparison in this paper, residual prediction also worked well in our experiments, sometimes slightly better than full-target prediction. We use full-target prediction in the main formulation for simplicity, but residual prediction is another practical option within correction training.

Choosing a formulation. For a first implementation, particularly when training from scratch, we recommend clean-sample input and direct clean-sample output. When reusing a pretrained denoiser, clean-sample input with gated native-space prediction is another practical starting point. We retained this form in the main pretrained-model experiments for compatibility with the pretrained output representation and its slightly better empirical performance. Neither choice is required by correction training: input space, output space, full-target versus residual prediction, and loss space are distinct design choices. Other combinations can also be used, provided that the information needed for any conversion is available. The network can also take noise or velocity proposals as input; we use clean-sample input in the equations to illustrate our default. Full-target versus residual prediction can be chosen independently.

Design choice	Option	Used in
Output formulation	Direct x	Table 1 — JiT
	Gated native q	Table 1 — DiT, SiT, SiT-REPA; Tables 2–3
Prediction type	Full target	Tables 1–3
	Residual	Table 9 — timestep conditioning

Table 16: **Corrector parameterization choices.** We distinguish the output formulation from full-target versus residual prediction, which can be used with either formulation. Here q denotes the chosen output space. The equations use clean-sample input by default; input parameterization and loss space are separate choices.

B.3 FIGURE AND TABLE DETAILS

Unconditional generation (Table 2; Figure 8). We evaluate SiT_{REPA}-XL/2 with the null class after 50K updates, using EMA weights, 50,000 images, and the native Euler–Maruyama sampler. Correction detaches both the proposal and the base prediction. We sweep $K = 0, \ldots, 10$ using $\lfloor 250/(K+1) \rfloor$ denoising steps, and autoguidance $w = 1, \ldots, 10$ using the released pretrained guide at 250 NFE. The table reports the lowest-FID settings from these sweeps, in addition to $K = 1$. Sampling settings are summarized in Table 14.

Text-to-image examples (Figure 10). We compare $K = 0$ and $K = 6$ using the same 50K-update EMA checkpoint, without CFG, and 50 Euler steps (50 and 350 NFE). Paired images share their prompt and recorded rank seed. Full prompts are listed in Table 17.

Table 17: **Full prompts for the text-to-image examples in Figure 10.**

Figure column	Full prompt
1. Green guitar	In the center of a dimly lit stage, a vintage green guitar with a glossy finish leans gently against a classic black amplifier. The polished wooden flooring of the stage reflects the warm, golden hue from a single spotlight above, which casts an inviting glow over the instrument. A microphone stand with a chrome finish stands to the left of the guitar, poised for the evening's performance.
2. Vintage car	a vintage blue Porsche 356 is captured mid-turn on a winding asphalt road, its polished surface reflecting the bright sunlight. The car's classic design is accentuated by its rounded headlights and sleek bodywork. Alongside the road, a low stone wall partially covered in moss provides a contrast to the well-maintained vehicle.
3. Sunlit kitchen	A spacious kitchen with a large wooden table at the center, cluttered with an assortment of pots, pans, and cooking utensils. The table is surrounded by matching wooden chairs, and the walls are lined with shelves filled with spices and kitchenware. Sunlight streams in through a window, casting a warm glow on the array of cookware.
4. Three violins	Three exquisitely crafted violins, with their elegant F-shaped sound holes and smooth, glossy wooden surfaces, rest gently beside a grand piano. The piano's polished ebony finish reflects the soft, gentle curves of the violins that lie on a velvet-lined case. Behind them, the piano's open lid reveals its intricate strings and hammers, inviting a symphony of sounds to be played.
5. Rabbit and seashell	On the soft, warm sand of the beach, a fluffy white rabbit with rounded ears is caught in a curious moment, gently placing its paw on the ribbed surface of a pink scallop shell. The scallop, slightly open, reveals its smooth interior contrasting with its coarse outer texture, while hues of pink and orange from the setting sun reflect off its surface. There's a tranquil ocean backdrop with the gentle ebbing of the tide, and the fading daylight casts a golden glow over the scene, highlighting the rabbit's soft fur and the shell's subtle color.
6. Victorian library	A voluminous library, bathed in soft, muted blue hues, stretches majestically across a spacious Victorian-style living room. Intricately carved wooden bookshelves, filled with an array of books of various sizes and colors, line the walls and a grand fireplace sits as the room's focal point. The vray render showcases hyperrealistic textures, from the plush, patterned rug beneath the polished wooden tables, to the delicate fabric of the antique armchairs scattered throughout the space.

Guidance and correction (Figure 11). We compare $w \in \{1, 6\}$ and $K \in \{0, 6\}$ at 50 Euler steps. The selected prompt is *a photo of a bench*, with all eight seeds retained per setting. CFG is applied before correction.

Parameter sharing (Table 5). We compare plain SiT-XL/2 models after 100K updates, without CFG, using FID50K. The shared model uses separate input embedders and shares the transformer blocks and output head. Both separate and shared correctors detach the proposal and base prediction, but shared parameters receive gradients from both losses. The baseline is a sep-

arate denoising-only continuation. Baseline, separate-corrector, and shared-corrector settings use $(K, \text{steps}) = (0, 250), (2, 83), (3, 62)$, respectively.

Distillation (Table 6). We freeze the separate SiT-XL/2 teacher at EMA100K and initialize the student from its denoiser. The student minimizes velocity MSE against the teacher's $K = 2$ prediction, without a ground-truth loss, for 100K updates on class-conditional ImageNet. We sample Gaussian noise and uniform time, using batch size 256, AdamW with learning rate 10^{-4}, 100 warmup updates, zero weight decay, and EMA decay 0.9999. We evaluate the final EMA with FID@50K using the native SiT Euler–Maruyama sampler and ADM evaluator.

Power spectra (Figure 36). We average radial power over 100 ImageNet-256 samples per setting for $K = 0, \ldots, 10$ at approximately 250 NFE. The panels show $\log_{10}$ power and $\log_{10}(P_{\text{sample}}/P_{\text{real}})$ over normalized frequencies 0.1–0.5.

C Additional Results

C.1 Comparing Correction and CFG

Fig. 24 compares correction and CFG on ImageNet using the same checkpoint and initial noise. In these selected examples, increasing guidance produces more similar close-up compositions and stronger colors, whereas correction retains more varied poses and backgrounds. These observations concern the displayed samples; they do not by themselves establish distributional mode collapse.

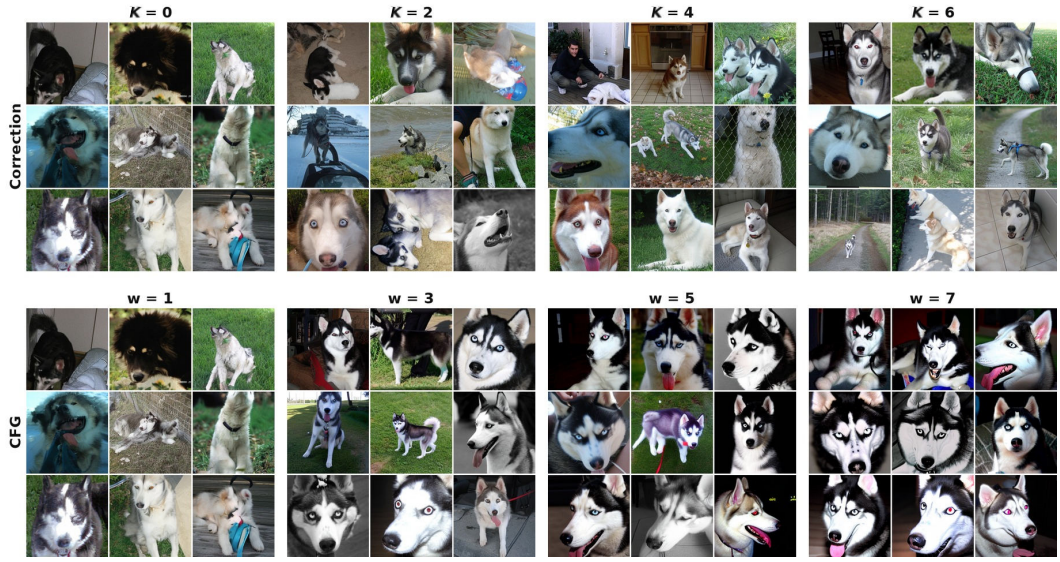


Figure 24: **Correction and guidance produce different visual trade-offs.** Matched-noise samples from SiT_{REPA}-XL/2 as correction depth K (top) or guidance scale w (bottom) increases, at approximately matched NFE.

Fig. 25 quantifies the trade-offs on DiT. At approximately matched sampling compute, correction reaches a lower minimum FID than the best evaluated CFG setting.

Following Sadat et al. (2024), we also measure mean HSV saturation against a real-image reference. Stronger CFG raises saturation above the reference, whereas correction remains near or below the unguided level. This ImageNet measurement complements the text-to-image examples; the latter illustrate refinement at weaker guidance rather than measuring a reduction in saturation caused by correction.

Additional Correction and Guidance Comparisons Figures 26, 27, and 28 retain all 16 fixed noise seeds (0–15) for nine classes selected from an exploratory 24-class comparison. No samples are omitted within these classes. The correction and guidance rows use separately trained SiT-XL/2

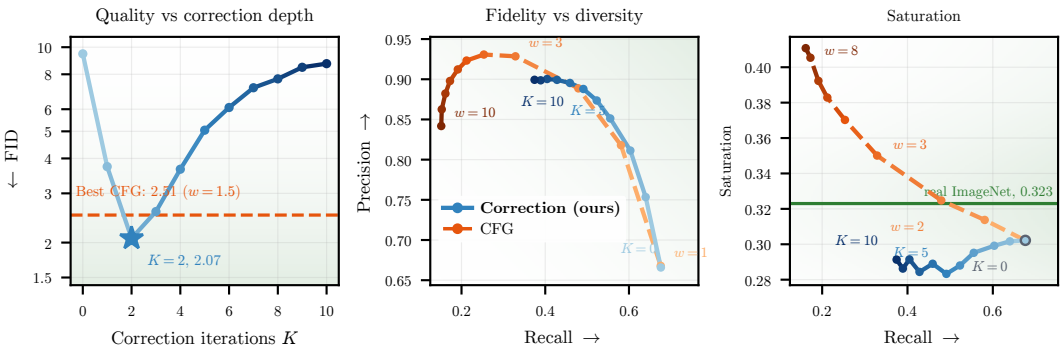


Figure 25: **Correction improves FID with a different precision–recall and saturation trade-off.** DiT-XL/2 at approximately matched NFE. The dashed orange line marks the best evaluated CFG FID; the green line marks real-image saturation.

models at the same 100K-update budget, with EMA weights. Correction uses no CFG. All settings use approximately 250 network evaluations (249–252). Panels share the same initial Gaussian noise; their denoising grids and subsequent stochastic trajectories differ. FID labels are existing ImageNet-wide FID@50K results, rather than FID estimates for the displayed class. These examples illustrate visual trade-offs and do not estimate the number of modes covered by either method.

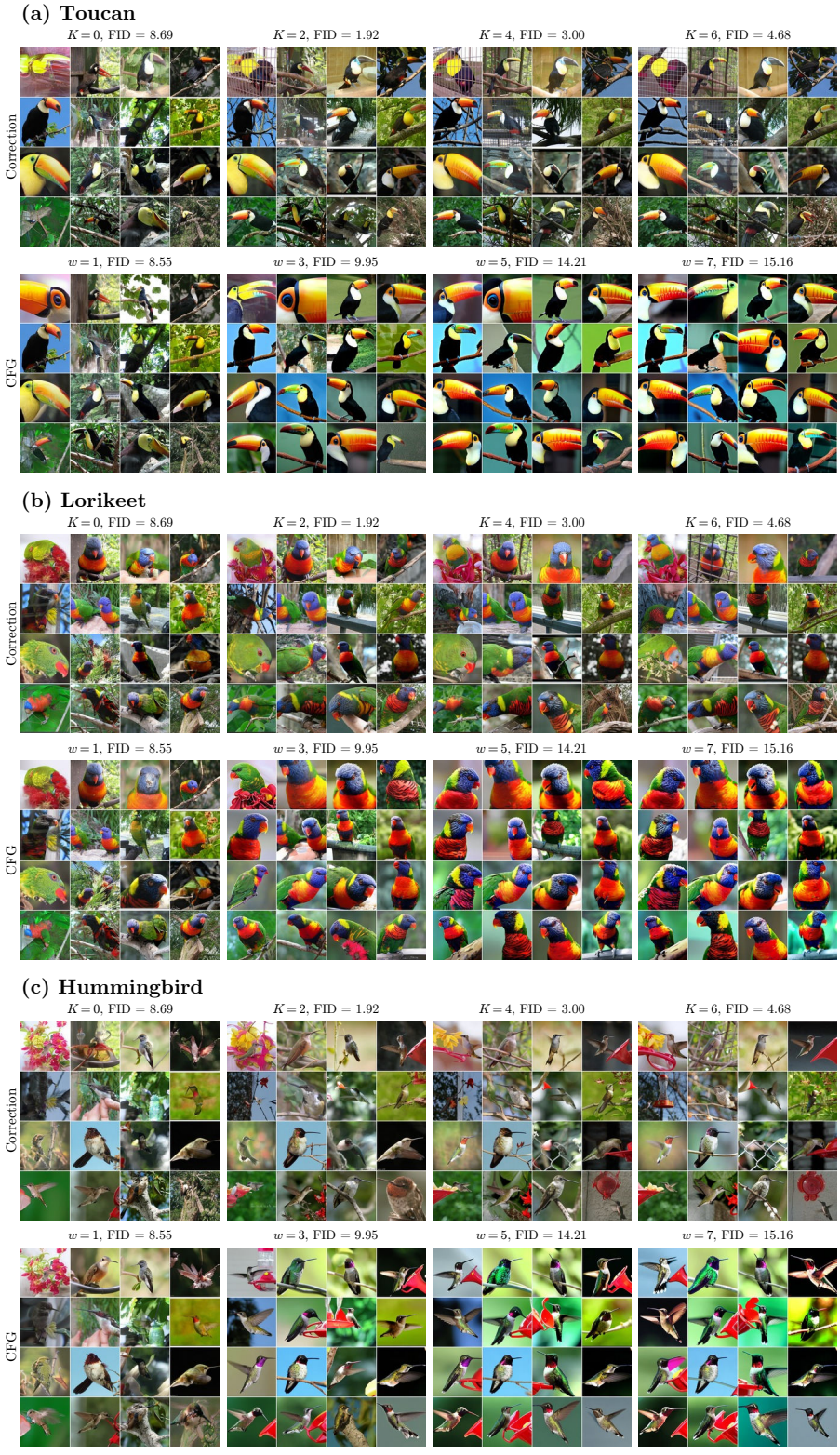


Figure 26: **Correction and guidance comparisons.** **(a)** toucan, **(b)** lorikeet, and **(c)** hummingbird samples from SiT-XL/2. For each class, the upper row varies correction depth $K = 0, 2, 4, 6$ without CFG; the lower row varies guidance scale $w = 1, 3, 5, 7$ in a separately trained denoising-only model. Both models use EMA weights after 100K updates and approximately 250 NFE. All 16 fixed seeds appear in the same row-major order in every panel. FID labels use 50K samples across ImageNet classes.

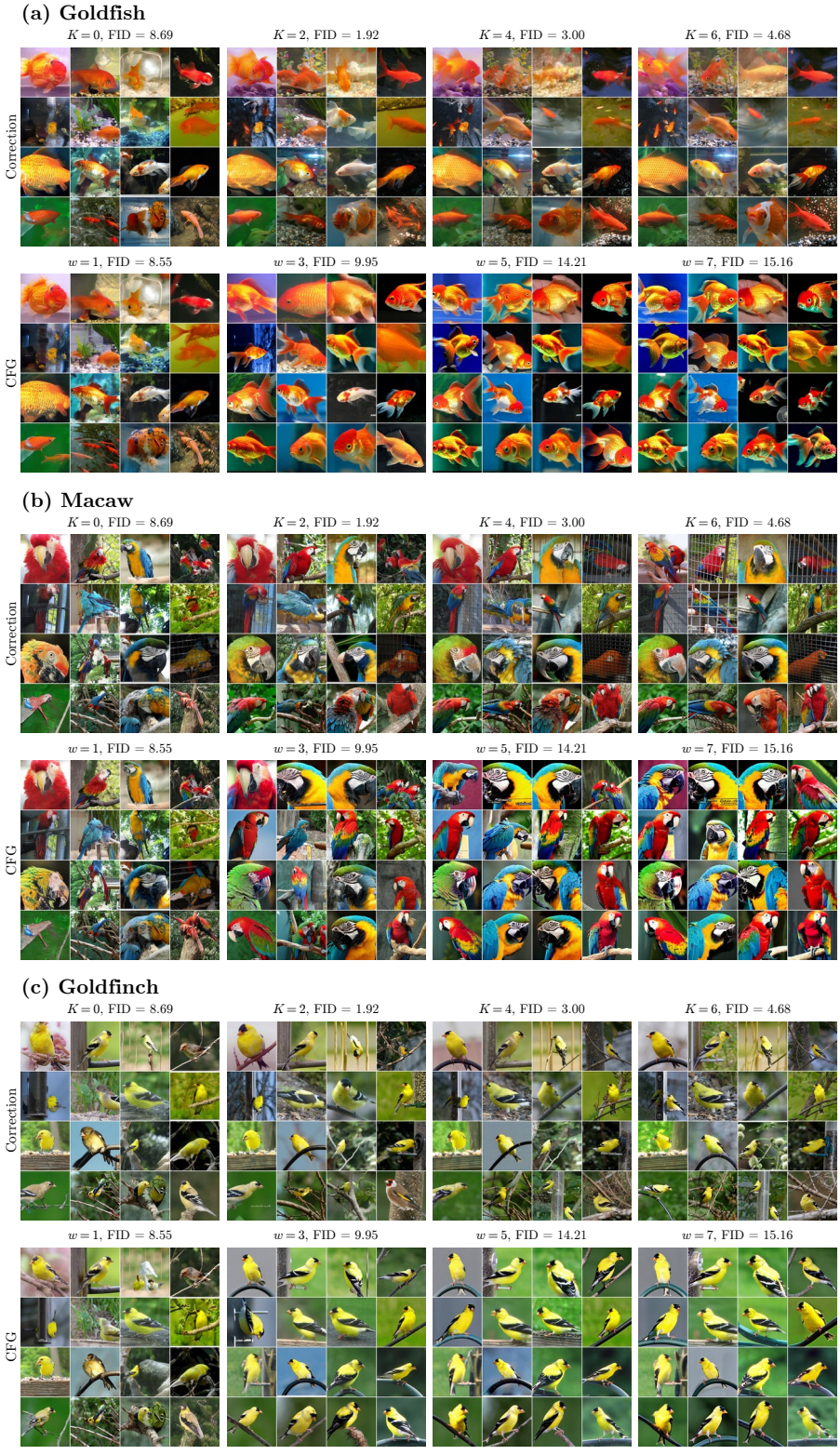


Figure 27: **Correction and guidance comparisons.** **(a)** goldfish, **(b)** macaw, and **(c)** goldfinch samples from SiT-XL/2. For each class, the upper row varies correction depth $K = 0, 2, 4, 6$ without CFG; the lower row varies guidance scale $w = 1, 3, 5, 7$ in a separately trained denoising-only model. Both models use EMA weights after 100K updates and approximately 250 NFE. All 16 fixed seeds appear in the same row-major order in every panel. FID labels use 50K samples across ImageNet classes.

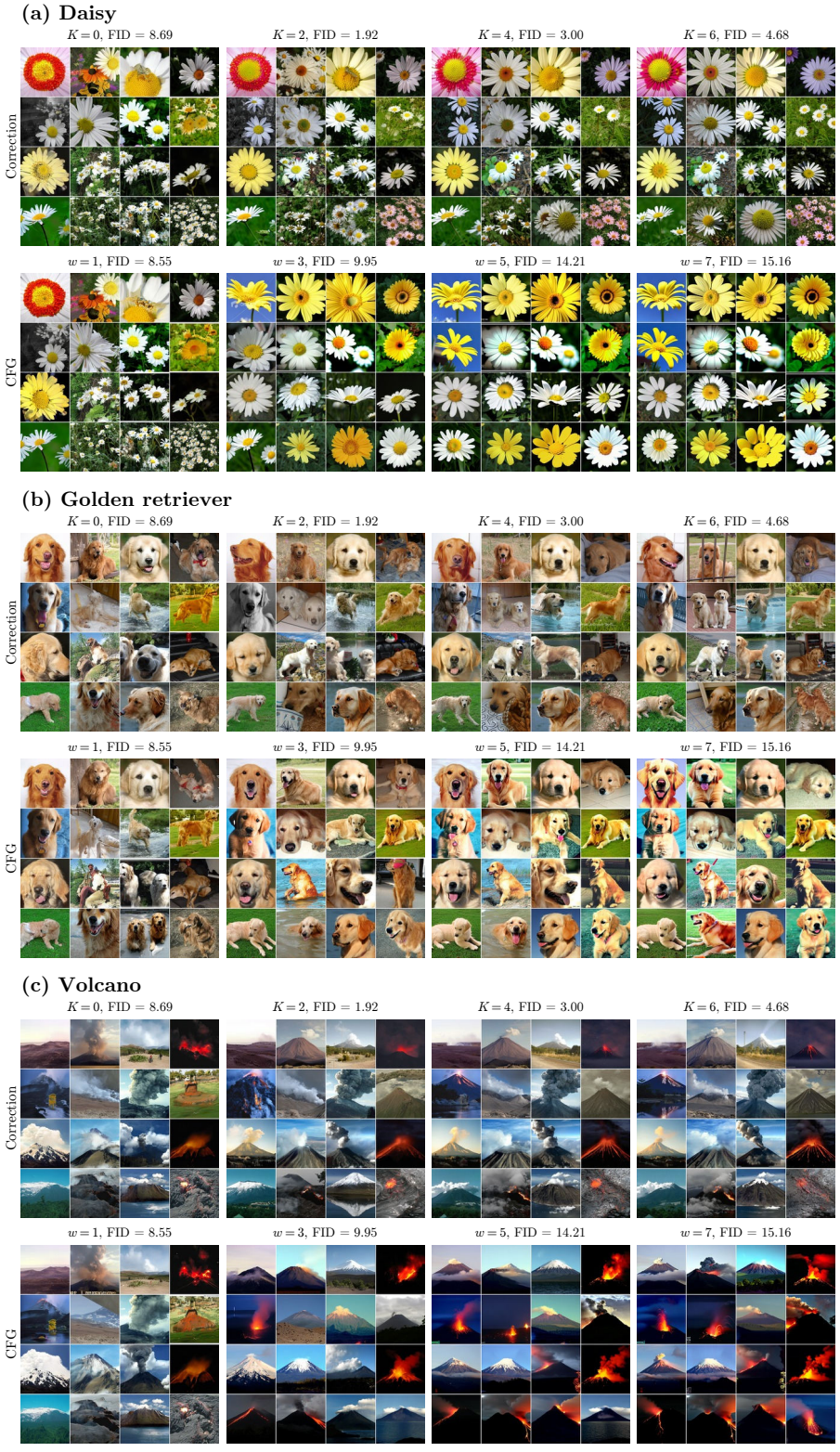


Figure 28: **Correction and guidance comparisons across additional classes.** **(a)** Daisy, **(b)** golden retriever, and **(c)** volcano. SiT-XL/2 with the same checkpoints, sampling settings, and 16 fixed seeds as Figs. 26 and 27. Top: correction $K = 0, 2, 4, 6$ without CFG. Bottom: CFG $w = 1, 3, 5, 7$. FID labels are ImageNet-wide FID@50K.

C.2 COMBINING CFG AND CORRECTION

Combination order. CFG and correction can be applied in either order. Applying CFG first combines the conditional and null predictions before the corrector refines the guided prediction using the conditional text. Applying correction first refines both branches independently and then combines them with CFG. We use CFG $\rightarrow$ correction for the guided qualitative examples and the main human-preference comparison because we prefer its visual appearance, while correction $\rightarrow$ CFG is used for the guided alignment results in Tab. 3.

Table 18 compares quality and diversity preferences at matching w, K across the two orders. Among the evaluated guided settings, correction before CFG more often yields higher quality preference, while correction after CFG more often yields higher diversity preference. Thus neither order consistently leads on both criteria.

Table 18: **Neither correction order consistently leads in both quality and diversity.** Full sweep of candidate vote shares (%) against released CFG ($w = 6$, $K = 0$); higher is better. Bold marks the setting reported in Table 4. Each entry contains 800 responses across 80 prompts. The two orders were rated separately against the baseline.

		CFG $\rightarrow$ Correction		Correction $\rightarrow$ CFG	
w	K	Quality	Diversity	Quality	Diversity
1	2	33.38	55.38	37.25	54.50
1	4	49.38	57.25	47.50	59.38
1	6	55.75	55.75	52.38	60.75
1	8	54.13	56.38	56.75	63.88
2	2	48.50	60.50	55.38	59.13
2	4	55.25	59.25	65.00	59.00
2	6	57.25	58.75	64.25	61.50
2	8	60.25	59.38	65.75	59.50
4	2	54.63	61.25	61.63	52.25
4	4	58.25	59.50	67.00	57.50
4	6	61.75	55.38	66.88	55.63
4	8	**60.75**	**60.25**	60.50	56.63
6	2	57.13	54.38	59.50	49.63
6	4	59.63	52.75	61.00	53.75
6	6	57.38	58.38	63.63	54.00
6	8	58.63	58.38	59.75	58.13

C.3 HUMAN EVALUATION

We evaluate our method with human raters, using Rapidata to collect human-feedback. It enables high-throughput collection of diverse human judgments, roughly takes few minutes on thousands of human responses. Annotator qualification and reliability filtering are applied to control response quality. We additionally used language filter "English" and age filter "adult".

Table 4 reports unweighted preference votes on all 80 GenEval single-object prompts, with 10 judgments per prompt (800 per metric). Quality uses one image per method and diversity uses four-image sets. We first apply CFG and then correct the guided prediction using the conditional text. Both methods use 50 Euler steps.

Table 18 provides the full results for both correction orders over $w \in \{1, 2, 4, 6\}$ and $K \in \{2, 4, 6, 8\}$, highlighting the setting reported in the main text. Each setting is compared separately against the same released baseline. At $w = 1$, the two orders use identical images, so differences reflect repeated human ratings. Figure 29 illustrates the questions and stimuli for the main table's CFG-then-correction setting ($w = 4$, $K = 8$).

Prompt: “a photo of a teddy bear”

(a) Quality

Which image has better visual quality?

CFG baseline ($w = 6$)



Corrective sampling ($w = 4$, $K = 8$)



(b) Diversity

Which group is more diverse? Ignore image quality.

CFG baseline ($w = 6$)



Corrective sampling ($w = 4$, $K = 8$)



Figure 29: **Quality and diversity are assessed separately.** Actual teddy-bear stimuli and questions for Table 4, using CFG followed by correction ($w = 4$, $K = 8$): single images for quality and four-image sets for diversity.

D Discussion

D.1 Training and capacity controls

Additional-training and capacity controls. We examine how quickly the corrector converges by comparing correction training with three baselines: continued training of the base denoiser and two deeper denoisers with different initialization schemes. We use the pretrained SiT_{REPA}-XL/2 denoiser with 28 transformer layers from Tab. 1. For the deeper denoisers, we append 28 transformer layers before the final unembedding layer, matching the total number of transformer layers in the denoiser–corrector pair. The added layers are either randomly initialized or copied from the pretrained denoiser, with the latter matching our corrector initialization. We train all configurations for 50K steps and report their progress in Fig. 30. We use a single correction pass ($K = 1$) to isolate its benefit from the additional gains of iterative refinement.

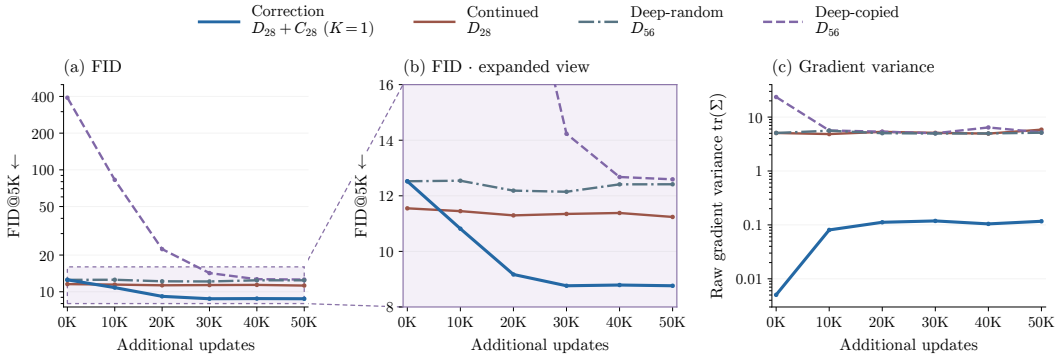


Figure 30: **Correction improves generation quality beyond continued denoiser training or parameter-matched deeper denoisers.**

As can be seen in Fig. 30 (a) and (b), correction improves FID substantially faster over the 50K-update training window we used, whereas continued training and both deeper-denoiser controls improve much less. Importantly, this gain appears with only a *single correction pass*: despite using the same total sequential transformer depth as the 56-layer denoisers, the $D_{28} + C_{28}$ correction model achieves substantially better generation quality. These results suggest that *additional training or model depth alone cannot explain the gains*, pointing to an advantage of learning to refine an already structured prediction over learning directly from noisy observations.



Figure 31: **Matched effective depth.** Correction outperforms a deeper denoiser at matched effective depth per denoising step and inference budget, using fewer parameters.

Iterative correction improves quality at matched effective depth. One might argue that the gain from repeating correction simply comes from increasing the effective depth within each denoising step. To test whether this is the case, we compare the 28-layer monolithic denoiser (D_{28}) with a 7-layer denoiser and a 7-layer corrector ($D_7 + C_7$), using $K = 3$ corrections to match the effective depth of the monolithic denoiser ($7 + 3 \times 7 = 28$; Fig. 31). For this comparison, we train both configurations from scratch for 80K updates. Both use exactly 240 denoising steps and 6,720 transformer-layer evaluations per sample. Surprisingly, despite using roughly half as many parameters (346M versus 675M), the shallow denoiser–corrector pair achieves a lower FID@50K (4.21 versus 4.88; Tab. 7). This highlights that iterative correction does more than improve prediction accuracy through additional denoiser layers.

D.2 ITERATIVE CORRECTION VISUALIZATIONS

Analytic target. Let $\mathcal{B}_y = \{\boldsymbol{x}_i\}_{i=1}^{N_y}$ be a bank of clean images from class y, with a uniform empirical prior. For $\boldsymbol{z}_t = (1-t)\boldsymbol{x} + t\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and $0 < t \leq 1$, the posterior mean under this prior is

$$\begin{aligned} \boldsymbol{x}_{\text{analytic}}(\boldsymbol{z}_t, t, y) &= \frac{\sum_{i=1}^{N_y} w_i(\boldsymbol{z}_t, t)\, \boldsymbol{x}_i}{\sum_{i=1}^{N_y} w_i(\boldsymbol{z}_t, t)}, \\ w_i(\boldsymbol{z}_t, t) &= \exp\left(-\frac{\|\boldsymbol{z}_t - (1-t)\boldsymbol{x}_i\|_2^2}{2t^2}\right). \end{aligned} \tag{10}$$

Additional iterative-correction examples. Figures 32(a) and 32(b) show the remaining selected examples from the visualization in Fig. 13, without repeating its six rows. All use the same JiT-L/16 checkpoint after 100K training updates and its second EMA weights, with class conditioning and CFG scale 1. For each row, the denoiser receives a forward input $\boldsymbol{z}_t = (1-t)\boldsymbol{x} + t\boldsymbol{\epsilon}$, and the corrector is repeatedly applied to the clean prediction at that fixed timestep. The selected examples use noise levels $t \in \{0.8, 0.9, 0.95, 0.975\}$, where 1 denotes noise and 0 denotes data. The Analytic column uses the same formula (Eq. 10), with a uniform empirical prior over a finite same-class image bank excluding the original query cohort. The examples were selected for qualitative illustration and are not a random-sample quality estimate.

Interpolation of corrected predictions toward noise. Figure 33 shows all eight selected noisy-state comparisons. They use $t = 0.9 \to s = 0.5$ or $t = 0.95 \to s = 0.55$, with the same interval $t - s = 0.4$ throughout. For each example and every K, we hold the original forward input $\boldsymbol{z}_t$ fixed and construct $\boldsymbol{z}_s^{(K)} = (s/t)\boldsymbol{z}_t + (1 - s/t)\hat{\boldsymbol{x}}^{(K)}$. The real reference is $(1-s)\boldsymbol{x} + s\boldsymbol{\epsilon}$ with the same original Gaussian. Interpolation is applied to unclipped predictions; all cells use the same display range, without contrast normalization. This is an illustrative interpolation to a lower noise level rather than a single step from the sampling grid. The corrected predictions need not reconstruct the original image: the comparison concerns the plausibility of visible structure after interpolation, rather than paired reconstruction accuracy.

Correction within a denoising step. Fig. 3 provides a qualitative view of how correction changes intermediate clean-sample predictions during sampling. Without correction, structural errors in an intermediate prediction can persist into subsequent denoising steps because the prediction is immediately used to update the noisy state. Correction instead refines the prediction at fixed t before this update. In the examples in Fig. 3, successive corrections sharpen object structure and suppress visible artifacts within a single denoising step, so the subsequent sampler state is constructed from an improved prediction.

To illustrate what correction does at each step, Fig. 34 shows the individual correction passes for three examples from SiT$_{\text{REPA}}$-XL/2. For each example, the baseline and correction trajectories start from the same initial noise and use the same 15 Euler denoising steps, with $K = 8$ for correction. In the dog example (Fig. 34), already at early-to-middle stages of sampling ($t = 0.73$–0.60), successive corrections reduce structural artifacts and reveal a coherent dog shape. These improvements carry through to the final image, while the baseline retains structural artifacts even after completing all denoising steps.

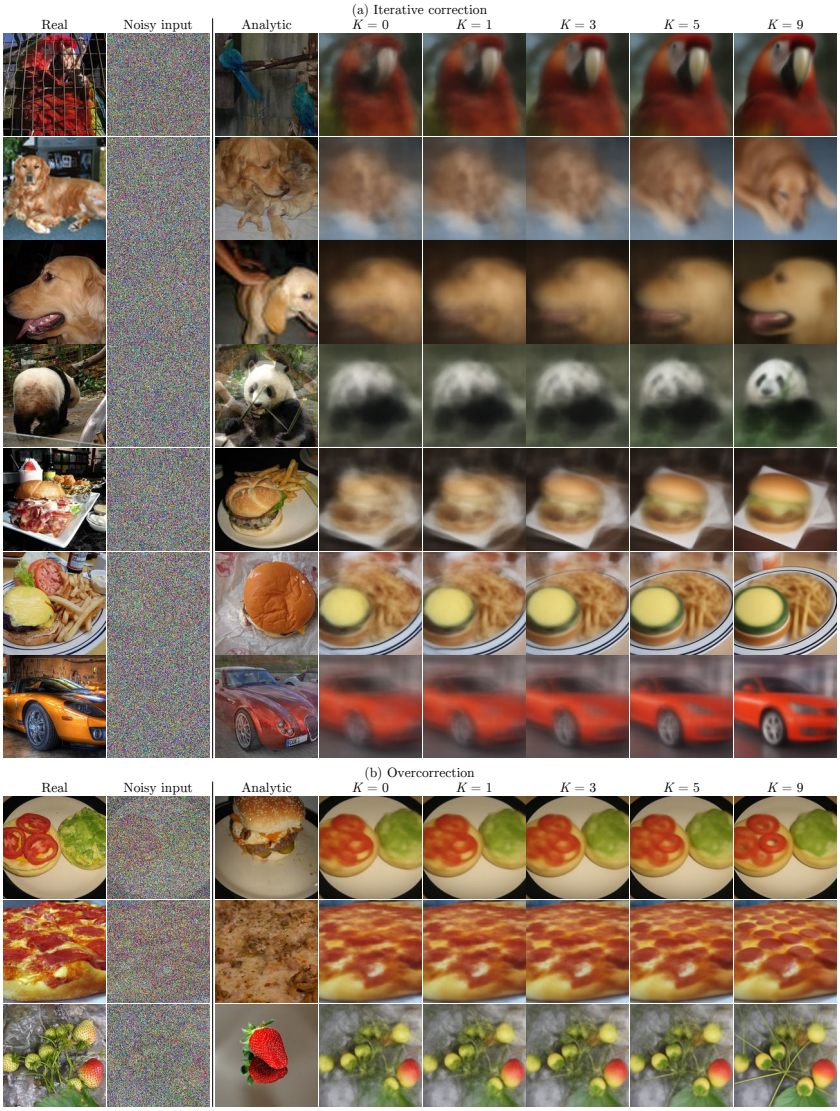


Figure 32: **Additional iterative-correction examples.** The seven selected examples not included in Fig. 13(a), using the same column layout and fixed-input protocol. Repetition strengthens object contours and details, while the predictions need not align with the original clean image or the finite-bank analytic estimate.

Additional overcorrection examples. The three selected examples not included in Fig. 13(b). Repeated correction can simplify irregular textures or amplify thin structures into unnatural patterns.

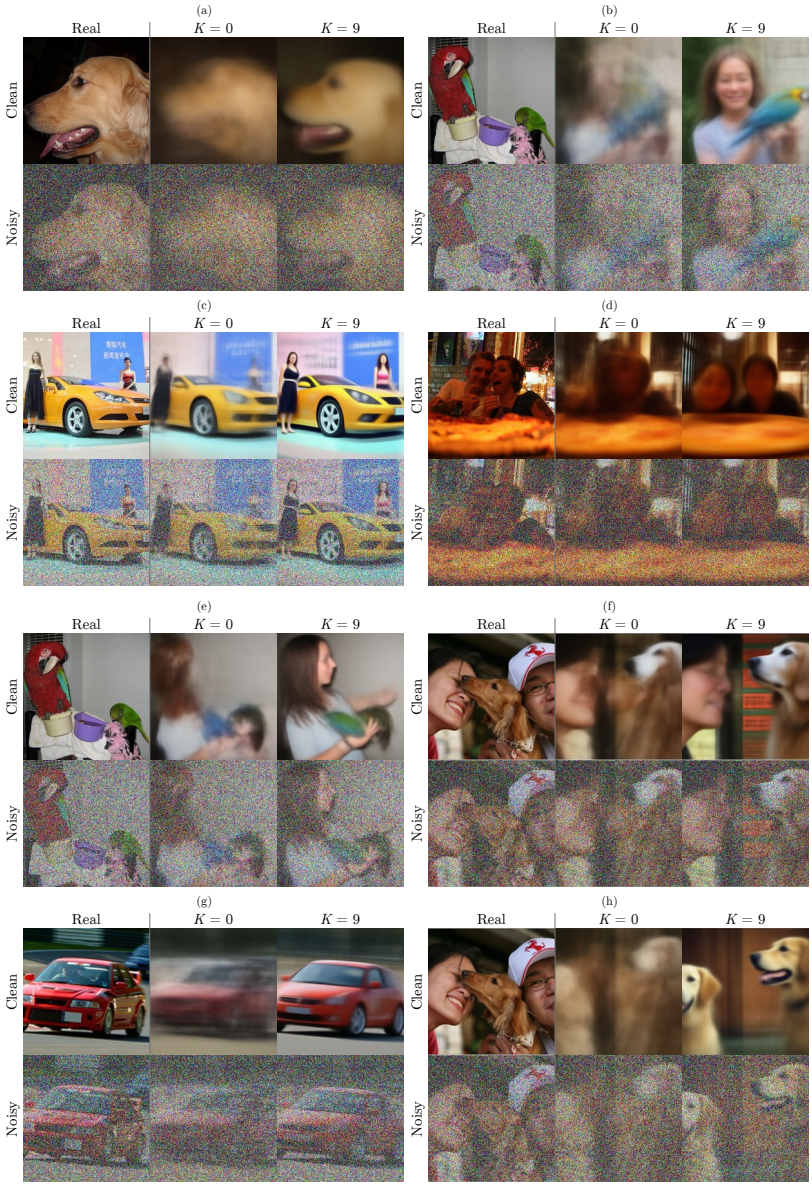


Figure 33: **Shared-input noisy-state comparisons.** Clean images (top) and interpolated noisy states (bottom), comparing *Real*, $K = 0$, and $K = 9$. Panels **(a,b,g,h)** use $t = 0.95 \rightarrow s = 0.55$; panels **(c,d,e,f)** use $t = 0.9 \rightarrow s = 0.5$. Source queries (classes), in panel order, are 264649 (207), 114599 (88), 1047154 (817), 1233725 (963), 114599 (88), 265339 (207), 1047733 (817), and 265339 (207). Repeated query IDs are shown at different noise levels.

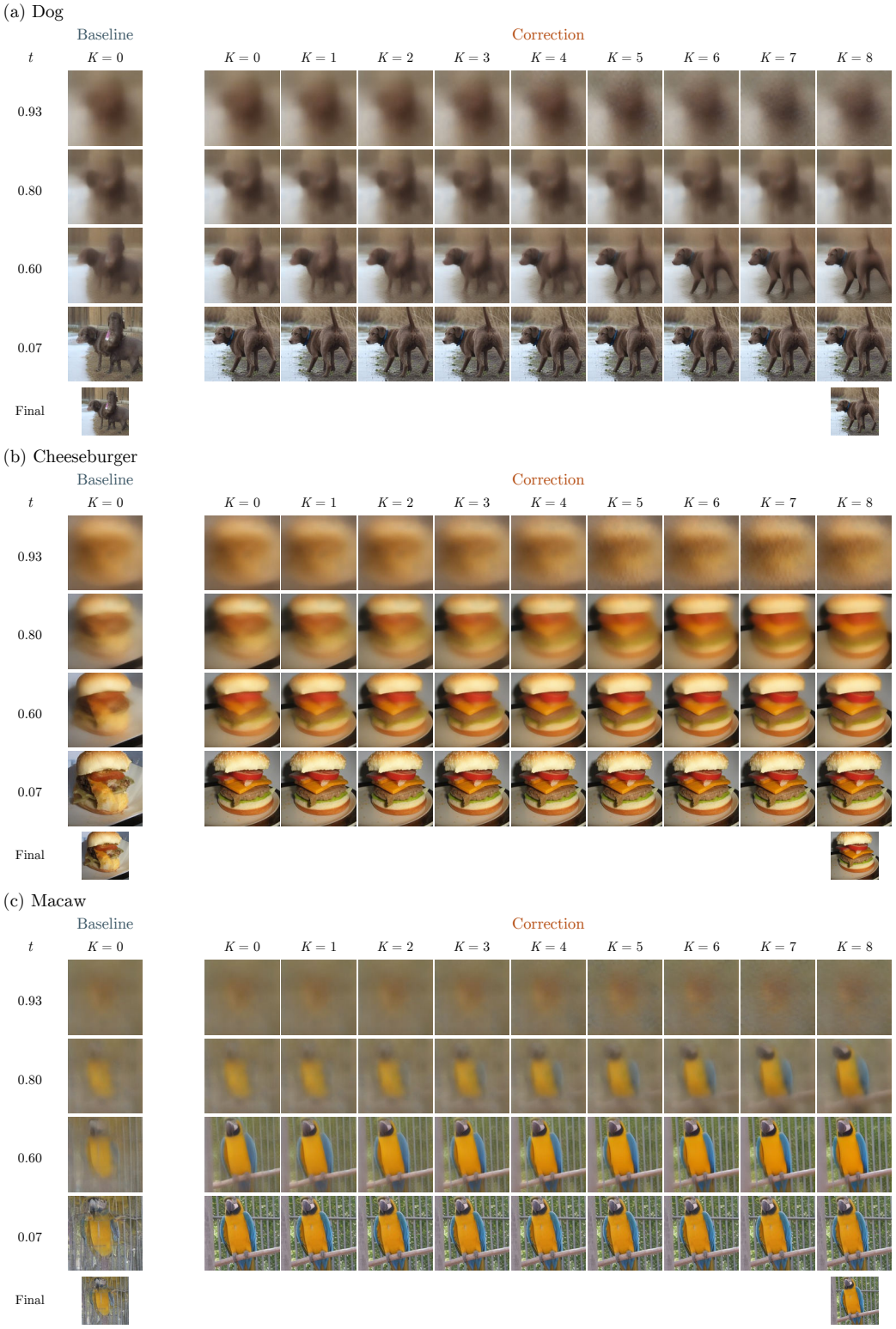


Figure 34: **Correction refines intermediate predictions before denoising continues.** Baseline and correction start from the same noise and use 15 denoising steps. Panels (a–c) show dog, cheeseburger, and macaw examples at selected timesteps $t \in \{0.93, 0.80, 0.60, 0.07\}$. Each prediction row shows all eight correction passes at fixed t; the final row of each panel shows the actual final samples.

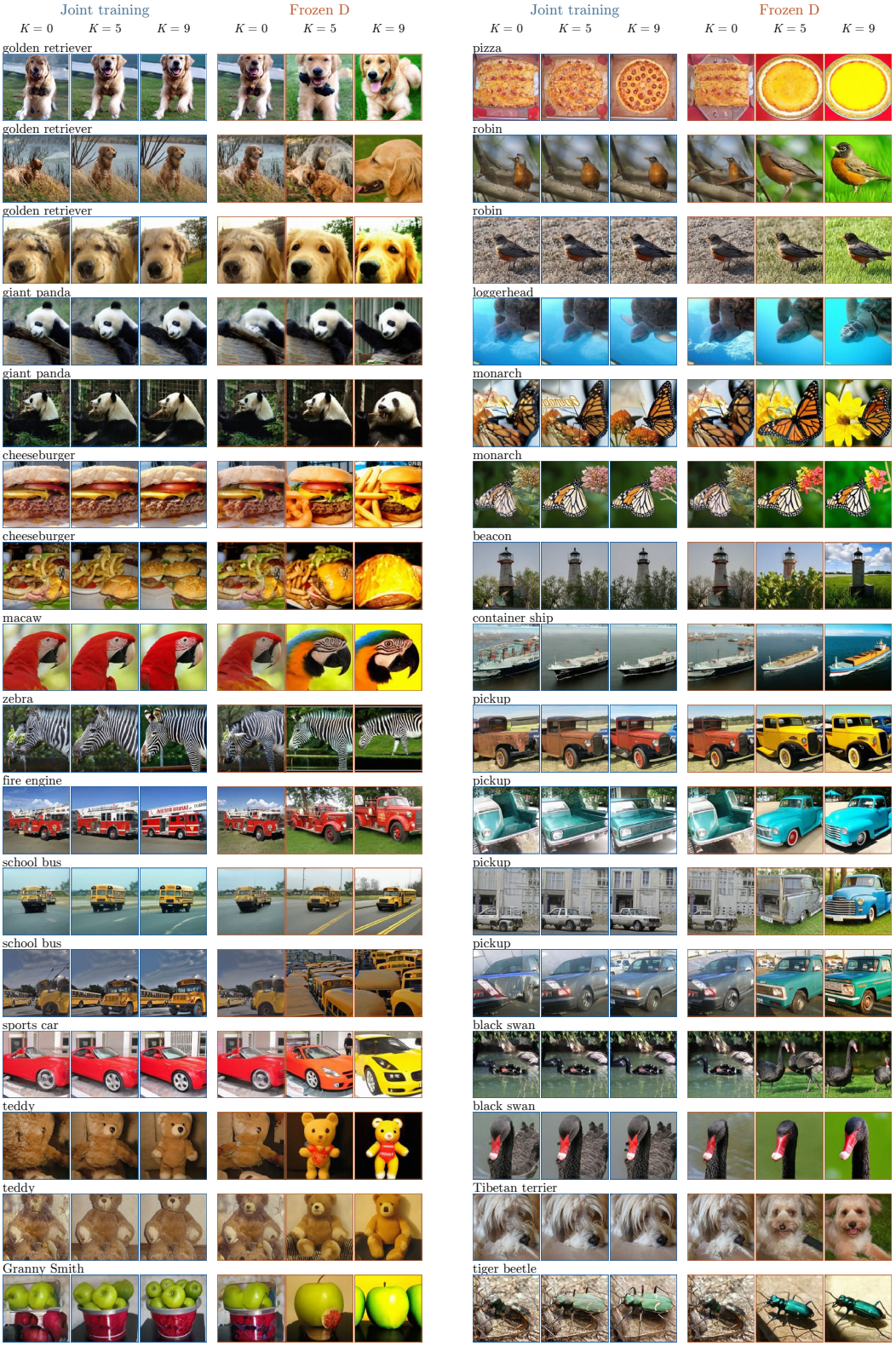


Figure 35: **Additional examples with joint and frozen denoisers.** Each row compares joint training (left triplet) and a frozen denoiser (right triplet) at $K = 0, 5, 9$, starting from the same noise. Selected JiT-L/16 samples use 100K checkpoints, EMA weights, no CFG, and 45 Heun steps; larger K adds correction compute.

D.3 Power spectrum analysis

Frequency-domain effects of correction. To further characterize the changes induced by correction, we examine the frequency content of generated samples. Prior work (Yu & Zhan, 2025; Kobayashi et al., 2026) has reported lower subband energy in reverse-process samples than in forward-corrupted real data. This motivates examining whether correction recovers frequency components that are underrepresented in baseline samples.

We compare the power spectra of samples generated with $K = 0, \ldots, 10$ correction iterations against those of real images. As shown in Fig. 36, the baseline samples exhibit a spectral deficit, particularly at high frequencies. Increasing the number of correction iterations reduces this deficit, with $K = 6$ yielding a spectrum that approximately matches the real-image reference. Further iterations can raise both low- and high-frequency power above the reference, indicating that the effect depends on the correction count.

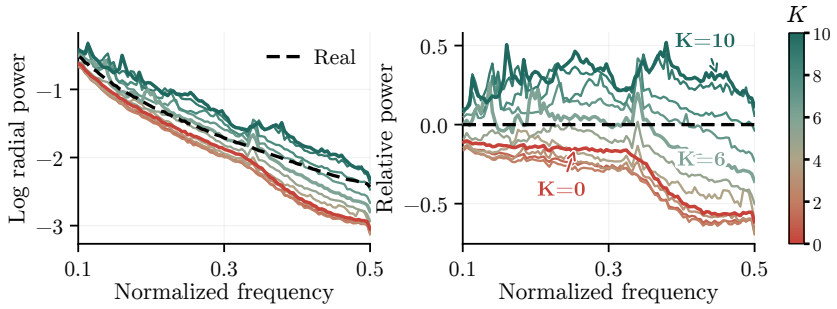


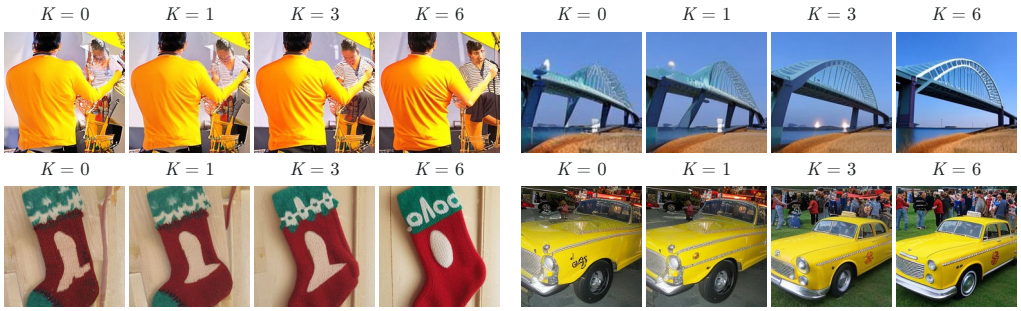
Figure 36: **Correction reduces the high-frequency deficit of generated samples.** Left: $\log_{10} P$. Right: $\log_{10}(P/P_{\text{real}})$. Dashed lines indicate real-image references.

E Additional Samples

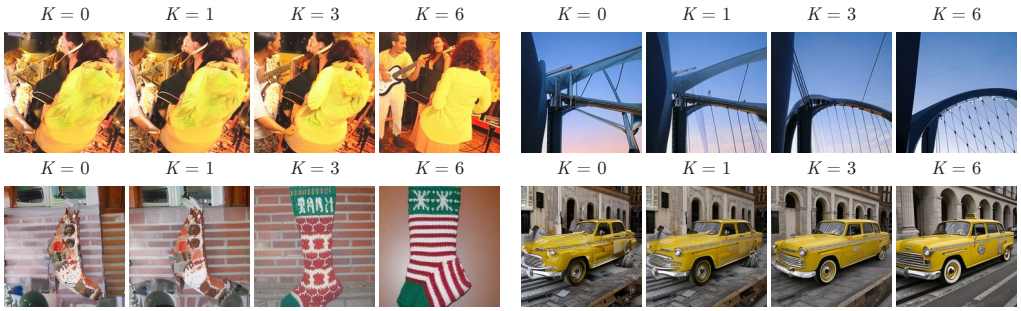
E.1 Effect of Correction Depth

We show the same class and initial noise as correction depth increases, followed by paired galleries over randomly drawn ImageNet classes. All examples use class conditioning without CFG.

SiT-XL/2



SiT-REPA-XL/2



JiT-L/16

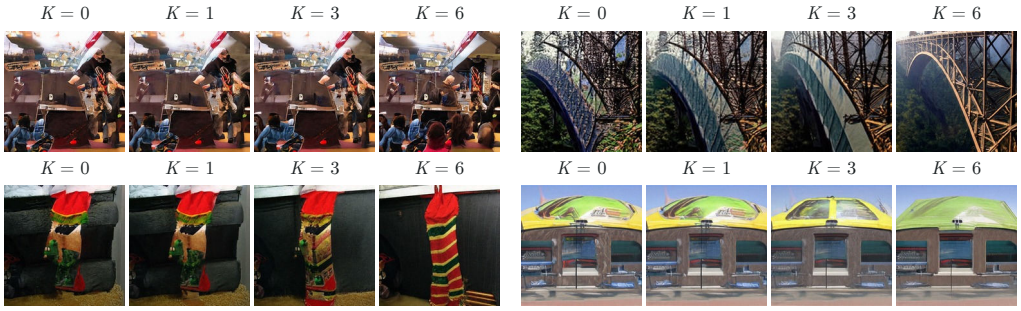


Figure 37: ***Uncurated* samples: correction depth changes both structure and detail.** Each strip follows one sample as K increases, with the initial noise and denoising grid fixed.

E.2 SiT: Random-Class Generation

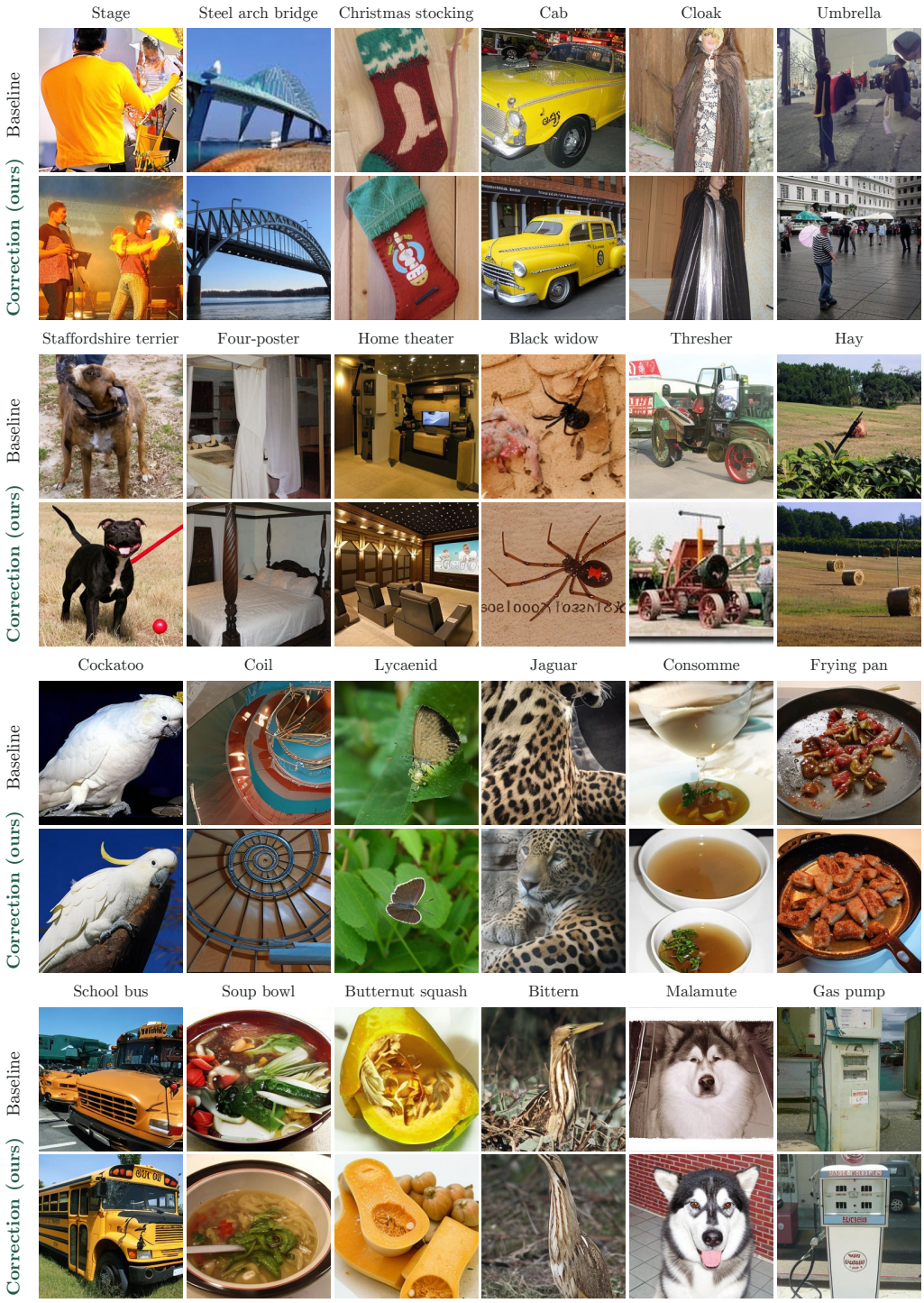


Figure 38: ***Uncurated* samples: correction refines unguided SiT samples across random classes.** Within each pair of rows, baseline ($K = 0$) and correction ($K = 6$) share the same class and initial noise at approximately matched NFEs.

E.3 SiT$_{\text{REPA}}$: Random-Class Generation

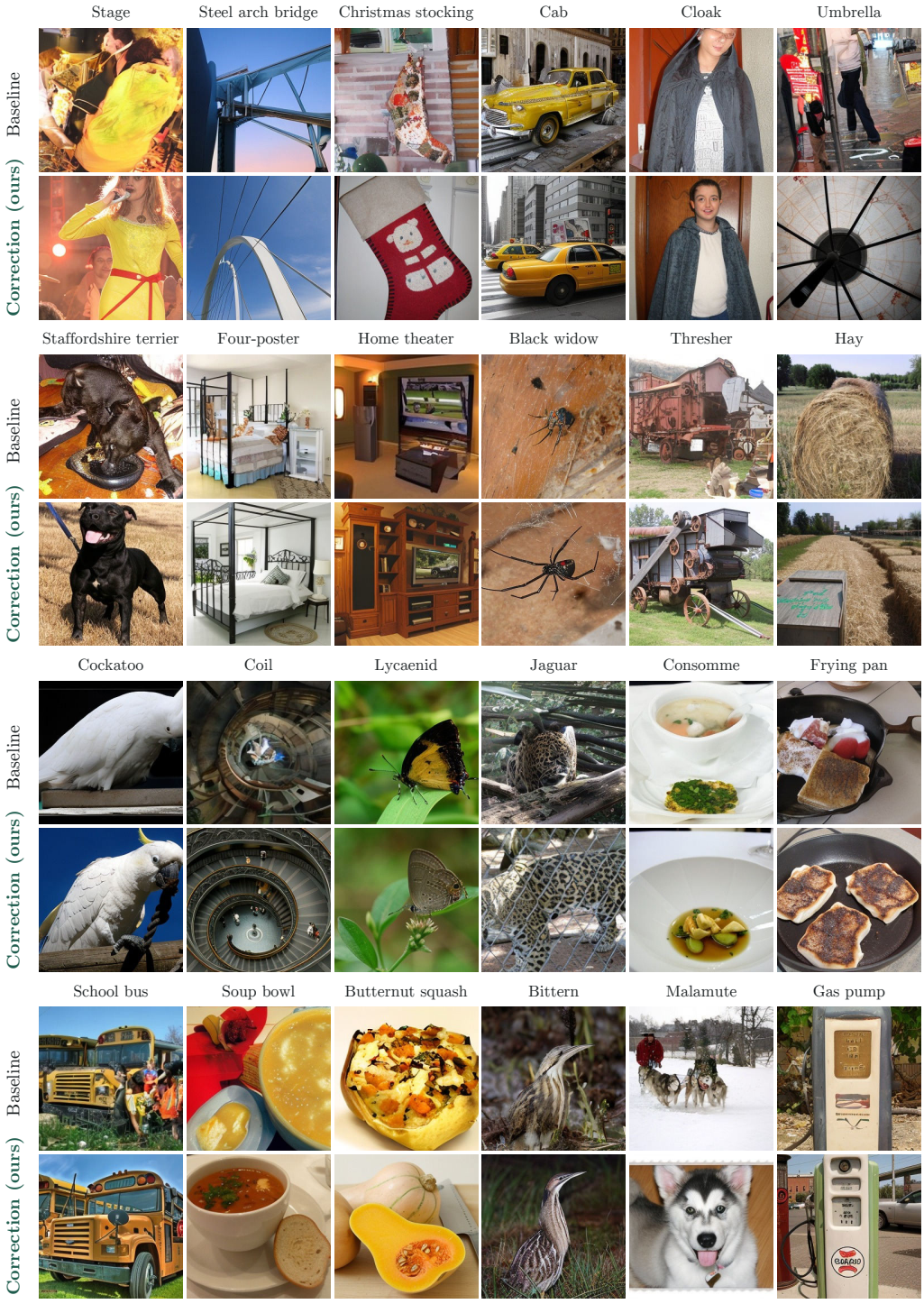


Figure 39: ***Uncurated* samples: correction refines unguided SiT$_{\text{REPA}}$ samples across random classes.** Within each pair of rows, baseline ($K = 0$) and correction ($K = 6$) share the same class and initial noise at approximately matched NFEs.

E.4 JiT: Random-Class Generation

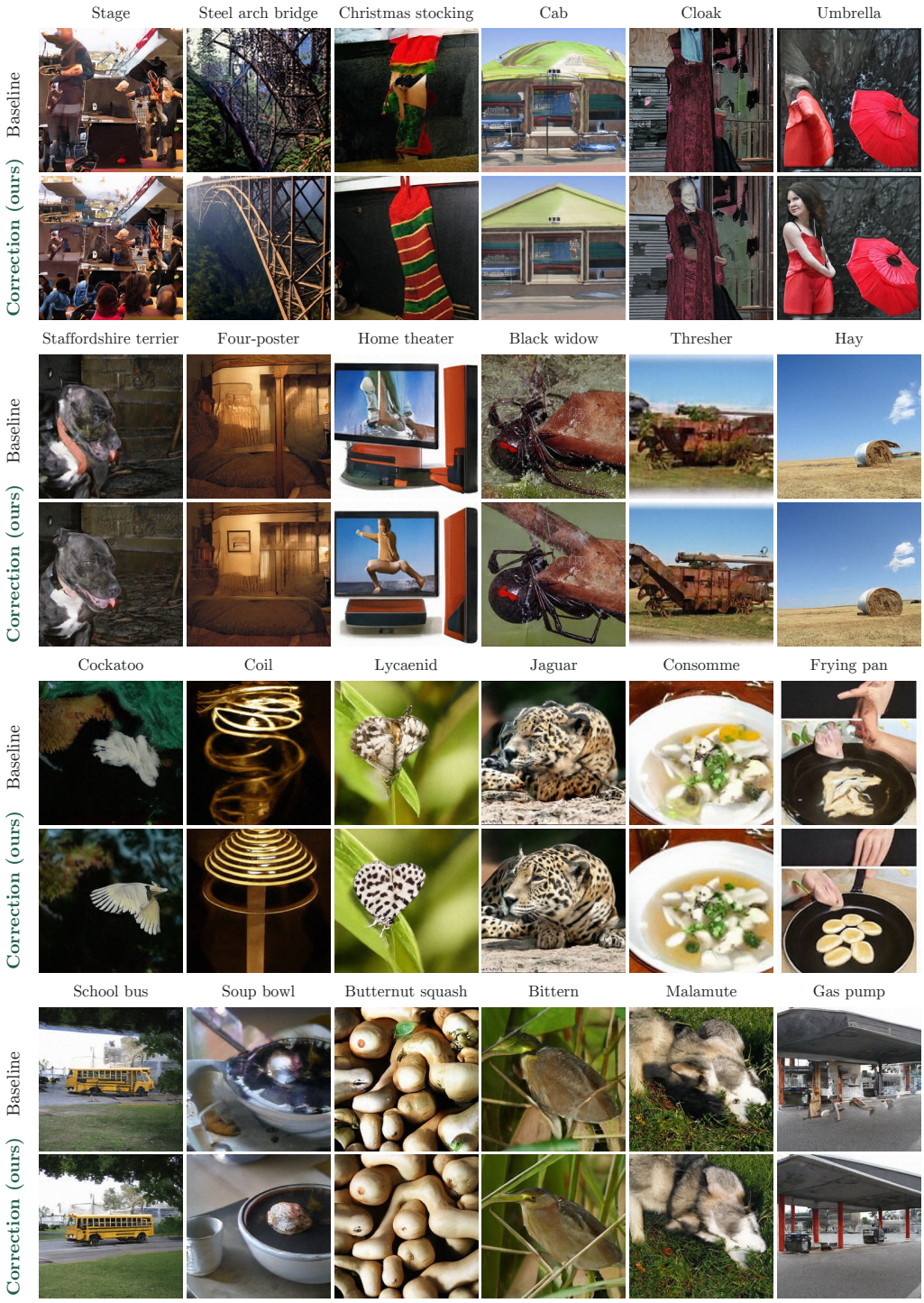


Figure 40: ***Uncurated* samples: early correction changes structure as well as appearance in JiT.** Within each pair of rows, baseline ($K = 0$) and correction ($K = 6$) share the same class and initial noise at matched NFEs.

E.5 UNCONDITIONAL GENERATION

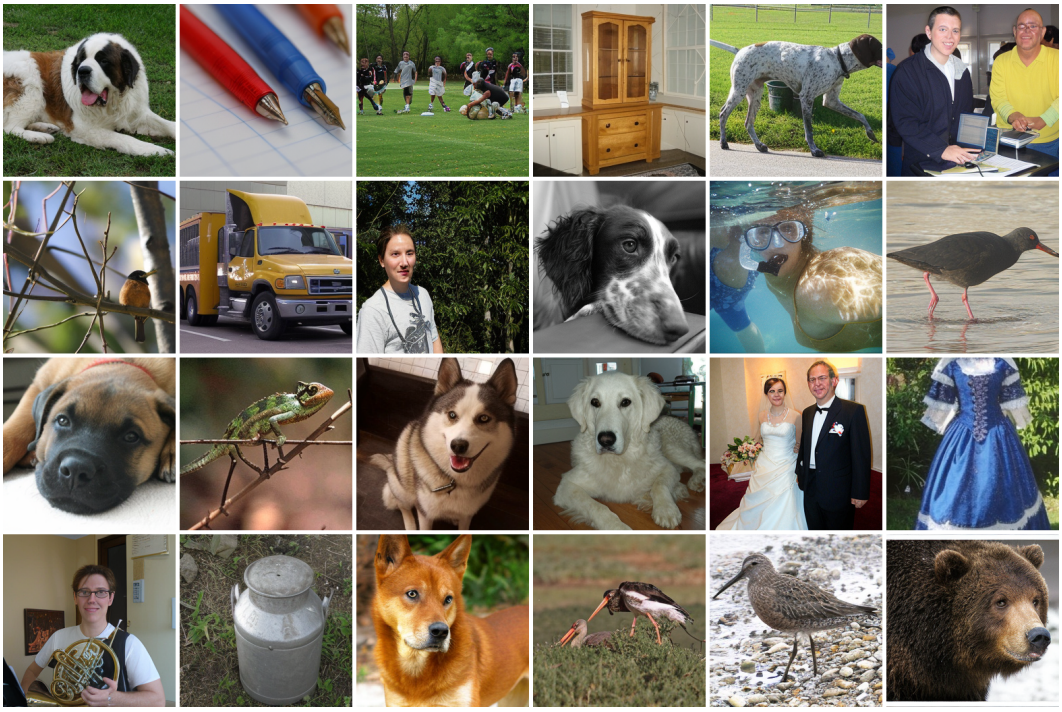


Figure 41: **High-quality unconditional generation without guidance.** Selected ImageNet samples from SiT_{REPA}-XL/2 with iterative correction.

E.6 Unconditional Generation from a Text-to-Image Model

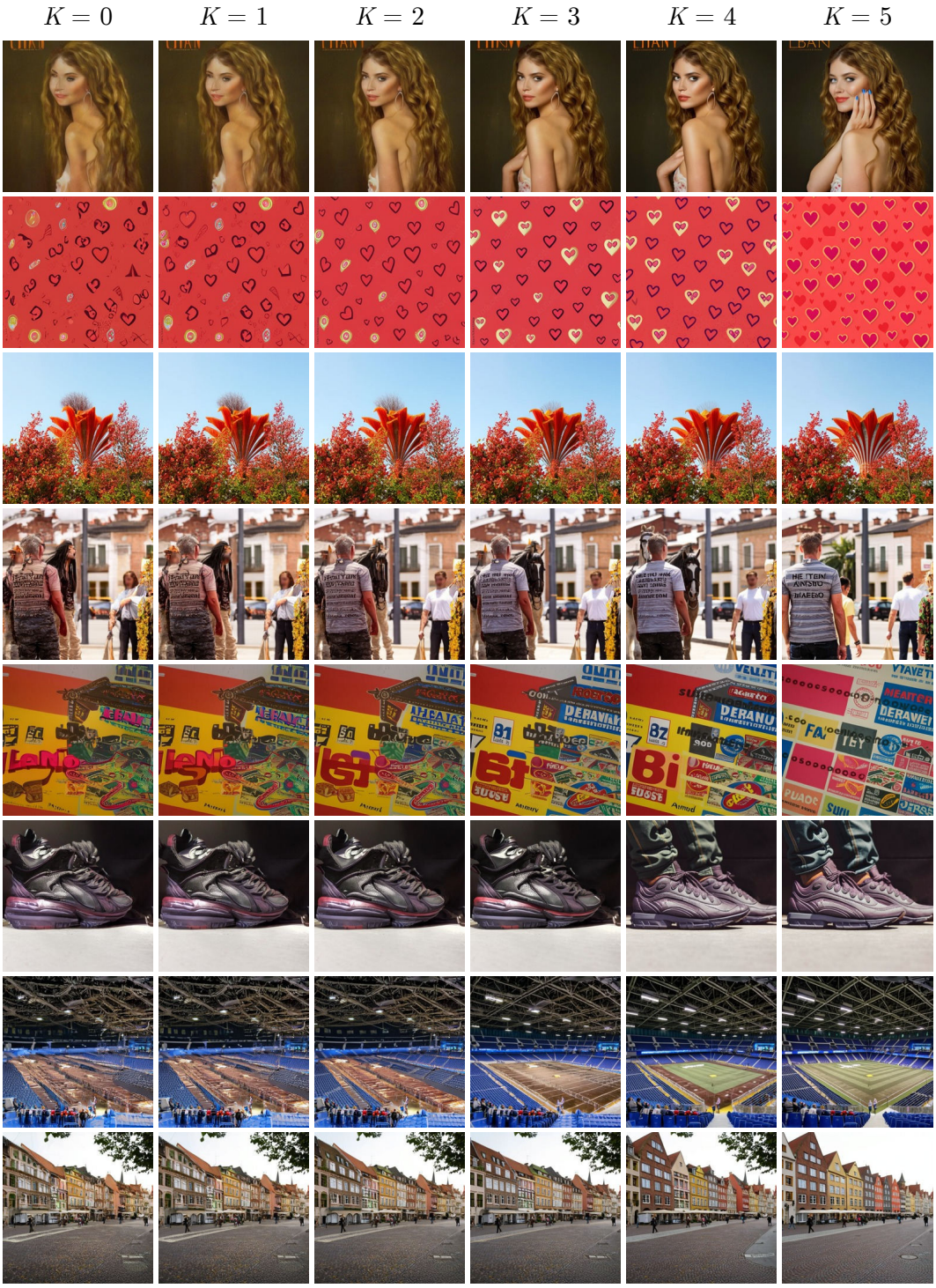


Figure 42: ***Uncurated* samples: unconditional generation supports iterative correction without text or CFG.** Each row shares its initial noise across $K = 0, \ldots, 5$. The checkpoint and denoising steps are fixed; larger K adds correction compute. All seeds are shown without selection.

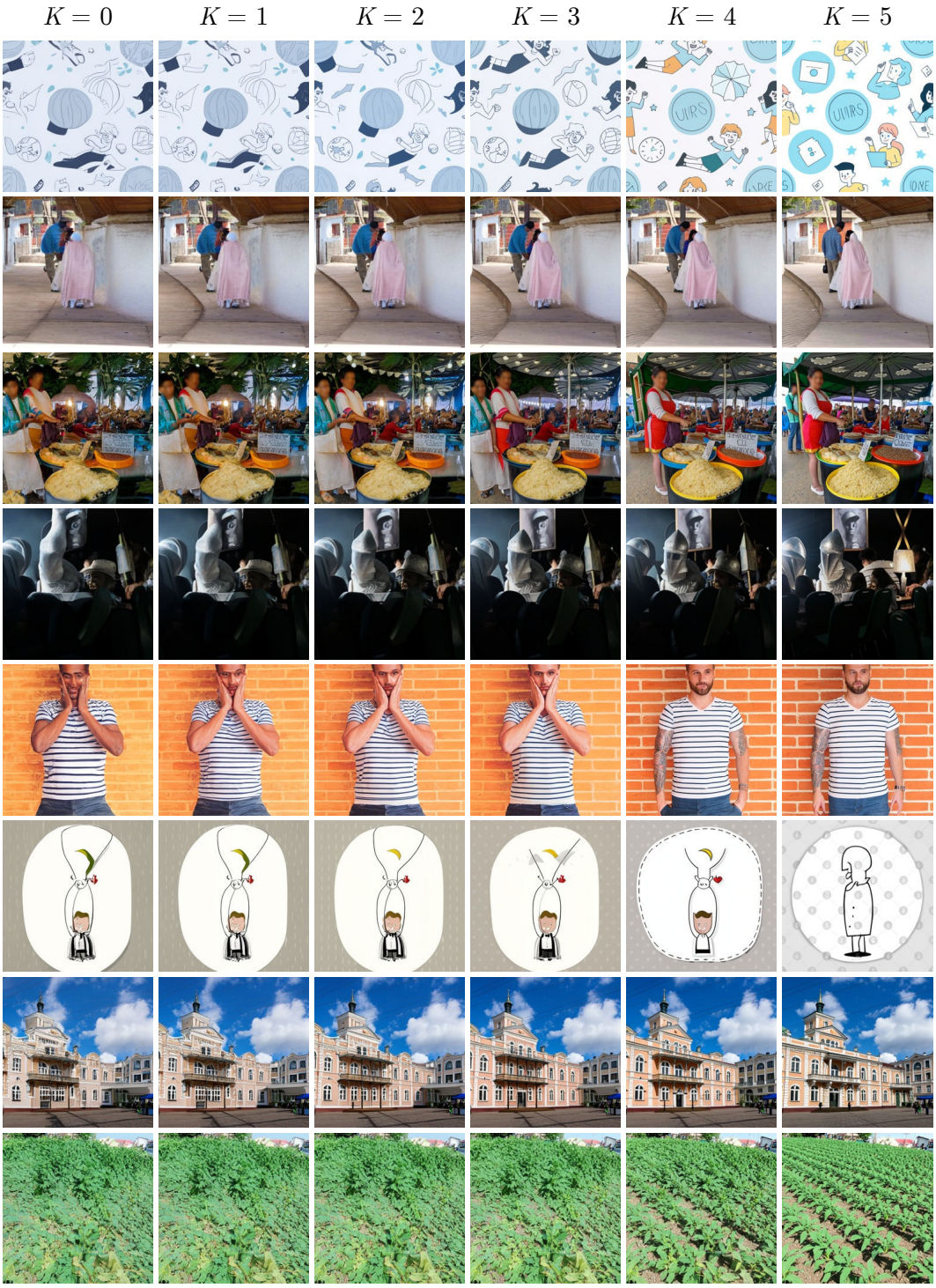


Figure 43: ***Uncurated* samples: unconditional generation supports iterative correction without text or CFG.** Each row shares its initial noise across $K = 0, \ldots, 5$. The checkpoint and denoising steps are fixed; larger K adds correction compute. All seeds are shown without selection.

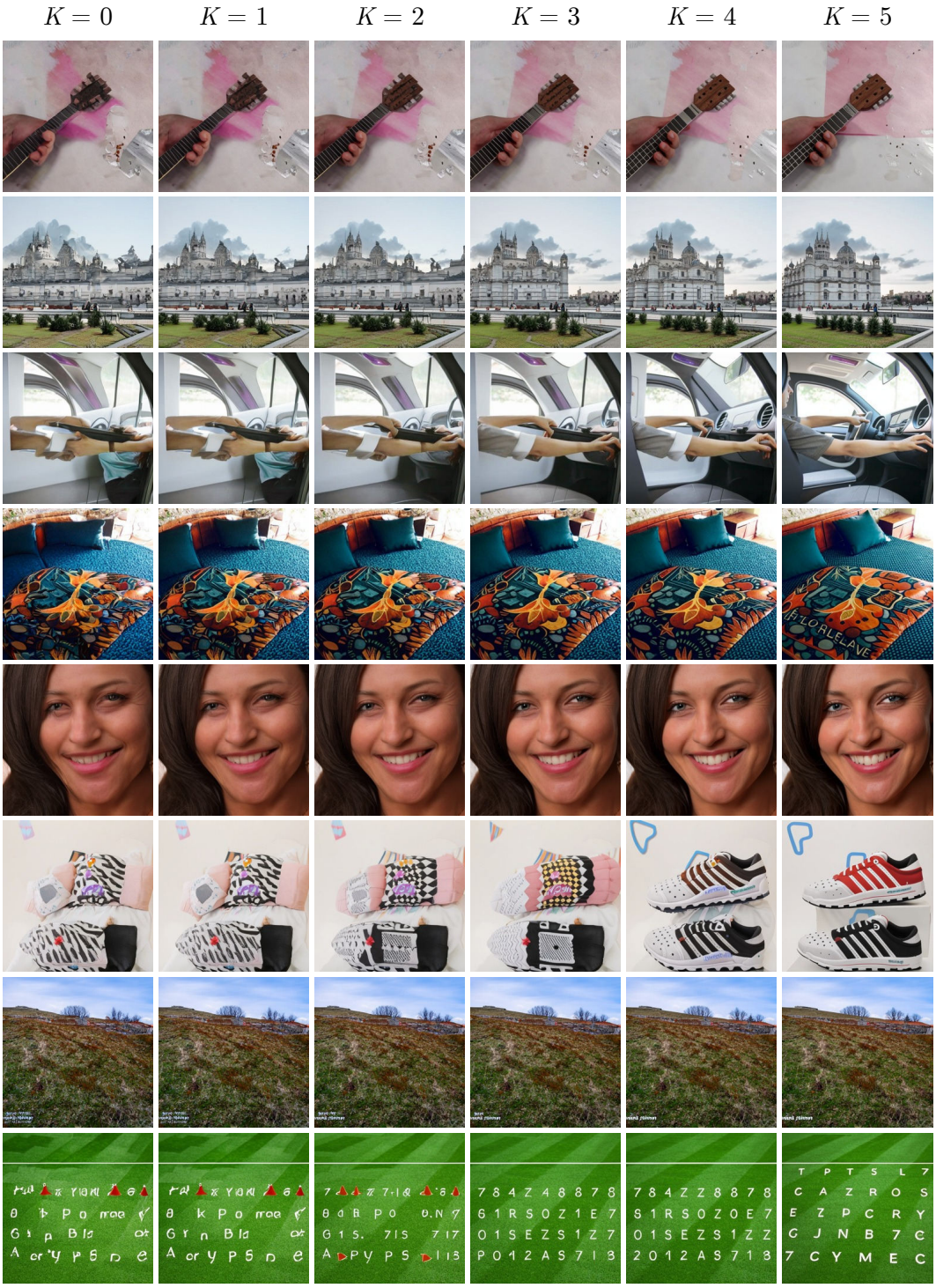


Figure 44: ***Uncurated* samples: unconditional generation supports iterative correction without text or CFG.** Each row shares its initial noise across $K = 0, \ldots, 5$. The checkpoint and denoising steps are fixed; larger K adds correction compute. All seeds are shown without selection.

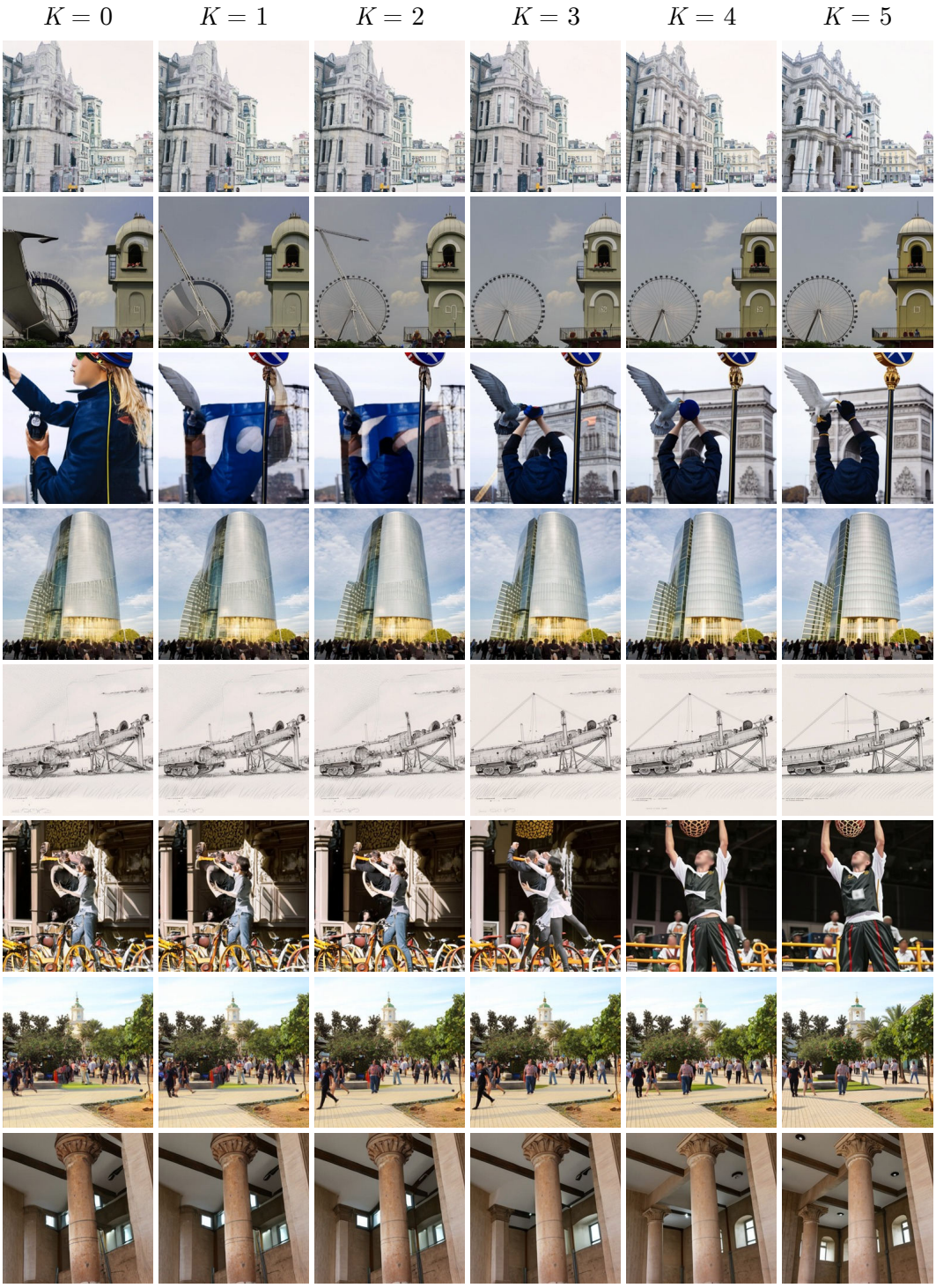


Figure 45: ***Uncurated* samples: unconditional generation supports iterative correction without text or CFG.** Each row shares its initial noise across $K = 0, \ldots, 5$. The checkpoint and denoising steps are fixed; larger K adds correction compute. All seeds are shown without selection.