FlowDPS : Flow-Driven Posterior Sampling for Inverse Problems

Supplementary Material

6. Derivation of DDIM Form

The stochastic part of the equation Eq. (14) is

$$\begin{aligned} &C_2(t)\tilde{\boldsymbol{x}}_{1|t} \\ &= C_2(t)(\sqrt{1-\eta_t}\hat{\boldsymbol{x}}_{1|t} + \sqrt{\eta_t}\boldsymbol{\epsilon}) \\ &= \sqrt{C_2(t)^2 - C_2(t)^2\eta_t}\hat{\boldsymbol{x}}_{1|t} + \sqrt{C_2(t)^2\eta_t}\boldsymbol{\epsilon}. \end{aligned} \tag{38}$$

If we set $k_t = C_2(t)\sqrt{\eta_t}$, this is equivalent to

$$\sqrt{C_2(t)^2 - k_t^2}\hat{\boldsymbol{x}}_{1|t} + \sqrt{k_t^2}\boldsymbol{\epsilon} \tag{39}$$

and the one step update is expressed as

$$\begin{aligned} &\boldsymbol{x}_{t+dt} \\ &= C_1(t)\hat{\boldsymbol{x}}_{0|t} + \sqrt{C_2(t)^2 - k_t^2}\hat{\boldsymbol{x}}_{1|t} + \sqrt{k_t^2}\boldsymbol{\epsilon} \end{aligned} \tag{40}$$

which is in the same form as DDIM [23] but with different coefficients $C_1(t) = a_t + \dot{a}_t dt = a_{t+dt}$ and $C_2(t) = b_t + \dot{b}_t dt = b_{t+dt}$.

7. Proofs

Proposition 1 (Tweedie Formula)**.** *The denoised and noisy estimate given $\boldsymbol{x}_t$ are given by*

$$\mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t] = \left[a_t - \dot{a}_t\frac{b_t}{\dot{b}_t}\right]^{-1}\left(\boldsymbol{x}_t - \frac{b_t}{\dot{b}_t}v_t(\boldsymbol{x}_t)\right)$$
$$\mathbb{E}[\boldsymbol{x}_1|\boldsymbol{x}_t] = \left[b_t - \dot{b}_t\frac{a_t}{\dot{a}_t}\right]^{-1}\left(\boldsymbol{x}_t - \frac{a_t}{\dot{a}_t}v_t(\boldsymbol{x}_t)\right)$$

Proof. According to Eq. (9), we have $v_t(\boldsymbol{x}_t|\boldsymbol{x}_0) = \dot{a}_t\boldsymbol{x}_0 + \dot{b}_t\boldsymbol{x}_1$. This leads to the representation of $\boldsymbol{x}_t$ with respect to the conditional flow velocity:

$$\begin{aligned} \boldsymbol{x}_t &= a_t\boldsymbol{x}_0 + b_t\boldsymbol{x}_1 \\ &= a_t\boldsymbol{x}_0 + b_t\left(\frac{v_t(\boldsymbol{x}_t|\boldsymbol{x}_0) - \dot{a}_t\boldsymbol{x}_0}{\dot{b}_t}\right) \\ &= \left(a_t - \dot{a}_t\frac{b_t}{\dot{b}_t}\right)\boldsymbol{x}_0 + \frac{b_t}{\dot{b}_t}v_t(\boldsymbol{x}_t|\boldsymbol{x}_0) \end{aligned} \tag{41}$$

Thus we have

$$\begin{aligned} \mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t] &= \left(a_t - \dot{a}_t\frac{b_t}{\dot{b}_t}\right)^{-1}\left(\boldsymbol{x}_t - \frac{b_t}{\dot{b}_t}\mathbb{E}[v_t(\boldsymbol{x}_t|\boldsymbol{x}_0)]\right) \\ &= \left(a_t - \dot{a}_t\frac{b_t}{\dot{b}_t}\right)^{-1}\left(\boldsymbol{x}_t - \frac{b_t}{\dot{b}_t}v_t(\boldsymbol{x}_t)\right) \end{aligned}$$

Similarly, we have

$$\begin{aligned} \boldsymbol{x}_t &= a_t\boldsymbol{x}_0 + b_t\boldsymbol{x}_1 \\ &= a_t\left(\frac{v_t(\boldsymbol{x}_t|\boldsymbol{x}_0) - \dot{b}_t\boldsymbol{x}_1}{\dot{a}_t}\right) + b_t\boldsymbol{x}_1 \\ &= \left(b_t - \dot{b}_t\frac{a_t}{\dot{a}_t}\right)\boldsymbol{x}_1 + \frac{a_t}{\dot{a}_t}v_t(\boldsymbol{x}_t|\boldsymbol{x}_0) \end{aligned} \tag{42}$$

which leads to

$$\begin{aligned} \mathbb{E}[\boldsymbol{x}_1|\boldsymbol{x}_t] &= \left(b_t - \dot{b}_t\frac{a_t}{\dot{a}_t}\right)^{-1}\left(\boldsymbol{x}_t - \frac{a_t}{\dot{a}_t}\mathbb{E}[v_t(\boldsymbol{x}_t|\boldsymbol{x}_0)]\right) \\ &= \left(b_t - \dot{b}_t\frac{a_t}{\dot{a}_t}\right)^{-1}\left(\boldsymbol{x}_t - \frac{a_t}{\dot{a}_t}v_t(\boldsymbol{x}_t)\right) \end{aligned}$$

□

Proposition 2. *Suppose the clean data manifold $\mathcal{M}$ is represented as an affine subspace and assume uniform distribution over $\mathcal{M}$. Then,*

$$J_\theta(\boldsymbol{x}_t) = \frac{\partial\hat{\boldsymbol{x}}_{0|t}}{\partial\boldsymbol{x}_t} = \frac{1}{a_t}\mathcal{P}_{\mathcal{M}} \tag{24}$$

where $\mathcal{P}_{\mathcal{M}}$ denotes the orthogonal projection to $\mathcal{M}$.

Proof. Using Eq. (11) and Eq. (18), we can express the Jacobian as

$$J_\theta(\boldsymbol{x}) = \frac{1}{a_t}\frac{\partial(\boldsymbol{x}_t - b_t^2\nabla_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t))}{\partial\boldsymbol{x}_t}. \tag{43}$$

Here, we will derive the score function $\nabla_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t)$ in a closed form solution with assumptions and prove the result.

From the definition of affine conditional flow,

$$\boldsymbol{x}_t = \psi_t(\boldsymbol{x}_1|\boldsymbol{x}_0) = a_t\boldsymbol{x}_0 + b_t\boldsymbol{x}_1, \tag{44}$$

where $\boldsymbol{x}_1 \sim \mathcal{N}(0, \boldsymbol{I}_d)$, we get the explicit expression of

$$p(\boldsymbol{x}_t|\boldsymbol{x}_0) = \frac{1}{(2\pi b_t^2)^{d/2}}\exp\left(-\frac{\|\boldsymbol{x}_t - a_t\boldsymbol{x}_0\|^2}{2b_t^2}\right). \tag{45}$$

Assume that the clean images are distribution on subspace $\mathcal{M}$ uniformly. To express this, we start from defining $p(\boldsymbol{x}_0)$ as a zero-mean Gaussian distribution with isotropic variance σ.

$$p(\boldsymbol{x}_0) = \frac{1}{(2\pi\sigma^2)^{l/2}}\exp\left(-\frac{\|\mathcal{P}_{\mathcal{M}}\boldsymbol{x}_0\|^2}{2\sigma^2}\right) \tag{46}$$

as $\mathcal{P}_M \boldsymbol{x}_0 = \boldsymbol{x}_0$. Considering the marginal density

$$p(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_t|\boldsymbol{x}_0)p(\boldsymbol{x}_0)d\boldsymbol{x}_0, \tag{47}$$

we have to compute $p(\boldsymbol{x}_t|\boldsymbol{x}_0)p(\boldsymbol{x}_0)$ that is

$$p(\boldsymbol{x}_t|\boldsymbol{x}_0)p(\boldsymbol{x}_0) = \frac{1}{(2\pi b_t^2)^{d/2}(2\pi\sigma^2)^{l/2}} \exp(-d(\boldsymbol{x}_t, \boldsymbol{x}_0)), \tag{48}$$

where

$$\begin{aligned} d(\boldsymbol{x}_t, \boldsymbol{x}_0) &= \frac{\|\boldsymbol{x}_t - a_t\boldsymbol{x}_0\|^2}{2b_t^2} + \frac{\|\mathcal{P}_{\mathcal{M}}\boldsymbol{x}_0\|^2}{2\sigma^2} \\ &= \frac{\|\mathcal{P}_{\mathcal{M}}^{\perp}\boldsymbol{x}_t\|^2}{2b_t^2} + \frac{\|\mathcal{P}_{\mathcal{M}}\boldsymbol{x}_t - a_t\boldsymbol{x}_0\|^2}{2b_t^2} + \frac{\|\mathcal{P}_{\mathcal{M}}\boldsymbol{x}_0\|^2}{2\sigma^2} \\ &= \frac{\|\mathcal{P}_{\mathcal{M}}^{\perp}\boldsymbol{x}_t\|^2 - c_t\|\mathcal{P}_{\mathcal{M}}\boldsymbol{x}_t\|^2}{2b_t^2} + \frac{\|\mathcal{P}_{\mathcal{M}}\boldsymbol{x}_0 - \frac{1-c_t}{a_t}\mathcal{P}_{\mathcal{M}}\boldsymbol{x}_t\|^2}{\sigma^2 c_t}, \end{aligned}$$

and

$$c_t = \frac{b_t^2}{b_t^2 + \sigma^2 + a_t^2}. \tag{49}$$

Therefore, after integrating out with respect to $\boldsymbol{x}_0$, we have

$$\log p(\boldsymbol{x}_t) = -\frac{\|\mathcal{P}_{\mathcal{M}}^{\perp}\boldsymbol{x}_t\|^2 - c_t\|\mathcal{P}_{\mathcal{M}}\boldsymbol{x}_t\|^2}{2b_t^2} + const., \tag{50}$$

leading to

$$\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t) = -\frac{\mathcal{P}_{\mathcal{M}}^{\perp}\boldsymbol{x}_t - c_t\mathcal{P}_{\mathcal{M}}\boldsymbol{x}_t}{b_t^2}. \tag{51}$$

By the assumption of uniform distribution in Eq. (46) with $\sigma \to \infty$, we have $c_t \to 0$. Therefore,

$$\lim_{\sigma\to\infty} \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t) = -\frac{1}{b_t^2}\mathcal{P}_{\mathcal{M}}^{\perp}\boldsymbol{x}_t \tag{52}$$

and we conclude that

$$\begin{aligned} J_\theta(\boldsymbol{x}_t) &= \frac{1}{a_t}\frac{\partial(\boldsymbol{x}_t - b_t^2\nabla_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t))}{\partial \boldsymbol{x}_t} && (53) \\ &= \frac{1}{a_t}\frac{\partial(\boldsymbol{x}_t - \mathcal{P}_{\mathcal{M}}^{\perp}\boldsymbol{x}_t)}{\partial \boldsymbol{x}_t} && (54) \\ &= \frac{1}{a_t}\frac{\partial\mathcal{P}_{\mathcal{M}}\boldsymbol{x}_t}{\partial \boldsymbol{x}_t} = \frac{1}{a_t}\mathcal{P}_{\mathcal{M}}. && (55) \end{aligned}$$

□

8. Implementation Details

In this section, we provide implementation details of the FlowDPS and baselines. For Stable Diffusion 3 and Flux.1, you can refer the pseudocode of FlowDPS in Algorithm 1.

Implementation with flow models For a fair comparison, we re-implement baselines that are proposed with score-based diffusion models. In the following, we provide details for each implementation.

PSLD [20] extends DPS [3] to latent diffusion models by introducing a novel regularization loss for the autoencoder. Since its gradient is computed with respect to the intermediate sample $\boldsymbol{x}_t$ during the reverse diffusion process, the same algorithm can be implemented using Euler's method without loss of generality.

ReSample incorporates data consistency into the reverse sampling process of LDMs by solving an optimization problem on some time steps. The key ideas of this framework is Stochastic Resampling, for renoising the optimized latent, and hard data consistency. We implement Resample for linear flow-based models by setting $\bar{\alpha}_t = \frac{(1-\sigma_t)^2}{\sigma_t^2+(1-\sigma_t)^2}$. Resample uses various techniques such as dividing the sampling process into three stages and separately using soft data consistency and hard data consistency. We find that adopting this same setting for our comparison produces extremely poor results, primarily because of the small number of ODE steps of the Euler solver. We empirically find that using skip step size of 1 to perform hard data consistency on all steps produces best results, and use this setting for comparison.

LatentDAPS proposes a noise annealing process to decouple consecutive samples in a sampling trajectory, which enables solvers to create errors made in earlier steps. We implement LatentDAPS from the official code, with modification of the solving process to use the linear flow-based backbone model StableDiffusion 3.0 [6] and Euler solver.

Hyper-parameter setting For all implementations, we use StableDiffusion 3.0 [6] as our baseline model. Also, we set the shift factor of time scheduler to 4.0.

- **PSLD** We set $\eta = 1.0$ and $\gamma = 0.1$ by following the original paper setting, and use 200 NFEs as in [8].
- **Resample** We use the same resampling hyperparameter $\gamma\left(\frac{1-\bar{\alpha}_{t-1}}{\bar{\alpha}_t}\right)\left(1-\frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}\right)$ with $\gamma = 40$ as proposed in the original paper, reparameterizing $\bar{\alpha}_t$ as previously explained. Skip step size is set to 1 to perform sufficient hard data consistency steps.
- **LatentDAPS** We use ODE solver steps $N_{ODE} = 5$ and annealing scheduler $N_A = 28$, resulting in a total NFE of 120. Total step number N in Langevin Dynamics is set to 50, following the settings of the original paper [27].
- **FlowChef** For a constant step size for the FlowChef, we find the best configuration by grid search with 100 images. In consequence, we set the step size to 200 for the super-resolution tasks and 50 for the deblurring tasks.
- **FlowDPS** For data consistency optimization, we use 3 steps of gradient descent with step size 15 for all tasks.

Algorithm 1 Algorithm of FlowDPS (SD3.0, FLUX)

Require: Measurement $\boldsymbol{y}$, Linear operator $\boldsymbol{A}$, Pre-trained flow-based model $\boldsymbol{v}_\theta$, VAE encoder and Decoder $\mathcal{E}, \mathcal{D}$, Text embeddings $c_\varnothing, c$, CFG scale λ, Stochasticity level η, Noise Schedule σ_t

$\boldsymbol{z} \sim \mathcal{N}(0, \boldsymbol{I}_d)$
for $t : 1 \to 0$ **do**
  $\boldsymbol{v}_t(\boldsymbol{z}) \leftarrow \boldsymbol{v}_\theta(\boldsymbol{z}, c_\varnothing) + \lambda(\boldsymbol{v}_\theta(\boldsymbol{z}, c) - \boldsymbol{v}_\theta(\boldsymbol{z}, c_\varnothing))$
  $\hat{\boldsymbol{z}}_{0|t} \leftarrow \boldsymbol{z} - \sigma_t \boldsymbol{v}_t(\boldsymbol{z})$
  $\hat{\boldsymbol{z}}_{1|t} \leftarrow \boldsymbol{z} + (1 - \sigma_t)\boldsymbol{v}_t(\boldsymbol{z})$
  ▷ 1. Likelihood Gradient
  $\hat{\boldsymbol{z}}_{0|t}(\boldsymbol{y}) \leftarrow \arg\min_{\boldsymbol{z}} \|\boldsymbol{y} - \boldsymbol{A}\mathcal{D}(\boldsymbol{z})\|^2$.
  $\tilde{\boldsymbol{z}}_{0|t} \leftarrow \sigma_t \hat{\boldsymbol{z}}_{0|t}(\boldsymbol{y}) + (1 - \sigma_t)\hat{\boldsymbol{z}}_{0|t}$
  ▷ 2. Stochasticity
  $\epsilon \sim \mathcal{N}(0, \boldsymbol{I}_d)$
  $\tilde{\boldsymbol{z}}_{1|t} \leftarrow \sqrt{\sigma_{t+dt}}\boldsymbol{z}_{1|t} + \sqrt{1 - \sigma_{t+dt}}\epsilon$
  ▷ 3. Euler update
  $\boldsymbol{z} \leftarrow (1 - \sigma_{t+dt})\tilde{\boldsymbol{z}}_{0|t} + \sigma_{t+dt}\tilde{\boldsymbol{z}}_{1|t}$
end for

9. Ablation Study

9.1. Analysis on β_t and γ_t

In FlowDPS, we interpolate $\hat{\boldsymbol{z}}_{0|t}$ and $\hat{\boldsymbol{z}}_{0|t}(\boldsymbol{y})$ with coefficient γ_t to ensure the data consistency update does not lead to excessive divergence for the flow model's trajectory. For the selection of $\gamma_t = \sigma_t$, we refer to the progression of the adaptive step size β_t in Eq. (28). Specifically, our likelihood gradient is applied to clean estimation as

$$\hat{\boldsymbol{x}}_{0|t}(\boldsymbol{y}) = \hat{\boldsymbol{x}}_{0|t} - \beta_t \nabla_{\hat{\boldsymbol{x}}_{0|t}} \log p(\boldsymbol{y}|\hat{\boldsymbol{x}}_{0|t})$$

where $\beta_t = \frac{\zeta_t}{a_t}\frac{dt}{C_1(t)}$. For a linear flow with $\sigma_t = t$,

$$\frac{\zeta_t}{a_t} = \dot{a}_t \frac{b_t}{a_t}\left(\frac{\dot{b}_t}{\dot{a}_t} - \frac{b_t}{a_t}\right) = \frac{\sigma_t}{(1 - \sigma_t)^2} \tag{56}$$

and β_t is expressed as

$$\beta_t = \frac{dt\sigma_t}{(1 - \sigma_t)^2(1 - \sigma_{t+dt})} \tag{57}$$

with $dt = \sigma_{t+dt} - \sigma_t < 0$. Figure 6 illustrates the progress of $-\beta_t$ during the sampling process. As we mentioned in the main paper, it rapidly decreases to zero which leads to higher stepsize for likelihood gradient in the early stage. In our method, we imitate the theory-driven behavior of step size for likelihood gradient by introducing interpolation coefficient γ that emphasizes data consistency (i.e. likelihood gradient) in the early stage.

9.2. CFG scale and NFE

CFG scale λ is an important factor for generating high-frequency details based on provided text prompts. When using low λ, the model fails to capture fine details, producing blurry results. Using high λ enables the model to better generate fine details, but can also guide sample generation in a completely wrong direction when excessively high. Figure 7 in the Appendix shows how when using higher NFEs, the model requires more guidance for production of high-frequency details, causing higher values of λ to perform better.

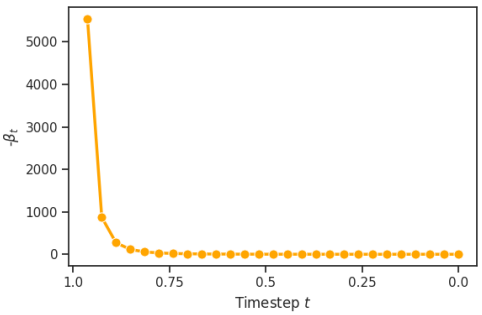

Figure 6. Evolution of $-\beta_t$ during sampling.

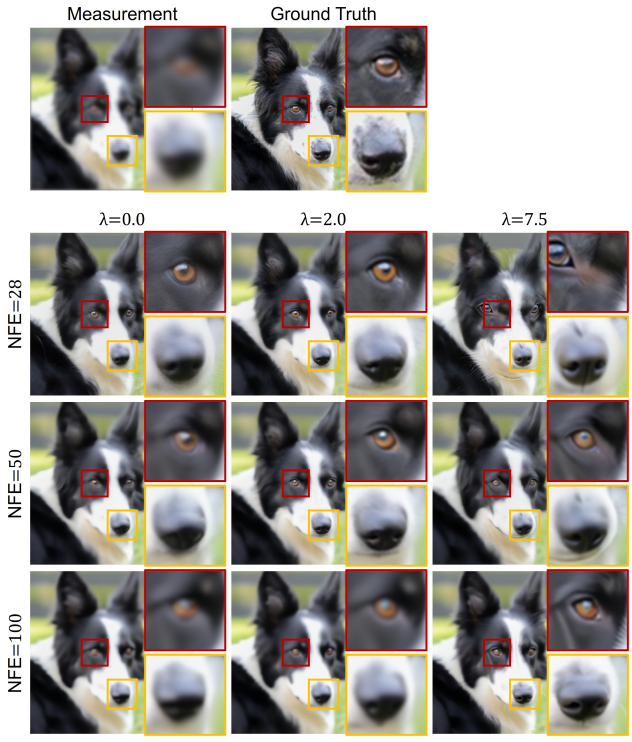

Figure 7. Relation between NFE and CFG scale λ. For varying NFEs (28, 50, 100), we study the impact of λ (0.0, 2.0, 7.5).

10. Non-Linear Inverse Problems

We provide additional results on using FlowDPS to solve non-linear inverse problems. Specifically, we use Latent

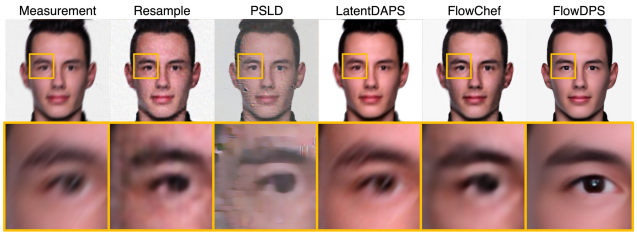

Figure 8. Qualitative comparison for non-uniform deblur.

	Resample	LatentDAPS	FlowChef	FlowDPS
PSNR ↑	<u>24.24</u>	22.42	23.17	**25.48**
FID ↓	128.7	<u>72.18</u>	110.5	**61.03**

Table 3. Quantitative comparison for the non-uniform deblurring task (512×512). Various inverse problem solvers are compared on 1k images of FFHQ. **Bold**: best, <u>Underline</u>: second best.

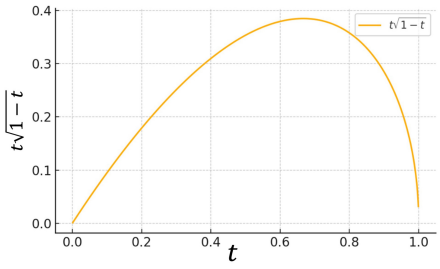

Figure 9. Amount of stochastic noise added over time steps t.

FlowDPS (with Stable Diffusion 3.0) to solve non-uniform deblur on 512×512 images. Qualitative and quantitative comparison is given in Figure 8 and Table 3, respectively. Results prove that FlowDPS is effective in solving any inverse problem with differentiable forward operators, including non-linear inverse problems.

11. Importance of Stochastic Noise

We provide further insight into the performance improvement brought about by stochastic noise. Stochastic noise aids the escape from local minima during the sampling process via a cleverly designed coefficient. Figure 9 is a plot of the coefficient multiplied to stochastic noise during sampling with linear flow. The coefficient increases early ($t \approx 1$) to aid exploration, then decreases as data consistency establishes low-frequency content and sampling converges. Such adaptive behavior significantly improves performance.

12. Conditional Generation with FlowDPS

Beyond the scope of inverse problems, we show how a correct connection between the flow framework and posterior sampling opens up interesting prospects for style/text guided generation. In Figure 10 we confirm the potential of FlowDPS via experiments related to FreeDoM, in which the forward operator is the Gram matrix of the intermediate feature of the CLIP encoder. Results show that our method successfully performs conditional generation, thus proving the wide applicability of FlowDPS.

13. Runtime Comparison

Among baselines, PSLD takes the longest time as it requires computation of the Jacobian in terms of the transformer denoiser. In contrast, ReSample, FlowChef, and FlowDPS do not perform this computation due to approximation of the Jacobian, making them relatively efficient. Theoretically, the main computational bottleneck for these methods is backpropagation through the decoder and all other computations are relatively small, giving nearly identical runtime. Although LatentDAPS does not require backpropagation through a neural network, repeatedly solving the flow ODE and performing Langevin dynamics multiple times to achieve sufficient performance induces longer runtime compared to FlowDPS.

14. Additional Results

14.1. Higher pixel resolution

The diffusion process defined in latent space enables control over the generation process through various conditions, such as text, while also improving computational efficiency. Specifically, latent flow models like Stable Diffusion 3.0 can generate images with resolutions exceeding 1K pixels and support various aspect ratios beyond standard square formats. We leverage this advantage of latent flow models to solve inverse problems, demonstrating the practical applicability of FlowDPS in real-world scenarios. In Figure 11, we address the motion blur problem using 1024×1408 images, obtained by cropping the central region of the DIV2K training set. For the text guidance, we use text description of the image extracted by LLaVA [13]. As demonstrated in the main experiment, FlowDPS successfully solves inverse problems defined in higher pixel resolutions without loss of generality.

14.2. Inverse problem solving with FLUX

The FlowDPS framework is designed for general affine conditional flows, enabling the construction of an inverse problem solver with various flow models. In our main experiment, we demonstrate the efficiency of FlowDPS using a pre-trained linear conditional flow model, Stable Diffusion 3. To further illustrate the generality of FlowDPS, we also incorporate another open-source linear conditional flow model, FLUX. Specifically, we set $\eta_t = 0$ and use 10 iterations for data consistency optimization. The step size is set to 7.0 for super-resolution and 12.0 for deblurring, while all other settings remain consistent with Stable

Figure 10. Style/text guided conditional generation with FlowDPS for 768×768 images using Stable Diffusion 3.0. All settings are equal to the settings we use for solving inverse problems, except for a CFG scale of 7.5 (inverse problems are solved with CFG scale of 2.0).

Diffusion 3. Figure 12 and 13 present the reconstruction results of FlowDPS with FLUX 1.0-schnell [10] on DIV2K images. As in the main experiment, we utilize text prompts extracted from measurements using DAPE [26]. Regardless of the backbone model, FlowDPS effectively solves inverse problems.

14.3. Additional Qualitative comparisons

In this section, we further provide qualitative comparisons for the four inverse problems across the AFHQ and FFHQ datasets. Figures 14-21 show that FlowDPS consistently achieves promising reconstruction results and outperforms all the baselines.

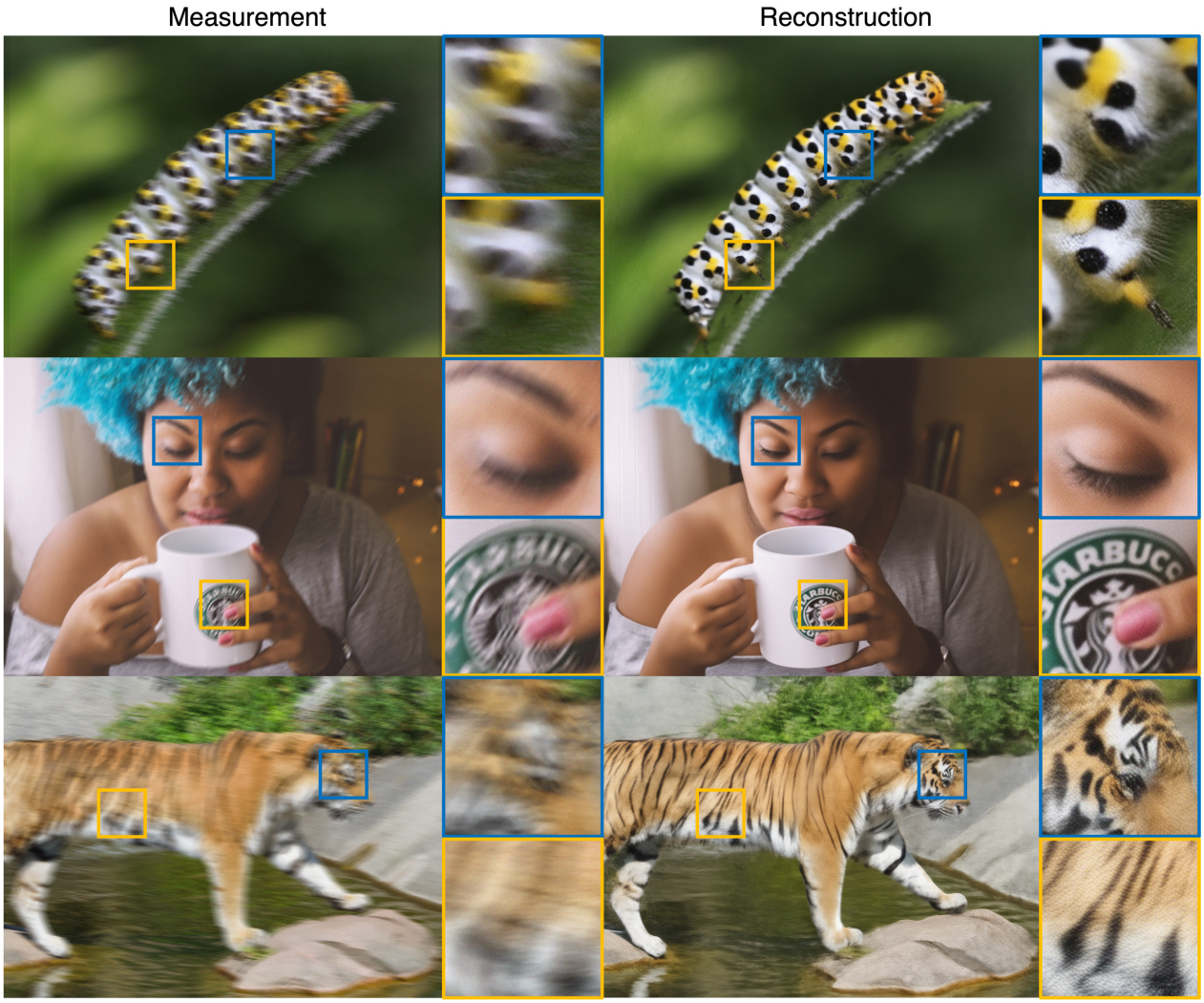

Figure 11. Motion deblur results by FlowDPS with SD3.0 for DIV2K images (1024x1408)

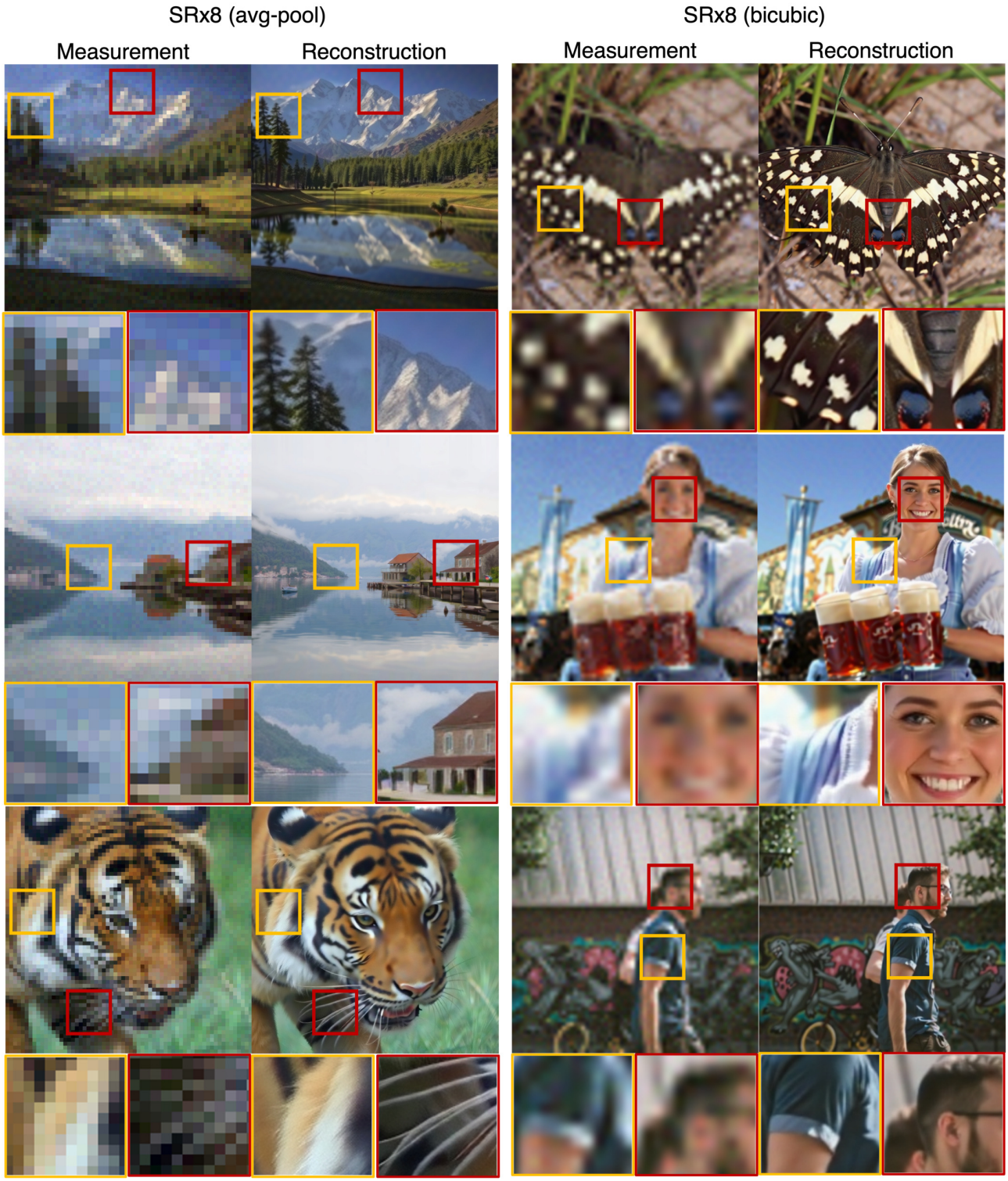

Figure 12. Super-resolution (x8) results by FlowDPS with Flux for DIV2K images (512x512)

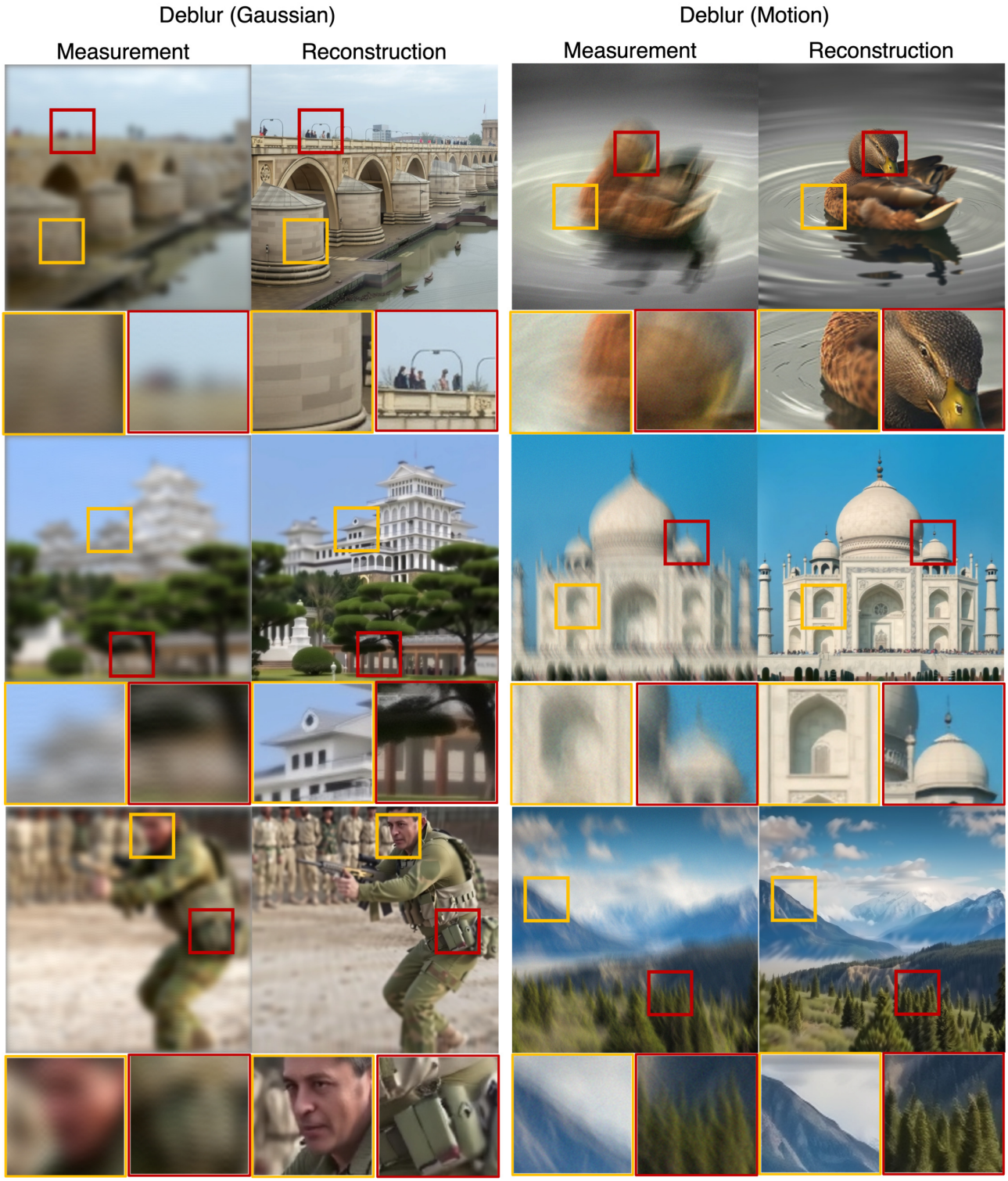

Figure 13. Deblurring results by FlowDPS with Flux for DIV2K images (512x512)

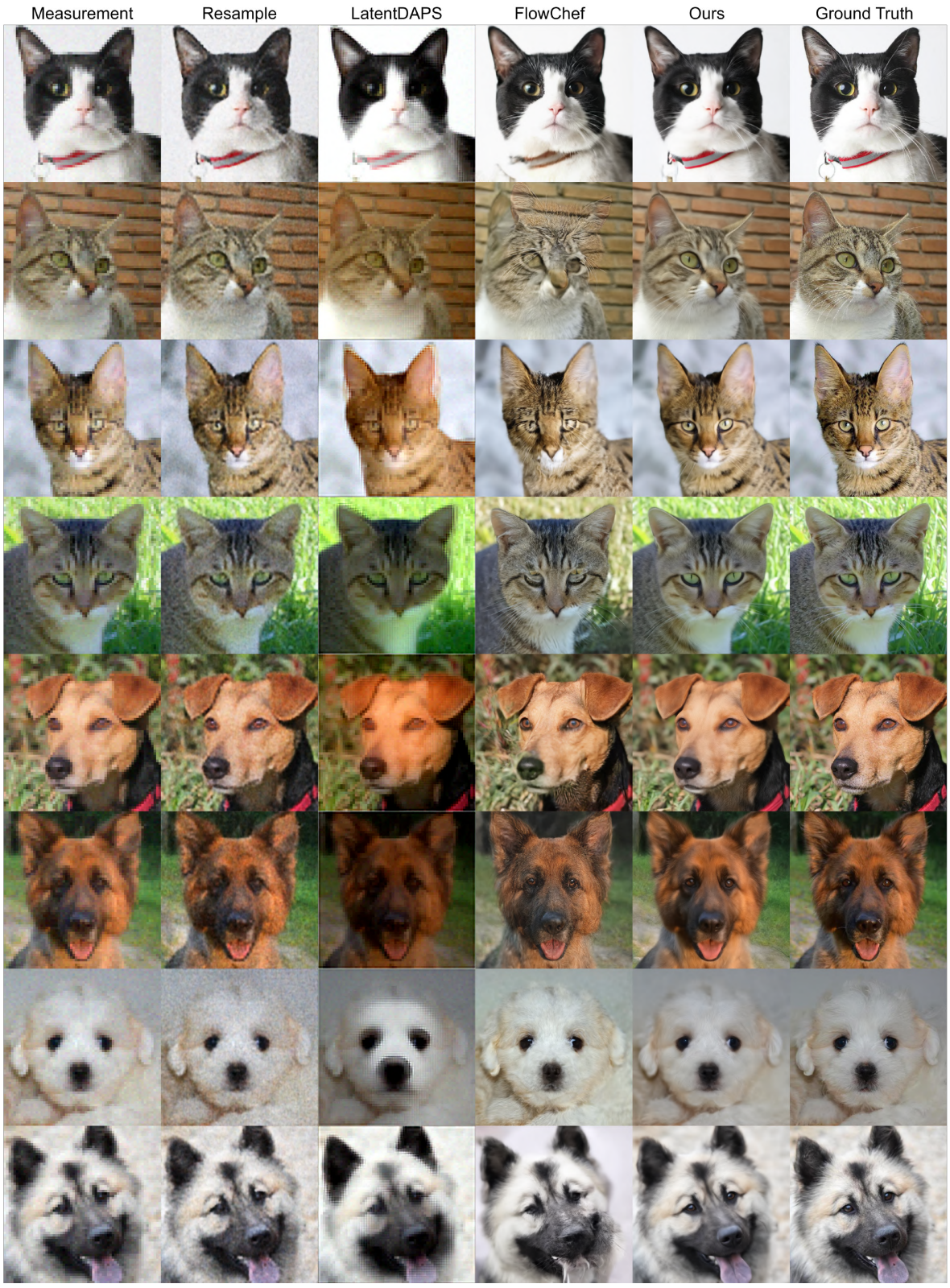

Figure 14. Qualitative comparison for x12 super-resolution from average pooling on the AFHQ dataset.

Figure 15. Qualitative comparison for x12 super-resolution from bicubic downsampling on the AFHQ dataset.

Figure 16. Qualitative comparison for Gaussian deblurring on the AFHQ dataset.

Figure 17. Qualitative comparison for motion deblurring on the AFHQ dataset.

Figure 18. Qualitative comparison for x12 super-resolution from average pooling on the FFHQ dataset.

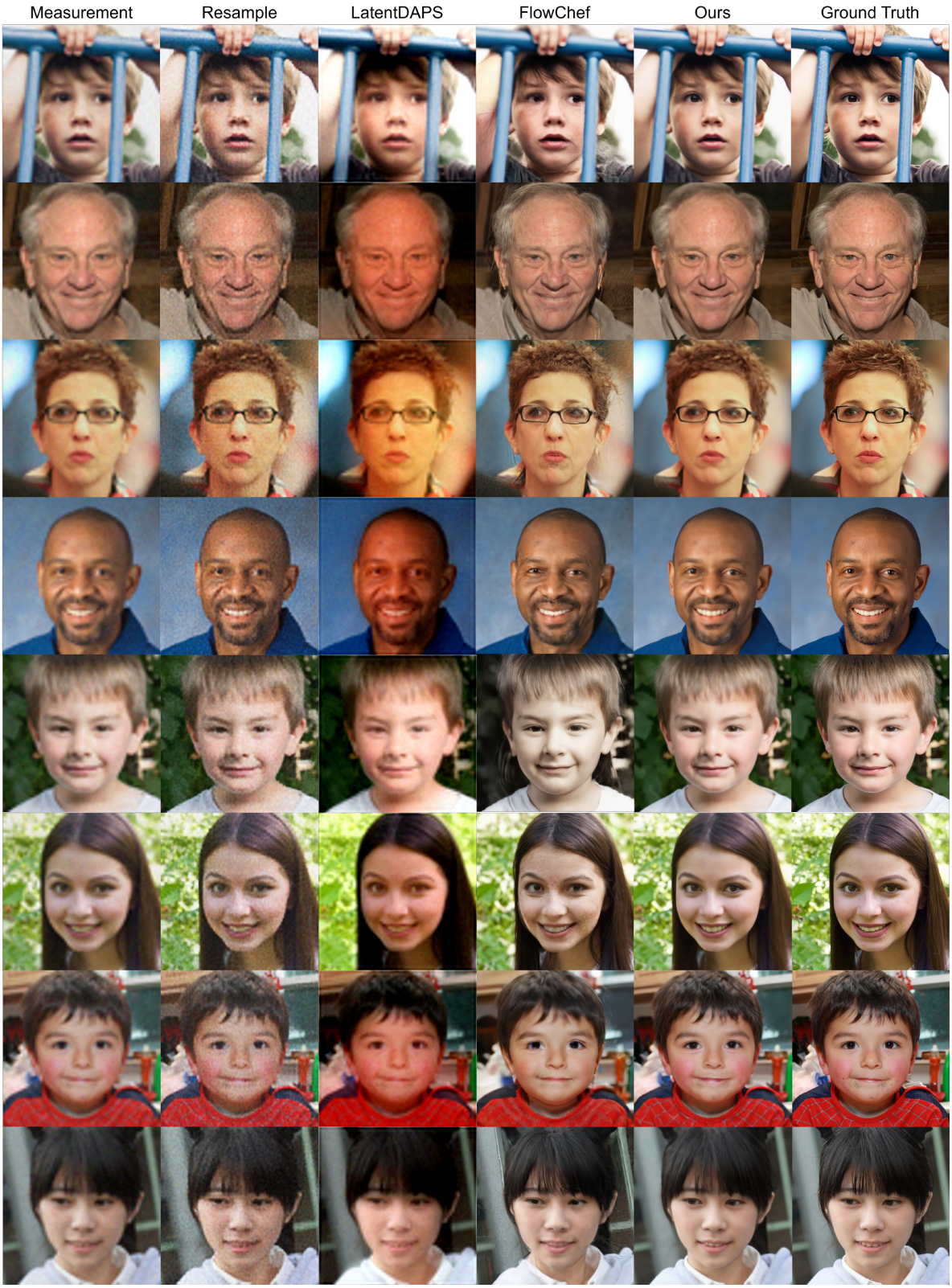

Figure 19. Qualitative comparison for x12 super-resolution from bicubic downsampling on the FFHQ dataset.

Figure 20. Qualitative comparison for Gaussian deblurring on the FFHQ dataset.

Figure 21. Qualitative comparison for motion deblurring on the FFHQ dataset.