Uncertainty-guided Perturbation for Image Super-Resolution Diffusion Model

Supplementary Material

In this supplementary material, we present more implementation details of the proposed UPSR method and additional visual results. Firstly, we introduce the details of the sampling process of UPSR in Sec. A. Secondly, we present the details of weighting coefficient $u(\cdot)$ in Sec. B. Then, comparisons to several existing methods are made in Sec. C and Sec. D. Lastly, additional visual examples are shown in Sec. E.

A. Details of the Sampling Process

Derivation of Eq. 7, 8: As discussed in Sec. 3.3, we apply region-specific weighting coefficient to the noise level based on the sampling process of ResShift [44] and replace $\boldsymbol{x}_t = \boldsymbol{x}_{t-1} + \alpha_t(\boldsymbol{y}_0 - \boldsymbol{x}_0) + \kappa\sqrt{\alpha_t}\boldsymbol{\xi}_t$ with $\boldsymbol{x}_t = \boldsymbol{x}_{t-1} + \alpha_t(\boldsymbol{y}_0 - \boldsymbol{x}_0) + \kappa w_u(\boldsymbol{y}_0)\sqrt{\alpha_t}\boldsymbol{\xi}_t$, where $\boldsymbol{\xi}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ is from independently distributed Gaussian distribution and α is a scaling factor. Therefore, we rewrite the forward transition distribution as:

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0, \boldsymbol{y}_0) = \mathcal{N}\left(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1} + \alpha_t(\boldsymbol{y}_0 - \boldsymbol{x}_0), \kappa^2 w_u(\boldsymbol{y}_0)^2 \alpha_t \boldsymbol{I}\right). \tag{10}$$

Areas with higher uncertainty, e.g. edge and texture areas, are assigned larger weighting coefficients $w_u(\boldsymbol{y}_0)$ and therefore greater noise $\kappa w_u(\boldsymbol{y}_0)\sqrt{\alpha_t}\boldsymbol{\xi}_t$. Meanwhile, $\boldsymbol{x}_t$ can be reparameterized as:

$$\begin{aligned} \boldsymbol{x}_t &= \boldsymbol{x}_0 + \sum_{i=1}^{t}(\boldsymbol{x}_i - \boldsymbol{x}_{i-1}) = \boldsymbol{x}_0 + \sum_{i=1}^{t}(\alpha_t(\boldsymbol{y}_0 - \boldsymbol{x}_0) + \kappa w_u(\boldsymbol{y}_0)\sqrt{\alpha_t}\boldsymbol{\xi}_t) \\ &= \boldsymbol{x}_0 + \eta_t(\boldsymbol{y}_0 - \boldsymbol{x}_0) + \kappa w_u(\boldsymbol{y}_0)\sqrt{\eta_t}\boldsymbol{\xi}_t, \end{aligned} \tag{11}$$

leading to the marginal distribution at time t as:

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_0, \boldsymbol{y}_0) = \mathcal{N}\left(\boldsymbol{x}_t \mid \boldsymbol{x}_0 + \eta_t(\boldsymbol{y}_0 - \boldsymbol{x}_0), \kappa^2 w_u(\boldsymbol{y}_0)^2 \eta_t \boldsymbol{I}\right). \tag{12}$$

According to Bayes's theorem, the reverse transition distribution can be written as

$$\begin{aligned} q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0, \boldsymbol{y}_0) &= \frac{q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0, \boldsymbol{y}_0) q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0, \boldsymbol{y}_0)}{q(\boldsymbol{x}_t \mid \boldsymbol{x}_0, \boldsymbol{y}_0)} \\ &\propto q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0, \boldsymbol{y}_0) q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0, \boldsymbol{y}_0). \end{aligned} \tag{13}$$

Incorporating Eq. 10, Eq. 12, and Eq. 13, we consider the distribution at each pixel i as:

$$\begin{aligned} q(x_{t-1}^i | x_t^i, x_0^i, y_0^i) &\propto q(x_t^i \mid x_{t-1}^i, x_0^i, y_0^i) q(x_{t-1}^i \mid x_0^i, y_0^i) \\ &\propto \exp\left(-\frac{(x_t^i - \mu_1^i)^2}{2\kappa^2 w_u(y_0^i)^2 \alpha_t}\right) \exp\left(-\frac{(x_{t-1}^i - \mu_2^i)^2}{2\kappa^2 w_u(y_0^i)^2 \eta_{t-1}}\right) \\ &= \exp\left(-\frac{\eta_{t-1}(x_{t-1}^i - \mu_3^i)^2 + \alpha_t(x_{t-1}^i - \mu_2^i)^2}{2\kappa^2 w_u(y_0^i)^2 \alpha_t \eta_{t-1}}\right) \\ &= \exp\left(-\frac{\eta_t(x_{t-1}^i)^2 - x_{t-1}^i(\eta_{t-1}\mu_3^i + \alpha_t\mu_2^i) - (\eta_{t-1}\mu_3^i + \alpha_t\mu_2^i)x_{t-1}^i}{2\kappa^2 w_u(y_0^i)^2 \alpha_t \eta_{t-1}}\right), \end{aligned} \tag{14}$$

where $\mu_1^i = x_{t-1}^i + \alpha_t(y_0^i - x_0^i)$, $\mu_2^i = x_0^i + \eta_{t-1}(y_0^i - x_0^i)$, and $\mu_3^i = (x_t^i - \mu_1^i) + x_{t-1}^i = x_t^i - \alpha_t(y_0^i - x_0^i)$. Next, we further simplify Eq. 14 to:

$$\begin{aligned} q(x_{t-1}^i | x_t^i, x_0^i, y_0^i) &\propto \exp\left(-\frac{(x_{t-1}^i - \frac{1}{\eta_t}(\eta_{t-1}\mu_3^i + \alpha_t\mu_2^i))^2}{2\kappa^2 w_u(y_0^i)^2 \alpha_t \eta_{t-1}/\eta_t} + \text{constant}\right) \\ &= \exp\left(-\frac{(x_{t-1}^i - (\frac{\eta_{t-1}}{\eta_t}x_t^i + \frac{\alpha_t}{\eta_t}x_0^i))^2}{2\kappa^2 w_u(y_0^i)^2 \alpha_t \eta_{t-1}/\eta_t} + \text{constant}\right). \end{aligned} \tag{15}$$

Therefore, we present the reverse transition distribution in Eq. 8 based on Eq. 15:

$$
\begin{aligned}
q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0, \boldsymbol{y}_0) &= \prod_i q(x_{t-1}^i | x_t^i, x_0^i, y_0^i) \\
&= \mathcal{N}\left(\boldsymbol{x}_{t-1} \mid \frac{\eta_{t-1}}{\eta_t}\boldsymbol{x}_t + \frac{\alpha_t}{\eta_t}\boldsymbol{x}_0, \kappa^2 w_u(\boldsymbol{y}_0)^2 \frac{\eta_{t-1}}{\eta_t}\alpha_t \boldsymbol{I}\right).
\end{aligned} \tag{16}
$$

Algorithm 1 Training procedure of UPSR.

Require: Diffusion model $f_\theta(\cdot)$, pre-trained SR network $g(\cdot)$
Require: Paired training dataset (X, Y)
while not converged **do**
  sample $\boldsymbol{x}_0, \boldsymbol{y}_0 \sim (X, Y)$
  sample $t \sim U(1, T)$
  compute $g(\boldsymbol{y}_0)$
  $\boldsymbol{\psi}_{est}(\boldsymbol{y}_0) = \frac{1}{2}|g(\boldsymbol{y}_0) - \boldsymbol{y}_0|$
  $w_u(\boldsymbol{y}_0) = u(\boldsymbol{\psi}_{est}(\boldsymbol{y}_0))$
  sample $\epsilon \sim \mathcal{N}(\boldsymbol{0}, \kappa^2 \eta_t w_u(\boldsymbol{y}_0)^2 \boldsymbol{I})$
  $\boldsymbol{x}_t = \boldsymbol{x}_0 + \eta_t(\boldsymbol{y}_0 - \boldsymbol{x}_0) + \epsilon$
  $\mathcal{L}(\theta) = \sum_t \left[||f_\theta(\boldsymbol{x}_t, \boldsymbol{y}_0, g(\boldsymbol{y}_0), t) - \boldsymbol{x}_0||_2^2 + \lambda L_{per}(f_\theta(\boldsymbol{x}_t, \boldsymbol{y}_0, g(\boldsymbol{y}_0), t), \boldsymbol{x}_0)\right]$
  Take a gradient descent step on $\nabla_\theta \mathcal{L}(\theta)$
end while
return Converged diffusion model $f_\theta(\cdot)$.

Details of the training procedure. We present the detailed training pipeline of the proposed UPSR method in Alg. 1.

B. Implementation of the Weighting Coefficient for Uncertainty-guided Perturbation

As a supplement to Sec. 3.3, we model the relationship between the weighting coefficient of region-specific perturbation and the uncertainty estimate as a monotonically increasing function $u'(\cdot)$ followed by a diagonalization process, i.e., $w_u(\boldsymbol{y}_0) = u(\boldsymbol{\psi}_{est}(\boldsymbol{y}_0)) = \mathrm{diag}(u'(\boldsymbol{\psi}_{est}(\boldsymbol{y}_0)))$. In this section, we will detailedly introduce the implementation of this weighting coefficient function $u'(\cdot)$ in scalar form.

As illustrated in Fig. 7, the function consists of two major parts. For regions where the uncertainty estimate $\psi_{est}(y_0^i) \in [0, \psi_{max}]$, we define $u'(\cdot)$ as a linear function with an offset b_u and a slope of $(1 - b_u)/\psi_{max}$, ensuring the output remains within the range $[b_u, 1]$. This part comprises both low-uncertainty and high-uncertainty regions, and we find this linear modeling of the relationship between perturbation and uncertainty value offers a simple yet effective solution. Meanwhile, the positive offset b_u ensure a minimum noise level, preventing edge and texture areas from being assigned extremely low noise levels due to occasionally inaccurate uncertainty estimates. We empirically find that setting $\psi_{max} = 0.05$ and $b_u = 0.4$ leads to better perceptual quality, and several experimental results are shown in Tab. 5. In contrast, for regions where the uncertainty estimate $\psi_{est}(y_0^i) \in (\psi_{max}, +\infty)$, we set their weighting coefficients to a constant, i.e., $w_u(y_0) = 1.0$. A large amount of isotropic noise is then applied in these areas to provide sufficient perturbation for the score estimation to ensure the perceptual quality. In general, the weighting coefficient function $u'(\cdot)$ can be formulated as:

$$
u'(\psi) = \begin{cases} \frac{(1-b_u)}{\psi_{max}}\psi + b_u & \text{if } 0 \le \psi \le \psi_{max} \\ 1 & \text{otherwise} \end{cases}. \tag{17}
$$

C. Comparisons to pretraining-based SR methods.

Pretraining-based SR methods [35, 39, 40] harness the generative power from pretrained text-to-image models, such as stable diffusion, to enhance perceptual quality. However, they follow an entirely different track from ours. Firstly, their high perceptual quality comes from SD's capability to generate details inconsistent to LR inputs, therefore resulting in worse

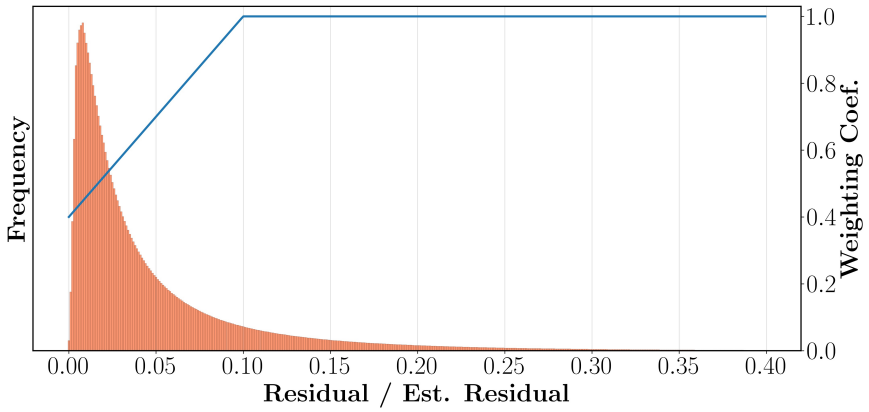

Figure 7. A combined visualization of the distribution of the **residual** $|y - x|$ (left), and the weighting coefficient $u(\psi_{est}(y))$ with respect to the **estimated residual** $|y - g(y)|$. In the region where the estimated residual is within [0, 0.1] (involving more than 80% of the data), the value of weight coefficient function increases linearly with the input.

Table 5. Ablation study on effects of offset b_u in the weighting coefficient function. The best results are highlighted in **bold**.

Model	b_u	RealSR			RealSet		
		CLIPIQA↑	MUSIQ↑	NIQE↓	CLIPIQA↑	MUSIQ↑	NIQE↓
w/o offset	0.0	0.4464	55.671	4.74	0.5445	57.335	4.84
w/ offset	0.4	**0.6010**	**64.541**	**4.02**	**0.6389**	**63.498**	**4.24**
w/o uncertainty	1.0	0.5191	61.728	4.40	0.5781	61.371	4.58

fidelity. Secondly, the model size and GPU memory consumption of these methods surpass those of the proposed UPSR method by up to 10 times. Thirdly, these methods are constrained by their fixed backbone, i.e., the stable diffusion model, making them less adaptable for rescaling. This limitation reduces their practicality when deployed on lightweight devices. Besides, these methods have to crop large input image to 128×128 patches, while UPSR can be directly applied to 512×512 or larger images.

D. Comparisons to one-step methods.

In the main paper, we apply five inference steps because we believe several more steps can better unleash the potential of diffusion models. The performance of distillation-based methods (e.g., SinSR [38]) is closely tied to that of their multi-step teacher models (e.g., ResShift [44]). We therefore develop UPSR which could work as a better teacher model to support the training of better one-step models. Meanwhile, the target of reducing diffusion steps is speeding up inference, which can also be achieved by rescaling the model size of denoiser. To make comparison with SinSR, we present UPSR-light with smaller model size. As shown in Tab. 7, the rescaled model achieves better performance with comparable inference speed and less than 1/3 of total model size.

Table 6. Qualitative comparison with SD-based models on RealSR dataset. Quality metrics are re-evaluated on uncropped image.

Models	Params	Runtime	Memory	PSNR↑	LPIPS↓	MUSIQ↑	NIQE↓
StableSR-200	1410M	33.0s	8.51G	25.80	**0.2665**	48.346	5.87
AddSR-1	2280M	0.659s	8.47G	25.23	0.2986	63.011	5.17
OSEDiff-1	1775M	0.310s	5.85G	24.57	0.3035	**67.310**	4.34
UPSR-5	**122M**	**0.212s**	**2.83G**	**26.44**	0.2871	64.541	**4.02**

Table 7. Qualitative comparison with one-step model on RealSR dataset. Quality metrics are re-evaluated on uncropped image.

Models	Params	Time	Memory	PSNR↑	LPIPS↓	MUSIQ↑	NIQE↓
SinSR-1	174M	**0.141s**	4.03G	26.01	0.4015	59.344	6.26
UPSR-light-4	**52.7M**	0.148s	**2.48G**	**26.28**	**0.3025**	**63.785**	**4.13**

E. Additional Visual Examples

In Fig. 8 and Fig. 9, we present more visual comparisons between the proposed UPSR and existing diffusion-based SR methods [26, 43, 44].

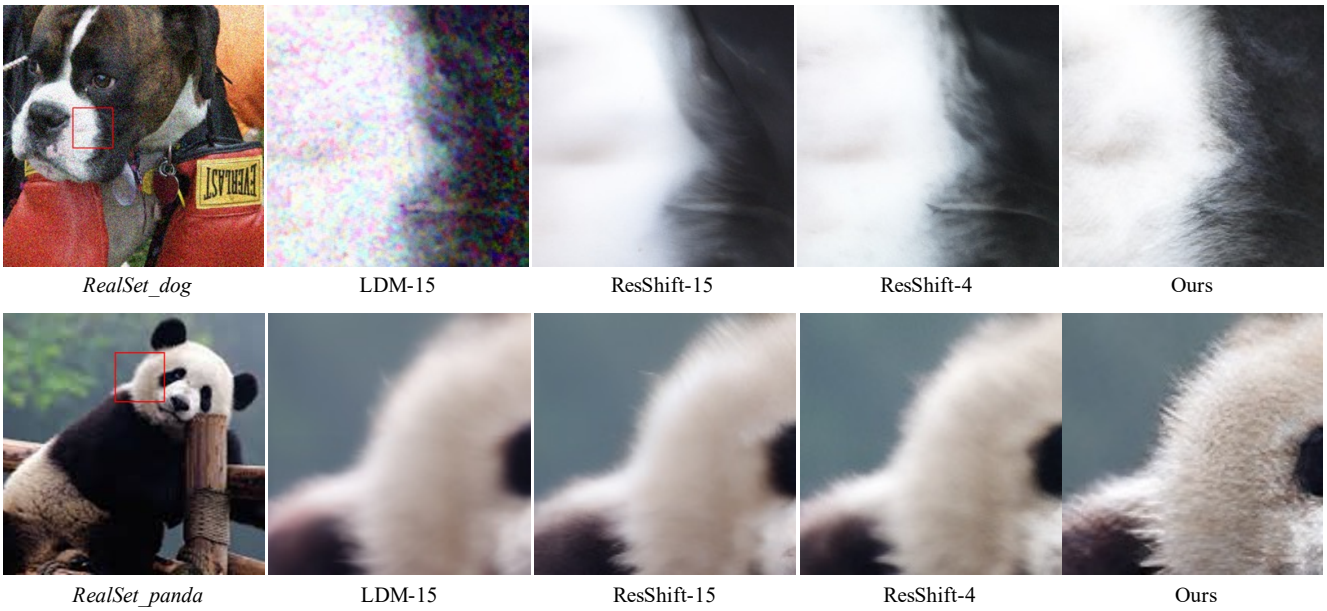

Figure 8. Additional visual comparisons on RealSet [44].

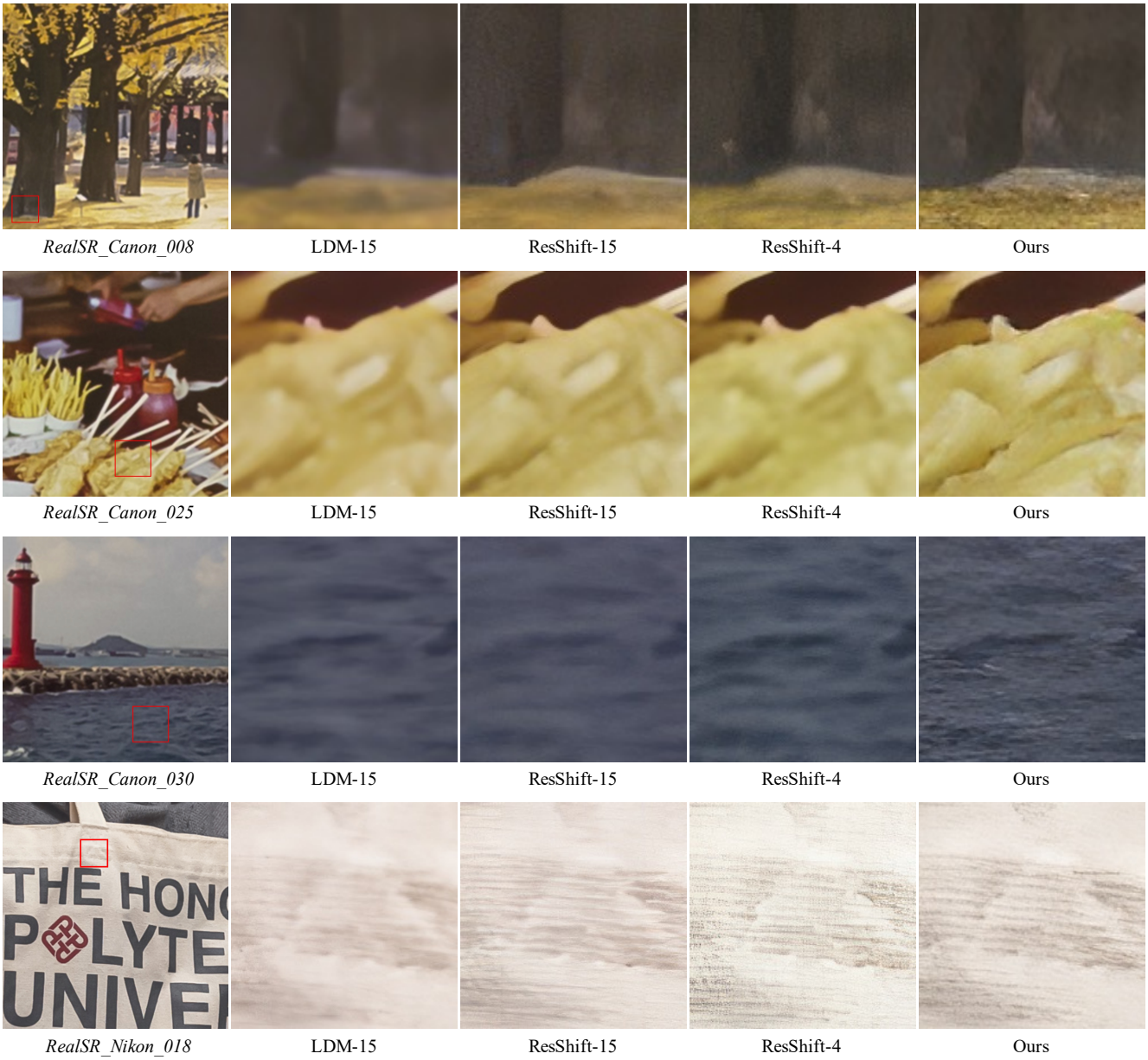

Figure 9. Additional visual comparisons on RealSR [1].