Kernel Density Steering: Inference-Time Scaling via Mode Seeking for Image Restoration

Yuyang Hu[1,2*], **Kangfu Mei**[1], **Mojtaba Sahraee-Ardakan**[1],
Ulugbek S. Kamilov[2], **Peyman Milanfar**[1], **Mauricio Delbracio**[1],
[1]Google,[2]Washington University in St. Louis
{kangfumei,mojtabaa,milanfar,mdelbra}@google.com
{h.yuyang,kamilov}@wustl.edu,

Abstract

Diffusion models show promise for image restoration, but existing methods often struggle with inconsistent fidelity and undesirable artifacts. To address this, we introduce Kernel Density Steering (KDS), a novel inference-time framework promoting robust, high-fidelity outputs through explicit local mode-seeking. KDS employs an N-particle ensemble of diffusion samples, computing patch-wise kernel density estimation gradients from their collective outputs. These gradients steer patches in each particle towards shared, higher-density regions identified within the ensemble. This collective local mode-seeking mechanism, acting as ”collective wisdom”, steers samples away from spurious modes prone to artifacts, arising from independent sampling or model imperfections, and towards more robust, high-fidelity structures. This allows us to obtain better quality samples at the expense of higher compute by simultaneously sampling multiple particles. As a plug-and-play framework, KDS requires no retraining or external verifiers, seamlessly integrating with various diffusion samplers. Extensive numerical validations demonstrate KDS substantially improves both quantitative and qualitative performance on challenging real-world super-resolution and image inpainting tasks.

1 Introduction

Image restoration (IR) seeks to recover a high-quality image $\boldsymbol{x}$ from its degraded observation $\boldsymbol{y}$. Recently, diffusion models (DMs) [1–3] have been successfully applied to challenging IR tasks [4–8], including super-resolution (SR) [9–14], image deblurring [15–17], and image inpainting [18–20]. DMs operate on a dual process: a forward process introduces noise to a clean image $\boldsymbol{x}$ over T timesteps, creating a sequence of noisy latents $\boldsymbol{z}_t$ where $\boldsymbol{z}_T$ is approximately pure noise. A learned reverse process then iteratively refines these latents from $\boldsymbol{z}_T$ back to an estimate of $\boldsymbol{z}_0$.

To enable DMs for IR task, a straightforward way is to train diffusion models directly conditioned on information $\boldsymbol{c}$ derived from the low-quality image $\boldsymbol{y}$ [21–23]. Conditioning can be implemented through various techniques, for example, input concatenation [21], cross-attention [24, 25], adapters like ControlNet [26] or LoRA [27]. This allows the model's network $\boldsymbol{\epsilon}_\theta(\boldsymbol{z}_t, t, \boldsymbol{c})$ to estimate the conditional score function $\nabla_{\boldsymbol{z}_t} \log p_t(\boldsymbol{x}_t|\boldsymbol{c})$ to directly approximate target score function $\nabla_{\boldsymbol{z}_t} \log p_t(\boldsymbol{z}_t|\boldsymbol{y})$, enabling standard sampling at inference.

A key challenge here lies in navigating the perception-distortion trade-off [28]. Standard posterior sampling can yield perceptually sharp results, but often suffers from high distortion and hallucinations. Conversely, averaging multiple independent samples approximates the Minimum Mean Squared Error (MMSE) estimate, minimizing average distortion but leading to blurriness and poor perceptual

[*]This work was done during an internship at Google.

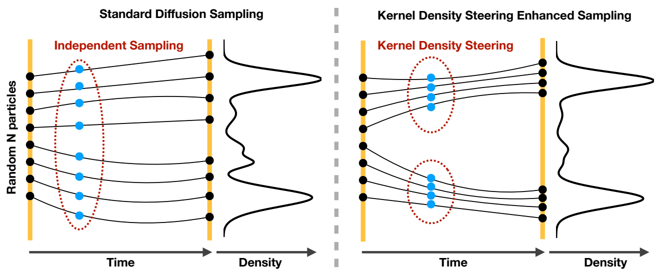

Figure 1: Conceptual illustration of Kernel Density Steering (KDS) for diffusion sampling. Left: Standard diffusion sampling with N independent particles (blue dots) can result in high variance. Right: KDS utilizes the ensemble of N particles to estimate local density. It then guides these particles towards shared high-density modes (peaks in the density curve) during the diffusion process.

quality [29]. Neither extreme is ideal for high-quality IR. Many existing methods improve this trade-off by additional model training or specialized network architectures [4, 29–34].

This paper introduces **Kernel Density Steering (KDS)**, an inference-time framework designed to improve the distortion-perception trade-off without requiring any model retraining. KDS leverages collective information from an N-particle ensemble by steering particles towards regions of high concentration within the predicted solution ensemble.

This mechanism is implemented as a two-step process. First, at each step t, the denoiser predicts the clean data $\hat{\mathbf{z}}_0^{(i)}$ for each particle i. Second, it performs a non-parametric mode-seeking operation to steer each prediction $\hat{\mathbf{z}}_0^{(i)}$ toward a local region of high density. This is achieved by implicitly estimating the ensemble's density using Kernel Density Estimation (KDE) [35, 36] and then applying a single mean-shift update—which is equivalent to a gradient ascent step on this estimated density—to find a consensus point $\tilde{\mathbf{z}}_0^{(i)}$:

$$\tilde{\mathbf{z}}_0^{(i)} = \frac{\sum_{k=1}^{N} K\left(\frac{\|\hat{\mathbf{z}}_0^{(i)} - \hat{\mathbf{z}}_0^{(k)}\|^2}{h^2}\right)\hat{\mathbf{z}}_0^{(k)}}{\sum_{k=1}^{N} K\left(\frac{\|\hat{\mathbf{z}}_0^{(i)} - \hat{\mathbf{z}}_0^{(k)}\|^2}{h^2}\right)}. \tag{1}$$

This refined estimate $\tilde{\mathbf{z}}_0^{(i)}$ defines a new consensus score, $\tilde{\mathbf{s}}_t^{(i)}$. The original and consensus scores are defined as:

$$\mathbf{s}_t^{(i)} = \frac{\mathbf{z}_t^{(i)} - \sqrt{\bar{\alpha}_t}\hat{\mathbf{z}}_0^{(i)}}{\sigma_t^2} \quad \text{and} \quad \tilde{\mathbf{s}}_t^{(i)} = \frac{\mathbf{z}_t^{(i)} - \sqrt{\bar{\alpha}_t}\tilde{\mathbf{z}}_0^{(i)}}{\sigma_t^2}. \tag{2}$$

This guidance is incorporated into the probability flow ODE via a weighted average of the two scores:

$$d\mathbf{z}_t^{(i)} = \left[f(t)\mathbf{x}_t^{(i)} - \frac{1}{2}g(t)^2\left((1-\delta_t)\mathbf{s}_t^{(i)} + \delta_t\tilde{\mathbf{s}}_t^{(i)}\right)\right]dt, \tag{3}$$

where δ_t is the KDS guidance strength. Thus, particles are guided by the original score and the KDS term. This term functions as a form of collaborative filtering, using non-parametric mode-seeking to guide the ensemble toward areas of high consensus. This adaptive, internal guidance aims for restorations that are both quantitatively accurate and perceptually sharp.

To demonstrate the effectiveness of this approach, we first evaluate KDS's impact on posterior sampling in a controlled synthetic task. Subsequently, we provide extensive empirical validation, showing significant improvements in both quantitative performance and qualitative robustness on challenging real-world SR and inpainting tasks.

Our contributions are: (1) We introduce Kernel Density Steering (KDS), a novel ensemble-based, inference time guidance method that uses patch-wise KDE gradients to steer diffusion sampling towards high-density posterior modes. **(2)** KDS improves the distortion-perception trade-off in image restoration by operating directly on the sampling process without retraining. **(3)** We design a patch-wise mechanism to overcome the curse of dimensionality, enabling KDS to scale to high-dimensional latent space. **(4)** Empirical validation showing significant improvements in fidelity, perception, and robustness on challenging real-world super-resolution and inpainting tasks.

2 Background

2.1 Latent Diffusion Models

Latent Diffusion Models (LDMs) [24] efficiently perform image generation by operating in a lower-dimensional latent space learned by a pre-trained autoencoder [37], mitigating the computational cost of pixel-space methods on high-resolution images. The autoencoder includes an encoder $\mathcal{E}$ mapping an image $\boldsymbol{x}$ to a latent $\boldsymbol{z} = \mathcal{E}(\boldsymbol{x})$, and a decoder $\mathcal{D}$ mapping $\boldsymbol{z}$ back to $\hat{\boldsymbol{x}} = \mathcal{D}(\boldsymbol{z})$.

The forward diffusion process gradually adds Gaussian noise to the target latent sample $\boldsymbol{z}_0 = \mathcal{E}(\boldsymbol{x}_0)$ over T timesteps according to a noise schedule α_t. This process results in the following marginal distribution for the latent $\boldsymbol{z}_t$ at any timestep t:

$$q(\boldsymbol{z}_t|\boldsymbol{z}_0) := \mathcal{N}(\boldsymbol{z}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{z}_0, (1-\bar{\alpha}_t)\mathbf{I}), \text{where } \bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s. \tag{4}$$

The reverse process aims to generate a clean latent variable $\boldsymbol{z}_0$ from a noisy latent variable $\boldsymbol{z}_T \sim \mathcal{N}(0, \mathbf{I})$. This is achieved by training a neural network $\boldsymbol{\epsilon}_\theta(\boldsymbol{z}_t, t)$ to predict the noise component $\boldsymbol{\epsilon}_t$ that was added during the forward process. This predicted noise is directly related to the score function of the noisy latent distribution $p_t(\boldsymbol{z}_t)$: $\nabla_{\boldsymbol{z}_t}\log p_t(\boldsymbol{z}_t) \approx -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\boldsymbol{z}_t, t)$. By accurately predicting the noise, the network learns to estimate this score function, thereby implicitly capturing the data distribution $p(\boldsymbol{z}_t)$ at different noise levels.

To sample from the learned distribution, Denoising Diffusion Implicit Models (DDIM) [38] offer a fast, non-Markovian sampling procedure. The sampling step from $\boldsymbol{z}_t$ to $\boldsymbol{z}_{t-1}$ in DDIM, which predicts the latent state at $t-1$ based on the estimated clean latent at time t, is given by:

$$\boldsymbol{z}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\left(\frac{\boldsymbol{z}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_\theta(\boldsymbol{z}_t, t, \boldsymbol{c})}{\sqrt{\bar{\alpha}_t}}\right) + \sqrt{1-\bar{\alpha}_{t-1}}\boldsymbol{\epsilon}_\theta(\boldsymbol{z}_t, t, \boldsymbol{c}). \tag{5}$$

At the end of sampling, the predicted $\boldsymbol{z}_0$ is then decoded by $\mathcal{D}$ to obtain the generated image $\hat{\boldsymbol{x}}_0$.

2.2 LDM for Image restoration

Two primary methods exist for leveraging diffusion models in IR. Conditional LDMs [4–6, 9–11, 15, 16, 18–20, 39], learn the posterior $p(\boldsymbol{z}|\boldsymbol{c})$ directly, where the conditioning $\boldsymbol{c}$ is derived from $\boldsymbol{y}$. These models can be highly effective but may face challenges in accurately modeling the true posterior for complex degradations, potentially impacting restoration fidelity. Alternatively, unconditional LDMs can be employed with inference-time guidance using Bayes' rule [40–45]. These methods typically require computing gradients of the log-likelihood $\log p(\boldsymbol{y}|\boldsymbol{x}_t)$ (often approximated via the estimate $\hat{\boldsymbol{x}}_{0|t}$). This necessitates knowledge of the degradation process (e.g., the operator A in $\boldsymbol{y} \approx A\boldsymbol{x}$), limiting their applicability in scenarios where the degradation is unknown or cannot be accurately modeled which is the case in many real-world image SR or deblurring.

2.3 Inference-Time Scaling in Diffusion Models

Merely increasing the number of denoising steps (NFEs) for a single sample yields diminishing returns [46], motivating research into more effective inference-time strategies [47, 48]. One prominent avenue, particularly explored in Text-to-Image (T2I) generation, involves actively searching for optimal initial noise vectors ($\boldsymbol{z}_T$) or resampling generation trajectories, often guided by external verifiers (e.g., CLIP score, pre-trained classifiers) to build a reward function and select promising candidates [47, 48]. However, this introduces reliance on these external verifiers, which can suffer from biases or misalignment with the true image restoration (IR) target.

A second class of strategies are specifically designed for imaging inverse problems. These often employ ensemble methods, such as those based on Sequential Monte Carlo (SMC) techniques [49–52], employ likelihood approximation and reweighting sampling trajectories to achieve more accurate approximation of the posterior distribution $p(\boldsymbol{x}|\boldsymbol{y})$ with unconditional DMs. These approaches, however, require accurate knowledge of the degradation model specific to the IR task. This assumption is often invalid in many real-world applications, such as real-world image SR and deblurring.

In contrast to both these approaches, KDS introduces an ensemble method that operates without external verifiers or the need for the knowledge of degradation model. By focusing on internal consensus guidance derived directly from particle interactions within the ensemble, KDS offers a more robust and broadly applicable solution for enhancing image restoration quality.

2.4 Mean Shift for Mode Seeking

Mean Shift [53, 54] is a non-parametric clustering and mode-seeking algorithm. Given a set of points $\{\boldsymbol{x}_k\}_{k=1}^N$, the goal is to find the modes (local maxima) of the underlying density function. The algorithm iteratively shifts each point towards the local mean of points within its neighborhood, weighted by a kernel function (often Gaussian). For a point $\boldsymbol{x}$ and a kernel K with bandwidth h, the mean shift vector is calculated as:

$$\mathbf{m}(\boldsymbol{x}) = \frac{\sum_{k=1}^N K\left(\frac{\|\boldsymbol{x}-\boldsymbol{x}_k\|^2}{h^2}\right)\boldsymbol{x}_k}{\sum_{k=1}^N K\left(\frac{\|\boldsymbol{x}-\boldsymbol{x}_k\|^2}{h^2}\right)} - \boldsymbol{x}. \tag{6}$$

A crucial property of the mean shift vector, $\mathbf{m}(\boldsymbol{x})$, is its direct proportionality to the gradient of the logarithm of the Kernel Density Estimate (KDE): $\mathbf{m}(\boldsymbol{x}) \propto \nabla_{\boldsymbol{x}} \log \hat{p}_K(\boldsymbol{x})$. Therefore, iteratively updating a point by adding its mean shift vector ($\boldsymbol{x} \leftarrow \boldsymbol{x} + \mathbf{m}(\boldsymbol{x})$) is an effective method for performing gradient ascent on the KDE, guiding the point towards a local mode of the estimated density. This established connection is leveraged in our proposed KDS framework.

3 Kernel Density Steering for Diffusion Samplers

Kernel Density Steering (KDS) is an inference-time technique for Image Restoration that enhances diffusion model sampling by leveraging an N-particle latent ensemble, $\boldsymbol{Z}_t = \{\boldsymbol{z}_t^{(i)}\}_{i=1}^N$. Standard diffusion sampling can result in outputs with inconsistent fidelity and undesirable artifacts. KDS addresses this by guiding all particles towards regions of high consensus within their own ensemble which is hypothesized to lead to more robust and high-fidelity restorations.

KDS achieves this ensemble-driven mode-seeking by incorporating an additional steering term into the sampling update for each particle. This steering is inspired by the Mean Shift algorithm (Sec. 2.4). Specifically, for each particle, KDS computes a mean shift vector $\mathbf{m}(\boldsymbol{z}_t^{(i)})$ based on the current ensemble $\boldsymbol{Z}_t$. Then KDS steering is applied by taking a step in the direction of this mean shift vector, which directs to a local mode of the ensemble's Kernel Density Estimate (KDE).

3.1 Patch-wise Mechanism and Integration

Directly applying KDE and mean shift in the full, high-dimensional latent space $\boldsymbol{z}_t$ is not achievable. A prohibitively large number of particles would be needed to obtain a reasonable density estimate via KDE in such high-dimensional spaces. To address this, we introduce a patch-wise mechanism to apply the KDS principle to smaller, more manageable, lower-dimensional patches. These patches are extracted from the predicted clean latent state $\hat{\boldsymbol{z}}_{0|t}^{(i)}$ corresponding to each particle $\boldsymbol{z}_t^{(i)}$ in the ensemble. This predicted clean latent, $\hat{\boldsymbol{z}}_{0|t}^{(i)}$, is the output of the diffusion model's denoising step at time t, obtained via:

$$\hat{\boldsymbol{z}}_{0|t}^{(i)} = \frac{1}{\sqrt{\bar{\alpha}_t}}\left(\boldsymbol{z}_t^{(i)} - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_\theta(\boldsymbol{z}_t^{(i)}, t, \boldsymbol{c})\right). \tag{7}$$

Patch-wise Mechanism: To the ensemble of predicted clean latent states $\{\hat{\boldsymbol{z}}_{0|t}^{(i)}\}_{i=1}^N$, our patch-wise mechanism involves the following steps at each timestep t:

1. *Patch Extraction:* For each predicted clean latent $\hat{\boldsymbol{z}}_{0|t}^{(i)}$ in the ensemble (obtained from Eq. 7), extract a set of non-overlapped patches $\{\boldsymbol{p}_j^{(i)}\}_j$ where we have omitted the time index to simplify the notation. This yields an ensemble of corresponding patches $\boldsymbol{P}_j = \{\boldsymbol{p}_j^{(k)}\}_{k=1}^N$ for each given spatial patch location j.
2. *Compute and Apply Patch-wise Steering:* For each patch $\boldsymbol{p}_j^{(i)}$ (from particle i at location j), its mean shift vector $\mathbf{m}(\boldsymbol{p}_j^{(i)})$ is first computed based on the local ensemble of corresponding patches

$\boldsymbol{P}_j = \{\boldsymbol{p}_j^{(k)}\}_{k=1}^N$. This vector, which directs the patch towards a region of higher consensus within the ensemble, is given by:

$$\mathbf{m}(\boldsymbol{p}_j^{(i)}) = \frac{\sum_{k=1}^N G\left(\frac{\|\boldsymbol{p}_j^{(i)} - \boldsymbol{p}_j^{(k)}\|^2}{h^2}\right)\boldsymbol{p}_j^{(k)}}{\sum_{k=1}^N G\left(\frac{\|\boldsymbol{p}_j^{(i)} - \boldsymbol{p}_j^{(k)}\|^2}{h^2}\right)} - \boldsymbol{p}_j^{(i)}. \tag{8}$$

Here, G is a Gaussian kernel function and h is the bandwidth hyperparameter. As discussed in Sec. 2.4, this vector $\mathbf{m}(\boldsymbol{p}_j^{(i)})$ points from the current patch towards the estimated local mode of its ensemble's density. This computed mean shift vector is then used to update the patch:

$$\hat{\boldsymbol{p}}_j^{(i),\text{KDS}} \leftarrow \boldsymbol{p}_j^{(i)} + \delta_t \mathbf{m}(\boldsymbol{p}_j^{(i)}), \tag{9}$$

where $\delta_t \in [0, 1]$ is a time-dependent steering strength.

3. *Reconstruct Guided Latent Prediction:* After all patches $\{\boldsymbol{p}_j^{(i)}\}_j$ from a given $\hat{\boldsymbol{z}}_{0|t}^{(i)}$ have been updated to $\{\hat{\boldsymbol{p}}_j^{(i),\text{KDS}}\}_j$, they are reassembled by direct replacement to original coordinate to form the KDS-refined clean latent prediction, $\hat{\boldsymbol{z}}_{0|t}^{(i),\text{KDS}}$.

Integration with Diffusion Samplers: KDS integrates seamlessly with standard diffusion samplers (e.g., first-order ODE DDIM solver and high-order ODE DPM-Solver++ [55]). At each sampling step t, the sampler first computes the predicted clean latent for each particle, $\hat{\boldsymbol{z}}_{0|t}^{(i)}$, using (Eq. 7). Patch-wise KDS is then applied to this ensemble of predictions $\{\hat{\boldsymbol{z}}_{0|t}^{(i)}\}_{i=1}^N$ to produce the ensemble of refined predictions $\{\hat{\boldsymbol{z}}_{0|t}^{(i),\text{KDS}}\}_{i=1}^N$. The sampler subsequently uses these KDS-refined estimates to compute the next latent states $\{\boldsymbol{z}_{t-1}^{(i)}\}_{i=1}^N$. After the full reverse diffusion process, an ensemble of KDS-guided clean latent states $\{\hat{\boldsymbol{z}}_0^{(i),\text{KDS}}\}_{i=1}^N$ is obtained.

Final Particle Selection: To produce a single, representative output image from an ensemble of generated candidates $\{\hat{\boldsymbol{z}}_0^{(i),\text{KDS}}\}_{i=1}^N$, a selection is made in the latent space. Our proposed strategy is to choose the latent vector from the ensemble that is closest to the ensemble's mean $\bar{\boldsymbol{z}}_0$:

$$\hat{\boldsymbol{z}}_0^{\text{selected}} = \arg\min_{\hat{\boldsymbol{z}}_0^{(i)}} \|\hat{\boldsymbol{z}}_0^{(i)} - \bar{\boldsymbol{z}}_0\|_2^2. \tag{10}$$

This selected latent vector $\hat{\mathbf{z}}_0^{\text{selected}}$ is then transformed by the decoder $\mathcal{D}$ into the final image $\hat{\mathbf{x}}_0$, as represented by the equation $\hat{\mathbf{x}}_0 = \mathcal{D}(\hat{\mathbf{z}}_0^{\text{selected}})$.

4 Experiments

In this section, we present experiments to numerically evaluate our proposed method. First, we utilize a 2D toy example (Sec. 4.1) to visually illustrate the impact of our proposed Kernel Density Steering on the diffusion model sampling process, providing intuition for its mechanism. Subsequently, we demonstrate the effectiveness and practical applicability of our approach on two challenging real-world tasks: image super-resolution (SR) (Sec. 4.2.1) and image inpainting (Sec. 4.2.2).

4.1 KDS Sampling in a 2D Example Case

To visualize KDS's effect, we consider sampling from a 2D Mixture of Gaussians (MoG) target distribution $p(\boldsymbol{x}_0) = \sum_{c=1}^C \pi_c \mathcal{N}(\boldsymbol{x}_0; \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$. In this controlled setting, the exact score $\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t)$ of the noisy distribution is known. KDS is applied by adding its steering term (derived from Eq. 9 using the 2D particle states $\{\boldsymbol{x}_t^{(i)}\}_{i=1}^N$) to the update driven by the exact score within a DDIM sampler. As the data dimension here is very low, we let the patch size equal to the full data size.

As illustrated in Figure 2, KDS significantly sharpens the resulting sample distribution. Samples cluster more tightly around the true mode means ($\boldsymbol{\mu}_c$), effectively reducing the spread (variance) of samples within each mode compared to using the exact score alone. This occurs because KDS actively guides particles towards their respective mode centers—which correspond to regions of high sample density in the ensemble—thereby improving concentration of samples around each mode.

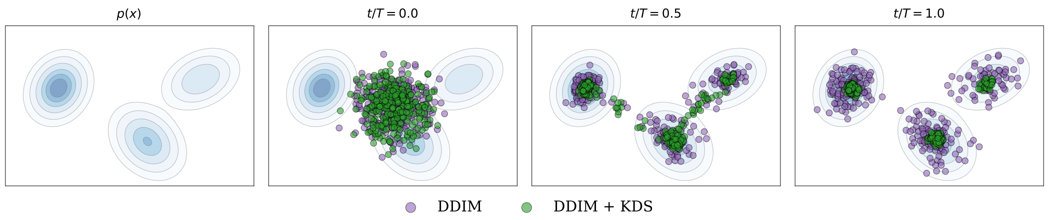

Figure 2: Kernel Density Steering (KDS) sharpens sample distributions in a 2D Mixture of Gaussians (MoG) toy problem. The target distribution $p(\boldsymbol{x}_0)$ consists of three distinct Gaussian modes. Samples are drawn using DDIM with the exact score function (purple dots) versus DDIM augmented with KDS (green dots, $N = 50$ particles). KDS guides particles through the reverse diffusion progresses, leading to significantly higher sample concentration at the mode peaks compared to standard DDIM.

Table 1: Super-Resolution Performance on DIV2K dataset with DDIM (50 steps)

Method	PSNR	SSIM	LPIPS (↓)	NIMA	DISTS (↓)	MANIQA	CLIPIQA	FID (↓)
LDM-SR	22.05	0.531	0.307	4.922	0.179	0.390	0.594	20.89
+ KDS ($N = 10$)	**22.37**$\star$	**0.549**$\star$	**0.292**$\star$	**4.949**$\star$	**0.176**$\star$	**0.399**$\star$	**0.601**$\star$	**20.78**$\star$
DiffBIR	21.56	0.488	0.377	5.195	0.223	0.566	0.730	32.28
+ KDS ($N = 10$)	**22.44**$\star$	**0.535**$\star$	**0.348**$\star$	**5.219**$\star$	**0.220**$\star$	**0.571**$\star$	**0.744**$\star$	**30.67**$\star$
SeeSR	22.43	0.573	0.340	4.902	0.200	0.423	0.605	25.96
+ KDS ($N = 10$)	**22.79**$\star$	**0.587**$\star$	**0.313**$\star$	**5.026**$\star$	**0.191**$\star$	**0.488**$\star$	**0.679**$\star$	**25.44**$\star$

Statistically significant differences ($P < 0.05$) compared with DDIM are marked with a $\star$.

4.2 Real-world Image Restoration

To demonstrate the effectiveness of our Kernel Density Steering (KDS), we compare its performance against baseline sampling method on image super-resolution and inpainting tasks. Our experiments utilize two widely used samplers (DDIM, DPM-Solver++). For each configuration, we contrast the results obtained with standard sampling versus KDS-enhanced sampling, ensuring fairness by using the same initial noise. We compare performance with and without KDS, using the same initial noise $\{\boldsymbol{z}_T^{(i)}\}_{i=1}^N$ for fairness across an ensemble of $N = 10$ particles unless specified otherwise. Full experimental details, including hyperparameter settings (e.g., patch size, steering strength δ_t) and comprehensive ablation studies, are provided in the Appendix.

Table 2: Super-Resolution Performance on real-world collected datasets with DDIM (50 steps)

Datasets	RealSR					DrealSR				
Method	PSNR	SSIM	LPIPS (↓)	DISTS (↓)	CLIPIQA	PSNR	SSIM	LPIPS (↓)	DISTS (↓)	CLIPIQA
LDM-SR	24.01	0.666	0.308	0.211	**0.624**	26.13	0.689	0.342	0.222	0.610
+ KDS ($N = 10$)	**24.68**$\star$	**0.703**$\star$	**0.275**$\star$	**0.201**$\star$	0.622	**26.32**$\star$	**0.702**$\star$	**0.331**$\star$	**0.317**$\star$	**0.617**$\star$
DiffBIR	23.21	0.610	0.370	0.250	0.689	23.99	0.551	0.491	0.293	**0.701**
+ KDS ($N = 10$)	**24.39**$\star$	**0.669**$\star$	**0.339**$\star$	**0.244**$\star$	**0.692**$\star$	**25.63**	**0.645**	**0.422**$\star$	**0.276**$\star$	0.693$\star$
SeeSR	24.29	0.710	0.279	**0.204**	0.588	26.97	0.750	0.300	0.218	0.625
+ KDS ($N = 10$)	**24.50**$\star$	**0.719**$\star$	**0.272**$\star$	0.206$\star$	**0.640**$\star$	**27.42**$\star$	**0.765**$\star$	**0.287**$\star$	**0.212**$\star$	**0.651**$\star$

Statistically significant differences ($P < 0.05$) compared with DDIM are marked with a $\star$.

4.2.1 Real-world Image Super-Resolution

We evaluated the performance of KDS for $4\times$ real-world super-resolution (SR). This evaluation utilized the LDM-SR [24], DiffBIR [4], and SeeSR [33] backbones across several datasets: DIV2K [56], RealSR [57], DrealSR [58]. Performance was evaluated using a comprehensive suite of metrics, including PSNR, SSIM [59], LPIPS [60], FID [61], NIMA [62], MANIQA [63], and CLIPIQA [64].

Results: Quantitative results consistently show KDS's benefits. Tables 1 and Tables 2 demonstrate that adding KDS significantly improves both distortion and perceptual metrics across different

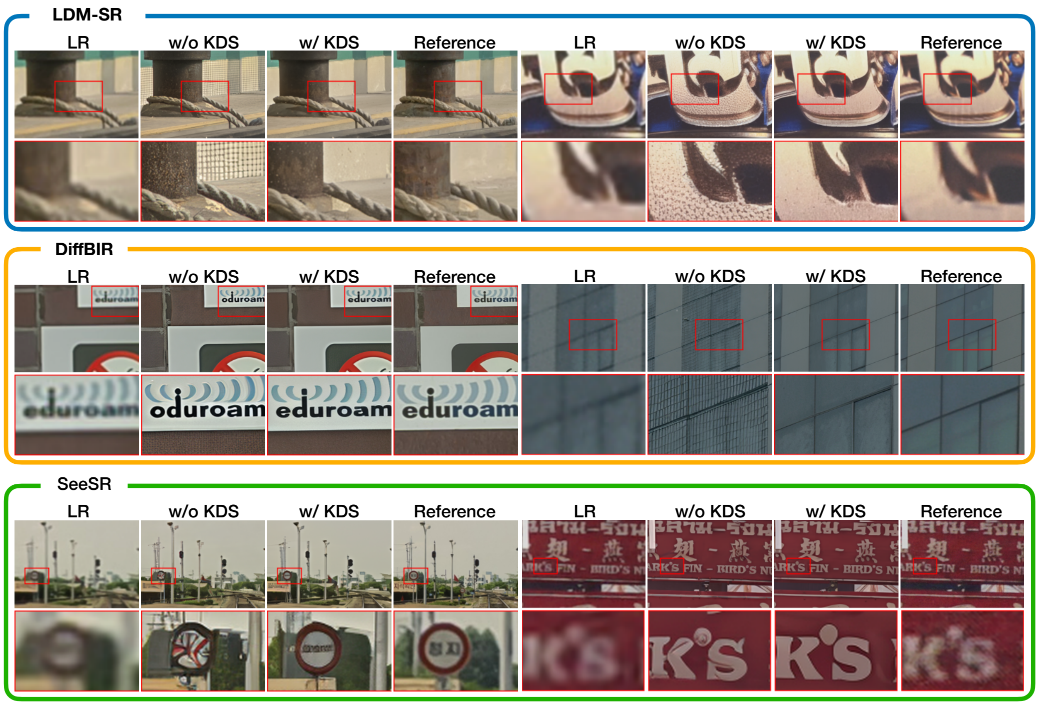

Figure 3: Qualitative comparison of $4\times$ Real-world Super-Resolution with KDS-enhanced DDIM sampling (Number of Particles $N = 10$). 'w/o KDS' shows results from baseline DDIM, while 'w/ KDS' shows results with KDS. KDS consistently produces images with improved sharpness, finer details, and reduced artifacts across LDM-SR, DiffBIR, and SeeSR backbones.

datasets, backbones, and samplers (DDIM, DPM-Solver++) compared to the respective baselines without KDS. Qualitative results in Figure 3 corroborate these findings, showing restorations with higher fidelity and fewer artifacts when using KDS.

Compare with Best-of-N approaches: While inference-time scaling allows for the generation of multiple candidate solutions, especially beneficial for low-quality inputs, selecting the optimal one from an N-particle ensemble is non-trivial. As Table 4 shows, using a non-reference metric like LIQE [65] to pick the best particle results in significantly lower performance compared to our kernel density steering (KDS) method. This demonstrates that, despite comparable computational costs, traditional post-sampling selection methods do not achieve the same level of performance as KDS.

Table 3: Inpainting Performance on ImageNet with LDM-inpainting backbone.

Method	PSNR	SSIM	LPIPS(↓)	FID(↓)
DPM-Solver	19.94	0.725	0.144	11.70
+ KDS ($N=10$)	**21.29**	**0.747**	**0.135**	**11.29**
DDIM	21.03	0.736	0.140	11.47
+ KDS ($N=10$)	**21.35**	**0.748**	**0.131**	**11.18**

Table 4: Comparison to Best-of-N (BoN) using LIQE [65] on RealSR with DiffBIR.

Method	N	PSNR	LPIPS (↓)	Time	Memory
DDIM	1	23.21	0.370	7.9s	8.0Gb
+ BoN	5	23.72	0.361	14.9s	9.7Gb
+ KDS	5	**24.17**	**0.349**	15.6s	9.7Gb
+ BoN	10	23.77	0.366	27.3s	10.4Gb
+ KDS	10	**24.39**	**0.339**	28.4s	10.4Gb

4.2.2 Image Inpainting

We evaluate the performance of KDS (N=10) on a center box inpainting task using the ImageNet dataset. For this task, the central 30% square region of each image is masked. We employ a Latent Diffusion Model (LDM) for inpainting, initialized with weights from a pretrained text-to-image model that matches the architecture and size of Stable Diffusion v2 [24]. The LDM was fine-tuned on ImageNet dataset for random box inpainting. The training utilized a batch size of 1024 and a

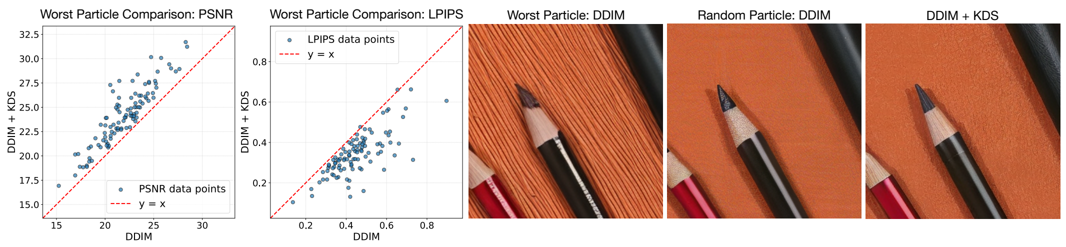

Figure 4: Robustness analysis of KDS on the RealSR dataset. **Left**: Scatter plots comparing the PSNR and LPIPS of the DDIM with KDS versus a worst-performing particle of standard DDIM ensemble ($N = 10$). KDS consistently improves the quality of the worst-case samples. **Right**: Qualitative examples comparing the worst-performing output from a DDIM ensemble with KDS-guided DDIM with the same random seed, demonstrating KDS's superior consistency and artifact reduction.

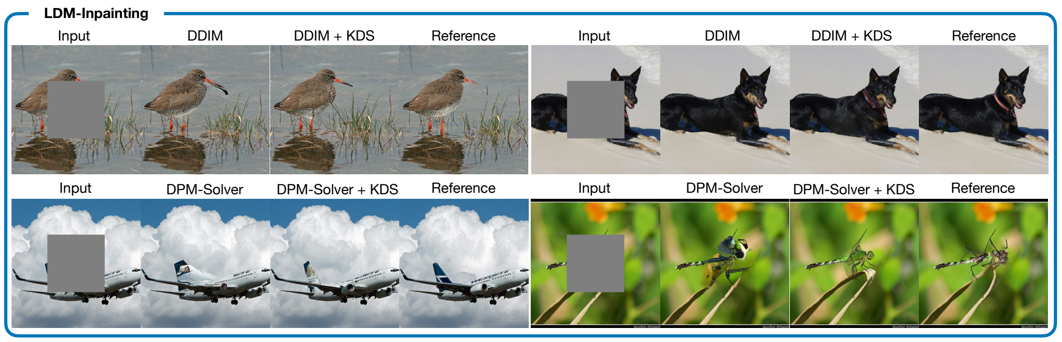

Figure 5: Box-inpainting performance of KDS. KDS generates more coherent and detailed inpainted regions compared to standard DDIM sampling.

learning rate of 1e-4, parameters empirically selected to maximize compute efficiency. Performance was measured using PSNR, SSIM, LPIPS, and FID.

As shown in Table 3, our proposed KDS improves quantitative inpainting performance. Furthermore, Figure 5 demonstrates that KDS generates inpainted regions with enhanced fidelity, greater coherence with surrounding image content, and more plausible details compared to standard sampling.

4.3 Analysis of KDS Robustness

A key objective of KDS is to enhance the reliability and consistency of diffusion-based restorations, particularly by reducing artifacts and improving fidelity.

Figure 4 demonstrates this improved stability. The scatter plots (left) compare the performance (PSNR and LPIPS) of DDIM enhanced by KDS against the worst-performing particle from a baseline DDIM ensemble (also $N = 10$) on RealSR dataset. Points above (PSNR) or below (LPIPS) the $y = x$ line signify that KDS improves the worst-case performance. These two scatter plots clearly demonstrate enhanced restoration reliability. Qualitative examples (right part of Figure 4) further illustrate KDS's consistency and artifact reduction.

4.4 Impact of KDS Hyperparameters: Number of Particles and Bandwidth

The performance of KDS, as an ensemble-based technique, is inherently linked to its key hyperparameters: the number of particles N and the kernel bandwidth h. We analyze their impact on SR performance (PSNR, SSIM, LPIPS, FID) using the LDM-SR model on the DIV2K dataset, as illustrated in Figure 6.

Number of Particles N: Increasing the ensemble size N consistently enhances restoration quality. As shown in Figure 6, both distortion metrics (e.g., PSNR, SSIM) and perceptual metrics (e.g., LPIPS, FID) benefit, particularly as N increases from small (e.g., $N \approx 5 - 10$), where kernel density estimates are less robust, to moderate (e.g., $N \approx 15 - 20$) values. While further increasing N (e.g.,

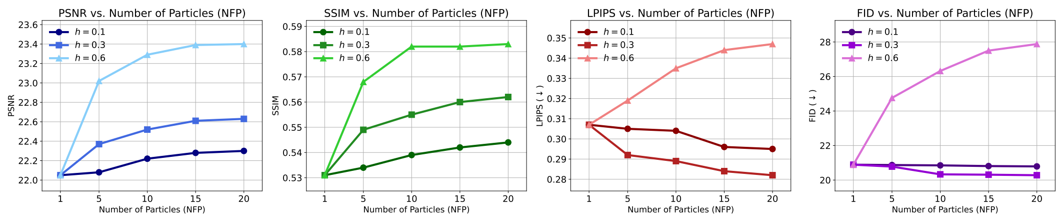

Figure 6: Influence of Particle Number (N) and Bandwidth (h) on Real-world SR. Performance metrics (PSNR, SSIM, LPIPS, FID) are plotted against the number of particles (N) for different bandwidth (h) settings on the DIV2K dataset, using LDM-SR backbone with KDS.

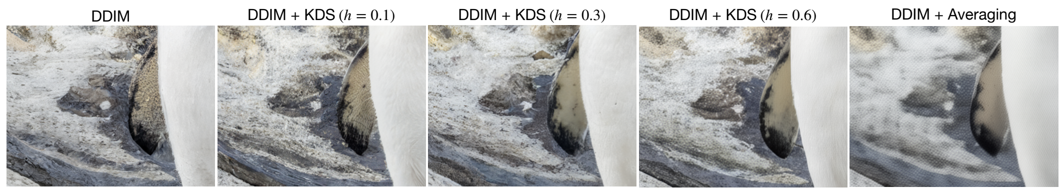

Figure 7: Influence of Bandwidth (h) on Real-world SR based on LDM-SR backbone. Note, DDIM + averaging means a post-sampling averaging across the whole ensemble.

beyond $N > 15$ in depicted scenarios) can yield additional improvements, the gains may become marginal, leading to saturation. This underscores the inherent trade-off between the computational cost, which scales linearly with N, and the achievable performance boost.

Kernel Bandwidth h: The kernel bandwidth h plays a crucial role in balancing the perception-distortion trade-off. Figure 6 (different colored lines representing different h settings) demonstrates that while larger ensembles generally improve performance, the choice of h is critical. Excessively large bandwidths (e.g., $h = 0.6$ in some cases) can lead to over-smoothing, potentially enhancing distortion metrics like PSNR & SSIM at the cost of perceptual quality (higher LPIPS & FID). Conversely, a very small h might not capture sufficient local consensus. Optimal h selection is therefore essential for achieving the desired balance between quantitative accuracy and perceptual sharpness. Our experiments suggest that a moderate h often provides a good compromise. Furthermore, in Appendix A.2, we show that standard automatic bandwidth selection strategies (e.g., median heuristic) achieve performance nearly identical to our manually tuned h. This demonstrates that KDS can be deployed effectively without tedious manual tuning, enhancing its practical applicability.

Further parameter analyses in Appendix A.2 confirm KDS is robust to hyperparameter choices, including different kernel functions (e.g., Gaussian, Laplacian). We also explore the trade-off between particle count (N) and patch size. As shown in the appendix, increasing N enables the use of larger patch sizes, which enhances performance by better mitigating the curse of dimensionality.

5 Discussion

We further analyze KDS by comparing it to other inference-time techniques. We compare it against other mode-seeking sampling strategies (HDS [66], DGS [67]) and separately against other methods, like InDI [29], which are also designed to improve the perception-distortion trade-off.

Comparison with HDS [66] and DGS [67]. To position KDS relative to other inference-time mode-seeking techniques, we compare it against High Density Sampling (HDS) [66] and Density-Guided Sampling (DGS) [67]. We implemented both methods with the DiffBIR [4] backbone and swept their key hyperparameters (time t for HDS, quantile q for DGS) to find their best-performing configurations on the RealSR dataset, with a full parameter sweep available in Appendix A.2.

As shown in Table 5, HDS achieves high PSNR at a severe cost to perceptual quality (LPIPS, CLIPIQA), while DGS offers moderate improvement. In contrast, KDS ($N = 10$) provides the best balance, uniquely achieving both a strong PSNR and the best perceptual scores (LPIPS, DISTS, CLIPIQA), confirming its superior perception-distortion trade-off.

The core conceptual difference lies in their assumptions. Both HDS [66] and DGS [67] are mode-seeking but rely on different assumptions compared with KDS. HDS assumes a unimodal Gaussian distribution, which is often violated by highly multi-modal IR posteriors. DGS, while non-parametric in a different sense, guides a single sample by progressively sharpening the marginal density $p_t(x_t)$ at each step, forcing it toward a single high-density mode. In sharp contrast, KDS makes no explicit assumption about the posterior's shape. It uses a non-parametric KDE on an ensemble of outputs ($\hat{\boldsymbol{z}}_{0|t}$), making it better suited to finding a consensus within the multi-modal posteriors common in IR tasks.

Comparison with InDI [29]. We also distinguish KDS from InDI [29], which also adjusts the distortion-perception trade-off at inference time. Their core approaches differ: InDI is not a posterior mode-seeking algorithm. It performs progressive refinement to the final estimation $\boldsymbol{z}_0$, which does not aim to sample from the posterior mode. KDS, in contrast, directly seeks a consensus mode from the final clean output ensemble ($\hat{\boldsymbol{z}}_0$). Moreover, KDS is a training-free, plug-and-play module, whereas InDI requires an integrated training and sampling pipeline.

Table 5: Comparison with other Mode-Seeking Methods on RealSR dataset with DiffBIR backbone.

Method	PSNR $\pm\sigma$	SSIM $\pm\sigma$	LPIPS (↓) $\pm\sigma$	DISTS (↓) $\pm\sigma$	CLIPIQA $\pm\sigma$
DiffBIR	23.21±0.293	0.610±0.015	0.370±0.011	0.250±0.005	0.689±0.013
+ HDS [66]	24.56±0.344⋆	0.646±0.020⋆	0.386±0.017⋆	0.274±0.016⋆	0.647±0.025⋆
+ DGS [67]	24.05±0.287⋆	0.651±0.014⋆	0.355±0.012⋆	0.246±0.004⋆	0.687±0.013
+ KDS (ours)	24.39±0.271⋆	0.669±0.013⋆	0.339±0.011⋆	0.244±0.004⋆	0.692±0.011⋆

Statistically significant differences ($P < 0.05$) compared with DDIM are marked with a ⋆.

6 Limitations & Future Work

Limitations. The primary limitation of KDS is its computational overhead, which scales linearly with particle count N. Although $N = 10 - 15$ provides a good performance-cost trade-off, this overhead could be mitigated by future work on adaptive ensemble sizes or more efficient KDE techniques.

Future Work. As a training-free, plug-and-play framework, KDS is highly generalizable. Promising directions include applying it to other inverse problems (e.g., deblurring, denoising, medical imaging) and video restoration, where its patch-wise mechanism could be extended to 3D spatio-temporal patches to enhance temporal consistency.

7 Conclusion

This work introduces Kernel Density Steering (KDS), a novel inference-time framework that enhances the fidelity and robustness of diffusion-based image restoration. The core of KDS lies in its local mode seeking strategy: it utilizes an N-particle ensemble and computes patch-wise kernel density estimation (KDE) gradients from predicted clean latent patches. These gradients guide samples towards shared, high-density regions, effectively steering them away from spurious modes to produce robust, high-fidelity restorations. As a plug-and-play approach, KDS requires no retraining, external verifiers, or explicit degradation models, making it broadly applicable.

Acknowledgements

The authors would like to thank our colleagues Keren Ye, Viraj Shah, Ashwini Pokle and Mo Zhou for reviewing the manuscript and providing valuable feedback.

References

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

[2] Jonathan Ho, Ajay Jain, and Pieter Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[3] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole, "Score-based generative modeling through stochastic differential equations," in *International Conference on Learning Representations*, 2021.

[4] Xinqi Lin, Jingwen He, Ziyan Chen, Zhaoyang Lyu, Bo Dai, Fanghua Yu, Yu Qiao, Wanli Ouyang, and Chao Dong, "Diffbir: Toward blind image restoration with generative diffusion prior," in *European Conference on Computer Vision*. Springer, 2024, pp. 430–448.

[5] Bin Xia, Yulun Zhang, Shiyin Wang, Yitong Wang, Xinglong Wu, Yapeng Tian, Wenming Yang, and Luc Van Gool, "Diffir: Efficient diffusion model for image restoration," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 13095–13105.

[6] Xin Li, Yulin Ren, Xin Jin, Cuiling Lan, Xingrui Wang, Wenjun Zeng, Xinchao Wang, and Zhibo Chen, "Diffusion models for image restoration and enhancement: a comprehensive survey," *International Journal of Computer Vision*, pp. 1–31, 2025.

[7] Zhenning Shi, Chen Xu, Changsheng Dong, Bin Pan, Along He, Tao Li, Huazhu Fu, et al., "Resfusion: Denoising diffusion probabilistic models for image restoration based on prior residual noise," *Advances in Neural Information Processing Systems*, vol. 37, pp. 130664–130693, 2024.

[8] Yuyang Hu, Satya VVN Kothapalli, Weijie Gan, Alexander L Sukstanskii, Gregory F Wu, Manu Goyal, Dmitriy A Yablonskiy, and Ulugbek S Kamilov, "Diffgepci: 3d mri synthesis from mgre signals using 2.5 d diffusion model," in *2024 IEEE International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2024, pp. 1–4.

[9] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi, "Image super-resolution via iterative refinement," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 4, pp. 4713–4726, 2022.

[10] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi, "Image super-resolution via iterative refinement," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 4, pp. 4713–4726, 2022.

[11] Jianyi Wang, Zongsheng Yue, Shangchen Zhou, Kelvin CK Chan, and Chen Change Loy, "Exploiting diffusion prior for real-world image super-resolution," *International Journal of Computer Vision*, vol. 132, no. 12, pp. 5929–5949, 2024.

[12] Xinyi Zhang, Qiqi Bao, Qingmin Liao, Lu Tian, Zicheng Liu, Zhongdao Wang, Emad Barsoum, et al., "Taming diffusion prior for image super-resolution with domain shift sdes," *Advances in Neural Information Processing Systems*, vol. 37, pp. 42765–42797, 2024.

[13] Zheng Chen, Haotong Qin, Yong Guo, Xiongfei Su, Xin Yuan, Linghe Kong, and Yulun Zhang, "Binarized diffusion model for image super-resolution," in *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

[14] Rongyuan Wu, Lingchen Sun, Zhiyuan Ma, and Lei Zhang, "One-step effective diffusion network for real-world image super-resolution," *Advances in Neural Information Processing Systems*, vol. 37, pp. 92529–92553, 2024.

[15] Jay Whang, Mauricio Delbracio, Hossein Talebi, Chitwan Saharia, Alexandros G Dimakis, and Peyman Milanfar, "Deblurring via stochastic refinement," in *CVPR*, 2022, pp. 16293–16303.

[16] Mengwei Ren, Mauricio Delbracio, Hossein Talebi, Guido Gerig, and Peyman Milanfar, "Multiscale structure guided diffusion for image deblurring," in *ICCV*, 2023, pp. 10721–10733.

[17] Zheng Chen, Yulun Zhang, Ding Liu, Jinjin Gu, Linghe Kong, Xin Yuan, et al., "Hierarchical integration diffusion model for realistic image deblurring," *Advances in neural information processing systems*, vol. 36, 2024.

[18] Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi, "Palette: Image-to-image diffusion models," in *ACM SIGGRAPH 2022 conference proceedings*, 2022, pp. 1–10.

[19] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool, "Repaint: Inpainting using denoising diffusion probabilistic models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 11461–11471.

[20] Ciprian Corneanu, Raghudeep Gadde, and Aleix M Martinez, “Latentpaint: Image inpainting in latent space with diffusion models,” in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2024, pp. 4334–4343.

[21] Prafulla Dhariwal and Alex Nichol, “Diffusion models beat gans on image synthesis,” *Neural Information Processing Systems*, 2021.

[22] Georgios Batzolis, Jan Stanczuk, Carola-Bibiane Schönlieb, and Christian Etmann, “Conditional image generation with score-based diffusion models,” *arXiv preprint arXiv:2111.13606*, 2021.

[23] Kangfu Mei, Hossein Talebi, Mojtaba Ardakani, Vishal M Patel, Peyman Milanfar, and Mauricio Delbracio, “The power of context: How multimodality improves image super-resolution,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[24] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10684–10695.

[25] Yuyang Hu, Suhas Lohit, Ulugbek S Kamilov, and Tim K Marks, “Multimodal diffusion bridge with attention-based sar fusion for satellite image cloud removal,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 63, pp. 1–12, 2025.

[26] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala, “Adding conditional control to text-to-image diffusion models,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 3836–3847.

[27] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen, “LoRA: Low-rank adaptation of large language models,” in *International Conference on Learning Representations*, 2022.

[28] Yochai Blau and Tomer Michaeli, “The perception-distortion tradeoff,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6228–6237.

[29] Mauricio Delbracio and Peyman Milanfar, “Inversion by direct iteration: An alternative to denoising diffusion for image restoration,” *Transactions on Machine Learning Research*, 2023, Featured Certification, Outstanding Certification.

[30] Guy Ohayon, Michael Elad, and Tomer Michaeli, “Perceptual fairness in image restoration,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 70259–70312, 2024.

[31] Guy Ohayon, Tomer Michaeli, and Michael Elad, “Posterior-mean rectified flow: Towards minimum mse photo-realistic image restoration,” *arXiv preprint arXiv:2410.00418*, 2024.

[32] Guy Ohayon, Tomer Michaeli, and Michael Elad, “The perception-robustness tradeoff in deterministic image restoration,” in *Forty-first International Conference on Machine Learning*, 2024.

[33] Rongyuan Wu, Tao Yang, Lingchen Sun, Zhengqiang Zhang, Shuai Li, and Lei Zhang, “Seesr: Towards semantics-aware real-world image super-resolution,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024, pp. 25456–25467.

[34] Sean Man, Guy Ohayon, Ron Raphaeli, and Michael Elad, “Proxies for distortion and consistency with applications for real-world image restoration,” *arXiv preprint arXiv:2501.12102*, 2025.

[35] Murray Rosenblatt, “Remarks on Some Nonparametric Estimates of a Density Function,” *The Annals of Mathematical Statistics*, vol. 27, no. 3, pp. 832 – 837, 1956.

[36] Emanuel Parzen, “On Estimation of a Probability Density Function and Mode,” *The Annals of Mathematical Statistics*, vol. 33, no. 3, pp. 1065 – 1076, 1962.

[37] Patrick Esser, Robin Rombach, and Bjorn Ommer, “Taming transformers for high-resolution image synthesis,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 12873–12883.

[38] Jiaming Song, Chenlin Meng, and Stefano Ermon, “Denoising diffusion implicit models,” in *International Conference on Learning Representations*, 2021.

[39] Chi-Wei Hsiao, Yu-Lun Liu, Cheng-Kun Yang, Sheng-Po Kuo, Kevin Jou, and Chia-Ping Chen, “Ref-ldm: A latent diffusion model for reference-based face image restoration,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 74840–74867, 2024.

[40] Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye, "Diffusion posterior sampling for general noisy inverse problems," in *International Conference on Learning Representations*, 2023.

[41] Yang Song, Liyue Shen, Lei Xing, and Stefano Ermon, "Solving inverse problems in medical imaging with score-based generative models," in *International Conference on Learning Representations*, 2022.

[42] Yinhuai Wang, Jiwen Yu, and Jian Zhang, "Zero-shot image restoration using denoising diffusion null-space model," in *The Eleventh International Conference on Learning Representations*, 2023.

[43] Yuanzhi Zhu, Kai Zhang, Jingyun Liang, Jiezhang Cao, Bihan Wen, Radu Timofte, and Luc Van Gool, "Denoising diffusion models for plug-and-play image restoration," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 1219–1229.

[44] Yuyang Hu, Albert Peng, Weijie Gan, Peyman Milanfar, Mauricio Delbracio, and Ulugbek S. Kamilov, "Stochastic deep restoration priors for imaging inverse problems," in *Forty-second International Conference on Machine Learning*, 2025.

[45] Peyman Milanfar and Mauricio Delbracio, "Denoising: a powerful building block for imaging, inverse problems and machine learning," *Philosophical Transactions A*, vol. 383, no. 2299, pp. 20240326, 2025.

[46] Kangfu Mei, Zhengzhong Tu, Mauricio Delbracio, Hossein Talebi, Vishal M Patel, and Peyman Milanfar, "Bigger is not always better: Scaling properties of latent diffusion models," *Transactions on Machine Learning Research*, 2024.

[47] Nanye Ma, Shangyuan Tong, Haolin Jia, Hexiang Hu, Yu-Chuan Su, Mingda Zhang, Xuan Yang, Yandong Li, Tommi Jaakkola, Xuhui Jia, et al., "Scaling inference time compute for diffusion models," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 2523–2534.

[48] Raghav Singhal, Zachary Horvitz, Ryan Teehan, Mengye Ren, Zhou Yu, Kathleen McKeown, and Rajesh Ranganath, "A general framework for inference-time scaling and steering of diffusion models," in *Forty-second International Conference on Machine Learning*, 2025.

[49] Idan Achituve, Hai Victor Habi, Amir Rosenfeld, Arnon Netzer, Idit Diamant, and Ethan Fetaya, "Inverse problem sampling in latent space using sequential monte carlo," in *Forty-second International Conference on Machine Learning*, 2025.

[50] Luhuan Wu, Brian Trippe, Christian Naesseth, David Blei, and John P Cunningham, "Practical and asymptotically exact conditional sampling in diffusion models," *Advances in Neural Information Processing Systems*, vol. 36, pp. 31372–31403, 2023.

[51] Amir Nazemi, Mohammad Hadi Sepanj, Nicholas Pellegrino, Chris Czarnecki, and Paul Fieguth, "Particle-filtering-based latent diffusion for inverse problems," *arXiv preprint arXiv:2408.13868*, 2024.

[52] Zehao Dou and Yang Song, "Diffusion posterior sampling for linear inverse problem solving: A filtering perspective," in *The Twelfth International Conference on Learning Representations*, 2024.

[53] Keinosuke Fukunaga and Larry Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Transactions on information theory*, vol. 21, no. 1, pp. 32–40, 1975.

[54] Yizong Cheng, "Mean shift, mode seeking, and clustering," *IEEE transactions on pattern analysis and machine intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[55] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu, "Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps," *Advances in Neural Information Processing Systems*, vol. 35, pp. 5775–5787, 2022.

[56] Eirikur Agustsson and Radu Timofte, "NTIRE 2017 challenge on single image super-resolution: Dataset and study," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2017, Honolulu, HI, USA, July 21-26, 2017*. 2017, pp. 1122–1131, IEEE Computer Society.

[57] Jianrui Cai, Hui Zeng, Hongwei Yong, Zisheng Cao, and Lei Zhang, "Toward real-world single image super-resolution: A new benchmark and a new model," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 3086–3095.

[58] Pengxu Wei, Ziwei Xie, Hannan Lu, Zongyuan Zhan, Qixiang Ye, Wangmeng Zuo, and Liang Lin, "Component divide-and-conquer for real-world image super-resolution," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VIII 16*. Springer, 2020, pp. 101–117.

[59] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[60] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang, “The unreasonable effectiveness of deep features as a perceptual metric,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 586–595.

[61] 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*, vol. 30, 2017.

[62] Hossein Talebi and Peyman Milanfar, “Nima: Neural image assessment,” *IEEE transactions on image processing*, vol. 27, no. 8, pp. 3998–4011, 2018.

[63] Sidi Yang, Tianhe Wu, Shuwei Shi, Shanshan Lao, Yuan Gong, Mingdeng Cao, Jiahao Wang, and Yujiu Yang, “Maniqa: Multi-dimension attention network for no-reference image quality assessment,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 1191–1200.

[64] Jianyi Wang, Kelvin CK Chan, and Chen Change Loy, “Exploring clip for assessing the look and feel of images,” in *Proceedings of the AAAI conference on artificial intelligence*, 2023, pp. 2555–2563.

[65] Weixia Zhang, Guangtao Zhai, Ying Wei, Xiaokang Yang, and Kede Ma, “Blind image quality assessment via vision-language correspondence: A multitask learning perspective,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 14071–14081.

[66] Rafal Karczewski, Markus Heinonen, and Vikas Garg, “Diffusion models as cartoonists: The curious case of high density regions,” in *International Conference on Learning Representations*, 2025.

[67] Rafal Karczewski, Markus Heinonen, and Vikas K Garg, “Devil is in the details: Density guidance for detail-aware generation with flow models,” in *International Conference on Machine Learning*, 2025.

[68] Xintao Wang, Liangbin Xie, Chao Dong, and Ying Shan, “Real-esrgan: Training real-world blind super-resolution with pure synthetic data,” in *ICCV*, 2021, pp. 1905–1914.

[69] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu, “Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models,” *Machine Intelligence Research*, pp. 1–22, 2025.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

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

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect our paper's contributions and scope .

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

Justification: We include a specific section to discuss the limitation of this work.

Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [NA]

   Justification: We do not make theoretical claims in this work.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We have included all the information to the experiment details. Due to the limitation of pages in the main paper, we moved some of them to the supplementary material.

   Guidelines:

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

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

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

   Answer: [NA]

   Justification: Some of the code and data is confidential.

   Guidelines:

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

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We have included all the information to the experiment details. Due to the limitation of pages in the main paper, we moved some of them to the supplementary material.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We reported error bars suitably and correctly defined.

   Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We have a table to provide information on the computer resources.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.

- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: This work has no negative societal impact.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [NA]

    Justification: No.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: All the open accessed code we used has been properly cited.

    Guidelines:

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

13. **New assets**

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

    Answer: [NA]

    Justification: No new assets introduced in the paper.

    Guidelines:

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

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

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

    Answer: [NA]

Justification: There is no research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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

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

    Answer: [NA]

    Justification: There is no research with human subjects.

    Guidelines:

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

16. **Declaration of LLM usage**

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

    Answer: [NA]

    Justification: The usage of LLMs is not an important, original, or non-standard component of the core methods in this research

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

A Supplementary Material

This supplementary material provides further details to the main paper. It covers the following aspects:

- **Section A.1: Experiment Details for Real-world Image Super-Resolution**: Including dataset descriptions, backbone model configurations, an in-depth discussion of the patch-wise KDS mechanism with ablation studies, interaction of KDS with various diffusion samplers (DDIM, DPM-Solver++), and additional comparative results on severe degradation datasets.
- **Section A.3: Experiment Details for Image Inpainting**: Covering the datasets used and the hyperparameter settings for the inpainting task.
- **Section A.4: Additional Visual Results**: Presenting more qualitative examples for both super-resolution and image inpainting tasks to further demonstrate the efficacy of KDS.

A.1 Experiment details: Real-world Image Super Resolution

A.1.1 Datasets

Our real-world image super-resolution experiments utilized a synthesized dataset derived from DIV2K [56], alongside two real-world datasets: RealSR [57] and DRealSR [58]. For the DIV2K-based synthesized data, we used the same dataset provided by StableSR [11], which consists of 3,000 randomly cropped patches (resolution: 512×512 each) from the DIV2K validation set [56]. Subsequently, we generated corresponding low-resolution (LR) images (resolution: 128×128)) using the degradation model adopted in Real-ESRGAN [68]. For the RealSR [57] and DRealSR [58] datasets, we adhered to the protocol from [11] to center-crop the provided LR images to 128×128.

A.1.2 Backbone Model Configurations

To assess the effectiveness of our proposed KDS method, we applied it to three established backbone models: LDM-SR [24], DiffBIR [4], and SeeSR [33]. For LDM-SR, the LR image was directly employed as a conditional input, concatenated with the LDM's primary input. For DiffBIR [4] and SeeSR [33], we adopted the hyperparameter settings recommended on their official GitHub pages. This included their specified parameters for text prompting, classifier-free guidance, and the use of their official pre-trained model checkpoints.

A.1.3 Patch-wise KDS Mechanism: Configuration and Discussion

As introduced in Section 3.1, we employ a patch-wise mechanism to facilitate mode-seeking within the high-dimensional latent space. The motivation is that the accuracy of the mode-seeking process is intrinsically linked to both the number of samples available for kernel density estimation (KDE) and the dimensionality of these samples. In our case, direct mode-seeking on the entire $64 \times 64 \times 8$ latent vector $\boldsymbol{z}$ (which encodes features for the $512 \times 512 \times 3$ image space) is impractical with a practical number of particles (e.g., 10-20).

Patch Size Configuration: To effectively implement our patch-wise strategy, the patch size is chosen to balance accurate mode-seeking with computational efficiency. Accurate mode-seeking across the entire latent space would necessitate thousands of samples, which is impractical for real-world applications. Therefore, to achieve robust estimation with minimal samples, we set the patch size to 1×1. This processes each spatial location (h, w) in the latent map as an individual 8-dimensional vector (i.e., batches of $1 \times 1 \times 8$ vectors). This strategic choice significantly reduces the dimensionality for each kernel density estimation (KDE), enabling accurate estimation with limited samples while leveraging the rich, learned features of the 8 channels. Consequently, this facilitates more effective mode-seeking with a limited particle count. Ablation studies (Table 6) demonstrate that with a restricted particle count ($N = 10$), larger patch sizes lead to inaccurate mode-seeking and sub-optimal performance.

Steering Strength $\boldsymbol{\delta_t}$ Configuration: Another key hyperparameter in KDS is the steering strength, denoted as δ_t. During the diffusion process, the sampling procedure exhibits varying sensitivity to guidance. Specifically, at later stages of the sampling process (i.e., smaller t values, approaching

Table 6: Ablation Study on Patch-size ($N = 10$)

	PSNR	SSIM	LPIPS ↓	FID
DDIM	22.05	0.531	0.307	20.89
+ KDS (patch-size=1)	22.52	0.555	0.289	20.33
+ KDS (patch-size=4)	22.35	0.547	0.296	20.77
+ KDS (patch-size=16)	22.17	0.538	0.304	20.86

the data), the model can be more sensitive. To ensure stability and effective guidance, we define δ_t conditionally based on the timestep t. Assuming T is the total number of diffusion steps, we set:

$$\delta_t = \begin{cases} 0 & \text{if } t/T < 0.3 \\ 0.3 & \text{if } t/T \geq 0.3 \end{cases} \tag{11}$$

Note that, this hyperparameter setting is fixed for all the experiments across different applications and backbones.

A.1.4 Integration with Standard Diffusion Samplers

As discussed in the main paper, KDS is a flexible, plug and play approach that can be applied to all existing samplers. In this section, we will detailed introduce how to apply KDS to DDIM, and DPM-solver and how to interactive with Classifier-Free Guidance.

Interaction with DDIM: To better illustrate the plug-and-play nature of KDS, we provide pseudocode for three scenarios:

1. **Algorithm 1:** Standard DDIM sampling.
2. **Algorithm 2:** DDIM integrated with KDS.
3. **Algorithm 3:** DDIM combined with Classifier-Free Guidance (CFG) and KDS.

As demonstrated in the pseudocode, KDS functions as a straightforward plug-in module (Line 5 - 15 in Algorithm 2). It enhances the predicted ensemble $\hat{\mathbf{Z}}^{\text{pred}}_{0|t}$ and maintain the rest sampling design of the base sampler.

Interaction with Classifier-Free Guidance (CFG): CFG is frequently used in conditional diffusion models for SR (like DiffBIR [4], SeeSR [33]) to further improve perceptual quality. CFG is applied by adjusting the noise prediction, commonly via the extrapolation formula

$$\tilde{\boldsymbol{\epsilon}}(\boldsymbol{z}_t, t, \boldsymbol{c}) = \boldsymbol{\epsilon}(\boldsymbol{z}_t, t, \varnothing) + w(\boldsymbol{\epsilon}(\boldsymbol{z}_t, t, \boldsymbol{c}) - \boldsymbol{\epsilon}(\boldsymbol{z}_t, t, \varnothing)), \tag{12}$$

where w is the guidance scale. While higher w can enhance perception, it sometimes introduces artifacts. We investigated how KDS interacts with CFG by varying w in the DiffBIR model using DDIM sampling on the DrealSR dataset. As shown in Table 7, KDS consistently boosts performance across different CFG strengths ($w = 1, 2, 4$). For each value of w, adding KDS leads to substantial improvements in PSNR and SSIM, along with generally better perceptual metrics (LPIPS, NIMA, CLIPIQA). This suggests that KDS provides benefits complementary to CFG, enhancing fidelity without hindering the perceptual adjustments offered by CFG.

Table 7: Performance of DDIM with Varying CFG Weights w on DrealSR Dataset.

Method	PSNR	SSIM	LPIPS (↓)	DISTS(↓)
DDIM ($w = 1$)	25.11	0.576	0.492	0.298
+ KDS ($N = 10$)	**26.94**	**0.677**	**0.427**	**0.283**
DDIM ($w = 2$)	24.75	0.569	0.486	0.293
+ KDS ($N = 10$)	**26.60**	**0.667**	**0.428**	**0.278**
DDIM ($w = 4$)	23.99	0.551	0.491	0.293
+ KDS ($N = 10$)	**25.63**	**0.645**	**0.422**	**0.276**
DDIM ($w = 6$)	23.27	0.534	0.497	0.296
+ KDS ($N = 10$)	**25.04**	**0.629**	**0.440**	**0.282**

Algorithm 1 Standard DDIM Sampling

```
Require: Model $\boldsymbol{\epsilon}_\theta$, condition: $\boldsymbol{c}$, Schedule $\bar{\alpha}_t$
1: $\boldsymbol{z}_T \sim \mathcal{N}(0, \mathbf{I})$                                        ▷ Init noise
2: for $t = T, \dots, 1$ do
3:     $\boldsymbol{\epsilon}_{\theta,t} \leftarrow \boldsymbol{\epsilon}_\theta(\boldsymbol{z}_t, t, \boldsymbol{c})$                              ▷ Predict noise
4:     $\hat{\boldsymbol{z}}_{0|t} \leftarrow (\boldsymbol{z}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_{\theta,t})/\sqrt{\bar{\alpha}_t}$              ▷ Predict $\boldsymbol{z}_0$
5:     $\boldsymbol{\epsilon}'_t \leftarrow (\boldsymbol{z}_t - \sqrt{\bar{\alpha}_t}\hat{\boldsymbol{z}}_{0|t})/\sqrt{1-\bar{\alpha}_t}$       ▷ Update direction based on $\hat{\boldsymbol{z}}_{0|t}$
6:     $\boldsymbol{z}_{t-1} \leftarrow \sqrt{\bar{\alpha}_{t-1}}\hat{\boldsymbol{z}}_{0|t} + \sqrt{1-\bar{\alpha}_{t-1}}\boldsymbol{\epsilon}'_t$               ▷ DDIM step
7: end for
8: return $\boldsymbol{z}_0$
```

Algorithm 2 DDIM + KDS

```
Require: Model $\boldsymbol{\epsilon}_\theta$, Schedule $\bar{\alpha}_t$, condition: $\boldsymbol{c}$, Number of particles: $N$, bandwidth: $h$, steering
    strength: $\delta_t$, PatchSize
 1: $\boldsymbol{Z}_T \sim \mathcal{N}(0, \mathbf{I})$ (ensemble of $N$ samples)    ▷ Init noise ensemble
 2: for $t = T, \ldots, 1$ do
 3:     $\mathbf{E}_{\theta,t} \leftarrow \boldsymbol{\epsilon}_\theta(\boldsymbol{Z}_t, t, \boldsymbol{c})$    ▷ Predict ensemble noise
 4:     $\hat{\boldsymbol{Z}}_{0|t}^{\text{pred}} \leftarrow (\boldsymbol{Z}_t - \sqrt{1-\bar{\alpha}_t}\mathbf{E}_{\theta,t})/\sqrt{\bar{\alpha}_t}$    ▷ Predict $\mathbf{x}_0$ ensemble
 5:     Patches $\leftarrow$ Patchify($\hat{\boldsymbol{Z}}_{0|t}^{\text{pred}}$)    ▷ Extract all non-overlapped patches: Patches$[k, loc]$.
 6:     for each patch location $j$ do    ▷ This loop over patch locations, can be executed in parallel.
 7:         $\boldsymbol{P}_j \leftarrow$ Patches$[:, j]$    ▷ Ensemble of $N$ original patches at location $j$.
 8:         for $i = 1, \ldots, N$ do    ▷ For particle $i$'s patch at location $j$, can be computed in parallel.
 9:             $\boldsymbol{p}_j^{(i)} \leftarrow \boldsymbol{P}_j[i]$    ▷ Patch from particle $i$ at location $j$.
10:             $\mathbf{m}(\boldsymbol{p}_j^{(i)}) \leftarrow \frac{\sum_{k=1}^{N} G\left(\frac{\|\boldsymbol{p}_j^{(i)} - \boldsymbol{P}_j^{(k)}\|^2}{h^2}\right)\boldsymbol{P}_j^{(k)}}{\sum_{k=1}^{N} G\left(\frac{\|\boldsymbol{p}_j^{(i)} - \boldsymbol{P}_j^{(k)}\|^2}{h^2}\right)} - \boldsymbol{p}_j^{(i)}$    ▷ Mean shift vector (Eq. 8).
11:             $\hat{\boldsymbol{p}}_j^{(i),\text{KDS}} \leftarrow \boldsymbol{p}_j^{(i)} + \delta_t \mathbf{m}(\boldsymbol{p}_j^{(i)})$    ▷ Apply steering (Eq. 9)
12:             Patches$[i, j] \leftarrow \hat{\boldsymbol{p}}_j^{(i),\text{KDS}}$    ▷ Update the patch set with the guided patch.
13:         end for
14:     end for
15:     $\hat{\boldsymbol{Z}}_{0|t}^{\text{KDS}} \leftarrow$ Unpatchify(Patches)    ▷ Reconstruct guided latent prediction.
16:     $\mathbf{E}'_t \leftarrow (\boldsymbol{Z}_t - \sqrt{\bar{\alpha}_t}\hat{\boldsymbol{Z}}_{0|t}^{\text{KDS}})/\sqrt{1-\bar{\alpha}_t}$    ▷ Update direction
17:     $\boldsymbol{Z}_{t-1} \leftarrow \sqrt{\bar{\alpha}_{t-1}}\hat{\boldsymbol{Z}}_{0|t}^{\text{KDS}} + \sqrt{1-\bar{\alpha}_{t-1}}\mathbf{E}'_t$    ▷ DDIM step
18: end for
19: return $\hat{\boldsymbol{Z}}_0^{\text{KDS}}$    ▷ Return KDS-guided result
```

Algorithm 3 DDIM + CFG + KDS

```
Require: Model $\boldsymbol{\epsilon}_\theta$, Schedule $\bar{\alpha}_t$, condition: $\boldsymbol{c}$, Number of particles: $N$, bandwidth: $h$, steering
    strength: $\delta_t$, CFG strength: $w$, PatchSize
1: $\boldsymbol{Z}_T \sim \mathcal{N}(0, \mathbf{I})$ (ensemble of $N$ samples)    ▷ Init noise ensemble
2: for $t = T, \ldots, 1$ do
3:     $\mathbf{E}_{\theta,t} \leftarrow \boldsymbol{\epsilon}_\theta(\boldsymbol{Z}_t, t, \varnothing) + w(\boldsymbol{\epsilon}_\theta(\boldsymbol{Z}_t, t, \boldsymbol{c}) - \boldsymbol{\epsilon}_\theta(\boldsymbol{Z}_t, t, \varnothing))$    ▷ Predict ensemble noise with CFG
4:     $\hat{\boldsymbol{Z}}_{0|t}^{\text{pred}} \leftarrow (\boldsymbol{Z}_t - \sqrt{1-\bar{\alpha}_t}\mathbf{E}_{\theta,t})/\sqrt{\bar{\alpha}_t}$    ▷ Predict $\mathbf{x}_0$ ensemble
5:     $\hat{\boldsymbol{Z}}_{0|t}^{\text{KDS}} \leftarrow$ Patch-wise KDS($\hat{\boldsymbol{Z}}_{0|t}^{\text{pred}}$)    ▷ Same as Step 5-15 in Algorithm 2
6:     $\mathbf{E}'_t \leftarrow (\boldsymbol{Z}_t - \sqrt{\bar{\alpha}_t}\hat{\boldsymbol{Z}}_{0|t}^{\text{KDS}})/\sqrt{1-\bar{\alpha}_t}$    ▷ Update direction based on KDS-guided $\mathbf{x}_0$
7:     $\boldsymbol{Z}_{t-1} \leftarrow \sqrt{\bar{\alpha}_{t-1}}\hat{\boldsymbol{Z}}_{0|t}^{\text{KDS}} + \sqrt{1-\bar{\alpha}_{t-1}}\mathbf{E}'_t$    ▷ DDIM step with KDS-guided $\mathbf{x}_0$
8: end for
9: return $\hat{\boldsymbol{Z}}_0^{\text{KDS}}$    ▷ Return KDS-guided result
```

Algorithm 4 DPM-Solver++.

```
Require: initial value $\boldsymbol{Z}_T$, time steps $\{t_i\}_{i=0}^{M}$ and $\{s_i\}_{i=1}^{M}$, data prediction model $\boldsymbol{Z}_\theta$.
 1: $\boldsymbol{Z}_T \sim \mathcal{N}(0, \mathbf{I})$ (ensemble of $N$ samples)                ▷ Init noise ensemble
 2: $\tilde{\boldsymbol{Z}}_{t_0} \leftarrow \boldsymbol{Z}_T$.
 3: for $i \leftarrow 1$ to $M$ do
 4:     $h_i \leftarrow \lambda_{t_i} - \lambda_{t_{i-1}}$
 5:     $r_i \leftarrow \frac{\lambda_{s_i} - \lambda_{t_{i-1}}}{h_i}$
 6:     $\hat{\boldsymbol{Z}}_{0|t}^{\text{pred}} \leftarrow \boldsymbol{Z}_\theta(\tilde{\boldsymbol{Z}}_{t_{i-1}}, t_{i-1})$
 7:     $\boldsymbol{U}_i \leftarrow \frac{\sigma_{s_i}}{\sigma_{t_{i-1}}}\tilde{\boldsymbol{Z}}_{t_{i-1}} - \alpha_{s_i}\left(e^{-r_i h_i} - 1\right)\hat{\boldsymbol{Z}}_{0|t}^{\text{pred}}$
 8:     $\hat{\boldsymbol{U}}_{0|s}^{\text{pred}} \leftarrow \boldsymbol{Z}_\theta(\boldsymbol{U}_i, s_i)$
 9:     $\boldsymbol{D}_i \leftarrow (1 - \frac{1}{2r_i})\hat{\boldsymbol{Z}}_{0|t}^{\text{pred}} + \frac{1}{2r_i}\hat{\boldsymbol{U}}_{0|s}^{\text{pred}}$
10:     $\tilde{\boldsymbol{Z}}_{t_i} \leftarrow \frac{\sigma_{t_i}}{\sigma_{t_{i-1}}}\tilde{\boldsymbol{Z}}_{t_{i-1}} - \alpha_{t_i}\left(e^{-h_i} - 1\right)\boldsymbol{D}_i$
11: end for
12: return $\tilde{\boldsymbol{Z}}_{t_M}$
```

Algorithm 5 DPM-Solver++ with KDS.

```
Require: initial value $\boldsymbol{Z}_T$, time steps $\{t_i\}_{i=0}^{M}$ and $\{s_i\}_{i=1}^{M}$, data prediction model $\boldsymbol{Z}_\theta$.
 1: $\boldsymbol{Z}_T \sim \mathcal{N}(0, \mathbf{I})$ (ensemble of $N$ samples)                ▷ Init noise ensemble
 2: $\tilde{\boldsymbol{Z}}_{t_0} \leftarrow \boldsymbol{Z}_T$.
 3: for $i \leftarrow 1$ to $M$ do
 4:     $h_i \leftarrow \lambda_{t_i} - \lambda_{t_{i-1}}$
 5:     $r_i \leftarrow \frac{\lambda_{s_i} - \lambda_{t_{i-1}}}{h_i}$
 6:     $\hat{\boldsymbol{Z}}_{0|t}^{\text{pred}} \leftarrow \boldsymbol{Z}_\theta(\tilde{\boldsymbol{Z}}_{t_{i-1}}, t_{i-1})$
 7:     $\hat{\boldsymbol{Z}}_{0|t}^{\text{KDS}} \leftarrow$ Patch-wise KDS$(\hat{\boldsymbol{Z}}_{0|t}^{\text{pred}})$        ▷ Same as Step 5-15 in Algorithm 2
 8:     $\boldsymbol{U}_i \leftarrow \frac{\sigma_{s_i}}{\sigma_{t_{i-1}}}\tilde{\boldsymbol{Z}}_{t_{i-1}} - \alpha_{s_i}\left(e^{-r_i h_i} - 1\right)\hat{\boldsymbol{Z}}_{0|t}^{\text{KDS}}$
 9:     $\hat{\boldsymbol{U}}_{0|s}^{\text{pred}} \leftarrow \boldsymbol{Z}_\theta(\boldsymbol{U}_i, s_i)$
10:     $\hat{\boldsymbol{U}}_{0|s}^{\text{KDS}} \leftarrow$ Patch-wise KDS$(\hat{\boldsymbol{U}}_{0|s}^{\text{pred}})$        ▷ Same as Step 5-15 in Algorithm 2
11:     $\boldsymbol{D}_i \leftarrow (1 - \frac{1}{2r_i})\hat{\boldsymbol{Z}}_{0|t}^{\text{KDS}} + \frac{1}{2r_i}\hat{\boldsymbol{U}}_{0|s}^{\text{KDS}}$
12:     $\tilde{\boldsymbol{Z}}_{t_i} \leftarrow \frac{\sigma_{t_i}}{\sigma_{t_{i-1}}}\tilde{\boldsymbol{Z}}_{t_{i-1}} - \alpha_{t_i}\left(e^{-h_i} - 1\right)\boldsymbol{D}_i$
13: end for
14: return $\tilde{\boldsymbol{Z}}_{t_M}$
```

A.1.5 Additional Comparisons on Various Degradations and Samplers

In this subsection, we further evaluate KDS's performance on several additional real-world SR degradations and its effectiveness as a plug-in module for DPM-Solver++ [69]. We introduce a novel dataset, DF2K, generated by synthesizing 3,000 randomly degraded image pairs from the original DF2K dataset. While adopting the Real-ESRGAN pipeline, we employed hyperparameters that introduce more significant blur, noise, and JPEG artifacts, making it a more challenging benchmark compared to standard degradation levels, such as those in DIV2K. As shown in Table 8, KDS consistently improves performance on both the challenging DF2K dataset and the standard DIV2K degradation dataset.

Table 8: Performance with LDM-SR backbone on different Real-world SR degradation levels.

Datasets	DF2k					DIV2k				
Metrics	PSNR	SSIM	LPIPS (↓)	NIMA	FID (↓)	PSNR	SSIM	LPIPS (↓)	NIMA	FID (↓)
DPM-Solver++	23.11	0.579	0.276	4.968	18.76	22.06	0.532	0.306	4.922	20.88
+ KDS	**23.70**	**0.594**	**0.265**	**4.972**	**18.38**	**22.29**	**0.542**	**0.290**	**4.947**	**20.65**
DDIM	22.88	0.542	0.276	4.930	18.59	22.05	0.531	0.307	4.922	20.89
+ KDS	**23.71**	**0.597**	**0.261**	**4.943**	**18.11**	**22.37**	**0.549**	**0.292**	**4.949**	**20.78**

A.1.6 Additional comparison with Best-of-N approaches:

While inference-time scaling allows for generating multiple candidate solutions, particularly useful for low-quality inputs, selecting the optimal one from an N-particle ensemble remains a challenge. In our main paper, we didn't cover the full scope of this experiment. Here, we expand on that by including more metrics. As Table 9 now demonstrates, using non-reference metrics like LIQE [65] or ClipiQA to pick the best particle from an N-selection (i.e., "best LIQE best of N" or "best CLIPIQA best of N") results in significantly lower performance compared to our Kernel Density Steering (KDS) method. This shows that, despite comparable computational costs, traditional post-sampling selection methods don't achieve the same performance level as KDS. As shown in Figure 8, KDS method achieves the most stable performance compared with both BoN baselines, which suffers from the artifacts which confused the non-reference metrics.

Table 9: Comparison of BoN Selection Methods

Method	PSNR	SSIM	LPIPS (↓)	DISTS (↓)	LIQE	CLIPIQA	Time(s)	Memory
DDIM	23.21	0.610	0.370	0.250	4.046	0.689	7.9s	8.0Gb
+ BoN (LIQE)	23.72	0.622	0.361	0.246	**4.351**	0.741	27.3s	10.4Gb
+ BoN (CLIPIQA)	23.01	0.592	0.382	0.247	4.187	**0.774**	28.9s	10.4Gb
+ KDS	**24.39**	**0.669**	**0.339**	**0.245**	3.819	0.692	28.4s	10.4Gb

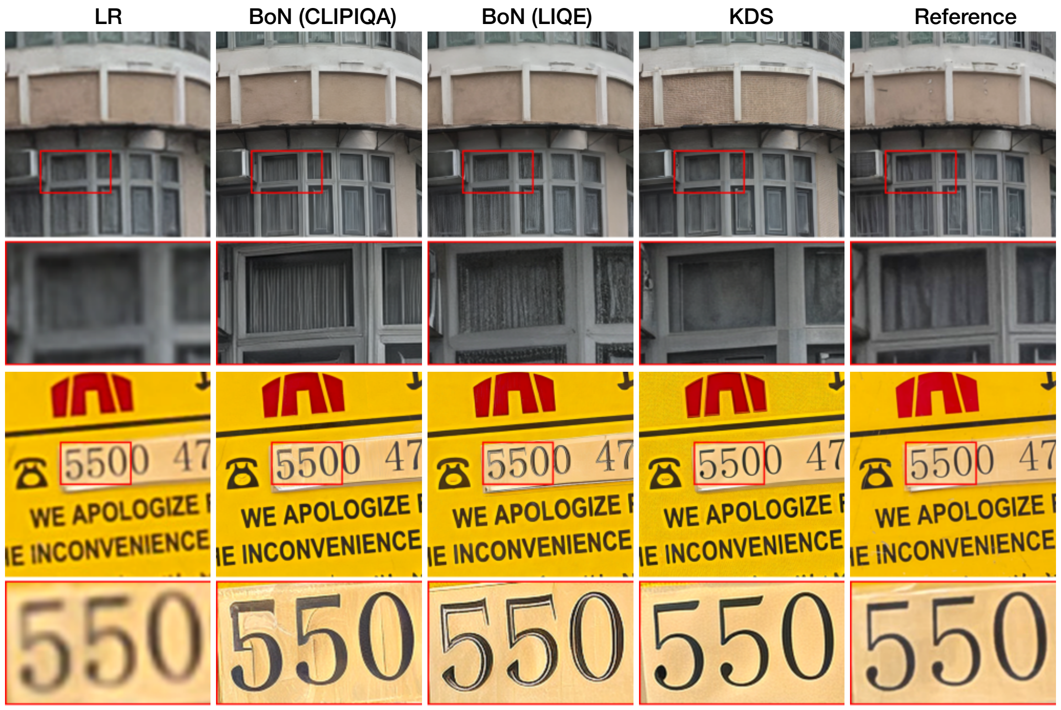

Figure 8: Visual comparison between KDS and Best-of-N (BoN) selection. BoN (CLIPIQA) and BoN (LIQE) means the best particle in terms of these two metrics correspondingly.

A.1.7 KDS with post-hoc Best-of-N Selection:

We also investigate replacing the standard KDS final particle selection (Equation 10) with a post-hoc BoN selection. This involves first guiding the entire ensemble using KDS and then applying a non-reference metric (CLIPIQA) to select the best particle from the guided ensemble.

As shown in Table 10, this combination further validates our approach. KDS first enhances the overall quality of the ensemble, making the subsequent BoN selection more reliable. Without KDS, BoN selects from a noisy set of candidates and favors samples with artifacts that score well. With KDS, the candidates are already steered toward a high-fidelity consensus, providing a cleaner, more faithful set for the BoN selector. This results in a better perceptual–distortion trade-off than using BoN alone.

In summary, KDS provides a principled, task-aligned alternative to BoN by enforcing ensemble consistency. It improves performance across the board and can even strengthen BoN when the two are combined.

Table 10: Comparison of BoN and BoN + KDS Methods

Method	PSNR	SSIM	LPIPS (↓)	DISTS (↓)	LIQE	CLIPIQA
DDIM	23.21	0.610	0.370	0.250	4.046	0.689
+ BoN (CLIPIQA)	23.01	0.592	0.382	0.247	**4.187**	**0.774**
+ KDS	**24.39**	**0.669**	**0.339**	**0.245**	3.819	0.692
+ KDS+ BoN (CLIPIQA)	23.69	0.636	0.359	0.243	4.161	0.770

A.2 Additional Ablation Studies

This section provides the full empirical results for the ablation studies and comparisons mentioned in the main paper, including the full hyperparameter sweep for mode-seeking baselines and detailed ablations on KDS's components.

A.2.1 Mode-Seeking Method Parameter Sweep

Table 11 provides the full parameter sweep for HDS [66] and DGS [67] on the RealSR dataset, supporting the analysis in Section 5.

Table 11: Full parameter sweep for HDS [66] and DGS [67] on RealSR.

Method	PSNR	SSIM	LPIPS (↓)	DISTS (↓)	CLIPIQA
DiffBIR (Baseline)	23.21	0.610	0.370	0.250	0.689
+ HDS (t=20)	23.95	0.614	0.382	0.254	0.651
+ HDS (t=50)	24.33	0.633	0.385	0.266	0.650
+ HDS (t=100)	24.56	0.646	0.386	0.274	0.647
+ HDS (t=200)	25.00	0.665	0.378	0.288	0.567
+ HDS (t=500)	25.79	0.679	0.377	0.309	0.295
+ DGS (q=0.1)	22.57	0.580	0.381	0.253	0.679
+ DGS (q=0.3)	22.95	0.598	0.375	0.252	0.674
+ DGS (q=0.5)	23.21	0.610	0.370	0.249	0.689
+ DGS (q=0.99)	24.05	0.651	0.355	0.246	0.687
+ DGS (q=0.999)	24.03	0.647	0.357	0.246	0.680
+ KDS (N=10)	**24.39**	**0.669**	**0.339**	**0.244**	**0.692**

A.2.2 Ablation on Guidance and Kernel Functions

Table 12 investigates the choice of kernel function and compares our mode-seeking approach to a simpler L2 guidance. The results show consistent performance between Gaussian and Laplacian kernels, demonstrating robustness. Both kernel-based methods outperform the L2 guidance baseline in perceptual metrics, validating our mode-seeking design.

Table 12: KDS with different guidance and kernels on RealSR dataset.

Method	PSNR	SSIM	LPIPS (↓)	DISTS (↓)	CLIPIQA
DiffBIR	23.21	0.610	0.370	0.250	0.689
+ KDS (Gaussian)	**24.39**	0.669	0.339	0.244	**0.692**
+ KDS (Laplacian)	24.30	0.665	**0.337**	**0.241**	0.690
+ KDS (L2)	**24.63**	**0.679**	0.349	0.257	0.663

A.2.3 Ablation on Automatic Bandwidth Selection

Table 13 demonstrates KDS's robustness to bandwidth selection. An automatic *median heuristic* strategy (auto bw) achieves performance nearly identical to the manually tuned bandwidth (bw=0.3), confirming KDS is not reliant on tedious tuning.

Table 13: Effect of automatic bandwidth selection for KDS.

Method	PSNR	SSIM	LPIPS (↓)	DISTS (↓)	CLIPIQA
DiffBIR	23.21	0.610	0.370	0.250	0.689
+ KDS (bw=0.3)	**24.39**	**0.669**	0.339	0.244	**0.692**
+ KDS (auto bw)	24.24	0.661	**0.338**	**0.237**	0.691

A.2.4 Ablation on Patch Size and Particle Count

Table 14 explores the trade-off between patch size and particle count (N). As shown, increasing the number of particles (e.g., from $N = 10$ to $N = 40$) allows for the effective use of a larger patch size (from 1 to 4), which in turn enhances overall performance by better mitigating the curse of dimensionality.

Table 14: Ablation Study on patch size and particle count (N).

	PSNR	SSIM	LPIPS $\downarrow$	FID $\downarrow$
DDIM	22.05	0.531	0.307	20.89
+ KDS (patch-size=1, $N = 10$)	22.52	0.555	0.289	20.33
+ KDS (patch-size=4, $N = 40$)	22.70	0.563	0.281	20.08

A.3 Experiment details: Image Inpainting

Datasets We generated our inpainting test set by center cropping 30% square region of each image. We generate used first 1,000 images from ImageNet testset.

Hyperparameters: Similar to real-world SR settings, we fixed the patch size to 1, bandwidth h to 0.3, steering strength δ_t same as introduced in (Eq: 11).

A.4 Additional Visual Results

This section provides additional qualitative results to visually demonstrate the effectiveness of Kernel Density Steering (KDS). The figures included are:

- **Figure 9, Figure 10 and Figure 11:** These figures showcase super-resolution performance on the DIV2K dataset using LDM-SR, DiffBIR and SeeSR backbones, respectively. They illustrate improvements in sharpness and detail recovery achieved with KDS.
- **Figure 12:** This figure highlights KDS's robustness, demonstrating its performance on the more challenging DF2K real-world SR dataset with the LDM-SR backbone.
- **Figures 13, 14, and 15:** These figures display image inpainting results on the ImageNet dataset using LDM-inpainting. Each figure presents all 10 particles sampled with standard DDIM versus DDIM enhanced with KDS. They visually confirm KDS's ability to improve fidelity and reduce artifacts across the ensemble for the inpainted regions.

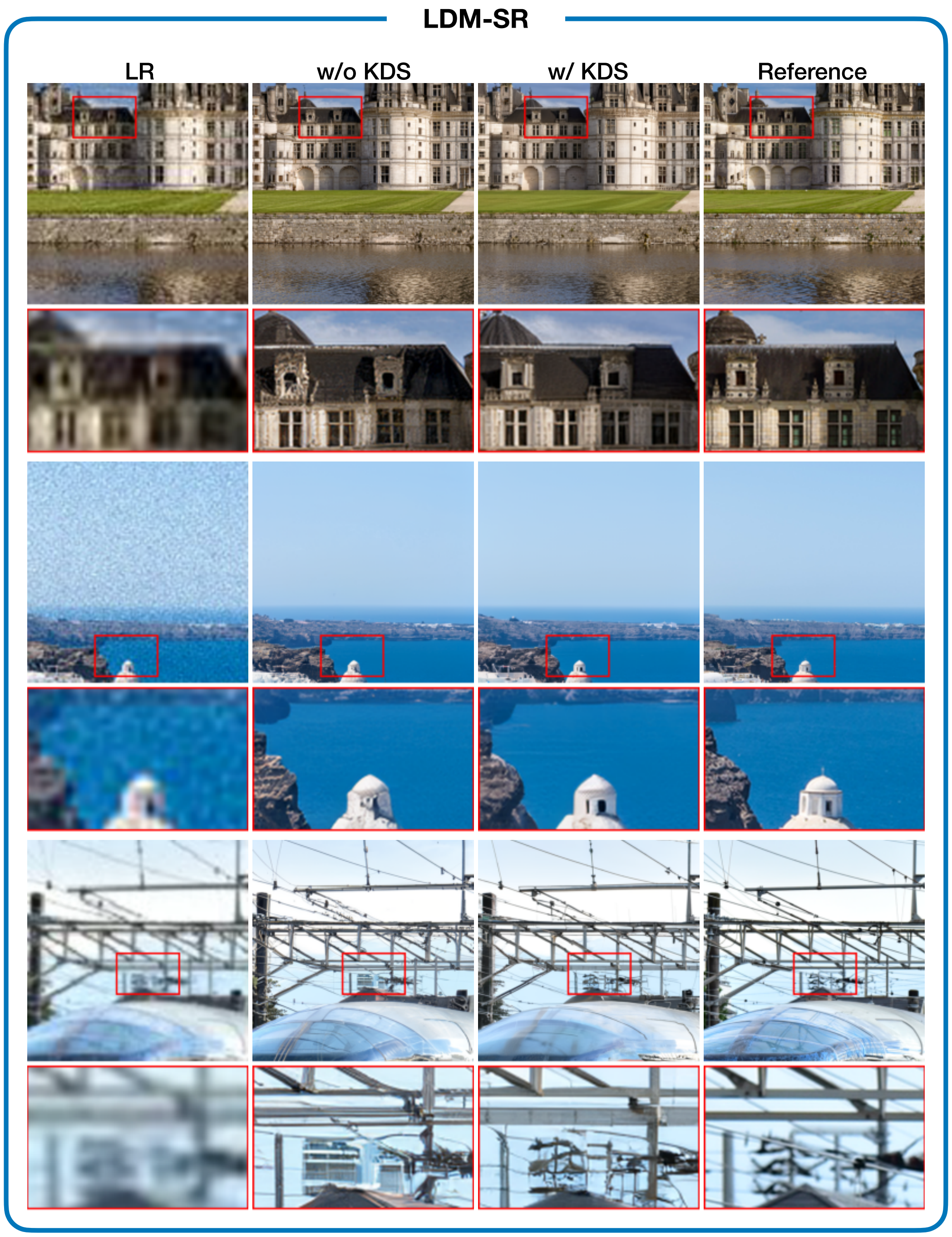

Figure 9: Real-world image super-resolution performance with LDM-SR on DIV2K dataset.

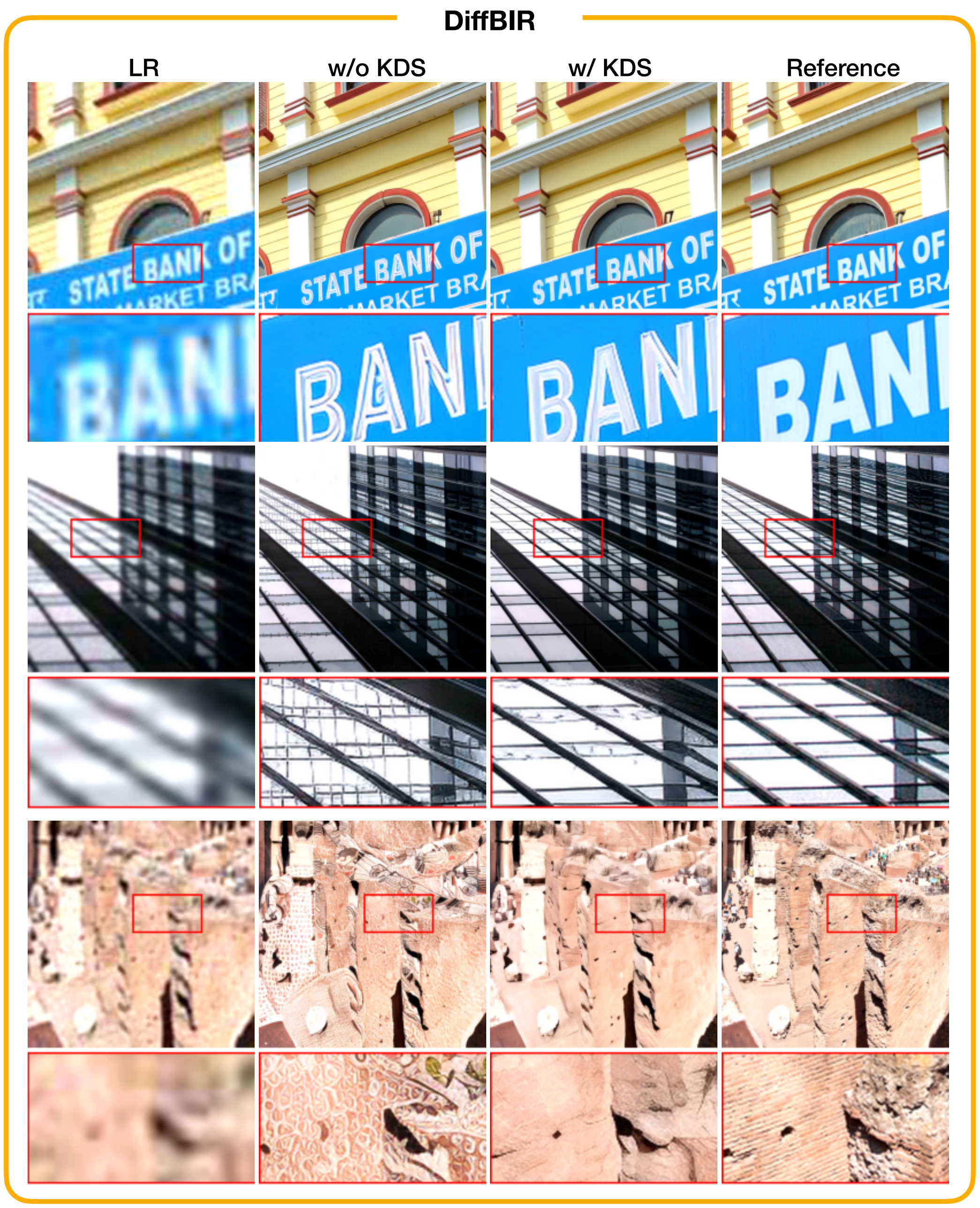

Figure 10: Real-world image super-resolution performance with DiffBIR on DIV2K dataset.

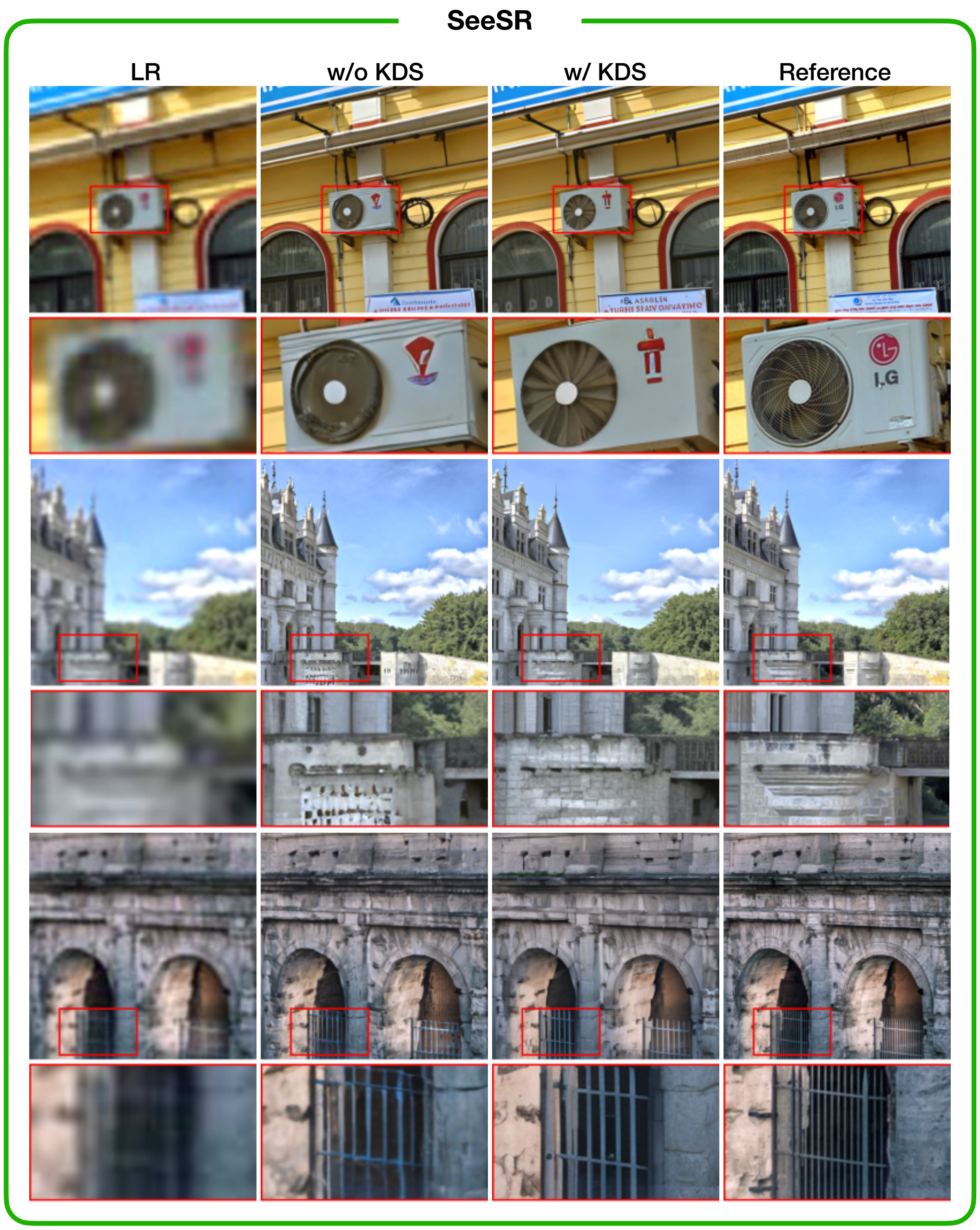

Figure 11: Real-world image super-resolution performance with SeeSR on DIV2K dataset.

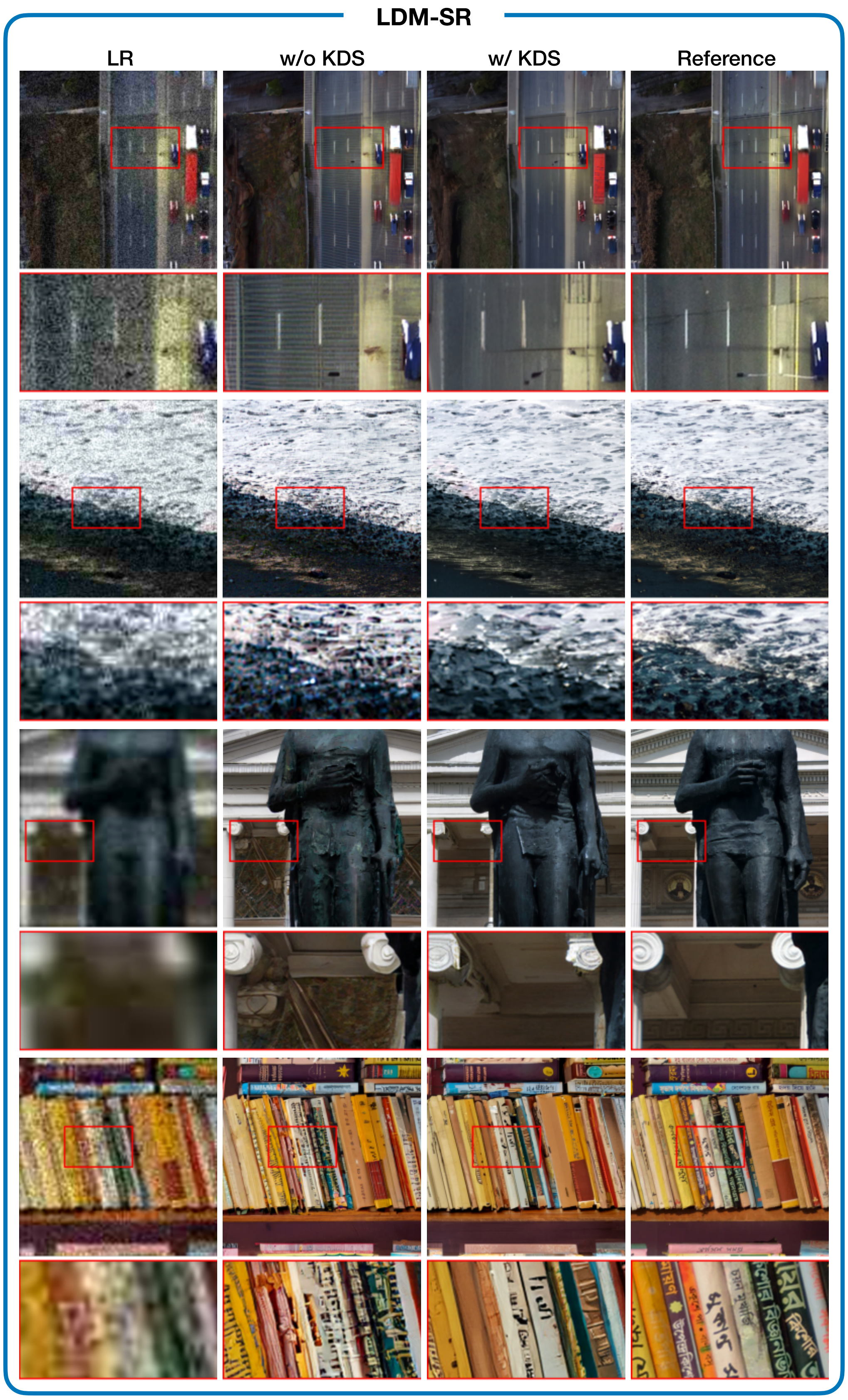

Figure 12: Real-world image super-resolution performance with LDM-SR on DF2K dataset.

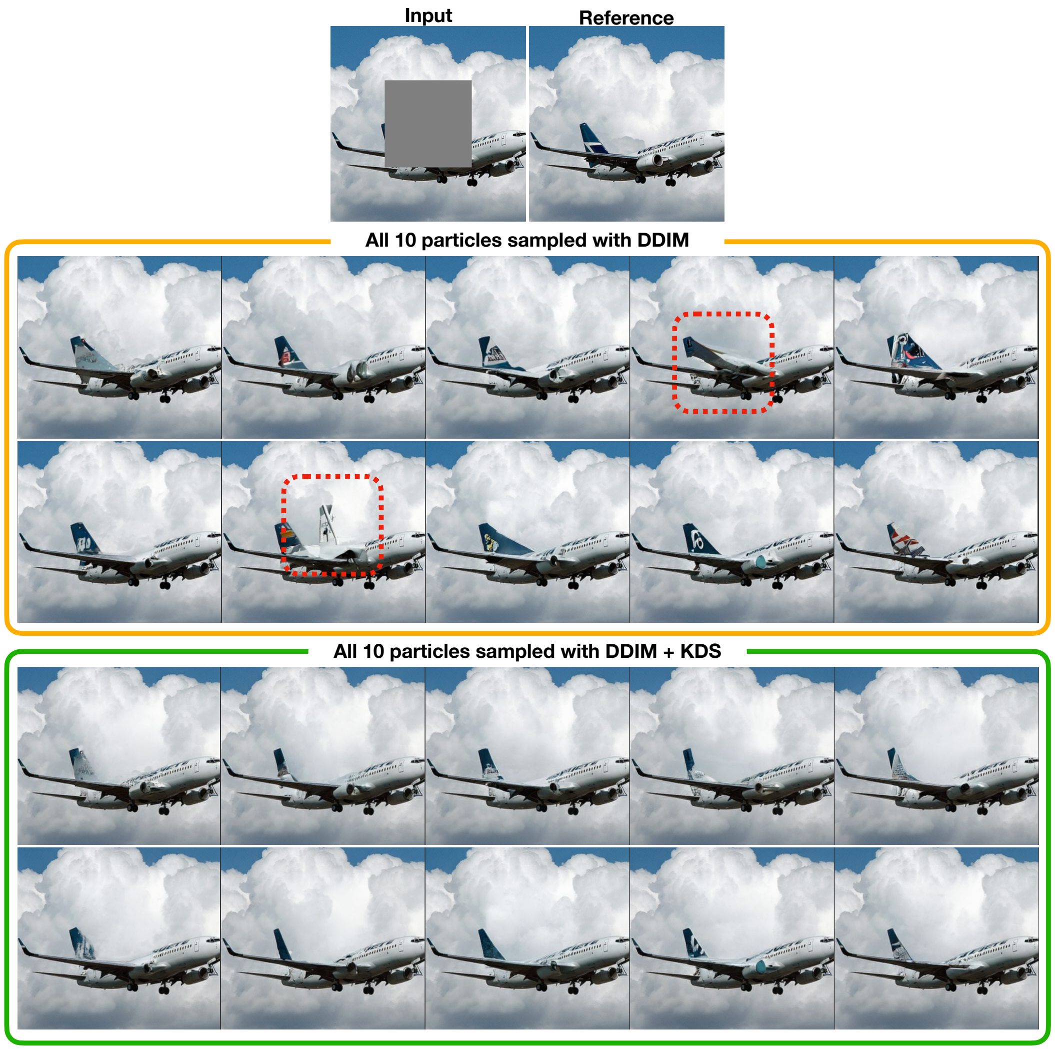
Input
Reference
All 10 particles sampled with DDIM
All 10 particles sampled with DDIM + KDS

Figure 13: Image Inpainting performance with LDM-inpainting on ImageNet dataset. Visualizes all 10 particles for DDIM vs. DDIM + KDS. Regions with artifacts were highlighted with red box.

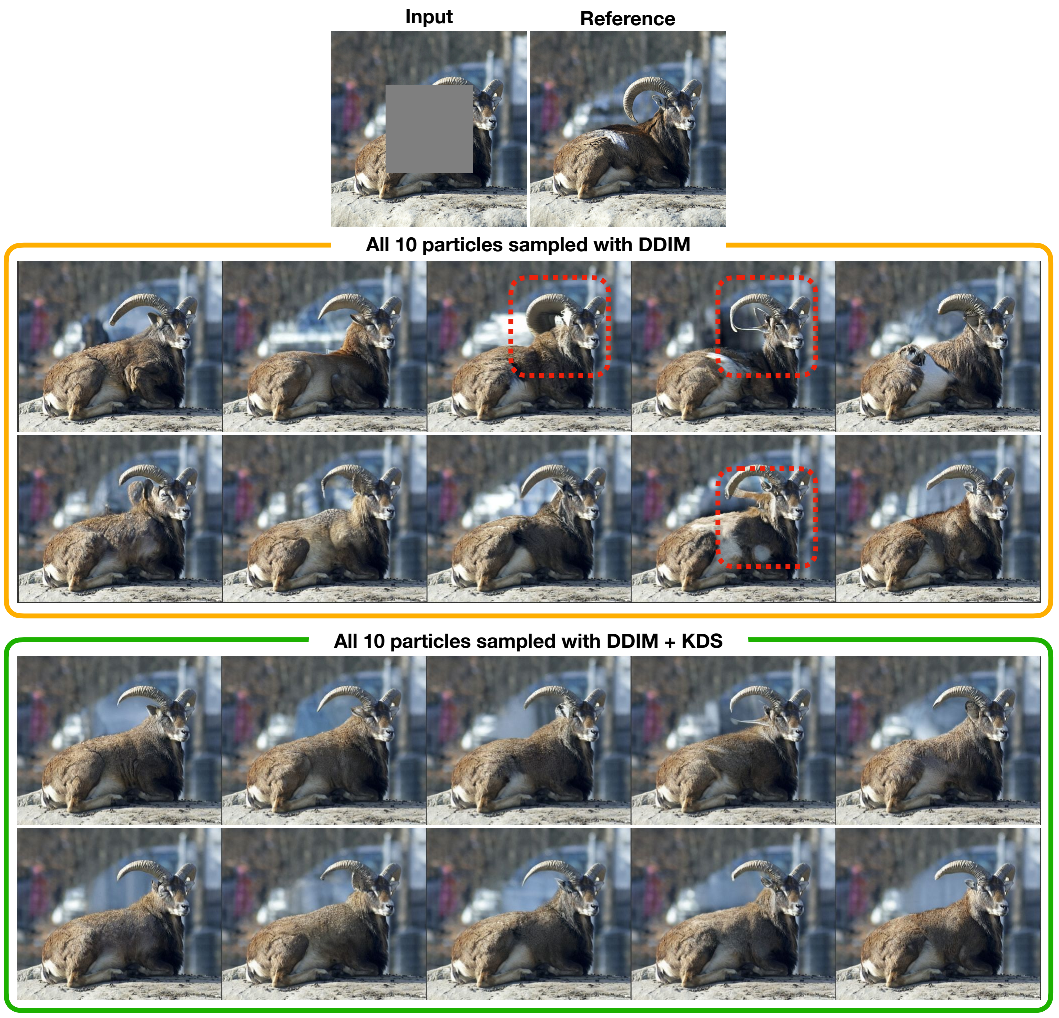
Input
Reference
All 10 particles sampled with DDIM
All 10 particles sampled with DDIM + KDS

Figure 14: Image Inpainting performance with LDM-inpainting on ImageNet dataset. Visualizes all 10 particles for DDIM vs. DDIM + KDS. Regions with artifacts were highlighted with red box.

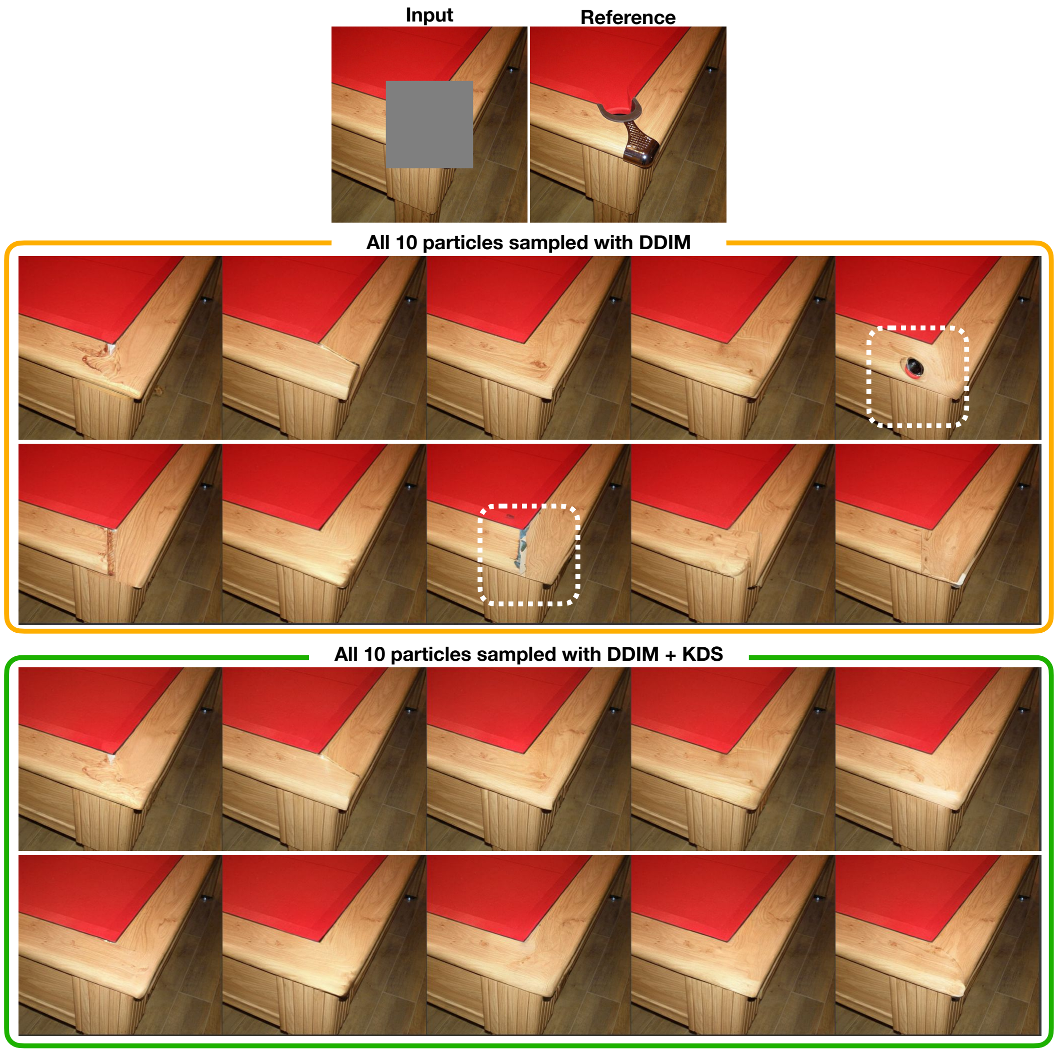
Input
Reference
All 10 particles sampled with DDIM
All 10 particles sampled with DDIM + KDS

Figure 15: Image Inpainting performance with LDM-inpainting on ImageNet dataset. Visualizes all 10 particles for DDIM vs. DDIM + KDS. Regions with artifacts were highlighted with white box.