# LD-RPS: Zero-Shot Unified Image Restoration via Latent Diffusion Recurrent Posterior Sampling

Supplementary Material

| Methods | LOLv1 | | | | |
|---|---|---|---|---|---|
| **TAO** | PSNR | SSIM | LPIPS | PI | NIQE |
| seed10 | 17.22 | 0.746 | 0.362 | 6.46 | 9.16 |
| seed20 | 15.32 | 0.761 | 0.336 | 6.30 | 8.69 |
| seed123 | 14.97 | 0.765 | 0.390 | 6.26 | 8.53 |
| Avg. | 15.84 | 0.757 | 0.363 | 6.34 | 8.79 |
| **Ours** | PSNR | SSIM | LPIPS | PI | NIQE |
| seed10 | 17.37 | 0.797 | 0.276 | 4.64 | 5.32 |
| seed20 | 17.73 | 0.807 | 0.288 | 4.98 | 5.71 |
| seed123 | 17.24 | 0.807 | 0.266 | 4.75 | 5.55 |
| Avg. | 17.45 | 0.804 | 0.277 | 4.79 | 5.52 |

Table 7. Experimental results of multi-seed randomness test on the LOLv1 dataset.

| Methods | HSTS | | | Kodak | | |
|---|---|---|---|---|---|---|
| **TAO** | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS |
| seed10 | 18.90 | 0.823 | 0.118 | 27.67 | 0.817 | 0.163 |
| seed20 | 18.07 | 0.828 | 0.150 | 28.11 | 0.821 | 0.174 |
| seed123 | 18.16 | 0.819 | 0.174 | 27.38 | 0.808 | 0.170 |
| Avg. | 18.38 | 0.823 | 0.147 | 27.72 | 0.815 | 0.169 |
| **Ours** | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS |
| seed10 | 21.60 | 0.810 | 0.179 | 28.48 | 0.839 | 0.184 |
| seed20 | 21.13 | 0.811 | 0.177 | 28.60 | 0.841 | 0.171 |
| seed123 | 21.63 | 0.817 | 0.175 | 28.83 | 0.842 | 0.170 |
| Avg. | 21.45 | 0.813 | 0.177 | 28.64 | 0.841 | 0.175 |

Table 8. Experimental results of multi-seed randomness test on the HSTS and Kodak24 dataset.

## 8. Implementation Details

### 8.1. Parameter Settings

In all experiments, LD-RPS consistently uses the pre-trained stable diffusion model [47], with sampling conducted via the DDIM scheduler [54]. The time step $T$ is set to 1000, which is divided into 450 sampling steps. The interval where $T > 700$ constitutes the first stage, during which the adapter is trained independently. The second stage is defined by a threshold of $T = 150$, where the quality function is introduced for $T < 150$. Each experiment is conducted using a single Nvidia H20 GPU.

For the weight selection among different loss functions in posterior sampling, i.e., $w_1$, $w_2$, $w_3$, $w_4$, and $w_5$ as well as the loss ratio $\lambda_1$, $\lambda_2$, $\lambda_3$ for F-PAM optimization, we observed a strong task dependency. We first select a subset of the test set for parameter tuning and apply the optimal parameters to the restoration of this type of degradation.

Considering the potential impact of initial sampling noise on text-to-image models, which introduces a certain degree of randomness, we average the results obtained from three different random seeds to ensure consistency in the experiments (three seeds are randomly selected, and the same seeds are used across different methods). To enable comparison across various methods, a $256 \times 256$ patch is cropped from the center of each image in the test set. For commonly used parameters in stable diffusion, we selected the following set: both the text-to-image pipeline and the subsequent iterative image-to-image pipeline are set to 450 timesteps. The resampling intensity for iterative posterior sampling is set to $\gamma = 0.5$, based on relevant descriptions in SDEdit.

### 8.2. Thresholding Strategy

Additionally, for the post-processing of diffusion iterations, we employ a thresholding strategy. This part of the code is integrated within the diffusers library, which we have modified to adapt to latent diffusion.

**Dynamic thresholding:** At each sampling step, we set $s$ to a specific percentile of the absolute values in $\hat{\mathbf{z}}_0$ (the prediction of $\mathbf{z}_0$ at timestep $t$). If $s < 1$, we sort the values of $\hat{\mathbf{z}}_0$ by their absolute magnitude and select the value at the $s$ percentile, denoted as $k$, and we threshold $\hat{\mathbf{z}}_0$ to fall within the range $[-k, k]$. This approach filters out outlier values that are far from the data distribution at each diffusion step and pushes the edge values closer to the center. In our experiments, we consistently set $s = 0.995$. Previous research [49] has indicated that dynamic thresholding significantly enhances photorealism and improves image-text alignment, particularly when employing very large guidance weights. The specific implementation of this algorithm is as follows:

$$\hat{\mathbf{z}}_0 = r(\hat{\mathbf{z}}_0, B, C \times \prod_i D_i), \mathbf{a} = |\hat{\mathbf{z}}_0| \tag{12}$$

$$\mathbf{k} = q(\mathbf{a}, s, \text{dim} = 1) \tag{13}$$

$$\hat{\mathbf{z}}_0 = c(\hat{\mathbf{z}}_0, -k, k) \tag{14}$$

$$\hat{\mathbf{z}}_0 = r(\hat{\mathbf{z}}_0, B, C, D_1, D_2, \ldots, D_n) \tag{15}$$

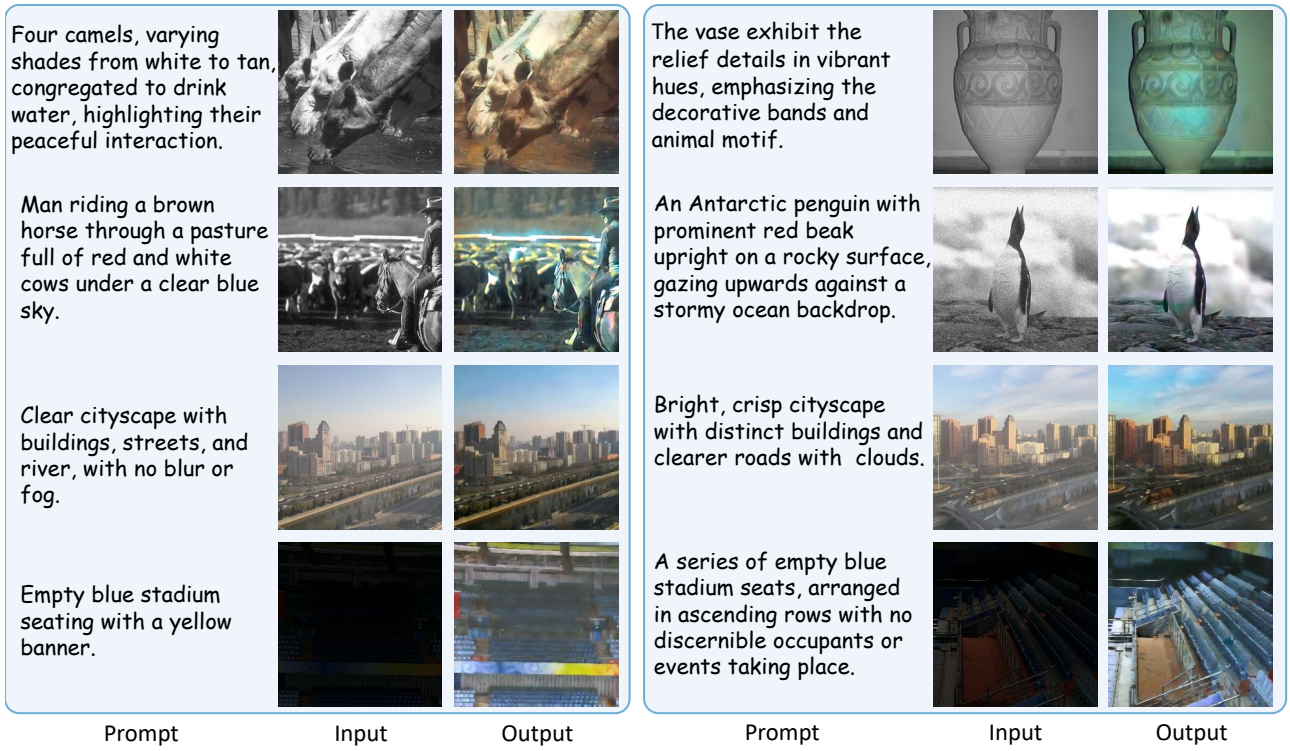


Figure 9. Some supplementary results of LD-RPS: we provide the input and output under different tasks, along with the textual prompts used for guidance.

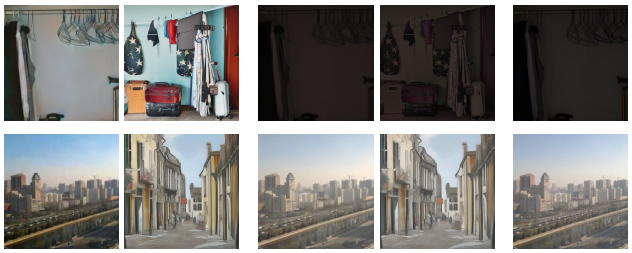


Figure 10. Visualization results illustrating the inputs and outputs of F-PAM at $T = 0$, accompanied by the corresponding conditional image $\boldsymbol{y}$.

In this implementation, $r(.)$ represents the reshape operation, $q(.)$ denotes sorting the data along a specific dimension, and $c(.)$ stands for the torch.clamp function. We assume a batch of $B$ images, each with $C$ channels, and each channel has dimensions $D1, D2, \ldots, Dn$. First, we flatten each image into a one-dimensional array, then perform the dynamic thresholding operation, and finally restore the original dimensions.

### 8.3. Dataset Settings

For the selection of comparison methods, we classify them into three categories: unified supervised methods, task-specific unsupervised methods, and zero-shot methods using posterior sampling. In the comparison of unified supervised methods, we employ AirNet [28], PromptIR [45], and DiffUIR [65]. For the evaluation of zero-shot methods, we utilize GDP [11] and TAO [14].

## 9. Supplementary Experimental Results

Due to the observed stochastic nature of the generated results, in the main text, we report the statistical values for our method and TAO [14] over three different random seeds. Detailed experimental data are presented in the Tab. 7 and Tab. 8.

We observe that in the early stages of image generation, our model exhibits strong hallucinations. As mentioned in the main text, this is due to the dataset of the pre-trained model not fully matching the target of the zero-shot generation. Consequently, the model tends to generate more frequently occurring content from the training dataset, such as faces and animals. To address this, we employ a recurrent posterior sampling strategy to optimize the initial point of the model. As shown in Fig. 11, without iterations, the model struggles to generate the desired content even at $T = 450$. This condition improves as the number of iterations increases.

Additionally, we provide some other visual comparison

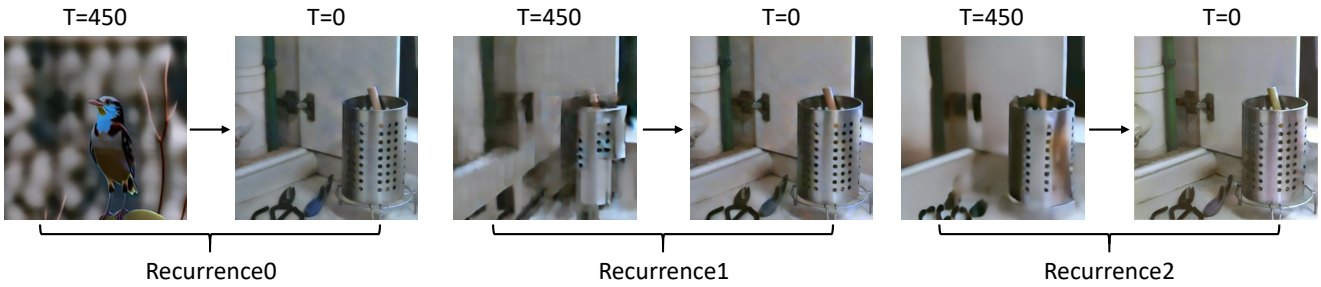


Figure 11. The optimization states of LD-RPS at different time steps during various recurrences. The first recurrence tends to produce some artifacts. By utilizing the initial restoration results for initialization, the influence of these artifacts is reduced.


Figure 12. Supplementary results of visual comparison experiment on the LOLv1 dataset.

results. Fig. 12 and Fig. 13 are supplementary results on the LOL dataset, Fig. 14 and Fig. 15 are supplementary results on the HSTS dataset, and Fig. 16 presents supplementary results on the Kodak dataset.


Figure 13. Supplementary results of visual comparison experiment on the LOLv1 dataset.

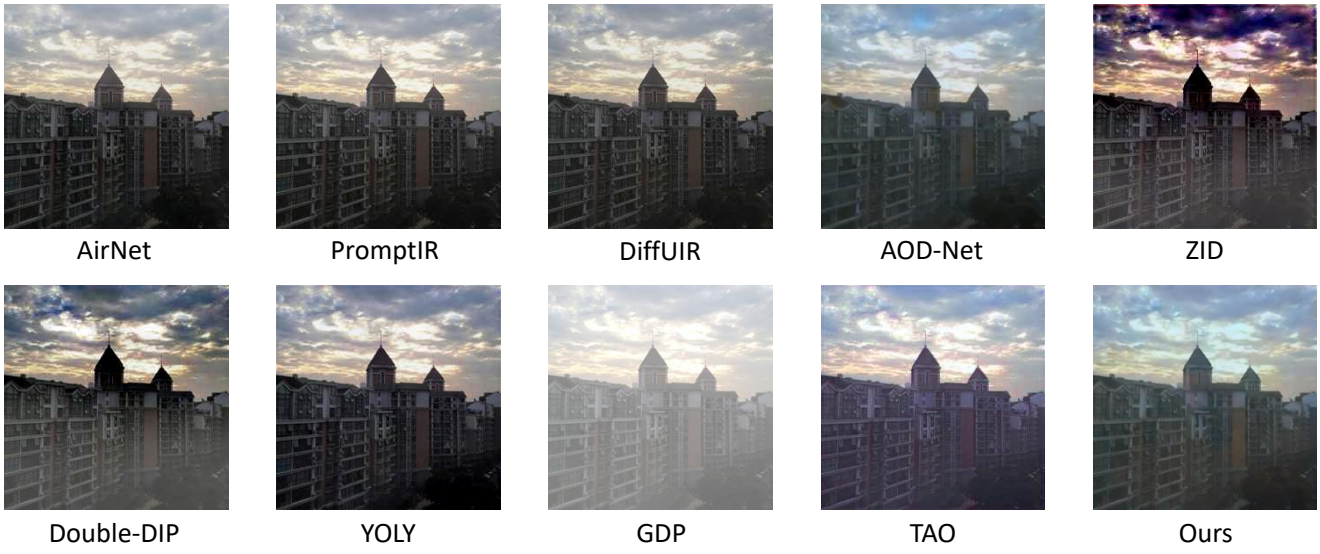


Figure 14. Supplementary results of visual comparison experiment on the HSTS dataset.

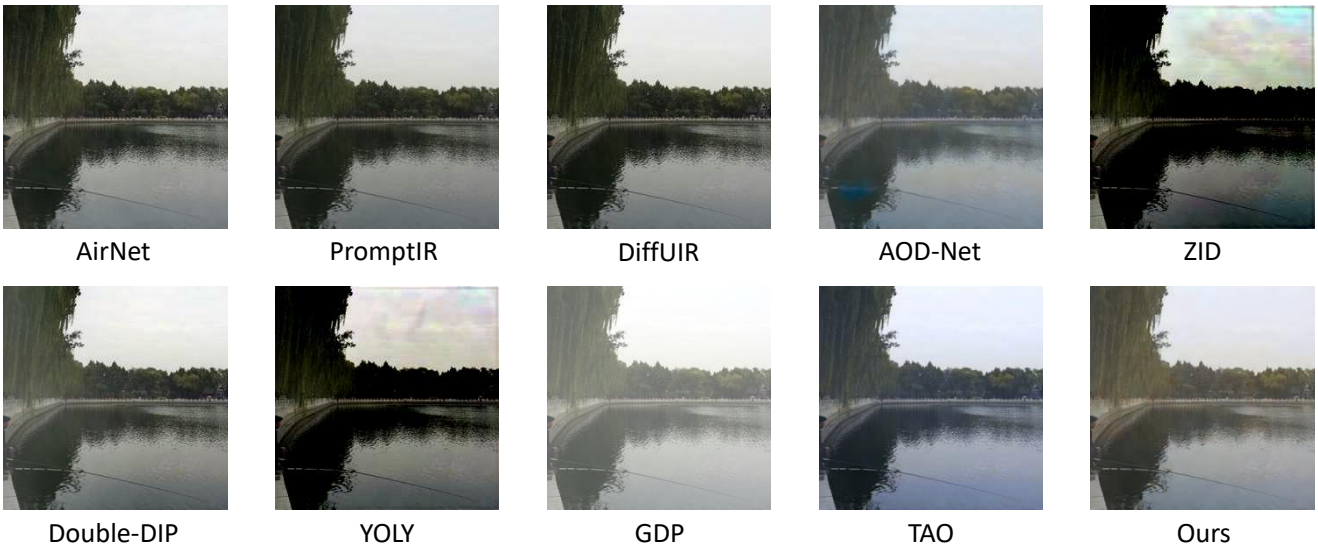


Figure 15. Supplementary results of visual comparison experiment on the HSTS dataset.

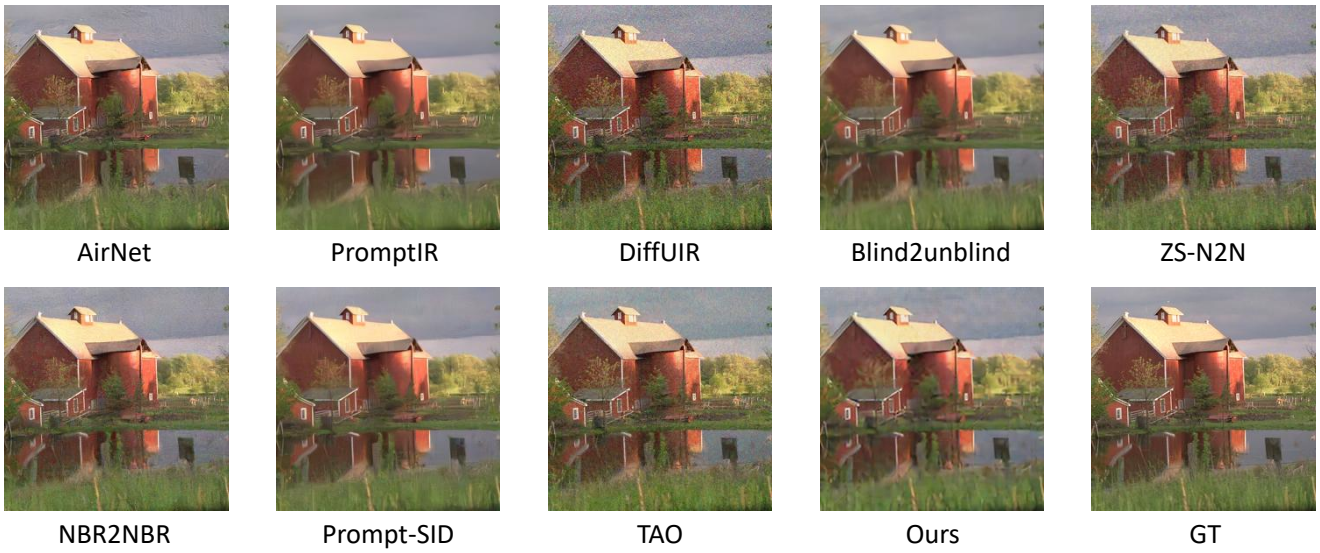


Figure 16. Supplementary results of visual comparison experiment on the Kodak24 dataset.