## Appendix

## A. Derivations and Proofs

### A.1. Perturbed Generation Process

For the reverse generation process from $I_t$ to $I_{t-1}$, we can represent the transfer probabilities $q(I_{t-1}|I_t, I_0, I_{res})$ by Bayes' rule:

$$q(I_{t-1}|I_t, I_0, I_{res}) = q(I_t|I_{t-1}, I_0, I_{res})\frac{q(I_{t-1}|I_0, I_{res})}{q(I_t|I_0, I_{res})}, \tag{22}$$

where $q(I_{t-1}|I_0, I_{res}) = \mathcal{N}(I_{t-1}; I_0 + \bar{\alpha}_{t-1}I_{res}, \bar{\beta}_{t-1}^2\mathbf{I})$ from Eq. 7, and $q(I_t|I_{t-1}, I_0, I_{res}) = q(I_t|I_{t-1}, I_{res})$[8]$= \mathcal{N}(I_t; I_{t-1} + \alpha_t I_{res}, \beta_t^2\mathbf{I})$ from Eq. 9. Thus, we have (considering only the exponential term)

$$\begin{aligned} & q(I_{t-1}|I_t, I_0, I_{res}) = \mathcal{N}(I_{t-1}; \mu_t(x_t, I_0, I_{res}), \Sigma_t(x_t, I_0, I_{res})\mathbf{I}) \\ & \propto exp\left(-\frac{1}{2}(\frac{(I_t - I_{t-1} - \alpha_t I_{res})^2}{\beta_t^2} + \frac{(I_{t-1} - I_0 - \bar{\alpha}_{t-1}I_{res})^2}{\bar{\beta}_{t-1}^2}) - \frac{(I_t - I_0 - \bar{\alpha}_t I_{res})^2}{\bar{\beta}_t^2})\right) \\ & = exp\left(-\frac{1}{2}((\frac{\bar{\beta}_t^2}{\beta_t^2\bar{\beta}_{t-1}^2})I_{t-1}^2 - 2(\frac{I_t - \alpha_t I_{res}}{\beta_t^2} + \frac{\bar{\alpha}_{t-1}I_{res} + I_0}{\bar{\beta}_{t-1}^2})I_{t-1} + C(I_t, I_0, I_{res}))\right), \end{aligned} \tag{23}$$

where the $C(I_t, I_0, I_{res})$ term is not related to $I_{t-1}$. From Eq. 23, $\mu_t(x_t, I_0, I_{res})$ and $\Sigma_t(x_t, I_0, I_{res})$ are represented as follows,

$$\mu_t(x_t, I_0, I_{res}) = (\frac{I_t - \alpha_t I_{res}}{\beta_t^2} + \frac{\bar{\alpha}_{t-1}I_{res} + I_0}{\bar{\beta}_{t-1}^2})/\frac{\bar{\beta}_t^2}{\beta_t^2\bar{\beta}_{t-1}^2} \tag{24}$$

$$= \frac{\bar{\beta}_{t-1}^2}{\bar{\beta}_t^2}I_t + \frac{\beta_t^2\bar{\alpha}_{t-1} - \bar{\beta}_{t-1}^2\alpha_t}{\bar{\beta}_t^2}I_{res} + \frac{\beta_t^2}{\bar{\beta}_t^2}I_0 \tag{25}$$

$$= I_t - \alpha_t I_{res} - \frac{\beta_t^2}{\bar{\beta}_t}\epsilon, \tag{26}$$

$$\Sigma_t(x_t, I_0, I_{res}) = \frac{\beta_t^2\bar{\beta}_{t-1}^2}{\bar{\beta}_t^2}. \tag{27}$$

Eq. 7 is used for the derivation from Eq. 25 to Eq. 26. Then, we define the generation process to start from $p_\theta(I_T) \sim \mathcal{N}(I_T; \mathbf{0}, \mathbf{I})$,

$$p_\theta(I_{t-1}|I_t) = q(I_{t-1}|I_t, I_0^\theta, I_{res}^\theta), \tag{28}$$

where $I_0^\theta = I_t - \bar{\alpha}_t I_{res}^\theta - \bar{\beta}_t\epsilon_\theta$ by Eq. 7. Here we only consider $L_{t-1}$ in [17],

$$L_{t-1} = D_{KL}(q(I_{t-1}|I_t, I_0, I_{res})||p_\theta(I_{t-1}|I_t)) \tag{29}$$

$$= \mathbb{E}\left[\left\|I_t - \alpha_t I_{res} - \frac{\beta_t^2}{\bar{\beta}_t}\epsilon - (I_t - \alpha_t I_{res}^\theta - \frac{\beta_t^2}{\bar{\beta}_t}\epsilon_\theta)\right\|^2\right], \tag{30}$$

where $D_{KL}$ denotes KL divergence. The noise $\epsilon$ can be represented as a transformation of $I_{res}$ using Eq. 16, and then Eq. 30 simplifies to:

$$L_{res}(\theta) := \mathbb{E}\left[\lambda_{res}\left\|I_{res} - I_{res}^\theta(I_t, t, I_{in})\right\|^2\right] \tag{31}$$

In addition, the residuals $I_{res}$ can be represented as a transformation of $\epsilon$ using Eq. 16, and then Eq. 30 simplifies to:

$$L_\epsilon(\theta) := \mathbb{E}\left[\lambda_\epsilon\left\|\epsilon - \epsilon_\theta(I_t, t, I_{in})\right\|^2\right], \tag{32}$$

We reset the weights, i.e., $\lambda_{res}, \lambda_\epsilon \in \{0, 1\}$, similar to DDPM [17] discarding weights (or regarding them as reweighting) in simplified training objective.

[8]Each step in Eq. 7 adds a new random Gaussian noise in the random forward diffusion. Thus for simplicity, we assume $q(I_t|I_{t-1}, I_0, I_{res}) = q(I_t|I_{t-1}, I_{res})$, it follows that $I_0$ is not important for $I_t$ when $I_{t-1}$ presents as a condition.

## A.2. Deterministic Implicit Sampling

If $q_\sigma(I_{t-1}|I_t, I_0, I_{res})$ is defined in Eq. 11, we have:

$$q(I_t|I_0, I_{res}) = \mathcal{N}(I_t; I_0 + \bar{\alpha}_t I_{res}, \bar{\beta}_t^2 \mathbf{I}). \tag{33}$$

*Proof.* Similar to the evolution from DDPM [17] to DDIM [51], we can prove the statement with an induction argument for $t$ from $T$ to 1. Assuming that Eq. 33 holds at $T$, we just need to verify $q(I_{t-1}|I_0, I_{res}) = \mathcal{N}(I_{t-1}; I_0 + \bar{\alpha}_{t-1} I_{res}, \bar{\beta}_{t-1}^2 \mathbf{I})$ at $t-1$ from $q(I_t|I_0, I_{res})$ at $t$ using Eq. 33. Given:

$$q(I_t|I_0, I_{res}) = \mathcal{N}(I_t; I_0 + \bar{\alpha}_t I_{res}, \bar{\beta}_t^2 \mathbf{I}), \tag{34}$$

$$q_\sigma(I_{t-1}|I_t, I_0, I_{res}) = \mathcal{N}(I_{t-1}; I_0 + \bar{\alpha}_{t-1} I_{res} + \sqrt{\bar{\beta}_{t-1}^2 - \sigma_t^2}\frac{I_t - (I_0 + \bar{\alpha}_t I_{res})}{\bar{\beta}_t}, \sigma_t^2 \mathbf{I}), \tag{35}$$

$$q(I_{t-1}|I_0, I_{res}) := \mathcal{N}(\tilde{\mu}_{t-1}, \tilde{\Sigma}_{t-1}) \tag{36}$$

Similar to obtaining $p(y)$ from $p(x)$ and $p(y|x)$ using Eq.2.113-Eq.2.115 in [5], the values of $\tilde{\mu}_{t-1}$ and $\tilde{\Sigma}_{t-1}$ are derived as following:

$$\tilde{\mu}_{t-1} = I_0 + \bar{\alpha}_{t-1} I_{res} + \sqrt{\bar{\beta}_{t-1}^2 - \sigma_t^2}\frac{(I_0 + \bar{\alpha}_t I_{res}) - (I_0 + \bar{\alpha}_t I_{res})}{\bar{\beta}_t} = I_0 + \bar{\alpha}_{t-1} I_{res}, \tag{37}$$

$$\tilde{\Sigma}_{t-1} = \sigma_t^2 \mathbf{I} + (\frac{\sqrt{\bar{\beta}_{t-1}^2 - \sigma_t^2}}{\bar{\beta}_t})^2 \bar{\beta}_t^2 \mathbf{I} = \bar{\beta}_{t-1}^2 \mathbf{I}. \tag{38}$$

Therefore, $q(I_{t-1}|I_0, I_{res}) = \mathcal{N}(I_{t-1}; I_0 + \bar{\alpha}_{t-1} I_{res}, \bar{\beta}_{t-1}^2 \mathbf{I})$. In fact, the case ($t = T$) already holds, thus Eq. 33 holds for all $t$.

**Simplifying Eq. 11.** Eq. 11 can also be written as:

$$\begin{aligned}
I_{t-1} &= I_0 + \bar{\alpha}_{t-1} I_{res} + \sqrt{\bar{\beta}_{t-1}^2 - \sigma_t^2}\frac{I_t - (I_0 + \bar{\alpha}_t I_{res})}{\bar{\beta}_t} + \sigma_t \epsilon_t, && (39)\\
&= \frac{\sqrt{\bar{\beta}_{t-1}^2 - \sigma_t^2}}{\bar{\beta}_t} I_t + (1 - \frac{\sqrt{\bar{\beta}_{t-1}^2 - \sigma_t^2}}{\bar{\beta}_t}) I_0 + (\bar{\alpha}_{t-1} - \frac{\sqrt{\bar{\alpha}_t \bar{\beta}_{t-1}^2 - \sigma_t^2}}{\bar{\beta}_t}) I_{res} + \sigma_t \epsilon_t && (40)\\
&= I_t - (\bar{\alpha}_t - \bar{\alpha}_{t-1}) I_{res} - (\bar{\beta}_t - \sqrt{\bar{\beta}_{t-1}^2 - \sigma_t^2})\epsilon + \sigma_t \epsilon_t, && (41)
\end{aligned}$$

where $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Eq. 41 is consistent with Eq. 12, and Eq. 7 is used for the derivation from Eq. 40 to Eq. 41.

## A.3. Coefficient Transformation

For image generation, $I_{in} = 0$, thus Eq. 16 can also be written as:

$$I_t = (\bar{\alpha}_t - 1) I_{res} + \bar{\beta}_t \epsilon \tag{42}$$

$$I_{res} = \frac{I_t - \bar{\beta}_t \epsilon}{\bar{\alpha}_t - 1}. \tag{43}$$

If the residuals $I_{res}^\theta$ are represented as a transformation of $\epsilon_\theta$ using Eq. 43, Eq. 12 is simplified to

$$\begin{aligned}
I_{t-1} &= I_t - (\bar{\alpha}_t - \bar{\alpha}_{t-1}) I_{res}^\theta - (\bar{\beta}_t - \sqrt{\bar{\beta}_{t-1}^2 - \sigma_t^2})\epsilon_\theta + \sigma_t \epsilon_t && (44)\\
&= I_t - (\bar{\alpha}_t - \bar{\alpha}_{t-1})\frac{I_t - \bar{\beta}_t \epsilon_\theta}{\bar{\alpha}_t - 1} - (\bar{\beta}_t - \sqrt{\bar{\beta}_{t-1}^2 - \sigma_t^2})\epsilon_\theta + \sigma_t \epsilon_t && (45)\\
&= \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} I_t - (\frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t}\bar{\beta}_t - \sqrt{\bar{\beta}_{t-1}^2 - \sigma_t^2})\epsilon_\theta + \sigma_t \epsilon_t && (46)\\
&= \frac{\sqrt{\bar{\alpha}_{DDIM}^{t-1}}}{\sqrt{\bar{\alpha}_{DDIM}^{t}}} I_t - (\frac{\sqrt{\bar{\alpha}_{DDIM}^{t-1}}\sqrt{1 - \bar{\alpha}_{DDIM}^{t}}}{\sqrt{\bar{\alpha}_{DDIM}^{t}}} - \sqrt{1 - \bar{\alpha}_{DDIM}^{t-1} - \sigma_t^2})\epsilon_\theta + \sigma_t \epsilon_t && (47)\\
&= \sqrt{\bar{\alpha}_{DDIM}^{t-1}}\left(\frac{I_t - \sqrt{1 - \bar{\alpha}_{DDIM}^{t}}\epsilon_\theta}{\sqrt{\bar{\alpha}_{DDIM}^{t}}}\right) + \sqrt{1 - \bar{\alpha}_{DDIM}^{t-1} - \sigma_t^2}\epsilon_\theta + \sigma_t \epsilon_t. && (48)
\end{aligned}$$

**Algorithm 1:** Coefficient initialization, transformation, and adjustment.

```
Input : The initial conditions $\bar{\alpha}_T = 1$, $\bar{\beta}^2_T > 0$, $T = 1000$, and $t \in \{1, 2, \dots T\}$. The hyperparameter $\eta = 1$ for the
        random generation process and $\eta = 0$ for deterministic implicit sampling. Variance modes have "DDIM"
        and "DDIM→RDDM". The coefficient adjustment mode Adjust="Alpha", "Beta", or "Alpha+Beta".
Output: The adjusted coefficients $\bar{\alpha}^*_t$, $\bar{\beta}^*_t$ and $\sigma^*_t$.
   // (a) Coefficient initialization of DDIM [51]
1  $\beta^t_{DDIM} \leftarrow$ Linspace $(0.0001, 0.02, T)$                          ▷ linear schedule [51]
2  $\alpha^t_{DDIM} \leftarrow 1 - \beta^t_{DDIM}$
3  $\bar{\alpha}^t_{DDIM} \leftarrow$ Cumprod $(\alpha^t_{DDIM})$                            ▷ cumulative multiplication
4  $\sigma^2_t(DDIM) \leftarrow \eta \frac{(1-\bar{\alpha}^{t-1}_{DDIM})}{1-\bar{\alpha}^t_{DDIM}}(1 - \frac{\bar{\alpha}^t_{DDIM}}{\bar{\alpha}^{t-1}_{DDIM}})$
   // (b) Coefficient transformation from DDIM [51] to our RDDM
5  $\bar{\alpha}_t \leftarrow 1 - \sqrt{\bar{\alpha}^t_{DDIM}}$                                     ▷ Eq. 17
6  $\bar{\beta}_t \leftarrow \sqrt{1 - \bar{\alpha}^t_{DDIM}}$                                     ▷ Eq. 17
7  $\sigma^2_t(RDDM) \leftarrow \eta \frac{(\bar{\beta}^2_t - \bar{\beta}^2_{t-1})\bar{\beta}^2_{t-1}}{\bar{\beta}^2_t}$
   // (c) Select variance schedule
8  if Variance=="DDIM" then
9      $\sigma^*_t \leftarrow \sqrt{\sigma^2_t(DDIM)}$          ▷ consistent sampling process with DDIM [51] and DDPM [17]
10 else if Variance=="DDIM→RDDM" then
11     $\sigma^*_t \leftarrow \sqrt{\sigma^2_t(RDDM)}$                                ▷ sum-constrained variance schedule
12 end
   // (d) Coefficient adjustment
13 $\alpha_t \leftarrow$Power $(1 - t/T, 1)$                                  ▷ linearly decreasing by Eq. 18
14 $\beta^2_t \leftarrow$Power $(t/T, 1) \cdot \bar{\beta}^2_T$                     ▷ control the noise intensity in $I_T$ by $\bar{\beta}^2_T$
15 if Adjust=="Alpha" then
16     $\bar{\alpha}^*_t \leftarrow$Cumsum $(\alpha_t)$, $\bar{\beta}^*_t \leftarrow \bar{\beta}_t$                                ▷ cumulative sum
17 else if Adjust=="Beta" then
18     $\bar{\alpha}^*_t \leftarrow \bar{\alpha}_t$, $\bar{\beta}^*_t \leftarrow \sqrt{\text{Cumsum}(\beta^2_t)}$                          ▷ cumulative sum
19 else if Adjust=="Alpha+Beta" then
20     $\bar{\alpha}^*_t \leftarrow$Cumsum $(\alpha_t)$, $\bar{\beta}^*_t \leftarrow \sqrt{\text{Cumsum}(\beta^2_t)}$           ▷ coefficient reinitialization
21 else
22     $\bar{\alpha}^*_t \leftarrow \bar{\alpha}_t$, $\bar{\beta}^*_t \leftarrow \bar{\beta}_t$
23 end
24 return $\bar{\alpha}^*_t$, $\bar{\beta}^*_t$, $\sigma^*_t$                         ▷ sampling with adjusted coefficients by Eq. 12
```

Eq. 48 is consistent with Eq.12 in DDIM [51] by replacing $\sigma^2_t$ with $\sigma^2_t(DDIM)$, and Eq. 17 is used for the derivation from Eq. 46 to Eq. 47. Thus, our sampling process is consistent with that of DDPM [51] and DDIM [17] by transforming coefficient/variance schedules.

We present the pipeline of coefficient transformation in Algorithm 1. Fig. 4 (b)(c) show the result of coefficient transformation. In Eq. 17, in addition to the coefficient $\bar{\alpha}_t, \bar{\beta}^2_t$ being replaced by $\bar{\alpha}^t_{DDIM}$, the variance $\sigma^2_t$ is also replaced with $\sigma^2_t(DDIM)$ to be consistent with DDIM [51] ($\eta = 0$) and DDPM [17] ($\eta = 1$). In fact, for DDIM [51] ($\eta = 0$), the variance is equal to 0 and does not need to be converted. Therefore, we analyze the difference between the variance of our RDDM and the variance of DDPM [17] in Appendix A.4.

## A.4. Perturbed Generation Process with Sum-constrained Variance

From Eq. 12, the variance of our RDDM ($\eta = 1$) is

$$\sigma^2_t(RDDM) = \eta \frac{\beta^2_t \bar{\beta}^2_{t-1}}{\bar{\beta}^2_t}. \tag{49}$$

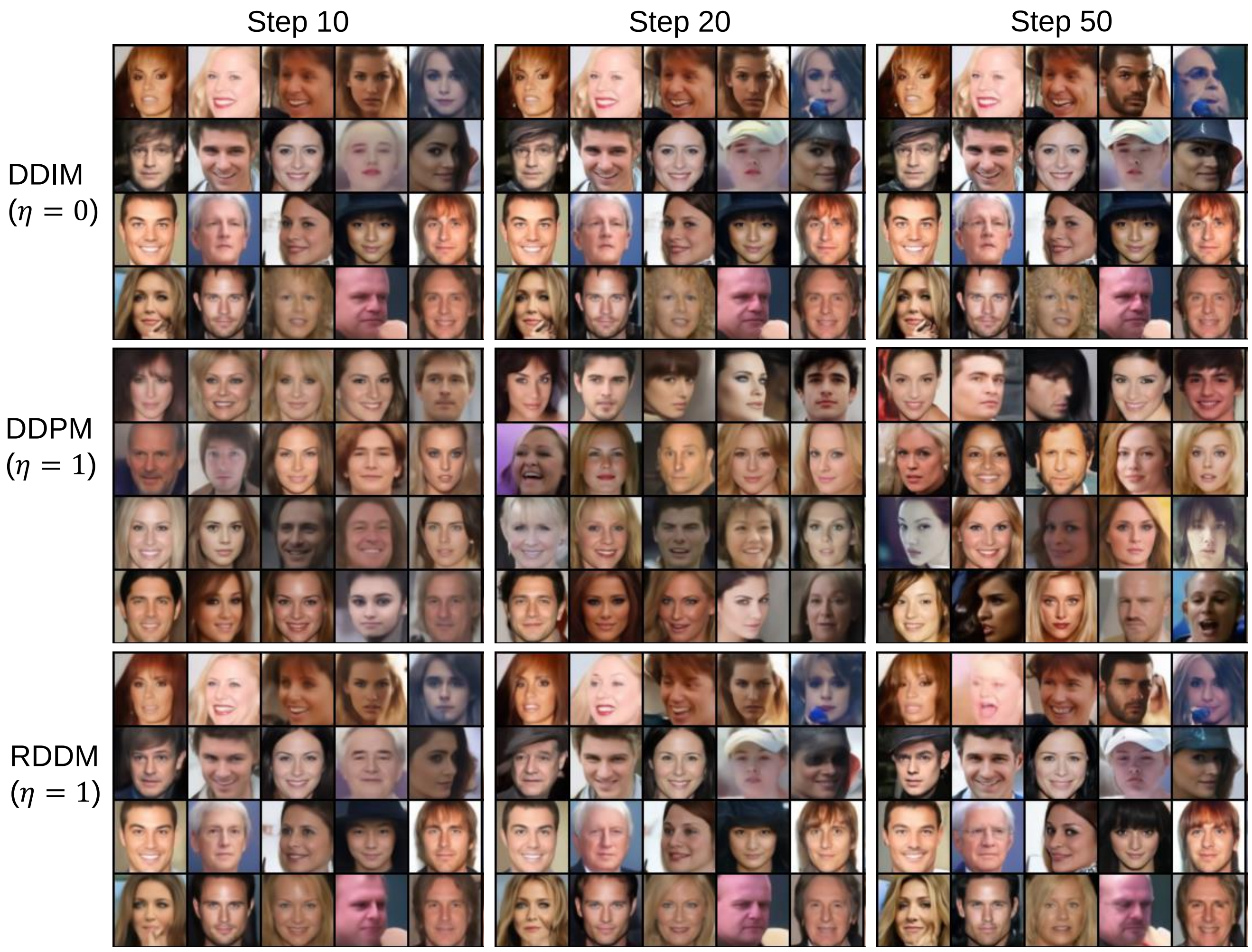

Figure 8. Perturbed generation process with sum-constrained variance on the CelebA 64×64 dataset [36]. When the coefficient schedules $(\bar{\alpha}_t, \bar{\beta}_t)$ are consistent, RDDM ($\eta = 0$) is the same as DDIM ($\eta = 0$), but RDDM ($\eta = 1$) is different from DDPM ($\eta = 1$) due to different variance schedules ($\sigma_t^2$). At different sampling steps, our RDDM has the same total noise, while DDPM has a different total noise. Notably, all the results in Fig. 8 can be generated from the same pre-trained model via variance transformation in Appendix A.3. In other words, the RDDM provides a sum-constrained variance strategy, which can be used directly in the pre-trained DDPM without re-training the model.

We replace $\bar{\beta}_t^2$ by $\bar{\alpha}_{DDIM}^t$ using Eq. 17 and replace $\beta_t^2$ by $\bar{\beta}_t^2 - \bar{\beta}_{t-1}^2$,

$$\sigma_t^2(RDDM) = \eta \bar{\alpha}_{DDIM}^{t-1} \frac{(1 - \bar{\alpha}_{DDIM}^{t-1})}{1 - \bar{\alpha}_{DDIM}^t} (1 - \frac{\bar{\alpha}_{DDIM}^t}{\bar{\alpha}_{DDIM}^{t-1}}), \tag{50}$$

while the variance of DDPM [17] ($\eta = 1$) is

$$\sigma_t^2(DDIM) = \eta \frac{(1 - \bar{\alpha}_{DDIM}^{t-1})}{1 - \bar{\alpha}_{DDIM}^t} (1 - \frac{\bar{\alpha}_{DDIM}^t}{\bar{\alpha}_{DDIM}^{t-1}}). \tag{51}$$

Our variance is much smaller than the variance of DDPM [17] because $0 < \bar{\alpha}_{DDIM}^{t-1} < \alpha_{DDIM}^1 < 1$ (e.g., $\alpha_{DDIM}^1 = 0.02$ in linear schedule [51]). Compared to $\sigma_t^2(DDIM) \approx 1$ [53], the variance of our RDDM is sum-constrained,

$$\sum_{i=1}^{T} \sigma_t^2(RDDM) = \sum_{i=1}^{T} \eta \beta_t^2 \frac{\bar{\beta}_{t-1}^2}{\bar{\beta}_t^2} \leq \sum_{i=1}^{T} \beta_t^2 \leq 1, \tag{52}$$

where $\sum_{i=1}^{T}\beta_t^2 = \bar{\beta}_T^2 = 1$ for image generation. This is also consistent with the directional residual diffusion process with perturbation defined in Eq. 7. A qualitative comparison of our RDDM ($\eta = 1$) with DDIM [51] ($\eta = 0$) and DDPM [17] ($\eta = 1$) is shown in Fig. 8. Notably, for $\eta = 0$, our RDDM is consistent with DDIM [51] (in Fig. 5(a)(b)).

## A.5. Comparison With Other Methods

The main difference is that to adapt the denoising diffusion, score, flow, or Schrödinger's bridge to image restoration, they choose the noise (Shadow Diffusion [14], SR3 [49], and WeatherDiffusion [82]), the residual (DvSR [62] and Rectified Flow [35]), the target image (ColdDiffusion [2] and InDI [11]), **or** its linear transformation term (I2SB [29]), which is similar to a special case of our RDDM when it only predicts noise (SM-N) or residuals (SM-Res), while we introduce residual estimation but also embrace noise both for generation and restoration (SM-Res-N). We highlight that residuals and noise are equally important, e.g., the residual prioritizes certainty while the noise emphasizes diversity.

**Differences from DDPM [17].** 1) DDPM is not interpretable for image restoration, while our RDDM is a unified, interpretable diffusion model for both image generation and restoration. 2) Differences in the definition of the forward process lead to different variance strategies. Our RDDM has sum-constrained variance (much smaller than the variance of DDPM), while DDPM has preserving variance [53] (see Appendix A.4).

**Differences from IDDPM [44].** In the original DDPM [17], for the transfer probabilities $p_\theta(I_{t-1}|I_t)$ in Eq. 5, the mean $\mu_\theta(I_t, t)$ is learnable, while the variance $\Sigma_t$ is fixed. IDDPM [44] highlights the importance of **estimating both the mean and variance**, demonstrating that learning variances allow for fewer sampling steps with negligible differences in sample quality. However, IDDPM [44] still only involves denoising procedures, and crucially, IDDPM, like DDPM, is thus not interpretable for image restoration. In addition, IDDPM [44] proposes three alternative ways to parameterize $\mu_\theta(I_t, t)$, i.e., predict mean $\mu_\theta(I_t, t)$ directly with a neural network, predict noise $\epsilon$, or predict clean image $I_0$. IDDPM [44] does not predict the clean image $I_0$ and noise $\epsilon$ at the same time, while both the residuals and the noise are predicted for our SM-Res-N.

The **essential difference** is that, our RDDM contains a mixture of three terms (i.e., input images $I_{in}$, target images $I_0$, and noise $\epsilon$) beyond DDPM/IDDPM (a mixture of two terms, i.e, $I_0$ and $\epsilon$). **We emphasize that residuals and noise are equally important**: the residual prioritizes certainty, while the noise emphasizes diversity. Furthermore, our RDDM preserves the original DDPM generation framework by coefficient/variance transformation (Eq. 17), enabling seamless transfer of improvement techniques from DDPM, such as variance optimization from IDDPM.

**Differences from ColdDiffusion [2].** 1) ColdDiffusion aims to remove the random noise entirely from the diffusion model, and replace it with other transforms (e.g., blur, masking), while our RDDM still embraces noise diffusion. Notably, we argue that noise is necessary for generative tasks that emphasize diversity (see Table 1). In fact, since ColdDiffusion discards random noise, extra noise injection is required to improve generation diversity. 2) To simulate the degradation process for different restoration tasks, ColdDiffusion attempts to use a Gaussian blur operation for deblurring, a snowification transform for snow removal., etc. These explorations may lose generality and differ fundamentally from our residual learning. RDDM represents directional diffusion from target images to input images using residuals, without designing specific degradation operators for each task. Additionally, RDDM provides solid theoretical derivation, while ColdDiffusion lacks theoretical justification.

**Differences from DvSR [62].** [62] indeed use residual. But they 1) predict the initial clean image from a blurring image via a traditional (non-diffusion) network, calculate the residuals between the ground truth of the clean image and the predicted clean image 2) use denoising-based diffusion models predict noise like DDPM [17] and use a linear transformation of the noise to represent the residuals. They treat the residual predictions as an image generation task, aiming to produce diverse and plausible outputs based on the initial predicted clean image. Beyond simply building a diffusion model on top of residuals, we redefine a new forward process that allows simultaneous diffusion of residuals and noise, wherein the target image progressively diffuses into a purely noise or a noise-carrying input image.

**Differences from InDI [11] and I2SB [29].** We can conclude that the forward diffusion of InDI, I2SB, and our RDDM is consistent in the form of a mixture of three terms (i.e., input images $I_{in}$, target images $I_0$, and noise $\epsilon$) beyond the denoising-based diffusion (a mixture of two terms, i.e, $I_0$ and $\epsilon$). Substituting $I_{res} = I_{in} - I_0$ into Eq. 16 results in $I_t = \bar{\alpha}_t I_{in} + (1 - \bar{\alpha}_t) I_0 + \bar{\beta}_t \epsilon$. This resulted $I_t$ has the same format as Eq.8 in InD ($x_t = ty + (1 - t)x + \sqrt{t}\epsilon_t \eta_t$), and is the same format as Eq.11 in I2SB. Similar to Eq. 17 (from our RDDM to DDPM/DDIM), transforming coefficients leads to complete consistency. However, our RDDM can further extend DDPM/DDIM, InD, and I2SB to independent double diffusion processes, and holds the potential to pave the way for the multi-dimensional diffusion process. From the initial stages of constructing a new forward process, our RDDM uses independent coefficient schedules to control the diffusion of residuals and noise. This provides a more general, flexible, and scalable framework, and inspires our partially path-independent generation process, demonstrated in Fig. 6 and Fig. 16(b-f) with stable generation across various diffusion rates

| Tasks | Image Restoration | | | | Image | Image | Image |
|---|---|---|---|---|---|---|---|
| | Shadow Removal | Low-light | Deblurring | Deraining | Generation | Inpainting | Translation |
| Datasets | ISTD | LOL | GoPro | RainDrop | CelebA | CelebA-HQ | CelebA-HQ AFHQ |
| Batch size | 1 | 1 | 1 | 1 | 128 | 64 | 64 |
| Image size | 256 | 256 | 256 | 256 | 64 | 64 | 64 |
| $\bar{\beta}_T^2$ | 0.01 | 1 | 0.01 | 1 | 1 | 1 | 1 |
| $I_{in}$ | $I_{in}$ | $I_{in}$ | $I_{in}$ | $I_{in}$ | 0 | 0 | 0 |
| Sampling steps | 5 | 2 | 2 | 5 | 10 | 10 | 10 |
| Loss type | $\ell_1$ | $\ell_1$ | $\ell_1$ | $\ell_1$ | $\ell_2$ | $\ell_2$ | $\ell_2$ |
| Loss | $L_{res} + L_\epsilon$ | $L_{res}$ | $L_{res} + L_\epsilon$ | $L_{res} + L_\epsilon$ | $L_\epsilon$ | $L_{res}, L_\epsilon$ | $L_{res}, L_\epsilon$ |
| Sampling Method | SM-Res-N-2Net | SM-Res | SM-Res-N-2Net | SM-Res-N-2Net | SM-N | SM-Res-N-2Net | SM-Res-N-2Net |
| Optimizer | Adam | Adam | Adam | Adam | RAdam | RAdam | RAdam |
| Learning rate | 8e-5 | 8e-5 | 8e-5 | 8e-5 | 2e-4 | 2e-4 | 2e-4 |
| Training iterations | 80k | 80k | 400k | 120k | 100k | 100k | 100k |
| Schedules | $\alpha_t : P(1-x,1)$ $\beta_t^2 : P(x,1)$ | $\alpha_t : P(1-x,1)$ $\beta_t^2 : P(x,1)$ | $\alpha_t : P(1-x,1)$ $\beta_t^2 : P(x,1)$ | $\alpha_t : P(1-x,1)$ $\beta_t^2 : P(x,1)$ | $\alpha_{DDIM}^t \rightarrow$ $\alpha_t, \beta_t^2$ | $\alpha_t : P(1-x,1)$ $\beta_t^2 : P(x,1)$ | $\alpha_{DDIM}^t \rightarrow$ $\alpha_t, \beta_t^2$ |

Table 4. Experimental settings for training our RDDM."SM-Res-N-2Net" is described in Appendix B.2. Two optimizers can be implemented in $L_{res}, L_\epsilon$. We use "SM-Res-N-2Net" and $L_{res} + L_\epsilon$ on the SID-RGB dataset [65].

and path variations.

## B. Experiments

### B.1. Training Details

We use a UNet architecture[9] for both residual prediction and noise prediction in our RDDM. The UNet settings remain consistent across all tasks, including the channel size (64) and channel multiplier (1,2,4,8). Detailed experimental settings can be found in Table 4. Training and testing for all experiments in Table 4 can be conducted on a single Nvidia GTX 3090.

**Image Generation.** For comparison with DDIM [51], we convert the $\alpha_{DDIM}^t$ schedule of DDIM [51] into the $\alpha_t$ and $\beta_t^2$ schedules of our RDDM using Eq. 17 in Section 5.2 and Section 6. In fact, a better coefficient schedule can be used for training in our RDDM, e.g., $\alpha_t$ (linearly decreasing) and $\beta_t^2$ (linearly increasing) in Table 2. The quantitative results were evaluated by Frechet Inception Distance (FID) and Inception Score (IS).

**Image Restoration.** We extensively evaluate our method on several image restoration tasks, including shadow removal, low-light enhancement, image deraining, and image deblurring on 5 different datasets. For fair comparisons, the results of other SOTA methods are provided from the original papers whenever possible. For all image restoration tasks, the images are resized to 256, and the networks are trained with a batch size of 1. We use shadow masks and shadow images as conditions for shadow removal (similar to [28, 79]), while other image restoration tasks use the degraded image as condition inputs. For low-light enhancement, we use histogram equalization for pre-processing. To cope with the varying tasks and dataset sizes, we only modified the number of training iterations, $\bar{\beta}_T^2$ and sampling steps (5 steps for shadow removal and deraining, 2 steps for low-light and deblurring) as shown in Table 4. $\alpha_t$ is initialized using a linearly decreasing schedule (i.e., $P(1-x,1)$ in Eq. 18), while $\beta_t^2$ is initialized using a linearly increasing schedule (i.e., $P(x,1)$). The quantitative results were evaluated by Mean Absolute Error (MAE), Peak Signal to Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS) [73].

Notably, our RDDM uses an identical UNet architecture and is trained with a batch size of 1 for all these tasks. In contrast, SOAT methods often involve elaborate network architectures, such as multi-stage [13, 70, 80], multi-branch [10], and GAN [27, 45, 57], or sophisticated loss functions like the chromaticity [20], texture similarity [74], and edge loss [70].

**Image Inpainting and Image Translation.** To increase the diversity of the generated images, conditional input images were not fed into the deresidual and denoising network (see Fig. 19).

[9]Our RDDM is implemented by modifying https://github.com/lucidrains/denoising-diffusion-pytorch repository.

**Algorithm 2:** Training Pipeline Using AOSA.

**Input :** A degraded input image $I_{in}$ and its corresponding ground truth image $I_0$. Gaussian noise $\epsilon$. Time condition $t$. Coefficient schedules $\bar{\alpha}$ and $\bar{\beta}_t$. The initial learnable parameters $\lambda_{res}^{\theta} = 0.5$. Network $G$ with parameters $\theta$. The initial learning rate $l$. $n$ is the training iterations number. $m$ is the iterations number of AOSA. The threshold of shifting training strategies, $\delta = 0.01$.

**Output:** Trained well parameters, $\theta, \lambda_{res}^{\theta}$.

```
1  $\theta \leftarrow$ InitWight ($G$)                                        ▷ initialize network parameters
2  for $i \leftarrow 1$ to $n + m$ do
3      $t \sim$Uniform ($\{1, 2, ..., T\}$), $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $I_{res} \leftarrow I_{in} - I_0$
4      $I_t \leftarrow I_0 + \bar{\alpha}_t I_{res} + \bar{\beta}_t \epsilon$                      ▷ synthesize $I_t$ by Eq. 7
5      $I_{out} \leftarrow G(I_t, t, I_{in})$
6      $I_{res}^{\theta} \leftarrow \lambda_{res}^{\theta} \times I_{out} + (1 - \lambda_{res}^{\theta}) \times f_{\epsilon \to res}(I_{out})$   ▷ $f_{\epsilon \to res}(\cdot)$: from $\epsilon$ to $I_{res}$ using Eq. 16
7      $\epsilon_{\theta} \leftarrow \lambda_{res}^{\theta} \times f_{res \to \epsilon}(I_{out}) + (1 - \lambda_{res}^{\theta}) \times I_{out}$   ▷ $f_{res \to \epsilon}(\cdot)$: from $I_{res}$ to $\epsilon$ using Eq. 16
8      $L_{auto} \leftarrow$Loss ($I_{res}^{\theta}, I_{res}, \epsilon_{\theta}, \epsilon$)                ▷ based on Eq. 53
9      $\theta, \lambda_{res}^{\theta} \xleftarrow{+} -\nabla_{\theta, \lambda_{res}^{\theta}}(\mathcal{L}_{auto}, l)$                  ▷ updating gradient
10     if abs ($\lambda_{res}^{\theta} - 0.5$) $< \delta$ then
11         pass                                                      ▷ adversarial-like training
12     else
13         $\lambda_{res}^{\theta} \leftarrow$Detach ($\lambda_{res}^{\theta}$)                        ▷ halt the gradient updates
14         $\theta \leftarrow$ InitWight ($G$)        ▷ reinitialize network parameters if $\lambda_{res}^{\theta} > 0.5$ then
15             $\lambda_{res}^{\theta} \leftarrow 1$                                        ▷ SM-Res
16         else
17             $\lambda_{res}^{\theta} \leftarrow 0$                                        ▷ SM-N
18         end
19     end
20 end
```

| Sampling Method | Network | MAE(↓) | SSIM(↑) | PSNR(↑) |
|---|---|---|---|---|
| SM-Res | Residual network | 4.76 | 0.959 | 30.72 |
| SM-Res-N-2Net | Residual network+noise network | <u>4.67</u> | <u>0.962</u> | <u>30.91</u> |
| SM-Res-N-1Net | One network, only shared encoder | 4.72 | 0.959 | 30.73 |
| SM-Res-N-1Net | One network | **4.57** | **0.963** | **31.10** |

Table 5. Ablation studies of sampling methods and network structures on the ISTD dataset [57]. "SM-Res-N-1Net+One network" denotes to output 6 channels using a network, where the 0-3-th channels are residual and the 3-6-th channels are noise.

## B.2. SAMPLING Details

In Section 4.3,we described the empirical selection process for SM-Res or SM-N. Below, we give some additional methods in detail.

**Automatic selection of SM-Res or SM-N by AOSA.** We present the motivation, conceptualization, and implementation pipeline (Algorithm 2) of the Automatic Objective Selection Algorithm (AOSA) as follows:

Step 1. At the initial simultaneous training (similar to SM-Res-N), we do not know whether the network output ($I_{out}$) is residual or noise. Therefore, we set $\lambda_{res}^{\theta} = 0.5$ to denote the probability that the output is residual ($I_{res}^{\theta}$), and $1 - \lambda_{res}^{\theta}$ is the probability that the output is noise ($\epsilon_{\theta}$).

Step 2. We then impose loss constraints on both residual and noise estimation weighted by the learned parameter ($\lambda_{res}^{\theta}$), as follows:

$$L_{auto}(\theta) := \lambda_{res}^{\theta} E\left[\left\|I_{res} - I_{res}^{\theta}(I_t, t, I_{in})\right\|^2\right] + (1 - \lambda_{res}^{\theta}) E\left[\left\|\epsilon - \epsilon_{\theta}(I_t, t, I_{in})\right\|^2\right]. \tag{53}$$

The joint loss functions $L_{auto}(\theta)$ drive the network to gradually tend to output residuals or noise based on the input. For

| Sampling Method | Shadow PSNR/SSIM | Deraining PSNR/SSIM | Deblurring PSNR/SSIM | Generation FID(↓) | IS(↑) |
|---|---|---|---|---|---|
| SM-Res | 30.72/0.959 | <u>31.96</u>/<u>0.9509</u> | <u>32.32</u>/<u>0.957</u> | 31.47 | 1.73 |
| SM-N | 11.34/0.175 | 19.15/0.7179 | 9.49/0.087 | **23.25** | **2.05** |
| SM-Res-N-2Net | <u>30.91</u>/<u>0.962</u> | **32.51/0.9563** | **32.40/0.963** | 28.90 | 1.78 |
| SM-Res-N-1Net | **31.10/0.963** | 31.79/0.9504 | 31.69/0.951 | <u>28.57</u> | <u>1.81</u> |

Table 6. Generalization analysis of "SM-Res-N-1Net+One network".

example, in the image restoration task with deterministic input, it should be simpler for the network to estimate a clear image than noise. In contrast, for the image generation task with random noise input, it is simpler for the network to estimate the noise than a clear image.

Step 3. To enable learning of $\lambda^{\theta}_{res}$, we then include it in the network computation, allowing gradient transmission. Since $\lambda^{\theta}_{res}$ denotes the probability that the output is residual, the estimated residual $I^{\theta}_{res}$ can be represented as $\lambda^{\theta}_{res} \times I_{out} + (1 - \lambda^{\theta}_{res}) \times f_{\epsilon \to res}(I_{out})$. $f_{\epsilon \to res}(\cdot)$ represents the transformation from $\epsilon$ to $I_{res}$ using Eq. 16. Similarly, $\epsilon_{\theta}$ can be represented as $\lambda^{\theta}_{res} \times f_{res \to \epsilon}(I_{out}) + (1 - \lambda^{\theta}_{res}) \times I_{out}$. This is very similar to the cross-entropy loss function.

Step 4. As the training process is completed, our objective should be to estimate only noise (SM-N) or residuals (SM-Res). By utilizing the learned $\lambda^{\theta}_{res}$, we can determine when to switch from an adversarial-like process (residuals vs. noise in Step 2) to a single prediction (residuals or noise). This transition can be controlled, for instance, by setting a condition such as abs($\lambda^{\theta}_{res} - 0.5) \geq 0.01$. When the network's tendency to estimate residuals surpasses 51% probability, we set $\lambda^{\theta}_{res}$ to 1 and halt the gradient updates for $\lambda^{\theta}_{res}$.

The experimental results were consistent with the empirical analysis in Section 4.3 and verified the effectiveness of AOSA. For instance, the initial simultaneous training switches to residual learning (SM-Res) for shadow removal and low-light enhancement in approximately 300 iterations, and to denoising learning (SM-N) for image generation in approximately 1000 iterations. To summarize, AOSA achieves the same inference cost as the current denoising-based diffusion methods [17] with the plug-and-play training strategy.

**SM-Res-N.** Both the residuals and the noise are predicted, which can be implemented with two or one networks. **SM-Res-N-2Net.** If computational resources are sufficient, two separate networks can be trained for noise and residual predictions, and the optimal sampling method can be determined during testing. This setting easily obtains a well-suited network for the target task, and facilitates the exploration of the decoupled diffusion process and the partially path-independent generation process in Section 5. **SM-Res-N-1Net.** To avoid training two separate networks, another solution is to simply use a joint network (i.e., a shared encoder and decoder) to output 6 channels where the 0-3-th channels are residual and the 3-6-th channels are noise. This setting loses the decoupling property of RDDM, but can achieve dual prediction with a slight cost. Table 5 shows that the joint network (i.e., SM-Res-N-1Net+One network) achieves the best shadow removal results (MAE 4.57), even better than two independent networks (4.67). A network with the shared encoder (MAE 4.72) has a slight performance degradation compared to the independent two networks (4.67). We conduct further generalization experiments on Table 6, indicating

| Method | MAE(↓) | | | SSIM(↑) | | | PSNR(↑) | | | LPIPS(↓) |
|---|---|---|---|---|---|---|---|---|---|---|
| | S | NS | ALL | S | NS | ALL | S | NS | ALL | |
| ST-CGAN [57] | 10.33 | 6.93 | 7.47 | 0.981 | 0.958 | 0.929 | 33.74 | 29.51 | 27.44 | - |
| DSC [19] ¶ | 9.48 | 6.14 | 6.67 | 0.967 | - | - | 33.45 | - | - | - |
| DHAN [10] | 8.14 | 6.04 | 6.37 | 0.983 | - | - | 34.50 | - | - | - |
| CANet [8] | 8.86 | 6.07 | 6.15 | - | - | - | - | - | - | - |
| LG-ShadowNet [37] | 10.23 | 5.38 | 6.18 | 0.979 | 0.967 | 0.936 | 31.53 | 29.47 | 26.62 | - |
| FusionNet [13] | 7.77 | 5.56 | 5.92 | 0.975 | 0.880 | 0.945 | 34.71 | 28.61 | 27.19 | 0.1204 |
| UnfoldingNet [80] | 7.87 | 4.72 | 5.22 | 0.987 | <u>0.978</u> | 0.960 | **36.95** | 31.54 | 29.85 | - |
| BMNet [79] | 7.60 | 4.59 | 5.02 | <u>0.988</u> | 0.976 | 0.959 | 35.61 | 32.80 | 30.28 | 0.0377 |
| DMTN [31] | <u>7.00</u> | <u>4.28</u> | <u>4.72</u> | **0.990** | **0.979** | **0.965** | 35.83 | <u>33.01</u> | <u>30.42</u> | <u>0.0368</u> |
| Ours (RDDM) | **6.67** | **4.27** | **4.67** | <u>0.988</u> | **0.979** | <u>0.962</u> | <u>36.74</u> | **33.18** | **30.91** | **0.0305** |

Table 7. Shadow removal results on the ISTD dataset [57]. We report the MAE, SSIM and PSNR in the shadow area (S), non-shadow area (NS), and whole image (ALL).

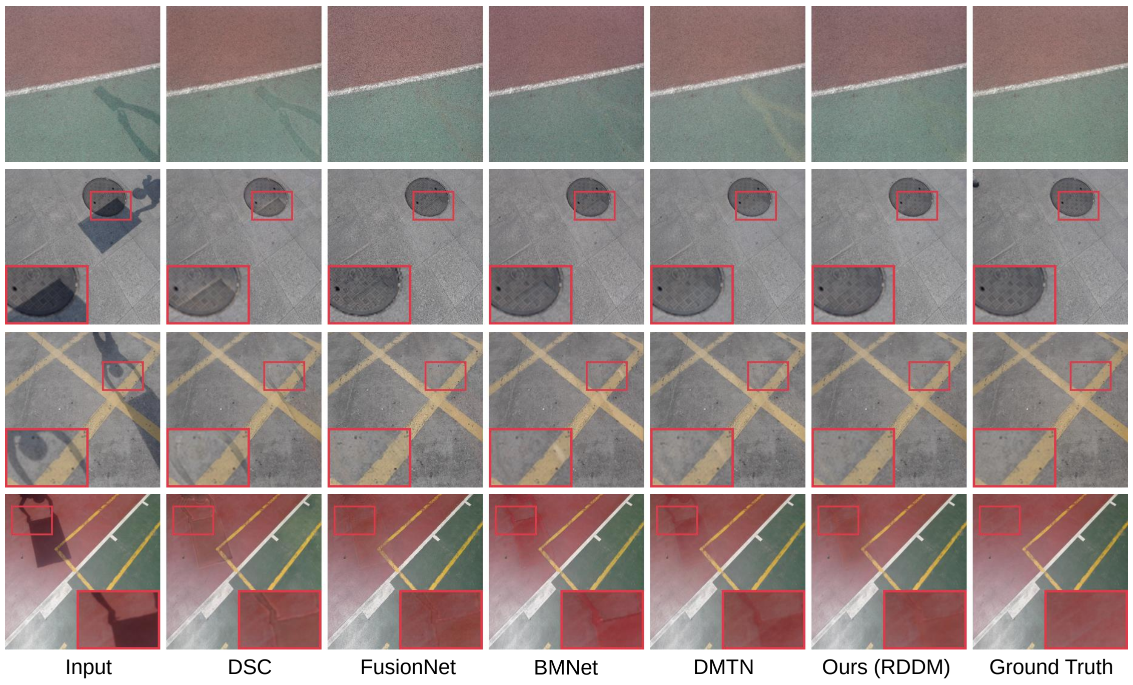

Figure 9. More visual comparison results for shadow removal on the ISTD dataset [57].

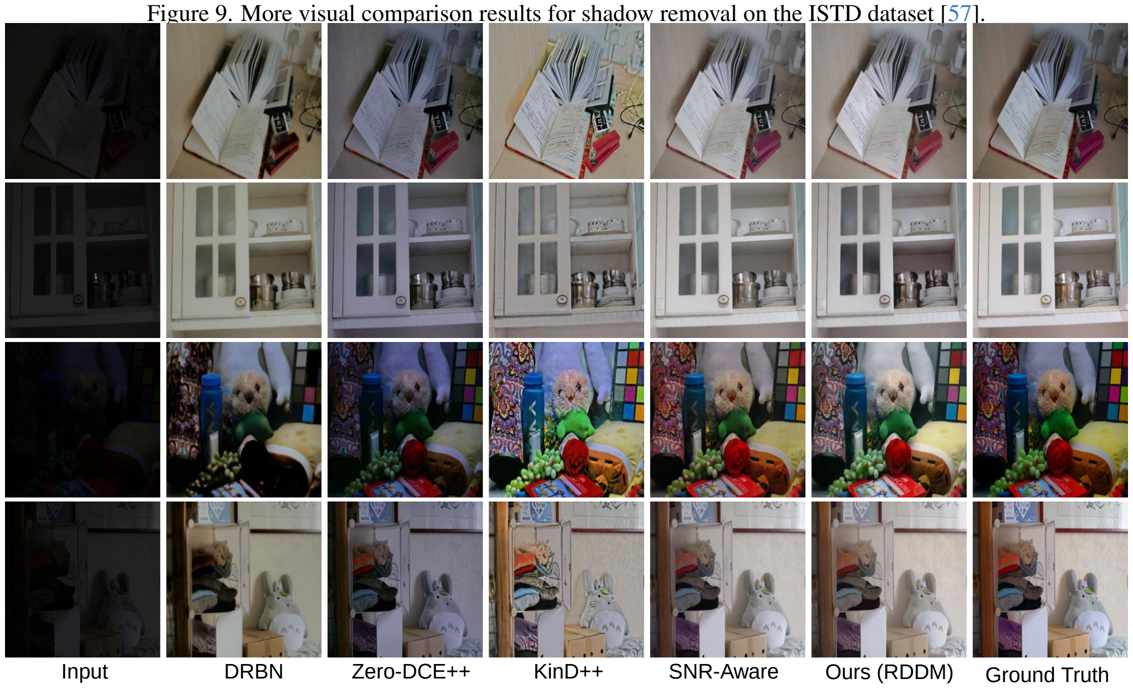

Figure 10. More visual comparison results for Low-light enhancement on the LOL dataset [61].

"SM-Res-N-1Net+One network" isn't bad, but not always the best.

### B.3. More Results

**Shadow removal.** We compare RDDM with DSC [19], FusionNet [13], BMNet [79] and DMTN [31] on the ISTD dataset [57]. The ISTD dataset [57] contains shadow images, shadow masks, and shadow-free image triplets (1,330 for training; 540 for testing). Table 3(b), Fig. 7(b), and Fig. 9 demonstrate the superiority of our method. In addition, we compare RDDM with more shadow removal methods (e.g., ST-CGAN [57], DHAN [10], CANet [8], LG-ShadowNet [37],

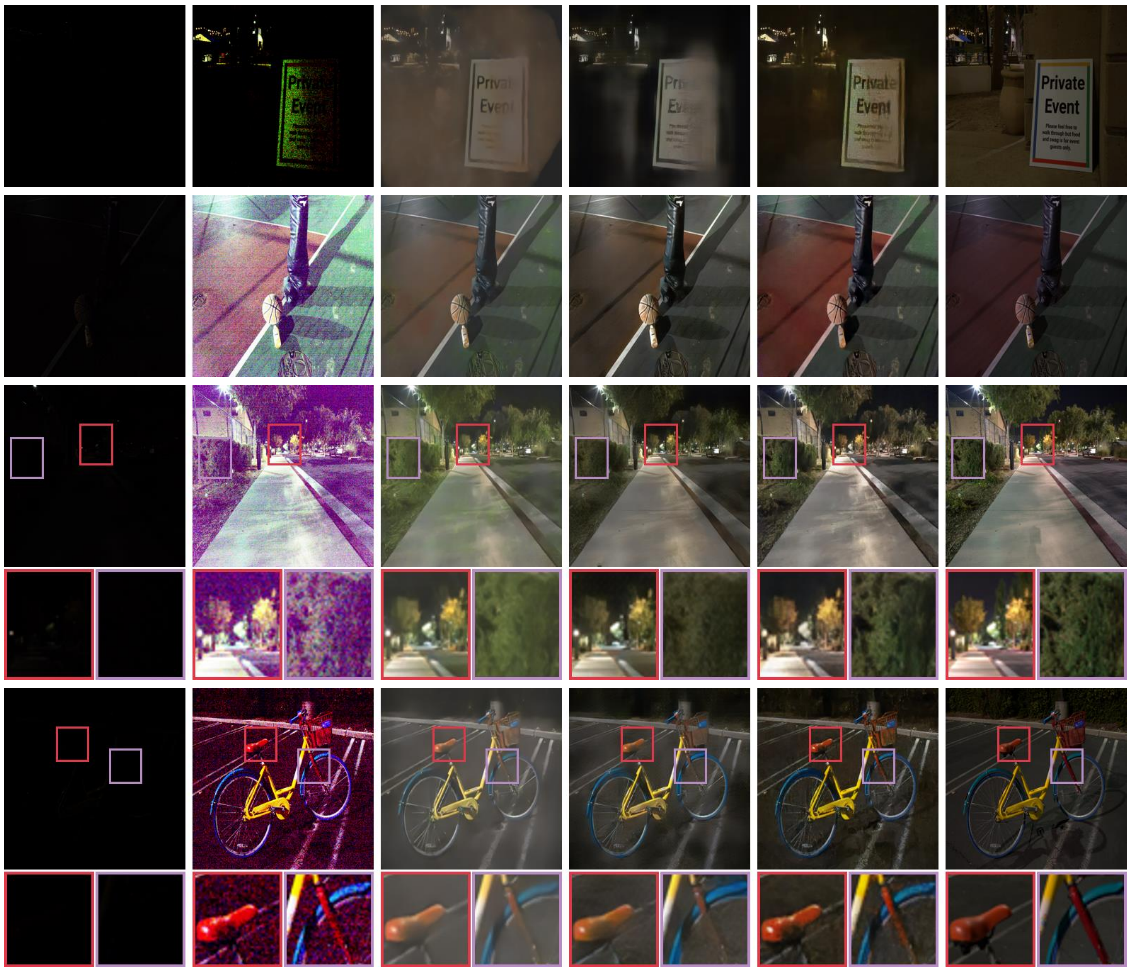

Figure 11. More visual comparison results for Low-light enhancement on the SID-RGB dataset [65].

| Low-light (SID-RGB) | PSNR(↑) | SSIM(↑) | LPIPS(↓) |
|---|---|---|---|
| SID [7] | 21.16 | 0.6398 | 0.4026 |
| D&E [65] | 22.13 | 0.7172 | 0.3794 |
| MIR-Net[69, 71] | 22.34 | 0.7031 | 0.3562 |
| UTVNet [78] | 22.69 | <u>0.7179</u> | 0.3417 |
| SNR-Aware [66] | <u>22.87</u> | 0.625 | - |
| Our RDDM (2 step) | **23.97** | **0.8392** | <u>0.2433</u> |
| Our RDDM (5 step) | <u>23.80</u> | <u>0.8378</u> | **0.2289** |

Table 8. Quantitative comparison results of Low-light enhancement on the SID-RGB dataset [65]. The results of MIR-Net are reported by [78].

UnfoldingNet [80]) in Table 7.

**Low-light enhancement.** We evaluate our RDDM on the LOL datasets [61] (500 images) and SID-RGB [65] dataset (5,094 images), and compare our method with the current SOTA methods [33, 66, 68, 71, 76–78]. To unify and simplify the data loading pipeline for training, we only evaluate the RGB low-light image dataset [61, 65], not the RAW datasets (e.g., FiveK [6]). Table 3(c), Fig. 7(c), and Fig. 10 show that our RDDM achieves the best SSIM and LPIPS [73] and can recover better visual quality on the LOL [61] dataset. Table 8 shows the low-light enhancement results on the SID-RGB [65]

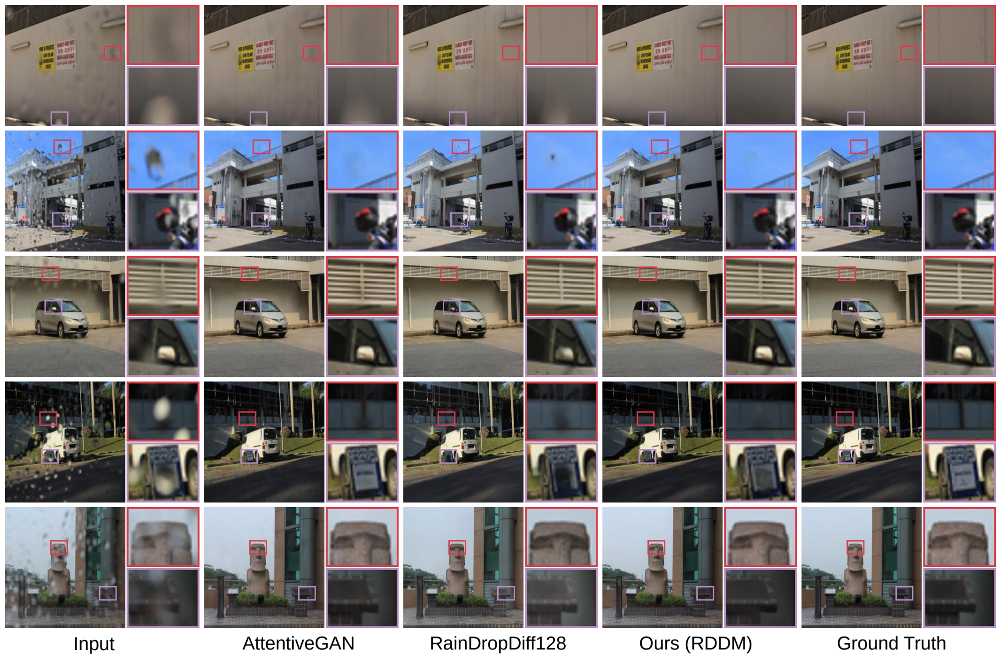

Figure 12. More visual comparison results for deraining on the RainDrop [45] dataset.

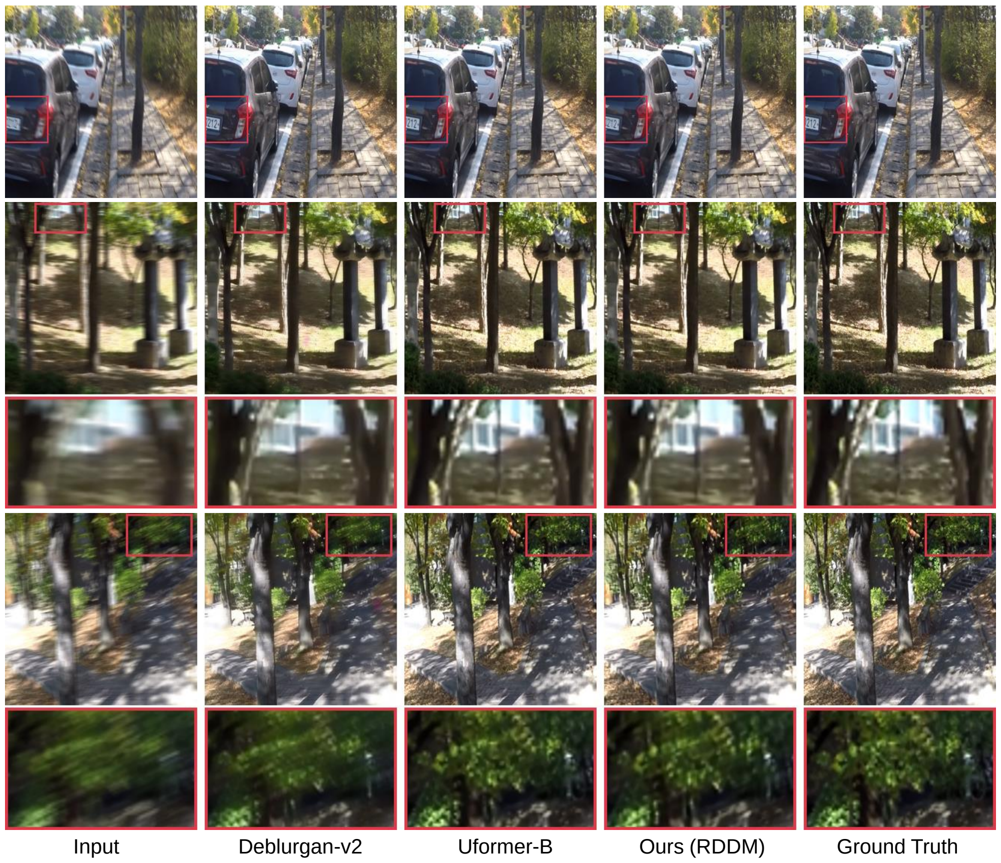

Figure 13. More visual comparison results for deblurring on the GoPro [43] dataset.

dataset of different methods. Our RDDM outperforms the state-of-the-art SNR-Aware [66] by a $\mathbf{4.8\%}$ PSNR and a $\mathbf{34.2\%}$

| Deblurring (GoPro) | PSNR(↑) | SSIM(↑) | LPIPS(↓) |
|---|---|---|---|
| Deblurgan-v2 [27] | 29.55 | 0.934 | 0.117 |
| Suin *et al.* [55] | 31.85 | 0.948 | - |
| MPRNet [70] | <u>32.66</u> | 0.959 | 0.089 |
| DvSR [62] | 31.66 | 0.948 | 0.059 |
| Uformer-B [60] | **32.97** | **0.967** | **0.0089** |
| I2SB [29] | 29.31 | 0.906 | 0.0961 |
| InDI [11] | 31.49 | 0.946 | 0.058 |
| Our RDDM (2 step) | 32.40 | <u>0.963</u> | 0.0415 |
| Our RDDM (10 step) | 31.67 | 0.950 | <u>0.0379</u> |

Table 9. Quantitative comparison results of deblurring on the GoPro dataset [43].

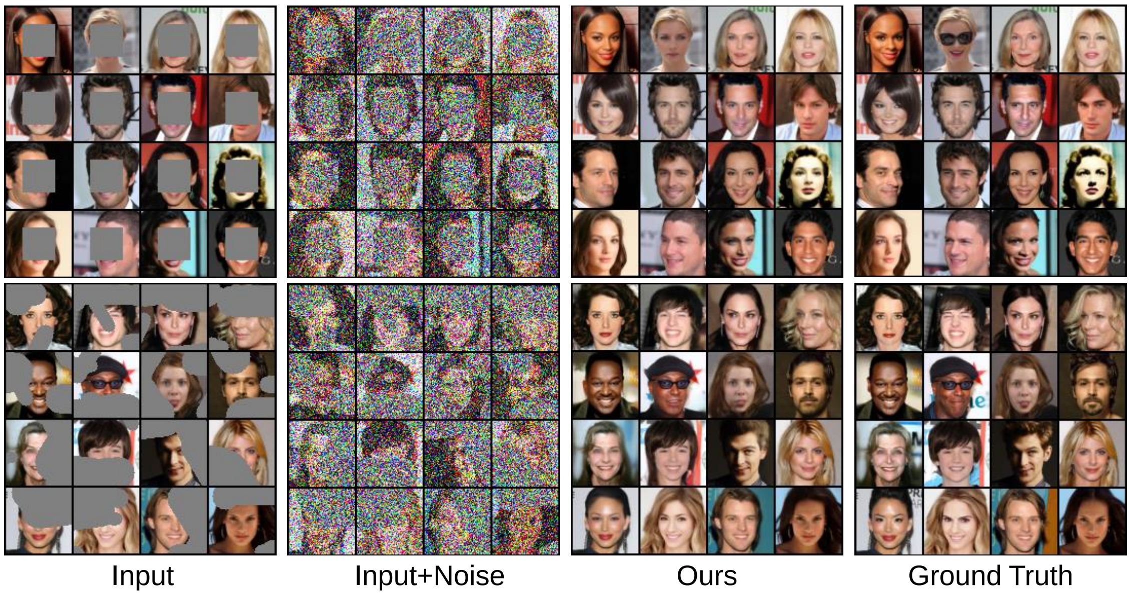

Figure 14. More visual results for image inpainting on the CelebA-HQ [23] dataset.

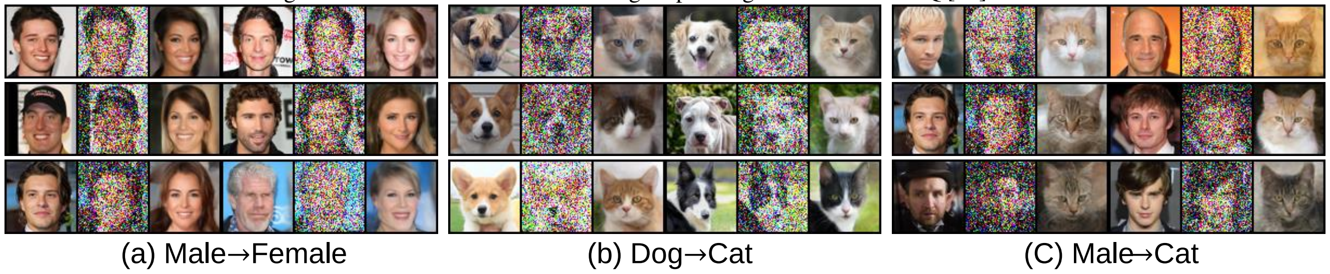

Figure 15. More visual results for image translation on the CelebA-HQ [23] and AFHQ [9] datasets.

SSIM improvement on the SID-RGB [65] dataset. Fig. 11 shows that our RDDM outperforms competitors in detail recovery (sharper text of 1st row), and color vibrancy (2nd & 3rd rows), avoiding issues like gray shading and detail blurring.

**Image deraining.** We make a fair comparison with the current SOTA diffusion-based image restoration method - RainDiff128 [82] ("128" denotes the 128×128 patch size for training) on the RainDrop dataset [45] (1119 images). RainDiff128 [82] feeds the degraded input image as a condition to the denoising network, which requires 50 sampling steps to generate a clear image from the noise, while our RDDM requires only 5 sampling steps to recover the degraded image from the noise-carrying input image and outperforms RainDiff128 [82], as shown in Table 3(d) and Fig. 12.

**Image deblurring.** We evaluate our method on the widely used deblurring dataset - GoPro [43] (3,214 images). Table 9

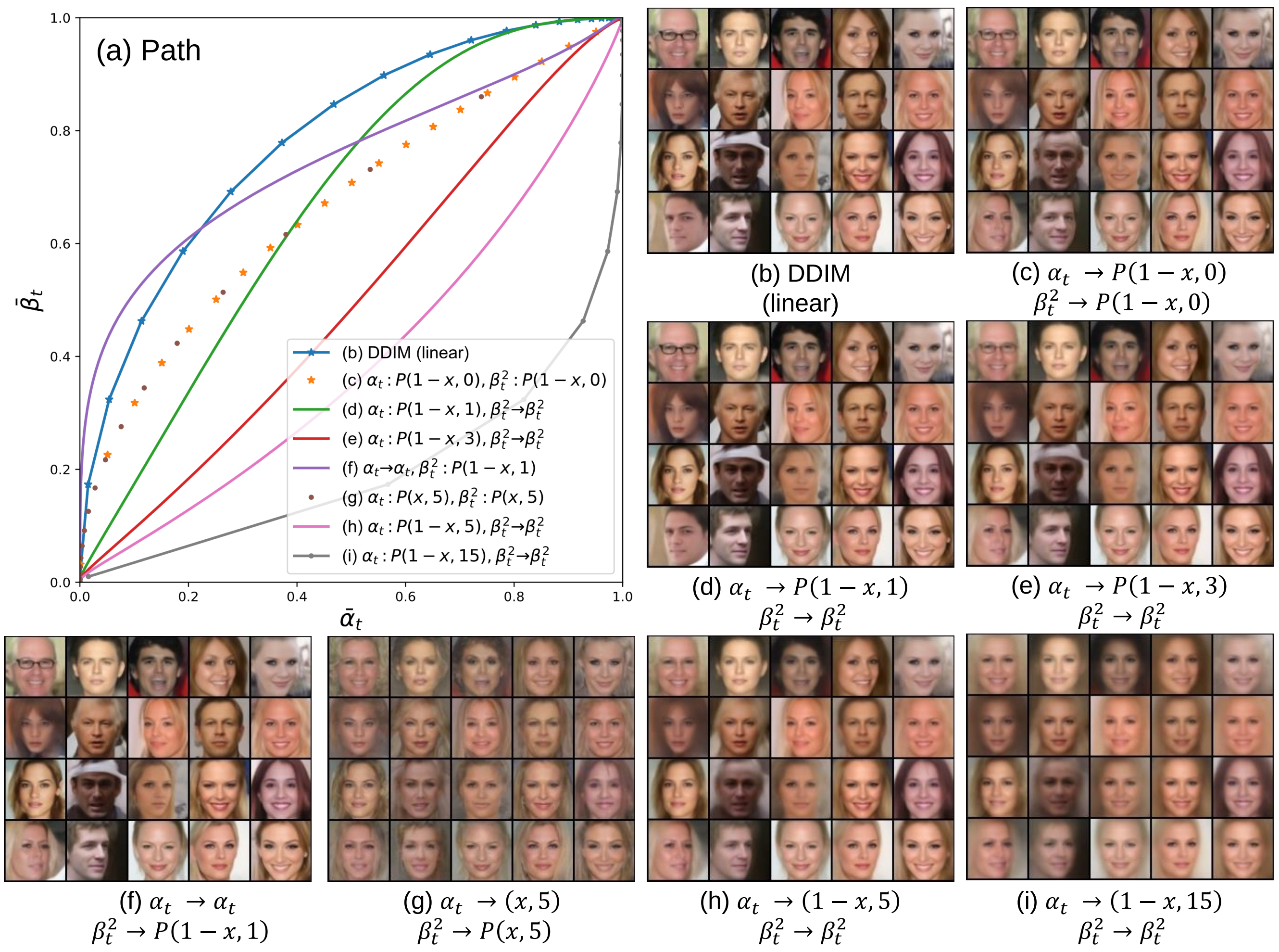

Figure 16. More visual results for the partially path-independent generation process. Two networks are used to estimate residuals and noise separately, i.e., $I_{res}^{\theta}(I_t, \bar{\alpha}_t \cdot T)$ and $\epsilon_\theta(I_t, \bar{\beta}_t \cdot T)$ ($\eta = 0$).

shows that our RDDM is the second best LPIPS. Fig. 13 shows that our method is competitive with the SOTA deblurring methods (e.g, Uformer-B [60]) in terms of visual quality.

**Image Inpainting and Image Translation.** We show more qualitative results of image inpainting (Fig. 14) and translation (Fig. 15).

## B.4. Partially Path-independent Generation Process

Fig. 16(b-f) provides evidence supporting the partially path-independent generation process, demonstrating the robustness of the generative process within a certain range of diffusion rates (step size per step) and path variation, e.g., converting DDIM [51] to a uniform diffusion speed in Fig. 16(c). However, excessive disturbances can result in visual inconsistencies, as depicted in Fig. 16(h)(i). Furthermore, Fig. 16(c) and Fig. 16(g) illustrate that even when the paths are the same, the variation in diffusion speed significantly impacts the quality of the generated images. This highlights the importance of carefully considering and controlling the diffusion speed and generation path during the generation process.

We also investigated two reverse paths to gain insight into the implications of the proposed partial path independence. In the first case, the residuals are removed first, followed by the noise: $I(T) \xrightarrow{-I_{res}} I(0) + \bar{\beta}_T\epsilon \xrightarrow{-\bar{\beta}_T\epsilon} I(0)$, as shown in Fig. 17(b1)(b2). The second case involves removing the noise first and then the residuals: $I(T) \xrightarrow{-\bar{\beta}_T\epsilon} I_{in} \xrightarrow{-I_{res}} I(0)$. In the first case, images are successfully generated (as shown in Fig. 17(b)) which exhibit a striking similarity to the default images in Fig. 17(a). However, the second case shown in Fig. 17(c) fails to go from $I_{in}$ to $I(0)$ due to $I_{in} = 0$ in the generation task. Figure 17(d) shows the intermediate visualization results of removing the noise first.

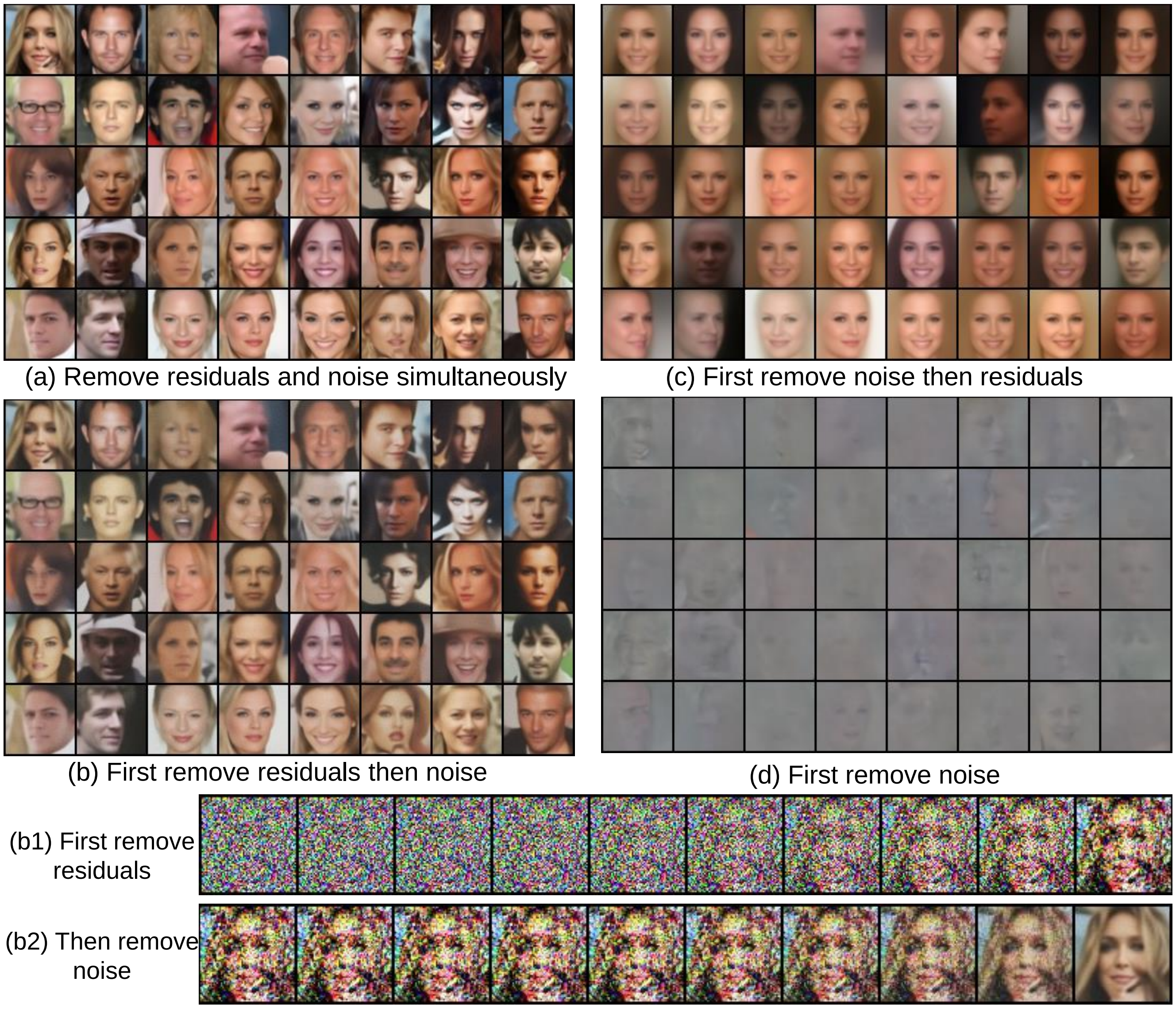
(a) Remove residuals and noise simultaneously
(c) First remove noise then residuals
(b) First remove residuals then noise
(d) First remove noise
(b1) First remove residuals
(b2) Then remove noise

Figure 17. Special paths of the partially path-independent generation process. Two networks are used to estimate residuals and noise separately, i.e., $I^{\theta}_{res}(I_t, \bar{\alpha}_t \cdot T), \epsilon_\theta(I_t, \bar{\beta}_t \cdot T)$ ($\eta = 0$).

## B.5. Ablation Studies

We have analyzed the sampling method in Table 1, the coefficient schedule in Table 2, and the network structure for SM-Res-N in Table 5.

**Sampling Methods.** We present the results for noise predictions only (SM-N) in Fig. 18. Fig. 18 (b) and (c) illustrate that estimating only the noise poses challenges as colors are distorted, and it becomes difficult to retain information from the input shadow image. We found that increasing sampling steps does not lead to improved results from Fig. 18 (b) to Fig. 18 (c), which may be an inherent limitation when estimating only the noise for image restoration. Actually, this is also reflected in DeS3 [21] (a shadow removal method based on a denoising diffusion model), where DeS3 [21] specifically designs the loss against color bias. Additionaly, training with batch size 1 may contribute to poor results of only predicting noise. However, estimating only the residuals (SM-Res) with batch size 1 does not exhibit such problems for image restoration, as demonstrated in Fig. 18 (d)&(e) and Table 1, further demonstrating the merits of our RDDM. For image inpainting, SM-Res-N-2Net can generate more realistic face images compared to SM-N and SM-Res, as shown in Fig. 19(d-f). If computational resources are sufficient, to obtain better image quality for an unknown task, we suggest that two separate networks can be trained for noise and residual predictions, and the optimal sampling method can be determined during

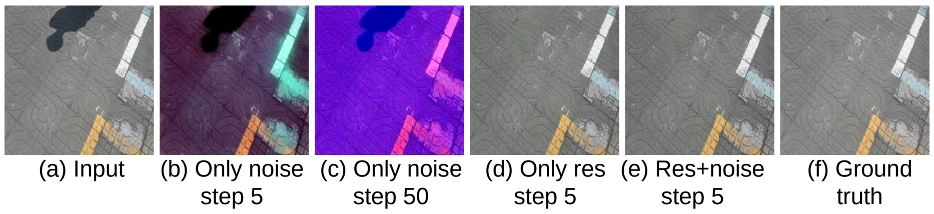

Figure 18. Visualizing ablation studies of sampling methods.

testing. If computational resources are limited, the sampling method can be determined empirically (see Section 4.3).

| RDDM (SM-Res-N-2Net) | metric | 1 step | 2 step | 5 step | 10 step | 100 step |
|---|---|---|---|---|---|---|
| $\bar{\beta}_T^2 = 0.01$ | MAE-ALL (↓) | 4.83 | 4.69 | **4.67** | 4.72 | 4.90 |
| | PSNR-S (↑) | 36.83 | **36.98** | 36.74 | 36.59 | 36.41 |
| | LPIPS (↓) | 0.0344 | 0.0308 | **0.0305** | 0.0314 | 0.0334 |
| $\bar{\beta}_T^2 = 1$ | MAE-ALL (↓) | 5.07 | 4.94 | 4.90 | **4.87** | 4.99 |
| | PSNR-S (↑) | 36.93 | **37.20** | 37.07 | 37.01 | 36.62 |
| | LPIPS (↓) | 0.0346 | 0.0314 | **0.0298** | 0.0300 | 0.0319 |

Table 10. Ablation studies of sampling steps for shadow removal on the ISTD dataset [57].

**Sampling Steps.** Table 10 shows that our RDDM performance improves as the number of sampling steps increases for shadow removal. Unlike the findings in InDI [11] where one step yields the best MSE reconstruction, RDDM achieved its best MAE at 5 steps for shadow removal, which may be due to differences in coefficient schedules and sampling methods.

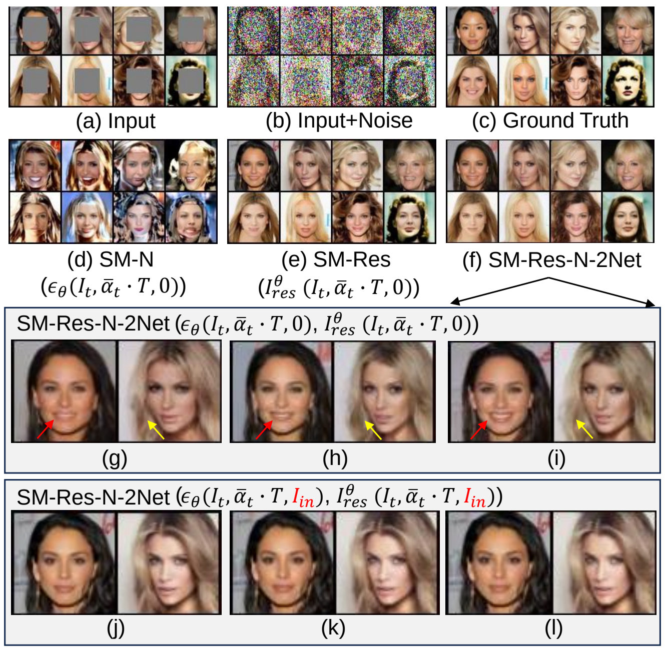
(a) Input
(b) Input+Noise
(c) Ground Truth
(d) SM-N
$(\epsilon_\theta(I_t, \bar{\alpha}_t \cdot T, 0))$
(e) SM-Res
$(I_{res}^\theta\ (I_t, \bar{\alpha}_t \cdot T, 0))$
(f) SM-Res-N-2Net
SM-Res-N-2Net $(\epsilon_\theta(I_t, \bar{\alpha}_t \cdot T, 0), I_{res}^\theta\ (I_t, \bar{\alpha}_t \cdot T, 0))$
(g)
(h)
(i)
SM-Res-N-2Net $(\epsilon_\theta(I_t, \bar{\alpha}_t \cdot T, I_{in}), I_{res}^\theta\ (I_t, \bar{\alpha}_t \cdot T, I_{in}))$
(j)
(k)
(l)

Figure 19. Visualizing ablation studies of sampling methods for image inpainting on the CelebA-HQ [23] dataset. (g-i) The conditional input image (a) is not used as an input to the deresidual and denoising network in the generation process from (b) to (f). Compared to (g-i), the diversity of the generated images in (j-l) decreases.

**Certainty and diversity.** Indeed, feeding conditional input images (Fig. 19(a)) into the deresidual and denoising network enhances the certainty of the generated images, while diminishing diversity, as shown in Fig. 19(j-l). Generating a clear target image directly from a noisy-carrying degraded image (Fig. 19(b)) without any conditions increases diversity, but changes non-missing regions (Fig. 19(g-i)).

**Noise Perturbation Intensity.** Table 10 shows that noise might play a beneficial role in restoring image details and enhancing perceptual quality. For image generation, the diversity of the generated images decreases significantly as $\bar{\beta}_T^2$ decreases from 1 in Fig. 20(c) to 0.01 in Fig. 20(b). The experiment is related to the mean face [18, 38, 42, 63] and

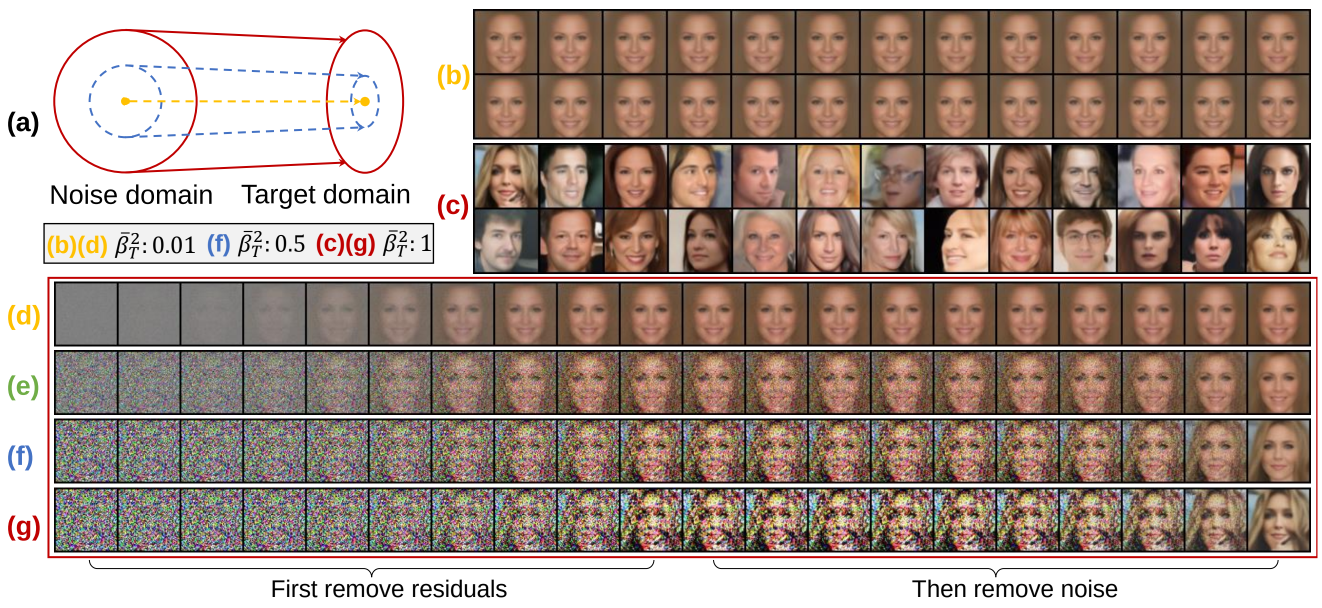

Figure 20. Visualizing ablation studies of noise perturbation intensity ($\bar\beta_T^2$). (a) We change the variance ($\bar\beta_T^2$) during testing, specifically by variance transformation via Algorithm 1. (b-c) When $\bar\beta_T^2$ decreases from 1 in (c) to 0.01 in (b), the diversity of the generated images decreases significantly. (d-g) We visualize each step in the generation process. $\bar\beta_T^2 = 0.01$ in (d), $\bar\beta_T^2 = 0.1$ in (e), $\bar\beta_T^2 = 0.5$ in (f), and $\bar\beta_T^2 = 1$ in (g). The sampling method is SM-Res-N-2Net with 10 sampling steps.

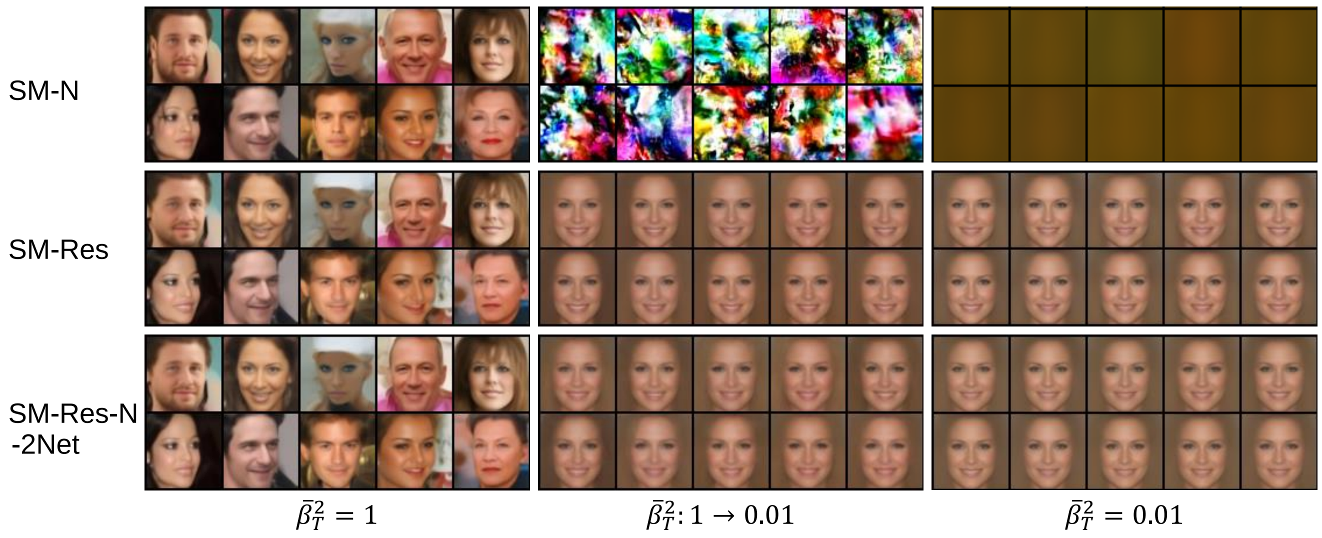

Figure 21. Visualizing ablation studies of sampling methods with different intensities of noise perturbation ($\bar\beta_T^2$). "$\bar\beta_T^2 : 1 \to 0.01$" denotes that the variance ($\bar\beta_T^2$) is changed during testing by variance transformation via Algorithm 1. The sampling steps are 10.

could provide useful insights to better understand the generative process. Fig. 21 shows that modifying $\bar\beta_T^2$ during testing ($\bar\beta_T^2 : 1 \to 0.01$) causes SM-N to fail to generate meaningful faces. SM-Res-N-2Net including deresidual and denoising networks can generate meaningful face images like SM-Res, indicating that the denoising network can perform denoising when modifying $\bar\beta_T^2$, but cannot obtain robust residuals ($I_{res}^{\theta}$) in the sampling process by Eq. 16. In summary, the deresidual network is relatively robust to noise variations compared to the denoising network.

**Resource efficiency.** Due to fewer sampling steps, our RDDM inference time and performance is comparable to lllflow [59], and LLFormer [58] (not diffusion-based). Compared to SR3 [49], our RDDM (only res in Table 11(b)) has 10x fewer training iterations, 10x fewer parameters, 10x faster inference time, and 10% improvement in PSNR and SSIM on the ISTD [57] dataset (shadow removal). For a fair comparison, priori shadow masks are used in SR3 [49] with a batch

| (a) Low-light | PSNR(↑) | SSIM(↑) | LPIPS(↓) | Params(M) | MACs(G)×Steps | Inference Time(s) |
|---|---|---|---|---|---|---|
| LLformer | 23.649 | 0.816 | 0.169 | 24.51 | **22.0×1 = 22.0** | 0.09×1 = 0.09 |
| LLFlow | 25.19 | 0.93 | **0.11** | 17.42 | 286.33×1 = 286.3 | 0.18×1 = 0.18 |
| Ours(RDDM) | **25.392** | **0.937** | 0.116 | **7.73** | 32.9×2 = 65.8 | **0.03×2 = 0.06** |

| (b) Shadow Removal | MAE(↓) | PSNR(↑) | SSIM(↑) | Params(M) | MACs(G) × Steps | Inference Time(s) |
|---|---|---|---|---|---|---|
| Shadow Diffusion [14] | **4.12** | **32.33** | **0.969** | - | - | - |
| SR3 [49] (80k) | 14.22 | 25.33 | 0.780 | 155.29 | 155.3×100=15530.0 | 0.02×100 = 2.00 |
| SR3 [49] (500K) | 13.38 | 26.03 | 0.820 | 155.29 | 155.3×100=15530.0 | 0.02×100 = 2.00 |
| SR3 [49] (1000K) | 11.61 | 27.49 | 0.871 | 155.29 | 155.3×100=15530.0 | 0.02×100 = 2.00 |
| Ours (only res, 80k) | 4.76 | 30.72 | 0.959 | **7.74** | **33.5×5 = 167.7** | **0.03×5 = 0.16** |
| Ours (80k) | 4.67 | 30.91 | 0.962 | 15.49 | 67.1×5 = 335.5 | 0.06×5 = 0.32 |

| (c) Deraining | PSNR(↑) | SSIM(↑) | Params(M) | MACs(G) × Steps | Inference Time(s) |
|---|---|---|---|---|---|
| RainDiff64[28] | 32.29 | 0.9422 | 109.68 | 252.4×10 = 2524.2 | 0.03×10 = 0.38 |
| RainDiff128[28] | 32.43 | 0.9334 | 109.68 | 248.4×50 = 12420.0 | 0.038×50 = 1.91 |
| Ours (only res) | 31.96 | 0.9509 | **7.73** | **32.9×5 = 164.7** | **0.032×5 = 0.16** |
| Ours | **32.51** | **0.9563** | 15.47 | 65.8×5 = 329.3 | 0.07×5 = 0.35 |

Table 11. Resource efficiency and performance analysis by THOP. "MAC" means multiply-accumulate operation. (a) Low-light enhancement on the LoL dataset [61]. (b) Shadow removal on the ISTD dataset [57]. For a fair comparison, a priori shadow mask are used in SR3 with a batch size of 1. (c) Deraining on the RainDrop dataset [45].

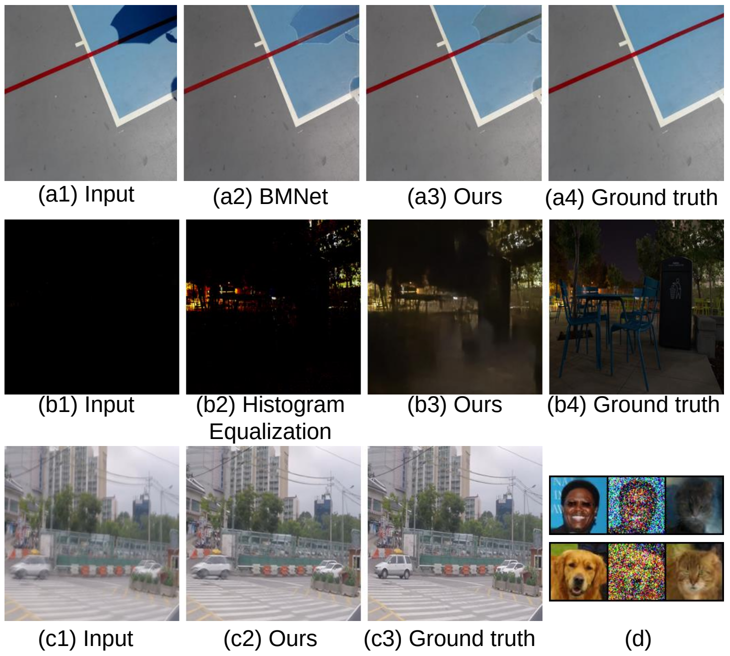

Figure 22. Failure cases. (a1-a4) Shadow removal on the ISTD dataset [57]. (b1-b4) Low-light enhancement on the SID-RGB dataset [65]. (c1-c3) Deblurring on the GoPro [43] dataset. (d) Image translation (male/dog→cat) on the CelebA-HQ [23] and AFHQ [9] datasets.

size of 1. ShadowDiffusion [14] uses SR3 [49] and Uformer [60], which has a higher PSNR but is also expected to be more computationally expensive. **Our RDDM with SM-Res requires only 4.8G of GPU memory for training.** Experiments in low-light enhancement, shadow removal, and deraining demonstrate the effectiveness of RDDM, enabling computationally-constrained researchers to utilize diffusion modeling for image restoration tasks.

**Accelerating Convergence.** The residual prediction in our RDDM helps the diffusion process to be more certain, which can accelerate the convergence process, e.g., fewer training iterations and higher performance in Table 11(b).

**Failure case.** We present some failure cases in Fig. 22.

## C. Discussions, Limitations, and Further Work

**Limitations.** Our primary focus has been on developing a unified prototype model for image restoration and generation, which may result in certain performance limitations when compared to task-specific state-of-the-art methods. To further improve the performance of a specific task, potential avenues for exploration include using a UNet with a larger number of parameters, increasing the batch size, conducting more iterations, and implementing more effective training strategies, such as learning rate adjustments customized for different tasks. For the image generation task, although Table 2 showcases the development of an improved coefficient schedule, attaining state-of-the-art performance in image generation necessitates further investigation and experimentation. In summary, while we recognize the existing performance limitations for specific tasks, we are confident that our unified prototype model serves as a robust foundation for image restoration and generation.

**Further Work.** Here are some interesting ways to extend our RDDM.

1. In-depth analysis of the relationship between RDDM and curve/multivariate integration.
2. Development of a diffusion model trained with one set of pre-trained parameters to handle several different tasks.
3. Implementing adaptive learning coefficient schedules to reduce the sampling steps while improving the quality of the generated images.
4. Constructing interpretable multi-dimensional latent diffusion models for multimodal fusion, e.g., generating images using text and images as conditions.
5. Adaptive learning noise intensity ($\beta_T^2$) for an unknown new task.
6. Exploring residuals in distillation (e.g., introducing dual diffusion into consistency models [54]).