Effective Cloud Removal for Remote Sensing Images by an Improved Mean-Reverting Denoising Model with Elucidated Design Space

Supplementary Material

A. Derivation of formulas

A.1. Forward Process

The forward process (*i.e.*, diffusion process) is defined as the SDE in Eq. (3). The goal of this section is to derive the form of $p_{0t}\big(\boldsymbol{x}(t) \mid \boldsymbol{x}(0), \boldsymbol{\mu}\big)$, which is also called the perturbation kernel. We can rewrite the form of Eq. (3) into:

$$\mathrm{d}\boldsymbol{x} = -f(t)(\boldsymbol{\mu} - \boldsymbol{x})\mathrm{d}t + g(t)\mathrm{d}\boldsymbol{\omega}_t, \tag{20}$$

whose solution has already been solved in IR-SDE (Eq. (6) in [34]), as

$$p_{0t}\big(\boldsymbol{x}(t) \mid \boldsymbol{x}(0), \boldsymbol{\mu}\big) = \mathcal{N}\big(\boldsymbol{x}(t); \boldsymbol{m}_t, v_t\boldsymbol{I}\big), \tag{21}$$

$$\boldsymbol{m}_t = \boldsymbol{\mu} + \big(\boldsymbol{x}(0) - \boldsymbol{\mu}\big)e^{-\bar{\theta}_{0:t}}, v_t = \int_0^t g(\xi)^2 e^{-2\bar{\theta}_{\xi:t}}\mathrm{d}\xi, \tag{22}$$

where $\bar{\theta}_{s:t} = \int_s^t -f(\xi)\mathrm{d}\xi$. Thus,

$$\boldsymbol{m}_t = \boldsymbol{\mu} + \big(\boldsymbol{x}(0) - \boldsymbol{\mu}\big)\exp\left(-\int_0^t -f(\xi)\mathrm{d}\xi\right) = \boldsymbol{\mu} + \big(\boldsymbol{x}(0) - \boldsymbol{\mu}\big)s(t), \tag{23}$$

$$v(t) = \int_0^t g(\xi)^2 \exp\left(-2\int_\xi^t -f(z)\mathrm{d}z\right)\mathrm{d}\xi = \int_0^t \left[g(\xi)\exp\left(\int_\xi^t f(z)\mathrm{d}z\right)\right]^2\mathrm{d}\xi \tag{24}$$

$$= \int_0^t \left[g(\xi)\exp\left(\int_0^t f(z)\mathrm{d}z - \int_0^\xi f(z)\mathrm{d}z\right)\right]^2\mathrm{d}\xi = \int_0^t \left[\left(\frac{g(\xi)}{\exp\left(\int_0^\xi f(z)\mathrm{d}z\right)}\right)^2 \exp\left(2\int_0^t f(z)\mathrm{d}z\right)\right]\mathrm{d}\xi \tag{25}$$

$$= \exp\left(2\int_0^t f(z)\mathrm{d}z\right)\int_0^t \left(\frac{g(\xi)}{s(\xi)}\right)^2\mathrm{d}\xi = s(t)^2\sigma(t)^2, \tag{26}$$

where $s(t)$ and $\sigma(t)$ is detailed in Eq. (7). Hence, the perturbation kernel can be rewritten as:

$$p_{0t}\big(\boldsymbol{x}(t) \mid \boldsymbol{x}(0), \boldsymbol{\mu}\big) = \mathcal{N}\left(\boldsymbol{x}(t); \boldsymbol{\mu} + s(t)\left(\boldsymbol{x}\left(0\right) - \boldsymbol{\mu}\right), s(t)^2\sigma(t)^2\boldsymbol{I}\right) \tag{27}$$

$$= s(t)^{-d}\mathcal{N}\left(\frac{\boldsymbol{x}(t)}{s(t)}; \boldsymbol{x}(0) + \frac{1 - s(t)}{s(t)}\boldsymbol{\mu}, \sigma(t)^2\boldsymbol{I}\right) \tag{28}$$

$$= s(t)^{-d}\tilde{p}_{0t}\left(\tilde{\boldsymbol{x}}(t) \mid \tilde{\boldsymbol{x}}_0(t)\right), \tag{29}$$

where d is the dimension of $\boldsymbol{x}$, $\tilde{\boldsymbol{x}}(t)$ is equal to $\dfrac{\boldsymbol{x}(t)}{s(t)}$, and $\tilde{\boldsymbol{x}}_0(t)$ along with $\tilde{p}_{0t}$ is defined in Eqs. (5) and (6). Eq. (29) is the same as Eq. (4).

A.2. Backward Process

As we have mentioned in Sec. 3.1, our forward SDE in Eq. (3) can be viewed as a special case of Eq. (1) proposed by [48], by defining $\boldsymbol{f}(x, t) = f(t)(\boldsymbol{x} - \boldsymbol{\mu})$. Thus, the backward ODE can also be seen as a special case of Eq. (2). By substituting the relationship between $\boldsymbol{f}(x, t)$ and $f(t)$ into Eq. (2), we can acquire:

$$\mathrm{d}\boldsymbol{x} = \left[f(t)(\boldsymbol{x} - \boldsymbol{\mu}) - \frac{1}{2}g(t)^2\nabla_{\boldsymbol{x}}\log p_t(\boldsymbol{x})\right]\mathrm{d}t, \tag{30}$$

where we simplify $\boldsymbol{x}(t)$ to $\boldsymbol{x}$. According to Eq. (7), we can derive the relationship between $s(t)$, $\sigma(t)$ and $f(t)$,$g(t)$. This has already been demonstrated in the Eqs. (28) and (34) in [21], which is

$$f(t) = \frac{\dot{s}(t)}{s(t)}, g(t) = s(t)\sqrt{2\dot{\sigma}(t)\sigma(t)}, \tag{31}$$

where $\dot{s}(t)$ and $\dot{\sigma}(t)$ are the derivatives of $s(t)$ and $\sigma(t)$, respectively. We can rewrite the form of Eq. (30) by substituting Eq. (31) into it:

$$\mathrm{d}\boldsymbol{x} = \left[\frac{\dot{s}(t)}{s(t)}(\boldsymbol{x} - \boldsymbol{\mu}) - s(t)^2\dot{\sigma}(t)\sigma(t)\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})\right] \mathrm{d}t. \tag{32}$$

Since we define $\tilde{\boldsymbol{x}}(t) = \dfrac{\boldsymbol{x}(t)}{s(t)}$. We can obtain

$$\boldsymbol{x}(t) = s(t)\tilde{\boldsymbol{x}}(t). \tag{33}$$

We can differentiate both sides of Eq. (33):

$$\frac{\mathrm{d}s(t)}{\mathrm{d}t}\tilde{\boldsymbol{x}}(t) + s(t)\frac{\mathrm{d}\tilde{\boldsymbol{x}}(t)}{\mathrm{d}t} = \frac{\mathrm{d}\boldsymbol{x}(t)}{\mathrm{d}t}, \tag{34}$$

$$\dot{s}(t)\tilde{\boldsymbol{x}}(t)\mathrm{d}t + s(t)\mathrm{d}\tilde{\boldsymbol{x}}(t) = \mathrm{d}\boldsymbol{x}(t). \tag{35}$$

Substitute Eq. (35) and Eq. (33) into Eq. (32):

$$\dot{s}(t)\tilde{\boldsymbol{x}}(t)\mathrm{d}t + s(t)\mathrm{d}\tilde{\boldsymbol{x}}(t) = \left[\frac{\dot{s}(t)}{s(t)}(s(t)\tilde{\boldsymbol{x}}(t) - \boldsymbol{\mu}) - s(t)^2\dot{\sigma}(t)\sigma(t)\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})\right] \mathrm{d}t, \tag{36}$$

$$s(t)\mathrm{d}\tilde{\boldsymbol{x}}(t) = \left[-\frac{\dot{s}(t)}{s(t)}\boldsymbol{\mu} - s(t)^2\dot{\sigma}(t)\sigma(t)\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})\right] \mathrm{d}t, \tag{37}$$

$$\mathrm{d}\tilde{\boldsymbol{x}}(t) = \left[-\frac{\dot{s}(t)}{s(t)^2}\boldsymbol{\mu} - s(t)\dot{\sigma}(t)\sigma(t)\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})\right] \mathrm{d}t, \tag{38}$$

The term $\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})$ is the score function, which is predicted by the denoiser D_θ mentioned in Sec. 3.2. However, we aim to use $\tilde{\boldsymbol{x}}(t)$ rather than $\boldsymbol{x}(t)$ as the input of D_θ. Hence, the relationship between $\nabla_{\tilde{\boldsymbol{x}}(t)} \log p_t\big(\tilde{\boldsymbol{x}}(t)\big)$ and $\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})$ should be clarified. This is demonstrated as follows:

$$\nabla_{\tilde{\boldsymbol{x}}(t)} \log p_t\big(\tilde{\boldsymbol{x}}(t)\big) = \nabla_{\boldsymbol{x}(t)/s(t)} \log\left[s(t)^{-d} p_t\left(\frac{\boldsymbol{x}(t)}{s(t)}\right)\right] \tag{39}$$

$$= s(t)\nabla_{\boldsymbol{x}(t)} \log\left[p_t\big(\boldsymbol{x}(t)\big)\right]. \tag{40}$$

Eq. (40) is based on $p_t\big(\boldsymbol{x}(t)\big) = s(t)^{-d} p_t\left(\dfrac{\boldsymbol{x}(t)}{s(t)}\right)$, which can be derived the same as Eq. (29). Eq. (40) can be substituted into Eq. (38):

$$\mathrm{d}\tilde{\boldsymbol{x}}(t) = \left[-\frac{\dot{s}(t)}{s(t)^2}\boldsymbol{\mu} - \dot{\sigma}(t)\sigma(t)\nabla_{\tilde{\boldsymbol{x}}(t)} \log p_t\big(\tilde{\boldsymbol{x}}(t)\big)\right] \mathrm{d}t, \tag{41}$$

which aligns with Eq. (8), with $\nabla_{\tilde{\boldsymbol{x}}(t)} \log p_t\big(\tilde{\boldsymbol{x}}(t)\big) = s_\theta\big(\tilde{\boldsymbol{x}}(t)\big)$.

Next, we illuminate the relationship between $\nabla_{\tilde{\boldsymbol{x}}(t)} \log p_t\big(\tilde{\boldsymbol{x}}(t)\big)$ and the output of D_θ. Therefore, we can directly use the output of D_θ within the sampling process. Generally, we hope that when D_θ is trained to be ideal, the discrepancy between the predicted distribution and the target distribution of $\tilde{\boldsymbol{x}}(t)$ is minimized. This can be achieved using the score matching method [17, 48]. Specifically, we regulate the score function calculated from the output of D_θ to match the theoretical target score function. In other words, the training goal is to let $\nabla_{\tilde{\boldsymbol{x}}(t)} \log q_t\big(\tilde{\boldsymbol{x}}(t)\big) = \nabla_{\tilde{\boldsymbol{x}}(t)} \log p_t\big(\tilde{\boldsymbol{x}}(t)\big)$, where we denote the target score function as $\nabla_{\tilde{\boldsymbol{x}}(t)} \log q_t\big(\tilde{\boldsymbol{x}}(t)\big)$ and the target distribution of $\tilde{\boldsymbol{x}}(t)$ in the sampling process as $q_t\big(\tilde{\boldsymbol{x}}(t)\big)$. Since the

integrals of $q_t\big(\tilde{\boldsymbol{x}}(t)\big)$ and $p_t\big(\tilde{\boldsymbol{x}}(t)\big)$ over the domain of $\tilde{\boldsymbol{x}}(t)$ are both equal to one, $q_t\big(\tilde{\boldsymbol{x}}(t)\big) = p_t\big(\tilde{\boldsymbol{x}}(t)\big)$ can be derived from $\nabla_{\tilde{\boldsymbol{x}}(t)} \log q_t\big(\tilde{\boldsymbol{x}}(t)\big) = \nabla_{\tilde{\boldsymbol{x}}(t)} \log p_t\big(\tilde{\boldsymbol{x}}(t)\big)$. The training goal can be achieved by optimizing the Fisher divergence [18, 38], which is indicated by D_F. Assuming we are at diffusion step t, D_F is given by:

$$D_F\left(q_t\big(\tilde{\boldsymbol{x}}(t)\big) \parallel p_t\big(\tilde{\boldsymbol{x}}(t)\big)\right) = \mathbb{E}_{q_t(\tilde{\boldsymbol{x}}(t))}\left[\frac{1}{2}\left\|\nabla_{\tilde{\boldsymbol{x}}(t)} \log p_t\big(\tilde{\boldsymbol{x}}(t)\big) - \nabla_{\tilde{\boldsymbol{x}}(t)} \log q_t\big(\tilde{\boldsymbol{x}}(t)\big)\right\|^2\right]. \tag{42}$$

Thereby, we aim to demonstrate that optimizing Eq. (42) is theoretically equivalent to optimizing our practical loss function $L\big(D_\theta, \sigma(t)\big)$ in Eq. (9). Therefore, we can use Eq. (9) instead of Fisher divergence. We select the training objective in Eq. (9) to align with current generative DMs [14, 21, 42], given that this objective has been proven effective [21]. [51] proposes another elegant and scalable form of Eq. (42):

$$D_F\left(q_t\big(\tilde{\boldsymbol{x}}(t)\big) \parallel p_t\big(\tilde{\boldsymbol{x}}(t)\big)\right) = \mathbb{E}_{q_t(\tilde{\boldsymbol{x}}(t),\tilde{\boldsymbol{x}}_0(t))}\left[\frac{1}{2}\left\|\nabla_{\tilde{\boldsymbol{x}}(t)} \log p_t\big(\tilde{\boldsymbol{x}}(t)\big) - \nabla_{\tilde{\boldsymbol{x}}(t)} \log q_t\big(\tilde{\boldsymbol{x}}(t) \mid \tilde{\boldsymbol{x}}_0(t)\big)\right\|^2\right] + \text{const}, \tag{43}$$

where const is a constant, and $\mathbb{E}_{q_t(\tilde{\boldsymbol{x}}(t),\tilde{\boldsymbol{x}}_0(t))}$ is the expectation of the joint distribution of $\tilde{\boldsymbol{x}}(t)$ and $\tilde{\boldsymbol{x}}_0(t)$. Here, $q_t\big(\tilde{\boldsymbol{x}}(t) \mid \tilde{\boldsymbol{x}}_0(t)\big)$ represents the conditional pdf of $\tilde{\boldsymbol{x}}(t)$ given $\tilde{\boldsymbol{x}}_0(t)$. As we have the relationship between $\tilde{\boldsymbol{x}}(t)$ and $\tilde{\boldsymbol{x}}_0(t)$, the concrete form of $\nabla_{\tilde{\boldsymbol{x}}(t)} \log q_t\big(\tilde{\boldsymbol{x}}(t) \mid \tilde{\boldsymbol{x}}_0(t)\big)$ can derived as:

$$\begin{aligned}
&\nabla_{\tilde{\boldsymbol{x}}(t)} \log q_t\big(\tilde{\boldsymbol{x}}(t) \mid \tilde{\boldsymbol{x}}_0(t)\big) && (44)\\
&= \nabla_{\tilde{\boldsymbol{x}}(t)} \log \mathcal{N}\left(\tilde{\boldsymbol{x}}(t); \tilde{\boldsymbol{x}}_0(t), \sigma(t)^2\boldsymbol{I}\right) && (45)\\
&= \nabla_{\tilde{\boldsymbol{x}}(t)} \log \left[(2\pi)^{-\frac{d}{2}}\left(\det\left(\sigma(t)^2\boldsymbol{I}\right)\right)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\big(\tilde{\boldsymbol{x}}(t) - \tilde{\boldsymbol{x}}_0(t)\big)^T\big(\sigma(t)^2\boldsymbol{I}\big)^{-1}\big(\tilde{\boldsymbol{x}}(t) - \tilde{\boldsymbol{x}}_0(t)\big)\right)\right] && (46)\\
&= \nabla_{\tilde{\boldsymbol{x}}(t)}\left(-\frac{1}{2}\big(\tilde{\boldsymbol{x}}(t) - \tilde{\boldsymbol{x}}_0(t)\big)^T\big(\sigma(t)^2\boldsymbol{I}\big)^{-1}\big(\tilde{\boldsymbol{x}}(t) - \tilde{\boldsymbol{x}}_0(t)\big)\right) && (47)\\
&= -\frac{\tilde{\boldsymbol{x}}(t) - \tilde{\boldsymbol{x}}_0(t)}{\sigma(t)^2}, && (48)
\end{aligned}$$

which, along with $\nabla_{\tilde{\boldsymbol{x}}(t)} \log p_t\big(\tilde{\boldsymbol{x}}(t)\big) = s_\theta\big(\tilde{\boldsymbol{x}}(t)\big)$ and Eq. (6), can be substituted into Eq. (43):

$$\begin{aligned}
D_F\left(q_t\big(\tilde{\boldsymbol{x}}(t)\big) \parallel p_t\big(\tilde{\boldsymbol{x}}(t)\big)\right) &= \mathbb{E}_{q_t(\tilde{\boldsymbol{x}}(t),\tilde{\boldsymbol{x}}_0(t))}\left[\frac{1}{2}\left\|s_\theta\big(\tilde{\boldsymbol{x}}(t)\big) + \frac{\tilde{\boldsymbol{x}}(t) - \tilde{\boldsymbol{x}}_0(t)}{\sigma(t)^2}\right\|^2\right] + \text{const} && (49)\\
&= \mathbb{E}_{q_t(\tilde{\boldsymbol{x}}(t),\tilde{\boldsymbol{x}}_0(t))}\left[\frac{1}{2}\left\|s_\theta\big(\tilde{\boldsymbol{x}}(t)\big) + \frac{\tilde{\boldsymbol{x}}(t) - \boldsymbol{x}(0) - \frac{1-s(t)}{s(t)}\boldsymbol{\mu}}{\sigma(t)^2}\right\|^2\right] + \text{const} && (50)\\
&= \frac{1}{2}\mathbb{E}_{q_t(\tilde{\boldsymbol{x}}(t),\tilde{\boldsymbol{x}}_0(t))}\left[\frac{1}{\sigma(t)^4}\left\|\sigma(t)^2 s_\theta\big(\tilde{\boldsymbol{x}}(t)\big) + \tilde{\boldsymbol{x}}(t) - \frac{1-s(t)}{s(t)}\boldsymbol{\mu} - \boldsymbol{x}(0)\right\|^2\right] + \text{const}. && (51)
\end{aligned}$$

To achieve the alignment between the optimization results of Eq. (51) and the training objective in Eq. (9), we can unify the forms of the two objectives. Concretely, if we let

$$\sigma(t)^2 s_\theta\big(\tilde{\boldsymbol{x}}(t)\big) + \tilde{\boldsymbol{x}}(t) - \frac{1-s(t)}{s(t)}\boldsymbol{\mu} = D_\theta\big(\tilde{\boldsymbol{x}}(t); \sigma(t); c\big), \tag{52}$$

then we obtain:

$$s_\theta\big(\tilde{\boldsymbol{x}}(t)\big) = \frac{1}{\sigma(t)^2}\left(D_\theta\big(\tilde{\boldsymbol{x}}(t); \sigma(t); c\big) + \frac{1-s(t)}{s(t)}\boldsymbol{\mu} - \tilde{\boldsymbol{x}}(t)\right), \tag{53}$$

which formally establishes the relationship between the score function $s_\theta\big(\tilde{\boldsymbol{x}}(t)\big)$ and the denoiser output $D_\theta\big(\tilde{\boldsymbol{x}}(t); \sigma(t); c\big)$, the same as Eq. (10). We can substitute Eq. (52) into Eq. (51):

$$D_F\left(q_t\big(\tilde{\boldsymbol{x}}(t)\big) \parallel p_t\big(\tilde{\boldsymbol{x}}(t)\big)\right) = \frac{1}{2}\mathbb{E}_{q_t(\tilde{\boldsymbol{x}}(t),\tilde{\boldsymbol{x}}_0(t))}\left[\frac{1}{\sigma(t)^4}\left\|D_\theta\big(\tilde{\boldsymbol{x}}(t); \sigma(t); c\big) - \boldsymbol{x}(0)\right\|^2\right] + \text{const}. \tag{54}$$

Given that $q_t\left(\tilde{\boldsymbol{x}}(t), \tilde{\boldsymbol{x}}_0(t)\right) = q_t\big(\tilde{\boldsymbol{x}}(t) \mid \tilde{\boldsymbol{x}}_0(t)\big) q_t\big(\tilde{\boldsymbol{x}}_0(t)\big)$, we can acquire $\mathbb{E}_{q_t(\tilde{\boldsymbol{x}}(t), \tilde{\boldsymbol{x}}_0(t))}[\cdot] = \mathbb{E}_{q_t(\tilde{\boldsymbol{x}}_0(t))} \mathbb{E}_{q_t(\tilde{\boldsymbol{x}}(t) \mid \tilde{\boldsymbol{x}}_0(t))}[\cdot]$. According to Eq. (6), $\tilde{\boldsymbol{x}}_0(t)$ depends entirely on $\boldsymbol{x}(0)$, $\boldsymbol{\mu}$ and $s(t)$. At any fixed diffusion step t, $s(t)$ is a specific determined value. Furthermore, $\boldsymbol{x}(0)$ and $\boldsymbol{\mu}$ are drawn from the data distribution. Thus, we can denote the distribution of $\tilde{\boldsymbol{x}}_0(t)$ as p_{data}, as indicated in Eq. (9). As for $q_t\big(\tilde{\boldsymbol{x}}(t) \mid \tilde{\boldsymbol{x}}_0(t)\big)$, according to Eq. (5), $\tilde{\boldsymbol{x}}(t)$ equals $\tilde{\boldsymbol{x}}_0(t) + \boldsymbol{n}$, where $\boldsymbol{n} \sim \mathcal{N}\left(0, \sigma(t)^2 \boldsymbol{I}\right)$. Hence, given $\tilde{\boldsymbol{x}}_0(t)$, $\tilde{\boldsymbol{x}}(t) \sim \mathcal{N}\left(\tilde{\boldsymbol{x}}_0(t), \sigma(t)^2 \boldsymbol{I}\right)$. Based on the aforementioned analysis, we can rewrite Eq. (54) as:

$$D_F\left(q_t\big(\tilde{\boldsymbol{x}}(t)\big) \parallel p_t\big(\tilde{\boldsymbol{x}}(t)\big)\right) = \frac{1}{2} \mathbb{E}_{\tilde{\boldsymbol{x}}_0(t) \sim p_{\text{data}}} \mathbb{E}_{\tilde{\boldsymbol{x}}(t) \sim \mathcal{N}(\tilde{\boldsymbol{x}}_0(t), \sigma(t)^2 \boldsymbol{I})} \left[\frac{1}{\sigma(t)^4} \left\| D_\theta\big(\tilde{\boldsymbol{x}}(t); \sigma(t); c\big) - \boldsymbol{x}(0) \right\|^2\right] + \text{const} \tag{55}$$

$$= \frac{1}{2} \mathbb{E}_{\tilde{\boldsymbol{x}}_0(t) \sim p_{\text{data}}} \mathbb{E}_{\boldsymbol{n} \sim \mathcal{N}(0, \sigma(t)^2 \boldsymbol{I})} \left[\frac{1}{\sigma(t)^4} \left\| D_\theta\big(\tilde{\boldsymbol{x}}_0(t) + \boldsymbol{n}; \sigma(t); c\big) - \boldsymbol{x}(0) \right\|^2\right] + \text{const}, \tag{56}$$

which aligns with the practical training objective in Eq. (9), as $\boldsymbol{x}(0) = \tilde{\boldsymbol{x}}_0(0)$, differing only by the coefficients $\frac{1}{2}$ and $\frac{1}{\sigma(t)^2}$. Note that the coefficients $\frac{1}{2}$ and $\frac{1}{\sigma(t)^2}$ both remain fixed at any given t. Consequently, at diffusion step t, optimizing $D_F\left(q_t\big(\tilde{\boldsymbol{x}}(t)\big) \parallel p_t\big(\tilde{\boldsymbol{x}}(t)\big)\right)$ is theoretically equivalent to optimizing $L\big(D_\theta, \sigma(t)\big)$ in Eq. (9), enabling us to directly use $L\big(D_\theta, \sigma(t)\big)$ rather than $D_F\left(q_t\big(\tilde{\boldsymbol{x}}(t)\big) \parallel p_t\big(\tilde{\boldsymbol{x}}(t)\big)\right)$ as the training objective.

By substituting Eq. (53) into Eq. (41), we obtain the ODE in Eq. (11), which is practically used in our sampling process.

A.3. Preconditioning

In this proof, we use t to represent the diffusion step and l to denote the time or the time point, in order to distinguish between these two key concepts. Note that l is an integer, while t is continuous. Substituting Eq. (13) into Eq. (18) yields:

$$\mathcal{L} = \mathbb{E}_{\sigma, \tilde{\boldsymbol{x}}_0(t), \boldsymbol{n}} \left[\lambda(\sigma) \left\| \text{mean}\left(\left\{c_{\text{skip}}(\sigma)\, \tilde{\boldsymbol{x}}^l(t)\right\}_{l=1}^{L}\right) + c_{\text{out}}(\sigma)\, F_\theta - \boldsymbol{x}(0) \right\|_2^2\right], \tag{57}$$

$$= \mathbb{E}_{\sigma, \tilde{\boldsymbol{x}}_0(t), \boldsymbol{n}} \left[\lambda(\sigma) \left\| \text{mean}\left(\left\{c_{\text{skip}}(\sigma) \left(\boldsymbol{x}(0) + \frac{1-s}{s} \boldsymbol{\mu}^l + \boldsymbol{n}^l\right)\right\}_{l=1}^{L}\right) + c_{\text{out}}(\sigma) F_\theta - \boldsymbol{x}(0) \right\|_2^2\right], \tag{58}$$

$$= \mathbb{E}\left[\underbrace{\lambda(\sigma)\, c_{\text{out}}(\sigma)^2}_{\text{effective weight}} \left\| \underbrace{F_\theta}_{\text{network output}} - \underbrace{\frac{1}{c_{\text{out}}(\sigma)} \left(\boldsymbol{x}(0) - \text{mean}\left(\left\{c_{\text{skip}}(\sigma) \left(\boldsymbol{x}(0) + \frac{1-s}{s} \boldsymbol{\mu}^l + \boldsymbol{n}^l\right)\right\}_{l=1}^{L}\right)\right)}_{\text{effective training target}} \right\|_2^2\right], \tag{59}$$

where we omit the bracketed arguments in the functional notations $s(t)$, $\sigma(t)$ and $F_\theta\left(\left\{c_{\text{in}}(\sigma)\, \tilde{\boldsymbol{x}}^l(t)\right\}_{l=1}^{L}; c_{\text{noise}}(\sigma); c\right)$ for notational simplicity. The $\mathbb{E}_{\sigma, \tilde{\boldsymbol{x}}_0(t), \boldsymbol{n}}$ is simplified to $\mathbb{E}$ in Eq. (59). Note that while we have different corrupted images $\boldsymbol{\mu}^l$ across various time points, there is only a single target $\boldsymbol{x}(0)$.

Adhering to the EDM framework [21], we impose a variance normalization constraint on the training inputs of $F_\theta(\cdot)$, enforcing unit variance preservation at each temporal point l:

$$\text{Var}_{\boldsymbol{x}(0), \boldsymbol{\mu}^l, \boldsymbol{n}^l}\left[c_{\text{in}}(\sigma) \left(\boldsymbol{x}(0) + \frac{1-s}{s} \boldsymbol{\mu}^l + \boldsymbol{n}^l\right)\right] = 1, \tag{60}$$

$$c_{\text{in}}(\sigma)^2\, \text{Var}_{\boldsymbol{x}(0), \boldsymbol{\mu}^l, \boldsymbol{n}^l}\left(\boldsymbol{x}(0) + \frac{1-s}{s} \boldsymbol{\mu}^l + \boldsymbol{n}^l\right) = 1, \tag{61}$$

Thus,

$$c_{\text{in}}(\sigma) = \sqrt{\frac{1}{\text{Var}_{\boldsymbol{x}(0), \boldsymbol{\mu}^l, \boldsymbol{n}^l}\left(\boldsymbol{x}(0) + \frac{1-s}{s} \boldsymbol{\mu}^l + \boldsymbol{n}^l\right)}}, \tag{62}$$

where $\boldsymbol{n}^l$ is independent of $\boldsymbol{x}(0)$ and $\boldsymbol{x}(0) + \frac{1-s}{s}\boldsymbol{\mu}^l$. However, $\boldsymbol{x}(0)$ and $\boldsymbol{x}(0) + \frac{1-s}{s}\boldsymbol{\mu}^l$ are obviously not independent. Hence, we can calculate the variance of $\boldsymbol{x}(0) + \frac{1-s}{s}\boldsymbol{\mu}^l + \boldsymbol{n}^l$:

$$\mathrm{Var}_{\boldsymbol{x}(0),\boldsymbol{\mu}^l,\boldsymbol{n}^l}\left(\boldsymbol{x}(0) + \frac{1-s}{s}\boldsymbol{\mu}^l + \boldsymbol{n}^l\right) \tag{63}$$

$$= \mathrm{Var}_{\boldsymbol{x}(0),\boldsymbol{\mu}^l,\boldsymbol{n}^l}\left(\boldsymbol{x}(0) + \frac{1-s}{s}\boldsymbol{\mu}^l\right) + \mathrm{Var}_{\boldsymbol{n}^l}(\boldsymbol{n}^l) \tag{64}$$

$$= \mathrm{Var}_{\boldsymbol{x}(0)}\left(\boldsymbol{x}\left(0\right)\right) + \mathrm{Var}_{\boldsymbol{\mu}^l}\left(\frac{1-s}{s}\boldsymbol{\mu}^l\right) + 2\mathrm{Cov}\left(\boldsymbol{x}\left(0\right), \frac{1-s}{s}\boldsymbol{\mu}^l\right) + \mathrm{Var}_{\boldsymbol{n}^l}(\boldsymbol{n}^l) \tag{65}$$

$$= \mathrm{Var}_{\boldsymbol{x}(0)}\left(\boldsymbol{x}\left(0\right)\right) + \left(\frac{1-s}{s}\right)^2 \mathrm{Var}_{\boldsymbol{\mu}^l}\left(\boldsymbol{\mu}^l\right) + 2\frac{1-s}{s}\mathrm{Cov}\left(\boldsymbol{x}\left(0\right), \boldsymbol{\mu}^l\right) + \mathrm{Var}_{\boldsymbol{n}^l}(\boldsymbol{n}^l), \tag{66}$$

where $\mathrm{Cov}\left(\boldsymbol{x}\left(0\right), \frac{1-s}{s}\boldsymbol{\mu}^l\right)$ is the covariance of $\boldsymbol{x}\left(0\right)$ and $\frac{1-s}{s}\boldsymbol{\mu}^l$. Since $\boldsymbol{n}^l$ is drawn from $\mathcal{N}(0, \sigma^2\boldsymbol{I})$, its variance $\mathrm{Var}_{\boldsymbol{n}^l}(\boldsymbol{n}^l)$ is equal to σ^2. We denote $\mathrm{Var}_{\boldsymbol{x}(0)}\left(\boldsymbol{x}\left(0\right)\right)$ as σ^2_{data}. For simplicity in derivation, we assume:

Assumption A.1. *The variance of corrupted images at different time points remains constant,* i.e. $\forall l \in [1, L], Var_{\boldsymbol{\mu}^l}\left(\boldsymbol{\mu}^l\right) = \sigma^2_{mu}$.

Assumption A.2. *The covariance between corrupted images at different time points and the target image* $\boldsymbol{x}(0)$ *remains constant,* i.e. $\forall l \in [1, L], Cov\left(\boldsymbol{x}\left(0\right), \boldsymbol{\mu}^l\right) = \sigma_{cov}$.

Under the two assumptions, we can simplify Eq. (66) into

$$\mathrm{Var}_{\boldsymbol{x}(0),\boldsymbol{\mu}^l,\boldsymbol{n}^l}\left(\boldsymbol{x}(0) + \frac{1-s}{s}\boldsymbol{\mu}^l + \boldsymbol{n}^l\right) = \sigma^2_{\text{data}} + \left(\frac{1-s}{s}\right)^2 \sigma^2_{\text{mu}} + 2\left(\frac{1-s}{s}\right)\sigma_{\text{cov}} + \sigma^2. \tag{67}$$

According to Eq. (62) and Eq. (67), we can get the value of $c_{\text{in}}\left(\sigma\right)$ as

$$c_{\text{in}}\left(\sigma\right) = \frac{1}{\sqrt{\sigma^2_{\text{data}} + \left(\frac{1-s}{s}\right)^2 \sigma^2_{\text{mu}} + 2\left(\frac{1-s}{s}\right)\sigma_{\text{cov}} + \sigma^2}}. \tag{68}$$

Eq. (68) is the same as Eq. (14), if denoting $k = \frac{1-s}{s}$.

Following EDM [21], we rigorously enforce unit variance normalization on the effective training target in Eq. (59):

$$\mathrm{Var}_{\boldsymbol{x}(0),\boldsymbol{\mu}^l,\boldsymbol{n}^l}\left[\frac{1}{c_{\text{out}}\left(\sigma\right)}\left(\boldsymbol{x}(0) - \text{mean}\left(\left\{c_{\text{skip}}\left(\sigma\right)\left(\boldsymbol{x}(0) + \frac{1-s}{s}\boldsymbol{\mu}^l + \boldsymbol{n}^l\right)\right\}_{l=1}^{L}\right)\right)\right] = 1, \tag{69}$$

which leads to

$$c_{\text{out}}\left(\sigma\right)^2 = \mathrm{Var}_{\boldsymbol{x}(0),\boldsymbol{\mu}^l,\boldsymbol{n}^l}\left[\boldsymbol{x}(0) - \frac{c_{\text{skip}}\left(\sigma\right)}{L}\sum_{l=1}^{L}\left(\boldsymbol{x}(0) + \frac{1-s}{s}\boldsymbol{\mu}^l + \boldsymbol{n}^l\right)\right], \tag{70}$$

$$c_{\text{out}}\left(\sigma\right)^2 = \mathrm{Var}_{\boldsymbol{x}(0),\boldsymbol{\mu}^l,\boldsymbol{n}^l}\left[\left(1 - c_{\text{skip}}\left(\sigma\right)\right)\boldsymbol{x}(0) - \left(\frac{1-s}{s}\right)\frac{c_{\text{skip}}\left(\sigma\right)}{L}\sum_{l=1}^{L}\boldsymbol{\mu}^l - \frac{c_{\text{skip}}\left(\sigma\right)}{L}\sum_{l=1}^{L}\boldsymbol{n}^l\right], \tag{71}$$

where $\boldsymbol{n}^l$ is independent of both $\boldsymbol{x}(0)$ and $\boldsymbol{\mu}^l$, and it is also independent across different time points. Therefore,

$$c_{\text{out}}(\sigma)^2 = \text{Var}_{\boldsymbol{x}(0),\boldsymbol{\mu}^l}\left[\big(1-c_{\text{skip}}(\sigma)\big)\boldsymbol{x}(0) - \left(\frac{1-s}{s}\right)\frac{c_{\text{skip}}(\sigma)}{L}\sum_{l=1}^{L}\boldsymbol{\mu}^l\right] + \left(\frac{c_{\text{skip}}(\sigma)}{L}\right)^2 \text{Var}_{\boldsymbol{n}^l}\left(\sum_{l=1}^{L}\boldsymbol{n}^l\right), \tag{72}$$

$$c_{\text{out}}(\sigma)^2 = \text{Var}_{\boldsymbol{x}(0),\boldsymbol{\mu}^l}\left[\big(1-c_{\text{skip}}(\sigma)\big)\boldsymbol{x}(0) - \left(\frac{1-s}{s}\right)\frac{c_{\text{skip}}(\sigma)}{L}\sum_{l=1}^{L}\boldsymbol{\mu}^l\right] + \left(\frac{c_{\text{skip}}(\sigma)}{L}\right)^2 \sum_{l=1}^{L}\left(\text{Var}_{\boldsymbol{n}^l}\boldsymbol{n}^l\right), \tag{73}$$

$$c_{\text{out}}(\sigma)^2 = \text{Var}_{\boldsymbol{x}(0),\boldsymbol{\mu}^l}\left[\big(1-c_{\text{skip}}(\sigma)\big)\boldsymbol{x}(0) - \left(\frac{1-s}{s}\right)\frac{c_{\text{skip}}(\sigma)}{L}\sum_{l=1}^{L}\boldsymbol{\mu}^l\right] + \frac{c_{\text{skip}}(\sigma)^2}{L}\sigma^2. \tag{74}$$

Note that

$$\text{Var}_{\boldsymbol{x}(0),\boldsymbol{\mu}^l}\left[\big(1-c_{\text{skip}}(\sigma)\big)\boldsymbol{x}(0) - \left(\frac{1-s}{s}\right)\frac{c_{\text{skip}}(\sigma)}{L}\sum_{l=1}^{L}\boldsymbol{\mu}^l\right] \tag{75}$$

$$= \big(1-c_{\text{skip}}(\sigma)\big)^2\sigma_{\text{data}}^2 + \left(\frac{1-s}{s}\right)^2\left(\frac{c_{\text{skip}}(\sigma)}{L}\right)^2 \text{Var}_{\boldsymbol{\mu}^l}\left(\sum_{l=1}^{L}\boldsymbol{\mu}^l\right) \tag{76}$$

$$- 2\big(1-c_{\text{skip}}(\sigma)\big)\frac{1-s}{s}\frac{c_{\text{skip}}(\sigma)}{L}\text{Cov}\left((\boldsymbol{x}(0)), \sum_{l=1}^{L}\boldsymbol{\mu}^l\right). \tag{77}$$

We make another assumption for further derivations, as follows:

Assumption A.3. *The corrupted images exhibit complete mutual dependence across all time points,* i.e. $Var_{\boldsymbol{\mu}^l}\left(\sum_{l=1}^{L}\boldsymbol{\mu}^l\right) = Var_{\boldsymbol{\mu}^l}\left(L\mu^l\right) = L^2\sigma_{mu}^2$.

While this assumption is simplistic, as corrupted images at different times are not identical, it remains a valuable approximation for our derivation. This is because images corrupted at different time points can still exhibit significant similarity. The ablation experiments in Sec. 4.3 further demonstrate the effectiveness of the preconditioning method based on this assumption. Using our three assumptions and Eq. (77), we can derive:

$$\text{Var}_{\boldsymbol{x}(0),\boldsymbol{\mu}^l}\left[\big(1-c_{\text{skip}}(\sigma)\big)\boldsymbol{x}(0) - \left(\frac{1-s}{s}\right)\frac{c_{\text{skip}}(\sigma)}{L}\sum_{l=1}^{L}\boldsymbol{\mu}^l\right] \tag{78}$$

$$= \big(1-c_{\text{skip}}(\sigma)\big)^2\sigma_{\text{data}}^2 + \left(\frac{1-s}{s}\right)^2\left(\frac{c_{\text{skip}}(\sigma)}{L}\right)^2 L^2\sigma_{\text{mu}}^2 - 2\big(1-c_{\text{skip}}(\sigma)\big)\frac{1-s}{s}\frac{c_{\text{skip}}(\sigma)}{L}L\sigma_{\text{cov}} \tag{79}$$

$$= \big(1-c_{\text{skip}}(\sigma)\big)^2\sigma_{\text{data}}^2 + \left(\frac{1-s}{s}\right)^2 c_{\text{skip}}(\sigma)^2\sigma_{\text{mu}}^2 - 2\big(1-c_{\text{skip}}(\sigma)\big)c_{\text{skip}}(\sigma)\frac{1-s}{s}\sigma_{\text{cov}}. \tag{80}$$

Substitute Eq. (80) into Eq. (74), as follows:

$$c_{\text{out}}(\sigma)^2 = \big(1-c_{\text{skip}}(\sigma)\big)^2\sigma_{\text{data}}^2 + \left(\frac{1-s}{s}\right)^2 c_{\text{skip}}(\sigma)^2\sigma_{\text{mu}}^2 - 2\big(1-c_{\text{skip}}(\sigma)\big)c_{\text{skip}}(\sigma)\frac{1-s}{s}\sigma_{\text{cov}} + \frac{c_{\text{skip}}(\sigma)^2}{L}\sigma^2. \tag{81}$$

Following EDM [21], we then obtain the optimal $c_{\text{skip}}(\sigma)$ by minimizing $c_{\text{out}}(\sigma)$, so that the errors of F_θ can be amplified as little as possible. This is expressed as:

$$c_{\text{skip}}(\sigma) = \arg\min_{c_{\text{skip}}(\sigma)} c_{\text{out}}(\sigma) = \arg\min_{c_{\text{skip}}(\sigma)} c_{\text{out}}(\sigma)^2, \tag{82}$$

which is obtained by selecting $c_{\text{out}}(\sigma) \geq 0$, without loss of generality. To solve the optimal problem in Eq. (82), we set the

derivative w.r.t. $c_{\text{skip}}(\sigma)$ to zero:

$$0 = \frac{\mathrm{d}c_{\text{out}}(\sigma)^2}{\mathrm{d}c_{\text{skip}}(\sigma)}, \tag{83}$$

$$0 = \frac{\mathrm{d}\left[\left(1-c_{\text{skip}}(\sigma)\right)^2\sigma_{\text{data}}^2 + \left(\frac{1-s}{s}\right)^2 c_{\text{skip}}(\sigma)^2\sigma_{\text{mu}}^2 - 2\left(1-c_{\text{skip}}(\sigma)\right)c_{\text{skip}}(\sigma)\frac{1-s}{s}\sigma_{\text{cov}} + \frac{c_{\text{skip}}(\sigma)^2}{L}\sigma^2\right]}{\mathrm{d}c_{\text{skip}}(\sigma)}, \tag{84}$$

$$0 = \left[\sigma_{\text{data}}^2 + \left(\frac{1-s}{s}\right)^2\sigma_{\text{mu}}^2 + \frac{\sigma^2}{L} + 2\frac{1-s}{s}\sigma_{\text{cov}}\right]c_{\text{skip}}(\sigma) - \left(\sigma_{\text{data}}^2 + \frac{1-s}{s}\sigma_{\text{cov}}\right). \tag{85}$$

Thus, we can acquire the value of $c_{\text{skip}}(\sigma)$:

$$c_{\text{skip}}(\sigma) = \frac{\sigma_{\text{data}}^2 + \frac{1-s}{s}\sigma_{\text{cov}}}{\sigma_{\text{data}}^2 + \left(\frac{1-s}{s}\right)^2\sigma_{\text{mu}}^2 + \frac{\sigma^2}{L} + 2\frac{1-s}{s}\sigma_{\text{cov}}}, \tag{86}$$

which aligns with Eq. (15) with $k = \frac{1-s}{s}$.

By substituting Eq. (86) into Eq. (81), we can attain the value of $c_{\text{out}}(\sigma)$:

$$c_{\text{out}}(\sigma) = \sqrt{\frac{\left(\frac{1-s}{s}\right)^2\sigma_{\text{mu}}^2\sigma_{\text{data}}^2 + \frac{\sigma^2}{L}\sigma_{\text{data}}^2 - \left(\frac{1-s}{s}\right)^2\sigma_{\text{cov}}^2}{\sigma_{\text{data}}^2 + \left(\frac{1-s}{s}\right)^2\sigma_{\text{mu}}^2 + \frac{\sigma^2}{L} + 2\frac{1-s}{s}\sigma_{\text{cov}}}}, \tag{87}$$

which is the same as Eq. (16) since $k = \frac{1-s}{s}$.

The value of $c_{\text{noise}}(\sigma)$ is the same as that in EDM [21], which is obtained based on experiments:

$$c_{\text{noise}}(\sigma) = \frac{1}{4}\ln(\sigma). \tag{88}$$

A.4. Sampling

We present a detailed pseudocode for our stochastic sampler with arbitrary $s(t)$ and $\sigma(t)$ in Algorithm 3, which can be regarded as an extension of Algorithm 2. In Algorithm 3, we individually sample the initial states, *i.e.* $\boldsymbol{x}_0^l$, at each time point, from line 2 to line 3. Notably, The corrupted images $\boldsymbol{\mu}^l$ differ across different time points. In other words, $\boldsymbol{\mu}^{l_1} \neq \boldsymbol{\mu}^{l_2}$ if $l_1 \neq l_2$ and $l_1, l_2 \in [1, L]$. From line 4 to line 15, we loop N times to denoise $\{\boldsymbol{x}_0^l\}_{l=1}^L$. Specifically, from line 5 to line 8, we compute the value of γ_i, and γ_i is used in line 9 to increase the noise level by adjusting t_i to $\hat{t}_i$. Lines 11 to 12 involve performing stochastic perturbation on $\boldsymbol{x}_0^l$ at each time point l, using Eq. (19). In line 14, we use Eq. (11) to evaluate $\frac{\mathrm{d}\tilde{\boldsymbol{x}}(t)}{\mathrm{d}t}$ at diffusion step $\hat{t}_i$ and time point l. The denoiser D_θ takes images from all time points, *i.e.* $\left\{\hat{\boldsymbol{x}}_i^l\right\}_{l=1}^L$, as its input, since it can denoise sequential images in parallel as discussed in Sec. 3.3. By integrating information across time points, D_θ achieves improved results, aided by the TFSA module discussed in Sec. 3.3. We then apply an Euler step in line 15 to calculate the next-step image $\boldsymbol{x}_{i+1}^l$. Finally, we use a mean operator to reduce the temporal dimension of $\left\{\hat{\boldsymbol{x}}_N^l\right\}_{l=1}^L$, where $\left\{\hat{\boldsymbol{x}}_N^l\right\}_{l=1}^L \in \mathbb{R}^{L\times C\times H\times W}$ and $\boldsymbol{x}_N \in \mathbb{R}^{C\times H\times W}$, omitting batch size for clarity.

In Algorithm 3, there are seven key hyperparameters: N, S_{tmin}, S_{tmax}, S_{noise}, S_{churn}, σ_{min}, and σ_{max}, as mentioned in Sec. 3.5. Here we add some details. The S_{tmin} and S_{tmax} define the range for the stochastic sampling steps. Concretely, as

Algorithm 3 Our stochastic sampler with arbitrary $s(t)$ and $\sigma(t)$.

1: **procedure** StochasticSampler($D_\theta, \{\boldsymbol{\mu}^l\}_{l=1}^L, c$)
2: **for** $l \in \{1, 2, \cdots, L\}$ **do** ▷ Individually sample the initial state for L time points
3: **sample** $\boldsymbol{x}_0^l \sim \mathcal{N}(\frac{1-s(t_0)}{s(t_0)}\boldsymbol{\mu}^l, \sigma(t_0)^2\boldsymbol{I})$ ▷ $\boldsymbol{x}_0^l$ is a noisy corrupted image
4: **for** $i \in \{0, 1, \cdots, N-1\}$ **do** ▷ Repeat the sampling step N times
5: **if** $t_i \in [S_{\text{tmin}}, S_{\text{tmax}}]$ **then** ▷ $[S_{\text{tmin}}, S_{\text{tmax}}]$ define the stochastic sampling range
6: $\gamma_i \leftarrow \frac{S_{\text{churn}}}{N}$ ▷ S_{churn} and N determine γ_i
7: **else** ▷ For t_i outside the range $[S_{\text{tmin}}, S_{\text{tmax}}]$, use deterministic sampling
8: $\gamma_i \leftarrow 0$ ▷ Setting $\gamma_i = 0$ leads to deterministic sampling
9: $\hat{t}_i \leftarrow t_i + \gamma_i t_i$ ▷ γ_i regulates the extent of stochastic perturbation
10: **for** $l \in \{1, 2, \cdots, L\}$ **do** ▷ Individually perform denoising for L time points
11: **sample** $\boldsymbol{\epsilon}_i \in \mathcal{N}\left(0, S_{\text{noise}}^2\boldsymbol{I}\right)$ ▷ Sample the noise for stochastic perturbation
12: $\hat{\boldsymbol{x}}_i^l \leftarrow \boldsymbol{x}_i^l + \left(\frac{1-s(\hat{t})}{s(\hat{t}_i)} - \frac{1-s(t_i)}{s(t_i)}\right)\boldsymbol{\mu}^l + \sqrt{\sigma(\hat{t}_i)^2 - \sigma(t_i)^2}\boldsymbol{\epsilon}_i$ ▷ Use Eq. (19) for stochastic perturbation
13: **for** $l \in \{1, 2, \cdots, L\}$ **do** ▷ Individually take Euler step for L time points
14: $\boldsymbol{d}_i^l \leftarrow -\frac{\dot{s}(\hat{t}_i)}{s(\hat{t}_i)^2}\boldsymbol{\mu} - \frac{\dot{\sigma}(\hat{t}_i)}{\sigma(\hat{t}_i)}\left[D_\theta\left(\left\{\hat{\boldsymbol{x}}_i^l\right\}_{l=1}^L; \sigma(\hat{t}_i); c\right) + \frac{1-s(\hat{t}_i)}{s(\hat{t}_i)}\boldsymbol{\mu} - \hat{\boldsymbol{x}}_i^l\right]$ ▷ Use Eq. (11)
15: $\boldsymbol{x}_{i+1}^l \leftarrow \hat{\boldsymbol{x}}_i^l + (t_{i+1} - \hat{t}_i)\boldsymbol{d}_i^l$ ▷ Take an Euler step from $\hat{t}_i$ to t_{i+1}
16: $\boldsymbol{x}_N = \text{mean}\left(\{\boldsymbol{x}_N^l\}_{l=1}^L\right)$ ▷ Use the mean operator to collapse the temporal dimension and calculate the final result
17: **return** $\boldsymbol{x}_N$ ▷ The result is a single restored image

shown from line 5 to line 8, if t_i falls outside $[S_{\text{tmin}}, S_{\text{tmax}}]$, γ_i is set to 0. As a result, $\hat{t}_i$ is set to t_i (line 9), leading to $\hat{\boldsymbol{x}}_i^l = \boldsymbol{x}_i^l$, thus reducing the stochastic sampler to its deterministic counterpart. If t_i is within $[S_{\text{tmin}}, S_{\text{tmax}}]$, regular stochastic sampling occurs. S_{churn}, along with N, controls the value of γ_i in line 6, influencing the extent of stochastic perturbation in line 12. This approach is improved from the stochastic sampler in EDM [21] by removing the γ_i upper limit ($\sqrt{2}-1$ in EDM). Since our method yields larger γ_i due to small N, removing this limit can prevent restricting randomness. The effectiveness of this modification is demonstrated in Sec. 4.3.

B. Detailed Related Work

In Sec. 2, we provided a brief overview of related work. Here, we offer a more comprehensive introduction.

B.1. Cloud Removal

Traditional Methods. Traditional CR methods, with the use of mathematical transform [15, 56], physical principles [52, 55], information cloning [29, 43], offer great interpretability. However, they tend to underperform in comparison to deep learning techniques, which limits their practical applications.

GAN-based Methods. Current deep learning-based CR methods primarily use GANs, with cGANs [37] and Pix2Pix [19] as the vanilla paradigm. In CR tasks [1, 9, 12], both cloudy images and noise are fed into the generator to produce a cloudless image. The ground truth or predicted cloudless images, along with the cloudy image, are fed into the discriminator, which determines whether the input includes the ground truth image. Through adversarial training, the generator learns to produce nearly real cloudless images. To improve cGANs for CR tasks, SpA GAN [40] introduces a Spatial Attentive Network (SPANet) that incorporates a spatial attention mechanism in its generator to improve CR performance. The Simulation-Fusion GAN [10] further improves CR performance by integrating SAR images. It operates in two stages: first, it employs a specific convolutional neural network (CNN) to convert SAR images into optical images; then, it fuses the simulated optical images, SAR images, and original cloudy optical images using a GAN-based framework to reconstruct the corrupted regions. TransGAN-CFR [26] proposes an innovative transformer-based generator with a hierarchical encoder-decoder network. This design includes transformer blocks [50] using a non-overlapping window multi-head self-attention (WMSA) mechanism and

Table 5. Details of our best training and testing configurations.

	CUHK-CR1	**CUHK-CR2**	**SEN12MS-CR**	**Sen2_MTC_New**
Parameters	39.13M	39.13M	39.13M	148.88M
Training Steps	22,500	26,300	446,700	64,141
Training Epochs	500	470	46	500
Batch Size	4	2	2	8
Precision	tf32	tf32	tf32	tf32
Training Hardware	3 RTX 3090	4 RTX 4090	4 RTX 4090	4 RTX 4090
In Channels	8 (= 4 + 0 + 4)	8 (= 4 + 0 + 4)	28 (= 13 + 2 + 13)	7 (= 3 + 1 + 3)
Out Channels	4	4	13	3
Patch Size	1	1	1	4
Levels (Local + Global Attention)	2 + 2	2 + 2	2 + 2	2 + 1
Depth	[2, 2, 2, 2]	[2, 2, 2, 2]	[2, 2, 2, 2]	[2, 2, 16]
Widths	[128, 256, 384, 768]	[128, 256, 384, 768]	[128, 256, 384, 768]	[256, 512, 768]
FFN Intermediate Widths	[256, 512, 768, 1536]	[256, 512, 768, 1536]	[256, 512, 768, 1536]	[512, 1024, 1536]
Attention Heads (Width / Head Dim)	[2, 4, 6, 12]	[2, 4, 6, 12]	[2, 4, 6, 12]	[4, 8, 12]
Attention Head Dim	64	64	64	64
Neighborhood Kernel Size	7	7	7	7
Dropout Rate	[0.0, 0.0, 0.0, 0.1]	[0.0, 0.0, 0.0, 0.1]	[0.0, 0.0, 0.0, 0.1]	[0.0, 0.0, 0.0, 0.0]
Mapping Depth	2	2	2	2
Mapping Width	768	768	768	768
Mapping FFN Intermediate Width	1536	1536	1536	1536
Mapping Dropout Rate	0.1	0.1	0.1	0.1
α	3.0	3.0	3.0	3.0
σ_{data}	1.0	1.0	1.0	1.0
σ_{mu}	1.0	1.0	1.0	1.0
σ_{cov}	0.9	0.9	0.9	0.9
P_{mean} in Algorithm 1	-1.4	-1.2	-1.2	-1.4
P_{std} in Algorithm 1	1.4	1.2	1.2	1.4
Optimizer	AdamW	AdamW	AdamW	AdamW
Learning Rate	1e-4	1e-4	1e-4	1e-4
Betas	[0.9, 0.999]	[0.9, 0.999]	[0.9, 0.999]	[0.9, 0.999]
Eps	1e-8	1e-8	1e-8	1e-8
Weight Decay	1e-2	1e-2	1e-2	1e-2
EMA Decay	0.9999	0.9999	0.9999	0.9999
Sampling Steps N	4	4	5	5
σ_{min}	0.001	0.001	0.001	0.001
σ_{max}	100	100	100	100
S_{churn}	0.1	2.5	5.0	1.0
S_{noise}	0.995	1.0	1.023	1.0
S_{tmin}	0.0	0.0	0.0	0.0
S_{tmax}	100000000	100000000	100000000	100.0

a modified feed-forward network (FFN). SAR images are also integrated with cloudy images in this network, and a new triplet loss is introduced to improve CR capabilities.

DM-based Methods. Diffusion Models (DMs), a new type of generative model, have outperformed GANs in image generation tasks [4] and shown potential in image restoration tasks [27], including CR. Current diffusion-based CR methods mostly adhere to the basic DM framework. Concretely, DDPM-CR [20] leverages the DDPM [14] architecture to integrate both cloudy optical images and SAR images to extract DDPM features. The features are then used for cloud removal in the cloud removal head. DiffCR [64] introduces an efficient time and condition fusion block (TCFBlock) for building the denoising network and a decoupled encoder for extracting features from conditional images (*e.g.* SAR images) to guide the

DM generation process. SeqDM [62] is designed for multi-temporal CR tasks. It comprises a new sequential-based training and inference strategy (SeqTIS) that processes sequential images in parallel. It also extends vanilla DMs to multi-modal diffusion models (MmDMs) for incorporating the additional information from auxiliary modalities (*e.g*. SAR images).

Non-Generative Methods. Some non-generative methods have also been proposed for CR, serving as alternatives to GAN-based and DM-based methods. DSen2-CR [36] employs a super-resolution ResNet [25, 28] and can function as a multi-modal model as it can process optical images and SAR images together by concatenating them as inputs. GLF-CR [54], another multi-modal model, introduces a global-local fusion network to use the additional SAR information. Specifically, it is a dual-stream network where SAR image information is hierarchically integrated into feature maps to address cloud-corrupted areas, using global fusion for relationships among local windows and local fusion to transfer SAR features. UnCRtainTS [8] is designed for both multi-temporal and mono-temporal CR tasks. It includes an encoder for all time points, an attention-based temporal aggregator for fusing sequential observations, and a mono-temporal decoder. The model incorporates multivariate uncertainty quantification to enhance CR capabilities. The version with uncertainty quantification is called UnCRtainTS σ, as shown in Tab. 1, while the one with simple L2 loss is named UnCRtainTS L2, as shown in Fig. 4.

B.2. Diffusion Models

Generative DMs DMs are initially applied to image generation. The vanilla DM, known as DDPM, is proposed by [14]. Concurrently, Song *et al*. propose NCSN [47], a generative model similar to DDPM, by estimating gradients of the data distribution. Song *et al*. further clarify the underlying principles of DMs using score matching methods [48], unifying DDPM as the VP condition and NCSN as the VE condition. EDM [21] criticizes that the theory and practice of conventional generative DMs [48] are unnecessarily complex and simplify DMs by presenting a clear design space to separate the design choices of various modules, integrating both VP and VE DMs. They also redesign most key modules within their EDM to further enhance the generation abilities. Additional improvements include faster sampling [32, 33], new denoising networks [2, 41], and adjusted training loss weights [13]. Our denoising network is based on HDiT [2], which employs a scalable hourglass transformer as the denoising network, effectively generating high-quality images in the pixel space.

Restoration DMs Building on the success of DMs in image generation, researchers have investigated their application in image restoration [27]. The restoration DM can be categorized into supervised and zero-shot learning methods, as discussed in Sec. 2. The first type is more relevant to our work, as our method adopts the supervised learning paradigm. Early supervised methods condition DMs on low-quality reference images by simply concatenating them with noise as the input to the denoising network, as demonstrated in SR3 [45] and Palette [44]. Later improvements focus on conditioning the models on pre-processed reference images and features, as seen in CDPMSR [39] and IDM [11]. A significant advancement comes from methods that modify the diffusion process itself to incorporate conditions. Specifically, IR-SDE [34] introduces a mean-reverting SDE to define the forward process and derives the corresponding backward SDE, enabling generation from noisy corrupted images rather than pure noise and leading to improved restoration results. Refusion [35] enhances this approach by optimizing network architecture, incorporating VAE [22] for image compression, *etc*. ResShift [59] and RDDM [31] both adopt the DDPM framework (*i.e*. the VP condition). Similar to IR-SDE, they modify the forward process to incorporate both noise and residuals, facilitating diffusion from target images to noisy corrupted images. Notably, within the backward process, ResShift uses a single denoising network, while RDDM employs separate networks to predict noise and residuals. Similar strategies have also been employed by InDI [3], I2SB [30], *etc*.

C. Experiments

C.1. Implementation Details

C.1.1. Datasets

The CUHK-CR1 and CUHK-CR2 datasets, introduced by [49], consist of images captured by the Jilin-1 satellite with a size of 512×512. CUHK-CR1 contains 668 images of thin clouds, while CUHK-CR2 includes 559 images of thick clouds. These two datasets collectively form the CUHK-CR dataset. With an ultra-high spatial resolution of 0.5 m, the images encompass four bands: RGB and near-infrared (NIR). Following [49], the CUHK-CR1 dataset is split into 534 training and 134 testing images, while CUHK-CR2 is divided into 448 training and 111 testing images. The images are in PNG format, with integer values in the range $[0, 255]$.

The SEN12MS-CR dataset, introduced by [6], contains coregistered multi-spectral optical images with 13 bands from Sentinel-2 satellite and SAR images with 2 bands from Sentinel-1 satellite. Collected from 169 non-overlapping regions of interest (ROIs) across continents, each averaging approximately 52×40 km^2 in size, the scenes of ROIs are divided into 256×256 pixel patches, with 50% spatial overlap. We use 114,050 images for training, 7,176 images for validation, and

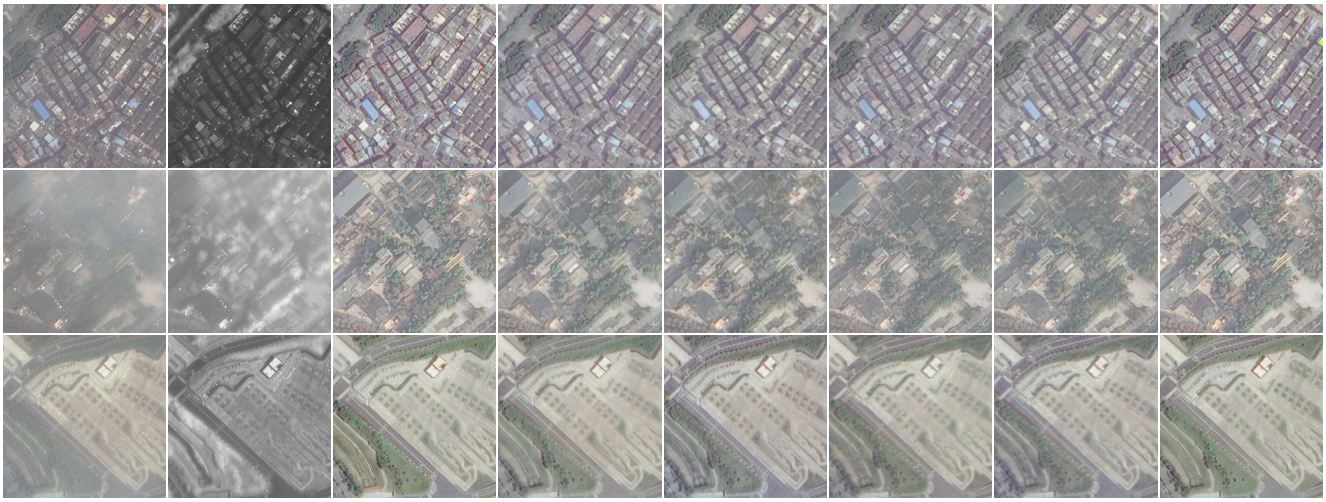

Figure 7. Additional visual results on the CUHK-CR1 dataset.

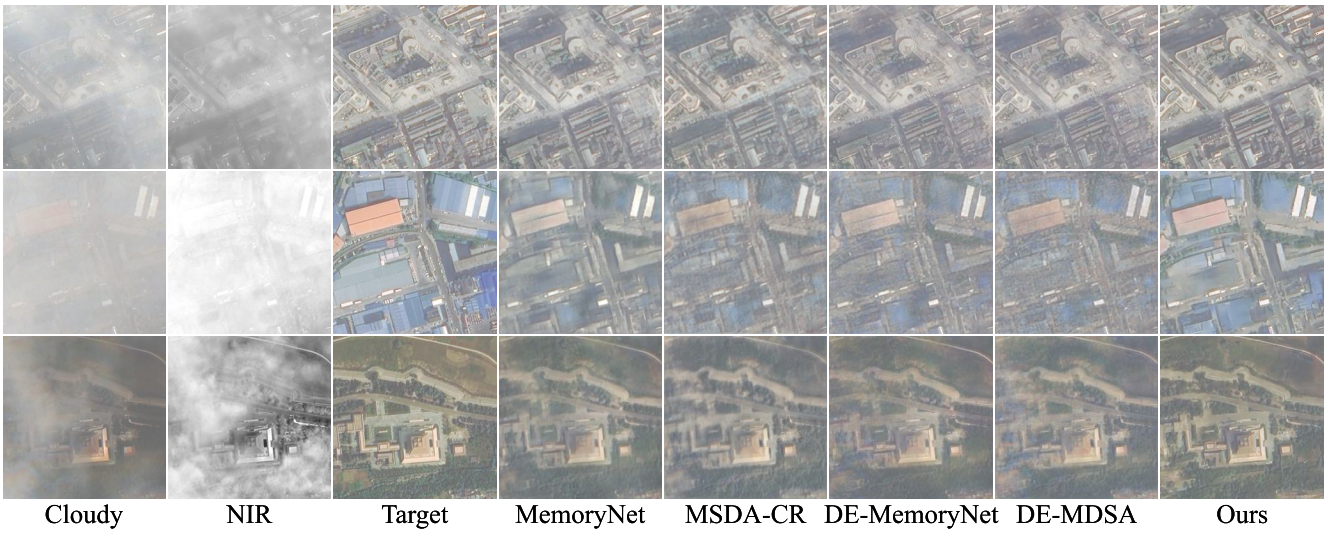

Figure 8. Additional visual results on the CUHK-CR2 dataset.

7,899 images for testing. The dataset split follows previous works [6, 8].

The Sen2_MTC_New dataset, introduced by [16], consists of coregistered RGB and IR images across approximately 50 non-overlapping tiles. Each tile includes around 70 pairs of cropped 256×256 pixel patches with pixel values ranging from 0 to $10,000$. Following [16], the dataset is divided into $2,380$ images for training, 350 for validation, and 687 for testing.

C.1.2. Pre-Processing

As with common deep learning methods, images must be pre-processed before being input into our neural network. Given that datasets vary in their characteristics, we apply distinct pre-processing techniques to each one, following established practices. Below, we provide a detailed explanation.

The CUHK-CR1 and CUHK-CR2 datasets. Following [49], we resize images from 512×512 pixels to 256×256 pixels. Subsequently, the pixel values are rescaled to a range of $[-1, 1]$.

The Sen2_MTC_New dataset. Following [16], the pixel values of images are initially scaled to the $[0, 1]$ range by dividing by $10,000$, then normalized using a mean of 0.5 and a standard deviation of 0.5. For the training split, data augmentation includes random flips and a 90-degree rotation every four images.

The SEN12MS-CR dataset. Following [7], the pixel values of SAR and optical images are clipped to the ranges of $[-25, 0]$

Figure 9. Additional visual results on the SEN12MS-CR dataset. As GLF-CR [54] can only process 128×128 images, unlike others (256×256), we divide each image into four parts, process them individually, and merge the results. Optical image brightness is linearly enhanced for visualization.

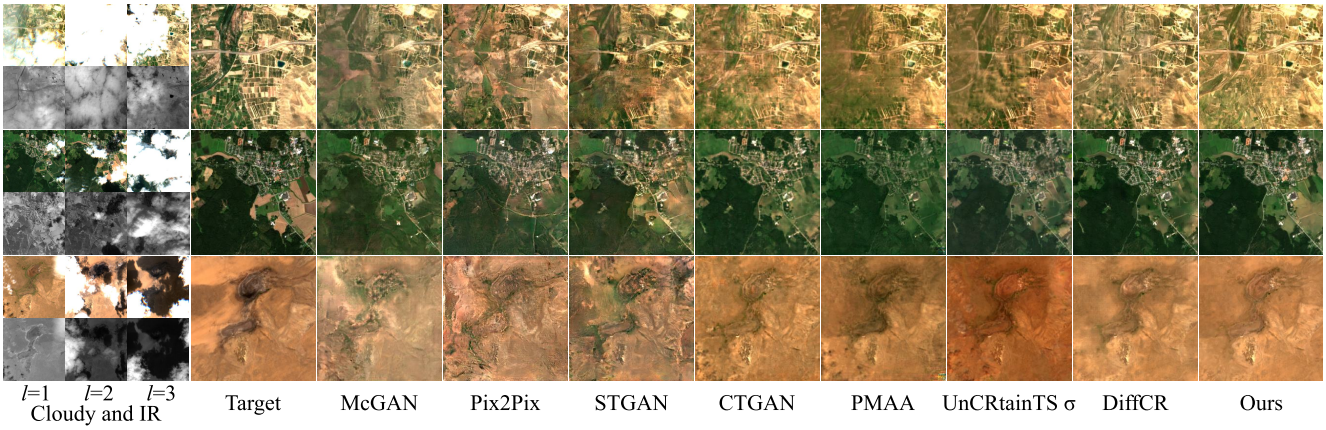

Figure 10. Additional visual results on the Sen2_MTC_New dataset.

and $[0, 10000]$, respectively. However, we rescale the pixel values of all images to the range of $[-1, 1]$ to achieve centrosymmetric pixel values, which is different from [7].

C.1.3. Configuration

The optimal configuration is detailed in Tab. 5. The number of input channels is the sum of channels from noisy corrupted images, auxiliary modal images, and original corrupted images, as shown in Fig. 3. The table lists these channels as (input noisy corrupted image channels + input auxiliary modal image channels + input original corrupted image channels). For example, in the *Input Channels* row in Tab. 5, $28(= 13 + 2 + 13)$ means that the noisy corrupted image has 13 channels, the auxiliary modal image has 2 channels and the original corrupted image has 13 channels. Notably, in CUHK-CR1 and CUHK-CR2 datasets, we reconstruct RGB and NIR channels following established methods, incorporating the NIR channel into the noisy corrupted image input rather than treated as auxiliary data. Consequently, the auxiliary modal image channel count for these datasets is zero.

C.1.4. Evaluation Metrics in Theory

To comprehensively evaluate the performance, we employ multiple metrics including peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM) [53], mean absolute error (MAE), spectral angle mapper (SAM) [24], and learned

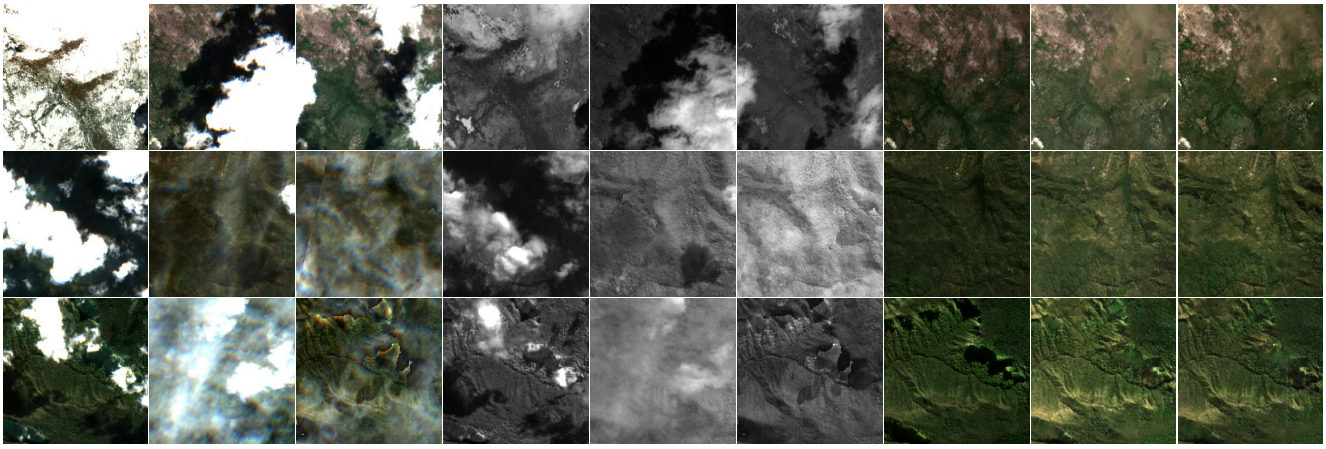

Figure 11. Visual results generated by the stochastic sampler and the deterministic sampler. For the deterministic sampler, we set $N = 5$, $\sigma_{\text{min}} = 0.001$ and $\sigma_{\text{max}} = 100$. For the stochastic sampler, we set $N = 5$, $\sigma_{\text{min}} = 0.001$, $\sigma_{\text{max}} = 100$, $S_{\text{churn}} = 1.0$, $S_{\text{noise}} = 1.0$, $S_{\text{tmin}} = 0$ and $S_{\text{tmax}} = 100$.

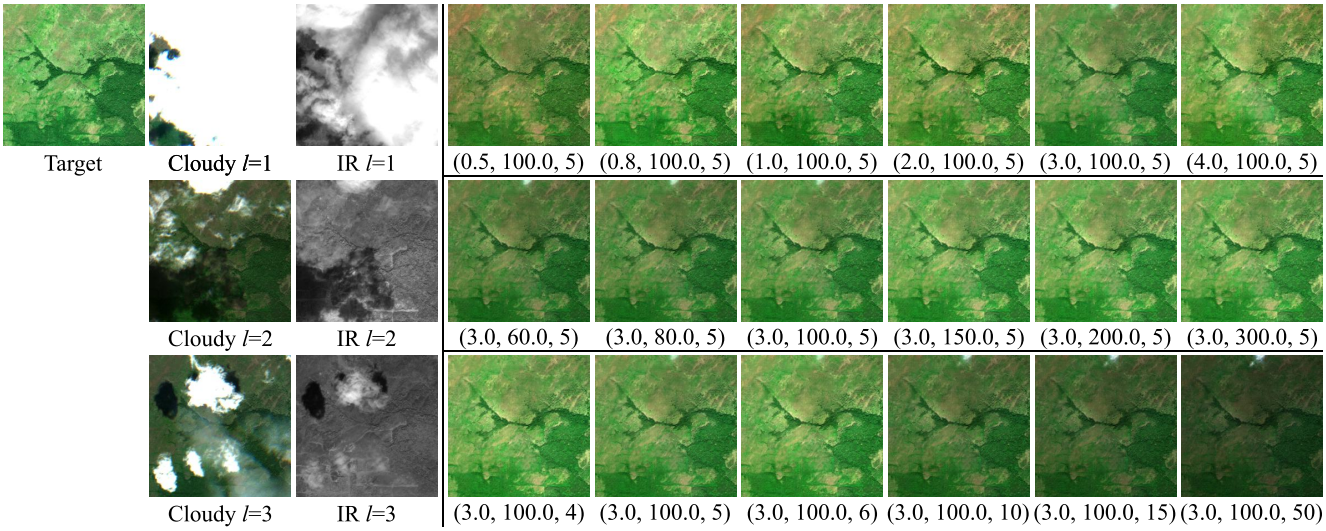

Figure 12. Visual results under different configurations of (α, σ_{max}, N). For example, $(3.0, 100.0, 5)$ represents the restored results with $\alpha = 3.0$, $\sigma_{\text{max}} = 100.0$ and $N = 5$.

perceptual image patch similarity (LPIPS) [60]. The precise computational formulations of these metrics are as follows:

$$\text{PSNR}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = 20 \log_{10} \left(\frac{1}{\text{RMSE}(\boldsymbol{y}, \hat{\boldsymbol{y}})} \right), \tag{89}$$

$$\text{SSIM}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{(2\mu_{\boldsymbol{y}}\mu_{\hat{\boldsymbol{y}}} + c_1)(2\sigma_{\boldsymbol{y}\hat{\boldsymbol{y}}} + c_2)}{\left(\mu_{\boldsymbol{y}}^2 + \mu_{\hat{\boldsymbol{y}}}^2 + c_1\right)\left(\sigma_{\boldsymbol{y}}^2 + \sigma_{\hat{\boldsymbol{y}}}^2 + c_2\right)}, \tag{90}$$

$$\text{MAE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{C \cdot H \cdot W} \sum_{c=1}^{C} \sum_{h=1}^{H} \sum_{w=1}^{W} |\boldsymbol{y}_{c,h,w} - \hat{\boldsymbol{y}}_{c,h,w}|, \tag{91}$$

$$\text{SAM}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \cos^{-1} \left(\frac{\sum_{c=1}^{C} \sum_{h=1}^{H} \sum_{w=1}^{W} \boldsymbol{y}_{c,h,w} \cdot \hat{\boldsymbol{y}}_{c,h,w}}{\sqrt{\sum_{c=1}^{C} \sum_{h=1}^{H} \sum_{w=1}^{W} \boldsymbol{y}_{c,h,w}^2 \cdot \sum_{c=1}^{C} \sum_{h=1}^{H} \sum_{w=1}^{W} \hat{\boldsymbol{y}}_{c,h,w}^2}} \right), \tag{92}$$

$$\text{LPIPS}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \sum_{i} \frac{1}{H_i \cdot W_i} \sum_{h=1}^{H} \sum_{w=1}^{W} \left\| w_i \odot \left(\hat{\boldsymbol{y}}_{h,w}^{i} - \boldsymbol{y}_{h,w}^{i} \right) \right\|_2^2 \tag{93}$$

where

$$\mathrm{RMSE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \sqrt{\frac{1}{C \cdot H \cdot W} \sum_{c=1}^{C} \sum_{h=1}^{H} \sum_{w=1}^{W} \left(\boldsymbol{y}_{c,h,w} - \hat{\boldsymbol{y}}_{c,h,w}\right)^2}. \tag{94}$$

Here, we denote the predicted image as $\hat{\boldsymbol{y}}$ and the ground truth image as $\boldsymbol{y}$. with channel number C, height H and width W. The notation $\boldsymbol{y}_{c,h,w}$ and $\hat{\boldsymbol{y}}_{c,h,w}$ refers to a specific pixel in $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$, indicated by subscript c, h, w. In Eq. (90), $\mu_{\boldsymbol{y}}$ and $\mu_{\hat{\boldsymbol{y}}}$ represent the means, and $\sigma_{\boldsymbol{y}}$ and $\sigma_{\hat{\boldsymbol{y}}}$ are the standard deviations of $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$, respectively. The covariance is symbolized by $\sigma_{\boldsymbol{y}\hat{\boldsymbol{y}}}$. The constants c_1 and c_2 stabilize the calculations. To compute LPIPS [60], a pre-trained network $\mathcal{F}$ processes $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$ to derive intermediate embeddings across multiple layers. The activations are normalized, scaled by a vector w, and the L2 distance between embeddings of $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$ is calculated and averaged over spatial dimensions and layers as the final LPIPS value, as shown in Eq. (93). In Eq. (93), i indicates the layer of $\mathcal{F}$, with H_i, W_i, and w_i being the height, width, and scaling factor at the layer i. The embeddings at the position (h, w) and the layer i are denoted as $\hat{\boldsymbol{y}}^i_{h,w}$ and $\boldsymbol{y}^i_{h,w}$. We use the official implementations of [60] to calculate the value of LPIPS.

C.1.5. Evaluation Metrics in Practice

Although the theoretical methods for these evaluation metrics are consistent across datasets, practical calculations may vary due to pre-processing, post-processing, *etc*. To ensure a fair comparison, we apply different computing methods for each dataset, in line with prior research. Detailed explanations for each dataset are provided here.

The CUHK-CR1 and CUHK-CR2 datasets. Following [49], we scale the pixel values of the restored and ground truth images, *i.e*. $\hat{\boldsymbol{y}}$ and $\boldsymbol{y}$, to the range $[0, 255]$, and clamp any out-of-range values. These pixel values are then converted to unsigned integers. PSNR is calculated using all channels, while SSIM and LPIPS are first calculated for each channel and then averaged. To calculate LPIPS, we employ a pre-trained AlexNet [23] as $\mathcal{F}$.

The Sen2_MTC_New dataset. We adopt the DiffCR [64] approach by rescaling the pixel values of the restored and ground truth images to the range $[0, 1000]$, clipping values outside $[0, 2000]$, and then rescaling back to $[0, 1]$. These processed images are used to compute PSNR and SSIM across all channels. For LPIPS, the input images are further rescaled to $[-1, 1]$ and processed using a pre-trained AlexNet [23] as $\mathcal{F}$.

The SEN12MS-CR dataset. All the images are rescaled to $[0, 1]$. Then, the rescaled images are used to compute PSNR, SSIM, MAE, and SAM, with all channels used.

C.1.6. Reproducing Details

For closed-source methods, we use the metric values they report. In contrast, for certain open-source methods, we implement the algorithms ourselves and present visual results in Fig. 4. When implementing previous methods, if pre-trained weights are available, we directly use them; otherwise, we retrain the models from scratch. Below, we briefly outline the implementation details of the reproduced methods.

The CUHK-CR1 and CUHK-CR2 datasets. The CUHK-CR1 and CUHK-CR2 datasets are relatively new, with limited prior research [49]. The authors evaluate five existing methods: SpA-GAN [40], AMGAN-CR [57], CVAE [5], MemoryNet [61], and MSDA-CR [58], alongside their proposed methods, DE-MemoryNet and DE-MSDA [49], on these two dataset. In [49], metrics for all methods are reported, with pre-trained weights provided only for MemoryNet and MSDA-CR. Consequently, we use these weights and retrain DE-MemoryNet and DE-MSDA to present visual results in Fig. 4. DE-MSDA is excluded from Fig. 4 as it performs worse than DE-MemoryNet, despite being introduced in the same study.

The SEN12MS-CR dataset. As McGAN [9] and SpA GAN [40] do not have pre-trained weights for this dataset, we retrain them and present the visual results in Fig. 4. In contrast, pre-trained weights for DSen2-CR [36], GLF-CR [54], and UnCRtainTS [8] are available and have also been used for visualization in Fig. 4. Notably, GLF-CR [54] operates on 128×128 images, while other methods use 256×256 images. To ensure consistency, we divide each image into four segments, process them independently, and subsequently merge them for visualization, as shown in Fig. 4. The performance metrics for all previous methods on this dataset are cited from [8] and [64].

The Sen2_MTC_New dataset. Metrics values are cited from [16], [63], and [64]. We retrain McGAN [9], Pix2Pix [19], STGAN [46] and UnCRtainTS [8], while using pre-trained weights of CTGAN [16], PMAA [63], and DiffCR [64] for visualization in Fig. 4.

C.2. Efficiency Analysis

We first present a comparative analysis of parameter counts (*Params*) and multiply-accumulate operations (*MACs*) of our proposed method against recent state-of-the-art approaches. in Tab. 6 across the four datasets. Our analysis excludes early methods due to their significantly inferior performance compared to EMRDM and the unavailability or irreproducibility of

Table 6. The Comparison of Params (the number of parameters) and MACs (multiply-accumulate operations).

(a) SEN12MS-CR	GLF-CR	UnCRtainTS L2	DiffCR	EMRDM
Params (M)	14.827	0.519	22.96	39.13
MACs (G)	245.28	28.02	29.37	83.57

(b) CUHK-CR	MemoryNet	MSDA-CR	DE	EMRDM
Params (M)	3.64	3.91	36.80	39.13
MACs (G)	548.65	53.45	199.15	83.33

(c) Sen2_MTC_New	STGAN	CTGAN	CR-TS Net	PMAA	UnCRtainTS	DDPM-CR	DiffCR	EMRDM
Params (M)	231.93	642.92	38.68	3.45	0.56	445.44	22.91	148.88
MACs (G)	1094.94	632.05	7602.97	92.35	37.16	852.37	45.86	74.39

their detailed implementations. All *MACs* are computed with a batch size of 1 and an input image resolution of 256×256 to ensure fair comparisons. It should be noted that although GLF-CR [54] typically operates on 128×128 resolution images, we evaluated it at 256×256 resolution for efficiency analysis to maintain consistency across comparisons. Moreover, for DiffCR, which lacks official implementation details for the SEN12MS-CR dataset, we reproduce it on this dataset based on the description outlined in [64] and report the corresponding *Params* and *MACs* in Tab. 6. The entries labeled "DE" in Tab. 6 denote DE-MemoryNet and DE-MSDA [49], which share identical *Params* and *MACs*. The results of efficiency analysis demonstrate that EMRDM achieves performance gains with reasonable increments in *Params* and *MACs*, particularly for mono-temporal tasks. While multi-temporal tasks necessitate additional parameters of EMRDM to effectively model complex temporal dependencies in image sequences, the corresponding *MACs* remain within reasonable bounds for real-world applications.

We further analyzed the training and sampling time of EMRDM across the four datasets. For standardization, we use the configurations in Tab. 5 and measured training time per batch with batch size unchanged and sampling time per image with batch size changed to 1. All experiments are conducted on a single NVIDIA RTX 4090 GPU to ensure fair comparisons. Per-batch training times measure 1,410.5 ms (CUHK-CR1), 1,237.7 ms (CUHK-CR2), 1,230.5 ms (SEN12MS-CR), and 204.7 ms (Sen2_MTC_New), with per-image sampling times of 131.2 ms (CUHK-CR1), 128.0 ms (CUHK-CR2), 136.4 ms (SEN12MS-CR), and 173.1 ms (Sen2_MTC_New). These timing measurements are hardware-dependent and may fluctuate. Hence, we report only mean values. Notable, sampling time is particularly significant since training occurs only once, while sampling is performed repeatedly in practical CR applications. The measured sampling times demonstrate that EMRDM meets real-time requirements for CR applications, a critical factor for remote sensing, while delivering significant performance advantages.

C.3. Additional Results

This section presents additional results, including visual examples from the CUHK-CR1, CUHK-CR2, SEN12MS-CR, and Sen2_MTC_New datasets in Fig. 7, Fig. 8, Fig. 9, and Fig. 10, respectively. Visual comparisons using our stochastic and deterministic samplers are shown in Fig. 11. Additionally, results under varying settings of $(\alpha, \sigma_{\max}, N)$ are provided in Fig. 12.

References

[1] Jose D Bermudez, Patrick Nigri Happ, Dario Augusto Borges Oliveira, and Raul Queiroz Feitosa. Sar to optical image synthesis for cloud removal with generative adversarial networks. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 4:5–11, 2018. 8

[2] Katherine Crowson, Stefan Andreas Baumann, Alex Birch, Tanishq Mathew Abraham, Daniel Z Kaplan, and Enrico Shippole. Scalable high-resolution pixel-space image synthesis with hourglass diffusion transformers. In *Forty-first International Conference on Machine Learning*, 2024. 10

[3] 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. 10

[4] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 9

[5] Haidong Ding, Yue Zi, and Fengying Xie. Uncertainty-based thin cloud removal network via conditional variational autoencoders. In *Proceedings of the Asian Conference on Computer Vision*, pages 469–485, 2022. 14

[6] Patrick Ebel, Andrea Meraner, Michael Schmitt, and Xiao Xiang Zhu. Multisensor data fusion for cloud removal in global and all-season sentinel-2 imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 59(7):5866–5878, 2020. 10, 11

[7] Patrick Ebel, Yajin Xu, Michael Schmitt, and Xiao Xiang Zhu. Sen12ms-cr-ts: A remote-sensing data set for multimodal multitemporal cloud removal. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–14, 2022. 11, 12

[8] Patrick Ebel, Vivien Sainte Fare Garnot, Michael Schmitt, Jan Dirk Wegner, and Xiao Xiang Zhu. Uncrtaints: Uncertainty quantification for cloud removal in optical satellite time series. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2086–2096, 2023. 10, 11, 14

[9] Kenji Enomoto, Ken Sakurada, Weimin Wang, Hiroshi Fukui, Masashi Matsuoka, Ryosuke Nakamura, and Nobuo Kawaguchi. Filmy cloud removal on satellite imagery with multispectral conditional generative adversarial nets. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 48–56, 2017. 8, 14

[10] Jianhao Gao, Qiangqiang Yuan, Jie Li, Hai Zhang, and Xin Su. Cloud removal with fusion of high resolution optical and sar images using generative adversarial networks. *Remote Sensing*, 12(1):191, 2020. 8

[11] Sicheng Gao, Xuhui Liu, Bohan Zeng, Sheng Xu, Yanjing Li, Xiaoyan Luo, Jianzhuang Liu, Xiantong Zhen, and Baochang Zhang. Implicit diffusion models for continuous super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10021–10030, 2023. 10

[12] Claas Grohnfeldt, Michael Schmitt, and Xiaoxiang Zhu. A conditional generative adversarial network to fuse sar and multispectral optical data for cloud removal from sentinel-2 images. In *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*, pages 1726–1729. IEEE, 2018. 8

[13] Tiankai Hang, Shuyang Gu, Chen Li, Jianmin Bao, Dong Chen, Han Hu, Xin Geng, and Baining Guo. Efficient diffusion training via min-snr weighting strategy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7441–7451, 2023. 10

[14] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 3, 9, 10

[15] Gensheng Hu, Xiaoyi Li, and Dong Liang. Thin cloud removal from remote sensing images using multidirectional dual tree complex wavelet transform and transfer least square support vector regression. *Journal of Applied Remote Sensing*, 9(1):095053–095053, 2015. 8

[16] Gi-Luen Huang and Pei-Yuan Wu. Ctgan: Cloud transformer generative adversarial network. In *2022 IEEE International Conference on Image Processing (ICIP)*, pages 511–515. IEEE, 2022. 11, 14

[17] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005. 2

[18] Aapo Hyvärinen, Jarmo Hurri, Patrik O Hoyer, Aapo Hyvärinen, Jarmo Hurri, and Patrik O Hoyer. Estimation of non-normalized statistical models. *Natural Image Statistics: A Probabilistic Approach to Early Computational Vision*, pages 419–426, 2009. 3

[19] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 8, 14

[20] Ran Jing, Fuzhou Duan, Fengxian Lu, Miao Zhang, and Wenji Zhao. Denoising diffusion probabilistic feature-based network for cloud removal in sentinel-2 imagery. *Remote Sensing*, 15(9):2217, 2023. 9

[21] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022. 2, 3, 4, 5, 6, 7, 8, 10

[22] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. 10

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012. 14

[24] Fred A Kruse, AB Lefkoff, y JW Boardman, KB Heidebrecht, AT Shapiro, PJ Barloon, and AFH Goetz. The spectral image processing system (sips)—interactive visualization and analysis of imaging spectrometer data. *Remote sensing of environment*, 44 (2-3):145–163, 1993. 12

[25] Charis Lanaras, José Bioucas-Dias, Silvano Galliani, Emmanuel Baltsavias, and Konrad Schindler. Super-resolution of sentinel-2 images: Learning a globally applicable deep neural network. *ISPRS Journal of Photogrammetry and Remote Sensing*, 146:305–319, 2018. 10

[26] Congyu Li, Xinxin Liu, and Shutao Li. Transformer meets gan: Cloud-free multispectral image reconstruction via multi-sensor data fusion in satellite images. *IEEE Transactions on Geoscience and Remote Sensing*, 2023. 8

[27] 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. *arXiv preprint arXiv:2308.09388*, 2023. 9, 10

[28] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017. 10

[29] Chao-Hung Lin, Po-Hung Tsai, Kang-Hua Lai, and Jyun-Yuan Chen. Cloud removal from multitemporal satellite images using information cloning. *IEEE transactions on geoscience and remote sensing*, 51(1):232–241, 2012. 8

[30] Guan-Horng Liu, Arash Vahdat, De-An Huang, Evangelos A. Theodorou, Weili Nie, and Anima Anandkumar. I2sb: image-to-image schrödinger bridge. In *Proceedings of the 40th International Conference on Machine Learning*. JMLR.org, 2023. 10

[31] Jiawei Liu, Qiang Wang, Huijie Fan, Yinong Wang, Yandong Tang, and Liangqiong Qu. Residual denoising diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2773–2783, 2024. 10

[32] 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*, 35:5775–5787, 2022. 10

[33] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022. 10

[34] Ziwei Luo, Fredrik K. Gustafsson, Zheng Zhao, Jens Sjölund, and Thomas B. Schön. Image restoration with mean-reverting stochastic differential equations. In *Proceedings of the 40th International Conference on Machine Learning*, pages 23045–23066. PMLR, 2023. 1, 10

[35] Ziwei Luo, Fredrik K Gustafsson, Zheng Zhao, Jens Sjölund, and Thomas B Schön. Refusion: Enabling large-size realistic image restoration with latent-space diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1680–1691, 2023. 10

[36] Andrea Meraner, Patrick Ebel, Xiao Xiang Zhu, and Michael Schmitt. Cloud removal in sentinel-2 imagery using a deep residual neural network and sar-optical data fusion. *ISPRS Journal of Photogrammetry and Remote Sensing*, 166:333–346, 2020. 10, 14

[37] Mehdi Mirza. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 8

[38] Kevin P Murphy. *Probabilistic machine learning: Advanced topics*. MIT press, 2023. 3

[39] Axi Niu, Kang Zhang, Trung X Pham, Jinqiu Sun, Yu Zhu, In So Kweon, and Yanning Zhang. Cdpmsr: Conditional diffusion probabilistic models for single image super-resolution. In *2023 IEEE International Conference on Image Processing (ICIP)*, pages 615–619. IEEE, 2023. 10

[40] Heng Pan. Cloud removal for remote sensing imagery via spatial attention generative adversarial network. *arXiv preprint arXiv:2009.13015*, 2020. 8, 14

[41] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023. 10

[42] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022. 3

[43] Fabrizio Ramoino, Florin Tutunaru, Fabrizio Pera, and Olivier Arino. Ten-meter sentinel-2a cloud-free composite—southern africa 2016. *Remote Sensing*, 9(7):652, 2017. 8

[44] 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*, pages 1–10, 2022. 10

[45] 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*, 45(4):4713–4726, 2022. 10

[46] Vishnu Sarukkai, Anirudh Jain, Burak Uzkent, and Stefano Ermon. Cloud removal from satellite images using spatiotemporal generator networks. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1796–1805, 2020. 14

[47] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019. 10

[48] 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. 1, 2, 10

[49] Jialu Sui, Yiyang Ma, Wenhan Yang, Xiaokang Zhang, Man-On Pun, and Jiaying Liu. Diffusion enhancement for cloud removal in ultra-resolution remote sensing imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 2024. 10, 11, 14, 15

[50] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017. 8

[51] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011. 3

[52] Tianxing Wang, Jiancheng Shi, Husi Letu, Ya Ma, Xingcai Li, and Yaomin Zheng. Detection and removal of clouds and associated shadows in satellite imagery based on simulated radiance fields. *Journal of Geophysical Research: Atmospheres*, 124(13):7207–7225, 2019. 8

[53] 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*, 13(4):600–612, 2004. 12

[54] Fang Xu, Yilei Shi, Patrick Ebel, Lei Yu, Gui-Song Xia, Wen Yang, and Xiao Xiang Zhu. Glf-cr: Sar-enhanced cloud removal with global–local fusion. *ISPRS Journal of Photogrammetry and Remote Sensing*, 192:268–278, 2022. 10, 12, 14, 15

[55] Meng Xu, Mark Pickering, Antonio J Plaza, and Xiuping Jia. Thin cloud removal based on signal transmission principles and spectral mixture analysis. *IEEE Transactions on Geoscience and Remote Sensing*, 54(3):1659–1669, 2015. 8

[56] Meng Xu, Xiuping Jia, Mark Pickering, and Sen Jia. Thin cloud removal from optical remote sensing images using the noise-adjusted principal components transform. *ISPRS Journal of Photogrammetry and Remote Sensing*, 149:215–225, 2019. 8

[57] Meng Xu, Furong Deng, Sen Jia, Xiuping Jia, and Antonio J Plaza. Attention mechanism-based generative adversarial networks for cloud removal in landsat images. *Remote sensing of environment*, 271:112902, 2022. 14

[58] Weikang Yu, Xiaokang Zhang, and Man-On Pun. Cloud removal in optical remote sensing imagery using multiscale distortion-aware networks. *IEEE Geoscience and Remote Sensing Letters*, 19:1–5, 2022. 14

[59] Zongsheng Yue, Jianyi Wang, and Chen Change Loy. Resshift: efficient diffusion model for image super-resolution by residual shifting. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2024. Curran Associates Inc. 10

[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*, pages 586–595, 2018. 13, 14

[61] Xiao Feng Zhang, Chao Chen Gu, and Shan Ying Zhu. Memory augment is all you need for image restoration. *arXiv preprint arXiv:2309.01377*, 2023. 14

[62] Xiaohu Zhao and Kebin Jia. Cloud removal in remote sensing using sequential-based diffusion models. *Remote Sensing*, 15(11): 2861, 2023. 10

[63] Xuechao Zou, Kai Li, Junliang Xing, Pin Tao, and Yachao Cui. Pmaa: A progressive multi-scale attention autoencoder model for high-performance cloud removal from multi-temporal satellite imagery. *arXiv preprint arXiv:2303.16565*, 2023. 14

[64] Xuechao Zou, Kai Li, Junliang Xing, Yu Zhang, Shiying Wang, Lei Jin, and Pin Tao. Diffcr: A fast conditional diffusion framework for cloud removal from optical satellite images. *IEEE Transactions on Geoscience and Remote Sensing*, 62:1–14, 2024. 9, 14, 15