# Efficient and Training-Free Single-Image Diffusion Models

Haojun Qiu[1] Kiriakos N. Kutulakos[1,2] David B. Lindell[1,2]
[1]Department of Computer Science, University of Toronto [2]Vector Institute
https://haojunqiu.github.io/efficient-SID/

## Abstract

*We consider the problem of generating images whose internal structure—defined by the distribution of patches across multiple scales—matches that of a single reference image. Recent approaches address this problem by training a diffusion model on a single image. But even in this setting, training is computationally expensive and requires hours of optimization. Instead, we model the image using a dataset of its patches at different scales. As this dataset is finite and the dimensionality of its patches is small, the score function for a noisy patch can be computed tractably using an optimal, closed-form denoiser, eliminating the need for neural network training. We integrate this patch-based denoiser into an efficient, training-free image diffusion model, and we describe how our method connects to classical patch-based image restoration techniques. Our approach achieves state-of-the-art generation quality and diversity compared to trained single-image diffusion models, and we demonstrate applications, including unconditional image generation, text-guided stylization, image symmetrization, and retargeting. Further, we show that our approach is compatible with latent space diffusion, and we show multiple additional acceleration techniques to achieve megapixel single-image generation in one second, and gigapixel generation in minutes.*

## 1. Introduction

A single image contains a dataset of thousands to millions of patches—local neighborhoods or groups of pixels—occurring across different positions and scales. The distribution of image patches conveys information about the *internal structure* of an image [73]; for example, most images have patches that are self-similar within a scale, correlated in appearance across scales, and similar in their spatial frequency content. Analyzing and modeling the internal structure of images has led to significant advances in applications such as unconditional image generation [26, 58], image manipulation [38, 50], and image restoration [12, 43, 59, 74].

Although non-parametric sampling methods for patch-based image synthesis and restoration have a long history in computer vision [3, 7, 12, 18, 19, 23, 34, 71], recent work has focused instead on using generative models to learn the distribution of patches from a single image. Techniques based on generative adversarial networks (GANs) generate images whose distribution of patches matches that of a source image based on the output of a discriminator [24, 28, 58]. Other techniques train a diffusion model to denoise a single image at multiple scales with varying amounts of Gaussian noise [38, 50]. After training, new images with a similar internal structure can be sampled by applying a coarse-to-fine denoising procedure. However, training single-image generative models is computationally expensive, requiring several hours of optimization even though the training data comprises only a single image. Further, such generative models can be difficult to optimize—especially GANs, which are susceptible to local minima and mode collapse [6, 15, 46, 53].

A key advantage of classical patch-based modeling techniques is that they require no training, and thus are far more computationally lightweight compared to GANs and diffusion models. Moreover, recent work in this direction has shown competitive results in terms of generated image quality [26] despite relying on nearest-neighbor patch matching [3] instead of internal learning. Similar to denoising diffusion models, these methods generate images through a coarse-to-fine processing procedure that begins from a noisy, coarse-resolution input. But, instead of denoising, images are generated by iteratively refining patches with nearest-neighbor matching and progressive upscaling. While this approach is efficient and effective, it is less flexible than score-based denoising diffusion models [29, 63], which explicitly model the prior probability of image patches. For example, diffusion models are easily combined with vision–language models for text-based editing [38, 52], and they can incorporate symmetry constraints or local edits during the generation process [42].

Here, we introduce a method for modeling the internal structure of images using a closed-form denoising diffusion procedure that is entirely training-free, similar to conventional patch-based methods. Hence, our approach avoids the computational cost of internal learning while inheriting the advantages of diffusion models in terms of explicit probabilistic modeling (see Figure 1).

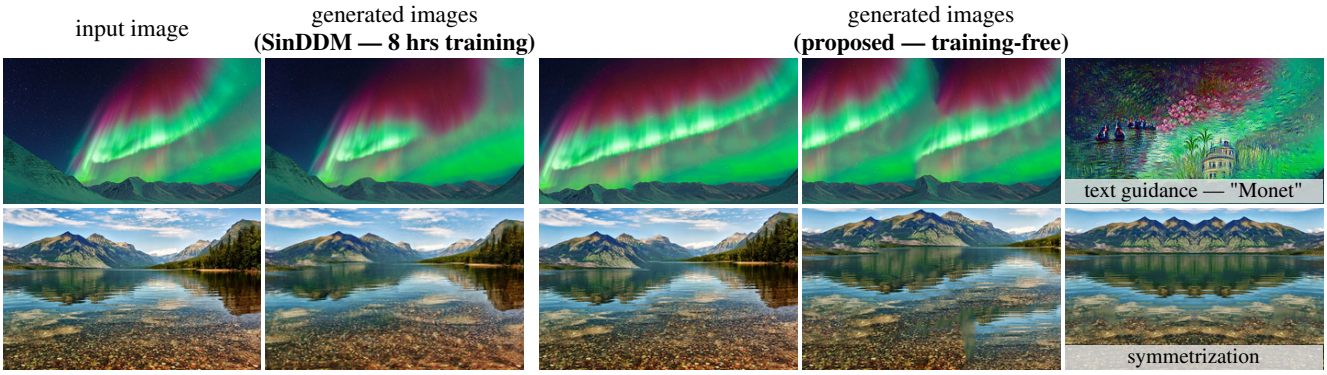


Figure 1. We introduce an efficient, training-free diffusion model that generates images based on the internal structure of a single input image. Our approach uses a closed-form solution for the optimal denoiser derived for noisy patches of the input image. As such, no training is required, and generated images achieve similar or better quality (based on single-image Fréchet Inception Distance [58]) relative to state-of-the-art single-image diffusion models such as SinDDM [38], which require hours of training time. Our approach is also compatible with text-based guidance from pre-trained vision–language models [52] and enables controllable generation, e.g., of symmetric images.

Our approach is based on the following key observation: since the set of patches in a single image is finite, the score function corresponding to the distribution of patches at all positions and scales can be computed in closed form [8, 36, 41, 49, 56, 64], without training a neural network. That is, evaluating the score function for a noisy patch corresponds to applying a denoiser that is optimal for the ensemble of patches in an image. At the image level, the optimal denoiser takes a form similar to a non-local-means denoiser, thereby drawing a connection between recent denoising diffusion models [29, 63] and classical patch-based methods [7, 74]. To generate coherent images across patch boundaries, we integrate this closed-form denoiser into a novel reverse diffusion process that operates in a coarse-to-fine fashion. Finally, we demonstrate our efficient, closed-form single-image denoiser for applications such as unconditional image generation, retargeting, text-based stylization and editing, and image symmetrization [42, 44].

Although the connection between the score function and denoising has long been known [31, 69], we show that the analytical denoising solution is especially well-suited to modeling internal image structure. Moreover, we find that patch-based diffusion is amenable to multiple acceleration techniques: (1) fused attention kernels originally developed for transformers [14, 68], (2) latent space diffusion [22, 55], and (3) approximate nearest neighbors for rapidly identifying similar patches [32]. Together, these techniques enable us to achieve megapixel generation in one second and gigapixel generation in minutes. Overall, our closed-form denoising solution achieves state-of-the-art capabilities in single-image modeling without the hours-long training of other diffusion-based methods [38, 50].

## 2. Related Work

**Classical patch-based models.** Modeling image structure using patches has long been attractive because it leads to tractable methods for image analysis, inference, and likelihood estimation. For example, analyzing the distribution of patches within natural images reveals that patches typically recur many times [73]. Patch self-similarity is the key principle of non-parametric single-image techniques for texture synthesis [18, 19, 27], stylization [3, 27], restoration [12, 20], and state-of-the-art non-local denoising methods [7, 12]. Parametric methods seek to explicitly model the prior probability of patches using models such as Gaussian mixtures [51, 74]. As we will show, our approach connects classical non-parametric and parametric techniques for patch-based modeling and grounds them in the modern framework of diffusion-based inference.

**Single-image GANs.** More recently, GANs have been used to learn the distribution of patches from a single image [28, 58, 60]. These methods train a generator to create images whose patch statistics match those of the input image across multiple scales. However, GANs do not support guidance (e.g., from text prompts) without re-training.

**Single-image diffusion models.** Diffusion models are an attractive alternative to GANs; they are easier to train, achieve higher-quality generation [15], and avoid challenges related to sample diversity [46, 53]. In the context of single-image modeling, recent methods based on diffusion models show compelling results in image generation, manipulation, and text-driven stylization [38, 50, 70] but are expensive to train. Moreover, they model patches implicitly by restricting the receptive field of the network [50, 70] or by generating images sequentially across scales [38]. Similar to our method, concurrent work investigates patch-based models in the context of closed-form diffusion, but focuses on texture synthesis rather than image generation [10]. Our method operates explicitly on patches using an optimal closed-form denoiser, achieving similar or better image generation quality relative to prior training-based diffusion models—without any training. See Sec. S1 for a detailed discussion of how our approach compares to previous work.

## 3. Method

We now describe our method for single-image generative modeling by closed-form denoising diffusion. We begin with an overview of diffusion sampling and the closed-form denoiser (Sec. 3.1) and connect our approach to classical patch-based methods (Sec. 3.2). We then describe how to integrate the closed-form denoiser into a multi-scale approach for image sampling (Secs. 3.3 and 3.4).

### 3.1. Closed-Form Denoising Diffusion

**Preliminaries.** Diffusion models [29, 62, 63, 65, 66] act on a forward diffusion process that adds an increasing amount of noise in steps $t \in [0, \dots, T]$ to a clean signal $\mathbf{y} \in \mathbb{R}^M$ sampled from a dataset $\mathcal{Y}$. The noisy signal $\mathbf{x}_t$ at step $t$ is given as $\mathbf{x}_t = \alpha(t)\mathbf{y} + \sigma(t)\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is standard normal distributed, and the noise-scheduling parameters $\alpha(t)$, $\sigma(t)$ are smooth functions chosen so that $\alpha(0) = \sigma(T) = 1$ and $\alpha(T) = \sigma(0) = 0$ (i.e., $t = 0$ corresponds to the clean signal). The diffusion model learns the structure of signals in $\mathcal{Y}$ by running the diffusion process in reverse. That is, the model consists of a denoiser $D$ that is trained to minimize

$$\mathbb{E}_{\mathbf{y}\sim\mathcal{Y}, t\sim\mathcal{U}[0,T], \boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})} \left[ w(t) \| D(\mathbf{x}_t, t) - \mathbf{y} \|_2^2 \right], \quad (1)$$

where $w(t)$ is a weight that depends on $t$ [63]. Hence, the model learns to denoise all signals in the dataset over all diffusion timesteps.

**Closed-form denoising.** Instead of training the diffusion model, our approach uses an optimal, closed-form denoiser inspired by previous work [5, 8, 35, 36, 41, 49, 56, 64]. We use this denoiser to iteratively reverse the diffusion process to recover a clean signal from noise. The closed-form denoiser is given as

$$D(\mathbf{x}_t, \mathcal{Y}, t) = \frac{\sum\limits_{\mathbf{y}\in\mathcal{Y}} p_{\mathcal{N}}(\mathbf{x}_t; \alpha\mathbf{y}, \sigma^2\mathbf{I})\, \mathbf{y}}{\sum\limits_{\mathbf{y}\in\mathcal{Y}} p_{\mathcal{N}}(\mathbf{x}_t; \alpha\mathbf{y}, \sigma^2\mathbf{I})}, \quad (2)$$

which computes the denoised signal as a weighted function of all clean signals in the dataset $\mathcal{Y} = \{\mathbf{y}^{(1)}, \dots \mathbf{y}^{(Y)}\}$ (please see Sec. S2 for the derivation). We use the denoiser to run the reverse diffusion process with the following iterative updates:

$$\hat{\mathbf{x}}_t \leftarrow D(\mathbf{x}_t, \mathcal{Y}, t) \quad \text{(denoised signal)}, \quad (3)$$

$$\hat{\boldsymbol{\epsilon}}_t \leftarrow (\mathbf{x}_t - \alpha(t)\hat{\mathbf{x}}_t)/\sigma(t) \quad \text{(estimated noise)}, \quad (4)$$

$$\mathbf{x}_{t-1} \leftarrow \alpha(t-1)\hat{\mathbf{x}}_t \quad \text{(noisy signal)}, \quad (5)$$
$$+ \sqrt{\sigma(t-1)^2 - c(t-1)^2}\,\hat{\boldsymbol{\epsilon}}_t + c(t-1)\boldsymbol{\epsilon}_t$$

$$t \leftarrow t - 1 \quad \text{(update timestep)}. \quad (6)$$

where $c(t-1) \in [0, \sigma(t-1)]$ modifies the amount of sampling stochasticity by adding random noise $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and we use $\eta(t) = c(t)/\sigma(t) \in [0, 1]$ to control the stochasticity. Equations 3–6 reduce to a deterministic Denoising Diffusion Implicit Model (DDIM) [63] when $\eta(t) = 0$. Iterating these steps yields the generated signal $\mathbf{x}_0$.

### 3.2. Connection to Patch-Based Image Restoration

Many classical image restoration methods that rely on non-parametric sampling [7] can be viewed as employing the exact same closed-form denoiser at the patch level, followed by an image reconstruction step that reassembles a full image from a set of denoised patches. Below, we describe connections of our framework to these classical techniques.

**Non-local means denoising [7].** Given an input noisy image $\mathbf{x}$, non-local means computes a denoised image $\hat{\mathbf{x}}$ from a dataset of all overlapping noisy image patches, $\mathcal{X} = \{\mathbf{P}^{(1)}, \dots, \mathbf{P}^{(N)}\mathbf{x}\} \stackrel{\text{def}}{=} \{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(N)}\}$, where $\mathbf{P}^{(i)}$ is a matrix that extracts a patch $\mathbf{x}^{(i)}$ from the image. Each patch is denoised as

$$\underbrace{\hat{\mathbf{x}}^{(i)}}_{\text{denoised patch}} \leftarrow D( \underbrace{\mathbf{x}^{(i)}}_{\text{patch to be denoised}}, \underbrace{\mathcal{X} - \{\mathbf{x}^{(i)}\}}_{\text{all other noisy patches}}, t). \quad (7)$$

The output image is then assembled as $\hat{\mathbf{x}} \leftarrow \sum_{i=1}^{N} \mathbf{R}^{(i)}\hat{\mathbf{x}}^{(i)}$, where $\mathbf{R}^{(i)}$ is a matrix that copies the center pixel of the patch back to its corresponding location in the image. Note that the formulation here is identical to Equation 2, except that the dataset $\mathcal{X}$ consists of noisy patches.

**Image restoration with GMM patch priors [74].** One way to model patch priors for image restoration is to fit a Gaussian mixture model (GMM) to the patch dataset [74]. This results in a prior patch probability of $p(\mathbf{x}^{(i)}) = \sum_{k=1}^{K} \pi_k p_{\mathcal{N}}(\mathbf{x}^{(i)}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, where $\pi_k$ are the mixing coefficients for the $K$ components, and $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ are the mean and covariance matrix, respectively. However, this prior does not yield a closed-form solution for the maximum a posteriori estimate of the clean patch given a noisy patch $\mathbf{x}^{(i)}$ [74]. Further, fitting the GMM prior to the patch dataset requires an expensive optimization using expectation maximization.

The closed-form denoiser of Equation 2 can be thought of as a restoration technique that replaces the above patch prior with a trivial GMM that centers a mixture component at each patch in the dataset: $p(\mathbf{x}^{(i)}) = \frac{1}{N}\sum_{j=1}^{N} p_{\mathcal{N}}(\mathbf{x}^{(i)}; \mathbf{y}^{(j)}, \sigma^2\mathbf{I})$. As $\sigma \to 0$, Equation 2 converges to the exact prior probability $p(\mathbf{x}^{(i)})$, and the closed-form denoiser gives the minimum mean squared error estimate of a clean patch given the noisy patch $\mathbf{x}^{(i)}$ (see Sec. S2). Hence, our formulation relies on a GMM prior—just like classical patch-based methods—but one that estimates clean patches efficiently in closed form.

### 3.3. Single-Scale Image Sampling

We now describe how to sample entire images by applying the closed-form denoiser at the patch level. The procedure is illustrated in Figure 2 and summarized in Algorithm 1.

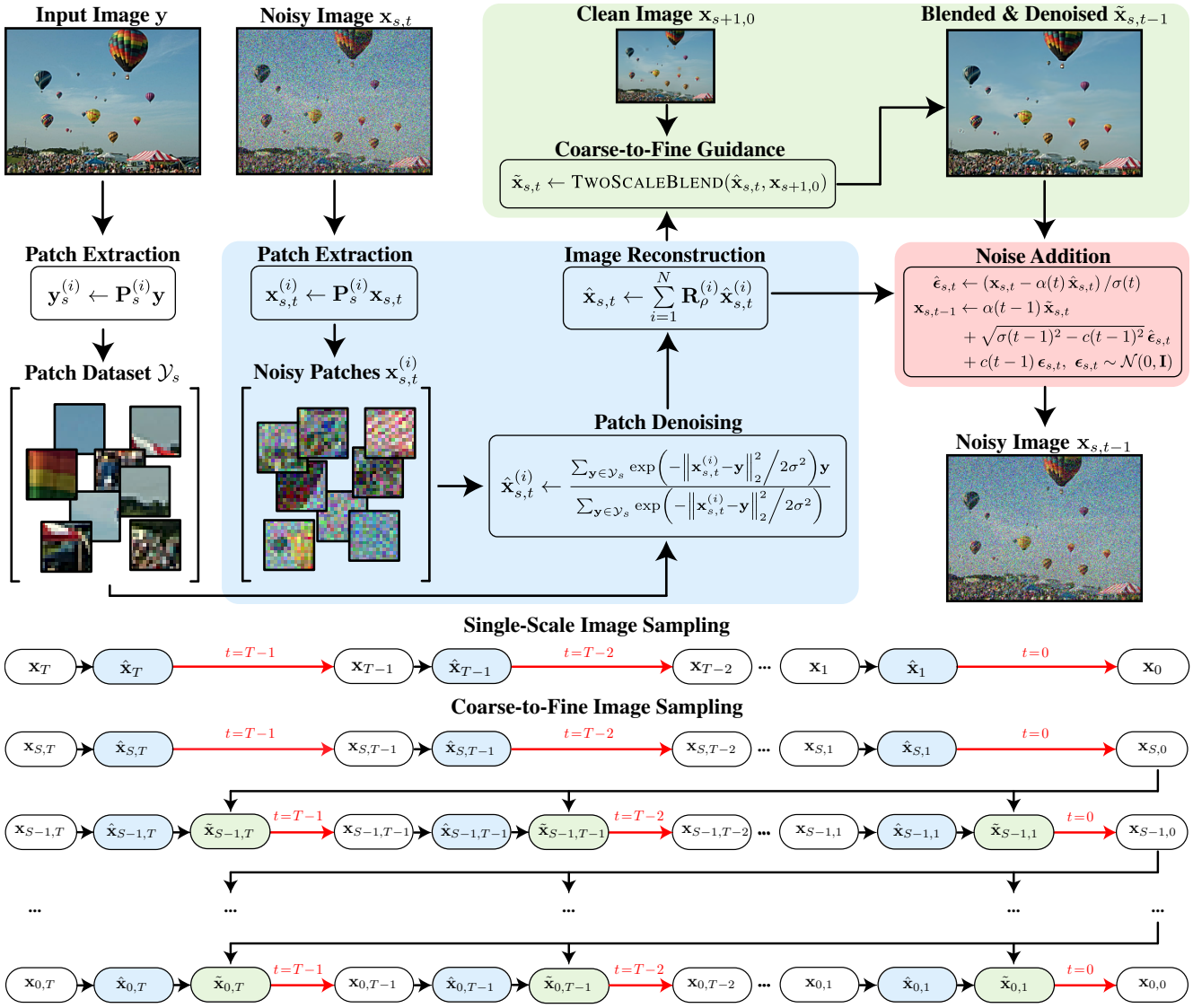


Figure 2. Method overview. Our approach takes a single image as input, extracts patches, and uses the patches to generate new images using denoising diffusion. **(top)** We illustrate a single step of the reverse diffusion process: *(blue)* patches from the noisy image are denoised and used to reconstruct an image; *(green)* the denoised image is blended with the output of the reverse diffusion process at a coarser scale ($\mathbf{x}_{s+1,t}$); *(red)* the noisy image at the previous diffusion timestep ($\mathbf{x}_{s,t-1}$) is sampled. **(bottom)** All steps of reverse diffusion process for single-scale image sampling (omits coarse-to-fine guidance (*green*)) and coarse-to-fine image sampling are shown.

The procedure begins by extracting a dataset $\mathcal{Y}$ of overlapping patches from the input image. These patches are treated as "clean," i.e., noise-free. To sample an image, we start at the last timestep of the forward diffusion by sampling a noise image $\mathbf{x}_{t=T} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. We then iteratively denoise it through the reverse diffusion process as follows. We extract noisy patches from the image as $\mathbf{x}_t^{(i)} \leftarrow \mathbf{P}^{(i)}\mathbf{x}_t$, and we apply Equations 3–6 to recover denoised patches $\hat{\mathbf{x}}_t^{(i)}$. Then, we reconstruct a full image as $\hat{\mathbf{x}}_t \leftarrow \sum_{i=1}^{N} \mathbf{R}_\rho^{(i)} \hat{\mathbf{x}}_t^{(i)}$, where $\mathbf{R}_\rho^{(i)}$ is a matrix that copies the patch back to its original location in the image after weighting by a Gaussian with standard deviation $\rho$. Here, $\rho$ controls how much of the patch outside of the center pixel is copied back to the image (i.e., $\rho = 0$ corresponds to the $\mathbf{R}^{(i)}$ matrix employed by non-local-means denoising). Finally, we add noise to sample a noisy image for timestep $t \leftarrow t-1$ of the forward diffusion. This process iterates until we sample the output image $\mathbf{x}_{t=0}$. Figure 2 (middle) illustrates this sequence of denoising and sampling steps.

We show examples of single-scale image sampling in Figure 3. Note that while image structures at the scale of a patch are preserved, the sampled images do not maintain the input image's global structure. To address this issue, we develop a coarse-to-fine image sampling procedure that preserves global image structure from coarse scales while maintaining high-frequency details from fine scales.

**Algorithm 1** Single-Scale Image Sampling

**procedure** SAMPLEIMAGE($\mathbf{y}$)
  $\mathcal{Y} = \{\mathbf{P}^{(1)}\mathbf{y}, \dots, \mathbf{P}^{(N)}\mathbf{y}\}$ ▷ extract clean patches
  $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  **for** $t = [T, \dots, 1]$ **do** ▷ reverse diffusion steps
    $\hat{\mathbf{x}}_t \leftarrow \text{IMGDENOISE}(\mathbf{x}_t, \mathcal{Y}, t)$
    $\hat{\boldsymbol{\epsilon}}_t \leftarrow (\mathbf{x}_t - \alpha(t)\hat{\mathbf{x}}_t)/\sigma(t)$
    $\mathbf{x}_{t-1} \leftarrow \alpha(t-1)\hat{\mathbf{x}}_t + \sqrt{\sigma(t-1)^2 - c(t-1)^2}\hat{\boldsymbol{\epsilon}}_t + c(t-1)\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  **end for**
  **return** $\mathbf{x}_{t=0}$
**end procedure**
**procedure** IMGDENOISE($\mathbf{x}_t, \mathcal{Y}, t$)
  $\{\mathbf{x}_t^{(i)}\}_{i=1}^N \leftarrow \{\mathbf{P}^{(1)}\mathbf{x}_t, \dots, \mathbf{P}^{(N)}\mathbf{x}_t\}$ ▷ extract patches
  $\{\hat{\mathbf{x}}_t^{(i)}\}_{i=1}^N \leftarrow \{\text{PATCHDENOISE}(\mathbf{x}_t^{(i)}, \mathcal{Y}, t)\}_{i=1}^N$ ▷ denoise
  $\hat{\mathbf{x}}_t \leftarrow \sum_{i=1}^N \mathbf{R}_\rho^{(i)}\hat{\mathbf{x}}_t^{(i)}$ ▷ reconstruct image
  **return** $\hat{\mathbf{x}}_t$
**end procedure**
**procedure** PATCHDENOISE($\mathbf{x}_t, \mathcal{Y}, t$)
  $\hat{\mathbf{x}}_t \leftarrow \frac{\sum_{\mathbf{y}\in\mathcal{Y}} \exp\left(-\|\mathbf{x}_t - \alpha\mathbf{y}\|_2^2/2\sigma^2\right)\mathbf{y}}{\sum_{\mathbf{y}\in\mathcal{Y}} \exp\left(-\|\mathbf{x}_t - \alpha\mathbf{y}\|_2^2/2\sigma^2\right)}$
  **return** $\hat{\mathbf{x}}_t$
**end procedure**

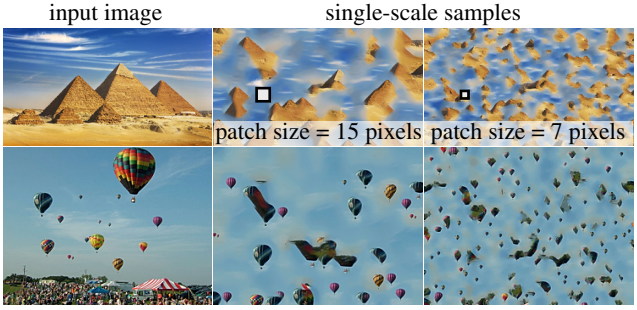


Figure 3. Illustration of single-scale sampling. This procedure (detailed in Algorithm 1) captures image statistics at the scale of an individual patch (white squares), but fails to capture the coarse structure of the image. We address this issue using coarse-to-fine image sampling (Algorithm 2, Figure 4).

### 3.4. Coarse-to-Fine Image Sampling

We achieve coarse-to-fine image generation by first running the single-scale image sampling procedure at the coarsest image scale. Then, we incorporate the coarse-scale output into the generation of an image at a finer scale, and we repeat this process until we output an image at the highest-resolution scale. We illustrate the method in Figure 2 (bottom) and provide a pseudocode description in Algorithm 2.

More specifically, we first initialize a noisy image pyramid as $\{\mathbf{x}_{s,T}\}_{s=0}^S \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Here, $s = S$ is the coarsest scale and $s = 0$ corresponds to the full-resolution image. For $s = S$, we follow the single-scale image sampling procedure exactly to sample an output image at that scale. For the other scales, $s \in \{S-1, \dots, 0\}$, we follow the single-scale sampling approach to gather and denoise patches from the noisy images $\mathbf{x}_{s,t}$ and reconstruct a denoised image $\hat{\mathbf{x}}_{s,t}$ at each diffusion timestep; additionally, we guide the generation process using the output $\mathbf{x}_{s+1,t=0}$ from the previous (coarser) scale. The coarse-scale guidance is incorporated by applying a high-pass filter to the denoised image $\hat{\mathbf{x}}_{s,t}$ at the current scale, and adding the result to an upsampled version of $\mathbf{x}_{s+1,t=0}$ (see the TWOSCALEBLEND function of L9 in Algorithm 2). This operation can be thought of as a two-scale version of Laplacian pyramid blending [9]. Finally, we add noise to this result to compute $\mathbf{x}_{s,t-1}$—the noisy image at the previous diffusion timestep for this scale. The same denoising, blending, and noise-addition steps are repeated for each scale until the output $\mathbf{x}_{s=0,t=0}$ at the finest scale is produced. Note that in this coarse-to-fine sampling procedure, we denoise the image patches using a dataset of patches $\mathcal{Y}_s$ extracted from the input image at the same scale.

**Algorithm 2** Coarse-to-Fine Image Sampling

1: **procedure** SAMPLEIMAGECOARSETOFINE($\mathbf{y}$)
2:  $\{\mathcal{Y}_s\}_{s=0}^S \leftarrow \{\mathbf{P}_s^{(1)}\mathbf{y}, \dots, \mathbf{P}_s^{(N_s)}\mathbf{y}\}_{s=0}^S$ ▷ extract patches
3:  $\{\mathbf{x}_{s,T}\}_{s=0}^S \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ ▷ initialize a noise pyramid
4:  **for** $s = [S, \dots, 0]$ **do** ▷ iterate over scales
5:  **for** $t = [T, \dots, 1]$ **do** ▷ diffusion timesteps
6:  $\hat{\mathbf{x}}_{s,t} \leftarrow \text{IMGDENOISE}(\mathbf{x}_{s,t}, \mathcal{Y}_s, t)$
7:  $\hat{\boldsymbol{\epsilon}}_{s,t} \leftarrow (\mathbf{x}_{s,t} - \alpha(t)\hat{\mathbf{x}}_{s,t})/\sigma(t)$
8:  **if** $s < S$ **then**
9:  $\tilde{\mathbf{x}}_{s,t} \leftarrow \text{TWOSCALEBLEND}(\hat{\mathbf{x}}_{s,t}, \mathbf{x}_{s+1,t=0})$
10:  **end if**
11:  $\mathbf{x}_{s,t-1} \leftarrow \alpha(t-1)\tilde{\mathbf{x}}_{s,t} + \sqrt{\sigma(t-1)^2 - c(t-1)^2}\hat{\boldsymbol{\epsilon}}_{s,t} + c(t-1)\boldsymbol{\epsilon}_{s,t}, \ \boldsymbol{\epsilon}_{s,t} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
12:  **end for**
13:  **end for**
14:  **return** $\mathbf{x}_{s=0,t=0}$
15: **end procedure**
16: **procedure** TWOSCALEBLEND($\hat{\mathbf{x}}_{s,t}, \mathbf{x}_{s+1,t=0}$)
17:  **return** $\hat{\mathbf{x}}_{s,t} - \text{BLUR}(\hat{\mathbf{x}}_{s,t}) + \text{UPSAMPLE}(\mathbf{x}_{s+1,t=0})$
18: **end procedure**

### 3.5. Acceleration Techniques

Our closed-form denoiser can be further accelerated using several complementary techniques. First, we leverage PyTorch's fused attention kernel (based on FlashAttention [13, 14]), by re-formulating our patch denoiser as scaled-dot-product attention (see Sec. S3). Additionally, we can apply our method in a compressed latent space, similar to large diffusion models [55]. That is, we encode the input image into a latent using a pre-trained variational autoencoder (VAE) [39] and then perform denoising in the latent space. Finally, we can approximate the summation in Equation 2 using $k$ approximate nearest neighbors (ANN) [32] found via a clustering-based index [61]. We use an inverted file index with $\sqrt{N}$ clusters and probe a fixed number of them at query time, which reduces the cost from $\mathcal{O}(N^2)$ to $\mathcal{O}(N^{3/2})$. Additional implementation details are provided in Sec. S3.


Figure 4. Unconditional single-image generation results. Our training-free, coarse-to-fine image sampling procedure based on closed-form denoising diffusion (right) produces results of the same quality as other state-of-the-art methods that require hours of training time.

| type | metric | SinGAN [58] | GPNN [26] | GPDM [21] | SinDDM [38] | SinFusion [50] | SinDiffusion [70] | proposed ($T$=10,$\eta$=0) | proposed ($T$=40, $\eta$=1) | proposed ($T$=10, $\eta$=0, $k$=5) |
|---|---|---|---|---|---|---|---|---|---|---|
| *patch distribution* | SIFID ↓ | 0.13±0.08 | 0.06±0.11 | **0.015±0.01** | 0.48±0.62 | 0.51±0.49 | 0.31±0.35 | 0.29±0.39 | 0.21±0.29 | 0.38±0.52 |
| *no reference IQA* | NIQE ↓ | 7.95±3.37 | 9.78±5.59 | 7.99±3.04 | 7.69±3.60 | 10.17±5.29 | **6.96±2.88** | 8.08±3.23 | 8.18±3.05 | 8.10±3.50 |
| | NIMA ↑ | 4.32±0.39 | 4.69±0.48 | 4.21±0.35 | 4.30±0.46 | **4.75±0.45** | 4.19±0.43 | 4.53±0.45 | 4.47±0.48 | 4.52±0.43 |
| | MUSIQ ↑ | 48.26±10.40 | **56.60±11.04** | 49.72±11.41 | 50.74±11.18 | 51.38±12.40 | 49.31±9.83 | 55.41±11.05 | 55.81±11.40 | 55.13±11.32 |
| *diversity* | Pixel Div.↑ | 0.09±0.03 | 0.08±0.02 | 0.10±0.04 | 0.10±0.03 | 0.11±0.03 | 0.11±0.03 | **0.15±0.04** | 0.13±0.03 | 0.15±0.03 |
| | LPIPS Div.↑ | 0.27±0.07 | 0.29±0.09 | 0.31±0.14 | 0.36±0.07 | 0.38±0.07 | 0.41±0.07 | **0.49±0.07** | 0.39±0.06 | 0.50±0.08 |
| *training time* | TITAN RTX (hrs) ↓ | 2.0 | **0.0** | **0.0** | 10.0 | 3.2 | 5.4 | **0.0** | **0.0** | **0.0** |
| | A6000 (hrs) ↓ | *not supported* | **0.0** | **0.0** | 8.0 | 1.5 | 4.2 | **0.0** | **0.0** | **0.0** |
| *inference time* | TITAN RTX (s) ↓ | **0.04±0.00** | 2.62±0.01 | 9.82±0.29 | 1.60±0.06 | 2.09±0.04 | 14.25±0.23 | 4.49±0.02 | 18.53±0.11 | 1.41±0.04 |
| | A6000 (s) ↓ | *not supported* | 2.08±0.10 | 11.49±0.20 | 1.25±0.05 | 1.99±0.09 | 12.10±0.08 | 3.09±0.02 | 12.57±0.05 | **0.88±0.02** |

Table 1. Quantitative assessment of unconditional generation. We report the mean and standard deviation for each metric. While GPNN and GPDM perform best in terms of SIFID [58], we find that they sample near-duplicates of the input image with high probability (see Sec. S5). Our approach performs on par with or better than other single-image diffusion models in terms of SIFID as well as NIQE, NIMA, and MUSIQ (no-reference image quality metrics) [37, 45, 67]. We improve over other methods in terms of the diversity of generated images (LPIPS distance and pixel diversity [38]), and we avoid the long optimization times of trained methods. Increasing the number of diffusion steps can improve SIFID ($T = 40$, $\eta = 1$), and using approximate nearest neighbours ($k = 5$) accelerates inference, with only minimal quality loss (a 0.09 increase in SIFID). We time the training and inference on a 186×248 image, averaged over 10 runs. Note that the publicly available codebase for SinGAN does not support inference on A6000 GPUs.

### 3.6. Implementation Details

We implement our approach in PyTorch and code is publicly available on the project webpage. For $T$=10 and `float32` precision, a naive PyTorch implementation of our coarse-to-fine sampling procedure takes approximately 3 seconds for a 186×248 image on an NVIDIA A6000 GPU. We found only a modest improvement in sampled image quality for $T \geq 10$ and so we use $T$=10 for all results unless otherwise stated. For images of $\approx$ 250×250 pixel resolution, we use a patch size of 15×15 pixels with $S$=4 scales. At subsequent scales, the image resolution changes by a factor of two in each dimension, and we set the number of scales so that the patch size is about half the size of the image at the coarsest scale. We extract patches with a stride of one pixel, and we use a Gaussian with a standard deviation of $\rho$=0.2 to weight the patches before they are reassembled into an image as $\sum_{i=1}^{N} \mathbf{R}_{\rho}^{(i)} \hat{\mathbf{x}}^{(i)}$. To set $\sigma(t)$ and $\alpha(t)$, we use the flow matching schedule [40], where $\alpha(t) = 1 - t/T$ and $\sigma(t) = t/T$, and we use deterministic sampling ($\eta(t) = 0$) unless otherwise specified. In practice, we omit the two-stage blending on the $t = 0$ diffusion step, which we find improves the results.

## 4. Experiments

We demonstrate our approach for applications of single-image generative modeling, including unconditional generation, image retargeting, symmetrization, structural analogies [4], and text-guided style transfer. We compare our approach to state-of-the-art single-image generative models: SinGAN [58], SinDDM [38], SinFusion [50], SinDiffusion [70], GPNN [26], and GPDM [21]. SinGAN uses a generative adversarial network, SinDDM, SinFusion, and SinDiffusion are based on diffusion models, GPNN uses patch nearest neighbors, and GPDM optimizes patch distributions via a Wasserstein distance.

| method | $256^2$ | $512^2$ | $1024^2$ | $2048^2$ | $4096^2$ | $8192^2$ |
|---|---|---|---|---|---|---|
| vanilla | 2.27 | 42.90 | 733.75 | >1 hr | >1 hr | >1 hr |
| + fused attention | 1.26 | 23.42 | 401.79 | >1 hr | >1 hr | >1 hr |
| + latent space | 0.36 | 0.39 | 0.65 | 3.43 | 36.65 | 523.97 |
| + ANN | 0.65 | 0.96 | 1.30 | 3.85 | 15.14 | 69.39 |

Table 2. Inference time versus image resolution (seconds; lower is better), measured with $T$=10 denoising steps on an RTX 6000 Ada (48 GB VRAM, 12 CPUs). Timing covers all stages (VAE encode/decode and ANN clustering); runs are in `bfloat16` except ANN in `float32`. **Fused attention** replaces the naive PyTorch implementation with a fused attention backend. **Latent space** uses the FLUX VAE [39] with 8× spatial compression and a patch size of 7 for 16 channels. **ANN** introduces approximate nearest neighbor search with clustering , reducing complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N^{3/2})$. All methods improve efficiency.

### 4.1. Unconditional Single Image Generation

We show examples of unconditional image sampling using our coarse-to-fine procedure in Figure 4. Qualitatively, our approach generates images with a similar appearance to the baselines, including methods that require hours of training. Additional qualitative results are included in Sec. S5.

We provide quantitative results in Table 1 on the


Figure 5. High-resolution generation. The input image is 308 MP, and we generate an image of size $14336 \times 70080$ (1 GP) in only 13.9 minutes (NVIDIA RTX A6000 PRO) by incorporating the three proposed acceleration techniques (see Sec. S3.5). Specifically, we use $T = 20$ sampling steps with $\eta = 1$ and ANN with $k = 5$. *Image: Duncan Rawlinson, CC BY-NC 2.0.*

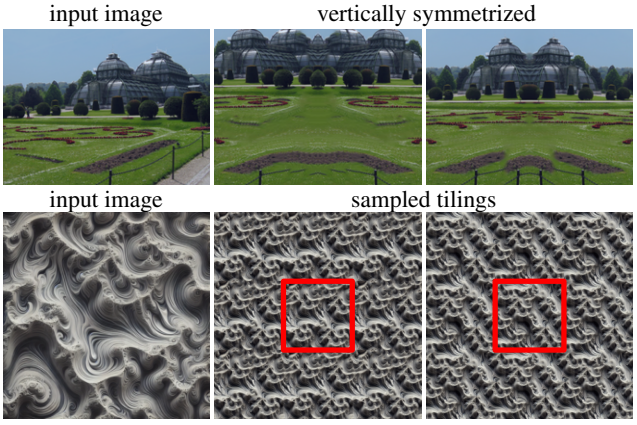


Figure 6. Image symmetrization. By enforcing constraints in the diffusion process, our approach can generate new images with horizontal or vertical symmetry. It can also create images that tile together without seams (the red box indicates the sampled image, which we use to create the $3\times3$ tiling shown).

single image Fréchet Inception distance (SIFID) [58], no-reference image quality metrics (NIQE, NIMA, and MUSIQ [37, 45, 67]), pixel diversity, and LPIPS diversity [38]. The latter metrics assess the diversity of generated images by measuring the average standard deviation of generated pixels across sampled images, and the average LPIPS distance between sampled image pairs. Each metric is computed using 50 generated samples, and we report the mean and standard deviation across 15 different input images.

Our approach achieves improved SIFID compared to trained single-image diffusion models. While GPNN and GPDM achieve the best SIFID scores, they often produce outputs that are nearly identical to the input image (see Figure S7 and Table S1). We achieve comparable image quality to prior methods (NIQE, NIMA, MUSIQ), while obtaining the highest diversity among all methods (LPIPS distance, pixel diversity). Overall, we achieve similar or better quality relative to other diffusion-based models with significantly lower computational overhead.

In Figure 5, we show generation of a 1 GP resolution image in 834 s by combining fused attention, latent space denoising, and approximate nearest neighbors (ANN). Additional gigapixel examples are provided in Figure S9. We find that high-resolution generation benefits from stochastic

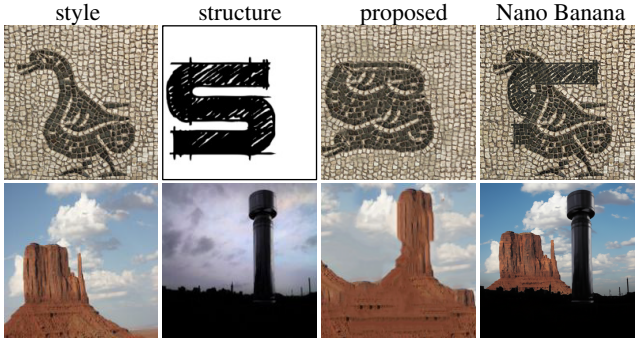


Figure 7. Structural analogies [4]. Our approach combines the style of one image and the structure of another image to generate a new image that combines both properties. Nano Banana Pro [25] preserves neither the structure nor the input patch distribution.

denoising ($\eta > 0$) and using ANN only at finer scales of diffusion sampling after low-spatial-frequency image components have converged (see Sec. S3.5). We assess how inference times scale with image resolution in Table 2; at 16 MP resolution, the proposed acceleration techniques achieve a $>1000\times$ speedup relative to a naive implementation.

### 4.2. Applications

**Image retargeting.** We show image retargeting results in Figures S10 and S11. Given a specified target resolution, our method first downsamples the input image to the resolution of the coarsest scale ($S$). Then, we resize the coarse-resolution image to the desired aspect ratio and use it to initialize $\hat{\mathbf{x}}_{0,S}$. After, we run coarse-to-fine image sampling to recover the retargeted image.

**Symmetrization.** We add constraints during the diffusion process to sample images that are vertically symmetric or that tile together without seams (see Figure 6). We create vertical symmetry by flipping and copying one half of the denoised image to the other half of the image after every denoising timestep during the generation process. To create tileable images, we run three separate denoising passes at every timestep. In the first pass, we directly denoise the image. In the second and third passes, we circularly shift the image along the horizontal or vertical dimensions by half the image width or height before denoising. After denoising, we shift the images back to their original configuration,

and we blend the results of each denoising pass to ensure that the boundaries of the image are tileable. We show results of this procedure for multiple different scenes in Figure 6; see Secs. S4 and S5 for additional details and results.

**Structural analogies.** In Figure 7, we show images created by structural analogy, which seeks to apply the "style" of one image to the "structure" of another image. Large diffusion models struggle with this task and fail to preserve the distribution of patches from the style image (e.g., Nano Banana Pro [25]; see Sec. S5 for implementation details).

This task is accomplished by first downsampling the structure image to a coarse scale and running the single-scale DDIM inversion [15, 47, 72] to a timestep $t'{=}T/10$ or $T/2$ (we use less noise for images with high-frequency structure). DDIM inversion converts a clean image into a corresponding noisy latent by applying Eqs. (3) to (6) in reverse time: at each step, we use the predicted $\hat{\mathbf{x}}_t$ and $\hat{\boldsymbol{\epsilon}}_t$ to deterministically produce a *noisier* sample $\mathbf{x}_{t+1} \leftarrow \alpha(t{+}1)\hat{\mathbf{x}}_t{+}\sigma(t{+}1)\hat{\boldsymbol{\epsilon}}_t$ with $\eta = 0$. Iterating the clean image from $t = 0$ to $t = t'$ yields an inverted noisy image $\mathbf{x}_{t'}$ that preserves the spatial structure of the original image, which is useful for editing tasks (see Sec. S4). Then, the inverted noisy image at the coarsest scale $S$ is used to initialize $\mathbf{x}_{t',S}$ for the coarse-to-fine image sampling procedure using the patches from the style image for denoising.

**Text-guided style transfer.** We combine our approach with pre-trained vision–language models to enable text-guided style transfer. We use the CLIP ViT-B/32 model [52] and a procedure similar to that of SinDDM for this task [38].

To start, we compute a noisy version of the input image via DDIM inversion, which we then use to initialize the reverse-diffusion process with the single-scale sampler (Algorithm 1) and CLIP-guided updates. Specifically, at each denoising step, we first compute the denoised image $\hat{\mathbf{x}}_t$, and the CLIP update rule is given as

$$\hat{\mathbf{x}}_{t,\text{CLIP}} \leftarrow \gamma \nabla_{\hat{\mathbf{x}}_t} \mathcal{L}_{\text{CLIP}} + \lambda \hat{\mathbf{x}}_t + (1-\lambda)\hat{\mathbf{x}}_{t+1,\text{CLIP}}, \quad (8)$$

where $\gamma$ is a parameter that controls the intensity of the CLIP guidance, and $\lambda$ is a momentum parameter that controls how much content to retain from the previous timestep after CLIP guidance. Using momentum helps to prevent image manipulations from being overridden by the denoising step [38]. The CLIP loss $\mathcal{L}_{\text{CLIP}}$ is the average cosine distance between a set of augmented text and image embeddings that we compute on the input image (see Sec. S4).

We show examples of text-guided style transfer in Figures 1 and S16. The proposed approach can be adapted to generate images corresponding to various styles and artists, and we show comparisons to SinDDM in Sec. S5.

### 4.3. Analysis of Hyperparameter Settings

We evaluate the sensitivity of the method to hyperparameters, including the number of diffusion timesteps, the patch size, and the value of $\rho$ (used to reassemble the image from patches $\mathbf{R}_\rho^{(i)}$), and we report the effect on the SIFID score. We compute metrics using the same 15 images as Table 1, but for computational expediency we report the mean and standard deviation over five generated samples per image.

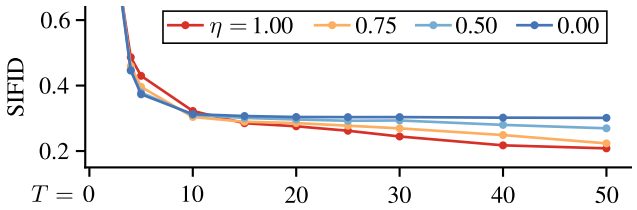

Figure 8. Plot of SIFID vs diffusion timesteps $T$ for coarse-to-fine image sampling across different $\eta$ values. The SIFID converges in roughly 10 timesteps.

| patch size | $\rho$ $\infty$ | 3.0 | 1.0 | 0.2 | 0.1 |
|---|---|---|---|---|---|
| 5 | $2.2 \pm 0.9$ | $2.2 \pm 0.9$ | $2.4 \pm 1.2$ | $2.7 \pm 1.9$ | $2.8 \pm 2.0$ |
| 7 | $2.7 \pm 1.4$ | $2.6 \pm 1.3$ | $2.1 \pm 1.0$ | $1.9 \pm 1.0$ | $1.8 \pm 1.2$ |
| 9 | $3.1 \pm 1.8$ | $3.1 \pm 1.8$ | $2.3 \pm 1.3$ | $1.6 \pm 0.9$ | $1.4 \pm 0.8$ |
| 11 | $3.5 \pm 2.1$ | $3.5 \pm 2.1$ | $2.4 \pm 1.6$ | $1.6 \pm 0.9$ | $1.3 \pm 0.8$ |
| 15 | $4.0 \pm 2.9$ | $3.9 \pm 2.9$ | $2.8 \pm 2.1$ | $1.6 \pm 1.3$ | $1.3 \pm 1.1$ |
| 19 | $4.2 \pm 3.5$ | $4.1 \pm 3.5$ | $3.0 \pm 2.8$ | $1.8 \pm 1.8$ | $1.5 \pm 1.5$ |
| 23 | $4.2 \pm 4.1$ | $4.2 \pm 4.0$ | $3.1 \pm 3.2$ | $1.9 \pm 2.3$ | $1.6 \pm 1.9$ |

Table 3. Analysis of the single-scale image sampling SIFID versus different patch sizes and values of $\rho$ (used in the operator $\mathbf{R}_\rho^{(i)}$ to assemble an image from patches). We find that smaller values of $\rho$ and patch sizes of around 11–15 pixels achieve the lowest SIFID.

Figure 8 plots the SIFID vs. diffusion timesteps for coarse-to-fine image sampling across different $\eta$ values. While increasing the number of diffusion timesteps improves the SIFID scores, there are diminishing returns for $T{>}10$. Higher $\eta$ values yield larger improvements in SIFID as $T$ increases, but at the cost of reduced sample diversity (see Table 1). We also plot the impact of patch size and $\rho$ on SIFID in Table 3 for single-scale image sampling. Based on these results, we choose a patch size of 15 and $\rho = 0.2$ for all our experiments.

## 5. Concluding Remarks

Recent diffusion models are trained on increasingly large datasets and have prohibitive computational costs. Counter to this trend, we explore reducing the size of the training dataset to the bare minimum—a single image. In this setting, closed-form denoising eliminates the need for hours of training and makes single-image generative modeling significantly more practical. Our work opens up multiple exciting directions: we envision improving the efficiency further to enable real-time generation, developing off-the-shelf single-image priors for solving inverse problems [11], and introducing new diffusion priors that leverage the internal structure of multiple images simultaneously. We discuss these extensions in more detail in Sec. S6.

**Acknowledgments.** DBL and KNK acknowledge support of NSERC under the RGPIN program. DBL also acknowledges support from the Canada Foundation for Innovation and the Ontario Research Fund.

## References

[1] Arpit Bansal, Eitan Borgnia, Hong-Min Chu, Jie S. Li, Hamid Kazemi, Furong Huang, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Cold diffusion: Inverting arbitrary image transforms without noise. In *Proc. NeurIPS*, 2023. 27

[2] Omer Bar-Tal, Dolev Ofri-Amar, Rafail Fridman, Yoni Kasten, and Tali Dekel. Text2LIVE: Text-driven layered image and video editing. In *Proc. ECCV*, 2022. 12

[3] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 28(3):24, 2009. 1, 2

[4] Saguy Benaim, Ron Mokady, Amit Bermano, and Lior Wolf. Structural analogy from a single image pair. *Computer Graphics Forum*, 40(1):249–265, 2021. 6, 7

[5] Quentin Bertrand, Anne Gagneux, Mathurin Massias, and Rémi Emonet. On the closed-form of flow matching: Generalization does not arise from target stochasticity. In *Proc. NeurIPS*, 2025. 3, 5

[6] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *Proc. ICLR*, 2019. 1

[7] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. A non-local algorithm for image denoising. In *Proc. CVPR*, 2005. 1, 2, 3

[8] Sam Buchanan, Druv Pai, Yi Ma, and Valentin De Bortoli. On the edge of memorization in diffusion models. In *Proc. NeurIPS*, 2025. 2, 3, 5

[9] Peter Burt and Edward Adelson. A multiresolution spline with application to image mosaics. *ACM Trans. Graph.*, 2 (4):217–236, 1983. 5

[10] Pierrick Chatillon, Julien Rabin, and David Tschumperlé. NIFTY: a non-local image flow matching for texture synthesis. *arXiv preprint arXiv:2509.22318*, 2025. 2

[11] Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *Proc. ICLR*, 2023. 8, 27

[12] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Trans. Image Process.*, 16(8):2080–2095, 2007. 1, 2

[13] Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *Proc. ICLR*, 2024. 5, 6, 8

[14] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Proc. NeurIPS*, 2022. 2, 5, 6, 8

[15] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *Proc. NeurIPS*, 2021. 1, 2, 8

[16] Sander Dieleman. Diffusion is spectral autoregression. https://sander.ai/2024/09/02/spectral-autoregression.html, 2024. Blog post. 27

[17] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The Faiss library. *IEEE Trans. Big Data*, 12(2):346–361, 2025. 9

[18] Alexei A Efros and William T Freeman. Image quilting for texture synthesis and transfer. In *Proc. SIGGRAPH*, 2001. 1, 2

[19] Alexei A Efros and Thomas K Leung. Texture synthesis by non-parametric sampling. In *Proc. ICCV*, 1999. 1, 2

[20] Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. Image Process.*, 15(12):3736–3745, 2006. 2

[21] Ariel Elnekave and Yair Weiss. Generating natural images with direct patch distributions matching. In *Proc. ECCV*, 2022. 6, 1

[22] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Proc. ICML*, 2024. 2

[23] Daniel Glasner, Shai Bagon, and Michal Irani. Super-resolution from a single image. In *Proc. ICCV*, 2009. 1

[24] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Commun. ACM*, 63(11):139–144, 2020. 1

[25] Google DeepMind. Introducing Nano Banana Pro (Gemini 3 Pro Image). https://blog.google/innovation-and-ai/products/nano-banana-pro/, 2025. 7, 8, 16

[26] Niv Granot, Ben Feinstein, Assaf Shocher, Shai Bagon, and Michal Irani. Drop the GAN: In defense of patches nearest neighbors as single image generative models. In *Proc. CVPR*, 2022. 1, 6, 19

[27] Aaron Hertzmann, Charles Jacobs, Nuria Oliver, Brian Curless, and David Salesin. Image analogies. In *Proc. SIGGRAPH*, 2001. 2

[28] Tobias Hinz, Matthew Fisher, Oliver Wang, and Stefan Wermter. Improved techniques for training single-image GANs. In *Proc. WACV*, 2021. 1, 2

[29] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proc. NeurIPS*, 2020. 1, 2, 3

[30] Emiel Hoogeboom and Tim Salimans. Blurring diffusion models. In *Proc. ICLR*, 2023. 27

[31] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *J. Mach. Learn. Res.*, 6(4):695–709, 2005. 2

[32] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proc. STOC*, 1998. 2, 5

[33] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Trans. Big Data*, 7(3):535–547, 2019. 9

[34] Nebojsa Jojic, Brendan J. Frey, and Anitha Kannan. Epitomic analysis of appearance and shape. In *Proc. ICCV*, 2003. 1

[35] Mason Kamb and Surya Ganguli. An analytic theory of creativity in convolutional diffusion models. In *Proc. ICML*, 2025. 3, 5

[36] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Proc. NeurIPS*, 2022. 2, 3, 5

[37] Junjie Ke, Qifei Wang, Yilin Wang, Peyman Milanfar, and Feng Yang. MUSIQ: Multi-scale image quality transformer. In *Proc. ICCV*, 2021. 6, 7

[38] Vladimir Kulikov, Shahar Yadin, Matan Kleiner, and Tomer Michaeli. SinDDM: A single image denoising diffusion model. In *Proc. ICML*, 2023. 1, 2, 6, 7, 8, 12, 13, 16, 17

[39] Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, et al. FLUX. 1 Kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint arXiv:2506.15742*, 2025. 5, 6, 9

[40] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *Proc. ICLR*, 2023. 6

[41] Artem Lukoianov, Chenyang Yuan, Justin Solomon, and Vincent Sitzmann. Locality in image diffusion models emerges from data statistics. In *Proc. NeurIPS*, 2025. 2, 3, 5

[42] Or Madar and Ohad Fried. Tiled diffusion. In *Proc. CVPR*, 2025. 1, 2

[43] Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro, and Andrew Zisserman. Non-local sparse models for image restoration. In *Proc. ICCV*, 2009. 1

[44] Niloy J Mitra, Leonidas J Guibas, and Mark Pauly. Symmetrization. *ACM Trans. Graph.*, 26(3):63–es, 2007. 2

[45] Anish Mittal, Rajiv Soundararajan, and Alan C Bovik. Making a "completely blind" image quality analyzer. *IEEE Signal Process. Lett.*, 20(3):209–212, 2012. 6, 7

[46] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *Proc. ICLR*, 2018. 1, 2

[47] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proc. CVPR*, 2023. 8

[48] Soumik Mukhopadhyay, Prateksha Udhayanan, and Abhinav Shrivastava. Scale space diffusion. *arXiv preprint arXiv:2603.08709*, 2026. 27

[49] Matthew Niedoba, Berend Zwartsenberg, Kevin Patrick Murphy, and Frank Wood. Towards a mechanistic explanation of diffusion model generalization. In *Proc. ICML*, 2025. 2, 3, 5, 27

[50] Yaniv Nikankin, Niv Haim, and Michal Irani. SinFusion: Training diffusion models on a single image or video. In *Proc. ICML*, 2023. 1, 2, 6

[51] Vardan Papyan and Michael Elad. Multi-scale patch-based image restoration. *IEEE Trans. Image Process.*, 25(1):249–261, 2015. 2

[52] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *Proc. ICML*, 2021. 1, 2, 8

[53] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with VQ-VAE-2. In *Proc. NeurIPS*, 2019. 1, 2

[54] Severi Rissanen, Markus Heinonen, and Arno Solin. Generative modelling with inverse heat dissipation. In *Proc. ICLR*, 2023. 27

[55] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proc. CVPR*, 2022. 2, 5

[56] Christopher Scarvelis, Haitz Sáez de Ocáriz Borde, and Justin Solomon. Closed-form diffusion models. *Trans. Mach. Learn. Res.*, 2025. 2, 3, 5

[57] Jay Shah, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. Flashattention-3: Fast and accurate attention with asynchrony and low-precision. In *Proc. NeurIPS*, 2024. 6, 8

[58] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. SinGAN: Learning a generative model from a single natural image. In *Proc. ICCV*, 2019. 1, 2, 6, 7

[59] Assaf Shocher, Nadav Cohen, and Michal Irani. Zero-shot super-resolution using deep internal learning. In *Proc. CVPR*, 2018. 1

[60] Assaf Shocher, Shai Bagon, Phillip Isola, and Michal Irani. InGAN: Capturing and remapping the "DNA" of a natural image. In *Proc. ICCV*, 2019. 2

[61] Sivic and Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proc. ICCV*, 2003. 5, 9

[62] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proc. ICML*, 2015. 3

[63] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *Proc. ICLR*, 2021. 1, 2, 3

[64] Kiwhan Song, Jaeyeon Kim, Sitan Chen, Yilun Du, Sham Kakade, and Vincent Sitzmann. Selective underfitting in diffusion models. *arXiv preprint arXiv:2510.01378*, 2025. 2, 3, 5

[65] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Proc. NeurIPS*, 2019. 3

[66] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *Proc. ICLR*, 2021. 3

[67] Hossein Talebi and Peyman Milanfar. NIMA: Neural image assessment. *IEEE Trans. Image Process.*, 27(8):3998–4011, 2018. 6, 7

[68] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proc. NeurIPS*, 2017. 2, 7

[69] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Comput.*, 23(7):1661–1674, 2011. 2

[70] Weilun Wang, Jianmin Bao, Wengang Zhou, Dongdong Chen, Dong Chen, Lu Yuan, and Houqiang Li. SinDiffusion: Learning a diffusion model from a single natural image. *IEEE Trans. Pattern Anal. Mach. Intell.*, 47(5):3412–3423, 2025. 2, 6

[71] Li-Yi Wei and Marc Levoy. Texture synthesis over arbitrary manifold surfaces. In *Proc. SIGGRAPH*, 2001. 1

[72] Biao Zhang, Jing Ren, and Peter Wonka. Geometry distributions. In *Proc. ICCV*, 2025. 8

[73] Maria Zontak and Michal Irani. Internal statistics of a single natural image. In *Proc. CVPR*, 2011. 1, 2

[74] Daniel Zoran and Yair Weiss. From learning models of natural image patches to whole image restoration. In *Proc. ICCV*, 2011. 1, 2, 3