
Flow Matching in Latent Space

Quan Dao *
 VinAI Research
 v.quandm7@vinai.io

Hao Phung *
 VinAI Research
 v.haopt12@vinai.io

Binh Nguyen
 Department of Mathematics,
 National University of Singapore
 tuanbinhs@gmail.com

Anh Tran
 VinAI Research
 v.anhtt152@vinai.io

Abstract

Flow matching is a recent framework to train generative models that exhibits impressive empirical performance while being relatively easier to train compared with diffusion-based models. Despite its advantageous properties, prior methods still face the challenges of expensive computing and a large number of function evaluations of off-the-shelf solvers in the pixel space. Furthermore, although latent-based generative methods have shown great success in recent years, this particular model type remains underexplored in this area. In this work, we propose to apply flow matching in the latent spaces of pretrained autoencoders, which offers improved computational efficiency and scalability for high-resolution image synthesis. This enables flow-matching training on constrained computational resources while maintaining their quality and flexibility. Additionally, our work stands as a pioneering contribution in the integration of various conditions into flow matching for conditional generation tasks, including label-conditioned image generation, image inpainting, and semantic-to-image generation. Through extensive experiments, our approach demonstrates its effectiveness in both quantitative and qualitative results on various datasets, such as CelebA-HQ, FFHQ, LSUN Church & Bedroom, and ImageNet. We also provide a theoretical control of the Wasserstein-2 distance between the reconstructed latent flow distribution and true data distribution, showing it is upper-bounded by the latent flow matching objective. Our code will be available at <https://github.com/VinAIResearch/LFM.git>.

1 Introduction

Generative AI is one of the most active fields of AI research in recent years, with a rapid growth of technologies and a blooming of applications. In the context of computer vision, Generative AI not only captures data distribution of a target domain, allowing producing novel data without or with conditional inputs such as text, sketches, semantic maps, or images, but it also brings in various downstream applications such as image restoration, image translation, image segmentation, and more. While GANs paved the way for high-fidelity image generation [3], diffusion models [9, 22, 29, 60, 76] have taken the lead with better mode coverage and excellent support on different conditional inputs.

However, diffusion models are not yet the perfect since it requires a long sampling time. A line of methods [11, 24, 40, 58, 79] is dedicated to improving sampling efficiency to address their slow training and testing speed. Despite some successes with these methods, diffusion models still suffer from other issues related to slow convergence and sub-optimal probability paths, which can hinder

* equal contribution

the performance of existing methods on large-scale datasets. Besides fixing the flaws of diffusion models, another promising research direction is to design a more efficient AI generation technique that inherits their advantages but does not suffer from mentioned problems.

Flow matching framework [1, 38, 39], a newly introduced line of research, show positive initial results and promise to become a better alternative for diffusion models. The essence of this method lies in learning an ordinary differential equation that follows a path from the source distribution to the target distribution. It has been pointed out that ODE-based generative models have favorable properties compared to SDE-based ones in diffusion models, most notably easier to train and faster to sample, due to lower curvature in the generative trajectories [34, 39]. Particularly, in [38], flow matching models beat the standard diffusion method DDPM on unconditional image generation tasks in terms of negative log-likelihood, sample quality, and inference speed. Despite its initial promising results, flow matching (FM) research is still in the infancy stage. In this paper, we aim to boost its development by adapting the key successful features of diffusion models into these systems.

First, although state-of-the-art diffusion models flourish via executing the diffusion process in the latent space, all FM models still perform path matching in pixel space. We propose to apply flow matching in the latent spaces of pretrained autoencoders, which offers improved computational efficiency and scalability for high-resolution image synthesis.

Second, one of the key successes of diffusion models is the versatile support of different conditional input types. Prior FM models are developed only for unconditional tasks. We redesign the velocity field network to support conditional inputs, such as class labels, segmentation masks, or images, and demonstrate this ability via different downstream tasks. To our understanding, our work are also the first to integrate classifier-free guidance in the sampling process of flow matching models.

Finally, we demonstrate that the Wasserstein-2 distance between the reconstructed latent flow distribution and true data distribution is upper-bounded by the latent flow matching objective. Additionally, our bound emphasizes the importance of choosing good backbone autoencoder architectures.

In summary, our paper offers the following contributions:

- To the best of our knowledge, our work is the first to introduce flow matching in latent space for high-resolution datasets, resulting in faster training time and improved computational efficiency.
- Our work is also the first to integrate conditional inputs into flow matching models and support class conditional image generation with classifier-free velocity field guidance. We demonstrate this ability via various downstream applications, including label-conditioned image generation, image inpainting, and mask-to-image generation.
- We provide a theoretical analysis (Wasserstein bound) on the tradeoff of performing flow matching of latent representations.
- Extensive experiments on various tasks demonstrate our method’s superiority in quantitative and qualitative results, narrowing the gap with state-of-the-art diffusion methods. Our approach is expected to facilitate the development of flow matching compared to other generative models in real-world applications.

2 Related works

Diffusion models [57] is an emergent type of generative framework that achieves unprecedented performance in multiple fields, including image generation [18, 44, 58, 9, 46], video generation [21, 17], 3D generation [41], and language generation [10, 37]. This type of model can be viewed as a score-based model [61–63, 69], as it relies on Stochastic Differential Equations (SDEs) to find trajectories from a tractable distribution (such as simple Gaussian noise) to the data distribution. However, these models typically require a significant number of function evaluations, ranging from hundreds to thousands, to generate a sample. This process can be slow and computationally inefficient for real-world applications. Hence, recent research has focused on improving the sampling efficiency by either modifying the diffusion process [58, 79, 26, 59] or using generative adversarial training [76, 48]. Another direction is to have instead shifted the focus to apply diffusion generative model in latent space to reduce the compute cost, benefiting from the compact dimension of latent representations learned by an autoencoder [51, 67, 56]. Still, diffusion models have some drawbacks

related to training convergence and sub-optimal trajectories, which might adversely affect their training time and overall performance. Motivating from [51, 67, 56], our work is also performed in the latent space of a pre-trained autoencoder. To the best of our knowledge, we are the first to leverage latent representation for flow-matching, specifically targeting high-resolution image generation.

Continuous normalizing flows (CNFs) [7, 15], unlike SDE-based methods, learn a deterministic mapping from the source distribution to the target distribution via neural Ordinary Differential Equation. However, previous framework for the training of CNFs are computational prohibitive and hard to scale up due to the need of solving ODEs at each iteration. As such, no existing methods are on par with the SDE-based models in the literature in terms of performance. Inspired by the per-sample optimization approach of score-based diffusion models, flow matching (FM) [1, 38, 39] was recently introduced as a simulation-free technique for training CNFs. It offers the utilization of flexible probability paths to efficiently training CNFs that transport noise samples into data samples. Moreover, based on optimal transport theory, [38, 39] argues that using training a constant velocity field result in simpler training objective, compared to the high-curvature probability path in diffusion models. The attractive properties of FM lead to a current trend in extending this method to more task-specific applications [6, 35, 75] and faster sampling [34, 49, 83].

However, a main drawback of current works on FM is that they are not yet ready for high-resolution image synthesis.

Our approach represents one of the pioneering attempts to thoroughly integrate and study latent representations for flow-matching models, with the aim of enhancing both scalability and performance.

Class-conditional generation has attained significant advancements in recent years, primarily driven by the success of diffusion models [9, 19, 82]. ADM [9], for instance, utilizes the gradient of a pre-trained classifier w.r.t. the input sample to update the denoising process. Notably, [20] introduced a simple yet elegant technique known as classifier-free guidance. This approach effectively balances the trade-off between the quality and diversity of generated samples without requiring a pretrained classifier. Besides, such a technique has made significant contributions to the advancement of conditional generation in various domains, such as text-to-image generation [50, 52] and text-to-video generation [17, 55]. However, the application of flow-based models in class-conditional generation has not yet been explored.

To tackle this gap, we incorporate classifier-free guidance from [20] for latent flow matching, which plays a pivotal role in enhancing the performance of conditional models. However, it is important to note that while we utilize the same technique, it differs from traditional generative frameworks in terms of the target being estimated. In flow matching, the sampling process is driven by velocity rather than noise, resulting in a distinct approach to conditional generation.

3 Background

Given access to empirical observations of data distribution $\mathbf{x}_0 \sim p_0$ and noise distribution $\mathbf{x}_1 \sim p_1$ (which usually is Gaussian), the goal of flow matching framework is to estimate a coupling $\pi(p_0, p_1)$ that describes the evolution between those two distributions. This objective could be formulated as solving an ordinary differential equation:

$$d\mathbf{x}_t = v(\mathbf{x}_t, t)dt, \quad (1)$$

on time $t \in [0, 1]$, where the velocity $v : \mathbb{R}^d \times [0, 1] \rightarrow \mathbb{R}^d$ is set to drive the flow from p_0 to p_1 .

We can parameterize the drift (velocity) by $v_\theta(\mathbf{x}_t, t)$ with an expressive learner (a neural net, for example) and estimate θ by solving a simple least square regression problem:

$$\hat{\theta} = \operatorname{argmin}_{\theta} \mathbb{E}_{t, \mathbf{x}_t} [\|v(\mathbf{x}_t, t) - v_\theta(\mathbf{x}_t, t)\|_2^2]. \quad (2)$$

With this estimation, we can do backward sampling by taking $\hat{\mathbf{x}}_0 = \int_1^0 v_\theta(\mathbf{x}_t, t)dt$ since we have access to the noise $\mathbf{x}_1 \sim p_1$, and solve the integration with numerical integrators. Formally, the ODE in (1) is called the Lagrangian flow, which describes the dynamic of point clouds. We have an equivalent view in Eulerian sense – a continuity equation that describes the dynamic of the measure p_t [2]:

$$\partial_t p_t = -\operatorname{div}(v(\mathbf{x}, t)p_t), \quad (3)$$

where div is the divergence operator.

Equation (2) allows for the inclusion of different options for $v(\mathbf{x}, t)$, which underscores the flexibility nature of the flow matching framework. In the following, we provide a brief overview of two widely-used variations of $v(\mathbf{x}, t)$: the probability flow ODE and the constant velocity ODE.

Probability flow ODE Introduced in [63], the velocity in probability flow ODE has the form

$$v_t(\mathbf{x}_t, t) = f(\mathbf{x}_t, t) - \frac{g_t^2}{2} \nabla \log p_t, \quad (4)$$

where $\nabla \log p_t$ is the score function, $f(\mathbf{x}_t, t)$ and g_t are the drift and diffusion coefficients of the equivalent form of the forward generative diffusion process, a stochastic differential equation (SDE)

$$d\mathbf{x}_t = f(\mathbf{x}_t, t)dt + g_t d\mathbf{w} \quad (5)$$

with \mathbf{w} the standard Wiener process (or Brownian motion). The flow matching loss then can be recasted as score matching loss [23] to estimate the score $\nabla \log p_t$, as suggested in [34, 83]. A common choice of the path \mathbf{x}_t is the Variance Preserving (VP) path:

$$\mathbf{x}_t \stackrel{\text{def.}}{=} \alpha_t \mathbf{x}_0 + (1 - \alpha_t^2)^{\frac{1}{2}} \mathbf{x}_1, \text{ where } \alpha_t = e^{-\frac{1}{2} \int_0^t \beta(s)ds}. \quad (6)$$

It is shown in [26] that using ODE sampler of the form (4) with path (6) can reduce sampling costs compared to sampling with discretization of the diffusion SDE in Equation (5).

Constant velocity ODE Liu et al. [39] pointed out that using a non-linear interpolation for \mathbf{x}_t in the VP probability flow ODE in (6) can result in an unnecessary increase in the curvature of generative trajectories. As a consequence, this can lead to reduced efficiency in the training and sampling of generative trajectories. Instead, they advocate the use of constant velocity ODE, where the path $\mathbf{x}_t \stackrel{\text{def.}}{=} (1 - t)\mathbf{x}_0 + t\mathbf{x}_1$ is the linear interpolation between \mathbf{x}_0 and \mathbf{x}_1 . This means the velocity drives the flow following the direction $v_t = \mathbf{x}_1 - \mathbf{x}_0$, and the flow matching loss in (2) becomes:

$$\hat{\theta} = \operatorname{argmin}_{\theta} \mathbb{E}_{t, \mathbf{x}_t} [\|\mathbf{x}_1 - \mathbf{x}_0 - v_{\theta}(\mathbf{x}_t, t)\|_2^2]. \quad (7)$$

We thus develop our method using the same framework: incorporating the use of linear interpolation for training and the ODE presented in Equation (1) for sampling.

4 Methodology

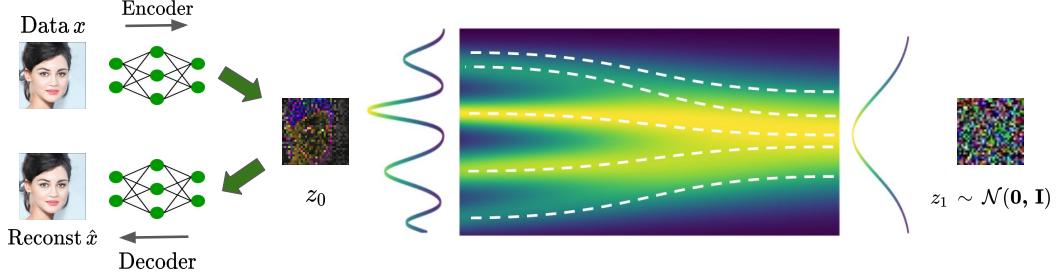


Figure 1: In our Latent Flow Matching framework’s training procedure, the input data \mathbf{x} is encoded to produce the latent representation \mathbf{z}_0 . A latent flow network is then trained to predict the velocity of the transformation from a standard normal distribution $p(\mathbf{z}_1) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ to the target latent distribution $p(\mathbf{z}_0)$. During the sampling process, random noise \mathbf{z}_1 is drawn from $p(\mathbf{z}_1)$, and the trained latent network is used to predict the velocity towards the target latent distribution $p(\mathbf{z}_0)$ through numerical integration. Finally, \mathbf{z}_0 is decoded to generate the corresponding image.

In this section, we first introduce the training and sampling procedure of our proposed framework, then present the classifier-free velocity field for class-conditional image generation.

4.1 Training and sampling procedure

We give a summary of our framework in [Figure 1](#). Given an input sample $\mathbf{x}_0 \sim p_0$, we encoded it into latent code $\mathbf{z}_0 = \mathcal{E}(\mathbf{x}_0) \in \mathbb{R}^{d/h}$ via a pretrained VAE encoder with KL regularization [30], where h is the downsampling rate of the encoder. In this latent space, the goal is to estimate a probability path to traverse from a random noise $\mathbf{z}_1 \sim \mathcal{N}(0, \mathbf{I})$ to the source distribution of latent codes \mathbf{z}_0 . The optimization of the velocity network can take advantage of the compact dimension of latent codes, where we use the same objective as in the vanilla flow matching with constant velocity in [Equation \(7\)](#):

$$\hat{\theta} = \operatorname{argmin}_{\theta} \mathbb{E}_{t, \mathbf{z}_t} [\|\mathbf{z}_1 - \mathbf{z}_0 - v_{\theta}(\mathbf{z}_t, t)\|_2^2]. \quad (8)$$

To train the conditional latent flow matching, we pass the conditional information \mathbf{c} to the flow model along with \mathbf{z}_t as:

$$\hat{\theta} = \operatorname{argmin}_{\theta} \mathbb{E}_{t, \mathbf{z}_t} [\|\mathbf{z}_1 - \mathbf{z}_0 - v_{\theta}(\mathbf{z}_t, \mathbf{c}, t)\|_2^2]. \quad (9)$$

With the sampling process, we test different numerical ODE integration schemes, such as Euler or a variant of Runge-Kutta using fixed or adaptive-step.

Following [26], we also create custom solvers by performing iterative sampling from discrete time intervals $\{t_i\}_{i=1}^N$ over N finite steps that works with any numerical ODE solvers given a velocity estimator v_{θ} .

Details on our selection of ODE solvers are shown in [Section 5.3](#), where we observed that adaptive solver such as `ropri5` [12] yields comparable results with Heun solver [82]. Still, for the sake of simplicity, we opt to use the adaptive-step one as the default algorithm for all of our experiments.

Finally, the desired sample $\hat{\mathbf{z}}$ is then decoded by a pretrained VAE decoder \mathcal{D} to yield the output image $\hat{\mathbf{x}} = \mathcal{D}(\hat{\mathbf{z}})$.

4.2 Conditional generation with classifier-free guidance for velocity field

Given the significance of class information \mathbf{c} in GANs [3, 42] and denoising diffusion models [9, 19, 20, 82] for conditional image synthesis, it is reasonable to incorporate class label information into the ODE sampler of our flow matching framework. In [9], the authors make use of a pre-trained classifier $p(\mathbf{c}|\mathbf{x})$ to guide the sampling process. However, this introduces additional complexity to the training pipeline due to the necessity of training an extra classifier. Moreover, this classifier must be trained on noisy interpolation path \mathbf{x}_t , making it generally hard to incorporate the pre-trained classifier seamlessly. As a result, we propose a formulation for classifier-free velocity field, inspired from [20].

First, we rewrite the generative SDE (5) using the associated Fokker-Planck equation:

$$\partial_t \tilde{p}_t = -\operatorname{div}(f(\mathbf{x}_t, t) \tilde{p}_t) + \frac{g_t^2}{2} \Delta \tilde{p}_t, \quad (10)$$

where $\tilde{p}_t \stackrel{\text{def.}}{=} p_t(\mathbf{x}_t | \mathbf{x}_0, \mathbf{c})$ is the conditional probability of a sample given its label \mathbf{c} . We can transform this into continuity equation of the form (3) by

$$\partial_t \tilde{p}_t = -\operatorname{div} \left[f(\mathbf{x}_t, t) p_t - \frac{g_t^2}{2} \frac{\nabla \tilde{p}_t}{\tilde{p}_t} \right] = -\operatorname{div} \left[\left(f(\mathbf{x}_t, t) - \frac{g_t^2}{2} \nabla \log \tilde{p}_t \right) \tilde{p}_t \right]. \quad (11)$$

This suggests the modified velocity vector has the form

$$\tilde{v}(\mathbf{x}_t, t) = f(\mathbf{x}_t, t) - \frac{g_t^2}{2} \nabla \log \tilde{p}_t = f(\mathbf{x}_t, t) - \frac{g_t^2}{2} (\nabla \log p_t + \nabla_x \log p(\mathbf{c}|\mathbf{x}_t)), \quad (12)$$

since we can factorize $\tilde{p}_t \stackrel{\text{def.}}{=} p_t(\mathbf{x}_t | \mathbf{x}_0, \mathbf{c}) = Z p_t(\mathbf{x}_t | \mathbf{x}_0) p(\mathbf{c}|\mathbf{x}_t)$, where Z is the normalizing constant that we can safely ignore after taking gradient of the log-likelihood. Using the same argument as in

[29] for variance preserving ODE, with the constant velocity path $\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\mathbf{x}_1$ that follows $\mathcal{N}(\mathbf{x}_t; (1-t)\mathbf{x}_0, t^2\mathbf{I})$, we have

$$\nabla_x \log p_t = \frac{-(\mathbf{x}_t - (1-t)\mathbf{x}_0)}{t^2}, \quad f(\mathbf{x}_t, t) = \frac{-\mathbf{x}_t}{1-t}, \text{ and } \frac{g_t^2}{2} = \frac{t}{1-t}.$$

Plugging these into (12), we get

$$\tilde{v}(\mathbf{x}_t, \mathbf{c}, t) = (\mathbf{x}_1 - \mathbf{x}_0) - \left(\frac{t}{1-t} \right) \nabla_x \log p(\mathbf{c}|\mathbf{x}_t) = v(\mathbf{x}_t, t) - \left(\frac{t}{1-t} \right) \nabla_x \log p(\mathbf{c}|\mathbf{x}_t). \quad (13)$$

This leads to the classifier-guided velocity field, similar to classifier-guided diffusion model in [9]) as

$$\tilde{v}(\mathbf{x}_t, \mathbf{c}, t) = v_\theta(\mathbf{x}_t, t) - \gamma \left(\frac{t}{1-t} \right) \nabla_x \log p(\mathbf{c}|\mathbf{x}_t), \quad (14)$$

where we add a scaling factor $\gamma > 0$ that controls the gradient strength. Finally, to arrive at the classifier-free velocity field formulation, first we add \mathbf{c} as an input of the velocity field estimation network, denoted $v_\theta(\mathbf{x}_t, \mathbf{c}, t)$. Then, factorizing again with Bayes rule the term $\nabla_x \log p(\mathbf{c}|\mathbf{x}_t)$ in (14) with $\nabla_x \log p(\mathbf{x}_t|\mathbf{x}_0, \mathbf{c}) - \nabla_x \log p(\mathbf{x}_t|\mathbf{x}_0)$. From (14), we have:

$$\begin{aligned} \tilde{v}(\mathbf{x}_t, \mathbf{c}, t) &= v_\theta(\mathbf{x}_t, t) - \gamma \left(\frac{t}{1-t} \right) \nabla_x \log p(\mathbf{c}|\mathbf{x}_t) \\ &= v_\theta(\mathbf{x}_t, t) - \gamma \left(\frac{t}{1-t} \right) (\nabla_x \log p(\mathbf{x}_t|\mathbf{x}_0, \mathbf{c}) - \nabla_x \log p(\mathbf{x}_t|\mathbf{x}_0)) \end{aligned} \quad (15)$$

Using (4), we have $-v_t(\mathbf{x}_t, \mathbf{c}, t) + f(\mathbf{x}_t, t) = \frac{g_t^2}{2} \nabla \log p_t(\mathbf{x}_t|\mathbf{c})$. Since \mathbf{x}_t is fixed and f, g are well-defined functions, we have:

$$\begin{aligned} \frac{g_t^2}{2} (\nabla \log p_t(\mathbf{x}_t|\mathbf{c}) - \nabla \log p_t(\mathbf{x}_t|\mathbf{c} = \emptyset)) &= -v_t(\mathbf{x}_t, \mathbf{c}, t) + f(\mathbf{x}_t, t) + v_t(\mathbf{x}_t, \mathbf{c} = \emptyset, t) - f(\mathbf{x}_t, t) \\ &= -v_t(\mathbf{x}_t, \mathbf{c}, t) + v_t(\mathbf{x}_t, \mathbf{c} = \emptyset, t) \end{aligned} \quad (16)$$

Substitute (16) into (15), we can approximate:

$$\tilde{v}_\theta(\mathbf{x}_t, \mathbf{c}, t) \approx v_\theta(\mathbf{x}_t, t) + \gamma(v_\theta(\mathbf{x}_t, \mathbf{c}, t) - v_\theta(\mathbf{x}_t, t)) = \gamma v_\theta(\mathbf{x}_t, \mathbf{c}, t) + (1-\gamma)v_\theta(\mathbf{x}_t, \mathbf{c} = \emptyset, t), \quad (17)$$

where $v_\theta(\mathbf{x}_t, t) = v_\theta(\mathbf{x}_t, \mathbf{c} = \emptyset, t)$ denote the unconditional velocity field trained with \mathbf{c} set to the empty token with a certain probability, akin to the philosophy of dropout in training neural networks. This enables joint training for the unconditional model and conditional model within a single neural network, which does not include integration of a pretrained classifier.

4.3 Theoretical analysis: bounding estimation error of latent flow matching

In order to perform theoretical analysis on performance of latent flow matching, we follow the settings of [4, 31]. We assume the decoder is deterministic and is parameterized by a function $g_\tau : \mathbb{R}^{d/h} \rightarrow \mathbb{R}^d$, the Gaussian encoder that parameterized the posterior $q^\phi(\mathbf{z} \mid \mathbf{x}) = \mathcal{N}(\mu_\phi(\mathbf{x}), \sigma_\phi^2(\mathbf{x})\mathbf{I})$. In other words, we use diagonal Gaussian reparameterization [30]:

$$\mathbf{z} = \mu_\phi(\mathbf{x}) + \eta \circ \sigma_\phi(\mathbf{x}) \quad \text{with } \eta \sim \mathcal{N}(0, \mathbf{I}), \quad (18)$$

where \circ denotes element-wise multiplication. We then have the following theorem.

Theorem 4.1 (Proof in Appendix A). *Let $p_0 : \mathbb{R}^d \rightarrow \mathbb{R}$ denote the ground truth distribution of data \mathbf{x}_0 , \hat{p}_0 the distribution of reconstructed sample $\hat{\mathbf{x}}$. Let the latent code \mathbf{z} be encoded following diagonal Gaussian reparameterization trick in (18). Let $v(\mathbf{z}_t, t)$ be the solution of the following continuity equation*

$$\partial_t q_t^\phi = -\text{div}(v(\mathbf{z}_t, t)q_t^\phi), \quad q_{t=1}^\phi = q_1^\phi,$$

and $\hat{v}(\hat{\mathbf{z}}_t, t)$ be an approximation of $v(\mathbf{z}_t, t)$ using the flow matching objective (8), or in other words, $\hat{v}(\hat{\mathbf{z}}_t, t)$ is the velocity field that defines \hat{q}_t , a solution of

$$\partial_t \hat{q}_t = -\text{div}(v(\hat{\mathbf{z}}_t, t)\hat{q}_t), \quad \hat{q}_{t=1} = q_1^\phi. \quad (19)$$

Assume that



Figure 2: Qualitative examples of our unconditional generation on FFHQ and LSUN Church & Bedroom and class conditional generation on ImageNet.



Figure 3: CelebA-HQ 512 and 256

1. *The estimated velocity \hat{v} is continuously differentiable in (\mathbf{z}, t) , and, Lipschitz in \mathbf{z} uniformly on $R^{d/h} \times [0, 1]$ with constant \hat{L} .*
2. *The deterministic decoder g_τ is Lipschitz continuous in \mathbf{z} with constant L_{g_τ} .*
3. *The distance from the quantity induced by taking $g_\tau(\mathbf{z}_0)$, i.e. a VAE reconstructed sample, with any real data point $\mathbf{x}_0 \sim p_0$ is defined by a quantity $\Delta_{f_\phi, g_\tau}(\mathbf{x}_0)$. In other words, $g_\tau(\mathbf{z}_0) = \mathbf{x}_0 + \Delta_{f_\phi, g_\tau}(\mathbf{x}_0)$, and we assume that it has bounded L_2 norm.*

Then we have the following bound on the squared Wasserstein-2 distance of the reconstructed distribution \hat{p}_0 and the ground truth p_0 :

$$\mathcal{W}_2^2(p_0, \hat{p}_0) \leq \|\Delta_{f_\phi, g_\tau}(\mathbf{x}_0)\|^2 + L_{g_\tau}^2 e^{1+2\hat{L}} \int_0^1 \int_{\mathbb{R}^{d/h}} |v(\mathbf{z}_t, t) - \hat{v}(\hat{\mathbf{z}}_t, t)|^2 d\mathbf{q}_t^\phi dt, \quad (20)$$

where $\Delta_{f_\phi, g_\tau}(\mathbf{x}_0)$ are constants that depend only on f_ϕ and g_τ , the stochastic encoder and decoder.

Theorem 4.1 shows that the minimizing flow matching objective (2) in latent space controls the Wasserstein distance between the target density p_0 and the reconstructed density \hat{p}_0 . We note that the bounded quantity on the LHS of (20) coincides with Fréchet inception distance (FID) [16], one of the most popular metrics for ranking generative models. This means latent flow matching is guaranteed to control this metric, given reasonable estimation of $\hat{v}(\mathbf{z}_t, t)$. Nonetheless, our theoretical analysis also suggests that the quality of matching samples in latent flow significantly depends on the constants that define the expressive capacities of the decoders and encoders. This highlights the tradeoff between computational speed and the quality of generated samples, which has been empirically observed in prior research on generative modeling in latent space: for example, both [51, 67] emphasize the critical role of utilizing good VAE architectures as backbone models for latent diffusion models.

5 Experiments

In our experiments, we use the pretrained VAE [30] from Stable Diffusion [51]. The VAE encoder has a downsampling factor of 8 given an RGB pixel-based image $\mathbf{x} \in \mathbb{R}^{h \times w \times 3}$, $\mathbf{z} = \mathcal{E}(\mathbf{x})$ has shape $\frac{h}{8} \times \frac{w}{8} \times 4$. All experiments are operated in the latent space.

5.1 Unconditional Generation

We conducted experiments on various datasets, including CelebA-HQ [25], FFHQ [28], LSUN Bedrooms [77], and LSUN Church [77], with the same resolution of 256×256 . We further benchmarked on CelebA-HQ with higher resolutions, including 512×512 . Additionally, we perform

Table 1: Results on CelebA-HQ and FFHQ datasets. (*) is measured on our machine

(a) CelebA-HQ 256 & 512					(b) FFHQ 256 × 256				
Model	NFE↓	FID↓	Recall↑	Time(s)↓	Model	NFE	FID↓	Recall↑	Time(s)↓
CelebA-HQ 256									
Ours (ADM)	85	5.82	0.41	3.42	Ours (ADM)	84	8.07	0.40	5.1
Ours (DiT L/2)	89	5.26	0.46	1.70	Ours (DiT L/2)	88	4.55	0.48	1.67
FM [38]*	128	7.34	-	-	LDM [51]	50	4.98	0.50	2.90*
LDM [51]	50	5.11	0.49	2.90*	ImageBART [13]	3	9.57	-	-
LSGM [67]	23	7.22	-	-	ProjectedGAN [53]	1	3.08	0.46	-
WaveDiff [48]	2	5.94	0.37	0.08	StyleGAN [28]	1	4.16	0.46	-
DDGAN [76]	2	7.64	0.36	-					
Score SDE [63]	4000	7.23	-	-					
CelebA-HQ 512									
Ours (ADM)	94	6.35	0.41	5.13					
WaveDiff [48]	2	6.40	0.35	0.59					
DDGAN [76]	2	8.43	0.33	1.49					

Table 2: Results on LSUN datasets. (*) is measured on our machine

(a) LSUN-Church 256 × 256				(b) LSUN-Bedrooms 256 × 256			
Model	FID↓	Recall↑	Time(s)↓	Model	FID↓	Recall↑	Time(s)↓
Ours (ADM)	7.7	0.39	3.5	Ours (ADM)	7.05	0.39	5.42
Ours (DiT L/2)	5.54	0.48	1.67	Ours (DiT L/2)	4.92	0.44	1.67
FM [38]	10.54	-	-	LDM [51]	2.95	0.48	2.90*
LDM [51]	4.02	0.52	3.79*	DDPM [18]	4.9	-	-
WaveDiff [48]	5.06	0.40	1.54	ImageBART [13]	5.51	-	-
DDPM [18]	7.89	-	-	ADM [9]	1.90	0.51	-
ImageBART [13]	7.32	-	-	PGGAN [25]	8.34	-	-
StyleGAN [28]	4.21	-	-	StyleGAN [28]	2.35	0.48	-
StyleGAN2 [27]	3.86	0.36	-	ProjectedGAN	1.52	0.34	-
ProjectedGAN [53]	1.59	0.44	-	DiffGAN [74]	1.43	0.58	-

our measurements on two distinct architectures: a CNN-based UNet (ADM) [9] and a transformer network (DiT) [46] for a comprehensive evaluation. To ensure a fair comparison, we employ two commonly used metrics for evaluating the performance of our proposed method: the Fréchet inception distance (FID) [16] and Recall [32]. We also measure the average running time and the number of function evaluations for single image generation to assess the computational efficiency of our method. In Table 1a, our method attains better FID scores than original flow matching FM on CelebA-HQ 256, at 5.82 and 5.26 for the CNN-based and transformer network architectures, respectively. On CelebA-HQ 512 and FFHQ, in Table 1, our method also shows superior performance than other diffusion counterparts. Furthermore, our method achieves comparable results with existing methods on the LSUN Church and Bedrooms datasets, as demonstrated in Table 2. In all experiments, it is clear that DiT produces better performance than ADM even though it is inherently designed for conditional image generation. Their qualitative results are provided at Figure 2 and Figure 3.

5.2 Conditional Generation

Conditional ImageNet [8]. Given the class label, our method can generate desired examples without relying on a pretrained classifier. As mentioned earlier, we select the DiT base backbone as the default network for this conditional task due to its comparability and relatively small size, making it well-suited for large-scale datasets like ImageNet. Our proposed classifier-free velocity field enables flexible control over class conditioning, leading to a significant improvement in image generation results. By incorporating this technique, we observe a remarkable reduction in the FID score for both the DiT and ADM variants. As shown in Table 3a, for the DiT variant, the FID score has

Table 3: Quantitative results on conditional tasks.

(a) Conditional ImageNet (256)				(b) Semantic-to-image		(c) Image Inpainting		
Model	FID \downarrow	Recall \uparrow	Params	Model	FID \downarrow	Model	FID \downarrow	P-IDS \uparrow
Ours (ADM)	16.71	0.56	387M	Ours (ADM)	26.3	Ours (ADM)	4.09	13.25
+ cfg=1.25	8.58	0.50	387M	Pix2PixHD [71]	38.5	MAT [36]	2.94	20.88
Ours (DiT B/2)	20.38	0.56	130M	SPADE [45]	29.2	LaMa [64]	3.98	8.82
+ cfg=1.25	8.77	0.48	130M	DAGAN [66]	29.1	ICT [70]	5.24	4.51
+ cfg=1.5	4.46	0.42	130M	CLADE [65]	30.6	MADF [84]	10.43	6.25
LDM-8 [51]	15.51	0.63	395M	SCGAN [73]	20.8	DeepFill v2 [78]	5.69	6.62
LDM-8-G	7.76	0.35	506M	SDM [72]	18.8	EdgeConnect [43]	5.24	5.61
DiT-B/2 [46]	43.47	-	130M					15.65

Table 4: Ablation study on both adaptive and fixed-step ODE solvers on CelebA-HQ (256).

Solver	NFE \downarrow	FID \downarrow	Time (s) \downarrow
Adaptive	89	5.26	1.7
Euler	90	5.51	1.78
Heun	50	5.26	1.83

decreased from 20.39 to 4.46 ($\text{cfg} = 1.5$) while for the ADM variant, the measure has similarly reduced from 16.71 to 8.58 ($\text{cfg} = 1.25$). Despite our network being significantly smaller than LDM, our classifier-free technique demonstrates superior performance gains compared to the larger model. Due to resource constraints, we are unable to scale our approach to the DiT network with a larger size, such as DiT-L/2 and DiT-XL/2, at this time. However, it is worth noting that if we were able to implement this scaling, it could potentially result in a further enlargement of the gap. This possibility could be explored in future development.

Inpainting Task. In the inpainting task, we must fill the missing region of input images with appropriate content. Following the training technique from [51], given a masked image \mathbf{x}_m and the corresponding mask \mathbf{m} , we pass the masked image through the encoder to get \mathbf{z}_m and resize the mask \mathbf{m} to $\bar{\mathbf{m}}$ which has the same shape with \mathbf{z}_m . To condition on the masked image and mask, we concatenate \mathbf{z}_t , \mathbf{z}_m , and $\bar{\mathbf{m}}$ and use it as input to flow matching network $v_\theta(\mathbf{z}_t, \mathbf{c} = [\mathbf{x}_m, \mathbf{m}], t) = v_\theta(\text{concat}[\mathbf{z}_t, \mathbf{z}_m, \bar{\mathbf{m}}], t)$. Our experiment is conducted on CelebA-HQ 256 [25] and follows the training and evaluation protocol of MAT [36]. We split data into 24,183 and 2,993 images for training and evaluation. To evaluate the inpainting performance, we utilize the following perceptual metrics FID [16], P-IDS [81], and U-IDS [80]. Table 3c shows that our method outperforms most existing inpainting methods and is comparable to LaMa [64]. We achieve the FID score of 4.09 using a very simple concatenation trick, and this result is not far from the SOTA number (2.94) produced by MAT [36], which is specifically designed for the inpainting task. Our qualitative result is shown in Figure 4b.

Semantic-to-image. For the semantic-to-image task, we have to synthesize photorealistic images given an input semantic layout \mathbf{m} with \mathcal{C} classes. Following the layout conditional technique from [51], the one-hot representation of \mathbf{m} is firstly interpolated to have the same spatial resolution with \mathbf{z}_t . We then train a conditional network ω_ϕ parameterized by ϕ that takes the interpolated one-hot representation of \mathbf{m} as input and reduces the number of channels from \mathcal{C} to the number channels of \mathbf{z}_t , $\mathbf{m}_c = \omega_\phi(\mathbf{m})$ where \mathbf{m}_c have the same shape with \mathbf{z}_t . Similar to inpainting task, the flow matching network $v_\theta(\mathbf{z}_t, \mathbf{c} = \mathbf{m}, t) = v_\theta(\text{concat}[\mathbf{z}_t, \mathbf{m}_c], t)$ takes the concatenation of \mathbf{m}_c and \mathbf{z}_t as input. The conditional network ω_ϕ is jointly trained along with the latent flow matching network v_θ . We evaluate our method on CelebA-HQ 256 [25] with 27,000 images for training and 3000 for evaluation FID score [16]. As shown in Figure 4a, our model can generate high-quality images, which are close to the ground truth images with regard to semantic layout. Our simple layout-to-image technique achieves an FID score of 26.3 (see Table 3b) in comparison with other methods such as SCGAN [73] using saliency guided technique and SDM [72] with special ADA normalization and classifier-free guidance.



(a) Semantic Image Synthesis

(b) Image Inpainting

Figure 4: **Image-conditional generation tasks.** For each task, we provide the conditional inputs (top), our results (middle), and the ground truth images (bottom).

5.3 Ablation on ODE solvers

As shown in Table 4, the Heun fixed-step solver achieves comparable performance to the adaptive solver with only 50 steps. However, it is slightly slower than the adaptive solver due to the requirement of an additional one-step forward to correct the estimated velocity. In this work, we have selected the adaptive solver instead of the fixed-steps solver, despite acknowledging the promising performance of the Heun solver. Further analysis of fix-steps solvers will be presented in Appendix B.

6 Conclusions

We introduce a novel approach, called latent flow matching for image generation, which attains competitive performance across various image synthesis tasks and helps bridge the gap of flow matching framework with existing diffusion models. By incorporating a classifier-free guidance for velocity field, we empower the seamless integration of class conditions into the process of flow matching for improving class-conditional image generation. Through extensive empirical evaluations, our method demonstrates the simplicity and effectiveness in enhancing the efficiency of flow matching models. These achievements prove the potential and applicability of our method in driving further generative AI research, particularly for a large-scale system like text-to-image synthesis.

Limitation and societal impact. As our method offers favorable performance across several tasks, it also poses some potential risks related to misinformation dissemination. Training such a model also emits a substantial amount of greenhouse gases.

References

- [1] Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *ICLR 2023, arXiv preprint arXiv:2209.15571*, 2022.
- [2] Luigi Ambrosio, Nicola Gigli, and Giuseppe Savaré. *Gradient flows: in metric spaces and in the space of probability measures*. Lectures in mathematics ETH Zürich. Birkhäuser, Boston, 2005.
- [3] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019.
- [4] Alexander Camuto, Matthew Willetts, Stephen Roberts, Chris Holmes, and Tom Rainforth. Towards a theoretical understanding of the robustness of variational autoencoders. In Arindam Banerjee and Kenji Fukumizu, editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 3565–3573. PMLR, 13–15 Apr 2021.

- [5] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11315–11325, 2022.
- [6] Ricky TQ Chen and Yaron Lipman. Riemannian flow matching on general geometries. *arXiv preprint arXiv:2302.03660*, 2023.
- [7] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.
- [8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, K. Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.
- [9] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.
- [10] Sander Dieleman. Diffusion language models, 2023.
- [11] Tim Dockhorn, Arash Vahdat, and Karsten Kreis. Genie: Higher-order denoising diffusion solvers. *arXiv preprint arXiv:2210.05475*, 2022.
- [12] J. R. Dormand and P. J. Prince. A family of embedded runge-kutta formulae. *Journal of Computational and Applied Mathematics*, 6:19–26, 1980.
- [13] Patrick Esser, Robin Rombach, Andreas Blattmann, and Björn Ommer. ImageBART: Bidirectional context with multinomial diffusion for autoregressive image synthesis. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.
- [14] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021.
- [15] Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.
- [16] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
- [17] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022.
- [18] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.
- [19] Jonathan Ho, Chitwan Saharia, William Chan, David J. Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *Journal of Machine Learning Research*, 23(47):1–33, 2022.
- [20] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.
- [21] Jonathan Ho, Tim Salimans, Alexey A. Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models. In *ICLR Workshop on Deep Generative Models for Highly Structured Data*, 2022.
- [22] Chin-Wei Huang, Jae Hyun Lim, and Aaron C Courville. A variational perspective on diffusion-based generative models and score matching. *Advances in Neural Information Processing Systems*, 34:22863–22876, 2021.

- [23] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.
- [24] Alexia Jolicoeur-Martineau, Ke Li, Rémi Piché-Taillefer, Tal Kachman, and Ioannis Mitliagkas. Gotta go fast when generating data with score-based models. *arXiv preprint arXiv:2105.14080*, 2021.
- [25] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018.
- [26] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.
- [27] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In *Advances in neural information processing systems*, 2020.
- [28] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.
- [29] Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.
- [30] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- [31] Abhishek Kumar and Ben Poole. On implicit regularization in β -vaes. In *International Conference on Machine Learning*, pages 5480–5490. PMLR, 2020.
- [32] Tuomas Kynkänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. *Advances in Neural Information Processing Systems*, 32, 2019.
- [33] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11523–11532, 2022.
- [34] Sangyun Lee, Beomsu Kim, and Jong Chul Ye. Minimizing trajectory curvature of ode-based generative models. *arXiv preprint arXiv:2301.12003*, 2023.
- [35] Lingxiao Li, Samuel Hurault, and Justin Solomon. Self-consistent velocity matching of probability flows. *arXiv preprint arXiv:2301.13737*, 2023.
- [36] Wenbo Li, Zhe Lin, Kun Zhou, Lu Qi, Yi Wang, and Jiaya Jia. Mat: Mask-aware transformer for large hole image inpainting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10758–10768, 2022.
- [37] Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35:4328–4343, 2022.
- [38] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.
- [39] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. February 2023.
- [40] 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. *arXiv preprint arXiv:2206.00927*, 2022.

- [41] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2837–2845, 2021.
- [42] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.
- [43] Kamyar Nazeri, Eric Ng, Tony Joseph, Faisal Z Qureshi, and Mehran Ebrahimi. Edgeconnect: Generative image inpainting with adversarial edge learning. *arXiv preprint arXiv:1901.00212*, 2019.
- [44] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.
- [45] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2337–2346, 2019.
- [46] William Peebles and Saining Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2022.
- [47] Gabriel Peyré and Marco Cuturi. Computational Optimal Transport: With Applications to Data Science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, February 2019. Publisher: Now Publishers, Inc.
- [48] Hao Phung, Quan Dao, and Anh Tran. Wavelet diffusion models are fast and scalable image generators. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.
- [49] Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, and Ricky Chen. Multisample flow matching: Straightening flows with minibatch couplings. *arXiv preprint arXiv:2304.14772*, 2023.
- [50] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.
- [51] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.
- [52] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Raphael Gontijo-Lopes, Burcu Karagol Ayan, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.
- [53] Axel Sauer, Kashyap Chitta, Jens Müller, and Andreas Geiger. Projected gans converge faster. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.
- [54] Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *ACM SIGGRAPH 2022 conference proceedings*, pages 1–10, 2022.
- [55] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022.
- [56] Abhishek Sinha, Jiaming Song, Chenlin Meng, and Stefano Ermon. D2c: Diffusion-decoding models for few-shot conditional generation. *Advances in Neural Information Processing Systems*, 34:12533–12548, 2021.
- [57] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

- [58] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.
- [59] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models, 2023.
- [60] Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. *Advances in Neural Information Processing Systems*, 34:1415–1428, 2021.
- [61] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.
- [62] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33:12438–12448, 2020.
- [63] 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*, 2020.
- [64] Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. Resolution-robust large mask inpainting with fourier convolutions. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2149–2159, 2022.
- [65] Zhentao Tan, Dongdong Chen, Qi Chu, Menglei Chai, Jing Liao, Mingming He, Lu Yuan, Gang Hua, and Nenghai Yu. Efficient semantic image synthesis via class-adaptive normalization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):4852–4866, 2021.
- [66] Hao Tang, Song Bai, and Nicu Sebe. Dual attention gans for semantic image synthesis. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1994–2002, 2020.
- [67] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.
- [68] Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2009.
- [69] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.
- [70] Ziyu Wan, Jingbo Zhang, Dongdong Chen, and Jing Liao. High-fidelity pluralistic image completion with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4692–4701, 2021.
- [71] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8798–8807, 2018.
- [72] Weilun Wang, Jianmin Bao, Wengang Zhou, Dongdong Chen, Dong Chen, Lu Yuan, and Houqiang Li. Semantic image synthesis via diffusion models. *arXiv preprint arXiv:2207.00050*, 2022.
- [73] Yi Wang, Lu Qi, Ying-Cong Chen, Xiangyu Zhang, and Jiaya Jia. Image synthesis via semantic composition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13749–13758, 2021.
- [74] Zhendong Wang, Huangjie Zheng, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. Diffusion-GAN: Training GANs with diffusion. In *The Eleventh International Conference on Learning Representations*, 2023.
- [75] Lemeng Wu, Dilin Wang, Chengyue Gong, Xingchao Liu, Yunyang Xiong, Rakesh Ranjan, Raghuraman Krishnamoorthi, Vikas Chandra, and Qiang Liu. Fast point cloud generation with straight flows. *arXiv preprint arXiv:2212.01747*, 2022.

- [76] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion gans. In *International Conference on Learning Representations*, 2022.
- [77] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.
- [78] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4471–4480, 2019.
- [79] Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. In *The Eleventh International Conference on Learning Representations*, 2023.
- [80] 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.
- [81] Shengyu Zhao, Jonathan Cui, Yilun Sheng, Yue Dong, Xiao Liang, Eric I Chang, and Yan Xu. Large scale image completion via co-modulated generative adversarial networks. *arXiv preprint arXiv:2103.10428*, 2021.
- [82] Guangcong Zheng, Shengming Li, Hui Wang, Taiping Yao, Yang Chen, Shouhong Ding, and Xi Li. Entropy-driven sampling and training scheme for conditional diffusion generation. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, pages 754–769, Cham, 2022. Springer Nature Switzerland.
- [83] Kaiwen Zheng, Cheng Lu, Jianfei Chen, and Jun Zhu. Improved techniques for maximum likelihood estimation for diffusion odes. *arXiv preprint arXiv:2305.03935*, 2023.
- [84] Manyu Zhu, Dongliang He, Xin Li, Chao Li, Fu Li, Xiao Liu, Errui Ding, and Zhaoxiang Zhang. Image inpainting by end-to-end cascaded refinement with mask awareness. *IEEE Transactions on Image Processing*, 30:4855–4866, 2021.

A Additional background and proofs

Before going into the proof of Theorem 4.1, we introduce additional background on optimal transport. Throughout this section, without loss of generality, we use $\|\cdot\|$ to denote the Euclidean norm in \mathbb{R}^d .

Definition A.1 (2-Wasserstein distance [68]). Given probability measures $\alpha, \beta \in \mathbb{R}^d$, we define the squared 2-Wasserstein distance as

$$\mathcal{W}_2^2(\alpha, \beta) \stackrel{\text{def.}}{=} \min_{\pi \in \Pi(\alpha, \beta)} \int_{\mathbb{R}^{d \times d}} \|\mathbf{x} - \mathbf{y}\|^2 d\pi(\mathbf{x}, \mathbf{y}), \quad (21)$$

where $\Pi(\alpha, \beta)$ is the set of couplings of α, β , that is, joint distributions on $\mathbb{R}^{d \times d}$ whose marginals are α, β , respectively. The distance is finite as long as α and β belong to the space $\mathcal{P}_2(\mathbb{R}^d)$ of probability measures over \mathbb{R}^d with finite second moments.

Note that technically, (21) is known as the Kantorovich formulation of optimal transport. In general, finding the 2-Wasserstein distance between two arbitrary probability measures involves solving optimization scheme, possibly with entropic regularization [47].

We also need the following result for the proof of our theoretical analysis.

Proposition A.2 (Proposition 3 in [1]). *Let p_t be the density of \mathbf{x}_t the interpolation path, defined by the dynamic in (3). Given a velocity field $\hat{v}(\mathbf{x}_t, t)$, we define \hat{p}_t as the solution of the initial value follow problem*

$$\partial_t \hat{p}_t = -\operatorname{div}(\hat{v}(\mathbf{x}, t) \hat{p}_t), \quad \hat{p}_0 = p_0. \quad (22)$$

Assume that $\hat{v}_t(\mathbf{x}, t)$ is continuously differentiable in (\mathbf{x}) and Lipschitz in \mathbf{x} , uniformly on $(\mathbf{x}, t) \in \mathbb{R}^d \times [0, 1]$ with Lipschitz constant \hat{L} . Then we have the following bound on the square of the W_2 distance between p_1 and \hat{p}_1

$$W_2^2(p_1, \hat{p}_1) \leq e^{1+2\hat{L}} \int_0^1 \int_{\mathbb{R}^d} \|v_t(\mathbf{x}_t, t) - \hat{v}_t(\mathbf{x}_t, t)\|^2 p_t d\mathbf{x} dt. \quad (23)$$

Proof of Theorem 4.1. By Definition A.1, we have

$$\mathcal{W}_2^2(p_0, \hat{p}_0) \stackrel{\text{def.}}{=} \min_{\pi \in \Pi(\alpha, \beta)} \int_{\mathbb{R}^{d \times d}} \|\mathbf{x}_0 - \hat{\mathbf{x}}_0\|^2 d\pi(\mathbf{x}_0, \hat{\mathbf{x}}_0) \quad (24)$$

$$= \min_{\pi \in \Pi(\alpha, \beta)} \int_{\mathbb{R}^{d \times d}} \|g_\tau(\mathbf{z}_0) - \Delta_{f_\phi, g_\tau}(\mathbf{x}) - \hat{\mathbf{x}}_0\|^2 d\pi(g_\tau(\mathbf{z}_0) - \Delta_{f_\phi, g_\tau}(\mathbf{x}), \hat{\mathbf{x}}_0) \quad (25)$$

$$= \|\Delta_{f_\phi, g_\tau}(\mathbf{x})\|^2 + \min_{\pi \in \Pi(\bar{\alpha}, \beta)} \int_{\mathbb{R}^{d \times d}} \|g_\tau(\mathbf{z}_0) - g_\tau(\hat{\mathbf{z}}_0)\|^2 d\pi(g_\tau(\mathbf{z}_0), g_\tau(\hat{\mathbf{z}}_0)). \quad (26)$$

where we use the assumption that the stochastic reconstruction $g_\tau(\mathbf{z}_0)$ is a distance $\Delta_{f_\phi, g_\tau}(\mathbf{x})$ away to the real data sample \mathbf{x}_0 , and $\hat{\mathbf{x}}_0 \stackrel{\text{def.}}{=} g_\tau(\hat{\mathbf{z}}_0)$. This implies

$$\mathcal{W}_2^2(p_0, \hat{p}_0) \leq \|\Delta_{f_\phi, g_\tau}(\mathbf{x})\|^2 + L_{g_\tau}^2 \mathcal{W}_2^2(q_0^\phi, \hat{q}_0^\phi) \quad (27)$$

by Lipschitz continuity of g_τ . Finally, using Proposition A.2 to bound $\mathcal{W}_2^2(q_0^\phi, \hat{q}_0^\phi)$, we have

$$\mathcal{W}_2^2(p_0, \hat{p}_0) \leq \|\Delta_{f_\phi, g_\tau}(\mathbf{x})\|^2 + L_{g_\tau}^2 e^{1+2\hat{L}} \int_0^1 \int_{\mathbb{R}^{d/h}} \|v_t(\mathbf{z}_t, t) - \hat{v}_t(\mathbf{z}_t, t)\|^2 q_t^\phi dz dt. \quad (28)$$

with a small remark that although in [1], the author use p_0 to denote the base (noise), and p_1 target (data) distributions, this does not prevent the direct usage of their result in our proof. \square

B Analysis of fixed-steps ODE solvers

We perform fixed-step ODE solvers with various numbers of steps, then measure FID and sampling time on each setting. As shown in Figure 5, the Heun solver consistently obtains better FID scores across different settings, but its sampling time is slightly higher than that of the Euler algorithm.

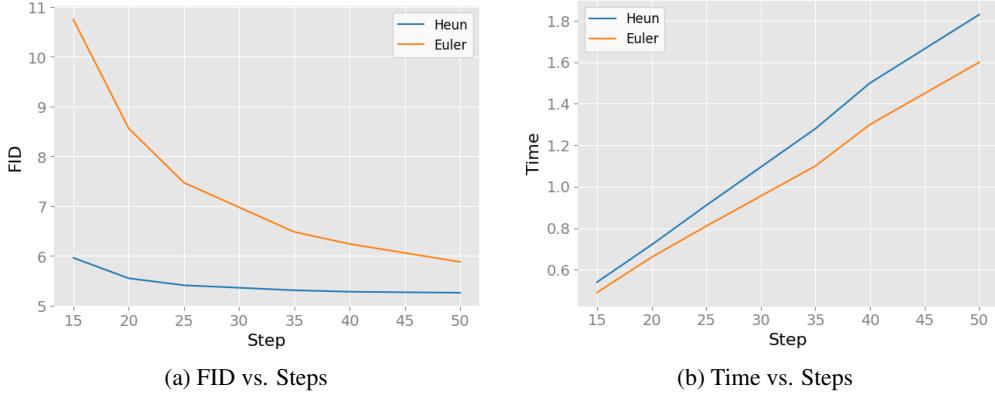


Figure 5: Fixed-step ODE solvers on CelebA-HQ 256.

C Algorithms

For unconditional image generation, Training and sampling procedure are included in Algorithm 1 and Algorithm 2.

Compared to the unconditional model, the training procedure for class-conditional image generation is basically similar but includes an extra class condition in the estimation function $v_\theta(\mathbf{z}, \mathbf{c}, t)$ in Algorithm 3. The sampling process with the proposed classifier-free velocity field is also portrayed in Algorithm 4.

Algorithm 1: Unconditional training

Input: data p_0 , VAE encoder \mathcal{E} , velocity estimator v_θ , learning rate η ;
repeat
 $\left| \begin{array}{l} \mathbf{x}_0 \sim p_0, \mathbf{z}_0 \leftarrow \mathcal{E}(\mathbf{x}_0); \\ \mathbf{z}_1 \leftarrow \mathcal{N}(0, \mathbf{I}), t \sim \mathcal{U}[0, 1]; \\ \mathbf{z}_t \leftarrow (1-t)\mathbf{z}_0 + t\mathbf{z}_1; \\ l \leftarrow \|\mathbf{z}_1 - \mathbf{z}_0 - v_\theta(\mathbf{z}_t, t)\|_2^2; \\ \theta \leftarrow \theta - \eta \nabla_\theta l; \end{array} \right.$
until convergence;

Algorithm 2: Unconditional Euler sampling

Input: initial noise \mathbf{z}_1 , VAE decoder \mathcal{D} , velocity estimator v_θ , steps N ;
 $t_n \leftarrow \frac{n}{N}$;
for $n = N - 1$ **to** 0 **do**
 $| \quad z_{t_n} \leftarrow z_{t_{n+1}} + (t_n - t_{n+1})v_\theta(\mathbf{z}_{t_{n+1}}, t_{n+1})$
end
 $\mathbf{x}_0 \leftarrow \mathcal{D}(\mathbf{z}_0);$
Output: \mathbf{x}_0 ;

Algorithm 3: Class-conditional training

Input: data p_0 , VAE encoder \mathcal{E} , velocity estimator v_θ , learning rate η , probability of unconditional training p_u ;
repeat
 $\left| \begin{array}{l} \mathbf{x}_0, \mathbf{c} \sim p_0, \mathbf{z}_0 \leftarrow \mathcal{E}(\mathbf{x}_0); \\ \mathbf{c} \leftarrow \emptyset \text{ with probability } p_u; \\ \mathbf{z}_1 \leftarrow \mathcal{N}(0, \mathbf{I}), t \sim \mathcal{U}[0, 1]; \\ \mathbf{z}_t \leftarrow (1-t)\mathbf{z}_0 + t\mathbf{z}_1; \\ l \leftarrow \|\mathbf{z}_1 - \mathbf{z}_0 - v_\theta(\mathbf{z}_t, \mathbf{c}, t)\|_2^2; \\ \theta \leftarrow \theta - \eta \nabla_\theta l; \end{array} \right.$
until convergence;

Algorithm 4: Class-conditional Euler sampling

Input: initial noise \mathbf{z}_1 , VAE decoder \mathcal{D} , velocity estimator v_θ , steps N , class c , guidance γ ;
 $t_n \leftarrow \frac{n}{N}$;
for $n = N - 1$ **to** 0 **do**
 $| \quad v_c \leftarrow v_\theta(\mathbf{z}_{t_{n+1}}, c, t_{n+1});$
 $| \quad v_u \leftarrow v_\theta(\mathbf{z}_{t_{n+1}}, \emptyset, t_{n+1});$
 $| \quad \tilde{v}_c \leftarrow v_u + \gamma(v_c - v_u)$
 $| \quad z_{t_n} \leftarrow z_{t_{n+1}} + (t_n - t_{n+1})\tilde{v}_c$
end
 $\mathbf{x}_0 \leftarrow \mathcal{D}(\mathbf{z}_0);$
Output: \mathbf{x}_0 ;

Model	FID \downarrow	Recall \uparrow	Params
Ours (ADM)	16.71	0.56	387M
+ cfg=1.25	8.58	0.50	387M
Ours (DiT B/2)	20.38	0.56	130M
+ cfg=1.25	8.77	0.48	130M
+ cfg=1.5	4.46	0.42	130M
LDM-8 [51]	15.51	0.63	395M
LDM-8-G	7.76	0.35	506M
LDM-4	10.56	0.62	400M
LDM-4-G (cfg=1.5)	3.60	0.48	400M
DiT-B/2 [46]	43.47	-	130M
DiT-XL/2	9.62	0.67	675M
DiT-XL/2-G (cfg=1.25)	3.22	0.62	675M
ADM [9]	10.94	0.63	554M
ADM-U	7.49	0.63	554M
ADM-G	4.59	0.52	608M
ADM-G, ADM-U	3.94	0.53	-
CDM [19]	4.88	-	-
ImageBART [13]	7.44	-	3.5B
VQGAN+T [14]	5.88	-	1.3B
MaskGIT [5]	6.18	0.51	227M
RQ-Transformer [33]	3.80	-	3.8B
BigGan-deep [3]	6.95	0.28	160M
StyleGAN-XL [54]	2.30	0.53	166M

Table 5: Class-conditional image generation on ImageNet 256×256

D Additional experiments

D.1 Class-conditional image generation

More comparisons of existing methods on ImageNet 256×256 are given in Table 5. It is important to emphasize that our approach, utilizing the DiT variant and the proposed classifier-free velocity guidance, has consistently demonstrated superior performance compared to both the original LDM and DiT-B/2 alternatives, despite having smaller model sizes. Additionally, our method exhibits a notable level of competitiveness when compared to established methods in the field.

D.2 More qualitative results

For better illustration, additional visual examples are given for CelebA-HQ 256 in Figure 6, FFHQ in Figure 7; LSUN Church in Figure 8; LSUN Bedroom in Figure 9; CelebA-HQ 512 in Figure 10, and ImageNet in Figures 11 to 15, Figures 16 to 18, and Figures 19 and 20 with guidance scale of 4.0, 2.0, and 1.5, respectively.

E Implementation details

Network configuration. We adopt two types of network architectures, namely UNet (ADM) and transformer network (DiT), for most experiments. Table 7 shows detailed configurations of the ADM network on different datasets. For DiT, we adopt the DiT-L/2 variant for the unconditional generation with shape 256×256 , and DiT-B/2 for class-conditional generation on ImageNet. Additionally, the size of DiT network is described at Table 6.

Table 6: Size of DiT Network.

Network	Image size	Params (M)	FLOPs (G)
DiT-L/2	256	457	80.74
DiT-B/2	256	130	23.01

Table 7: ADM configurations.

	CelebA, FFHQ & Bed(256)	Church(256)	CelebA(512)	Imnet(256)
# of ResNet blocks per scale	2	2	2	2
Base channels	256	256	256	256
Channel multiplier per scale	1,2,3,4	1,2,3,4	1,2,2,2,4	1,2,3,4
Attention resolutions	16, 8, 4	16, 8	16, 8	16, 8, 4
Channel multiplier for embeddings	4	4	4	4
Label dimensions	0	0	0	1000
Params (M)	368	356	352	369
FLOPs (G)	28.9	28.9	94.2	28.9

Training hyper-params. In Table 8 and Table 9, we provide training hyperparameters for unconditional image generation on ADM and DiT networks, respectively. In both tables, each setting also shows the estimated training days.

Table 8: Hyper-parameters of ADM network.

	CelebA 256	FFHQ	CelebA 512	Church & Bed	IMNET
lr	5e-5	2e-5	5e-5	5e-5	1e-4
Adam optimizer (β_1 & β_2)	0.9, 0.999	0.9, 0.999	0.9, 0.999	0.9, 0.999	0.9, 0.999
Batch size	112	128	64	128	96
# of epochs	600	500	500	500	1200
# of GPUs	2	1	2	1	8
# of training days	1.3	4.4	5.9	6.9 & 7.3	21.6

Table 9: Hyper-parameters of DiT network.

	CelebA 256	FFHQ	Church & Bed	IMNET
Model	DiT-L/2	DiT-L/2	DiT-L/2	DiT-B/2
lr	2e-4	2e-4	1e-4	1e-4
AdamW optimizer (β_1 & β_2)	0.9, 0.999	0.9, 0.999	0.9, 0.999	0.9, 0.999
Batch size	32	32	96 & 32	160
# of epochs	500	500	500	900
# of GPUs	1	1	2 & 1	8
# of training days	5.8	5.5	6.1 & 12.5	12.0



Figure 6: Non-curated generated samples of CelebA-HQ 256.

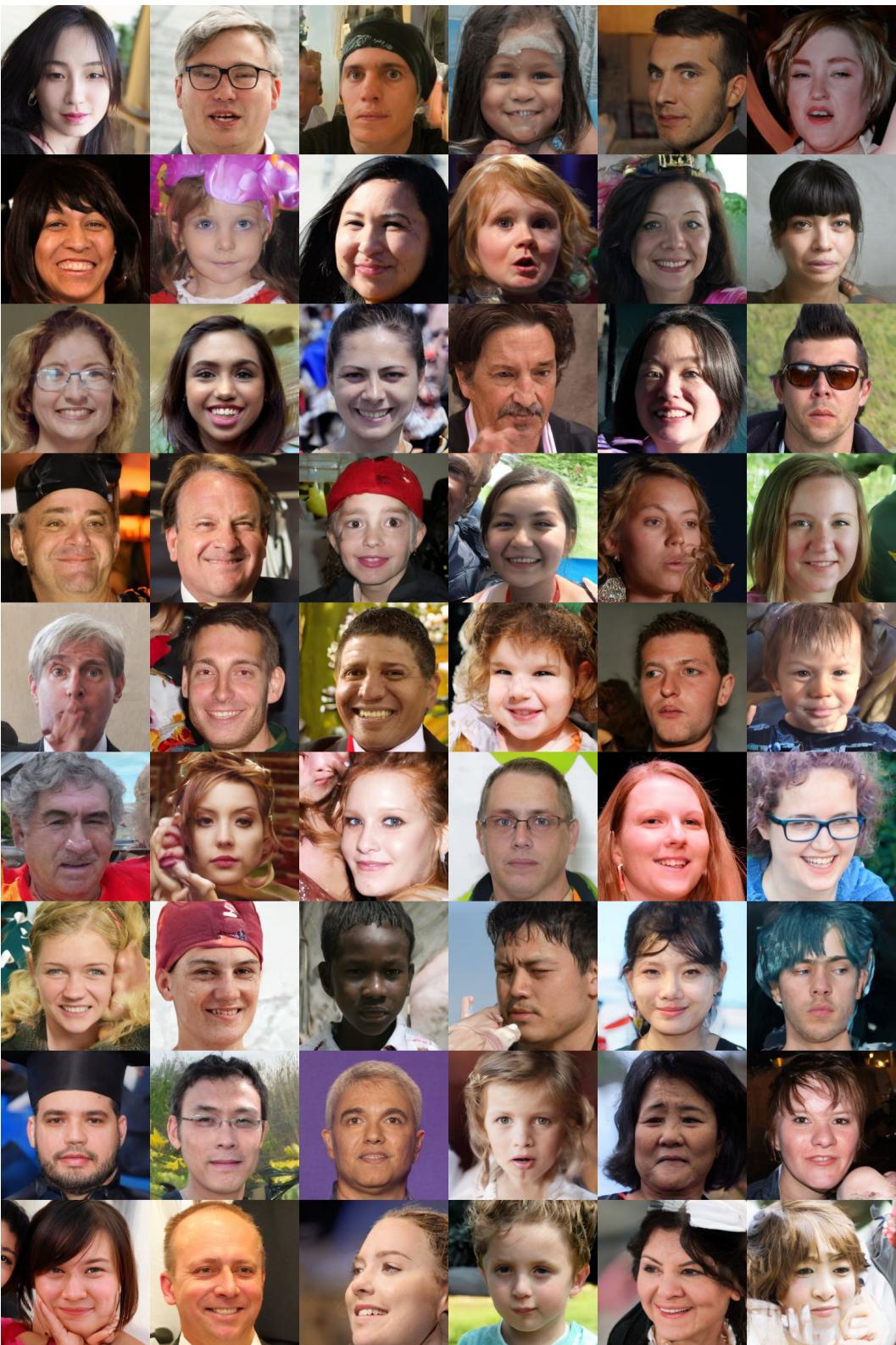


Figure 7: Non-curated generated samples of FFHQ 256.



Figure 8: Non-curated generated samples of LSUN-Church.

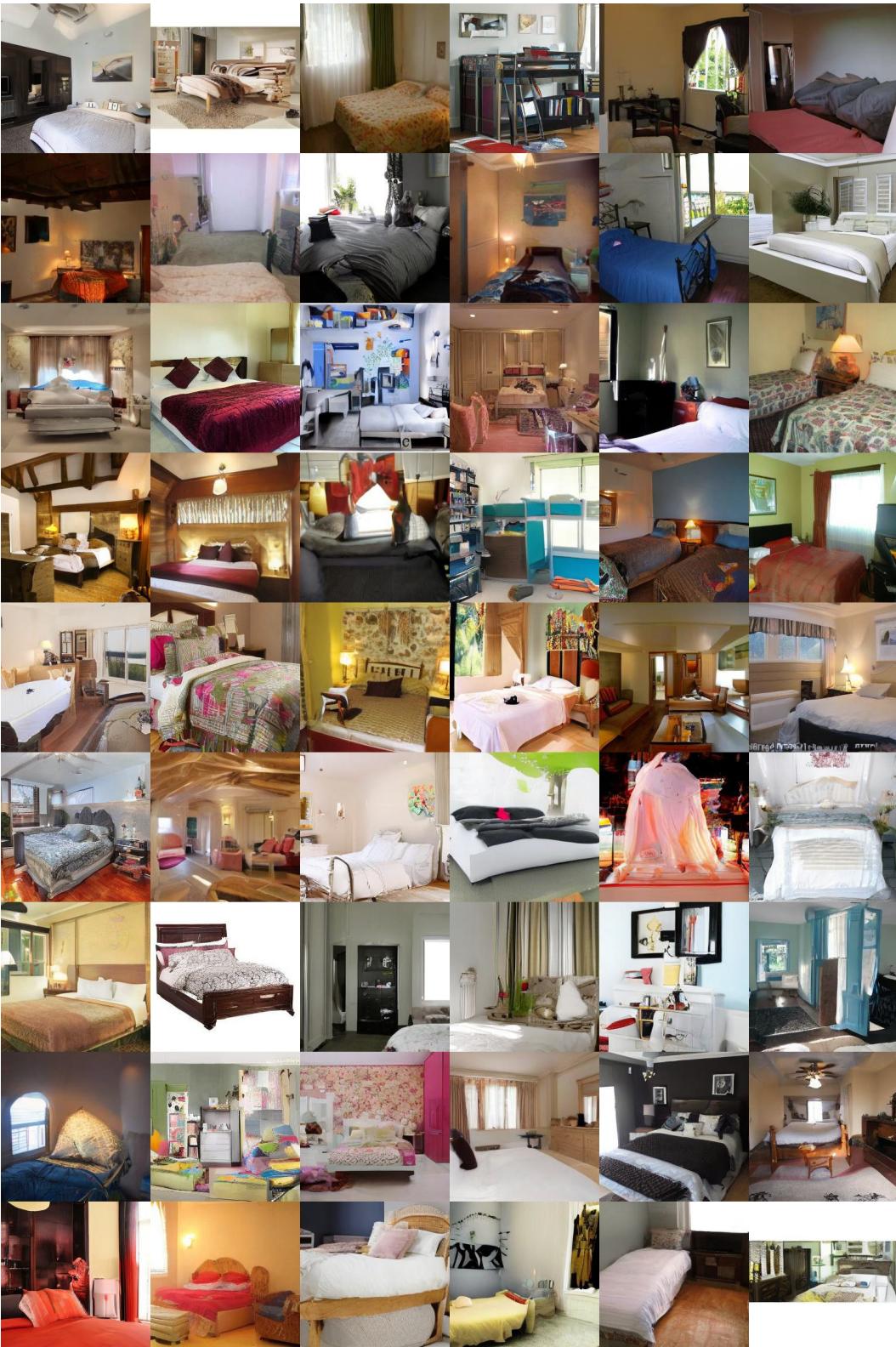


Figure 9: Non-curated generated samples of LSUN-Bedroom.



Figure 10: Non-curated generated samples of CelebA-HQ 512.



(a) Class: 88 - 'Macaw'

(b) Class: 417 - 'Balloon'

Figure 11: Non-curated samples of our DiT-B/2 variant on ImageNet (Guidance scale: 4.0).



(a) Class: 327 - ‘Starfish’



(b) Class: 947 - ‘Mushroom’

Figure 12: Non-curated samples of our DiT-B/2 variant on ImageNet (Guidance scale: 4.0).



(a) Class: 396 - ‘Lionfish’

(b) Class: 33 - ‘Loggerhead Turtle’

Figure 13: Non-curated samples of our DiT-B/2 variant on ImageNet (cfg = 4.0).



(a) Class: 449 - ‘Boathouse’

(b) Class: 972 - ‘Cliff’

Figure 14: Non-curated samples of our DiT-B/2 variant on ImageNet (cfg = 4.0).



(a) Class: 207 - ‘Golden retriever’

(b) Class: 279 - ‘Arctic fox’

Figure 15: Non-curated samples of our DiT-B/2 variant on ImageNet (cfg = 4.0).



(a) Class: 89 - ‘Sulphur-crested cockatoo’

(b) Class: 980 - ‘Volcano’

Figure 16: Non-curated samples of our DiT-B/2 variant on ImageNet (cfg = 2.0).



(a) Class: 973 - ‘Coral reef’

(b) Class: 360 - ‘Otter’

Figure 17: Non-curated samples of our DiT-B/2 variant on ImageNet (cfg = 2.0).



(a) Class: 937 - ‘Broccoli’

(b) Class: 963 - ‘Pizza’

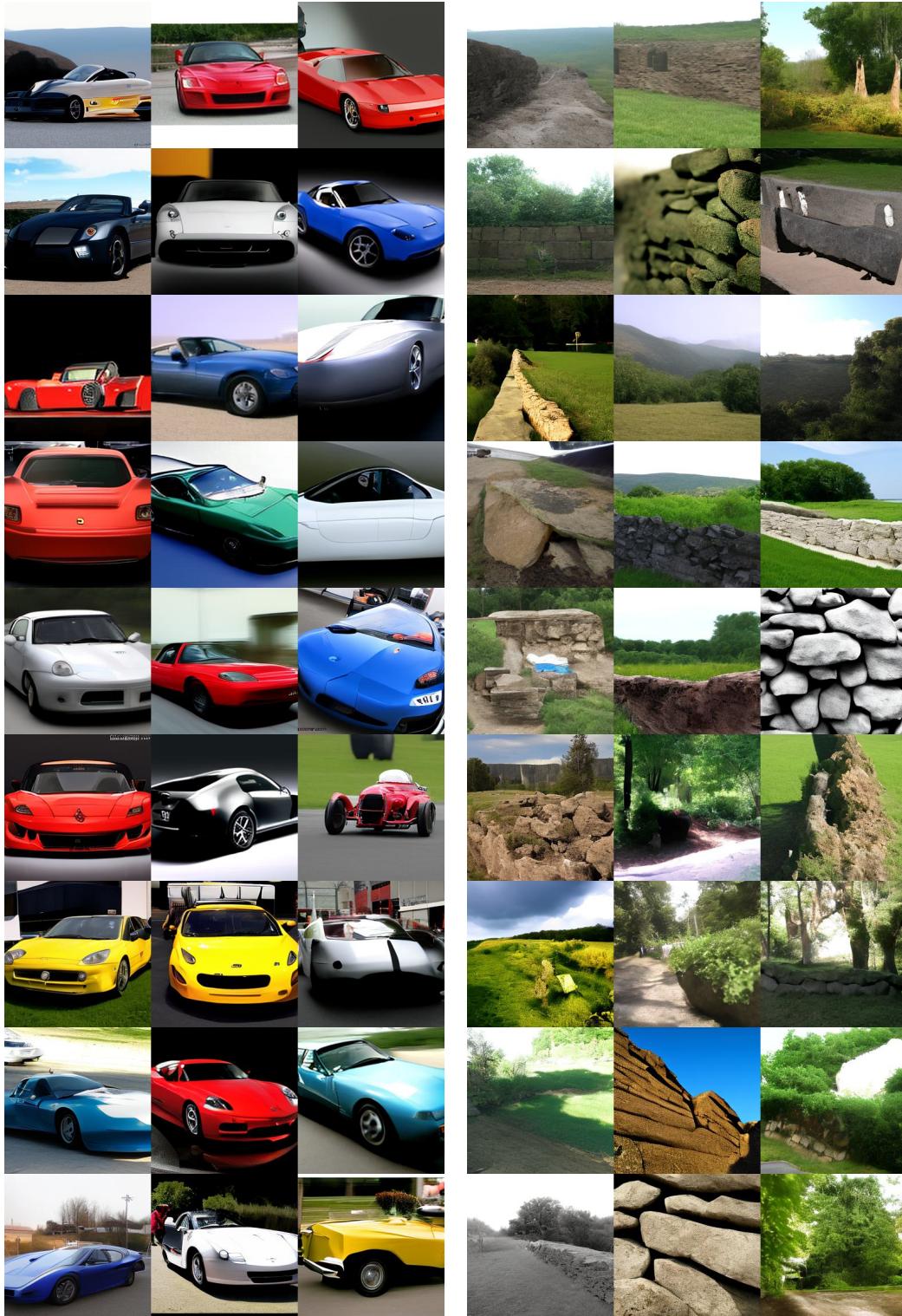
Figure 18: Non-curated samples of our DiT-B/2 variant on ImageNet (cfg = 2.0).



(a) Class: 975 - 'Lakeside'

(b) Class: 984 - 'Rapeseed'

Figure 19: Non-curated samples of our DiT-B/2 variant on ImageNet (cfg = 1.5).



(a) Class: 817 - ‘Sports car’

(b) Class: 825 - ‘Stone wall’

Figure 20: Non-curated samples of our ADM variant on ImageNet (cfg = 1.5).