COMPOSITIONAL GENERATIVE INFERENCE USING DIFFUSION-BASED OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Compositional generative tasks, despite being important and having potential applications, have not been thoroughly addressed due to the unclear formulation and the challenges associated with selecting composition strategies. In this paper, we propose a probabilistic graphical approach to tackle the problem of compositional generative tasks and alleviate these challenges. Our approach formulates the problem as a Bayesian inference problem using a representative bipartite Bayesian network. In this network, one set of random variables represents the generation targets, while the other set represents observable variables with explicit or implicit distribution information. To solve this problem, we employ variational inference on the marginal distribution of observable variables. We approximate this distribution using diffusion models. We view the diffusion models as approximate Markov Chain Monte Carlo (MCMC) samplers for the marginals. Based on this perspective, we introduce a novel MCMC-based inference algorithm that incorporates per-step optimization using aggregated objectives from the diffusion models. We demonstrate the generality of our method and conduct experiments to validate its applicability to various compositional generation tasks.

1 INTRODUCTION

We understand nature in a highly compositional manner. One good example is our visual memorization Luck & Hollingworth (2008); we decompose the visual scene into small components (e.g., to objects and background) and remember those entities with relation, where we later composite them back when bringing the memory out. The compositional nature of human behavior, while it does affect all aspects of parts of our lives, is especially significant in generative tasks that require creativity. Most of the creative generations involve compositing immanent concepts and ideas.

The need for compositional understanding applies the same to the field of generative AI. Generative AI models have shown remarkable strides in recent advancements for fields requiring creativity, such as text-to-image synthesis Rombach et al. (2022); Kumari et al. (2023); Kawar et al. (2023); Melas-Kyriazi et al. (2023); Nichol et al. (2021), text-to-video synthesis Ho et al. (2022); Blattmann et al. (2023), human motion generation Tevet et al. (2022), and so on. Such advancements are driven by diffusion models: a likelihood-based method that generates plausible samples by iterative denoising any random samples, which have shown astonishing improvements in terms of quality, plausibility, and coherency with given input conditions.

However, there has not been adequate discussion under the perspective of compositionality for generative models, and currently, the models cannot be actively applied to compositional generative tasks as they do not guarantee synergy when composited naively. The lack of compositional ability ultimately results in limitations for alternations in generative tasks, including even the slightest differences. This is because when the compositional ability is not provided, generative models (especially diffusion models) need to be trained on an extensive dataset of conditions and sample pairs for every specific generation task. As there exists a large obstacle to training due to the exhaustive data-collection procedure and costly training procedure, only a few models can be trained under large capital for the limited scope of tasks. This limitation poses a significant obstacle and hinders the exploration of various potential applications.

There do exist prior works that address the compositional generation problem (Refer to Sec. 2). However, these works only suggest a methodology or justification for limited types of data or ways of composition. This is due to challenges within the problem itself: (1) the formulation for the problem of compositing multiple generative models is ambiguous, unclear, and not mathematically formulated; (2) for selecting optimal composition strategies, what objectives should aim for are unclear; (3) efforts must be made to develop generalized formulations that are applicable to a wide range of tasks. In this paper, we present a probabilistic graphical approach for formulating the problem as a Bayesian inference problem. We model the situation as a bipartite Bayesian network, consisting of "control variables" which we can alter directly but do not have distribution information, and "observable variables" with explicit/implicit distribution information provided but mathematically dependent on control variables. Under this formulation, we aim to sample the mode of the joint probability, which is an inference problem.

Nonetheless, there exist several obstacles when devising a composition strategy, other than the challenges of defining the composition problem itself: (1) the inference problem can be notoriously difficult to solve when the distribution information is not provided for generation targets (i.e., control variables) or is only provided for partial observations of the targets (i.e., observable variables) as there are countless ways of compositing information; (2) it is challenging to use the raw explicit/implicit distribution information directly. To mitigate these challenges, we present an optimization-based sampling method that resembles Markov Chain Monte Carlo (MCMC) method under the provided formulation. We suggest a method to utilize diffusion models as approximate MCMC samplers, which are used to process the information from the raw distribution of observable variables into feasible score information. We then propose a method to aggregate such information from multiple diffusion models, to set an optimization objective for control variables. We propose that by adopting these methods, we can iteratively sample control variables via per-step optimization. This allows us to sample control variables from the mode effectively.

Our method is general and can be applied to arbitrary compositional generative tasks. We validate its applicability by experiments and empirically show that our method returns trustworthy generation results.

2 Related Work

Diffusion Model Diffusion model Ho et al. (2020); Song et al. (2020a) is a class of methods generating sample data, generally images, by denoising latent starting from random initial noise. To guide the diffusion model for image synthesis and editing method work, SDEdit Meng et al. (2021) introduces a novel method by interpreting the diffusion process as a stochastic differential equation problem Song et al. (2020b), which is the generalized representation of an ordinary differential equation. Latent Diffusion Model(LDM) Rombach et al. (2022) improved the quality and efficiency by performing diffusion in reduced latent space and larger dataset Schuhmann et al. (2021). Classifier-Free Guidance(CFG) Ho & Salimans (2022) is an approach that combines score estimation Hyvärinen & Dayan (2005) of condition and unconditional diffusion models to obtain a similar quality of classifier guidance Dhariwal & Nichol (2021) result.

Text-guided Diffusion Models Most of the recent diffusion-based work is strongly correlated to text-guided image synthesis Rombach et al. (2022); Kumari et al. (2023); Kawar et al. (2023); Melas-Kyriazi et al. (2023); Nichol et al. (2021); Blattmann et al. (2022) with the help of text embedding networks Radford et al. (2021b;a) for latent spaces. For text and image guiding for Diffusion models, Instruct-Pix2Pix-based recent works Brooks et al. (2023); Kamata et al. (2023); Haque et al. (2023) suggests the fine-tuned Stable Diffusion Rombach et al. (2022) in a supervised manner with custom datasets automatically leveraging Prompt-to-Prompt and GPT-3 Hertz et al. (2022); Brown et al. (2020). Also, ControlNet Zhang & Agrawala (2023) suggests guidance using various methods, such as Canny Edge Canny (1986), OpenPose Cao et al. (2019), user scribe, or text prompt.

Compositional GAN GIRAFFE Niemeyer & Geiger (2021) is a compositional GAN model generating scenes together with objects, i.e., a street with cars. To enable us to arrange objects freely, it models a scene in 3D and asks us to locate 3D objects in the scene, where each of them is represented implicitly. GIRAFFE generates composited images by rendering all entities together using NeRF pipeline Mildenhall et al. (2021), and it's trained with GAN loss Goodfellow et al. (2014); Mescheder et al. (2018). Though GIRAFFE can generate controllable high-quality 3D scenes, it has limitations on image resolution and scene scale.

Large Image Synthesis with Diffusion Though most diffusion models have translational equivariance, which allows us to generate an arbitrary-sized image, they are failed to produce plausible large images without additional fine-tuning Rombach et al. (2022); Avrahami et al. (2022).

Compositional Generation Using Composable Diffusion MultiDiffusion and DiffCollage Bar-Tal et al. (2023); Zhang et al. (2023) are the works to address the composition strategy for multiple diffusion. Two works have the purpose of compositing multiple text-to-image diffusion models to generate holistic images for images of local regions.

3 Preliminaries

3.1 Diffusion Models

In this paper, we follow the formulation of Denoising Diffusion Implicit Models (DDIM) Song et al. (2020a) to derive the training process and inference process. In a nutshell, diffusion models sample start from the initial random sample $\mathbf{y}^T$ from the prior distribution $\mathcal{N}(0, \mathbf{I})$ and generate samples iteratively ($\mathbf{y}^{T-1}, ..., \mathbf{y}^0$) by removing the noise from the sample, which can be viewed as MCMC sampling process for $\mathbf{y}^0$'s.

Training To train diffusion models, we first sample a data $\mathbf{y}^0$ from the dataset and sample the noisy latents $\mathbf{y}^t$ $(t = 1, ..., T)$ from $\mathbf{y}^0$ via non-Markovian forward kernel:

$$q(\mathbf{y}^{1:T} \mid \mathbf{y}^0) = q(\mathbf{y}^T \mid \mathbf{y}^0) \prod_{t=2}^{T} q(\mathbf{y}^{t-1} \mid \mathbf{y}^t, \mathbf{y}^0) \tag{1}$$

where the forward kernel for each step is provided as:

$$q(\mathbf{y}^{t-1} \mid \mathbf{y}^t, \mathbf{y}^0) = \mathcal{N}(\sqrt{\bar{\alpha}_t}\mathbf{y}^0 + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} \cdot \frac{\mathbf{y}^t - \sqrt{\bar{\alpha}_t}\mathbf{y}^0}{\sqrt{1 - \bar{\alpha}_t}}, \sigma_t^2 \mathbf{I}) \tag{2}$$

$$q(\mathbf{y}^T \mid \mathbf{y}^0) = \mathcal{N}(\sqrt{\bar{\alpha}_T}\mathbf{y}^0, (1 - \bar{\alpha}_T)\mathbf{I}) \tag{3}$$

Note that $\bar{\alpha}_t, \sigma_t$ are hyperparameter constants where $\bar{\alpha}_T \approx 0$, and σ_t is the scale for Langevin noise term Song et al. (2020a). We then approximate the distribution of latents $q(\mathbf{y}^{0:T} \mid \mathbf{y}^0)$ with the reverse process of diffusion model: $p_\theta(\mathbf{y}^{0:T})$ where the design choice of approximate distribution p_θ is set as Markovian:

$$p_\theta(\mathbf{y}^{t-1} \mid \mathbf{y}^t) = \begin{cases} \mathcal{N}(f_\theta^{(1)}(\mathbf{y}^1), \sigma_1^2 \mathbf{I}) & \text{if } t = 1 \\ q(\mathbf{y}^{t-1} \mid \mathbf{y}^t, f_\theta^{(t)}(\mathbf{y}^t)) & \text{otherwise} \end{cases} \tag{4}$$

$$f_\theta^{(t)}(\mathbf{y}^t) = (\mathbf{y}^t - \sqrt{1 - \bar{\alpha}_t} \cdot \epsilon_\theta^{(t)}(\mathbf{y}^t)) / \sqrt{\bar{\alpha}_t} \tag{5}$$

here $\epsilon_\theta^{(t)}$ is known as "score predictor" Song & Ermon (2019); Song et al. (2020a), where the "score" is the gradient of log-probability distribution. We can obtain optimal θ by minimizing the following variational inference objective:

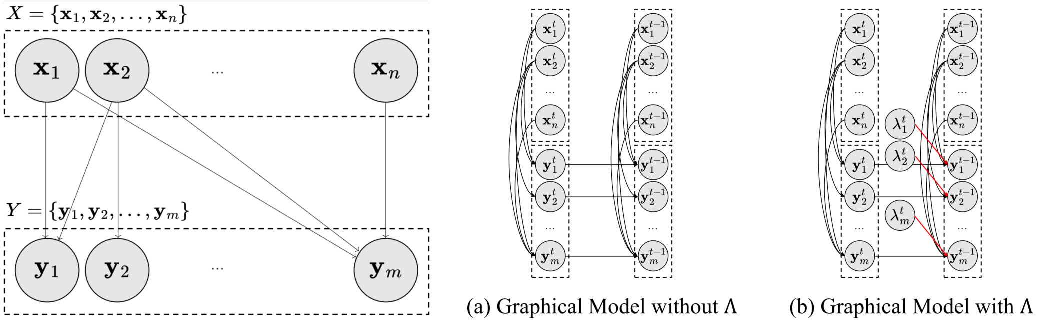
$$J(\theta) = \mathbb{E}_{\mathbf{y}^{0:T} \sim q(\mathbf{y}^{0:T})}[\log q(\mathbf{y}^{0:T} \mid \mathbf{y}^0) - \log p_\theta(\mathbf{y}^{0:T})] \tag{6}$$

According to Song et al. Song et al. (2020a), minimizing this objective is equivalent to minimizing a surrogate objective:

$$J_{\text{surr}}(\theta) = \sum_{t=1}^{T} \mathbb{E}_{\mathbf{y}^0 \sim q(\mathbf{y}^0), \epsilon_t \sim \mathcal{N}(0, \mathbf{I})}[\|\epsilon_\theta^{(t)}(\sqrt{\bar{\alpha}_t}\mathbf{y}^0 + \sqrt{1 - \bar{\alpha}_t}\epsilon_t) - \epsilon_t\|^2] \tag{7}$$

This process can be viewed as variational inference for the distribution of $\mathbf{y}^{0:T}$, and the approximated reverse process p_θ can be viewed as an MCMC sampler. Also, the surrogate objective in Eq. 7 is equivalent to the surrogate objective for Denoising Diffusion Probabilistic Models (DDPM) Ho et al. (2020), which enables us to use the pretrained DDPM model for DDIM inference.

Inference After training, we can sample plausible $\mathbf{y}^0$ by sampling initial latent $\mathbf{y}^T \sim \mathcal{N}(0, \mathbf{I})$ and sequentially sampling the previous timestep latent given the current timestep latent: $p_\theta(\mathbf{y}^{t-1} \mid \mathbf{y}^t)$. Hence, the inference procedure for the diffusion model can be viewed as an approximate MCMC sampling for distribution $q(\mathbf{y}^0)$.



(a) Graphical Model without Λ (b) Graphical Model with Λ

Figure 1: (left) Bipartite Bayesian network introduced in Formulation. (right) Graphical model for MCMC Sampling at timestep t.

3.2 Latent Diffusion Models

In Sec. 5.2, we use publicly available Latent Diffusion Models Rombach et al. (2022) for approximating the distribution of observable variables via variational inference. Latent Diffusion Model (LDM) is a diffusion model that operates on the latent space of an image, where the images are encoded into latents and decoded back to the image via pretrained VAE Rombach et al. (2022); Kingma & Welling (2013); Van Den Oord et al. (2017). Specifically, when given pretrained VAE encoder $\mathcal{E}$ and decoder $\mathcal{D}$, the relation between the latent and image satisfies:

$$z = \mathcal{E}(x),\ x \approx \mathcal{D}(z) \tag{8}$$

LDMs generate the latents z from initial random noise latent, which can be transformed to an image using the decoder as in Eq. 8. On account of this, we represent the image and image latent with the same notation (a y-notation, i.e., $\mathbf{y}^t$ or $\mathbf{y}_i^t$) for brevity. It is noteworthy that LDMs are trained following the DDPM training procedure Ho et al. (2020). Since the training procedure of DDPM and DDIM are equivalent, as we have mentioned earlier in Sec. 3.1, it is fine to use DDIM's inference procedure for approximate MCMC sampling, as we have done in our research.

4 Formulation

In this section, we will provide a general mathematical formulation for the compositional generation problem. Consider the following Bayesian inference problem. Given the bipartite Bayesian graphical model $G_{\mathcal{B}}(V, E)$, where vertices can be decomposed into two mutually exclusive sets $V = \{X, Y\}$. We denote $X = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ as a set of "control variables" and $Y = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_m\}$ as a set of "observable variables". Refer to Fig. 1-(left) for the visualization of the graphical model. Control variables are the target variables we aim to generate or possibly alter to have a higher probability. Observable variables are the variables we can "observe", which means that we can obtain the distribution information $p(Y)$:

$$p(Y) = \prod_{i=1}^{m} p(\mathbf{y}_i) \tag{9}$$

which suffices for each observable variable, obtaining distribution information $p(\mathbf{y}_i)$ either explicitly or implicitly. We may consider variables "containing explicit distribution information" if the exact marginal probability value $p(\mathbf{y}_i)$ can be evaluated and variables "containing implicit distribution information" (1) if the information is obtained from the generative models for the marginal distribution $p(\mathbf{y}_i)$; (2) if we can sample from the distribution; or so on.

This formulation represents the composition problem well, as we need to composite the information from observable variables in order to achieve meaningful updates for the compositions (which are represented as control variables). Under the intuition that the compositions we try to generate will likely have a mathematical relationship with each separate component in the direction starting from composition to the component (i.e., the component is dependent on composition), we formulate the dependency between control variables and observable variables as a conditional distribution:

$$p(Y \mid X) = \prod_{i=1}^{m} p(\mathbf{y}_i \mid \text{PA}(\mathbf{y}_i)) \tag{10}$$

where PA$(\cdot)$ denotes the parents of a given random variable. Note that such composition is assured since the graphical model is bipartite and the parents of $\mathbf{y}_i$ is a subset of X. Given the distribution information for observable variables Y and stochastic dependency between X and Y, the problem's goal is to find all realizations of control variables X that maximize the marginal probability of X:

$$X^* = \arg\max_X p(X) \tag{11}$$

This is a general formulation encompassing a wide range of compositional generation tasks, where the definition of tasks varies by the type of information provided for the marginal distribution of observable variables (i.e., $p(Y)$) or type of stochastic dependency between control variables and observable variables (i.e., $p(Y \mid X)$).

In this work, we focus on the case where we can sample from the marginal distribution of observable variable $p(\mathbf{y}_i)$ and no other distribution information is provided. We formulate the stochastic dependency $p(Y \mid X)$, as a deterministic observation with "observation noise" Ψ_i (for $i = 1, .., m$):

$$p(Y \mid X) = \prod_{i=1}^{m} \mathcal{N}(\mathbf{y}_i \mid f_i(\text{PA}(\mathbf{y}_i)), \Psi_i) \tag{12}$$

where observation functions f_i and observation noise Ψ_i, vary by the characteristics specific to the task. For brevity, we will denote $f_i(\text{PA}(\mathbf{y}_i))$ as $f_i(X)$, considering the function to be constant for inputs other than PA$(\mathbf{y}_i)$. In this work, we formulate Ψ_i to be scaled identity, i.e., $\Psi_i = \psi_i^2\mathbf{I}$.

5 Method

5.1 Overview

We now discuss our method under the provided formulation. Our method mitigates the aforementioned challenges by introducing various strategies. The method starts by training diffusion models on the marginal distribution of observable variables $p(\mathbf{y}_i)$ for $\forall i = 1, ..., m$, where the dataset is sampled from $p(\mathbf{y}_i)$'s. This procedure can be viewed as variational inference, and the trained diffusion model can be viewed as an approximate MCMC sampler, in which the resulting distribution information (for our method, score function value) is easier to use than just direct sampling. We then aggregate the score function values for $\mathbf{y}_i$'s to create an optimization objective with respect to control variables X, which resembles the objectives in Expectation-Maximization (EM) algorithm Moon (1996). We optimize X according to the objective, which can be viewed as sampling from an adaptive proposal. The method repeats the aforementioned process, finally giving optimal X.

5.2 Diffusion as Approximate MCMC

The method trains diffusion models, with different parameters θ_i for each observable variable $\mathbf{y}_i$ to approximate the marginal distribution $p(\mathbf{y}_i)$. The training procedure starts by sampling $\mathbf{y}_i^0$'s from each marginal distribution $p(\mathbf{y}_i)$. Then, following the provided forward kernel represented in Eq. 1, we sample the sequence of noisy latents $\mathbf{y}_i^{0:T}$ from each sample $\mathbf{y}_i^0$. Using the collected dataset, we minimize the surrogate objective $J_{\text{surr}}(\theta_i)$ from Eq. 7 for each i, resulting m different diffusion models:

$$\theta_i^* = \arg\min_{\theta_i} J_{\text{surr}}(\theta_i) \text{ for } \forall i = 1, ..., m \tag{13}$$

As mentioned in Sec. 3.1, this process can be viewed as variational inference. After training, the method utilizes trained diffusion models as approximate MCMC sampler, in which the adaptive proposal can be derived from Eq. 2∼5 as:

$$\mathbf{y}_i^{t-1} = \sqrt{\bar{\alpha}_{t-1}}\left(\frac{\mathbf{y}_i^t - \sqrt{1-\bar{\alpha}_t}\epsilon_{\theta_i}^{(t)}(\mathbf{y}_i^t)}{\sqrt{\bar{\alpha}_t}}\right) + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2} \cdot \epsilon_{\theta_i^*}^{(t)}(\mathbf{y}_i^t) + \sigma_t \mathbf{z}_t,\ \mathbf{z}_t \sim \mathcal{N}(0, \mathbf{I}) \tag{14}$$

In this work, we use the deterministic sampling process with zero Langevin noise, i.e., $\sigma_t = 0$ (Note that σ_t is a hyperparameter):

$$\mathbf{y}_i^{t-1} = \sqrt{\bar{\alpha}_{t-1}}\left(\frac{\mathbf{y}_i^t - \sqrt{1-\bar{\alpha}_t}\epsilon_{\theta_i}^{(t)}(\mathbf{y}_i^t)}{\sqrt{\bar{\alpha}_t}}\right) + \sqrt{1-\bar{\alpha}_{t-1}} \cdot \epsilon_{\theta_i^*}^{(t)}(\mathbf{y}_i^t) \tag{15}$$

5.3 Optimization-based Adaptive Proposal

Now that we have a diffusion-based approximate MCMC sampler for the marginal distribution $p(\mathbf{y}_i)$'s, we aim to sample X concurrent to the MCMC sampling process of $\mathbf{y}_i$'s. Specifically, when given control variables at timestep t ($X^t = \{\mathbf{x}_i^t\}_{i=1}^n$), we aim to sample the updated control variables X^{t-1} to eventually reach X^0, which is what we aim to generate. This process resembles sampling from an adaptive proposal, which is optimization-based. Such a sub-problem can be formulated again as an inference problem for the graphical model consisting of $X^t, Y^t, X^{t-1}, Y^{t-1}$. Refer to Fig. 1-(right)-(a) for visualization of the graphical model at timestep t.

We sample $X^{t-1(*)}$ when X^t is given via optimization:

$$X^{t-1(*)} = \underset{X^{t-1}}{\arg\max}\, \mathcal{Q}(X^{t-1} \mid X^t) \tag{16}$$

where $\mathcal{Q}$ is an optimization objective which resembles the EM algorithm Moon (1996):

$$\mathcal{Q}(X^{t-1} \mid X^t) := \mathbb{E}_{Y^t, Y^{t-1}}[\log p(X^{t-1}, Y^{t-1}, Y^t \mid X^t)] \tag{17}$$

From the graphical model provided in Fig. 1-(right)-(a), we can derive that pairs $(\mathbf{y}_i^t, \mathbf{y}_i^{t-1})$'s for all i's are mutually independent when given X. Hence, we can decompose the log-probability term as below:

$$\log p(X^{t-1}, Y^{t-1}, Y^t \mid X^t) = \sum_{i=1}^{m} \log p(X^{t-1}, \mathbf{y}_i^{t-1}, \mathbf{y}_i^t \mid X^t) \tag{18}$$

We can further factorize the log-probability term as:

$$\log p(X^{t-1}, Y^{t-1}, Y^t \mid X^t) = \sum_{i=1}^{m} [\log p(\mathbf{y}_i^{t-1}, X^{t-1} \mid \mathbf{y}_i^t) + \log p(\mathbf{y}_i^t \mid X^t)] \tag{19}$$

Using Eq. 19, we can also decompose the optimization objective as below:

$$\mathcal{Q}(X^{t-1} \mid X^t) = \sum_{i=1}^{m} \mathbb{E}_{\mathbf{y}_i^t, \mathbf{y}_i^{t-1}}[\log p(\mathbf{y}_i^{t-1}, X^{t-1} \mid \mathbf{y}_i^t) + \log p(\mathbf{y}_i^t \mid X^t)] \tag{20}$$

Inspired by the Hard EM algorithm Ruggieri et al. (2020), we approximate the expectation term w.r.t. $\mathbf{y}_i^t$ into a maximization term to approximate the optimization objective:

$$\mathcal{Q}(X^{t-1} \mid X^t) \approx \sum_{i=1}^{m} \mathbb{E}_{\mathbf{y}_i^{t-1}}[\log p(\mathbf{y}_i^{t-1}, X^{t-1} \mid \tilde{\mathbf{y}}_i^t) + \log p(\tilde{\mathbf{y}}_i^t \mid X^t)] \tag{21}$$

where $\tilde{\mathbf{y}}_i^t = \underset{\mathbf{y}_i^t}{\arg\max}\, p(\mathbf{y}_i^t \mid X^t)$. Recall from Eq. 12 that $\tilde{\mathbf{y}}_i^{t-1} \sim \mathcal{N}(f_i(X^{t-1}), \psi_i^2 \mathbf{I})$. From this, we can easily derive that $\tilde{\mathbf{y}}_i^t = f_i(X^t)$. We now argue that:

$$\log p(\mathbf{y}_i^{t-1}, X^{t-1} \mid \tilde{\mathbf{y}}_i^t) = \begin{cases} \log p(X^{t-1} \mid \tilde{\mathbf{y}}_i^{t-1}, \tilde{\mathbf{y}}_i^t) & \text{if } \mathbf{y}_i^{t-1} = \tilde{\mathbf{y}}_i^{t-1} \\ 0 & \text{otherwise} \end{cases} \tag{22}$$

where $\tilde{\mathbf{y}}_i^{t-1}$ is obtained via deterministic sampling described in Eq. 15 using $\tilde{\mathbf{y}}_i^t$, since for any $\mathbf{y}_i^{t-1} \neq \tilde{\mathbf{y}}_i^{t-1}$:

$$\int p(\mathbf{y}_i^{t-1}, X^{t-1} \mid \tilde{\mathbf{y}}_i^t) dX^{t-1} = p(\mathbf{y}_i^{t-1} \mid \tilde{\mathbf{y}}_i^t) = 0 \tag{23}$$

hence for $\forall X^{t-1}$:

$$p(\mathbf{y}_i^{t-1}, X^{t-1} \mid \tilde{\mathbf{y}}_i^t) = 0 \tag{24}$$

and for the case of $\mathbf{y}_i^{t-1} = \tilde{\mathbf{y}}_i^{t-1}$:

$$p(\tilde{\mathbf{y}}_i^{t-1}, X^{t-1} \mid \tilde{\mathbf{y}}_i^t) = p(X^{t-1} \mid \tilde{\mathbf{y}}_i^{t-1}, \tilde{\mathbf{y}}_i^t)p(\tilde{\mathbf{y}}_i^{t-1} \mid \tilde{\mathbf{y}}_i^t) = p(X^{t-1} \mid \tilde{\mathbf{y}}_i^{t-1}, \tilde{\mathbf{y}}_i^t) \tag{25}$$

since sampling for $\tilde{\mathbf{y}}_i^{t-1}$ is a deterministic procedure. Using Eq. 22, the optimization objectives can be reduced to:

$$\mathcal{Q}(X^{t-1} \mid X^t) \approx \sum_{i=1}^{m} [\log p(X^{t-1} \mid \tilde{\mathbf{y}}_i^{t-1}, \tilde{\mathbf{y}}_i^t) + \log p(\tilde{\mathbf{y}}_i^t \mid X^t)] \tag{26}$$

Applying Bayes' rule, we get:

$$\mathcal{Q}(X^{t-1} \mid X^t) \approx \sum_{i=1}^{m} [\log \frac{p(\tilde{\mathbf{y}}_i^{t-1} \mid X^{t-1})p(X^{t-1})}{p(\tilde{\mathbf{y}}_i^{t-1}, \tilde{\mathbf{y}}_i^t)} + \log p(\tilde{\mathbf{y}}_i^t \mid X^t)] \tag{27}$$

From $\tilde{\mathbf{y}}_i^{t-1} \sim \mathcal{N}(f_i(X^{t-1}), \psi_i^2\mathbf{I})$, if the magnitude of observation noise ψ_i is small, we can assume that marginal distribution $p(X^{t-1})$ is "locally constant" compared to $p(\tilde{\mathbf{y}}_i^{t-1} \mid X^{t-1})$ within the region around optimal X^{t-1}. The underlying idea behind this assumption is that the conditional distribution $p(\tilde{\mathbf{y}}_i^{t-1} \mid X^{t-1})$ will undergo significant changes when X^{t-1} deviates from the optimal point, particularly when the magnitude of Ψ_i is small. In contrast, the marginal distribution $p(X^{t-1})$, which represents the true probability value as an expectation considering numerous possible values of $\mathbf{y}_i^{t-1}$, will exhibit smoother behavior compared to the prior distribution. Using this assumption, the optimization problem suffices to:

$$X^{t-1(*)} \approx \underset{X^{t-1}}{\arg\max} \sum_{i=1}^{m} \log p(\tilde{\mathbf{y}}_i^{t-1} \mid X^{t-1}) \tag{28}$$

where we have also neglected all constant terms w.r.t. X^{t-1}. Using $\tilde{\mathbf{y}}_i^{t-1} \sim \mathcal{N}(f_i(X^{t-1}), \psi_i^2\mathbf{I})$, the problem becomes:

$$X^{t-1(*)} \approx \underset{X^{t-1}}{\arg\min} \sum_{i=1}^{m} \frac{1}{2\psi_i^2} \|\tilde{\mathbf{y}}_i^{t-1} - f_i(X^{t-1})\|^2 \tag{29}$$

We apply a gradient descent approach for optimizing X^{t-1}.

5.4 Auxiliary Variable Λ for Additional Flexibility

During experiments, we noticed some failure cases. We conjectured that the failure can be attributed to the excessive restriction imposed by the deterministic procedure described in Equation 15, which severely limits the flexibility in sampling. To mitigate this issue, we introduce auxiliary random variables $\Lambda^t = \{\lambda_i^t\}_{i=1}^m$ to the graphical model, as shown in Fig. 1-(right)-(b). λ_i^t replaces the deterministic procedure from Eq. 15 to:

$$\mathbf{y}_i^{t-1}(\lambda_i^t) = \sqrt{\bar{\alpha}_{t-1}}(\frac{\mathbf{y}_i^t - \sqrt{1-\bar{\alpha}_t} \cdot \lambda_i^t \cdot \epsilon_{\theta_i^*}^{(t)}(\mathbf{y}_i^t)}{\sqrt{\bar{\alpha}_t}}) + \sqrt{1-\bar{\alpha}_{t-1}} \cdot \lambda_i^t \cdot \epsilon_{\theta_i^*}^{(t)}(\mathbf{y}_i^t) \tag{30}$$

This is based on the intuition that predicted $\epsilon_{\theta_i^*}^{(t)}(\mathbf{y}_i^t)$ value is equivalent to a score Song & Ermon (2019); Ho et al. (2020); Song et al. (2020a); the gradient of log-probability distribution $p_{\theta_i^*}(\mathbf{y}_i)$. Hence, the procedure above can be viewed as pushing $\mathbf{y}_i^t$ towards the direction of $\epsilon_{\theta_i^*}^{(t)}(\mathbf{y}_i^t)$ with step-size λ_i^t, which increases the marginal probability of $\mathbf{y}_i$. Then, we can re-define the optimization problem as below:

$$X^{t-1(*)}, \Lambda^{t(*)} = \underset{X^{t-1}, \Lambda^t}{\arg\max}\, \mathcal{Q}'(X^{t-1}, \Lambda^t \mid X^t) \tag{31}$$

where the optimization objective is defined as:

$$\mathcal{Q}'(X^{t-1}, \Lambda^t \mid X^t) := \mathbb{E}_{Y^t, Y^{t-1}}[\log p(X^{t-1}, Y^{t-1}, Y^t, \Lambda^t \mid X^t)] \tag{32}$$

Following the similar way of Eq. 18∼21, the optimization objective is approximated as:

$$\mathcal{Q}'(X^{t-1}, \Lambda^t \mid X^t) \approx \sum_{i=1}^{m} \mathbb{E}_{\mathbf{y}_i^{t-1}}[\log p(\mathbf{y}_i^{t-1}, X^{t-1} \mid \lambda_i^t, \tilde{\mathbf{y}}_i^t) + \log p(\lambda_i^t) + \log p(\tilde{\mathbf{y}}_i^t \mid X^t)] \tag{33}$$

where $\tilde{\mathbf{y}}_i^t = \underset{\mathbf{y}_i^t}{\arg\max}\, p(\mathbf{y}_i^t \mid X^t) = f_i(X^{t-1})$. We now argue that:

$$\log p(\mathbf{y}_i^{t-1}, X^{t-1} \mid \lambda_i^t, \tilde{\mathbf{y}}_i^t) = \begin{cases} \log p(X^{t-1} \mid \tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t), \tilde{\mathbf{y}}_i^t) & \text{if } \mathbf{y}_i^{t-1} = \tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t) \\ 0 & \text{otherwise} \end{cases} \tag{34}$$

where $\tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t)$ is obtained via Eq. 30 using $\tilde{\mathbf{y}}_i^t$ and λ_i^t, since for any $\mathbf{y}_i^{t-1} \neq \tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t)$:

$$\int p(\mathbf{y}_i^{t-1}, X^{t-1} \mid \lambda_i^t, \tilde{\mathbf{y}}_i^t) dX^{t-1} = p(\mathbf{y}_i^{t-1} \mid \lambda_i^t, \tilde{\mathbf{y}}_i^t) = 0 \tag{35}$$

hence for $\forall X^{t-1}$:

$$p(\mathbf{y}_i^{t-1}, X^{t-1} \mid \lambda_i^t, \tilde{\mathbf{y}}_i^t) = 0 \tag{36}$$

and for the case of $\mathbf{y}_i^{t-1} = \tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t)$:

$$p(\tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t), X^{t-1} \mid \lambda_i^t, \tilde{\mathbf{y}}_i^t) = p(X^{t-1} \mid \tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t), \lambda_i^t, \tilde{\mathbf{y}}_i^t) p(\tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t) \mid \lambda_i^t, \tilde{\mathbf{y}}_i^t) = p(X^{t-1} \mid \tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t), \tilde{\mathbf{y}}_i^t) \tag{37}$$

since sampling for $\tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t)$ when $\lambda_i^t, \tilde{\mathbf{y}}_i^t$ given is a deterministic procedure. Using Eq. 34, the optimization objectives can be reduced to:

$$\mathcal{Q}'(X^{t-1}, \Lambda^t \mid X^t) \approx \sum_{i=1}^{m} [\log p(X^{t-1} \mid \tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t), \tilde{\mathbf{y}}_i^t) + \log p(\lambda_i^t) + \log p(\tilde{\mathbf{y}}_i^t \mid X^t)] \tag{38}$$

Applying Bayes' rule, we get:

$$\mathcal{Q}'(X^{t-1}, \Lambda^t \mid X^t) \approx \sum_{i=1}^{m} [\log \frac{p(\tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t) \mid X^{t-1}) p(X^{t-1})}{p(\tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t), \tilde{\mathbf{y}}_i^t)} + \log p(\lambda_i^t) + \log p(\tilde{\mathbf{y}}_i^t \mid X^t)] \tag{39}$$

Again, from $\tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t) \sim \mathcal{N}(f_i(X^{t-1}), \psi_i^2 \mathbf{I})$, if the magnitude of observation noise ψ_i is small, we can assume that marginal distribution $p(X^{t-1})$ is "locally constant" compared to $p(\tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t) \mid X^{t-1})$ within the region around optimal X^{t-1}, which reduces the optimization problem to:

$$X^{t-1(*)}, \Lambda^{t(*)} \approx \underset{X^{t-1}, \Lambda^t}{\arg\max} \sum_{i=1}^{m} [\log p(\tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t) \mid X^{t-1}) + \log p(\lambda_i^t)] \tag{40}$$

where we have also neglected all constant terms w.r.t. X^{t-1} and Λ^t. Using $\tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t) \sim \mathcal{N}(f_i(X^{t-1}), \psi_i^2 \mathbf{I})$, the problem becomes:

$$X^{t-1(*)}, \Lambda^{t(*)} \approx \underset{X^{t-1}, \Lambda^t}{\arg\min} \sum_{i=1}^{m} [\frac{1}{2\psi_i^2} \|\tilde{\mathbf{y}}_i^{t-1}(\lambda_i^t) - f_i(X^{t-1})\|^2 - \log p(\lambda_i^t)] \tag{41}$$

Intuitively, the above problem can be viewed as optimizing X^{t-1} with flexibility on the step size for an update in observable variables, and the negative-log-prior term (i.e., $-\log p(\lambda_i^t)$) can be seen as a regularization term for λ_i^t. In this work, we use L1-regularization for $\lambda_i^t - 1$; hence, Eq. 41 resembles the LASSO regression Ranstam & Cook (2018). We apply a gradient descent approach, same as in Sec. 5.3, for optimizing X^{t-1} and Λ^t.

6 Experiments

Our approach is a general solution for compositional generative tasks, providing broad applicability to diverse scenarios. In this section, we present a series of experiments that we conducted to evaluate the performance and effectiveness of our method. We provide detailed descriptions of these experiments along with their corresponding results.

6.1 Extensive Image Generation

Our method is capable of generating large images by generating local patches. such as panorama images. We demonstrate our method in the task of generating a panorama image composed of three consecutive local patches. In this case, the panorama image can be viewed as a control variable (i.e.,

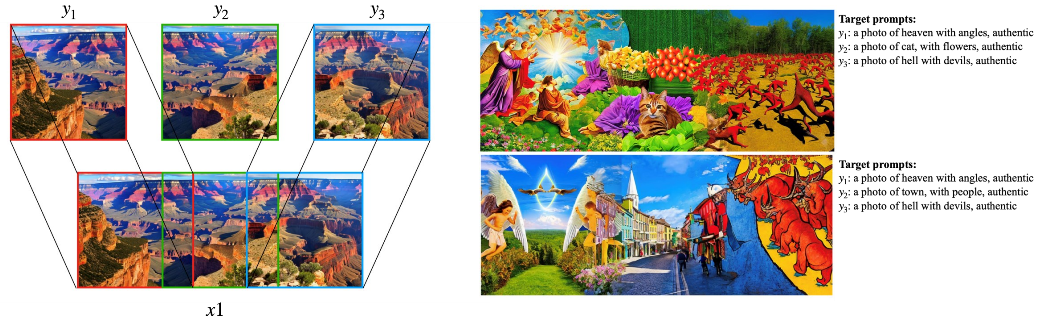

y_1
y_2
y_3
$x1$
Target prompts:
y_1: a photo of heaven with angles, authentic
y_2: a photo of cat, with flowers, authentic
y_3: a photo of hell with devils, authentic
Target prompts:
y_1: a photo of heaven with angles, authentic
y_2: a photo of town, with people, authentic
y_3: a photo of hell with devils, authentic

Figure 2: (left) Generated Panorama images using prompts with a shared prompt, ”a photo of the grand canyon”. (right) Generated Panorama images using different prompts for each patch

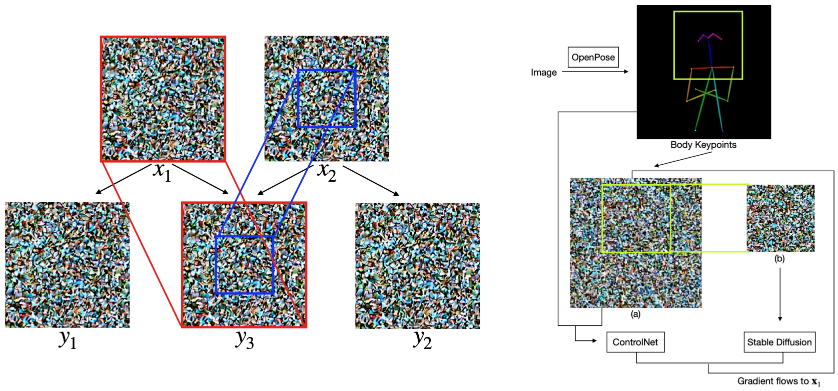


Figure 3: (left) Pipeline for producing y latent values using x values. (right) Pipeline for producing aligned face and body for Sec. 6.3.

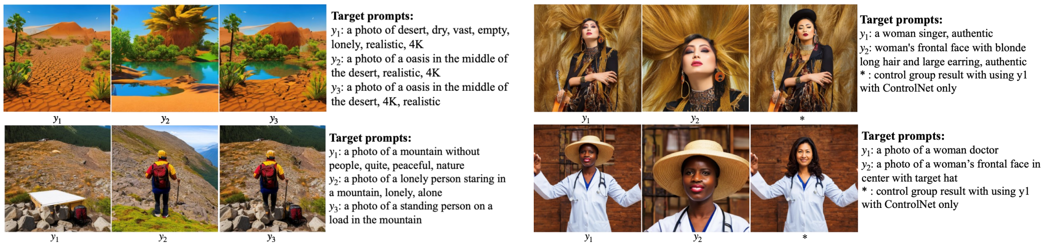
$\mathbf{x}_1$), and the local patches as observable variables (i.e., $\mathbf{y}_1 \sim \mathbf{y}_3$). Instead of training a diffusion model as in Sec. 5.2, we leverage a publicly available pretrained latent diffusion model Rombach et al. (2022) conditioned with the prompt we provide. We crop the control variable $\mathbf{x}_1$ to generate observable variables $\mathbf{y}_1 \sim \mathbf{y}_3$ along the wide panorama axis and force the patches to have consecutive, overlapped latent values. Each stable diffusion model computes the score functions value for each $\mathbf{y}_i$'s and optimizes control variable $\mathbf{x}_1$ to maximize our aggregated objective functions mentioned in Eq. 32.

We refer the readers to Fig. 2 for results. We observe that our method creates continuous and seamless panorama images. The generated local patches align with the provided prompt. We also provide additional results using different prompts for each local patch in Fig. 2. The results show a smooth transition between scenes, which demonstrates our method's compositional ability.

6.2 Compositional Scene Generation

In this section, we focus on the task of generating a background image and object simultaneously. To be specific, the control variables $\mathbf{x}_1, \mathbf{x}_2$ each denote the background without/with an object. The observable variables $\mathbf{y}_1$, $\mathbf{y}_2$ each are identical to $\mathbf{x}_1$, $\mathbf{x}_2$. Observable variable $\mathbf{y}_3$ is an image composited of “background region” of $\mathbf{x}_1$ and “object region” of $\mathbf{x}_2$. We manually set the background region of $\mathbf{x}_1$ and object region of $\mathbf{x}_2$ as partitions, where the object region is defined as an arbitrary bounding box. Refer to Fig 3-(left) for more details.

Our aim is to generate realistic and plausible composition $\mathbf{y}_3$, maintaining the background from $\mathbf{y}_1$ and object from $\mathbf{y}_2$. We set the conditioning prompt for $\mathbf{y}_1$, $\mathbf{y}_2$, $\mathbf{y}_3$ each to represent the background without an object, an object, and an overall scene with $\mathbf{y}_1$'s background and object from $\mathbf{y}_2$. For $\mathbf{y}_3$'s conditioning prompt, we do not provide a detailed description of the object as we represent the overall scene.



Target prompts:
y_1: a photo of desert, dry, vast, empty, lonely, realistic, 4K
y_2: a photo of a oasis in the middle of the desert, realistic, 4K
y_3: a photo of a oasis in the middle of the desert, 4K, realistic

Target prompts:
y_1: a photo of a mountain without people, quite, peaceful, nature
y_2: a photo of a lonely person staring in a mountain, lonely, alone
y_3: a photo of a standing person on a load in the mountain

Target prompts:
y_1: a woman singer, authentic
y_2: woman's frontal face with blonde long hair and large earring, authentic
* : control group result with using y1 with ControlNet only

Target prompts:
y_1: a photo of a woman doctor
y_2: a photo of a woman's frontal face in center with target hat
* : control group result with using y1 with ControlNet only

Figure 4: (left) A result of the compositional scene generation in Sec. 6.2 with used target prompts. (right) Generated body image with different prompt conditioning.

Results are shown in Fig. 4-(left). The first row of generated images aligns with the conditioning prompt as well, and we observe that $\mathbf{y}_3$ blends naturally. From the second row, we observe that we can generate an overall scene with the detailed object in $\mathbf{y}_3$ without directly inserting the detailed explanation to $\mathbf{y}_3$.

6.3 Composable Body Synthesis

Our method can be applied to generate a full body with detailed local parts by compositing multiple diffusion models with diverse conditions. Compared to compositional scene generation in Sec. 6.2, it requires accurate alignment between body and parts as plausible locations of parts are strongly conditioned on body pose. Here we show a few examples of merging the full body and face as local parts. In this task, we define single control variable $\mathbf{x}_1$ corresponding to 96×96 dimension latent, which encodes 768×768 body image. Then set two observation variables corresponding to full body $\mathbf{y}_1$ identical to $\mathbf{x}_1$ and $\mathbf{y}_2$ for closed-up face area, which is a crop of $\mathbf{x}_1$ with latent size 48×48. Different from Sec. 6.2, instead of randomly placing cropping bounding box for $\mathbf{y}_2$, we place the bounding box on a fixed point where the head locates. To adjust the head positioning of the generated image, we used ControlNet Zhang & Agrawala (2023) to estimate the score for full body $\mathbf{y}_1$ with OpenPose Cao et al. (2019) body keypoint conditioning as shown in Fig. 3-(right).

To check the composition of the head and body is successful, we gave a detailed prompt only on the diffusion model for face $\mathbf{y}_2$. As shown in Fig. 4-(right), our method succeed in generating images consistent with both of the prompts, even though the composed result is rare in the real world. When comparing the generated results to a control group, where the influence of $\mathbf{y}_2$ has been eliminated, we can observe that the desired property of the prompt is effectively injected into the outcome in a compositional manner.

7 Conclusion

We have presented a novel approach to address the compositional generation problem by formulating it as a Bayesian inference problem. Our formulation is versatile and applicable to a wide range of tasks in different scenarios. Under the provided formulation, we propose an optimization-based sampling method inspired by Markov Chain Monte Carlo (MCMC) techniques. The method leverages variational inference with diffusion models and aggregates information from these models to devise optimization-based adaptive proposals used for iterative sampling. Our method has demonstrated high performance across various tasks, as evidenced by our qualitative results.

However, it is important to acknowledge the limitations of our method. Specifically, our approach is currently only justified for scenarios with zero Langevin noise. It would be valuable for future work to explore and address the case with non-zero Langevin noise, as this may potentially enhance the performance of our method.

Furthermore, our provided formulation for the compositional generation problem offers flexibility in terms of the choice of stochastic dependency (i.e., $p(Y \mid X)$) and the information types for observables (i.e., $p(Y)$). This implies that there are numerous design choices available, which we leave as avenues for future research.

REFERENCES

Omri Avrahami, Ohad Fried, and Dani Lischinski. Blended latent diffusion. *arXiv preprint arXiv:2206.02779*, 2022.

Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. Multidiffusion: Fusing diffusion paths for controlled image generation. *arXiv preprint arXiv:2302.08113*, 2, 2023.

Andreas Blattmann, Robin Rombach, Kaan Oktay, Jonas Müller, and Björn Ommer. Retrieval-augmented diffusion models. *Advances in Neural Information Processing Systems*, 35:15309–15324, 2022.

Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22563–22575, 2023.

Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18392–18402, 2023.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986. doi: 10.1109/TPAMI.1986.4767851.

Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

Ayaan Haque, Matthew Tancik, Alexei A Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. *arXiv preprint arXiv:2303.12789*, 2023.

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

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.

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

Hiromichi Kamata, Yuiko Sakuma, Akio Hayakawa, Masato Ishii, and Takuya Narihira. Instruct 3d-to-3d: Text instruction guided 3d-to-3d conversion. *arXiv preprint arXiv:2303.15780*, 2023.

Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6007–6017, 2023.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Nupur Kumari, Bingliang Zhang, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Multi-concept customization of text-to-image diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1931–1941, 2023.

Steven J Luck and Andrew Hollingworth. *Visual memory*. OUP USA, 2008.

Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. Realfusion: 360deg reconstruction of any object from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8446–8455, 2023.

Chenlin Meng, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International conference on machine learning*, pp. 3481–3490. PMLR, 2018.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Todd K Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6): 47–60, 1996.

Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11453–11464, 2021.

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 *International conference on machine learning*, pp. 8748–8763. PMLR, 2021a.

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 *International conference on machine learning*, pp. 8748–8763. PMLR, 2021b.

Jonas Ranstam and JA Cook. Lasso regression. *Journal of British Surgery*, 105(10):1348–1348, 2018.

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*, pp. 10684–10695, 2022.

Andrea Ruggieri, Francesco Stranieri, Fabio Stella, and Marco Scutari. Hard and soft em in bayesian network learning from incomplete data. *Algorithms*, 13(12):329, 2020.

Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020a.

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

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020b.

Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H Bermano. Human motion diffusion model. *arXiv preprint arXiv:2209.14916*, 2022.

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Lvmin Zhang and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. *arXiv preprint arXiv:2302.05543*, 2023.

Qinsheng Zhang, Jiaming Song, Xun Huang, Yongxin Chen, and Ming-Yu Liu. Diffcollage: Parallel generation of large content with diffusion models. *arXiv preprint arXiv:2303.17076*, 2023.