DiffCollage: Parallel Generation of Large Content with Diffusion Models

Qinsheng Zhang Jiaming Song Xun Huang Yongxin Chen Ming-Yu Liu
{qzhang419,yongchen}@gatech.edu {jiamings,xunh,mingyul}@nvidia.com
Georgia Institute of Technology NVIDIA Corporation

Abstract

We present `DiffCollage`*, a compositional diffusion model that can generate large content by leveraging diffusion models trained on generating pieces of the large content. Our approach is based on a factor graph representation where each factor node represents a portion of the content and a variable node represents their overlap. This representation allows us to aggregate intermediate outputs from diffusion models defined on individual nodes to generate content of arbitrary size and shape in parallel without resorting to an autoregressive generation procedure. We apply* `DiffCollage` *to various tasks, including infinite image generation, panorama image generation, and long-duration text-guided motion generation. Extensive experimental results with a comparison to strong autoregressive baselines verify the effectiveness of our approach.*

1. Introduction

The success of diffusion models [20, 56] can largely be attributed to their scalability. With large-scale datasets and computing resources, practitioners can usually train high-capacity models that are able to produce high-fidelity images. The recent generative AI revolution led by large-scale text-to-image diffusion models is a great example [3, 44, 49]. The same procedure, collecting a large dataset and using it to train a large-scale model, has been applied to various problems and achieved great success [4, 43].

In this paper, we are interested in extending the success of diffusion models to a wider class of data. We focus on applications where a large-scale dataset of the target content does not exist or is prohibitively expensive to collect, but individual pieces of the content are available in great quantities. 360-degree panorama images are such an example. While 360-degree panorama images are considered niche image content and only exist in small quantities, there are a large number of normal perspective images available on the Internet, each of which can be treated as a piece of a 360-degree panorama image. Another example is generating images of extreme aspect ratios, as shown in Fig. 1. Each of the extreme-aspect-ratio images can be considered as the stitching of multiple images with normal aspect ratios. For such applications, while we cannot afford to collect a large-scale dataset of the target content to train a diffusion model, we wish to synthesize high-quality target content with a diffusion model trained on smaller pieces that are readily available.

A popular solution to this class of problems is to first train a diffusion model on small pieces of the content and then generate the large content piece by piece in an autoregressive manner [14, 48]. However, such an autoregressive approach has three drawbacks. First, as pieces are generated sequentially, the later-generated pieces have no influence on the prior-generated ones. Such a sequential scheme could lead to sub-optimal results, especially when there is a circular structure in the data. For example, it is hard to enforce consistency between the start and end frames when generating looped videos autoregressively. Second, autoregressive methods may suffer from error accumulation since the model was conditioned on ground-truth data during training but is conditioned on its own prediction at test time. Lastly, the time consumption of autoregressive generation increases linearly with the size of the data and could become prohibitive when generating very large content.

To address the large content generation problem, we propose `DiffCollage`, a generic algorithm that synthesizes large content by merging the results generated by diffusion models trained on small pieces of the large content. Our approach is based on a factor graph formulation where a datum is modeled by a set of nodes and the edges connecting them. In our formulation, each node represents a contiguous portion of the large content, and the portions of content in neighboring nodes have a small overlap. Each node is associated with a small diffusion model and each piece affects the generation of the other pieces. Our method generates multiple pieces of content in parallel, which can greatly accelerate sampling when a large pool of computation is available.

We evaluate our approach on multiple large content generation tasks, including infinity image generation, long-duration text-to-motion with complex actions, content with unusual structures such as looped motion, and 360-degree

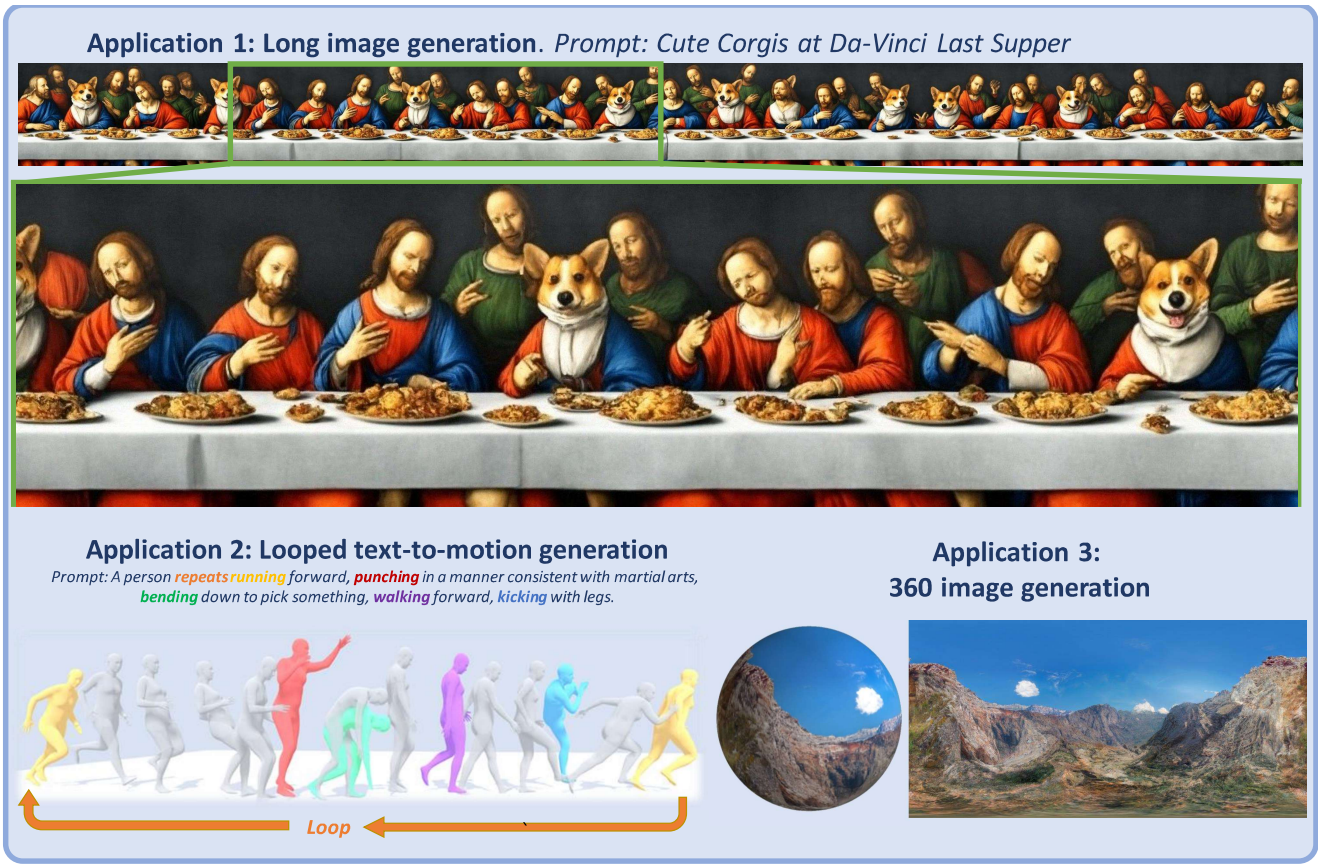

Figure 1. DiffCollage, a scalable probabilistic model that synthesizes large content, including long images, looped motions, and 360 images, with diffusion models only trained on pieces of the content.

images. Experiment results show that our approach outperforms existing approaches by a wide margin.

In summary, we make the following contributions.

- We propose DiffCollage, a scalable probabilistic model that synthesizes large content by merging results generated by diffusion models trained on pieces of the large content. It can synthesize large content efficiently by generating pieces in parallel.
- DiffCollage can work out-of-the-box when pre-trained diffusion models on different pieces are available.
- Extensive experimental results on benchmark datasets show the effectiveness and versatility of the proposed approaches on various tasks.

2. Related Work

Diffusion models Diffusion models [20, 54, 56] have achieved great success in various problems, such as text-to-image generation [3, 44, 49], time series modeling [58], point cloud generation [64, 68, 72], natural language processing [30], image editing [11, 28, 37, 60], inpainting [8, 26, 27, 35], and adversarial defense [39]. Recently, impressive progress has been made in improving its quality [3, 44, 45, 49], controllability [15, 18, 21, 28, 37, 46], and efficiency [24, 55, 69, 70]. In this paper, we aim to enlarge the kind of data that diffusion models can generate.

Large content generation Generating large content with generative models trained on small pieces of large content has been explored by prior works. One class of methods relies on latent variable models, *e.g.*, GANs [16], that map a global latent code and a spatial latent code to an output image. The global latent code represents the holistic appearance of the image and the spatial latent code is typically computed from a coordinate system. Some works [31, 53] generate different patches using the same global code and merge them to obtain the full image. A discriminator can be used to ensure the coherence of the full image. Instead of generating patches independently, some recent works generate the full image in one shot using architectures that guarantee translation equivariance, such as padding-free generators [32, 40, 57] or implicit MLP-based generators [2, 5, 52].

Another popular approach to generating large content

is to autoregressively apply "outpainting" to gradually enlarge the content. The outpainting could be implemented by a diffusion model [9, 10, 22, 26, 35, 48], an autoregressive transformer [7, 14, 63], or a masked transformer [6, 7, 71].

3. Preliminaries

Diffusion models consist of two processes: a forward diffusion process and a reverse process. The forward diffusion process progressively injects Gaussian noise into samples from the data distribution $q_0(\boldsymbol{u}_0)$ and results in a family of noised data distributions $q_t(\boldsymbol{u}_t)$. It can be shown that the distribution of $\boldsymbol{u}_t$ conditioned on the clean data $\boldsymbol{u}_0$ is also Gaussian: $q_{0t}(\boldsymbol{u}_t|\boldsymbol{u}_0) = \mathcal{N}(\boldsymbol{u}_0, \sigma_t^2 \boldsymbol{I})$. The standard deviation σ_t monotonically increases with respect to the forward diffusion time t. The reverse process is designed to iteratively remove the noise from the noised data to recover the clean data, which can be formulated as the following stochastic differential equation (SDE) [25, 50, 70]

$$d\boldsymbol{u} = -(1+\eta^2)\dot{\sigma}_t \sigma_t \nabla_{\boldsymbol{u}} \log q_t(\boldsymbol{u}) dt + \eta \sqrt{2\dot{\sigma}_t \sigma_t} d\boldsymbol{w}, \quad (1)$$

where $\nabla_{\boldsymbol{u}} \log q_t(\boldsymbol{u})$ is the score function of a noised data distribution, $\boldsymbol{w}_t$ is the standard Wiener process, and $\eta \geq 0$ determines the amount of random noise injected during the denoising process. When $\eta = 1$, Eq. (1) is known as reverse-time SDE of the forward diffusion process [1, 56], from which ancestral sampling and samplers based on Euler-Maruyama can be employed [20, 56]. Eq. (1) reduces to a probability flow ODE when $\eta = 0$ [56]. In practice, the unknown score function $\nabla_{\boldsymbol{u}} \log q_t(\boldsymbol{u})$ is estimated using a neural network $\boldsymbol{s}_\theta(\boldsymbol{u}, t)$ by minimizing a weighted sum of denoising autoencoder (score matching [61]) objectives:

$$\arg\min_\theta \mathbb{E}_{t, \boldsymbol{u}_0}[\omega(t) \| \nabla_{\boldsymbol{u}_t} \log q_{0t}(\boldsymbol{u}_t|\boldsymbol{u}_0) - \boldsymbol{s}_\theta(\boldsymbol{u}_t, t)\|^2], \quad (2)$$

where $\omega(t)$ denotes a time-dependent weight.

4. Diffusion Collage

`DiffCollage` is an algorithm that can generate large content in parallel using diffusion models trained on data consisting of portions of the large content. We first introduce the data representation and then discuss training and sampling. For simplicity, we derive the formulation for unconditional synthesis throughout this section; the formulation can be easily extended to conditional synthesis.

4.1. Representation

A simple example A simple use case of `DiffCollage` is to generate a long image by assembling diffusion models trained on shorter images. An autoregressive solution to this problem is to first generate an initial square image and then perform outpainting conditioned on a part of the previously generated image [48], which results in a slightly larger output

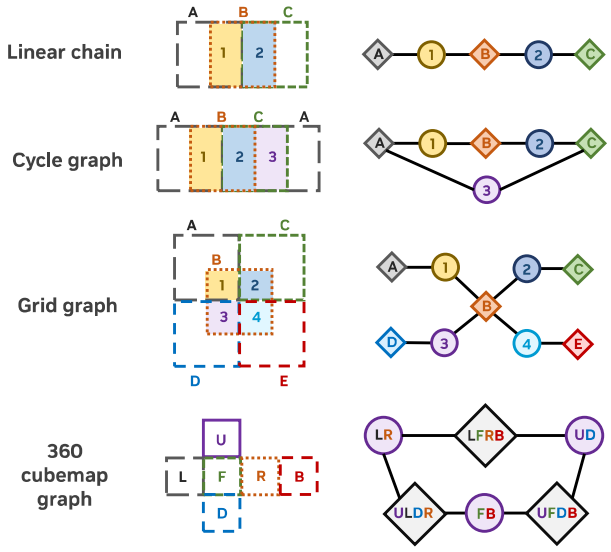

Figure 2. **Factor graphs for various applications.** From top to bottom: a linear chain for arbitrarily long sequences, a cycle graph for arbitrarily long loops, a grid graph for images of arbitrary height and width, and a complex factor graph for 360-degree panoramas.

image. We denote this larger image as $\boldsymbol{u} = [\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}]$ where $[\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}]$ is the initial image and $\boldsymbol{x}^{(3)}$ is the outpainted region generated by the conditional model $\boldsymbol{x}^{(3)}|\boldsymbol{x}^{(2)}$. Notably, this procedure makes a conditional independence assumption: conditioned on $\boldsymbol{x}^{(2)}$, $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(3)}$ are independent, *i.e.*, $q(\boldsymbol{x}^{(3)}|\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}) = q(\boldsymbol{x}^{(3)}|\boldsymbol{x}^{(2)})$. Therefore, the joint probability is

$$\begin{aligned} q(\boldsymbol{u}) &= q(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}) = q(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}) q(\boldsymbol{x}^{(3)}|\boldsymbol{x}^{(2)}) \\ &= \frac{q(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}) q(\boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)})}{q(\boldsymbol{x}^{(2)})}. \end{aligned} \quad (3)$$

The score function of $q(\boldsymbol{u})$ can be represented as a sum over the scores of smaller images

$$\begin{aligned} \nabla \log q(\boldsymbol{u}) = \nabla \log q(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}) + \nabla \log q(\boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}) \\ - \nabla \log q(\boldsymbol{x}^{(2)}). \end{aligned} \quad (4)$$

Each individual score can be estimated using a diffusion model trained on smaller images. Unlike the autoregressive method, which generates content sequentially, `DiffCollage` can generate different pieces in parallel since all individual scores can be computed independently.

Generalization to arbitrary factor graphs Now, we generalize the above example to more complex scenarios. For a joint variable $\boldsymbol{u} = [\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)}]$, a factor graph [29] is a bipartite graph connecting variable nodes $\{\boldsymbol{x}^{(i)}\}_{i=1}^n$ and factor nodes $\{f^{(j)}\}_{j=1}^m$, where $f^{(j)} \subseteq$

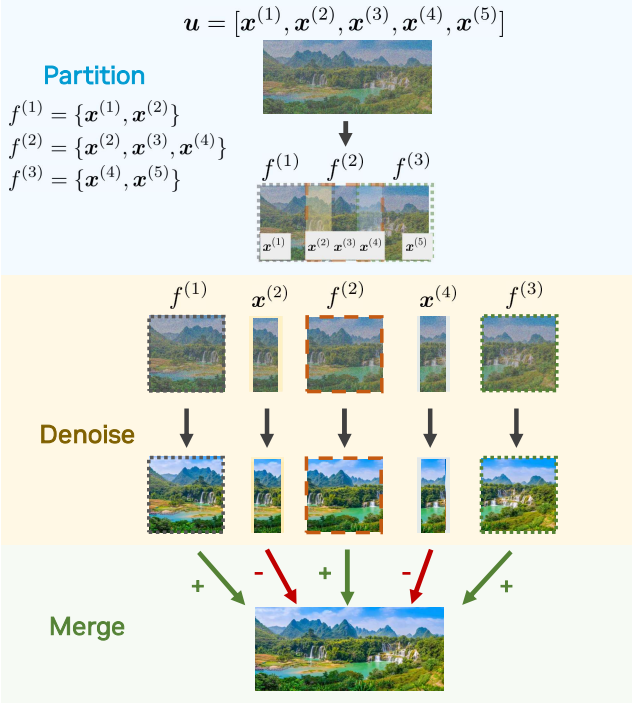

Figure 3. How `DiffCollage` synthesize long images. To calculate the score for each denoising step on the long image, we split the input based on the factor graph into regions of factor nodes and variables. We obtain the scores of individual regions by using the individual diffusion models. We then merge the scores to compute the score of the target diffusion model on the long image.

$\{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \dots, \boldsymbol{x}^{(n)}\}$. An undirected edge between $\boldsymbol{x}^{(i)}$ and $f^{(j)}$ exists if and only if $\boldsymbol{x}^{(i)} \in f^{(j)}$. In the above example, there are two factors $f^{(1)} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}\}$ and $f^{(2)} = \{\boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}\}$. Given a factor graph that represents the factorization of the joint distribution $q(\boldsymbol{u})$[1], `DiffCollage` approximates the distribution as follows:

$$p(\boldsymbol{u}) := \frac{\prod_{j=1}^{m} q(f^{(j)})}{\prod_{i=1}^{n} q(\boldsymbol{x}^{(i)})^{d_i - 1}}, \tag{5}$$

where d_i is the degree of the variable node $\boldsymbol{x}^{(i)}$. It is easy to verify that Eq. (5) reduces to Eq. (3) in the simple case since the nodes for $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(3)}$ have a degree of one (connected to $f^{(1)}$ and $f^{(2)}$ respectively) and the node for $\boldsymbol{x}^{(2)}$ has a degree of two (connected to both $f^{(1)}$ and $f^{(2)}$). Similar to Eq. (4), we can approximate the score of $q(\boldsymbol{u})$ by adding the scores over factor nodes (*i.e.*, $q(f^{(j)})$) and subtracting the scores over non-leaf variable nodes (*i.e.*, $q(\boldsymbol{x}^{(i)})$)

$$\nabla \log p(\boldsymbol{u}) := \sum_{j=1}^{m} \nabla \log q(f^{(j)}) + \sum_{i=1}^{n} (1 - d_i) \nabla \log q(\boldsymbol{x}^{(i)}). \tag{6}$$

[1]In order words, the joint distribution can be written as a product of functions, each of which is a function of a single factor.

In fact, Eq. (5) is also known in the probabilistic graphical model literature as the seminal *Bethe approximation*, which approximates the joint distribution $q(\boldsymbol{u})$ by its marginals defined over factor and variable nodes [65, 66]. The approximation is exact, *i.e.*, $p(\boldsymbol{u}) = q(\boldsymbol{u})$, when the factor graph is an acyclic graph. For a general graph with cycles, the Bethe approximation is widely used in practice and obtains good performance [29, 51]. More discussions and justification on Bethe approximation are in the supplementary material.

In practice, factor graphs are general enough to cover contents of arbitrary size and shape, such as those in Fig. 2:

- **An arbitrarily long sequence**: The factor graph is a linear chain in which each factor is connected to two variables, and each variable is connected to two factors (except for leaf variables). We show a detailed characterization in Fig. 3.
- **An arbitrarily long sequence with a loop**: Similar to the linear chain but with a variable node connecting the head and tail factor nodes.
- **An image of arbitrary height and width**: Here, each factor is an image patch that overlaps with 4 other factors at the 4 corners. Each overlapping region is a variable. Thus, each factor node is connected to 4 variables, and each variables node is connected to 2 factors (except for edge cases).
- **A 360-degree image represented as a cubemap**: A cube consists of 6 faces: **F**ront, **B**ack, **L**eft, **R**ight, **U**p, **D**own. There are three cycles (LFRB, ULDR, UFDB) can be modeled via the cycle graph, and these cycles overlap one another by two faces. Intuitively, we can treat these cycles as factors and faces as variables. We list more details in the supplementary material.

4.2. Training and Sampling

Training `DiffCollage` is trained to estimate the score of noised data distributions $q_t(\boldsymbol{u})$. Similar to the Bethe approximation (Eqs. (5) and (6)) for clean data, we factorize the score of the time-dependent noised data distributions:

$$\nabla \log p_\theta(\boldsymbol{u}, t) = \sum_{j=1}^{m} \nabla \log p_\theta(f^{(j)}, t) + \sum_{i=1}^{n} (1 - d_i) \nabla \log p_\theta(\boldsymbol{x}^{(i)}, t). \tag{7}$$

To close the gap between our learned model and the Bethe approximation in Eq. (5), we optimize θ by performing denoising score matching between the marginal scores of real data $\{q(\boldsymbol{x}^{(i)}, t), q(f^{(j)}, t)\}$ and learned marginal scores $\{p_\theta(\boldsymbol{x}^{(i)}, t), p_\theta(f^{(j)}, t)\}$ (following Eq. (2)). This can be done by learning a diffusion model for each marginal distribution of real data; we list the detailed algorithm for training in the supplementary material.

Figure 4. Long images generated by various approaches that only use diffusion models trained on smaller square images. For autoregressive approaches (Replace and Recon), we first generate the image in the middle then outpaint towards left and right. Replace and Recon introduce discontinuity artifacts while `DiffCollage` can generate high-fidelity images in parallel.

It should be noted that even though we aim to approximate one joint distribution, learning a diffusion model for one marginal distribution is independent of learning other marginals. With such independence, diffusion models on different marginals can be learned in parallel. Practically, diffusion models on different marginals can be amortized where we employ one shared diffusion model with conditional signals $\boldsymbol{y}[f^{(j)}]$ from factor node $f^{(j)}$ and $\boldsymbol{y}[i]$ from variable node $\boldsymbol{x}^{(i)}$ to learn various marginals.

Sampling After training the diffusion models for each marginal, the score model of `DiffCollage` for $p_\theta(\boldsymbol{u}, t)$ is simply obtained via Eq. (7), and it is a diffusion model with a specific score approximation. Thus `DiffCollage` is sampler-agnostic, and we can leverage existing solvers for Eq. (1) to generate samples with the approximated score in Eq. (7), such as DDIM [55], DEIS [69], DPM-Solver [34] and gDDIM [70], all without any modifications. We emphasize that diffuson models on various marginals can be evaluated at the same time and generate different pieces of data $\{f^{(j)}, \boldsymbol{x}^{(i)}\}$ in parallel, unlike the conventional autoregressive approaches, so with advanced samplers, the number of iterations taken by `DiffCollage` could be much less than that of an autoregressive model.

5. Experiments

Here, we present quantitative and qualitative results to show the effectiveness and efficiency of `DiffCollage`. We perform experiments on various generation tasks, such as infinite image generation (Sec. 5.1), arbitrary-sized image translation (Sec. 5.2), motion synthesis (Sec. 5.3), and 360-degree panorama generation (Sec. 5.4).

5.1. Infinite image generation

We first evaluate `DiffCollage` in the infinite image generation task [53] where the goal is to generate images extended to infinity horizontally. We employ a linear chain as shown in Fig. 2 and use **the same** score network for all factor nodes and variable nodes since the marginal image distribution is shift-invariant.

We finetune a pre-trained GLIDE model [38], which is a two-stage diffusion model consisting of a 64×64 square generator and one $64 \to 256$ upsampler on an internal landscape dataset. We additionally finetune a pre-trained eDiff-I [3] $256 \to 1024$ upsampler. Combining them together, we have a score model for individual nodes that can generate images of resolution up to 1024×1024. To control the style of the output [3, 44], the base diffusion model is conditioned on CLIP [43] image embeddings.

Other diffusion-based approaches tackle this problem by performing outpainting autoregressively [48]. Specifically, it generates the first image using a standard diffusion model, then extends the image through repeated application of outpainting toward left and right. The outpainting problem can be treated as an inpainting problem with 50% of the content masked out. While there exist diffusion models specifically trained for inpainting [48], we only perform comparisons with other generic methods that work on any pretrained diffusion models. We compare `DiffCollage` with two inpainting approaches. The first one is the "**replacement**" approach, where we constantly replace part of intermediate predictions with known pixels [8, 26, 35]. The second is the "**reconstruction**" approach, which uses the gradient of the reconstruction loss on known pixels to correct the unconditional samples [9, 22, 47]. This approach is slightly more computationally expensive since it needs to compute the gradient of the reconstruction loss w.r.t the intermediate predictions. We discuss more details in the supplementary.

To compare the generation quality of generated panorama images, we propose *FID Plus (FID+)*. We first generate 50k panorama images with spatial ratio $W/H = 6$ and randomly crop one square image $H \times H$ for each image. FID+ is the Frechet inception distance (FID, [19]) of the 50k randomly cropped images. We also include a **baseline** that naively concatenates independently generated images of $H \times H$ into a long image. Although each generated image is realistic, this approach has a bad FID+ because randomly cropped images may contain clear boundaries.

As shown in Tab. 1, `DiffCollage` outperforms other approaches in terms of sample quality evaluated by FID+. In Fig. 4, we show that the sample quality of autoregressive approaches deteriorates as the image grows due to error accumulation, while `DiffCollage` does not have this issue. Fig. 5 compares the latency of generating one panorama image with different image sizes. Thanks to its parallelization, `DiffCollage` is about $H/2W$ times and H/W times faster than replacement and reconstruction methods respectively when generating one $H \times W$ image with $H \geq 2W$. We further apply `DiffCollage` to eDiff-I [3], a recent large-scale text-to-image model in Fig. 1, to generate a wide image from the text prompt *"Cute Corgis at Da-Vinci Last Supper"*, which demonstrates the general applicability of `DiffCollage` to arbitrary diffusion models.

In addition, the score models for different nodes can be conditioned on different control signals. We illustrate this point by connecting any two landscape images of different styles. This is a challenging inpainting task where only pixels at two ends are given. The score models for intermediate nodes are conditioned on interpolated CLIP embeddings. As shown in Fig. 6, we are able to generate a long image that transitions from one style to another totally different one.

We compare `DiffCollage` with other methods specifically designed for long image generation tasks on LHQ [53] and LSUN Tower [67], following the setting in Skorokhodov *et al.* [53]. We evaluate standard FID over images of size $H \times H$, as well as FID+ in Tab. 2. As shown in the table, even though our approach is never trained on long image generation, it achieves competitive results compared with methods that are tailored to the task and require problem-specific networks for the image dataset.

Algo	Parallel	Gradients	FID+ ↓	Time ↓
Baseline	-	-	24.15	5.61
Replacement	No	Not required	10.25	14.99
Reconstruction	No	Required	8.97	26.43
Ours	Yes	Not required	**4.54**	**6.47**

Table 1. Comaprison among diffusion-based methods for infinite image generation on an internal landscape dataset. Our method achieves the best image quality while also being the fastest since we can compute individual scores in parallel and do not require backpropagating through the diffusion model to obtain gradients.

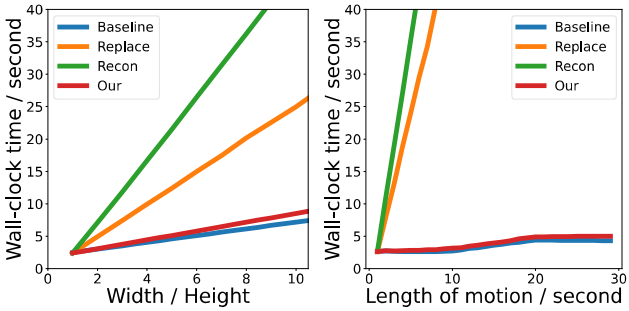

Figure 5. Wall-clock time for generating images with various lengths (Left) and motion sequences with various durations (Right).

Dataset	LHQ 256^2		Tower 256^2	
	FID	FID+	FID	FID+
VQGAN [14]	58.27	62.12	45.18	47.32
ALIS [53]	12.60	14.27	11.85	15.27
Replacement	6.28	28.94	7.15	30.19
Reconstruction	6.28	18.37	7.15	19.56
Ours	6.28	**16.43**	7.15	**13.27**

Table 2. Comparison against methods specifically designed for infinity image generation (Dark-colored rows). Our approach achieves higher quality despite being more general.

5.2. Arbitrary-sized image translation

Our method can be applied to various image translation tasks where the size of the input image is different from what the diffusion model is trained on. We use `DiffCollage` to aggregate the scores of individual nodes and the score of each node can be estimated using methods that are de-

Figure 6. **Connecting real images.** Given a pair of 64×64 real images $\boldsymbol{x}^{(0)}$ and $\boldsymbol{x}^{(N)}$, `DiffCollage` can generate a 1024×10752 image that transitions naturally from $\boldsymbol{x}^{(0)}$ into $\boldsymbol{x}^{(N)}$.

Figure 7. **Inpainting on non-square images.** The first row contains two masked images. The second row contains the inpainting results by splitting the input non-square images into a set of square images. In the left/right example, the input image is split into two/three square images. The Recon results in apparent boundary artifacts. The third row contains the inpainting results with `DiffCollage`.

veloped for standard diffusion models, such as replacement [10, 26, 35] or reconstruction methods [9, 22] for inpainting, and SDEdit [37] for stroke-based image synthesis. To achieve these with existing methods, one could also split the large image into several smaller ones and apply the conditional generation methods independently. However, this fails to model the interactions between the split images, resulting in discontinuities in the final image; we illustrate this in Fig. 7 for the task of inpainting from sparse pixels. In contrast, `DiffCollage` can capture global information and faithfully recover images from given sparse pixels. We include more image translation results in the supplementary.

5.3. Text-to-motion generation

Given a text description of the desired motion, the goal of this task is to synthesize a motion sequence corresponding to the description. In this section, we evaluate our approach on the popular benchmark HumanML3D [17, 36], using a pre-trained motion diffusion model [59]. The pre-trained diffusion model was trained with sequences of various lengths. As a result, it can use be used as the score estimator for both factor nodes and variable nodes in our formulation directly.

High-fidelity generated samples are expected to follow basic rules of physics and behave similarly to realistic human motions. Specifically, we adopt the set of metrics from Guo *et al.* [17], including *R-precision* and *Multimodal-Distance* that quantify the alignment between generated samples and the given prompt, *FID* that measures the distance between the distribution of ground truth motions and generated motions, and *Diversity* that measures the variability in samples generated by our methods.

Long-duration motion generation In HumanML3D, the average motion length is 7.1s, and the maximum duration is 10s. Our goal is to generate high-fidelity motion sequences that are much longer than what we have in the training data. To achieve this, we use a linear chain graph similar to the one used in infinite image generation. To evaluate our method, we generate a 24s motion for each text and randomly crop generated sequences, analogous to FID+ for images.

We compare our approach with several methods, including naively denoising a long sequence (Baseline) and autoregressive generation with replacement and reconstruction methods, respectively. The results in Tab. 3 show that `DiffCollage` outperforms other approaches in all evaluated metrics by a notable amount.

Compositing multiple actions The existing human motion generative model can only synthesize simple motions since there are only one or two actions for one motion sequence in the training dataset. With `DiffCollage`, we can augment the simple motion diffusion model with the ability to synthesize complex actions. We use the desired text prompts for the conditions of factors $\boldsymbol{y}[f_j]$, and the un-

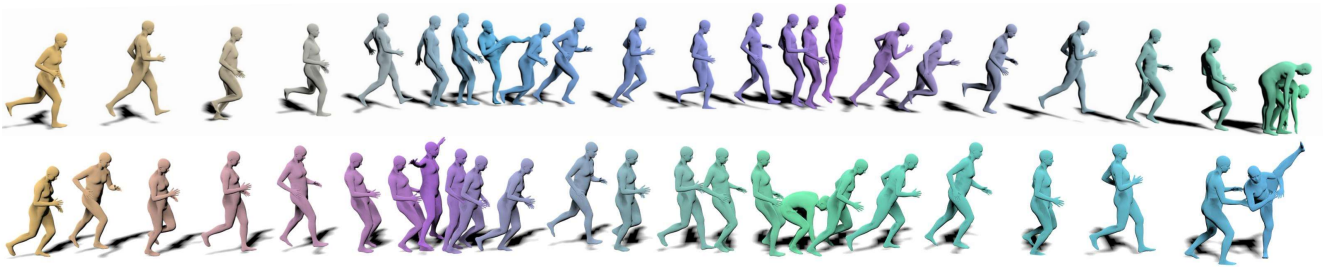

Figure 8. Complex motions synthesis. Though the pre-trained motion diffusion model [59] can only generate simple motions with one or two actions, `DiffCollage` can extend it to synthesize long sequences with an arbitrary number of actions. Prompts: (Top) *A person runs forward, then kicks his legs, then skips rope, then bends down to pick something up off the ground.* (Bottom) *A person runs forward, then skips rope, then bends down to pick something up off the ground, then kicks his legs.*

Method	R Precision (top 3)↑	FID↓	Multimodal Dist↓	Diversity→
Real data	0.798	0.001	2.960	9.471
MDM [59]	0.605	0.492	5.607	9.383
Baseline	0.298	10.690	7.512	6.764
Replacement	0.567	1.281	5.751	9.184
Reconstruction	0.585	1.012	5.716	9.175
Ours	**0.611**	**0.605**	**5.569**	**9.372**

Table 3. Quantitative results of long-duration generation on the HumanML3D test set [17]. Dark-colored rows are the results of short-duration motion samples (for reference only), while other rows evaluate methods that generate 24 seconds of motion, which is around 4 times longer than the average length of training data. All methods are based on pre-trained MDM [59]. → means the results are better if the metric is closer to real data.

conditional null token for that of variables $\boldsymbol{y}[i]$. As shown in Fig. 8, by constructing graphs with different marginal distributions specified by different conditions $\boldsymbol{y}[f_j], \boldsymbol{y}[i]$, we can generate complex motion sequences.

5.4. Generation with complex graphs

We further show that `DiffCollage` is able to generate data with a challenging dependency structure specified by a complex graph (such as the ones in Fig. 9). As shown in Fig. 9 (top), `DiffCollage` can generate a horizontal panorama by constructing a cycle graph. We also apply our method to generate a 360-degree panorama using a diffusion model trained only on normal perspective images conditioned on semantic segmentation maps (Fig. 9 bottom). This allows users to create beautiful panoramas from simple doodles, similar to some existing applications such as GauGAN [42] and GauGAN2 [23] but providing a more immersive experience to users.

6. Conclusion

In this work, we propose `DiffCollage`, a novel diffusion model that can synthesize large content via a collection of diffusion models trained on pieces of large content. `DiffCollage` is based on factor graph representation and inspired by Bethe approximation, both commonly used in probabilistic graphical models. `DiffCollage` is scalable; it allows different diffusion models trained only with samples from marginal distributions instead of joint data distribution, which are easier to obtain. `DiffCollage` is efficient; diffusion models for different marginals can be trained and sampled in parallel. Through `DiffCollage`, we enable large content generation with diffusion models.

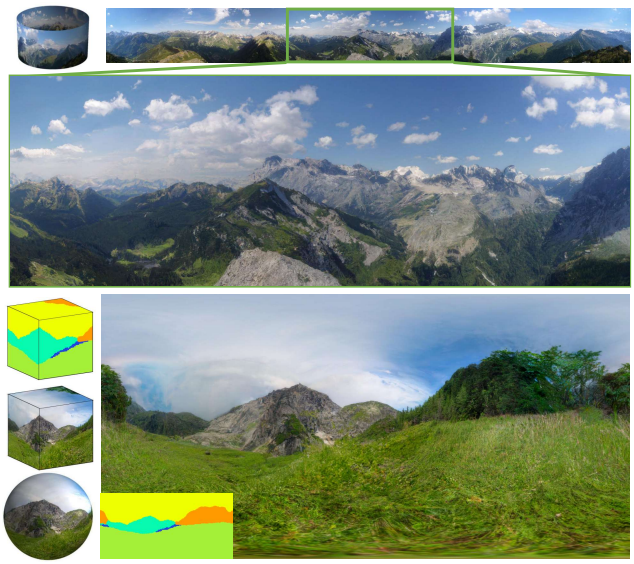

Figure 9. Top: a 1024×10240 horizontal panorama image. Bottom left: spherical/cube map representations of an input segmentation map and the output 360-degree panorama image. Each face of the cube is of size 1024×1024. Bottom right: equirectangular representation of the input segmentation and the output image.

References

[1] Brian D.O. Anderson. Reverse-time diffusion equation models. *Stochastic Process. Appl.*, 12(3):313–326, May 1982. 3

[2] Ivan Anokhin, Kirill Demochkin, Taras Khakhulin, Gleb Sterkin, Victor Lempitsky, and Denis Korzhenkov. Image generators with conditionally-independent pixel synthesis. In *CVPR*, 2021. 2

[3] Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, Tero Karras, and Ming-Yu Liu. ediffi: Text-to-image diffusion models with ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*, 2022. 1, 2, 5, 6, 15

[4] 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. *NeurIPS*, 2020. 1

[5] Lucy Chai, Michael Gharbi, Eli Shechtman, Phillip Isola, and Richard Zhang. Any-resolution training for high-resolution image synthesis. In *ECCV*, 2022. 2

[6] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *CVPR*, 2022. 3

[7] Yu-Jie Chen, Shin-I Cheng, Wei-Chen Chiu, Hung-Yu Tseng, and Hsin-Ying Lee. Vector quantized image-to-image translation. In *ECCV*, 2022. 3

[8] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. Ilvr: Conditioning method for denoising diffusion probabilistic models. *arXiv preprint arXiv:2108.02938*, 2021. 2, 6, 13, 14

[9] Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye. Improving diffusion models for inverse problems using manifold constraints. *arXiv preprint arXiv:2206.00941*, 2022. 3, 6, 7, 14

[10] Hyungjin Chung, Byeongsu Sim, and Jong Chul Ye. Come-Closer-Diffuse-Faster: Accelerating conditional diffusion models for inverse problems through stochastic contraction. *arXiv preprint arXiv:2112.05146*, 2021. 3, 7

[11] Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. DiffEdit: Diffusion-based semantic image editing with mask guidance. *arXiv preprint arXiv:2210.11427*, 2022. 2

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 15

[13] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021. 15

[14] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 2021. 1, 3, 6

[15] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022. 2

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 2

[17] Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating diverse and natural 3d human motions from text. In *CVPR*, 2022. 7, 8

[18] 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. 2

[19] 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 *NeurIPS*, 2017. 6

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

[21] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 2

[22] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *arXiv preprint arXiv:2204.03458*, 2022. 3, 6, 7

[23] Xun Huang, Arun Mallya, Ting-Chun Wang, and Ming-Yu Liu. Multimodal conditional image synthesis with product-of-experts GANs. In *ECCV*, 2022. 8

[24] Bowen Jing, Gabriele Corso, Renato Berlinghieri, and Tommi Jaakkola. Subspace diffusion generative models. In *ECCV*, 2022. 2

[25] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *NeurIPS*, 2022. 3, 15, 16

[26] Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. *arXiv preprint arXiv:2201.11793*, 2022. 2, 3, 6, 7, 13, 14

[27] Bahjat Kawar, Gregory Vaksman, and Michael Elad. Snips: Solving noisy inverse problems stochastically. In *NeurIPS*, 2021. 2

[28] 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. *arXiv preprint arXiv:2210.09276*, 2022. 2

[29] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009. 3, 4, 12

[30] Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *arXiv preprint arXiv:2205.14217*, 2022. 2

[31] Chieh Hubert Lin, Chia-Che Chang, Yu-Sheng Chen, Da-Cheng Juan, Wei Wei, and Hwann-Tzong Chen. COCO-GAN: Generation by parts via conditional coordinating. In *ICCV*, 2019. 2, 12

[32] Chieh Hubert Lin, Hsin-Ying Lee, Yen-Chi Cheng, Sergey Tulyakov, and Ming-Hsuan Yang. InfinityGAN: Towards infinite-pixel image synthesis. *arXiv preprint arXiv:2104.03963*, 2021. 2

[33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 15

[34] 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. 5

[35] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *CVPR*, 2022. 2, 3, 6, 7

[36] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. Amass: Archive of motion capture as surface shapes. In *ICCV*, 2019. 7

[37] Chenlin Meng, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Image synthesis and editing with stochastic differential equations. In *ICLR*, 2022. 2, 7

[38] 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. 2021. 5, 15

[39] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. Diffusion models for adversarial purification. In *ICML*, 2022. 2

[40] Evangelos Ntavelis, Mohamad Shahbazi, Iason Kastanis, Radu Timofte, Martin Danelljan, and Luc Van Gool. Arbitrary-scale image synthesis. In *CVPR*, 2022. 2

[41] Xingang Pan, Xiaohang Zhan, Bo Dai, Dahua Lin, Chen Change Loy, and Ping Luo. Exploiting deep generative prior for versatile image restoration and manipulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 14

[42] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019. 8

[43] 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 *ICML*, 2021. 1, 5

[44] 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. 1, 2, 5

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

[46] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. *arXiv preprint arxiv:2208.12242*, 2022. 2

[47] Dohoon Ryu and Jong Chul Ye. Pyramidal denoising diffusion probabilistic models. *arXiv preprint arXiv:2208.01864*, 2022. 6

[48] Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In *SIGGRAPH*, 2022. 1, 3, 6

[49] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022. 1, 2, 15

[50] Simo Särkkä and Arno Solin. *Applied stochastic differential equations*, volume 10. Cambridge University Press, 2019. 3

[51] Rahul Singh, Isabel Haasler, Qinsheng Zhang, Johan Karlsson, and Yongxin Chen. Inference with aggregate data: An optimal transport approach. *arXiv preprint arXiv:2003.13933*, 2020. 4

[52] Ivan Skorokhodov, Savva Ignatyev, and Mohamed Elhoseiny. Adversarial generation of continuous images. In *CVPR*, 2021. 2

[53] Ivan Skorokhodov, Grigorii Sotnikov, and Mohamed Elhoseiny. Aligning latent and image spaces to connect the unconnectable. In *ICCV*, 2021. 2, 5, 6, 12, 15

[54] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015. 2

[55] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021. 2, 5, 16

[56] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-Based Generative Modeling through Stochastic Differential Equations. In *ICLR*, 2021. 1, 2, 3

[57] Łukasz Struski, Szymon Knop, Przemysław Spurek, Wiktor Daniec, and Jacek Tabor. LocoGAN—locally convolutional GAN. *CVIU*, 2022. 2

[58] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. In *NeurIPS*, 2021. 2

[59] 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. 7, 8, 16

[60] Dani Valevski, Matan Kalman, Yossi Matias, and Yaniv Leviathan. UniTune: Text-driven image editing by fine tuning an image generation model on a single image. *arXiv preprint arXiv:2210.09477*, 2022. 2

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

[62] Tom White. Sampling generative networks. *arXiv preprint arXiv:1609.04468*, 2016. 16

[63] Chenfei Wu, Jian Liang, Xiaowei Hu, Zhe Gan, Jianfeng Wang, Lijuan Wang, Zicheng Liu, Yuejian Fang, and Nan Duan. Nuwa-infinity: Autoregressive over autoregressive generation for infinite visual synthesis. In *NeurIPS*, 2022. 3

[64] Mao Ye, Lemeng Wu, and Qiang Liu. First hitting diffusion models. *arXiv preprint arXiv:2209.01170*, 2022. 2

[65] Jonathan S Yedidia, William T Freeman, and Yair Weiss. Generalized belief propagation. In *NeurIPS*, pages 689–695, 2001. 4

[66] Jonathan S Yedidia, William T Freeman, and Yair Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on information theory*, 2005. 4

[67] 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. 6, 15

[68] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. In *NeurIPS*, 2022. 2

[69] Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. *arXiv preprint arXiv:2204.13902*, 2022. 2, 5

[70] Qinsheng Zhang, Molei Tao, and Yongxin Chen. gddim: Generalized denoising diffusion implicit models. *arXiv preprint arXiv:2206.05564*, 2022. 2, 3, 5

[71] Zhu Zhang, Jianxin Ma, Chang Zhou, Rui Men, Zhikang Li, Ming Ding, Jie Tang, Jingren Zhou, and Hongxia Yang. M6-ufc: Unifying multi-modal controls for conditional image synthesis. *arXiv preprint arXiv:2105.14211*, 2021. 3

[72] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *ICCV*, 2021. 2