Video Probabilistic Diffusion Models in Projected Latent Space

Sihyun Yu[1]   Kihyuk Sohn[2]   Subin Kim[1]   Jinwoo Shin[1]

[1]KAIST   [2]Google Research

{sihyun.yu, subin-kim, jinwoos}@kaist.ac.kr, kihyuks@google.com

Abstract

Despite the remarkable progress in deep generative models, synthesizing high-resolution and temporally coherent videos still remains a challenge due to their high-dimensionality and complex temporal dynamics along with large spatial variations. Recent works on diffusion models have shown their potential to solve this challenge, yet they suffer from severe computation- and memory-inefficiency that limit the scalability. To handle this issue, we propose a novel generative model for videos, coined projected latent video diffusion model (PVDM), a probabilistic diffusion model which learns a video distribution in a low-dimensional latent space and thus can be efficiently trained with high-resolution videos under limited resources. Specifically, PVDM is composed of two components: (a) an autoencoder that projects a given video as 2D-shaped latent vectors that factorize the complex cubic structure of video pixels and (b) a diffusion model architecture specialized for our new factorized latent space and the training/sampling procedure to synthesize videos of arbitrary length with a single model. Experiments on popular video generation datasets demonstrate the superiority of PVDM compared with previous video synthesis methods; e.g., PVDM obtains the FVD score of 639.7 on the UCF-101 long video (128 frames) generation benchmark, which improves 1773.4 of the prior state-of-the-art.

1. Introduction

Recent progresses of deep generative models have shown their promise to synthesize high-quality, realistic samples in various domains, such as images [9, 27, 41], audio [8, 31, 32], 3D scenes [6, 38, 48], natural languages [2, 5], *etc*. As a next step forward, several works have been actively focusing on the more challenging task of video synthesis [12, 18, 21, 47, 55, 67]. In contrast to the success in other domains, the generation quality is yet far from real-world videos, due to the high-dimensionality and complexity of videos that contain complicated spatiotemporal dynamics in high-resolution frames.

Inspired by the success of diffusion models in handling complex and large-scale image datasets [9, 40], recent approaches have attempted to design diffusion models for videos [16, 18, 21, 22, 35, 66]. Similar to image domains, these methods have shown great potential to model video distribution much better with scalability (both in terms of spatial resolution and temporal durations), even achieving photorealistic generation results [18]. However, they suffer from severe computation and memory inefficiency, as diffusion models require lots of iterative processes in input space to synthesize samples [51]. Such bottlenecks are much more amplified in video due to a cubic RGB array structure.

Meanwhile, recent works in image generation have proposed latent diffusion models to circumvent the computation and memory inefficiency of diffusion models [15, 41, 59]. Instead of training the model in raw pixels, latent diffusion models first train an autoencoder to learn a low-dimensional latent space succinctly parameterizing images [10, 41, 60] and then models this latent distribution. Intriguingly, the approach has shown a dramatic improvement in efficiency for synthesizing samples while even achieving state-of-the-art generation results [41]. Despite their appealing potential, however, developing a form of latent diffusion model for videos is yet overlooked.

Contribution. We present a novel latent diffusion model for videos, coined *projected latent video diffusion model* (PVDM). Specifically, it is a two-stage framework (see Figure 1 for the overall illustration):

- *Autoencoder:* We introduce an autoencoder that represents a video with *three 2D image-like latent vectors* by factorizing the complex cubic array structure of videos. Specifically, we propose 3D → 2D projections of videos at each spatiotemporal direction to encode 3D video pixels as three succinct 2D latent vectors. At a high level, we design one latent vector across the temporal direction to parameterize the common contents of the video (*e.g.*, background), and the latter two vectors to encode the motion of a video. These 2D latent vectors are beneficial for achieving high-quality and succinct encoding of videos, as well as enabling compute-efficient diffusion model architecture design due to their image-like structure.
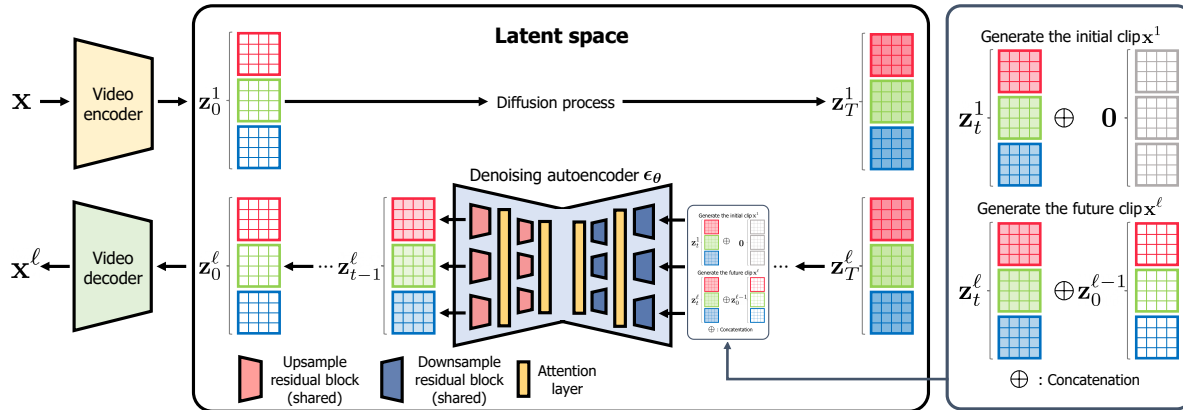
Figure 1. Overall illustration of our projected latent video diffusion model (PVDM) framework. PVDM is composed of two components: (a) (left) an autoencoder that maps a video into 2D image-like latent space (b) (right) a diffusion model operates in this latent space.

- *Diffusion model:* Based on the 2D image-like latent space built from our video autoencoder, we design a new diffusion model architecture to model the video distribution. Since we parameterize videos as image-like latent representations, we avoid computation-heavy 3D convolutional neural network architectures that are conventionally used for handling videos. Instead, our architecture is based on 2D convolution network diffusion model architecture that has shown its strength in handling images. Moreover, we present a joint training of unconditional and frame conditional generative modeling to generate a long video of arbitrary lengths.

We verify the effectiveness of our method on two popular datasets for evaluating video generation methods: UCF-101 [54] and SkyTimelapse [64]. Measured with Inception score (IS; higher is better [44]) on UCF-101, a representative metric of evaluating unconditional video generation, PVDM achieves the state-of-the-art result of 74.40 on UCF-101 in generating 16 frames, 256×256 resolution videos. In terms of Fréchet video distance (FVD; lower is better [58]) on UCF-101 in synthesizing long videos (128 frames) of 256×256 resolution, it significantly improves the score from 1773.4 of the prior state-of-the-art to 639.7. Moreover, compared with recent video diffusion models, our model shows a strong memory and computation efficiency. For instance, on a single NVIDIA 3090Ti 24GB GPU, a video diffusion model [21] requires almost full memory (≈24GB) to train at 128×128 resolution with a batch size of 1. On the other hand, PVDM can be trained with a batch size of 7 at most per this GPU with 16 frames videos at 256×256 resolution.

To our knowledge, the proposed PVDM is the first latent diffusion model designed for video synthesis. We believe our work would facilitate video generation research towards efficient real-time, high-resolution, and long video synthesis under the limited computational resource constraints.

2. Related work

Video generation. Video generation is one of the long-standing goals in deep generative models. Many prior works have attempted to solve the problem and they mostly fall into three categories. First, there exists numerous attempts to extend image generative adversarial networks (GANs) [13] to generate videos [1,7,11,14,23,36,42,43,46,47,55,57,62,67, 68]; however, GANs often suffer from mode collapse problem and these methods are difficult to be scaled to complex, large-scale video datasets. Other approaches have proposed learning the distribution via training autoregressive models [12,24,39,63,65] using Transformers [61]. They have shown better mode coverage and video quality than GAN-based approaches, but they require expensive computation and memory costs to generate longer videos [47]. Finally, recent works have attempted to build diffusion models [19] for videos [16,21,22,46,66], achieving state-of-the-art results, yet they also suffer from significant computation and memory inefficiency. Our method also takes an approach to diffusion models for videos; however, we consider the generative modeling in low-dimensional latent space to alleviate these bottlenecks of diffusion models.

Diffusion models. Diffusion models [19,49], which are categorized as score-based generative models [52,53], model the data distribution by learning a gradual iterative denoising process from the Gaussian distribution to the data distribution. Intriguingly, they show a strong promise in generating high-quality samples with wide mode coverage, even outperforming GANs in image synthesis [9] and enabling zero-shot text-to-image synthesis [40]. Not limited to images, diffusion models have shown their promise in other data domains, including point clouds [34], audio [31], etc. However, diffusion models suffer from severe computation inefficiency for data sampling due to the iterative denoising process in high-dimensional data space. To alleviate this problem, sev-
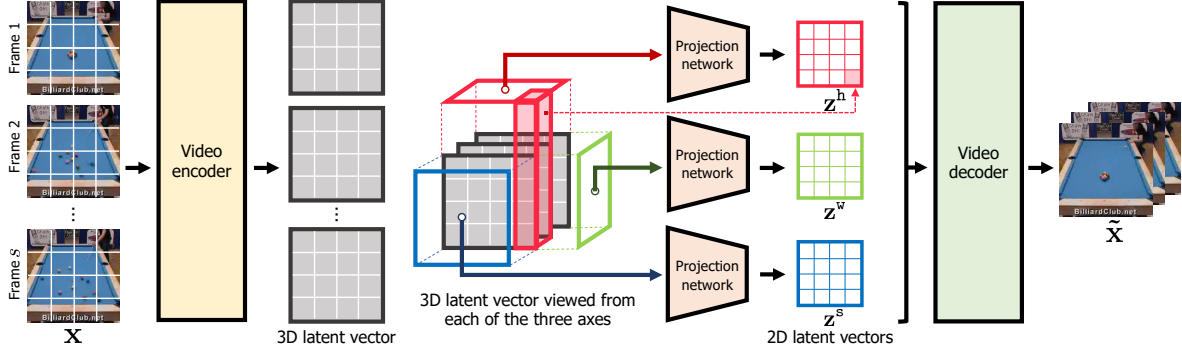
Figure 2. Detailed illustration of our autoencoder architecture in PVDM framework ((a) in Figure 1).

eral works have proposed an effective sampling process from trained diffusion models [51, 69] or learning the data distribution from low-dimensional latent space that amortizes the data [15, 41, 59]. We take the latter approach for designing a computation-efficient video diffusion model.

Diffusion models for videos. Following the remarkable success of diffusion models in image domains, several works [16, 18, 21, 22, 35, 66] have extended them for video generation. Intriguingly, they often show much better results than prior works, even can be scaled-up to complex datasets, and achieves successful generation results on challenging text-to-video synthesis task [18]. Despite their potential on modeling videos, scaling them for synthesizing high-resolution, long video is not straightforward due to a huge computation bottleneck caused by an unfavorable dimension increase of video data as 3D RGB arrays. We tackle this problem by modeling the video distribution in the low-dimensional latent space.

Triplane representations. Several recent works in 3D-aware generation [6, 48] have demonstrated that high-dimensional 3D voxels can be effectively parameterized with 2D triplane latent representations without sacrificing the encoding quality. In particular, prior works have proposed GAN architectures that synthesize a 3D scene by generating three image-like latents as an approximation of complex 3D voxels. In contrast, we explore the effectiveness of triplane representations for encoding videos, and we build diffusion models instead of GANs with such representations.

3. Projected latent video diffusion model

We first formalize the problem of *video generative modeling*. Consider a dataset $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^N$ of size N, where each $\mathbf{x} \in \mathcal{D}$ is sampled from the unknown data distribution $p_{\text{data}}(\mathbf{x})$. Here, each $\mathbf{x} \in \mathbb{R}^{3 \times S \times H \times W}$ is a *video clip* consisting of S frames at $H \times W$ (spatial) resolution. In video generative modeling, the goal is to learn a model distribution $p_{\text{model}}(\mathbf{x})$ that matches $p_{\text{data}}(\mathbf{x})$ using $\mathcal{D}$.

To accomplish the goal, we build a method based on diffusion models—a type of generative model that models the data distribution $p_{\text{data}}(\mathbf{x})$ by learning the reverse process of

the Markov diffusion process starting from $p_{\text{data}}(\mathbf{x})$ to the Gaussian prior distribution $\mathcal{N}(\mathbf{0_x}, \mathbf{I_x})$. Remarkably, diffusion models are shown to synthesize high-quality samples without mode collapse problems and can be scaled to model complex datasets. However, diffusion models directly operate in raw input space; unlike other data domains (*e.g.*, images), designing diffusion models for videos is challenging due to their cubic complexity and high-dimensionality as 3D tensors of RGB values. Our key contribution is to mitigate this issue by proposing diffusion models operating on a novel low-dimensional latent space that succinctly parameterizes videos by breaking down the complex 3D structure of video pixels into three 2D structures.

In the rest of this section, we describe our *projected latent video diffusion model* (PVDM) in detail. In Section 3.1, we provide an overview of (latent) diffusion models. In Section 3.2, we explain how we design PVDM in detail. Finally, in Section 3.3, we describe our training objective and sampling strategy to generate longer videos.

3.1. Latent diffusion models

At a high level, diffusion models learn the target distribution $p_{\text{data}}(\mathbf{x})$ by learning a gradual denoising process from Gaussian prior distribution to reach $p_{\text{data}}(\mathbf{x})$. Formally, diffusion models consider the reverse process $p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)$ of the Markov diffusion process $q(\mathbf{x}_t|\mathbf{x}_{t-1})$ of a fixed length $T > 0$ starting from $p(\mathbf{x}_0) := p_{\text{data}}(\mathbf{x})$ to $p(\mathbf{x}_T) := \mathcal{N}(\mathbf{0_x}, \mathbf{I_x})$. More specifically, $q(\mathbf{x}_t|\mathbf{x}_{t-1})$ is formalized as the following normal distribution with a pre-defined $0 < \beta_1, \ldots, \beta_T < 1$ and $\bar{\alpha}_t := \prod_{i=1}^t (1 - \beta_i)$:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1 - \beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I_x}),$$
$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1 - \bar{\alpha}_t)\mathbf{I_x}).$$

Using the reparameterization trick [30], Ho et al. [19] shows the corresponding $p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)$ can be learned as a denoising autoencoder $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ that denoises a noisy sample $\mathbf{x}_t$, which is trained with the following noise-prediction objective:

$$\mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}, t}\left[||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)||_2^2\right] \text{ where } \mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon},$$

Algorithm 1 projected latent video diffusion model (PVDM)

1: **for** $\ell = 1$ to L **do** $\triangleright$ *Iteratively generate video clips* $\mathbf{x}^\ell$.
2:      Sample the random noise $\mathbf{z}^\ell_T \sim p(\mathbf{z}_T)$.
3:      **for** $t = T$ to 1 **do**
4:         **if** $\ell = 1$ **then**
5:            Unconditional score $\boldsymbol{\epsilon}_t = \boldsymbol{\epsilon_\theta}(\mathbf{z}^\ell_t, \mathbf{0}, t)$.
6:         **else**
7:            Conditional score $\boldsymbol{\epsilon}_t = \boldsymbol{\epsilon_\theta}(\mathbf{z}^\ell_t, \mathbf{z}^{\ell-1}_0, t)$.
8:         **end if**
9:         Sample $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0_z}, \mathbf{I_z})$.
10:        Compute $\mathbf{z}^\ell_{t-1} = \frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{z}^\ell_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right) + \sigma_t\boldsymbol{\epsilon}$.
11:      **end for**
12:      Decode the ℓ-th clip $\mathbf{x}^\ell = g_{\boldsymbol{\psi}}(\mathbf{z}^\ell_0)$.
13: **end for**
14: Output the generated video $[\mathbf{x}^1, \dots, \mathbf{x}^L]$.

---

and $p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)$ can be approximately formulated as the following normal distribution with a small enough β_t [50]:

$$p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t) \coloneqq \mathcal{N}\left(\mathbf{x}_{t-1}; \mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon_\theta}(\mathbf{x}_t, t), \sigma_t^2\right),$$

with the variances $\sigma_t^2 \coloneqq \beta_t$ as pre-defined hyperparameters.

The main drawback of diffusion models is severe computation and memory inefficiency. To generate the sample, one should operate $p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)$ in high-dimensional input space $\mathcal{X}$ repeatedly (*e.g.*, $T = 1000$ in Ho et al. [19]). To tackle this issue, several works [15, 41, 59] have proposed latent diffusion models to learn the distribution in low-dimensional latent space $\mathcal{Z}$ that succinctly encodes the data, which is typically learned with autoencoders [41]. Specifically, latent diffusion models train the denoising autoencoder $\boldsymbol{\epsilon_\theta}(\mathbf{z}_t, t)$ in $\mathcal{Z}$ instead of $\mathcal{X}$ *i.e.*, learning $p_{\boldsymbol{\theta}}(\mathbf{z}_{t-1}|\mathbf{z}_t)$, so that $\mathbf{x}$ is generated by first sampling $\mathbf{z}$ and then decoding to $\mathbf{x}$ with the decoder. Due to the significant dimension reduction from $\mathbf{x}$ to $\mathbf{z}$, one can dramatically reduce the computation for sampling the data. Inspired by their success, we also model video distribution with latent diffusion models.

3.2. Designing efficient latent video diffusion model

Autoencoder. To represent a video $\mathbf{x}$ as a low-dimensional latent vector $\mathbf{z}$, we train an autoencoder composed of an encoder $f_{\boldsymbol{\phi}} : \mathcal{X} \rightarrow \mathcal{Z}$ with $f_{\boldsymbol{\phi}}(\mathbf{x}) = \mathbf{z}$ and a decoder $g_{\boldsymbol{\psi}} : \mathcal{Z} \rightarrow \mathcal{X}$ with $g_{\boldsymbol{\psi}}(\mathbf{z}) = \tilde{\mathbf{x}}$ so that $\tilde{\mathbf{x}}$ becomes $\mathbf{x}$. Motivated by VQGAN [10] for compressing images perceptually via latent codes [37], we encode videos with discrete latent codes by minimizing the sum of two terms: pixel-level reconstruction loss and the negative of perceptual similarity (LPIPS; [70]); we provide more details in Appendix A.

Conventional video autoencoders have mostly relied on frame-wise 2D convolutional networks [45] or 3D convolutional networks [65] to compress a given video $\mathbf{x}$. While

these approaches are fairly effective in amortizing $\mathbf{x}$ as a low-dimensional latent vector, they encode $\mathbf{x}$ as a 3D latent vector, which overlooks the temporal coherency and requiring the diffusion model architecture to deal with a 3D tensor and may cause a computation overhead. Instead, we take a different approach: given $\mathbf{x}$, we construct $\mathbf{z}$ as three 2D latent vectors $\mathbf{z}^{\mathtt{s}}, \mathbf{z}^{\mathtt{h}}, \mathbf{z}^{\mathtt{w}}$, *i.e.*, $\mathbf{z} \coloneqq [\mathbf{z}^{\mathtt{s}}, \mathbf{z}^{\mathtt{h}}, \mathbf{z}^{\mathtt{w}}]$, $\mathbf{z}^{\mathtt{s}} \in \mathbb{R}^{C \times H' \times W'}$, $\mathbf{z}^{\mathtt{h}} \in \mathbb{R}^{C \times S \times W'}$, $\mathbf{z}^{\mathtt{w}} \in \mathbb{R}^{C \times S \times H'}$, where C is a latent dimension and $H' = H/d$, $W' = W/d$ for $d > 1$. Here, we denote each $\mathbf{z}^{\mathtt{s}}, \mathbf{z}^{\mathtt{h}}, \mathbf{z}^{\mathtt{w}}$ the concatenation of latent codes $z^{\mathtt{s}}_{hw}, z^{\mathtt{h}}_{sw}, z^{\mathtt{w}}_{sh} \in \mathbb{R}^C$ (respectively); *e.g.*, for $\mathbf{z}^{\mathtt{s}}$, we write

$$\mathbf{z}^{\mathtt{s}} \coloneqq [z^{\mathtt{s}}_{hw}] \quad \text{for } 1 \leq h \leq H', 1 \leq w \leq W'.$$

We design $\mathbf{z}^{\mathtt{s}}$ to capture the common content across time in $\mathbf{x}$ (*e.g.*, background), and the latter two latent vectors $\mathbf{z}^{\mathtt{h}}, \mathbf{z}^{\mathtt{w}}$ to encode the underlying motion in $\mathbf{x}$ by learning the representations across two spatial axes of videos. Specifically, $[\mathbf{z}^{\mathtt{s}}, \mathbf{z}^{\mathtt{h}}, \mathbf{z}^{\mathtt{w}}]$ is computed with the encoder $f_{\boldsymbol{\phi}}$, where $f_{\boldsymbol{\phi}}$ is a composition of a video-to-3D-latent mapping $f^{\mathtt{shw}}_{\boldsymbol{\phi}_{\mathtt{shw}}}$ and a 3D-to-2D-latents projection $f^{\mathtt{s}}_{\boldsymbol{\phi}_{\mathtt{s}}} \times f^{\mathtt{h}}_{\boldsymbol{\phi}_{\mathtt{h}}} \times f^{\mathtt{w}}_{\boldsymbol{\phi}_{\mathtt{w}}}$ (with $\boldsymbol{\phi} \coloneqq (\boldsymbol{\phi}_{\mathtt{shw}}, \boldsymbol{\phi}_{\mathtt{s}}, \boldsymbol{\phi}_{\mathtt{h}}, \boldsymbol{\phi}_{\mathtt{w}})$; see Figure 2 for the illustration). More specifically, we compute $\mathbf{z}$ from $\mathbf{x}$ as follows:

$$\mathbf{u} \coloneqq f^{\mathtt{shw}}_{\boldsymbol{\phi}_{\mathtt{shw}}}(\mathbf{x}), \quad \text{where } \mathbf{u} = [u_{shw}] \in \mathbb{R}^{C \times S \times H' \times W'},$$
$$z^{\mathtt{s}}_{hw} \coloneqq f^{\mathtt{s}}_{\boldsymbol{\phi}_{\mathtt{s}}}(u_{1hw}, \dots, u_{Shw}), \ 1 \leq h \leq H', \ 1 \leq w \leq W',$$
$$z^{\mathtt{h}}_{sw} \coloneqq f^{\mathtt{h}}_{\boldsymbol{\phi}_{\mathtt{h}}}(u_{s1w}, \dots, u_{sH'w}), \ 1 \leq s \leq S, \ 1 \leq w \leq W',$$
$$z^{\mathtt{w}}_{sh} \coloneqq f^{\mathtt{w}}_{\boldsymbol{\phi}_{\mathtt{w}}}(u_{sh1}, \dots, u_{shW'}), \ 1 \leq s \leq S, \ 1 \leq h \leq H'.$$

With the latent vector $\mathbf{z}$ of $\mathbf{x}$ from $f_{\boldsymbol{\phi}}$, we construct the decoder $g_{\boldsymbol{\psi}}$ that (a) computes a 3D latent grid $\mathbf{v}$ from $\mathbf{z}$ and (b) reconstructs $\mathbf{x}$ from $\mathbf{v}$, where $\mathbf{v}$ is computed as follows:

$$\mathbf{v} = (v_{shw}) \in \mathbb{R}^{3C \times S \times H' \times W'}, \ v_{shw} \coloneqq [z_{hw}, z_{sw}, z_{sh}].$$

We use video Transformer (*e.g.*, TimeSformer [4]) for $f^{\mathtt{shw}}_{\boldsymbol{\phi}_{\mathtt{shw}}}(\mathbf{x})$ and $g_{\boldsymbol{\psi}}$, and a small Transformer [61] for projections $f^{\mathtt{s}}_{\boldsymbol{\phi}_{\mathtt{s}}}, f^{\mathtt{h}}_{\boldsymbol{\phi}_{\mathtt{h}}}, f^{\mathtt{w}}_{\boldsymbol{\phi}_{\mathtt{w}}}$. While our autoencoder design requires slightly more parameters and computations for encoding videos from additional projections, it provides dramatic computation efficiency for training (and sampling) of a diffusion model in latent space. In particular, conventional video autoencoder design requires $O(SHW)$ dimensions of latent codes for encoding videos, and thus diffusion models to utilize 3D convolution layers and self-attention layers, which lead the computation overhead to be $O(SHW)$ and $O((SHW)^2)$, respectively. In contrast, we represent a video as image-like latent vectors with $O(HW + SW + SH)$ latent codes; such latent representations enables more compute-efficient diffusion model design by utilizing 2D convolution layers ($O(HW + SW + SH)$) and self-attention layers ($O((HW + SW + SH)^2)$) based on modifying popular architectures used for image diffusion models [19].
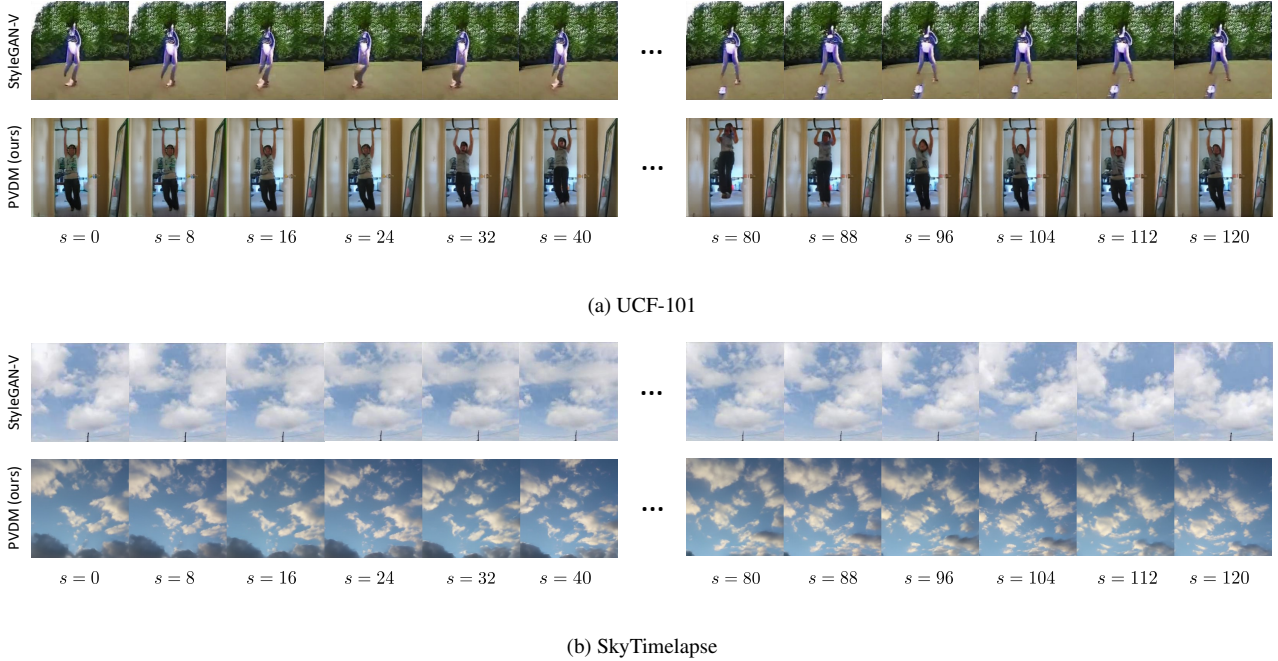
(a) UCF-101

(b) SkyTimelapse

Figure 3. 256×256 resolution, 128 frame video synthesis results of StyleGAN-V and PVDM, trained on (a) UCF-101 and (b) SkyTimelapse.[1]

The intuition behind our overall autoencoder design is that videos are temporally coherent signals and share the common contents across the temporal axis; capturing the common content as $\mathbf{z}^s$ can dramatically reduce the number of parameters for encoding videos [29]. Moreover, due to the high temporal coherency of videos, the temporal variation is often not very large; we empirically verify our succinct representation of motions as two spatial grids $\mathbf{z}^h, \mathbf{z}^w$ does not hurt the encoding quality (see Section 4.3).

Latent diffusion model. Recall that we encode the video $\mathbf{x}$ as three 2D latent vectors $\mathbf{z} = [\mathbf{z}^s, \mathbf{z}^h, \mathbf{z}^w]$; it is enough to model the distribution $p(\mathbf{z}^s, \mathbf{z}^h, \mathbf{z}^w)$ for learning $p_{\text{data}}(\mathbf{x})$. To train a denoising autoencoder for $[\mathbf{z}^s, \mathbf{z}^h, \mathbf{z}^w]$, we design the neural network architecture based on utilizing popular 2D convolutional U-Net architecture used for training diffusion models for image generation [9] instead of 3D convolutional networks [21]. Specifically, we use a single U-Net (*i.e.*, shared parameters) to denoise each $\mathbf{z}^s, \mathbf{z}^h, \mathbf{z}^w$. To handle the dependency among $\mathbf{z}^s, \mathbf{z}^h, \mathbf{z}^w$ to model the joint distribution $p(\mathbf{z}^s, \mathbf{z}^h, \mathbf{z}^w)$, we add attention layers that operates to the intermediate features of $\mathbf{z}^s, \mathbf{z}^h, \mathbf{z}^w$ from shared U-Net. We remark that such 2D convolutional architecture design is more computation-efficient than a naïve 3D convolutional U-Nets for videos, which is possible due to the "image-like" structure and a reduced dimension of our latent vectors from using less latent codes for encoding videos.

3.3. Generating longer videos with PVDM

Note that videos are sequential data; unlike our setup that assumes all videos $\mathbf{x} \in \mathcal{D}$ have the same length S, the length of real-world videos varies, and generative video models should be able to generate videos of arbitrary length. However, we only learn the distribution of the fixed-length video clips $p_{\text{data}}(\mathbf{x})$; to enable long video generation, one can consider learning a conditional distribution $p(\mathbf{x}^2|\mathbf{x}^1)$ of two consecutive video clips $[\mathbf{x}^1, \mathbf{x}^2]$ of $\mathbf{x}^1, \mathbf{x}^2 \in \mathbb{R}^{3 \times S \times H \times W}$ and sequentially generate future clips given the current ones.

A straightforward solution is to have two separate models to learn the unconditional distribution $p_{\text{data}}(\mathbf{x})$ and the conditional distribution $p(\mathbf{x}^2|\mathbf{x}^1)$. Instead of having an extra model, we propose to train a *single* diffusion model to jointly learn an unconditional distribution $p(\mathbf{x})$ and the conditional distribution $p(\mathbf{x}^2|\mathbf{x}^1)$. It can be achieved by training a conditional diffusion model $p(\mathbf{x}^2|\mathbf{x}^1)$ with introducing a null frame (*i.e.*, $\mathbf{x}^1 = \mathbf{0}$) for a joint learning of $p(\mathbf{x})$ [20]. More specifically, we consider training of a denoising autoencoder $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t^2, \mathbf{z}_0^1, t)$ in the latent space with the following objective:

$$\mathbb{E}_{(\mathbf{x}_0^1, \mathbf{x}_0^2), \boldsymbol{\epsilon}, t} \Big[\lambda ||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t^2, \mathbf{z}_0^1, t)||_2^2$$
$$+ (1 - \lambda)||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t^2, \mathbf{0}, t)||_2^2 \Big],$$

where $\mathbf{z}_0^1 = f_{\boldsymbol{\phi}}(\mathbf{x}_0^1)$, $\mathbf{z}_0^2 = f_{\boldsymbol{\phi}}(\mathbf{x}_0^2)$, $\mathbf{z}_t^2 = \sqrt{\bar{\alpha}_t}\mathbf{z}_0^2 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}$, and $\lambda \in (0, 1)$ is a hyperparameter that balances a learning between unconditional and conditional distribution.

---

[1]StyleGAN-V results are from https://universome.github.io/stylegan-v.
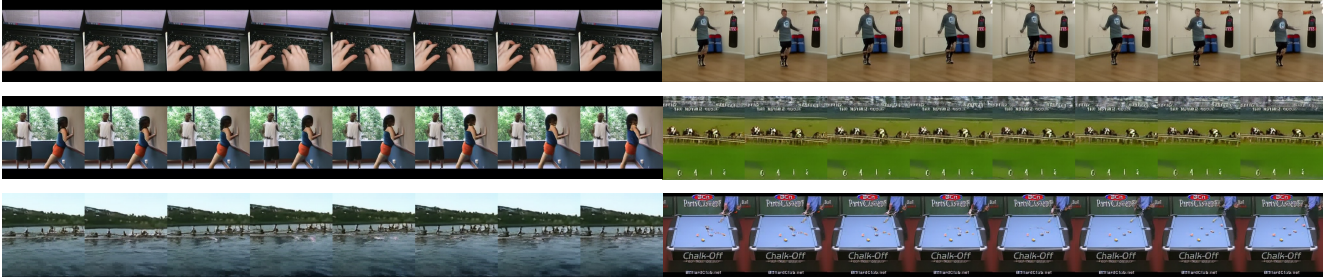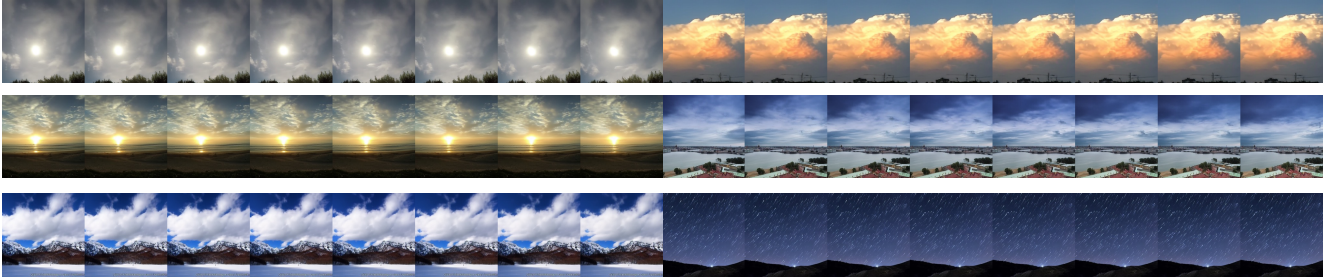
(a) UCF-101

(b) SkyTimelapse

Figure 4. Illustrations of random 16 frames, 256×256 resolution video synthesis results of PVDM trained on UCF-101 and SkyTimelapse datasets. We visualize the frames of each video with stride 2.

After training, one can generate the long video as follows: sample an initial video clip $\mathbf{x}^1 \sim p_{\boldsymbol{\theta}}(\mathbf{x})$, and repeat generating next clip $\mathbf{x}^{\ell+1} \sim p_{\boldsymbol{\theta}}(\mathbf{x}^{\ell+1}|\mathbf{x}^\ell)$ conditioned on the previous clip. Finally, we obtain a long video by concatenating all generated clips $[\mathbf{x}^1, \ldots, \mathbf{x}^L]$ of arbitrary length $L > 1$. See Algorithm 1 for details and Figure 3 for results.

4. Experiments

We validate the superiority of our PVDM framework under two representative datasets: UCF-101 [54] and Sky-Timelapse [64]. In Section 4.1, we provide the experimental setup and evaluation details that we used for experiments. In Section 4.2, we present the main qualitative and quantitative video synthesis results. Finally, in Section 4.3, we perform an ablation studies to verify the effect of components and efficiencies of our method.

4.1. Experiment setup

Datasets. We train our PVDM and compare with baselines on well-known video datasets used for video synthesis: UCF-101 [54] and SkyTimelapse [64]. Following the experimental setup used in recent video generation methods [47, 67], we preprocess these datasets as video clips of length 16 or 128 frames, where each frame is resized to 256×256 resolution. For training each model, we use the train split of the dataset for all of the experiments. We provide the detailed description of datasets in Appendix B.1.

Evaluation. For evaluation metrics for quantitative comparison, we use and report Inception score (IS) [44] and Fréchet video distance (FVD) [58]. We use the clip length of 16 for the evaluation of IS, following the prior experiment setups in unconditional video generation. For FVD to evaluate UCF-101 and SkyTimelapse, we used a *fixed protocol* proposed by StyleGAN-V that removes the potential bias from the existence of long videos in the dataset (we provide more details of metrics in Appendix B.2). We consider two different clip lengths (16 and 128) for FVD, where we denote FVD_{16} and FVD_{128} as the FVD score measured on video clips of lengths 16 and 128, respectively. We use 2,048 real/fake video clips for evaluating FVD_{16} and FVD_{128}, and generate 10,000 video clips for the evaluation of IS.

Baselines. Following the setup in StyleGAN-V, one of the state-of-the-art video generation methods, we mainly compare PVDM with the following recent video synthesis methods: VideoGPT [65], MoCoGAN [57], MoCoGAN-HD [55], DIGAN [67], and StyleGAN-V [47]. Moreover, we perform an additional comparison between PVDM and previous approaches on IS values on UCF-101: MoCoGAN, ProgressiveVGAN [1], LDVD-GAN [23], VideoGPT, TGANv2 [43], DVD-GAN [7], DIGAN, VDM [21], and TATS [12]. All reported values are collected from the recent prior works: StyleGAN-V, DIGAN and TATS, unless otherwise specified. In particular, we compare the memory and computation efficiency with VDM. We provide more details of each method and how they are implemented in Appendix C.

Table 1. FVD$_{16}$ and FVD$_{128}$ values (lower values are better) of video generation models on UCF-101 and SkyTimelapse. Bolds indicate the best results, and we mark our method as blue. We report FVD values of other baselines obtained by the reference (StyleGAN-V [47]). N/M-s denotes the model is evaluated with the DDIM sampler [51] with N steps (for the initial clip) and M steps (for future clips).

	UCF-101		SkyTimelapse	
Method	FVD$_{16}$ ↓	FVD$_{128}$ ↓	FVD$_{16}$ ↓	FVD$_{128}$ ↓
VideoGPT [65]	2880.6	N/A	222.7	N/A
MoCoGAN [57]	2886.8	3679.0	206.6	575.9
+ StyleGAN2 [28]	1821.4	2311.3	85.88	272.8
MoCoGAN-HD [55]	1729.6	2606.5	164.1	878.1
DIGAN [67]	1630.2	2293.7	83.11	196.7
StyleGAN-V [47]	1431.0	1773.4	79.52	197.0
PVDM-S (ours); 100/20-s	457.4	902.2	71.46	159.9
PVDM-L (ours); 200/200-s	398.9	**639.7**	61.70	137.2
PVDM-L (ours); 400/400-s	**343.6**	648.4	**55.41**	**125.2**

Table 2. IS values (higher values are better) of video generation models on UCF-101. Bolds indicate the best results and subscripts denote the standard deviations. * denotes the model is trained on train+test split, otherwise the method uses only the train split for training.

Method	IS ↑
MoCoGAN [57]	12.42$\pm$0.07
ProgressiveVGAN [1]	14.56$\pm$0.05
LDVD-GAN [23]	22.91$\pm$0.19
VideoGPT [65]	24.69$\pm$0.30
TGANv2 [43]	28.87$\pm$0.67
StyleGAN-V* [47]	23.94$\pm$0.73
DIGAN [67]	29.71$\pm$0.53
VDM* [21]	57.00$\pm$0.62
TATS [12]	57.63$\pm$0.24
PVDM-L (ours)	**74.40$\pm$1.25**

Training details. We use Adam [30] and AdamW optimizer [33] for training of an autoencoder and a diffusion model, respectively, where other training details mostly follow the setups in latent diffusion models for images [41]. For the autoencoder architecture, we use TimeSformer [4] to both encoder and decoder, where we use Transformer [61] architecture for 3D-to-2D projection mapping. We use two configurations for diffusion models, (denoted by PVDM-S and PVDM-L, respectively); please refer to Appendix D for more training details and model configurations.

4.2. Main results

Qualitative results. Figure 4 illustrates the video synthesis results from PVDM on UCF-101 and SkyTimelapse: our method shows realistic video generation results in both scenarios, namely, including the motion of the small object (Figure 4a) or the large transition of the over frames (Figure 4b). We also note that such training is achieved under high-fidelity (256×256 resolution) videos. Moreover, our method also has the capability to synthesize videos in the complex UCF-101 dataset with a plausible quality, as shown in Figure 4a, while other baselines often fail on such challenging dataset datasets [47, 67].[2]

Quantitative results. Table 1 and 2 summarize the quantitative comparison between our method and prior video generation methods: PVDM consistently outperforms other methods measured with diverse metrics. In particular, compared with VDM [21], which trains a diffusion model on video pixels, our method shows a better IS even though VDM uses more data (uses train+test split) than our setup (uses train split only). Intriguingly, our method shows a notable

improvement on UCF, a complicated, multi-class dataset, which shows the potential of our method to model complex video distribution. We also note that our method also achieves state-of-the-art results in 128 frame videos, which demonstrates the effectiveness of our method in generating longer videos. We provide the qualitative comparison with other video generation methods in Appedix E.

Long video generation. We visualize the long video generation results of our PVDM in Figure 3. As shown in this Figure, our PVDM has the powerful capability of generating long videos (128 frames) with 256×256 resolution frames while maintaining temporal coherency across timesteps. Here, we emphasize that our method produces long videos not only on fine-grained datasets with monotonic motion (*e.g.*, SkyTimelapse) but also on UCF-101, which is a complex dataset that contains various dynamic motions. We also note that synthesizing long videos on such a complex UCF-101 dataset has been regarded as a challenging task in prior state-of-the-art video generation approaches that target long video generation [12,47,67], and thus they often fail to generate temporally coherent and realistic videos on these datasets. We remark that the superiority of our method on long video synthesis is also verified quantitatively as FVD$_{128}$ in Table 1: our method significantly outperforms prior methods on this metric on UCF-101 as 1773.4 → 639.7.

4.3. Analysis

Reconstruction quality. Figure 5 and Table 4 summarize the results of the reconstructed videos from our autoencoder. We consider the following metrics to measure the quality: peak signal-to-noise-ratio (PSNR) for reconstruction quality and R-FVD for perceptual similarity. Here, R-FVD indicates FVD between reconstructions and the ground-truth real videos. Measured with these metrics, our method consis-

---

[2]We provide the illustrations of synthesized videos from our method and comparison with other baselines in the following project website: https://sihyun.me/PVDM.

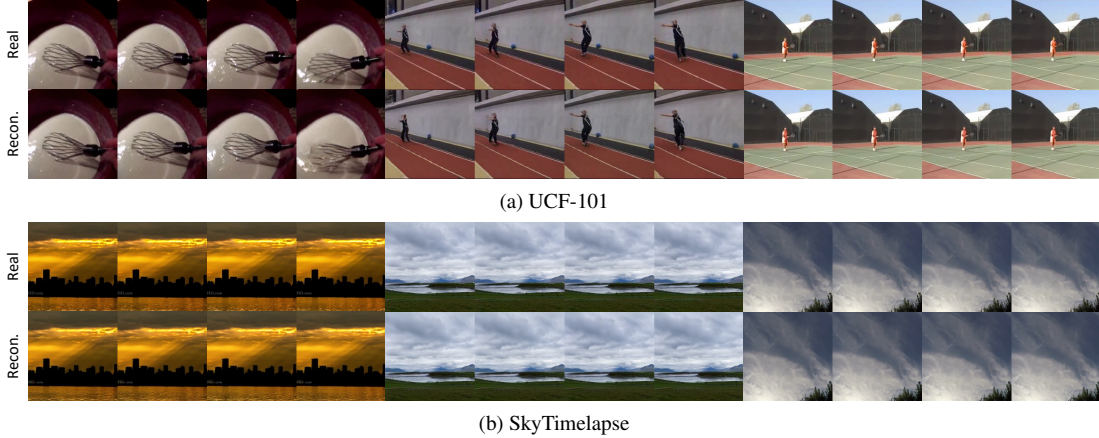(a) UCF-101

(b) SkyTimelapse

Figure 5. Reconstruction results from our autoencoder trained on (a) UCF-101 and (b) SkyTimelapse. We visualize the frames with stride 4.

Table 4. Quantitative evaluation results between reconstruction from the autoencoder of PVDM and the real videos.

	UCF-101		SkyTimelapse	
	Train	Test	Train	Test
R-FVD ↓	25.87	32.26	7.37	36.52*
PSNR ↑	27.34	26.99	34.33	32.68*

∗ Evaluated with 196 samples due to the smaller test set size (196) than 2,048.

tently shows accurate reconstructions similar to ground-truth videos. In particular, our method shows a small enough R-FVD, which validates the effectiveness of our latent space in compressing videos while preserving perceptual similarity. We also provide a more extensive analysis, including a comparison with other popular autoencoders in Appendix F.

Comparison with VDM. Compared with a recent video diffusion model (VDM) [21], our model achieves great computation and memory efficiency in generating samples due to the use of 2D instead of 3D convolution networks and the low-dimensionality of latent vector that encodes high-dimensional video pixels. Following the model configurations of VDM as described in [21], Table 5 compares the maximum batch size that can be allocated in training as well as the memory and time to synthesize a single video at a 256×256 resolution. Under the same sampler setup, PVDM achieves $\approx 17.6\times$ better computation efficiency: our method requires ≈ 7.88 seconds to generate a video with 256×256 resolution and length of 16, while VDM requires >2 minutes to generate such videos. Moreover, our PVDM shows at most $3.5\times$ better memory efficiency; it can be trained with videos at 256×256 resolution and length of 16 and synthesize longer videos (*e.g.*, length 128) under the limited memory constraint (24GB), yet VDM cannot be trained and generate these videos under these computational resources. We also provide the comparison with two recent autoregressive video synthesis methods (TATS and VideoGPT): PVDM still achieves superior efficiencies in both time and memory.

Table 5. Maximum batch size for training and time (s), memory (GB) for synthesizing a 256×256 resolution video measured with a single NVIDIA 3090Ti 24GB GPU. N/A denotes the values cannot be measured due to the out-of-memory problem. N/M-s denotes the model is evaluated with the DDIM sampler [51] with N steps (for the initial clip) and M steps (for future clips).

	Train	Inference (time/memory)	
Length →	16	16	128
TATS [12]	0	84.8/18.7	434/19.2
VideoGPT [65]	0	139/15.2	N/A
VDM [21]; 100/20-s	0	113/11.1	N/A
PVDM-L (ours); 200/200-s	2	20.4/5.22	166/5.22
PVDM-L (ours); 400/400-s	2	40.9/5.22	328/5.22
PVDM-S (ours); 100/20-s	**7**	**7.88/4.33**	**31.3/4.33**

5. Conclusion

We proposed PVDM, a latent diffusion model for video generation. Our key idea is based on proposing an image-like 2D latent space that effectively parameterizes a given video so that the given video data distribution can be effectively learned via diffusion models in latent space. We hope our method will initiate lots of intriguing directions in effectively scaling video synthesis methods.

Acknowledgements

References

[1] Dinesh Acharya, Zhiwu Huang, Danda Pani Paudel, and Luc Van Gool. Towards high resolution video generation with progressive growing of sliced Wasserstein GANs. *arXiv preprint arXiv:1810.02419*, 2018. 2, 6, 7, 13

[2] Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*, 2020. 1

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, 2017. 13

[4] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *International Conference on Machine Learning*, 2021. 4, 7

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, 2020. 1

[6] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 1, 3

[7] Aidan Clark, Jeff Donahue, and Karen Simonyan. Adversarial video generation on complex datasets. *arXiv preprint arXiv:1907.06571*, 2019. 2, 6, 13

[8] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020. 1

[9] Prafulla Dhariwal and Alex Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems*, 2021. 1, 2, 5

[10] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 1, 4, 13

[11] Gereon Fox, Ayush Tewari, Mohamed Elgharib, and Christian Theobalt. StyleVideoGAN: A temporal generative model using a pretrained StyleGAN. *arXiv preprint arXiv:2107.07224*, 2021. 2

[12] Songwei Ge, Thomas Hayes, Harry Yang, Xi Yin, Guan Pang, David Jacobs, Jia-Bin Huang, and Devi Parikh. Long video generation with time-agnostic vqgan and time-sensitive transformer. In *European Conference on Computer Vision*, 2022. 1, 2, 6, 7, 8, 12, 13

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 2014. 2, 12, 13

[14] Cade Gordon and Natalie Parde. Latent neural differential equations for video generation. In *NeurIPS 2020 Workshop on Pre-registration in Machine Learning*, 2021. 2

[15] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 1, 3, 4

[16] William Harvey, Saeid Naderiparizi, Vaden Masrani, Christian Weilbach, and Frank Wood. Flexible diffusion modeling of long videos. *arXiv preprint arXiv:2205.11495*, 2022. 1, 2, 3

[17] Yingqing He, Tianyu Yang, Yong Zhang, Ying Shan, and Qifeng Chen. Latent video diffusion models for high-fidelity video generation with arbitrary lengths. *arXiv preprint arXiv:2211.13221*, 2022. 15

[18] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P. Kingma, Ben Poole, Mohammad Norouzi, David J. Fleet, and Tim Salimans. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022. 1, 3

[19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020. 2, 3, 4

[20] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 5

[21] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *arXiv preprint arXiv:2204.03458*, 2022. 1, 2, 3, 5, 6, 7, 8, 13

[22] Tobias Höppe, Arash Mehrjou, Stefan Bauer, Didrik Nielsen, and Andrea Dittadi. Diffusion models for video prediction and infilling. *arXiv preprint arXiv:2206.07696*, 2022. 1, 2, 3

[23] Emmanuel Kahembwe and Subramanian Ramamoorthy. Lower dimensional kernels for video discriminators. *Neural Networks*, 132:506–520, 2020. 2, 6, 7, 13

[24] Nal Kalchbrenner, Aäron Oord, Karen Simonyan, Ivo Danihelka, Oriol Vinyals, Alex Graves, and Koray Kavukcuoglu. Video pixel networks. In *International Conference on Machine Learning*, 2017. 2

[25] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 12

[26] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. *International Conference on Learning Representations*, 2017. 13

[27] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *arXiv preprint arXiv:2106.12423*, 2021. 1

[28] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving

the image quality of StyleGAN. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 7

[29] Subin Kim, Sihyun Yu, Jaeho Lee, and Jinwoo Shin. Scalable neural video representations with learnable positional features. In *Advances in Neural Information Processing Systems*, 2022. 5

[30] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014. 3, 7

[31] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*, 2021. 1, 2

[32] Kushal Lakhotia, Evgeny Kharitonov, Wei-Ning Hsu, Yossi Adi, Adam Polyak, Benjamin Bolte, Tu-Anh Nguyen, Jade Copet, Alexei Baevski, Adelrahman Mohamed, et al. Generative spoken language modeling from raw audio. *arXiv preprint arXiv:2102.01192*, 2021. 1

[33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018. 7

[34] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2837–2845, 2021. 2

[35] Kangfu Mei and Vishal M. Patel. Vidm: Video implicit diffusion models. In *AAAI Conference on Artificial Intelligence*, 2023. 1, 3

[36] Andres Munoz, Mohammadreza Zolfaghari, Max Argus, and Thomas Brox. Temporal shift GAN for large scale video generation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021. 2

[37] Aaron Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George Driessche, Edward Lockhart, Luis Cobo, Florian Stimberg, et al. Parallel wavenet: Fast high-fidelity speech synthesis. In *International Conference on Machine Learning*, 2018. 4

[38] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022. 1

[39] Ruslan Rakhimov, Denis Volkhonskiy, Alexey Artemov, Denis Zorin, and Evgeny Burnaev. Latent video transformer. *arXiv preprint arXiv:2006.10704*, 2020. 2

[40] 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

[41] 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*, 2022. 1, 3, 4, 7, 12, 14

[42] Masaki Saito, Eiichi Matsumoto, and Shunta Saito. Temporal generative adversarial nets with singular value clipping. In *IEEE International Conference on Computer Vision*, 2017. 2, 13

[43] Masaki Saito, Shunta Saito, Masanori Koyama, and Sosuke Kobayashi. Train sparsely, generate densely: Memory-efficient unsupervised training of high-resolution temporal GAN. *International Journal of Computer Vision*, 128:2586–2606, 2020. 2, 6, 7, 13

[44] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. *Advances in Neural Information Processing Systems*, 2016. 2, 6, 12

[45] Younggyo Seo, Kimin Lee, Fangchen Liu, Stephen James, and Pieter Abbeel. Autoregressive latent video prediction with high-fidelity image generator. In *IEEE International Conference on Image Processing*, 2022. 4

[46] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. In *International Conference on Learning Representations*, 2023. 2, 15

[47] Ivan Skorokhodov, Sergey Tulyakov, and Mohamed Elhoseiny. StyleGAN-V: A continuous video generator with the price, image quality and perks of StyleGAN2. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2, 6, 7, 12, 13

[48] Ivan Skorokhodov, Sergey Tulyakov, Yiqun Wang, and Peter Wonka. Epigraf: Rethinking training of 3d gans. In *Advances in Neural Information Processing Systems*, 2022. 1, 3

[49] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, 2015. 2

[50] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, 2015. 4

[51] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. 1, 3, 7, 8

[52] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, 2019. 2

[53] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. 2

[54] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 2, 6, 12

[55] Yu Tian, Jian Ren, Menglei Chai, Kyle Olszewski, Xi Peng, Dimitris N Metaxas, and Sergey Tulyakov. A good image generator is what you need for high-resolution video synthesis. In *International Conference on Learning Representations*, 2021. 1, 2, 6, 7, 13

[56] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3D convolutional networks. In *IEEE International Conference on Computer Vision*, 2015. 12

[57] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. MoCoGAN: Decomposing motion and content for video generation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2, 6, 7, 13

[58] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018. 2, 6, 12

[59] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. In *Neural Information Processing Systems*, 2021. 1, 3, 4

[60] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, 2017. 1, 13

[61] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017. 2, 4, 7, 13

[62] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *Advances in Neural Information Processing Systems*, 2016. 2, 13

[63] Dirk Weissenborn, Oscar Täckström, and Jakob Uszkoreit. Scaling autoregressive video models. In *International Conference on Learning Representations*, 2020. 2

[64] Wei Xiong, Wenhan Luo, Lin Ma, Wei Liu, and Jiebo Luo. Learning to generate time-lapse videos using multi-stage dynamic generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2, 6, 12

[65] Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. VideoGPT: Video generation using VQ-VAE and transformers. *arXiv preprint arXiv:2104.10157*, 2021. 2, 4, 6, 7, 8, 13

[66] Ruihan Yang, Prakhar Srivastava, and Stephan Mandt. Diffusion probabilistic modeling for video generation. *arXiv preprint arXiv:2203.09481*, 2022. 1, 2, 3

[67] Sihyun Yu, Jihoon Tack, Sangwoo Mo, Hyunsu Kim, Junho Kim, Jung-Woo Ha, and Jinwoo Shin. Generating videos with dynamics-aware implicit generative adversarial networks. In *The Tenth International Conference on Learning Representations*, 2022. 1, 2, 6, 7, 12, 13

[68] Vladyslav Yushchenko, Nikita Araslanov, and Stefan Roth. Markov decision process for video generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019. 2

[69] Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. *arXiv preprint arXiv:2204.13902*, 2022. 3

[70] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 4, 12

[71] Daquan Zhou, Weimin Wang, Hanshu Yan, Weiwei Lv, Yizhe Zhu, and Jiashi Feng. Magicvideo: Efficient video generation with latent diffusion models. *arXiv preprint arXiv:2211.11018*, 2022. 15