Long Video Generation with Time-Agnostic VQGAN and Time-Sensitive Transformer

Songwei Ge[1], Thomas Hayes[2], Harry Yang[2], Xi Yin[2], Guan Pang[2]
David Jacobs[1], Jia-Bin Huang[1,2], and Devi Parikh[2,3]

[1]University of Maryland [2]Meta AI [3]Georgia Tech

Abstract. Videos are created to express emotion, exchange information, and share experiences. Video synthesis has intrigued researchers for a long time. Despite the rapid progress driven by advances in visual synthesis, most existing studies focus on improving the frames' quality and the transitions between them, while little progress has been made in generating longer videos. In this paper, we present a method that builds on 3D-VQGAN and transformers to generate videos with thousands of frames. Our evaluation shows that our model trained on 16-frame video clips from standard benchmarks such as UCF-101, Sky Time-lapse, and Taichi-HD datasets can generate diverse, coherent, and high-quality long videos. We also showcase conditional extensions of our approach for generating meaningful long videos by incorporating temporal information with text and audio. Videos and code can be found at https://songweige.github.io/projects/tats/index.html.

Keywords: video, generation

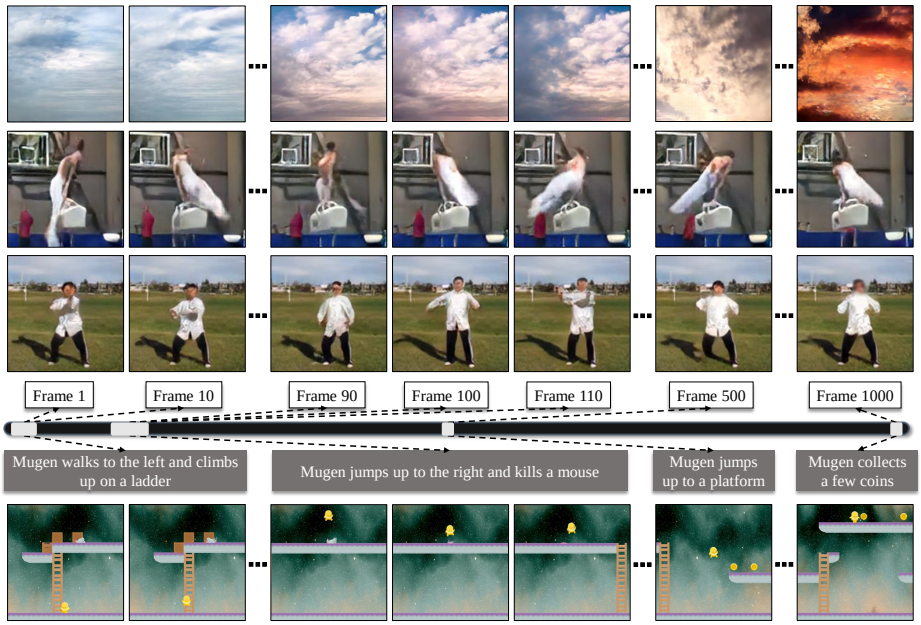


Fig. 1: **Generated long videos with** 1024 **frames.** We present TATS, a framework trained on short video clips that synthesizes videos with thousands of frames with natural motion, long-term coherence, and following real story flow.

1 Introduction

From conveying emotions that break language and cultural barriers to being the most popular medium on social platforms, videos are arguably the most informative, expressive, diverse, and entertaining visual form. Video synthesis has been an exciting yet long-standing problem. The challenges include not only achieving high visual quality in each frame and a natural transitions between frames, but a consistent theme, and even a meaningful storyline throughout the video. The former has been the focus of many existing studies [71,55,12,65,81] working with tens of frames. The latter is largely unexplored and requires the ability to model *long-range temporal dependence* in videos with many more frames.

Prior works and their limitations. *GAN-based methods* [71,55,12,40] can generate plausible short videos, but extending them to longer videos requires prohibitively high memory and time cost of training and inference.[1] *Autoregressive methods* alleviate the training cost constraints through *sequential prediction*. For example, RNNs and LSTMs generate temporal noise vectors for an image generator [55,67,12,56,45,65]; transformers either directly generate pixel values [28,75] or indirectly predict latent tokens [46,16,52,77,36,81]. These approaches circumvent training on the long videos directly by unrolling the RNN states or using a sliding window during inference. Recent works use *implicit neural representations* to reduce cost by directly mapping temporal positions to either pixels [82] or StyleGAN [32] feature map values [61]. However, the visual quality of the generated frames deteriorates quickly when performing generation beyond the training video length for all such methods as shown in Figure 2.

Our work. We tackle the problem of long video generation. Building upon the recent advances of VQGAN [16] for high-resolution *image generation*, we first develop a baseline by extending the 2D-VQGAN to 3D (2D space and 1D time) for modeling videos. This naively extended method, however, fails to produce high-quality, coherent long videos. Our work investigates the model design and identifies simple changes that significantly improve the capability to generate long videos of thousands of frames without quality degradation when conditioning on no or weak information. Our core insights lie in 1) removing the undesired dependence on time from VQGAN and 2) enabling the transformer to capture long-range temporal dependence. Below we outline these two key ideas.

Time-agnostic VQGAN. Our model is trained on short video clips, e.g., 16 frames, like the previous methods [71,55,67,12,56,65]. At inference time, we use a sliding window approach [7] on the transformer to sample tokens for a longer length. The sliding window repeatedly appends the most recently generated tokens to the partial sequence and drops the earliest tokens to maintain a fixed sequence length. However, applying a sliding attention window to 3D-VQVAE (e.g. VideoGPT [81]) or 3D-VQGAN fails to preserve the video quality *beyond the training length*, as shown in Figure 2. The reason turns out to be that the zero

[1] Training DVD-GAN [12] or DVD-GAN-FP [40] on 16-frame videos requires 32-512 TPU replicas and 12-96 hours.

Fig. 2: Video generation results with a vanilla video VQGAN and a time-agnostic VQGAN, within and beyond the training length using sliding window attention.

paddings used in these models *corrupts* the latent tokens and results in token sequences at the inference time that are drastically different from those observed during training when using sliding window. The amount of corruption depends on the temporal position of the token. [2] We address this issue by using *replicate padding* that mitigates the corruption by better approximating the real frames and brings no computational overhead. As a result, the transformer trained on the tokens encoded by our time-agnostic VQGAN effectively preserves the visual quality *beyond the training video length*.

Time-sensitive transformer. While removing the temporal dependence in VQGAN is desirable, long video generation certainly needs temporal information! This is necessary to model long-range dependence through the video and follow a sequence of events a storyline might suggest. While transformers can generate arbitrarily long sequences, errors tend to accumulate, leading to quality degradation for long video generation. To mitigate this, we introduce a hierarchical architecture where an *autoregressive transformer* first generates a set of sparse latent frames, providing a more global structure. Then an *interpolation transformer* fills in the skipped frames autoregressively while attending to the generated sparse frames on both ends. With these modifications, our transformer models long videos more effectively and efficiently. Together with our proposed 3D-VQGAN, we call our model Time-Agnostic VQGAN and Time-Sensitive Transformer (TATS). We highlight the capability of TATS by showing generated video samples of 1024 frames in Figure 1.

Our results. We evaluate our model on several video generation benchmarks. We first consider a standard short video generation setting. Then, we carefully analyze its effectiveness for long video generation, comparing it against several recent models. Given that the evaluation of long video generation has not been well studied, we generalize several popular metrics for this task considering im-

[2] The large spatial span in image synthesis disguises this issue in [16]. When using sliding window to generate tokens near the border, the problem resurfaces as shown in supp. mat. Figure 12.

portant evaluation axes for video generation models, including long term quality and coherence. Our model achieves state-of-the-art short and long video generation results on the UCF-101 [62], Sky Time-lapse [79], Taichi-HD [57], and AudioSet-Drum [19] datasets. We further demonstrate the effectiveness of our model by conditioning on temporal information such as text and audio.

Our contributions.

- We identify the undesired temporal dependence introduced by the zero paddings in VQGAN as a cause of the ineffectiveness of applying a sliding window for long video generation. We propose a simple yet effective fix.
- We propose a hierarchical transformer that can model longer time dependence and delay the quality degradation, and show that our model can generate meaningful videos according to the story flow provided by text or audio.
- To our knowledge, we are the first to generate long videos and analyze their quality. We do so by generalizing several popular metrics to a longer video span and showing that our model can generate more diverse, coherent, and higher-quality long videos.

2 Methodology

In this section, we first briefly recap the VQGAN framework and describe its extension to video generation. Next, we present our time-agnostic VQGAN and time-sensitive transformer models for long video generation (Figure 3).

2.1 Extending the VQGAN framework for video generation

Vector Quantised Variational AutoEncoder (VQVAE) [46] uses a discrete bottleneck as the latent space for reconstruction. An autoregressive model such as a transformer is then used to model the prior distribution of the latent space. VQGAN [16] is a variant of VQVAE that uses perceptual and GAN losses to achieve better reconstruction quality when increasing the bottleneck compression rates.

Vanilla video VQGAN. We adapt the VQGAN architecture for video generation by replacing its 2D convolution operations with 3D convolutions. Given a video $\mathbf{x} \in \mathbb{R}^{T \times H \times W \times 3}$, the VQVAE consists of an encoder $f_{\mathcal{E}}$ and a decoder $f_{\mathcal{G}}$. The discrete latent tokens $\mathbf{z} = \mathbf{q}(f_{\mathcal{E}}(\mathbf{x})) \in \mathbb{Z}^{t \times h \times w}$ with embeddings $\mathbf{c}_z \in \mathbb{R}^{t \times h \times w \times c}$ are computed using a quantization operation $\mathbf{q}$ which applies nearest neighbor search using a trainable codebook $\mathcal{C} = \{\mathbf{c}_i\}_{i=1}^{K}$. The embeddings of the tokens are then fed into the decoder to reconstruct the input $\hat{\mathbf{x}} = f_{\mathcal{G}}(\mathbf{c}_z)$. The VQVAE is trained with the following loss:

$$\mathcal{L}_{\text{vqvae}} = \underbrace{\|\mathbf{x} - \hat{\mathbf{x}}\|_1}_{\mathcal{L}_{\text{rec}}} + \underbrace{\|\text{sg}[f_{\mathcal{E}}(\mathbf{x})] - \mathbf{c}_z\|_2^2}_{\mathcal{L}_{\text{codebook}}} + \underbrace{\beta\|\text{sg}\,[\mathbf{c}_z] - f_{\mathcal{E}}(\mathbf{x})\|_2^2}_{\mathcal{L}_{\text{commit}}},$$

where sg is a stop-gradient operation and we use $\beta = 0.25$ following the VQGAN paper [16]. We optimize $\mathcal{L}_{\text{codebook}}$ using an EMA update and circumvent the non-differentiable quantization step $\mathbf{q}$ with a straight-through gradient estimator [46].

VQGAN additionally adopts a perceptual loss [26,83] and a discriminator $f_{\mathcal{D}}$ to improve the reconstruction quality. Similar to other GAN-based video generation models [67,12], we use two types of discriminators in our model - a spatial discriminator $f_{\mathcal{D}_s}$ that takes in random reconstructed frames $\hat{\mathbf{x}}_i \in \mathbb{R}^{H\times W\times 3}$ to encourage frame quality and a temporal discriminator $f_{\mathcal{D}_t}$ that takes in the entire reconstruction $\hat{\mathbf{x}} \in \mathbb{R}^{T\times H\times W\times 3}$ to penalize implausible motions:

$$\mathcal{L}_{\text{disc}} = \log f_{\mathcal{D}_{s/t}}(\mathbf{x}) + \log(1 - f_{\mathcal{D}_{s/t}}(\hat{\mathbf{x}}))$$

We also use feature matching losses [74,73] to stabilize the GAN training:

$$\mathcal{L}_{\text{match}} = \sum_i p_i \left\| f^{(i)}_{\mathcal{D}_{s/t}/\text{VGG}}(\hat{\mathbf{x}}) - f^{(i)}_{\mathcal{D}_{s/t}/\text{VGG}}(\mathbf{x}) \right\|_1,$$

where $f^{(i)}_{\mathcal{D}_{s/t}/\text{VGG}}$ denotes the i^{th} layer of either a trained VGG network [58] or discriminators with a scaling factor p_i. When using a VGG network, this loss is known as the perceptual loss [83]. p_i is a learned constant for VGG and the reciprocal of the number of elements in the layer for discriminators. Our overall VQGAN training objective is as follows:

$$\min_{f_{\mathcal{E}}, f_{\mathcal{G}}, \mathcal{C}} \left(\max_{f_{\mathcal{D}_s}, f_{\mathcal{D}_t}} (\lambda_{\text{disc}} \mathcal{L}_{\text{disc}}) \right) + \\ \min_{f_{\mathcal{E}}, f_{\mathcal{G}}, \mathcal{C}} (\lambda_{\text{match}} \mathcal{L}_{\text{match}} + \lambda_{\text{rec}} \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{codebook}} + \beta \mathcal{L}_{\text{commit}})$$

Directly applying GAN losses to VideoGPT [81] or 3D VQGAN [16] leads to training stability issues. In addition to the feature matching loss, we discuss other necessary architecture choices and training heuristics in supp. mat. A.1.

Autoregressive prior model. After training the video VQGAN, each video can be encoded into its discrete representation $\mathbf{z} = \mathbf{q}(f_{\mathcal{E}}(\mathbf{x}))$. Following VQGAN [16], we unroll these tokens into a 1D sequence using the row-major order frame by frame. We then train a transformer $f_{\mathcal{T}}$ to model the prior categorical distribution of $\mathbf{z}$ in the dataset autoregressively:

$$p(\mathbf{z}) = p(\mathbf{z}_0) \prod_{i=0}^{t\times h\times w-1} p(\mathbf{z}_{i+1}|\mathbf{z}_{0:i}),$$

where $p(\mathbf{z}_{i+1}|\mathbf{z}_{0:i}) = f_{\mathcal{T}}(\mathbf{z}_{0:i})$ and $\mathbf{z}_0$ is given as the start of sequence token. We train the transformer to minimize the negative log-likelihood over training samples:

$$\mathcal{L}_{\text{transformer}} = \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z}_{\text{data}})}[-\log p(\mathbf{z})]$$

At inference time, we randomly sample video tokens from the predicted categorical distribution $p(\mathbf{z}_{i+1}|\mathbf{z}_{0:i})$ in sequence and feed them into the decoder to generate the videos $\hat{\mathbf{x}} = f_{\mathcal{G}}(\hat{\mathbf{c}}_z)$. To synthesize videos longer than the training length, we generalize sliding attention window for our use [7] . A similar idea has been used in 2D to generate images of higher resolution [16]. We denote the

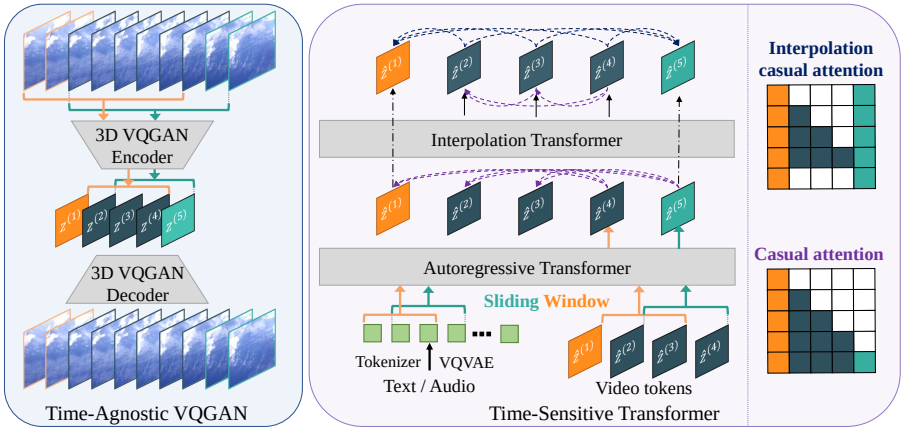


Fig. 3: **Overview of the proposed framework.** Our model contains two modules: time-agnostic VQGAN and time-sensitive transformer. The former compresses the videos both temporally and spatially into discrete tokens without injecting any dependence on the relative temporal position, which allows the usage of a sliding window during inference for longer video generation. The latter uses a hierarchical transformer for capturing longer temporal dependence.

j^{th} temporal slice of $\mathbf{z}$ to be $\mathbf{z}^{(j)} \in \mathbb{Z}^{h \times w}$, where $0 \leq j \leq t-1$. For instance, to generate $\mathbf{z}^{(t)}$ that is beyond the training length, we condition on the $t-1$ slices before it to match the transformer sequence length $p(\mathbf{z}^{(t)}|\mathbf{z}^{(1:t-1)}) = f_{\mathcal{T}}(\mathbf{z}^{(1:t-1)})$. However, as shown in Figure 2, when paired with sliding window attention, the vanilla video VQGAN and transformer cannot generate longer videos without quality degradation. Next, we discuss the reason and a simple yet effective fix.

2.2 Time-Agnostic VQGAN

When the Markov property holds, a transformer with a sliding window can generate arbitrarily long sequences as demonstrated in long article generation [7]. However, a crucial premise that has been overlooked is that the transformer needs to see sequences that start with tokens similar to $\mathbf{z}^{(1:t-1)}$ during its training to predict token $\mathbf{z}^{t}$. This premise breaks down in VQGAN. We provide some intuitions about the reason and defer a detailed discussion to supp. mat. A.2.

Different from natural language modeling where the tokens come from realistic data, VQGAN tokens are produced by an encoder $f_{\mathcal{E}}$ which by default adopts zero paddings for the desired output shape. When a short video clip is encoded, the zero paddings in the temporal dimension also get encoded and affect the output tokens, causing an unbalanced effects to tokens at different temporal position [24,33,80,3]. The tokens closer to the temporal boundary will be affected more significantly. As a result, for real data to match $\mathbf{z}^{(1:t-1)}$, they have to contain these zero-frames, which is not the case in practice. Therefore,

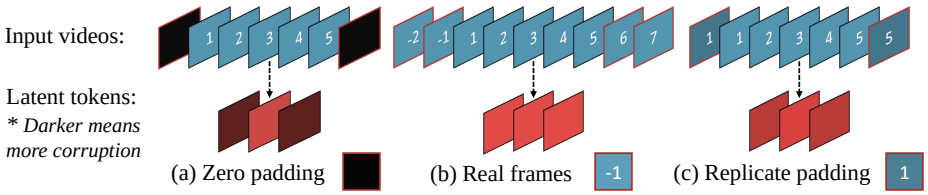


Fig. 4: **Illustration of the temporal effects induced by paddings**. Real frame padding makes the encoder temporally shift-equivariant[3] but introduces extra computations. Replicate padding makes decent approximation to the real frames while bringing no computational overhead.

removing those paddings in the temporal dimension is crucial for making the encoder *time-agnostic* and enabling sliding window.

After removing all the padding, one needs to pad real frames to both ends of the input videos to obtain the desired output size [30,61]. The number of needed real frames can be as large as $\mathcal{O}(Ld)$, where L is the number of layers and d is the compression rate in the temporal direction of the video. Note that the zeros are also padded in the intermediate layers when applied, and importantly, they need not be computed. But, if we want to pad with realistic values, the input needs to be padded long enough to cover the entire receptive field, and all these extra values in the intermediate layers need to be computed. Although padding with real frames makes the encoder fully time agnostic, it can be expensive with a large compression rate or a deep network. In addition, frames near both ends of the videos need to be discarded for not having enough real frames to pad, and consequently some short videos may be completely dropped.

Instead of padding with real values, we propose to approximate real frames with the values close to them. An assumption, which is often referred to as the "boring videos" assumption in the previous literature [8], is to assume that the videos are frozen beyond the given length. Following the assumption, the last boundary slices are always used for padding without computation. More importantly, this can be readily implemented by the replicate padding mode using a standard machine learning library. An illustration of the effects induced by different paddings can be found in Figure 4. Other reasonable assumptions can also be adopted and may correspond to different padding modes. For instance, the reflected padding mode can be used if videos are assumed to play in reverse beyond their length, which has more realistic motions than the frozen frames. In supp. mat. A.3, we provide a careful analysis of the time dependence when using different padding types and different numbers of real padding frames. We find that the replicate padding alone resolves the time-dependence issue well in practice and inherits the merit of zero paddings in terms of the computational cost. Therefore, in the following experiments, we use the replicate paddings and no real frames padded.

[3] The expressions *temporally shift-equivariant* and *time-agnostic* will be used interchangeably hereinafter.

2.3 Time-Sensitive Transformer

The time-agnostic property of the VQGAN makes it feasible to generate long videos using a transformer with sliding window attention. However, long video generation does need temporal information! To maintain a consistent theme running from the beginning of the video to the end requires the capacity to model long-range dependence. And besides, the spirit of a long video is in its underlying story, which requires both predicting the motions in the next few frames and the ability to plan on how the events in the video proceed. This section discusses the time-sensitive transformer for improved long video generation.

Due to the probabilistic nature of transformers, errors can be introduced when sampling from a categorical distribution, which then accumulate over time. One common strategy to improve the long-term capacity is to use a hierarchical model [17,9] to reduce the chance of drifting away from the target theme. Specifically, we propose to condition one interpolation transformer on the tokens generated by the other autoregressive transformer that outputs more sparsely sampled tokens. The autoregressive transformer is trained in a standard way but on the sampled frames with larger intervals. For the interpolation transformer, it fills in the missing frames between any two adjacent frames generated by the autoregressive transformer. To do so, we propose interpolation attention as shown in Figure 3, where the predicted tokens attend to both the previously generated tokens in a causal way and the tokens from the sparsely sampled frames at both ends. A more detailed description can be found in supp. mat. A.4. We consider an autoregressive transformer that generates $4\times$ more sparse video tokens. Generalizing this model to even more extreme cases such as video generation based on key-frames would be an interesting future work.

Another simple yet effective way to improve the long period coherence is to provide the underlying "storyline" of the desired video directly. The VQVAE framework has been shown to be an effective way to do conditional image generation [16,53]. We hence consider several kinds of conditional information that provide additional temporal information such as audio [10], text [76], and so on. To utilize conditional information for long video generation, we use either a tokenizer or an additional VQVAE to discretize the audio or text and prepend the obtained tokens to the video tokens and remove the start of the sequence token $\mathbf{z}_0$. At inference time, we extend the sliding window by simultaneously applying it to the conditioned tokens and video tokens.

3 Experiments

In this section, we evaluate the proposed method on several benchmark datasets for video generation with an emphasis on long video generation.

3.1 Experimental Setups

Datasets and evaluation. We show results on UCF-101 [62], Sky Time-lapse [79], and Taichi-HD [57] for unconditional or class-conditioned video generation with

Table 1: Quantitative results of standard video generation on different datasets. We report FVD and KVD on the Taichi-HD and Sky Time-lapse datasets, IS and FVD on the UCF-101 dataset, SSIM and PSNR at the 45^{th} frame on the AudioSet-Drum dataset. * denotes training on the entire UCF-101 dataset instead of the train split. The class column indicates whether the class labels are used as conditional information.

(a) Sky Time-lapse

Method	FVD (↓)	KVD (↓)
MoCoGAN-HD	$183.6_{\pm 5.2}$	$13.9_{\pm 0.7}$
DIGAN	$\mathbf{114.6}_{\pm 4.9}$	$6.8_{\pm 0.5}$
TATS-base	$132.56_{\pm 2.6}$	$\mathbf{5.7}_{\pm 0.3}$

(b) TaiChi-HD

Method	FVD (↓)	KVD (↓)
MoCoGAN-HD	$144.7_{\pm 6.0}$	$25.4_{\pm 1.9}$
DIGAN	$128.1_{\pm 4.9}$	$20.6_{\pm 1.1}$
TATS-base	$\mathbf{94.60}_{\pm 2.7}$	$\mathbf{9.8}_{\pm 1.0}$

(c) AudioSet-Drum

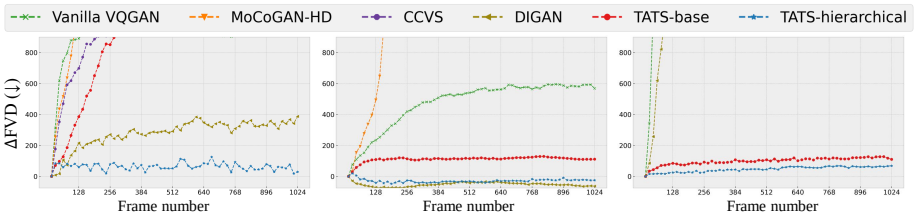
Method	SSIM (↑)	PSNR (↑)
SVG-LP	$0.510_{\pm 0.008}$	$13.5_{\pm 0.1}$
Vougioukas *et al.*	$0.896_{\pm 0.015}$	$23.3_{\pm 0.3}$
Sound2Sight	$0.947_{\pm 0.007}$	$27.0_{\pm 0.3}$
CCVS	$0.945_{\pm 0.008}$	$27.3_{\pm 0.5}$
TATS-base	$\mathbf{0.964}_{\pm 0.005}$	$\mathbf{27.7}_{\pm 0.4}$

(d) UCF-101

Method	Class	IS (↑)	FVD (↓)
VGAN	✓	$8.31_{\pm .09}$	-
TGAN	✗	$11.85_{\pm .07}$	-
TGAN	✓	$15.83_{\pm .18}$	-
MoCoGAN	✓	$12.42_{\pm .07}$	-
ProgressiveVGAN	✓	$14.56_{\pm .05}$	-
LDVD-GAN	✗	$22.91_{\pm .19}$	-
VideoGPT	✗	$24.69_{\pm .30}$	-
TGANv2	✓	$28.87_{\pm .67}$	$1209_{\pm 28}$
DVD-GAN*	✓	$27.38_{\pm .53}$	-
MoCoGAN-HD*	✗	32.36	838
DIGAN	✗	$29.71_{\pm .53}$	$655_{\pm 22}$
DIGAN*	✗	$32.70_{\pm .35}$	$577_{\pm 21}$
CCVS*+Real frame	✗	$41.37_{\pm .39}$	$389_{\pm 14}$
CCVS*+StyleGAN	✗	$24.47_{\pm .13}$	$386_{\pm 15}$
Real data	-	90.52	-
TATS-base	✗	$57.63_{\pm .24}$	$420_{\pm 18}$
TATS-base	✓	$\mathbf{79.28}_{\pm .38}$	$\mathbf{332}_{\pm 18}$

128×128 resolution following [82], AudioSet-Drum [19] for audio-conditioned video generation with 64×64 resolution following [36], and MUGEN [1] with 256×256 resolution for text-conditioned video generation. We follow the previous methods [65,82] to use Fréchet Video Distance (FVD) [68] and Kernel Video Distance (KVD) [68] as the evaluation metrics on UCF-101, Sky Time-lapse, and Taichi-HD datasets. In addition, we follow the methods evaluated on UCF-101 [12,36] to report the Inception Score (IS) [56] calculated by a trained C3D model [66]. For audio-conditioned generation evaluation, we measure the SSIM and PSNR at the 45^{th} frame which is the longest videos considered in the previous methods [10,36]. See supp. mat. B.1 for more details about the datasets.

Training details. To compare our methods with previous ones, we train our VQGAN on videos with 16 frames. We adopt a compression rate $d = 4$ in temporal dimension and $d = 8$ in spatial dimensions. For transformer models, we train a decoder-only transformer with size between GPT-1 [50] and GPT-2 [51] for a consideration of computational cost. We refer to our model with a single autoregressive transformer as **TATS-base** and the proposed hierarchical transformer as **TATS-hierarchical**. For audio-conditioned model, we train another VQGAN to compress the Short-Time Fourier Transform (STFT) data into a dis-



(a) UCF-101 (b) Sky Time-lapse (c) Taichi-HD

Fig. 5: **Quality degradation.** FVD changes of non-overlapping 16-frame clips from the long videos generated by models trained on the UCF-101, Sky Time-lapse, and Taichi-HD datasets. The lower value indicates the slower degradation.

crete space. For text-conditioned model, we use a BPE tokenizer pretrained by CLIP [49]. See supp. mat. B.2 for more details about the training and inference.

3.2 Quantitative Evaluation on Short Video Generation

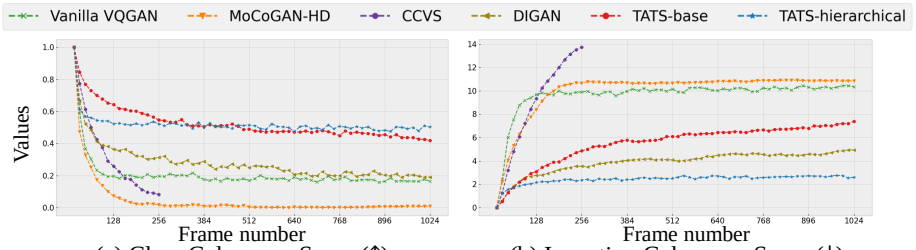
In this section, we demonstrate the effectiveness of our TATS-base model under a standard short video generation setting, where only 16 frames are generated for each video. The quantitative results are shown in Table 1.

Our model achieves state-of-art FVD and KVD on the UCF-101 and Taichi-HD datasets, and state-of-art KVD on the Sky Time-lapse dataset for unconditional video generation, and improved the quality on the AudioSet-Drum for audio-conditioned video generation. For instance, TATS-base improves on IS by 39.3% over the prior methods and 76.2% over the method that does not rely on real frames. On the UCF-101 dataset, we also explore generation with class labels as conditional information. Following the previous method [55], during inference, we sample labels from the prior distribution as the input to the generation. We find that conditioning on the labels significantly eases the transformer modeling task and boosts the generation quality, improving IS from 57.63 to 79.28, which is close to the upper bound shown in the "Real data" row [2,27]. This has been extensively observed in image generation [6,16] but not quite yet revealed in the video generation. TATS-base significantly advances the IS over the previous methods, demonstrating its power in modeling diverse video datasets.

3.3 Quantitative Evaluation on Long Video Generation

Quantitatively evaluating long video generation results has been under explored. In this section, we propose several metrics by generalizing existing metrics to evaluate the crucial aspects of the long videos. We generate 512 videos with 1024 frames on each of the Sky Time-lapse, Taichi-HD, and UCF-101 datasets. We compare our proposed methods, TATS-base and TATS-hierarchical, with the baseline Vanilla VQGAN and the state-of-the-art models including MoCoGAN-HD [65], DIGAN [82], and CCVS [36] by unrolling RNN states, directly sampling, and sliding attention window using their official model checkpoints.

Quality. We measure video quality with respect to the duration by evaluating every 16 frames extracted side-by-side from the generated videos. Ideally, every



(a) Class Coherence Score (↑) (b) Inception Coherence Score (↓)

Fig. 6: **Video coherency.** CCS and ICS values of every 16-frame clip extracted from long videos generated by different models trained on the UCF-101 dataset.

set of 16-frame videos should come from the same distribution as the training set. Therefore, we report the FVD changes of these generated clips compared with the first generated 16 frames in Figure 5, to measure the degradation of the video quality. The figure shows that our TATS-base model successfully delays the quality degradation compared with the vanilla VQGAN baseline, MoCoGAN-HD, and CCVS models. In addition, the TATS-hierarchical model further improves the long-term quality of the TATS-base model. The concurrent work DIGAN [82] also claims the ability of extrapolation, while we show that the generation still degrades severely after certain number frames on the UCF-101 and Taichi-HD datasets. We conjecture the unusual decrease of the FVD w.r.t. the duration of DIGAN and TATS-hierarchical on Sky Time-lapse can be explained by that the I3D model [8] used to calculate FVD is trained on Kinetics-400 dataset, and the sky videos can be outliers of the training data and lead to weak activation in the logit layers and therefore such unusual behaviors.

Coherence. The generated long videos should follow a consistent topical theme. We evaluate the coherence of the videos on the UCF-101 dataset since it has multiple themes (classes). We expect the generated long videos to be classified as the same class all the time. We adopt the same trained C3D model for IS calculated [56], and propose two metrics, Class Coherence Score (CCS) and Inception Coherence Score (ICS) at time step t, measuring the theme similarity between the non-overlapped 16 frames w.r.t. the first 16 frames, defined as below:

$$\mathrm{CCS}_t = \sum_i \frac{\mathbf{1}(\arg\max p_{C3D}(y|x_i^{(0:15)}), \arg\max p_{C3D}(y|x_i^{(t:t+15)})}{N}$$

$$\mathrm{ICS}_t = \sum_i p_{C3D}(y|x_i^{(t:t+15)}) \log \frac{p_{C3D}(y|x_i^{(t:t+15)})}{p_{C3D}(y|x_i^{(0:15)})}$$

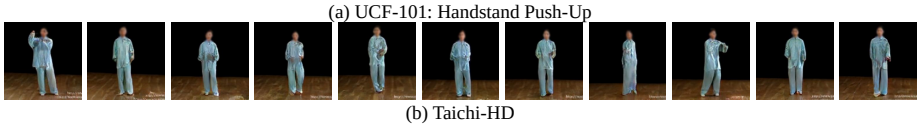
The ICS captures class shift more accurately than the CCS, which only looks at the most probable class. On the other hand, CCS is more intuitive and allows us to define single metrics such as the area under the curve. CCS also doesn't have the asymmetry issue (unlike KL divergence used in ICS). We show the scores of TATS models and several baselines in Figure 6. TATS-base achieves decent coherence as opposite to its quality degradation shown previously. Such

difference can be explained by its failure on a small portion of generated videos as shown in supp. mat. Figure 19, which dominates the FVD. This also shows that CCS and ICS measure coherence on individual videos that complementary to FVD changes. Furthermore, TATS-hierarchical outperforms the baselines on both metrics. For example, more than half of the videos are still classified consistently in the last 16 frames of the entire 1024 frames.

3.4 Qualitative Evaluation on Long Video Generation

This section shows qualitative results of videos with 1024 frames and discusses their common properties. 1024 frames per video approaches the upper bound in the training data as shown in supp. mat. Figure 15, which means the generated videos are probably different from the training videos, at least in duration.

(a) UCF-101: Handstand Push-Up



(b) Taichi-HD

Fig. 7: **Videos with recurrent events.** Every 100^{th} frame is extracted from the generated videos with 1024 frames on the UCF-101 and Taichi-HD datasets.

Recurrent actions. We find that some video themes contain repeated events such as the *Handstand Push-Up* classes in the UCF-101 dataset and videos in the Taichi-HD dataset. As shown in Figure 7, our model generalizes to long video generation by producing realistic and recurrent actions. However, we find that all the 1024 frames in these videos are unique, which shows that our model is not simply copying the short loops.

Smooth transitions. Videos with the same theme often share visual features such as scenes and motions. We show that with enough training data available for a single theme, our model learns to "stitch" the long videos through generating smooth transitions between different videos. For example in Figure 8, we show that our model generates a long sky video containing different weather and timing while the transitions between these conditions are still natural and realistic. To show that this is not the case in the training data, we compute the LPIPS score [83] and color histrogram correlation between the 1^{st} and the 1024^{th} frames and report the mean and standard deviation on the 500 generated and 216 real long videos in Table 2. It shows that such transitions are much more prominent on the generated videos than the training videos. Similar examples of the

Videos	LPIPS metric	Color Similarity
Real	0.1839 ± 0.0683	0.7330 ± 0.2401
Fake	0.3461 ± 0.1184	0.0797 ± 0.1445

Table 2: LPIPS and color histogram correlation between the first and the last frames on the sky videos.

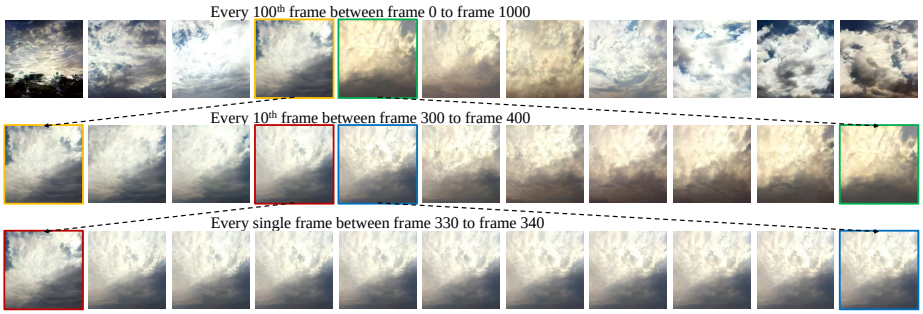


Fig. 8: **Videos with homomorphisms.** Every 100^{th}, 10^{th}, and consecutive frames are extracted from a generated sky video with 1024 frames.

UCF-101 and Taichi-HD videos can be found in supp. mat. Figure 18 A separation of content and motion latent features [67,82,65] may better leverage such transitions, which we leave as future work.

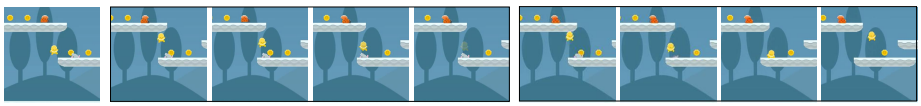


Fig. 9: **Videos with meanings.** Video manipulation by modifying the texts.

Meaningful synthesis. By conditioning on the temporal information, we can achieve more controllable generation, which allows us to directly create or modify videos based our own will. For example, in Figure 9, we show that it is possible to manipulate the videos by changing the underlying storyline - by replacing "killed by" with "kills" we completely change the destiny of Mugen!

4 Related Work

In this section, we discuss the related work in video generation using different models including Generative Adversarial Networks (GAN), Autoregressive models (AR), and implicit neural representations (INR). We focus on the different strategies adopted by these methods to handle the temporal dynamics and discuss their potential for and challenges associated with long video generation. Also see supp. mat. D for a discussion of other relevant models and tasks.

GAN-based video generator. GANs were first proposed to synthesize images and have seen remarkable progress [31,32,20]. Adapting GANs to video synthesis requires modeling the temporal dimension. The earliest work [71] generates entire videos using 3D deconvolutionals. Subsequent works separate the temporal branch from 2D frame generation and adopt RNN or LSTM [22] models to predict the latent vectors as the input to image generators [55,67,67,12,56,45,65]. For instance, MoCoGAN [67] utilizes a RNN to sample motion vectors to synthesize different frames. MoCoGAN-HD [65] leverages a LSTM to predict a trajectory

in the latent space of a trained image generator. By unrolling the steps taken by the RNNs, videos of duration longer than those seen during training can be generated. However, as shown in our experiments, the quality of these videos degrades quickly. In addition, without further modifications, the length of videos generated by these models is limited by the GPU memory (e.g., at most 140 frames can be generated on 32GB V100 GPU by HVG [9]).

AR-based video generator. Autoregressive models generate images or videos pixel-by-pixel and have become a ubiquitous generative model for video synthesis. [54,63] are the first to utilize RNN and LSTM for video prediction. Video Pixel Network [28] generalizes PixelCNN [47] for video generation. More recently, Video Transformer [75] builds its architecture on the Transformer [70]. A common challenge in AR models is their slow inference speed. Owing to the development of VQVAE [46], this problem is mitigated by training AR models on the compressed tokens [52]. Furthermore, NÜWA [77] unifies the training of generative models for images and videos using a frame-based VQGAN and adopts 3D nearby self-attention for better efficiency. A frame-based compression often leads to temporal artifacts and does not fully utilize temporal redundancy. To this end, CCVS [36] utilizes a flow module to improve temporal consistency. VideoGPT [81] further leverages 3D convolution and axial self-attention. Our model falls into this line of video generators. We show that such a VQVAE-based AR model is promising to generate long videos with long-range dependence.

INR-based video generator. Implicit Neural Representations [59,64] represent continuous signals such as images using a neural network by mapping the coordinate space to RGB space. Some recent works apply it to image synthesis [60,4] by training generators to generate the weights of the neural networks. DIGAN [82] first generalized this idea to video generation by decomposing the network weights for the spatial and temporal coordinates. StyleGAN-v [61] maps a continuous temporal positional embedding to the input feature map of the StyleGAN. The biggest advantage of these models is their ability to generate arbitrarily long videos non-autoregressively. However, their generated videos still suffer from the quality degradation or contain periodic artifacts due to the positional encoding and struggle at synthesizing new content.

The implicit bias of zero padding. The positional inductive bias introduced by zero padding has drawn increasing attention recently [24,33,80,3]. In most cases, such inductive bias is helpful in classification [24], generation [80], or object detection [3]. Our paper observes a case where this inductive bias is harmful.

5 Conclusion

We propose TATS, a time-agnostic VQGAN and time-sensitive transformer model, that is only trained on clips with tens of frames and can generate thousands of frames using a sliding window during inference time. Our model generates meaningful videos when conditioned on text and audio. our paper is a small step but we hope it can encourage future works on more interesting forms of video synthesis with a realistic number of frames, perhaps even movies.

A Implementation Details

In this section, we provide additional details on our proposed TATS model, including designs to stabilize training 3D-VQGAN with GAN losses, discussions on the undesired temporal dependence induced by the zero padding, ablations on different potential solutions, and description of our interpolation attention.

A.1 Training video VQGAN with GAN losses

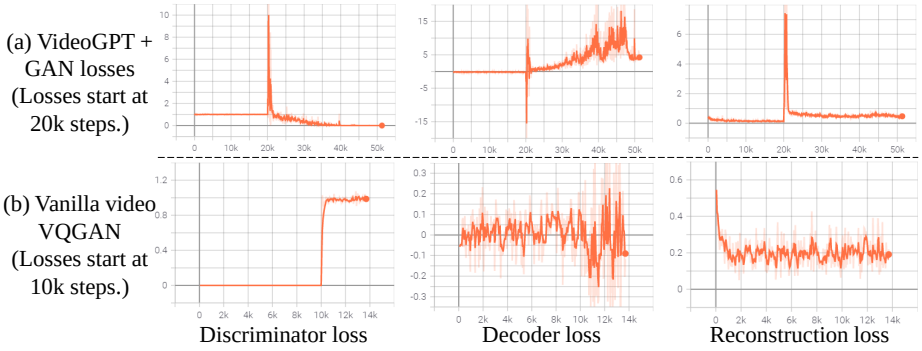


Fig. 10: Training VideoGPT with GAN losses leads to discriminator collapse.

In this section, we describe how we train 3D-VQGAN with GAN losses and clarify several major architecture choices of our proposed vanilla video VQGAN compared with VQGAN [16] and VideoGPT [81]. We find that directly applying GAN losses to VideoGPT leads to severe discriminator collapse. As shown in Figure 10 (a), after adding GAN losses at the 20k step, the discriminator losses saturate quickly and thus provide nonsensical gradient to the decoder. As a result, the decoder (generator) loss and reconstruction loss entirely fall apart. We found that the tricks proposed in VQGAN [16] such as starting GAN losses at a later step and using adaptive weight, do not help the situation. We present several empirical methods we found effective in stabilizing the GAN losses.

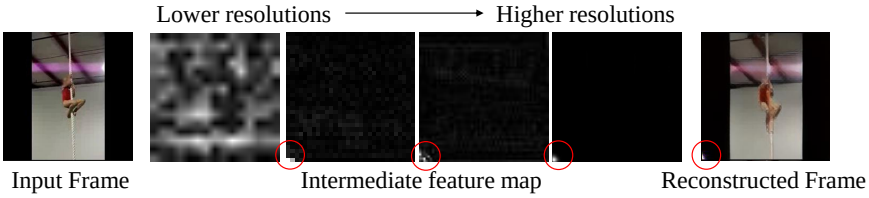


Fig. 11: Blob-shaped artifacts due the the normalization layers in VQGAN.

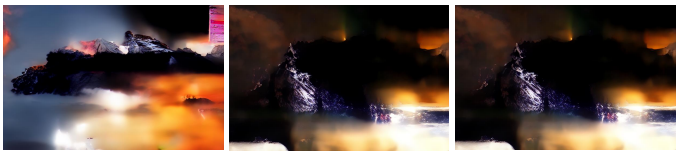
First, we find that the axial-attention layers introduced in VideoGPT interact poorly with the GAN losses and exacerbate the collapse, which is also noticed in recent work on training ViT with GAN losses [37]. Therefore, we utilize a pure

convolution architecture similar to VQGAN [16] for video compression. Second, a more powerful decoder helps the reconstruction follow the discriminator closely. We doubled the number of feature maps whenever the resolutions are halved, in contrast to VideoGPT where all the layers have a constant number of channels. As a consequence, similar to the previous observation [11], we find that training large VAE models would cause exploded gradients. Furthermore, similar to the proposed solution of gradient skipping [11], we always clip the gradient to have the Euclidean norm equal to 1 during the training. Last, we notice that the blob-shaped artifacts often appear in the reconstruction and exaggerate along with the intermediate feature maps as shown in Figure 11, which was also observed in StyleGANs [31,32] and can be attributed to the normalization layers. This is especially pronounced in the training video VQGAN due to the small batch sizes. We use Synced Batch Normalization as a replace of Group Normalization [78] used in original VQGAN [16] and accumulate gradients across multiple steps and successfully mitigate this issue. Our vanilla video VQGAN can be steadily trained with the GAN losses with the above choices of architecture designs.

A.2 Zero paddings inhibit sliding attention window

(a) Conditioned semantic map.

(b) High-resolution images generated by 2D-VQGAN through sampling the tokens in the center.



(c) High-resolution images generated by 2D-VQGAN through sampling the tokens near the border.

Fig. 12: 2D-VQGAN generates larger images by generating tokens in the center.

Following the intuition in Section 2, we provide a more detailed discussion on why zero paddings in VQGAN inhibit the usage of the sliding attention window. We aim to show that when zero padding is used, there are barely tokens starting with $\mathbf{z}^{(1:t-1)}$ in the training set. However, this is necessary for the transformer to generate $\mathbf{z}^{(t)}$ using a sliding window.

For simplicity, we absorb the quantization step $\mathbf{q}$ into $f_{\mathcal{E}}(\mathbf{x})$. In order for there to be tokens starting with $\mathbf{z}^{(1:t-1)}$ in the transformer training data, there need to be real video clips that can be encoded in $\mathbf{z}^{(1:t)}$. It is desired that $f_{\mathcal{E}}$ is temporally shift-equivariant given 3D convolutions so that $\mathbf{z}^{(1:t)}$ is the output of the clips slightly shifted from the original position, i.e. $\mathbf{z}^{(1:t)} = f_{\mathcal{E}}(\mathbf{x}^{(d:T+d-1)})$, where d is the compression rate of $f_{\mathcal{E}}$ in the temporal dimension. However, we find that the encoder is *not* temporally shift-equivariant and encodes $\mathbf{x}^{(d:T-1)}$

differently when these frames are positioned at different places, i.e.

$$[f_{\mathcal{E}}(\mathbf{x}^{(0:T-1)})]^{(1:t-1)} \neq [f_{\mathcal{E}}(\mathbf{x}^{(d:T+d-1)})]^{(0:t-2)} \tag{1}$$

To see this theoretically, we consider a shift-equivariant version of $f_{\mathcal{E}}$ by moving all the internal zero paddings to the input. We denote this encoder as $\hat{f}_{\mathcal{E}}$ such that

$$f_{\mathcal{E}}(\mathbf{x}) = \hat{f}_{\mathcal{E}}([\mathbf{0}^N \ \mathbf{x} \ \mathbf{0}^N]), \tag{2}$$

where $[\mathbf{0}^N \ \mathbf{x} \ \mathbf{0}^N]$ is the concatenation of $\mathbf{x}$ with N zero paddings $\mathbf{0} \in \mathbb{R}^{h \times w}$. One can show that the number of paddings needed is $N = \mathcal{O}(Ld)$, where L is the number of convolutional layers in the encoder. Given the shift equivariance of $\hat{f}_{\mathcal{E}}$, we can derive the desired latent tokens for transformer training as

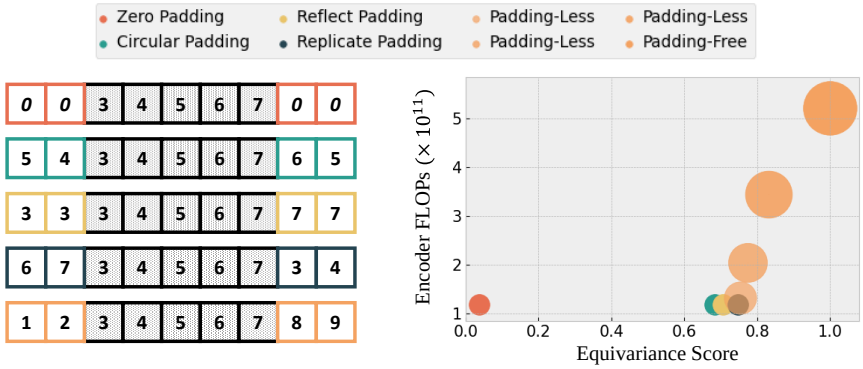
$$z^{(1:t)} = \hat{f}_{\mathcal{E}}([\mathbf{0}^{N-d} \ \mathbf{x}^{(d:T-1)} \ \mathbf{0}^{N+d}]).$$

However, we cannot find such real videos corresponding to $\hat{\mathbf{x}} = [\mathbf{0}^{N-d} \ \mathbf{x}^{(d:T-1)} \ \mathbf{0}^{N+d}]$ as the input to $f_{\mathcal{E}}$ based on the Equation 2 for two reasons. First, according to Equation 2 the input to $f_{\mathcal{E}}$ should be $\hat{\mathbf{x}}^{(N:N+T)} = [\mathbf{x}^{(T+d:T)} \ \mathbf{0}^d]$. There are rare videos whose last d frames are blank in the real datasets. In addition, $\hat{\mathbf{x}}^{(N-d:N)} = \mathbf{x}^{(d:T+d)}$ indicates that real frames are used to pad the input, while $f_{\mathcal{E}}$ only uses zeros. Therefore, we show that no such tokens are starting with $\mathbf{z}^{(1:t-1)}$ in the training set. As a result, the transformer cannot generalize to the sequence starting with $z^{(1:t-1)}$ using a sliding attention window.

This problem occurs in all the generative models that utilize VQVAE and transformers. However, it is often disguised and ignored in previous studies. In practice, the severity of this issue depends on the receptive field, and the relative position of the generated token to the border. In the case of the original VQGAN [16] that uses a sliding window to generate high-resolution images, this issue is disguised as the spatially centered token is always chosen to generate, which is far away from the border and much less affected by the zero-padding given the large spatial size (256). We show in Figure 12 that, when generating the tokens near the border using a sliding window using the high-resolution VQGAN, the quality of images degrades quickly, similar to our observation in video generation. Furthermore, tokens at any position are close to the borders for synthesizing long videos due to the small temporal length (16).

A.3 Quantifying the time agnostics of different padding types

Adding real frames makes the VQGAN fully time-agnostic but increases the computational cost. Using other paddings is less effective but brings no overheads. Therefore, it is essential to quantify the temporal dependence to understand the trade-off between the desired property and the cost to achieve it. To that end, we propose an equivariance score as a measure of time agnostic shown in Equation 1, which calculates the percentage of tokens that are identical when the same frames are positioned at the beginning of the end of the clips, which



(a) Demonstration of different padding types. The numbers represent the index of frames except that *0* represents a frame of 0s.

(b) Trade-off between equivariance scores and FLOPs. The relative GPU memory utilization is indicated by the volume, and the level of padding-free is indicated by the transparency.

Fig. 13: Demonstration of different padding types and their computational costs as well as effects on the consistency score. Note that when using less or no paddings, extra real paddings are added to the input videos.

are the two extreme cases that are mostly affected by the paddings from either side:

$$\sum_{i=1}^{t}\sum_{j=0}^{h-1}\sum_{k=0}^{w-1}\frac{\mathbf{1}\left([f_{\mathcal{E}}(\mathbf{x}_{0:T})]_{i,j,k},[f_{\mathcal{E}}(\mathbf{x}_{d:T+d})]_{i-1,j,k}\right)}{t\times h\times w}, \tag{3}$$

where $\mathbf{1}$ is an identity function. We report the mean and standard deviation across 1024 clips using a video VQGAN trained on the UCF-101 dataset across different padding strategies. We visualize the trade-off between the costs and effects on the consistency score in Figure 13. We also partially remove the zero paddings from different numbers of layers to picture the trade-off varies more accurately, where the padding type is called "Padding-Less".

As shown in Figure 13, we find that the more paddings are removed, the more time agnostic the encoder becomes. Note that when removing all the paddings and concatenating enough real frames to the input, we obtain a perfectly time-agnostic encoder that achieves the equivariance score = 1. However, it also significantly increases the memory and computational costs of the training. As for the other padding types, although the reflect and circular paddings provide more realistic video changes, they could drift further from the real frames and yield a smaller equivariance score than the replicate padding. For example, a walking person is more likely to stop than walk backward in the following frames. We find that the replicate padding, which gives 0.75 consistency score, already resolves the time-dependence issue well in practice. Given the little extra cost it brings, we use replicate paddings and no real frames in our experiments.

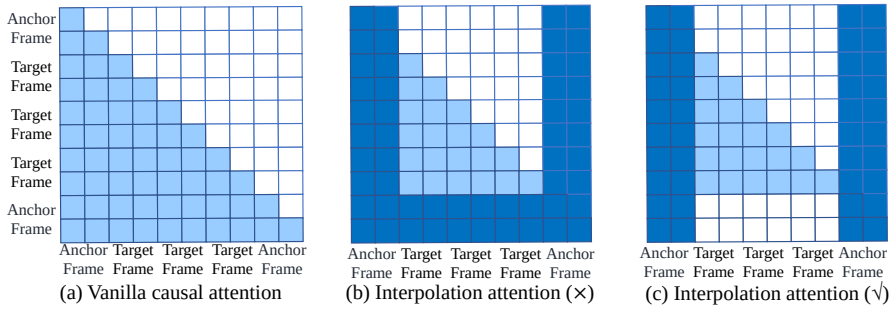


Fig. 14: Illustration of the vanilla causal attention and the interpolation causal attention. For simplification, every frame is assumed to have 2 tokens.

A.4 Details of the interpolation attention

We implement the interpolation transformer based on designed attention called interpolation causal attention, as shown in the right scheme of Figure 14(c). Specifically, the anchor frame (dark) is given during the inference, and the target frame (light) needs to be generated. In the vanilla causal attention shown in Figure 14(a), tokens attend to the tokens *in front of* it. In the interpolation causal attention, tokens attend to *both* the tokens before it and the anchor tokens, which allows acquiring information from the anchor frames at both ends generated by the autoregressive transformer. We want to stress that it is important to not attend the last anchor frame on the frames to be generated like the one in the Figure 14(b), since a multi-layer self-attention will form a shortcut and leak the information of the frames in the middle to themselves and cause the training to collapse.

B Experiment setups

In this section, we provide additional details of our experiments.

B.1 Dataset and evaluation details

We validate our approach on the UCF-101 [62], Sky Time-lapse [79], Taichi-HD [57], AudioSet-Drum [19], and MUGEN [1] datasets. Below we provide basic descriptions of these datasets and our data processing steps on each of them. We also report the relevant dataset statistics such as the number of long videos and the number of frames per video in Figure 15.

– **UCF-101** is a dataset designed for action recognition which contains 101 classes and 13,320 videos in total. We train our model on its *train split* which contains 9,537 videos following the official splits.[4] We also report number of frames in videos of every class in Figure 16.

[4] https://www.crcv.ucf.edu/data/UCF101.php

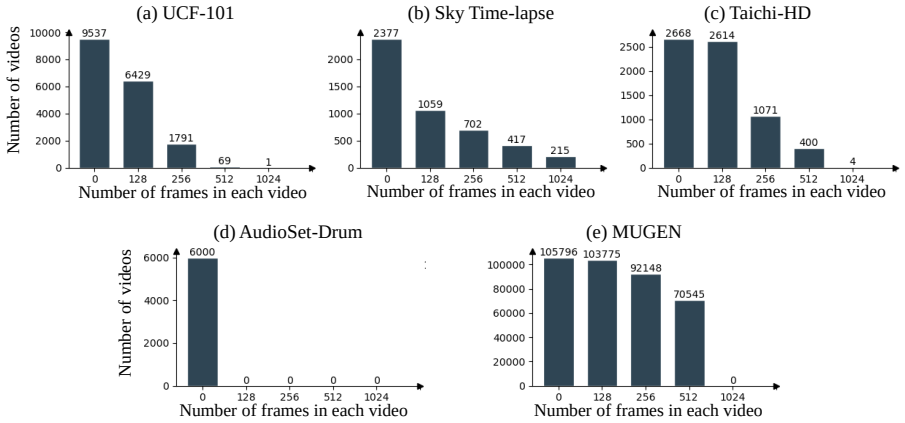


Fig. 15: **Dataset statistics.** Distribution of the number of videos in each dataset that have at least a certain number of frames.

- **Sky Time-lapse** contains time-lapse videos that depict the sky under different time and weather conditions. The paper [79] claims that 5,000 videos are collected but only 2,647 are actually released in the official dataset.[5] We follow [82] to train our model on the train split and test using videos from the test split.
- **Taichi-HD** has 2,668 videos in total recording a single person performing Taichi.[6] We follow [82] to sample frames from every 4 frames when training our TATS-base model for a fair comparison. However, training TATS-Hierarchical with this setting would drop 60% of videos for not having enough frames as shown in Figure 15. Therefore we do not skip frames for TATS-Hierarchical.
- **AudioSet-Drum** is a collection of drum kit videos with audio available in the dataset. The train split contains 6,000 videos, and the test split contains 1,000 videos. All the video clips have 90 frames. We use the STFT features extracted by [10] as the audio data[7], and follow its evaluation setting to measure the image quality of the 45^{th} frames on the test set.
- **MUGEN** is a dataset containing videos collected from the open-sourced platform game CoinRun [13] by recording the gameplay of trained RL agents. We use their template-based algorithm auto-text, which generates textual descriptions for videos with arbitrary lengths. The train and test splits contain 104,796 and 11,802 videos, respectively, each at 3.2s to 21s (96 to 602 frames) long.

To calculate the VFDs and DVDs, we generate 2,048 and 512 videos for short and long video evaluation, respectively, considering time cost. To calculate the

[5] https://github.com/weixiong-ur/mdgan

[6] https://github.com/AliaksandrSiarohin/first-order-model/blob/master/data/taichi-loading/README.md

[7] https://sites.google.com/site/metrosmiles/research/research-projects/sound2sight

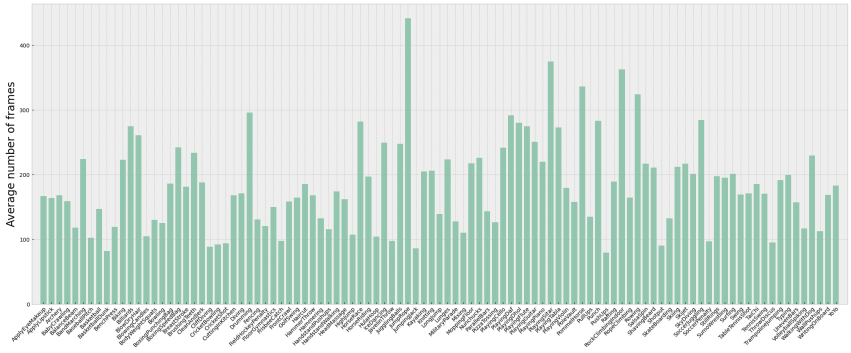


Fig. 16: Average number of frames for each of class in the UCF101 dataset.

IS, we generate 10,000 videos. To calculate the CCS and ICS for long video evaluation, we also generate 512 videos. We run the evaluations for 10 times and report the standard deviations.

B.2 Training and inference details

VQGAN. Suggested by the VQGAN training recipe [16], we start GAN losses after the reconstruction loss generally converges after $10K$ steps. We adopt a codebook with vocabulary size $K = 16,384$ and embedding size $c = 256$. We do not use the random start trick for codebook embeddings [15,81]. Following [73], we set $\lambda_{\text{rec}} = \lambda_{\text{match}} = 4.0$ and $\lambda_{\text{disc}} = 1.0$. We use the ADAM optimizer [34] with $lr = 3e^{-5}$ and $(\beta_1, \beta_2) = (0.5, 0.9)$. As discussed in Section A.1, we always clip the gradients to have euclidean norm = 1. We train the VQGAN on 8 NVIDIA V100 32GB GPUs with batch size = 2 on each gpu and accumulated batches = 6 for $30K$ steps. Each model usually takes around 57 hours to train.

Transformers. Both autoregressive and interpolation transformers contain 24 layers, 16 heads, and embedding size = 1024. We use the AdamW optimizer [39] with a base learning rate of $4.5e^{-6}$, where we linearly scale up by the total batch size. We train the transformers on 8 NVIDIA V100 32GB GPUs with batch size = 3 on each GPU until the training loss saturates. Specifically, we train the autoregressive transformers for $500K$ steps as a general setting except that we train TATS-base on the UCF-101 dataset for $1.35M$ as it keeps improving the results. It takes around 10 days to train the transformer models. In practice, we find that the interpolation setting simplifies the problem dramatically, so we only train the interpolation transformers for $30K$ for the best generalization performance. At the inference time, we adopt sampling strategy where temperature $t = 1$, top-k with $k = 2048$, and top-p with $p = 0.80$ as our general setting (for interpolation transformers, a smaller q and k usually produces better results). Sampling 1 video with 1024 frames takes around 30 minutes using TATS-base on a single Quadro P6000 GPU, while TATS-hierarchical reduces this time to 7.5 minutes for autoregressive transformer and 23 seconds for interpolation transformer.

Baselines Apart from those that have been discussed in the related work section, we provide additional descriptions on the baselines we compared with and other GAN-based video generation models in this section. TGAN [55] proposes to generate a fixed number of latent vectors as input to an image generator to synthesize the corresponding frames. DVD-GAN [12] adopts a similar architecture as MoCoGAN with a focus on scaling up training. TSB [45] further improves MoCoGAN by mixing information of adjacent frames using an operation called temporal shift. TGAN2 [56] proposes to divide the generator into multiple small sub-generators and introduces a subsampling layer that reduces the frame rate between each pair of consecutive sub-generators. HVG [9] further introduces a hierarchical pipeline to interpolate and upsample low-resolution and low-frame-rate videos gradually. ProgressiveVGAN [2] extends ProgressiveGAN [29] to video generation by simultaneously generating in the temporal direction progressively. LDVD-GAN [27] analyzes the discriminators used in video GANs and designs improved discriminators considering the convolution kernel dimensionality.

C Additional results on long video generation

More videos with repeated actions and smooth transitions. Video generation results with repeated actions can be widely seen in other UCF-101 classes as well as shown in Figure 17. In addition to the generated sky videos with smooth transitions, we also show that there are such cases in UCF-101 and Taichi-HD datasets in Figure 18. We can see in the example of UCF-101 *Sky Diving* video that the transition proceeds clearly with unrealistic content as only limited data are available per class. In the example of Taichi video, we can see that the color of the pants transforms smoothly. However, there is still unrealistic content in such videos. Therefore, we argue that a split of content and motion generation could be helpful in these cases [67,82,65].

Failure cases. Errors could occur and accumulate during the application of sliding window attention. We show several typical failure examples in Figure 19. In the example of *Boxing Punching Bag*, an error occurs at around 300 frames, and the general quality of the video deteriorates after that. Repeated tokens are generated in the deteriorated area, which is a commonly observed issue in sequence generation [23,25]. We also find that this kind of issue happens more often in the areas which contain large motions. The video quality could also degrade quickly if the thematic event of the video has a clear end. For example, the generated *Long Jump* video quickly transits to the other scene and degenerates when the person finishes the action.

D Additional related works

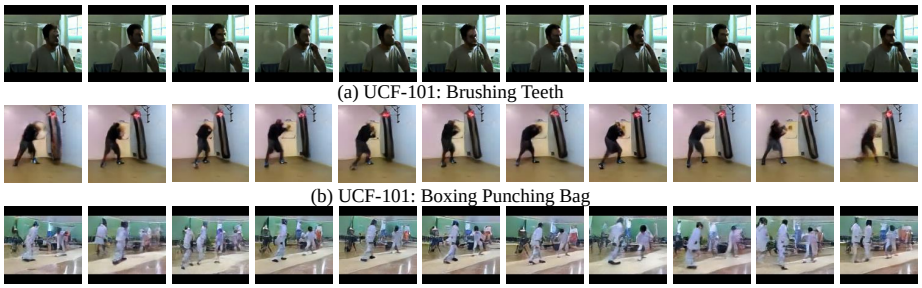
Video prediction. Video prediction aims at modeling the transformation between frames and predicting future frames given real frames [54,63,18,69,35,48].

For instance, [69] divides the frames into patches, calculates the affine transformation between temporally adjacent patches, and predicts such affine transformation to be applied to the most recent frame. [41] predicts videos of semantic maps and argues that autoregressive models lead to error propagation when more frames are predicted while being more accurate in the semantic segmentation space. Flow-based models [35] and ODE-based models [48] have also been used for video prediction. Different from video prediction, video generation focus on producing videos from noise. When applying to long videos, one substantial difference is that video prediction starts from a real frame while video generation starts from a generated frame. Some video generation models fall into the middle part [36] that predicts future frames given frames generated by an image generator. We have shown that these models also suffer from quality degradation.

Conditional video generation. Text-conditioned video generation has been studied in multiple papers. SyncDraw [44] first proposes to combine VAE and RNN for video generation while conditioning on texts. T2V [38] uses CVAE to generate the gist then GANs with 3D convolutions to generate fixed-length low-resolution videos. The text is encoded in a convolutions filter to process the gist. TFGAN [5] proposes a multi-scale text-conditioning discriminator and follows MoCoGAN [67] to use an RNN to model the temporal information. IRC-GAN [14] proposes to use a recurrent transconvolutional generator and mutual-information introspection to generate videos based text. Craft [21] sequentially composes a scene layout and retrieves entities from a video database to create scene videos. GODIVA [76] relies on a pretrained VQVAE model to produce video frame representations and autoregressively predicts the video representations based on the input text representations and former frames. Each element's prediction in the current frame's representation attends to the previous row, column, and time values. As for audio-conditioned video generation, Sound2Sight [10] proposes to model the previous frames and audios with a multi-head audio-visual transformer and predicts future frames with a prediction network based on sequence-to-sequence architecture. Vougioukas et al. [72] studies speech-driven facial animation, which generates face videos given the speech audios. They propose a framework based on RNNs and a frame generator. Some conditional generation frameworks are also able to generate long videos either when minimal changes occur in the video scenes [43] or strong conditional information is given such as segmentation maps [73,42]. Our paper focuses on more realistic and complex videos with weak or no conditional information available.

E Note on data, usage, and figures in the paper

Our models weren't trained on people in close proximity, weren't trained on people's faces, and shouldn't be used for generation of that type of content to mitigate the deep fake policy risk. The human faces in the following images are blurred to protect privacy: Frame 1000 of Figure 1; Frames 8, 32, 56, 64 of Figure 2; all of Figure 7(b); Figure 11 (left image); Figure 18 (several of the frames).

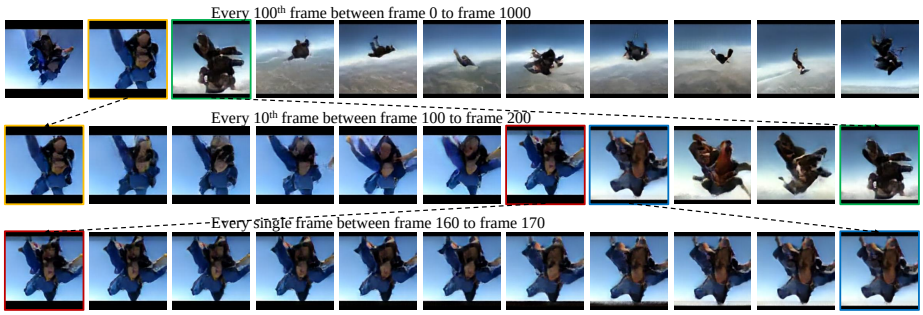


(a) UCF-101: Brushing Teeth

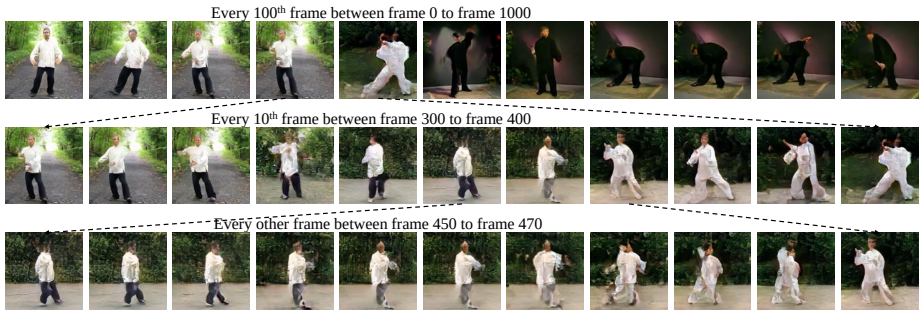
(b) UCF-101: Boxing Punching Bag

(c) UCF-101: Fencing

Fig. 17: More class-conditional generation results of UCF-101 videos with $1,024$ frames that contains repeated action.



(a) UCF-101: Sky Diving



(b) Taichi-HD

Fig. 18: Unconditional and class-conditional generation results of *Sky Diving* and Taichi-HD videos with $1,024$ frames that contain smooth transitions.

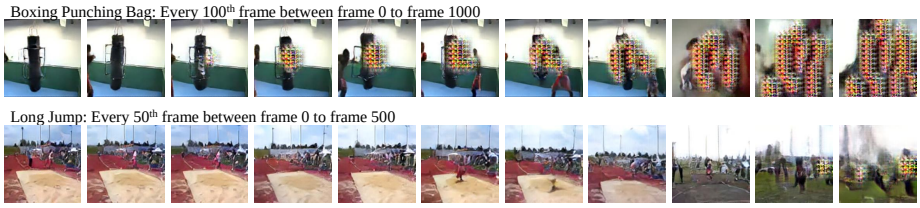


Fig. 19: Failure cases of class-conditional generation results of long videos on the UCF-101 dataset.

References

1. MUGEN: A playground for video-audio-text multimodal understanding and generation. Under Submission (2022) 9, 19
2. Acharya, D., Huang, Z., Paudel, D.P., Van Gool, L.: Towards high resolution video generation with progressive growing of sliced wasserstein gans. arXiv preprint arXiv:1810.02419 (2018) 10, 22
3. Alsallakh, B., Kokhlikyan, N., Miglani, V., Yuan, J., Reblitz-Richardson, O.: Mind the pad – CNNs can develop blind spots. In: ICLR (2021) 6, 14
4. Anokhin, I., Demochkin, K., Khakhulin, T., Sterkin, G., Lempitsky, V., Korzhenkov, D.: Image generators with conditionally-independent pixel synthesis. In: CVPR (2021) 14
5. Balaji, Y., Min, M.R., Bai, B., Chellappa, R., Graf, H.P.: Conditional gan with discriminative filter generation for text-to-video synthesis. In: IJCAI (2019) 23
6. Brock, A., Donahue, J., Simonyan, K.: Large scale gan training for high fidelity natural image synthesis. In: ICLR (2018) 10
7. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. NeurIPS (2020) 2, 5, 6
8. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: CVPR (2017) 7, 11
9. Castrejon, L., Ballas, N., Courville, A.: Hierarchical video generation for complex data. arXiv preprint arXiv:2106.02719 (2021) 8, 14, 22
10. Chatterjee, M., Cherian, A.: Sound2sight: Generating visual dynamics from sound and context. In: ECCV (2020) 8, 9, 20, 23
11. Child, R.: Very deep vaes generalize autoregressive models and can outperform them on images. In: ICLR (2020) 16
12. Clark, A., Donahue, J., Simonyan, K.: Adversarial video generation on complex datasets. arXiv preprint arXiv:1907.06571 (2019) 2, 5, 9, 13, 22
13. Cobbe, K., Klimov, O., Hesse, C., Kim, T., Schulman, J.: Quantifying generalization in reinforcement learning. In: ICML (2019) 20
14. Deng, K., Fei, T., Huang, X., Peng, Y.: Irc-gan: Introspective recurrent convolutional gan for text-to-video generation. In: IJCAI (2019) 23
15. Dhariwal, P., Jun, H., Payne, C., Kim, J.W., Radford, A., Sutskever, I.: Jukebox: A generative model for music. arXiv preprint arXiv:2005.00341 (2020) 21
16. Esser, P., Rombach, R., Ommer, B.: Taming transformers for high-resolution image synthesis. In: CVPR (2021) 2, 3, 4, 5, 8, 10, 15, 16, 17, 21
17. Fan, A., Lewis, M., Dauphin, Y.: Hierarchical neural story generation (2018) 8
18. Finn, C., Goodfellow, I., Levine, S.: Unsupervised learning for physical interaction through video prediction. NeurIPS (2016) 22
19. Gemmeke, J.F., Ellis, D.P., Freedman, D., Jansen, A., Lawrence, W., Moore, R.C., Plakal, M., Ritter, M.: Audio set: An ontology and human-labeled dataset for audio events. In: ICASSP (2017) 4, 9, 19
20. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. NeurIPS (2014) 13
21. Gupta, T., Schwenk, D., Farhadi, A., Hoiem, D., Kembhavi, A.: Imagine this! scripts to compositions to videos. In: ECCV (2018) 23
22. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation (8) (1997) 13

23. Holtzman, A., Buys, J., Du, L., Forbes, M., Choi, Y.: The curious case of neural text degeneration. In: ICLR (2020) 22
24. Islam, M.A., Jia, S., Bruce, N.D.: How much position information do convolutional neural networks encode? In: ICLR (2019) 6, 14
25. Jiang, S., de Rijke, M.: Why are sequence-to-sequence models so dull? understanding the low-diversity problem of chatbots. In: EMNLP Workshop (2018) 22
26. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: ECCV (2016) 5
27. Kahembwe, E., Ramamoorthy, S.: Lower dimensional kernels for video discriminators. Neural Networks (2020) 10, 22
28. Kalchbrenner, N., Oord, A., Simonyan, K., Danihelka, I., Vinyals, O., Graves, A., Kavukcuoglu, K.: Video pixel networks. In: ICML (2017) 2, 14
29. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. In: ICLR (2018) 22
30. Karras, T., Aittala, M., Laine, S., Härkönen, E., Hellsten, J., Lehtinen, J., Aila, T.: Alias-free generative adversarial networks. NeurIPS (2021) 7
31. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: CVPR (2019) 13, 16
32. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: CVPR (2020) 2, 13, 16
33. Kayhan, O.S., Gemert, J.C.v.: On translation invariance in cnns: Convolutional layers can exploit absolute spatial location. In: CVPR (2020) 6, 14
34. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. ICLR (2015) 21
35. Kumar, M., Babaeizadeh, M., Erhan, D., Finn, C., Levine, S., Dinh, L., Kingma, D.: Videoflow: A conditional flow-based model for stochastic video generation. In: ICLR (2019) 22, 23
36. Le Moing, G., Ponce, J., Schmid, C.: Ccvs: Context-aware controllable video synthesis. NeurIPS (2021) 2, 9, 10, 14, 23
37. Lee, K., Chang, H., Jiang, L., Zhang, H., Tu, Z., Liu, C.: Vitgan: Training gans with vision transformers. arXiv preprint arXiv:2107.04589 (2021) 15
38. Li, Y., Min, M., Shen, D., Carlson, D., Carin, L.: Video generation from text. In: AAAI (2018) 23
39. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (2018) 21
40. Luc, P., Clark, A., Dieleman, S., Casas, D.d.L., Doron, Y., Cassirer, A., Simonyan, K.: Transformation-based adversarial video prediction on large-scale data. arXiv preprint arXiv:2003.04035 (2020) 2
41. Luc, P., Neverova, N., Couprie, C., Verbeek, J., LeCun, Y.: Predicting deeper into the future of semantic segmentation. In: ICCV (2017) 23
42. Mallya, A., Wang, T.C., Sapra, K., Liu, M.Y.: World-consistent video-to-video synthesis. In: ECCV (2020) 23
43. Menapace, W., Lathuilière, S., Tulyakov, S., Siarohin, A., Ricci, E.: Playable video generation. In: CVPR (2021) 23
44. Mittal, G., Marwah, T., Balasubramanian, V.N.: Sync-draw: Automatic video generation using deep recurrent attentive architectures. In: MM (2017) 23
45. Munoz, A., Zolfaghari, M., Argus, M., Brox, T.: Temporal shift gan for large scale video generation. In: WACV (2021) 2, 13, 22
46. van den Oord, A., Vinyals, O., Kavukcuoglu, K.: Neural discrete representation learning. In: NeurIPS (2017) 2, 4, 14

47. Oord, A.v.d., Kalchbrenner, N., Vinyals, O., Espeholt, L., Graves, A., Kavukcuoglu, K.: Conditional image generation with pixelcnn decoders. In: NeurIPS (2016) 14
48. Park, S., Kim, K., Lee, J., Choo, J., Lee, J., Kim, S., Choi, Y.: Vid-ode: Continuous-time video generation with neural ordinary differential equation. In: AAAI. AAAI (2021) 22, 23
49. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: ICML (2021) 10
50. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding by generative pre-training 9
51. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners (2019) 9
52. Rakhimov, R., Volkhonskiy, D., Artemov, A., Zorin, D., Burnaev, E.: Latent video transformer. arXiv preprint arXiv:2006.10704 (2020) 2, 14
53. Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., Sutskever, I.: Zero-shot text-to-image generation. arXiv preprint arXiv:2102.12092 (2021) 8
54. Ranzato, M., Szlam, A., Bruna, J., Mathieu, M., Collobert, R., Chopra, S.: Video (language) modeling: a baseline for generative models of natural videos. arXiv preprint arXiv:1412.6604 (2014) 14, 22
55. Saito, M., Matsumoto, E., Saito, S.: Temporal generative adversarial nets with singular value clipping. In: ICCV (2017) 2, 10, 13, 22
56. Saito, M., Saito, S., Koyama, M., Kobayashi, S.: Train sparsely, generate densely: Memory-efficient unsupervised training of high-resolution temporal gan. IJCV (2020) 2, 9, 11, 13, 22
57. Siarohin, A., Lathuilière, S., Tulyakov, S., Ricci, E., Sebe, N.: First order motion model for image animation. NeurIPS (2019) 4, 8, 19
58. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014) 5
59. Sitzmann, V., Martel, J., Bergman, A., Lindell, D., Wetzstein, G.: Implicit neural representations with periodic activation functions. NeurIPS (2020) 14
60. Skorokhodov, I., Ignatyev, S., Elhoseiny, M.: Adversarial generation of continuous images. In: CVPR (2021) 14
61. Skorokhodov, I., Tulyakov, S., Elhoseiny, M.: Stylegan-v: A continuous video generator with the price, image quality and perks of stylegan2. arXiv preprint arXiv:2112.14683 (2021) 2, 7, 14
62. Soomro, K., Zamir, A.R., Shah, M.: Ucf101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 (2012) 4, 8, 19
63. Srivastava, N., Mansimov, E., Salakhudinov, R.: Unsupervised learning of video representations using lstms. In: ICML (2015) 14, 22
64. Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., Ng, R.: Fourier features let networks learn high frequency functions in low dimensional domains. NeurIPS (2020) 14
65. Tian, Y., Ren, J., Chai, M., Olszewski, K., Peng, X., Metaxas, D.N., Tulyakov, S.: A good image generator is what you need for high-resolution video synthesis. In: ICLR (2021) 2, 9, 10, 13, 22
66. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: ICCV (2015) 9
67. Tulyakov, S., Liu, M.Y., Yang, X., Kautz, J.: Mocogan: Decomposing motion and content for video generation. In: CVPR (June 2018) 2, 5, 13, 22, 23

68. Unterthiner, T., van Steenkiste, S., Kurach, K., Marinier, R., Michalski, M., Gelly, S.: Towards accurate generative models of video: A new metric & challenges. ICLR (2019) 9
69. Van Amersfoort, J., Kannan, A., Ranzato, M., Szlam, A., Tran, D., Chintala, S.: Transformation-based models of video sequences. arXiv preprint arXiv:1701.08435 (2017) 22, 23
70. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017) 14
71. Vondrick, C., Pirsiavash, H., Torralba, A.: Generating videos with scene dynamics. NeurIPS (2016) 2, 13
72. Vougioukas, K., Petridis, S., Pantic, M.: End-to-end speech-driven facial animation with temporal gans. BMVC (2018) 23
73. Wang, T.C., Liu, M.Y., Zhu, J.Y., Liu, G., Tao, A., Kautz, J., Catanzaro, B.: Video-to-video synthesis. In: NeurIPS (2018) 5, 21, 23
74. Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-resolution image synthesis and semantic manipulation with conditional gans. In: CVPR (2018) 5
75. Weissenborn, D., Täckström, O., Uszkoreit, J.: Scaling autoregressive video models. In: ICLR (2020) 2, 14
76. Wu, C., Huang, L., Zhang, Q., Li, B., Ji, L., Yang, F., Sapiro, G., Duan, N.: Godiva: Generating open-domain videos from natural descriptions. arXiv preprint arXiv:2104.14806 (2021) 8, 23
77. Wu, C., Liang, J., Ji, L., Yang, F., Fang, Y., Jiang, D., Duan, N.: N\" uwa: Visual synthesis pre-training for neural visual world creation. arXiv preprint arXiv:2111.12417 (2021) 2, 14
78. Wu, Y., He, K.: Group normalization. In: ECCV (2018) 16
79. Xiong, W., Luo, W., Ma, L., Liu, W., Luo, J.: Learning to generate time-lapse videos using multi-stage dynamic generative adversarial networks. In: CVPR (2018) 4, 8, 19, 20
80. Xu, R., Wang, X., Chen, K., Zhou, B., Loy, C.C.: Positional encoding as spatial inductive bias in gans. In: CVPR (2021) 6, 14
81. Yan, W., Zhang, Y., Abbeel, P., Srinivas, A.: Videogpt: Video generation using vq-vae and transformers. arXiv preprint arXiv:2104.10157 (2021) 2, 5, 14, 15, 21
82. Yu, S., Tack, J., Mo, S., Kim, H., Kim, J., Ha, J.W., Shin, J.: Generating videos with dynamics-aware implicit generative adversarial networks. In: ICLR (2021) 2, 9, 10, 11, 13, 14, 20, 22
83. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018) 5, 12