PHENAKI: VARIABLE LENGTH VIDEO GENERATION FROM OPEN DOMAIN TEXTUAL DESCRIPTIONS

Ruben Villegas[‡]
Google Brain
rubville@google.com

Mohammad Babaeizadeh[‡]
Google Brain
mbz@google.com

Pieter-Jan Kindermans[‡]
Google Brain
pikinder@google.com

Hernan Moraldo
Google Brain
hmoraldo@google.com

Han Zhang
Google Brain
zhanghan@google.com

Mohammad Taghi Saffar
Google Brain
msaffar@google.com

Santiago Castro[*]
University of Michigan
sacastro@umich.edu

Julius Kunze[*]
University College London
kjulius@google.com

Dumitru Erhan
Google Brain
dumitru@google.com

ABSTRACT

We present Phenaki, a model capable of realistic video synthesis, given a sequence of textual prompts. Generating videos from text is particularly challenging due to the computational cost, limited quantities of high quality text-video data and variable length of videos. To address these issues, we introduce a new model for learning video representation which compresses the video to a small representation of discrete tokens. This tokenizer uses causal attention in time, which allows it to work with variable-length videos. To generate video tokens from text we are using a bidirectional masked transformer conditioned on pre-computed text tokens. The generated video tokens are subsequently de-tokenized to create the actual video. To address data issues, we demonstrate how joint training on a large corpus of image-text pairs as well as a smaller number of video-text examples can result in generalization beyond what is available in the video datasets. Compared to the previous video generation methods, Phenaki can generate arbitrary long videos conditioned on a sequence of prompts (i.e. time variable text or *a story*) in open domain. To the best of our knowledge, this is the first time a paper studies generating videos from time variable prompts. In addition, compared to the per-frame baselines, the proposed video encoder-decoder computes fewer tokens per video but results in better spatio-temporal consistency.

1 INTRODUCTION

It is now possible to generate realistic high resolution images given a description [34, 35, 32, 38, 59], but generating high quality videos from text remains challenging. In essence, videos are just a sequence of images, but this does not mean that generating a long coherent video is easy. In practice, it is a significantly harder task because there is much less high quality data available and the computational requirements are much more severe [9]. For image generation, there are datasets with billions of image-text pairs (such as LAION-5B [41] and JFT4B [60]) while the text-video datasets are substantially smaller e.g. WebVid [4] with $\sim$10M videos, which is not enough given the higher complexity of open domain videos. As for computation, training current state-of-the-art image generation models is already pushing the state-of-the-art computational capabilities [59], leaving little to no room for generating videos, particularly videos of variable length.

To make the matters worse, one can argue that a single short text prompt is not sufficient to provide a complete description of a video (except for short clips), and instead, a generated video must be conditioned on a sequence of prompts, or *a story*, which narrates what happens over time. Ideally,

[‡]Equal contribution. * Intern at Google Brain while working on this project.

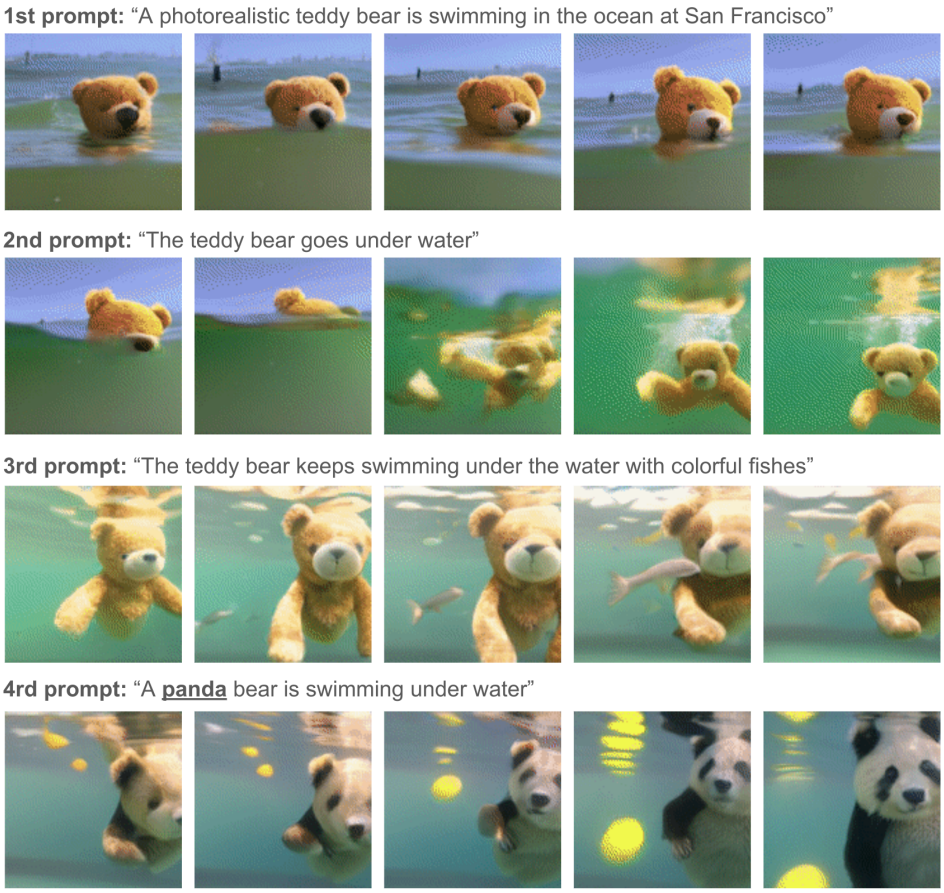

Figure 1. Time variable text (i.e. story) conditional video generation. The entire figure is **one continuous video** generated auto-regressively. We start by generating the video conditioned on the first prompt and then after a couple of frames we change the prompt to the next one. Each row contains a selected number of frames (from left to right in order) while the model was conditioned on that particular prompt. The model manages to preserve the temporal coherence of the video while adapting to the new prompt, usually taking the shortest path for the adaption (notice the *morphing* of the teddy bear to the panda). Please note that the generated video has complex visual features such as reflections, occlusions, interactions and scene transitions. Full video is available at phenaki.github.io.

a video generation model must be able to generate videos of arbitrary length, all the while having the capability of conditioning the generated frames at time t on prompts at time t that can vary over time. Such capability can clearly distinguish the *video* from a "moving image" and open up the way to real-world creative applications in art, design and content creation. To the best our knowledge, story based conditional video generation has never been explored before and this is the first paper to take early steps towards that goal. A traditional deep learning approach of simply learning this task from data is not possible, since there is no story-based dataset to learn from. Instead, to achieve this we rely on a model that is designed specifically with this capability in mind.

In this paper, we introduce Phenaki, a text to video model trained on both text to video and text to image data that can:

– Generate temporally coherent and diverse videos conditioned on open domain prompts even when the prompt is a new composition of concepts (Fig. 3). The videos can be long (minutes) even though the model is trained on 1.4 seconds videos (at 8 fps).

– Generate videos conditioned on a story (i.e. a sequence of prompts), e.g. Fig. 1 and Fig. 5.

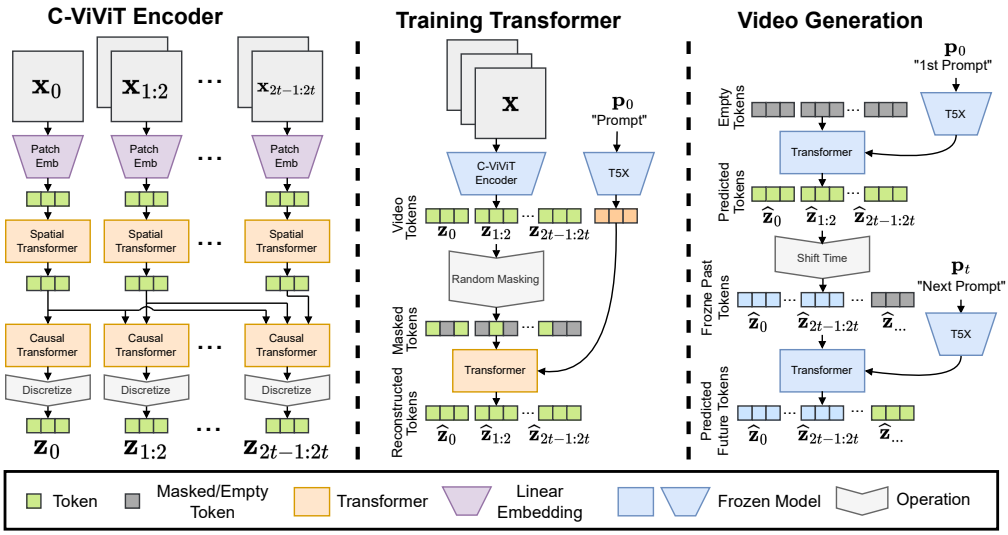

Figure 2. The architecture of Phenaki. **Left:** C-ViViT encoder architecture. The embeddings of images and video patches from raw frames $\mathbf{x}$ are processed by a spatial and then a causal transformer (auto-regressive in time) to generate video tokens $\mathbf{z}$. **Center:** MaskGiT is trained to reconstruct masked tokens $\mathbf{z}$ predicted by a frozen C-ViViT encoder and conditioned on T5X tokens of a given prompt $\mathbf{p}_0$. **Right:** How Phenaki can generate arbitrary long videos by freezing the *past* token and generating the future tokens. The prompt can change over time to enable time-variable prompt (i.e. story) conditional generation. The subscripts represent time (i.e. frame number).

To enable these capabilities, we could not rely on current video encoders, because they either can only decode fixed size videos or they encode frames independently. Hence, we introduce C-ViViT , a novel encoder-decoder architecture that:

– Exploits temporal redundancy in videos to improve reconstruction quality over a per frame model while compressing the number of video tokens by 40% or more.

– Allows encoding and decoding of variable length videos given its causal structure.

2 The Phenaki Model

Inspired by the previous work in auto-regressive text to image [34, 59, 38] and text to video [54, 53, 18], Phenaki is designed with two main components (see Figure 2): an encoder-decoder model which compresses videos to discrete embeddings (i.e. tokens) and a transformer model to *translate* text embeddings to video tokens. To get the text embeddings, Phenaki uses a pre-trained language model, T5X [37]. We will discuss each one of these components in the following subsections.

2.1 Encoder-decoder video model: C-ViViT

One of the primary challenges for generating video from text, is to get a compressed representation of videos. Previous work on text to video either use per-frame image encoders [18, 54, 57] such as VQ-GAN [12] or fixed length video encoders [52] such as VideoVQVAE [49]. The former allows for generating videos of arbitrary length, however in practice, the videos have to be short because the encoder does not compress the videos in time and the tokens are highly redundant in consecutive frames. The latter is more efficient in the number of tokens but it does not allow to generate variable length videos. In Phenaki, our goal is to generate videos of variable length while keeping the number of video tokens to a minimum so they can be modeled with a transformer within current computational limitations. To do so, we introduce C-ViViT , a causal variation of ViViT [1] with additional architectural changes for video generation, which can compress the videos in temporal and spatial dimensions, while staying auto-regressive in time, This capability allows for generating videos of arbitrary length auto-regressively.

Encoder architecture: As illustrated in Figure 2, we start with a video sequence of $t_x + 1$ frames with a resolution of $w_x \times h_x$ and c_x channels: $\mathbf{x} \in \mathbb{R}^{(t_x+1)\times h_x \times w_x \times c_x}$. This sequence will be compressed into a token representation of size $(t_z + 1) \times w_z \times h_z$ where the first $w_z \times h_z$ tokens represent the first frame independently from the rest of the video, and the remaining tokens represent spatio-temporal video tokens that auto-regressively depend on previous frames. To do so, we extract non-overlapping image patches of size $w_p \times h_p \times c_p$ from the first frame and video patches of size $t_p \times w_p \times h_p \times c_p$ from the rest of the video. We typically use all channels at once such that the number of patches equals the number of video tokens $t_z = \frac{t_x}{t_p}$, $w_z = \frac{w_x}{w_p}$ and $h_z = \frac{h_x}{h_p}$. Each of these patches is flattened and linearly projected into a d_z dimensional space. We combine the spatial dimensions to have a tensor of shape $(t_z+1)\times w_z * h_z \times d_z$ where the spatial and temporal dimensions are separated. Then multiple transformer layers are applied along the spatial dimensions with all-to-all attention. This is followed by multiple transformer layers over the temporal dimension with causal attention such that each spatial token only observes spatial tokens from previous frames in an auto-regressive manner. The effect of this is that the first frame can be completely independently encoded. This opens up the possibility of text to image training to be embedded naturally into our video model. The second advantage is that we can condition the video generation process on a number of starting frames. The resulting patch embeddings $\mathbf{z}$ of shape $t_z \times w_z \times h_z \times d_z$ are then tokenized into learned codewords c_z by vector quantization. The codebook learning will be discussed later together with the losses.

Decoder architecture: The C-ViViT decoder is simply an upside down version of the encoder. First tokens are transformed into embeddings. This is followed by the temporal transformer, then the spatial transformer. After the output of the spatial transformer, we apply a single linear projection without activation to map the tokens back to pixel space.

Quantization and Losses: To learn a discrete latent space, we quantize our encoder outputs into the entries of a learned codebook via the vector quantization (VQ) objective in VQVAEs [45],

$$L_{VQ} = \|\text{sg}(\mathbf{z}) - \mathbf{e}\|_2^2 + \beta\|\mathbf{z} - sg(\mathbf{e})\|_2^2, \tag{1}$$

where $\text{sg}(x) \equiv x$, and $\frac{\text{d}}{\text{d}x}\text{sg}(x) \equiv 0$ is the stop-gradient operator, β is the commitment loss weight, and $\mathbf{e}$ is a codebook vector from codebook $\mathbf{E}$. The index to the codebook vector closest to $\mathbf{z}$ is found by $i = \text{argmin}_j \|\mathbf{z} - \mathbf{E}_j\|_2^2$. In addition to the VQ objective, we adopt the factorized and ℓ_2-normalized codes from ViT-VQGAN [58] to improve codebook usage and reconstruction quality.

To train our model, we use a combination of L_2 loss, image perceptual loss L_{IP} [20, 61], video perceptual loss L_{VP} by using the I3D network [6] as feature extractor, and adversarial loss L_{Adv} with StyleGAN architecture [21]. As training objective, we use the following

$$L = L_{VQ} + 0.1 \times L_{Adv} + 0.1 \times L_{IP} + 1.0 \times L_{VP} + 1.0 \times L_2. \tag{2}$$

Novelty over the ViViT architecture: While our proposed C-ViViT architecture is inspired by the factorized encoder in ViViT [1], we modify their architecture to enable self-supervised learning from unlabeled videos. We first remove the `[CLS]` tokens in the spatial and the temporal transformers. Next, we apply temporal transformer for all spatial tokens computed by the spatial encoder, in contrast to single run of the temporal transformer over the `[CLS]` tokens in ViViT. Most importantly, the ViViT encoder requires a fixed length video input due to the all-to-all attention in time. Therefore, we apply causal attention instead such that our C-ViViT encoder becomes auto-regressive and allows for a variable number of input frames which are necessary to learn from image datasets, and auto-regressively extrapolate video or single frames into the future.

2.2 TEXT-TO-VIDEO GENERATION WITH BIDIRECTIONAL TRANSFORMERS

In this stage, the text-to-video task can be formulated as a sequence-to-sequence problem to predict video tokens given the paired text embeddings. Most of recent methods [34, 59, 54, 18] adopt a transformer model for these sequence-to-sequence tasks. In their models, they use an auto-regressive transformer which predicts the image or video tokens sequentially given the encoded text features. As a result, the sampling time scales linearly with the sequence length, even when caching is used. This becomes impractical for long video sequence generation.

Masked bidirectional transformer: In this work, we aim to reduce the sampling time by having a small and fixed sampling step disregarding different video sequence lengths. Inspired by previous work for image generation [8], we use a bidirectional transformer since it can predict different video tokens simultaneously. For training step i, we first sample a mask ratio γ_i from 0 to 1 and randomly replace $\lceil \gamma_i \cdot N \rceil$ tokens with the special token `[MASK]`, where N is the video sequence length. Then we learn the model parameters by minimizing the cross entropy loss on those masked tokens given the encoded text embeddings and unmasked video tokens. During inference, we first label all of the video tokens as the special token `[MASK]`. Then, at each inference step, we predict all the masked (unknown) video tokens in parallel conditioned on the text embeddings and unmasked (predicted) video tokens. We keep a ratio β_i of the predicted tokens at sampling step i and the remaining tokens are re-masked and re-predicted in the next step.

As discussed in MaskGIT [8], the masking schedule γ_i and sampling schedule β_i have a significant effect on the samples quality therefore we follow the same strategies. Compared to an auto-regressive transformer, the number of sampling steps is an order-of-magnitude smaller (typically we use values in the range of 12 to 48). Generally speaking, more sampling steps improves the quality.

Losses and training strategies: Given a pre-trained C-ViViT , videos are encoded into codebook ids $\mathbf{a}$ of shape $(t_z + 1) \times w_z \times h_z$ which are flattened into a long vector using the raster ordering from [58]. We then model the text-conditional video token distribution using *Masked Visual Token Modeling* (MVTM) [8]:

$$L_{\text{mask}} = -\sum\nolimits_{\forall i \in [1,N], m_i = 1} \log p(a_i | \mathbf{a}_{\bar{M}}, \mathbf{p}), \tag{3}$$

where $\mathbf{a}_{\bar{M}}$ represents the masked version of $\mathbf{a}$, m_i is a binary variable indicating whether a_i is masked or not, N is the number of video tokens, and $\mathbf{p}$ is the text condition embedding. In addition to the MVTM objective, we train using classifier-free guidance by dropping the text condition 10% of the time during training [16, 59] . Finally, we dynamically adjust the MVTM objective during training to allow the use of image and video datasets as a single large dataset. We achieve this by only applying the masking ratio and objective on the first $w_z \times h_z$ tokens if only a single frame is given or over all video tokens if a full video is given. This mixed image and video dataset training strategy allows our models to learn concepts only present in image datasets, and transfer them to concepts present video datasets (e.g., the pencil drawing styled video of the panda in Figure.3).

Inference and auto-regressive generation of long videos: At inference time, we sample videos tokens by the same iterative process used in [8] with classifier-free guidance scale λ to control alignment between the generation and the text condition. Once the first video is generated, we can extrapolate additional frames auto-regressively by encoding the last K generated frames in the last video using C-ViViT , initializing MaskGIT with the tokens computed by our C-ViViT encoder, and proceed to generate the remaining video tokens conditioned on a text input. During video extrapolation, the text condition can be the same or a different one which enables our model to dynamically create visual transitions between the previous and current text condition visual content, effective generating a visual story an described by the input text.

3 EXPERIMENTS

To evaluate Phenaki, we test it on the following tasks: 1) text conditional video generation, 2) text-image conditional video generation, 3) time variable text conditional video generation (i.e.) story mode, 4) video quantization and 5) image conditional video generation a.k.a. video prediction. To the best of our knowledge, 3) time variable text conditional video generation has not been explored in prior work. Given the dynamic nature of videos, we highly encourage readers to visit phenaki.github.io to check the generated videos. The website also includes qualitative comparisons to a subset of the prompts from the CogVideo paper [18]. While the focus is on the text to video generation tasks, it is remarkable that Phenaki is still competitive on the more traditional video tasks despite not being developed explicitly for these tasks. We implemented Phenaki in JAX [**?**] using FLAX [**?**] library.

Table 1. Text to video comparisons on Kinetics-400 [22].

Method	FID Image ↓	FID Video ↓
T2V [25]	82.13	14.65
SC [5]	33.51	7.34
TFGAN [5]	31.76	7.19
NUWA	28.46	7.05
Phenaki [0-Shot]	37.74	3.84

Table 2. Text to video and text to image results highlighting the importance of image datasets in video models. Text-to-image evaluation is done on ∼40K images of LAION-400M [41].

Data Split	Text to Video			Text to Image	
Vid% / Img%	CLIP ↑	FID ↓	FVD ↓	CLIP ↑	FID ↓
100% / 0%	0.298	19.2	168.9	0.240	53.9
80% / 20%	0.303	21.4	198.4	0.289	29.4
50% / 50%	0.302	21.4	239.7	0.287	30.5

3.1 TEXT CONDITIONAL VIDEO GENERATION

Currently there is no established benchmark for evaluating text to video methods. This makes comparing Phenaki to recent methods such as NUWA [54], CogVideo [18], NUWA-Infinity [53] and video diffusion models [17] difficult.

Unless specified otherwise, we train a 1.8B parameter Phenaki model on a corpus of ∼15M text-video pairs at 8 FPS mixed with ∼50M text-images plus ∼400M pairs of LAION-400M [41] (more details in Appendix B.3). The model used in the visualisations in this paper was trained for 1 million steps at a batch size of 512, which took less than 5 days. In this setup 80% of the training data came from the video dataset and each image dataset contributed 10%.

Qualitative evaluation: Samples from this model can be seen in Figure 3 and additional samples are provided at phenaki.github.io. We observe that there is a high degree of control over both the actors and the background dynamics in the videos. The appearance of the actors and the video style can be adjusted by the text prompt as well (e.g. a regular video, a cartoon or a pencil drawing).

On phenaki.github.io we provide examples from prompts that were provided in the CogVideo [18] demo. Since there are substantial differences between these methods it is hard to compare them on an equal footing. As an example, there are massive differences in scale: 9B parameters for CogVideo and 1.8B for our model. Additionally, the training data is different. Finally, we do not know how representative the prompts in the CogVideo demo are for the general performance of the CogVideo.

Quantative comparison: The NUWA [54] paper provided a qualitative evaluation on Kinetics-400. Since the NUWA model is only 0.9B parameters we also use a model of the same size. Our model was trained on 50% video and 50% image data in this experiment. The NUWA model finetuned on Kinetics but the Phenaki model is not: it is evaluated in a *zero shot setting*. The results in Table 1 show that Phenaki achieves comparable generation quality, in a zero-shot setting, compared to previous text to video methods that were actually trained or finetuned on this dataset.

On the importance of joint text-to-image and text-to-video training While there are some text-video datasets, text-image datasets dominate the internet in terms of quality and quantity [30]. Consequently, there is simply not enough video data available to cover all the concepts present in text-image datasets. For example using only our video data, concepts such as pencil drawings or different painting styles cannot be learned. To be able to learn a model that can combine video dynamics with these additional concepts we have to combine training on image and video data. In Table 2, we evaluate the performance of using different ratios of video and images. We start with data splits of only video, and vary the ratio of image and video datasets up to using 50% image and 50% video datasets. In our results, we find that there is a trade-off in performance between models trained with only video video (i.e., significantly better FVD), and models trained with more image data (i.e., better text-video and text-image alignment, and significantly better FID in image datasets). On phenaki.github.io we show samples from different models side by side where this trade-off between control over the content and the quality of the dynamics can be seen. We believe that the trade-off between concepts and dynamics will be improved as the quality and size of text-video datasets increases in the future.

3.2 TEXT-IMAGE CONDITIONAL VIDEO GENERATION

Given that Phenaki can be conditioned on both still images and text, an interesting setup is to *animate* existing images given a text prompt. For this experiment, we use the same model from Section 3.1 but conditioned on unseen pictures (captured with our phones from local subjects) and a related prompt. As it can be seen in Figure 4 the model can generate coherent videos starting from the given images, while following the given prompts.

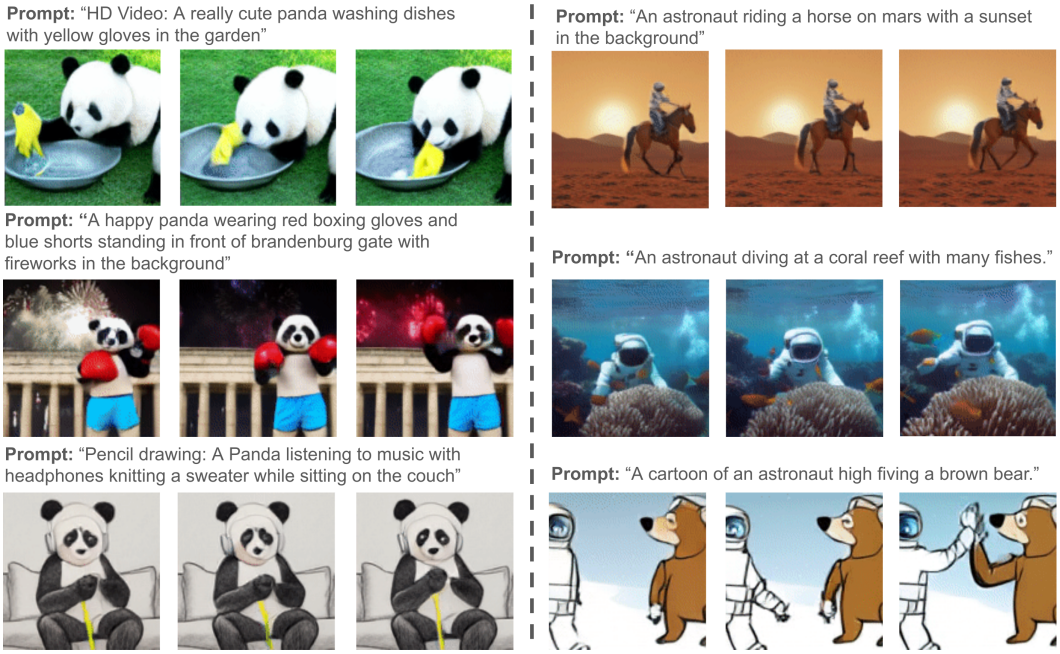

Figure 3. Text conditional video generation. Each row shows selected frames from a video generated given the prompt. The model is trained on a mix of images and videos. The video dataset does not include any *stylized* videos such as pencil drawings, however, the image dataset does. The model can generalize from still images to videos. This figure also demonstrate the capability of the model in generating new unseen compositions. Full videos are available at phenaki.github.io.

3.3 VISUAL STORY TELLING BY DYNAMIC TEXT INPUTS

A notable and useful feature of Phenaki is that it is auto-regressive in time. This allows for generating long videos, while the prompt changes over time. Time variable prompts can be thought of as a *story*; a narration of the entire video where each prompt corresponds to a scene from the video. This allows for creating dynamically changing scenes. To the best our knowledge, this paper is the first work to generate such videos. An example of this can be seen in Fig. 1 and on phenaki.github.io. The way it works is that we generate a video with the first prompt and then extend it in time by conditioning a possibly new prompt and on the last N, typically 5, previously generated frames.

3.4 VIDEO ENCODING

To evaluate the video encoding and reconstruction performance of C-ViViT , we use the Moments-in-Time (MiT) [29] dataset. MiT contains $\sim$802K training, $\sim$33K validation and $\sim$67K test videos at 25 FPS. The MiT dataset, in contrast to other publicly available video datasets, is a high quality balanced dataset with high coverage and density of verbs depicting moments of a few seconds [29]. We compare C-ViViT against per-frame image based encoder-decoders that have been used as video quantizers for conditional video generation [57, 54, 18, 54, 18, 52]: a ViT [58] and a convolutional VQ-GAN[12]. The experimental details can be found in the Appendix B.1.

Figure 4. Animating images conditioned on a prompt. Each row demonstrates multiple frames of a generated video conditioned on a given first frame as well as a given text prompt. The first frames are new (captured by author's phone) and not observed during the training. The model *animates* the given image while following the prompt. Full videos are available at phenaki.github.io.

Table 3. Video reconstruction results on Moments-in-Time. The number of tokens is computed for 10 frames with the exception of C-ViViT which is for 11, due to the isolated initial frame.

Method	FID ↓	FVD ↓	Number of Tokens ↓
Conv VQ-GAN [12]	7.5	306.1	2560
Conv VQ-GAN + Video loss	13.7	346.5	2560
ViT VQ-GAN [58]	3.4	166.6	2560
ViT VQ-GAN + Video loss	3.8	173.1	2560
C-ViViT VQ-GAN (Ours)	4.5	65.78	1536

As demonstrated in Table 3, we evaluate the video reconstruction quality using FID [15] and FVD [44]. Both FID and FVD compare the distribution of generated videos (or images) to the ground truth distribution. The FID ignores temporal coherency, while the FVD measures how well the spatio-temporal dynamics of the videos are reconstructed. Results in Table 3 show that per-frame image based methods slightly outperform our video method (indicated by marginally higher FID of C-ViViT), however, they do poorly at modeling the spatio-temporal dynamics in video (significantly lower FVD of C-ViViT). This is expected as C-ViViT has spatio-temporal connections between patches in each frame, allowing space and time to be modeled together. In addition, C-ViViT compresses the video into fewer tokens per video compared to the image based baselines. This is crucial as the number of tokens drastically impacts the computational cost of the transformer in downstream tasks. Furthermore, C-ViViT tokens are auto-regressive in time which enables variable length videos to be modeled with the same encoder which is important for video extrapolation conditioned on previously generated frames.

3.5 Image conditional video generation a.k.a Video prediction

To evaluate the learnt video representation of C-ViViT beyond reconstruction, we test it on the task of frame-conditioned video generation, also commonly known as video prediction [3]. In this experiment, we test Phenaki on BAIR Robot Pushing benchmark [11] where the task is to generate 15 frames conditioned on a given single frame. For open domain videos, we test Phenaki on Kinetics-600 [7] where the task is to predict 11 frames given 5 frames. More details about these experiments can be found in Appendix B.2. Tables 4 and 5 show the results of these experiments. Note that

Table 4. Video prediction on Kinetics-600 [7]. While Phenaki is not designed for video prediction it achieves comparable results with SOTA video prediction models.

Method	FVD ↓
Video Transformer [51]	170.0 ± 5.00
CogVideo [18]	109.2
DVD-GAN-FP [9]	69.1 ± 0.78
Video VQ-VAE [49]	64.3 ± 2.04
CCVS [28]	55.0 ± 1.00
TrIVD-GAN-FP [27]	25.7 ± 0.66
Transframer [31]	25.4
RaMViD [19]	16.5
Video Diffusion [17]	16.2 ± 0.34
Phenaki (Ours)	36.4 ± 0.19

Table 5. Video prediction on BAIR [11].

Method	FVD ↓
DVD-GAN [9]	109.8
VideoGPT [55]	103.3
TrIVD-GAN [27]	103.3
Transframer [31]	100.0
HARP [57]	99.3
CCVS [28]	99.0
Video Transformer [51]	94.0
FitVid [3]	93.6
MCVD [47]	89.5
NUWA [54]	86.9
RaMViD [19]	84.2
Phenaki (Ours)	97.0

Phenaki is not specifically designed for video prediction, therefore, it lacks components such as skip connections in U-Nets which are known to improve the performance for video prediction methods [10, 46, 3]. Nevertheless, our method is competitive on these benchmarks with SOTA video prediction methods. Overall, these experiments show that Phenaki is strong at modeling dynamics of the videos which is required for generating coherent videos from text.

4 Related Works

This paper is closely related to auto-regressive methods for text conditioned image and video generation. DALL-E [34] *translates* text tokens to discrete image embeddings learnt using a VQVAE [45]. Parti [59] has a similar architecture but can generate higher quality images by predicting tokens from a ViT-VQGAN [58] using a 21B parameters transformer. Similar architectures have been used for generating videos as well. GODIVA [52] uses a transformer to map text tokens to video tokens from a image based VQVAE. Given the large number of tokens from multiple frames, GODIVA relied on a local-attention mechanism. Similarly, NUWA [54] and NUWA-Infinity [53] both employ auto-regressive architectures to generate videos and images from text. NUWA generates fixed size outputs, while NUWA-Infinity introduces a second layer of auto-regressive computation to support variable size videos. Likewise, CogVideo [18] argues the main reason behind low quality video generation is the scarcity of good text-video data and tried to leverage pre-trained text to images models to generate high quality video.

While Phenaki sticks to the same architecture principles, it has major differences with previous work. Most notably, NUWA, NUWA-Infinity and CogVideo treat videos as a sequence of independent images. This can lead to poor modeling of dynamics and generate motion artifacts. To combat this, NUWA-infinity used the previous frame during decoding to combat this. In Phenaki, we go further and treat videos as a temporal sequence of images which substantially decreases the number of video tokens given the redundancy in video generation, and results in a much lower training cost. The auto-regressive nature of the Phenaki also allows us to effectively condition on previous frames and generates longer videos as detailed in Section 2.

Diffusion models are another class of models which recently have been used for conditional and unconditional video generation, which we call VDM [17]. In VDM, authors proposed replacing the conventional U-Net architectures for 2D image modeling with a 3D space-time model to run the diffusion process directly on pixels. While this approach provides an effective formulation for modeling videos, it is limited to fixed size videos. To address this issue, VDM provides an auto-regressive extension, which allows the model to generate longer videos but it is typically impractical due to high sampling time of diffusion models.

Text conditional video generation is a relatively new field of research, nonetheless, image conditional video generation, commonly known as video prediction, and unconditional video generation have been studied more comprehensively. These papers include deterministic methods using a combination of recurrent and convolutional networks [36, 42, 13, 50], variational based stochastic methods [2, 10, 46, 3] and more recently by learning a discrete representation [49, 33, 31], auto-regressive models [51, 55, 28, 57], diffusion models [47, 14, 56, 19] flow based models [24], and finally adversarial based methods [48, 39, 43, 9, 40, 27]. These works mostly consider limited domain (e.g.

robotic videos) prediction/generation, or short fixed size clips. Section 3 provides comparison with some of these models.

5 Conclusion

We introduced Phenaki, a model which is capable of generating variable length videos conditioned on a sequence of open domain text prompts. Phenaki uses C-ViViT as video encoder. C-ViViT is a new model which provides temporal-spatial compression while being auto-regressive in time. The C-ViViT model is a crucial part of Phenaki that allows it to generate variable length videos. We demonstrate how joint training on images and videos can improve the generation quality, and diversity, given the existence of much larger image-text dataset with order of magnitude more samples. The Phenaki model achieves good performance on video prediction, it can be used as to generate long videos conditioned on a text prompt. Additionally it is able to condition on both text and a starting frame. Finally, Phenaki is not limited to generating a video depicting a single concept or caption. It is actually able to generate longer coherent video stories based on a sequence of text prompts. The more complex narratives it can visualize demonstrate how this can become a great creative tool for story telling.

Ethics Statement

While we have not explored potential downstream applications of the generative models described in this work, we believe Phenaki can have a positive impact in a variety of creative settings. In general, many of the samples from the model will not perfectly correspond to the input caption or the user's intent; however, the end-user is likely to gain considerable time savings even if only one of the generated samples aligns with their intent. We thus foresee Phenaki being useful in eventually empowering users to accelerate their creativity, especially since the model can so quickly generate videos. Phenaki and similar models will be part of an ever-broad toolset for artists and non-artists alike, providing new and exciting ways to express creativity.

The flip-side of this acceleration and ease-of-use is the potential for harmful impact, as with many of the prior or concurrent work in generative modeling. An easy-to-use system like Phenaki can be repurposed for generating maliciously fake content and enable spreading of such content much easier. While the quality of the videos generated by Phenaki is not yet indistinguishable from real videos, getting to that bar for a specific set of samples is within the realm of possibility, even today. This can be particularly harmful if Phenaki is to be used to generate videos of someone without their consent and knowledge.

Like DALLE-2 [35], Imagen [38], Parti [59] and others, Phenaki is trained on a collection of datasets that is known to encode a number of undesirable biases. LAION-400M [41] specifically has a variety of issues regarding violence, pornography, gore. While our primary image and video datasets have minimal traits like this, we did incorporate LAION-400M into our training and observed better results. In a currently training version of Phenaki, we use a set of datasets that minimizes such problems.

Taken together, these issues contribute to our decision not to release the underlying models, code, data or interactive demo at this time. Before we can do that, we want to focus our efforts on better understanding of data, prompt and output filtering. We would also like to more explicitly measure the biases encoded in the outputs of Phenaki, so that we can further mitigate them actively, either in the data, models or pre/post-processing steps.

Acknowledgments

We would like to thank Niki Parmar for initial discussions. Special thanks to Gabriel Bender and Thang Luong for reviewing the paper and providing constructive feedback. We appreciate the efforts of Kevin Murphy and David Fleet for advising the project and providing feedback throughout. We are grateful to Evan Rapoport, Douglas Eck and Zoubin Ghahramani for supporting this work in a variety of ways. Tim Salimans and Chitwan Saharia helped us with brainstorming and coming up with shared benchmarks. Jason Baldridge was instrumental for bouncing ideas. Alex

Rizkowsky was very helpful in keeping things organized, while Erica Moreira and Victor Gomes ensured smooth resourcing for the project. Sarah Laszlo and Kathy Meier-Hellstern have greatly helped us incorporate important responsible AI practices into this project, which we are immensely grateful for. Finally, Blake Hechtman and Anselm Levskaya were generous in helping us debug a number of JAX issues.

REFERENCES

[1] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lucic, and Cordelia Schmid. Vivit: A video vision transformer. In *ICCV*, 2021.

[2] Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H Campbell, and Sergey Levine. Stochastic variational video prediction. *ICLR*, 2018.

[3] Mohammad Babaeizadeh, Mohammad Taghi Saffar, Suraj Nair, Sergey Levine, Chelsea Finn, and Dumitru Erhan. Fitvid: Overfitting in pixel-level video prediction. *arXiv preprint arXiv:2106.13195*, 2020.

[4] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1728–1738, 2021.

[5] Yogesh Balaji, Martin Renqiang Min, Bing Bai, Rama Chellappa, and Hans Peter Graf. Conditional gan with discriminative filter generation for text-to-video synthesis. In *IJCAI*, 2019.

[6] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017.

[7] Joao Carreira, Eric Noland, Andras Banki-Horvath, Chloe Hillier, and Andrew Zisserman. A short note about kinetics-600, 2018.

[8] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. Maskgit: Masked generative image transformer. *arXiv preprint arXiv:2202.04200*, 2022.

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

[10] Emily Denton and Rob Fergus. Stochastic video generation with a learned prior. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1174–1183, 2018.

[11] Frederik Ebert, Chelsea Finn, Alex X. Lee, and Sergey Levine. Self-supervised visual planning with temporal skip connections, 2017.

[12] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis, 2020.

[13] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in neural information processing systems*, pages 64–72, 2016.

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

[15] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[16] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2021.

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

[18] Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *arXiv preprint arXiv:2205.15868*, 2022.

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

[20] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. *arXiv preprint arXiv:1603.08155*, 2016.

[21] JTero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020.

[22] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset, 2017.

[23] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[24] Manoj Kumar, Mohammad Babaeizadeh, Dumitru Erhan, Chelsea Finn, Sergey Levine, Laurent Dinh, and Durk Kingma. Videoflow: A flow-based generative model for video. *arXiv preprint arXiv:1903.01434*, 2019.

[25] Yitong Li, Martin Min, Dinghan Shen, David Carlson, and Lawrence Carin. Video generation from text. In *AAAI*, 2018.

[26] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

[27] Pauline Luc, Aidan Clark, Sander Dieleman, Diego de Las Casas, Yotam Doron, Albin Cassirer, and Karen Simonyan. Transformation-based adversarial video prediction on large-scale data. *arXiv preprint arXiv:2003.04035*, 2019.

[28] Guillaume Le Moing, Jean Ponce, and Cordelia Schmid. CCVS: Context-aware controllable video synthesis. In *NeurIPS*, 2021.

[29] Mathew Monfort, Alex Andonian, Bolei Zhou, Kandan Ramakrishnan, Sarah Adel Bargal, Tom Yan, Lisa Brown, Quanfu Fan, Dan Gutfruend, Carl Vondrick, et al. Moments in time dataset: one million videos for event understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[30] Arsha Nagrani, Paul Hongsuck Seo, Bryan Andrew Seybold, Anja Hauth, Santiago Manen, Chen Sun, and Cordelia Schmid. Learning audio-video modalities from image captions. In *ECCV*, 2022.

[31] Charlie Nash, João Carreira, Jacob Walker, Iain Barr, Andrew Jaegle, Mateusz Malinowski, and Peter Battaglia. Transframer: Arbitrary frame prediction with generative models. *arXiv preprint arXiv:2203.09494*, 2019.

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

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

[34] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021.

[35] 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.

[36] MarcAurelio Ranzato, Arthur Szlam, Joan Bruna, Michael Mathieu, Ronan Collobert, and Sumit Chopra. Video (language) modeling: a baseline for generative models of natural videos. *arXiv preprint arXiv:1412.6604*, 2014.

[37] Adam Roberts, Hyung Won Chung, Anselm Levskaya, Gaurav Mishra, James Bradbury, Daniel Andor, Sharan Narang, Brian Lester, Colin Gaffney, Afroz Mohiuddin, et al. Scaling up models and data with t5x and seqio. *arXiv preprint arXiv:2203.17189*, 2022.

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

[39] Masaki Saito, Eiichi Matsumoto, and Shunta Saito. Temporal generative adversarial nets with singular value clipping. In *Proceedings of the IEEE international conference on computer vision*, pages 2830–2839, 2017.

[40] 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(10):2586–2606, 2020.

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

[42] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *International Conference on Machine Learning*, 2015.

[43] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. Mocogan: Decomposing motion and content for video generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1526–1535, 2018.

[44] 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.

[45] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *NeurIPS*, 2018.

[46] Ruben Villegas, Arkanath Pathak, Harini Kannan, Dumitru Erhan, Quoc V Le, and Honglak Lee. High fidelity video prediction with large stochastic recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 81–91, 2019.

[47] Vikram Voleti, Alexia Jolicoeur-Martineau, and Christopher Pal. Mcvd: Masked conditional video diffusion for prediction, generation, and interpolation. *arXiv preprint arXiv:2205.09853*, 2022.

[48] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. *arXiv preprint arXiv:1609.02612*, 2016.

[49] Jacob Walker, Ali Razavi, and Aäron van den Oord. Predicting video with vqvae. *arXiv preprint arXiv:2103.01950*, 2019.

[50] Yunbo Wang, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and Philip S Yu. Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms. *Advances in neural information processing systems*, 30, 2017.

[51] Dirk Weissenborn, Oscar Täckström, and Jakob Uszkoreit. Scaling autoregressive video models. In *ICLR*, 2020.

[52] Chenfei Wu, Lun Huang, Qianxi Zhang, Binyang Li, Lei Ji, Fan Yang, Guillermo Sapiro, and Nan Duan. Godiva: Generating open-domain videos from natural descriptions. *arXiv preprint arXiv:2104.14806*, 2021.

[53] Chenfei Wu, Jian Liang, Xiaowei Hu, Zhe Gan, Jianfeng Wang, Lijuan Wang, Zicheng Liu, Yuejian Fang, and Nan Duan. Nuwa-infinity: Autoregressive over autoregressive generation for infinite visual synthesis. *arXiv preprint arXiv:2207.09814*, 2022.

[54] Chenfei Wu, Jian Liang, Lei Ji, Fan Yang, Yuejian Fang, Daxin Jiang, and Nan Duan. NÜwa: Visual synthesis pre-training for neural visual world creation. In *ECCV*, 2022.

[55] Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. Videogpt: Video generation using vq-vae and transformers. *arXiv preprint arXiv:2104.10157*, 2019.

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

[57] Fangchen Liu Stephen James Pieter Abbeel Younggyo Seo, Kimin Lee. Harp: Autoregressive latent video prediction with high-fidelity image generator. *arXiv preprint arXiv:2209.07143*, 2022.

[58] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan. In *ICLR*, 2022.

[59] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, Ben Hutchinson, Wei Han, Zarana Parekh, Xin Li, Han Zhang, Jason Baldridge, and Yonghui Wu. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2022.

[60] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12104–12113, 2022.

[61] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, , and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. *CVPR*, 2018.

A HYPER-PARAMETERS

Symbol	Value	Description
t_x, w_x, h_x, c_x	$11, 128, 128, 3$	Video dimensions
t_p, w_p, h_p, c_p	$2, 8, 8, 3$	Patches dimensions (all frames except the first one)
t_z, w_z, h_z	$6, 16, 16$	Video tokens dimension (before linear projection)
h_z	512	Hidden size in the transformer layer
d_z	32	Embedding dimension (after linear projection)
$-$	4	Number of layers for spatial transformer
$-$	4	Number of layers for temporal transformer
$-$	2048	MLP size
$\|E\|$	8192	Codebook size
-	AdamW	Optimizer
β_1	0.9	first moment of gradient
β_2	0.99	second moment of gradient
-	1e-4	Learning rate
-	1e-4	Weight decay
-	Cosine decay	Learning rate scheduler
-	1M	Target number of training steps for learning rate scheduler
-	100K	Warmup steps
-	10	Gradient clipping magnitude
-	1028	Batch size

Table 6. Hyperparamters used for C-ViViT architecture and optimizer.

Symbol	Value	Description
$\|\mathbf{z}\|$	1536	Sequence Length
-	24	Number of layer
-	2048	Embedding dimension
-	8192	MLP dimension
-	32	Number of heads
-	AdamW	Optimizer
β_1	0.9	first moment of gradient
β_2	0.99	second moment of gradient
-	1e-4	Learning rate
-	1e-4	Weight decay
-	Cosine decay	Learning rate scheduler
-	4M	Target number of training steps for learning rate scheduler
-	10K	Warmup steps
-	10	Gradient clipping magnitude
-	512	Batch size

Table 7. Hyperparamters used for MaskGIT architecture and optimizer.

B DETAILS OF EXPERIMENTS

B.1 VIDEO QUANTIZATION

B.1.1 NETWORK ARCHITECTURE

All encoder-decoder baselines have approximately 50M parameters. The Convolutional baseline encoder architecture consists of 5 convolutional blocks with channel multipliers of $[1, 1, 2, 2, 4]$, 2 residual layers and 128 hidden units per block, and embedding dimension of 256. The ViT baseline encoder architecture consists of an image patchification step over non-overlapping 8×8 spatial patches which are linearly transformed into image tokens. Next, we follow with 8 transformer layers with 512 hidden units, 8 attention heads, 2048 mlp units, and embedding dimension of 32. C-ViViT encoder architecture patches the first frame to non-overlapping 8×8 patches, and then the rest of

the frames to non-overlapping $2 \times 8 \times 8$ spatio-temporal patches which are linearly transformed into video embeddings. Next, C-ViViT encoder architecture consists of 4 spatial and 4 temporal transformer layers with 512 hidden units, 8 attention heads, 2048 mlp hidden units, and embedding dimension of 32. The decoder architecture for all models is the same as the encoder but in reverse to put the latent embeddings back to image space. The VQ objective is trained with commitment loss of $\beta = 0.25$ and codebook size of 8192. The discriminator architecture is the StyleGAN [21] discriminator with blur resample, and channel multiplier of 1.

B.1.2 TRAINING

We train all encoder-decoder baselines and with StyleGAN [21] discriminators with a batch size of 128 using Adam optimizer [23] with $\beta_1 = 0.9$ and $\beta_2 = 0.99$. We use a linear learning rate warmup to a peak value of 1×10^{-4} over $100,000$ steps and then decaying over the remaining $900,000$ steps with a cosine schedule, and use a decoupled weight decay [26] of 1×10^{-4} for the encoder-decoder and discriminator. To capture longer time horizons during training and better evaluate temporal coherence, we downsample the MiT dataset from 25 FPS to 6 FPS and evaluate on videos of 10 frames at spatial resolution of 128×128.

B.2 IMAGE CONDITIONAL VIDEO GENERATION

B.2.1 BAIR ROBOT PUSH C-VIVIT ARCHITECTURE

We use a similar setup as in Section B.1, but the video tokenization step is done over 4×4 spatial patches on the first image and $2 \times 4 \times 4$ spatio-temporal patches in the rest of the video. The spatial encoder consists of 8 layers and the temporal encoder consists of 6 layers.

B.2.2 KINETICS-600 C-VIVIT ARCHITECTURE

We use a similar setup as in Section B.2.1, but both the spatial encoder and temporal encoder consist of 8 layers.

B.2.3 MASKGIT ARCHITECTURE

To perform video prediction in latent space in the BAIR Robot Push and Kinetics-600 datasets, we use an unconditional transformer architecture consisting of 24 layers, 768 hidden units, 16 attention heads, dropout and attention dropout rate of 0.1, 3072 mlp hidden units.

B.2.4 TRAINING AND INFERENCE

As described in Table 7, we train C-ViViT with the same optimizer setup as in Sec B.1, but we do not downsample the FPS of any of the datasets in this section for fair comparison with the video prediction baselines. We train MaskGIT on the video tokens extracted using C-ViViT in an unconditional setting, that is, we do not assume frames or text inputs to be given. During training, we use the Adam [23] optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.99$. We use a linear learning rate warmup up to a peak value of 1×10^{-4} over $10,000$ steps, and constant learning rate schedule for $\sim 2M$ steps. At inference time, we initialize MaskGIT given a number of input frames, and predict the rest of the frames depending on the dataset on which we evaluate.

B.3 TEXT CONDITIONAL VIDEO GENERATION

B.3.1 ARCHITECTURE

In our text conditional video generation, we use the same C-ViViT architecture and training described in Section B.1. To train MaskGIT, we include a text conditioning in the form of T5X embeddings [37] which are used as input through the use of cross attention with the video tokens. We reduce the number of parameters of our base model for fairness in the quantitative comparisons against NUWA. The MaskGIT architecture used against NUWA consists of 20 transformer layers with 1536 hidden units, 24 attention heads, and 6144 MLP hidden units, resulting in 0.9B parameters similar to NUWA. For the main experiments in this paper, we use a larger architecture that

consists of consists of 24 transformer layers with 2048 hidden units, 32 attention heads, and 8192 mlp hidden units, resulting in 1.8B parameters.

B.3.2 Training and inference

For all our text-conditional video generation, we use the training parameters Table 7.

B.3.3 Inference parameters against NUWA

We use $\lambda = 0.1$, 12 MaskGIT iterations, and temperature of 4.0.

B.3.4 Inference parameters for ablation of image and video data for training.

We use $\lambda = 6$, 24 MaskGIT iterations, and temperature of 4.0.

B.3.5 Inference parameters for all videos in the paper.

We use $\lambda = 12$, 48 MaskGIT iterations, and temperature of 8.0.

Figure 5. Another example of story conditional video generation. Full videos are available at phenaki.github.io.