Align your Latents: High-Resolution Video Synthesis with Latent Diffusion Models

Andreas Blattmann[1] *,† Robin Rombach[1] *,† Huan Ling[2,3,4] * Tim Dockhorn[2,3,5] *,†
Seung Wook Kim[2,3,4] Sanja Fidler[2,3,4] Karsten Kreis[2]

[1]LMU Munich [2]NVIDIA [3]Vector Institute [4]University of Toronto [5]University of Waterloo

Project page: https://research.nvidia.com/labs/toronto-ai/VideoLDM/

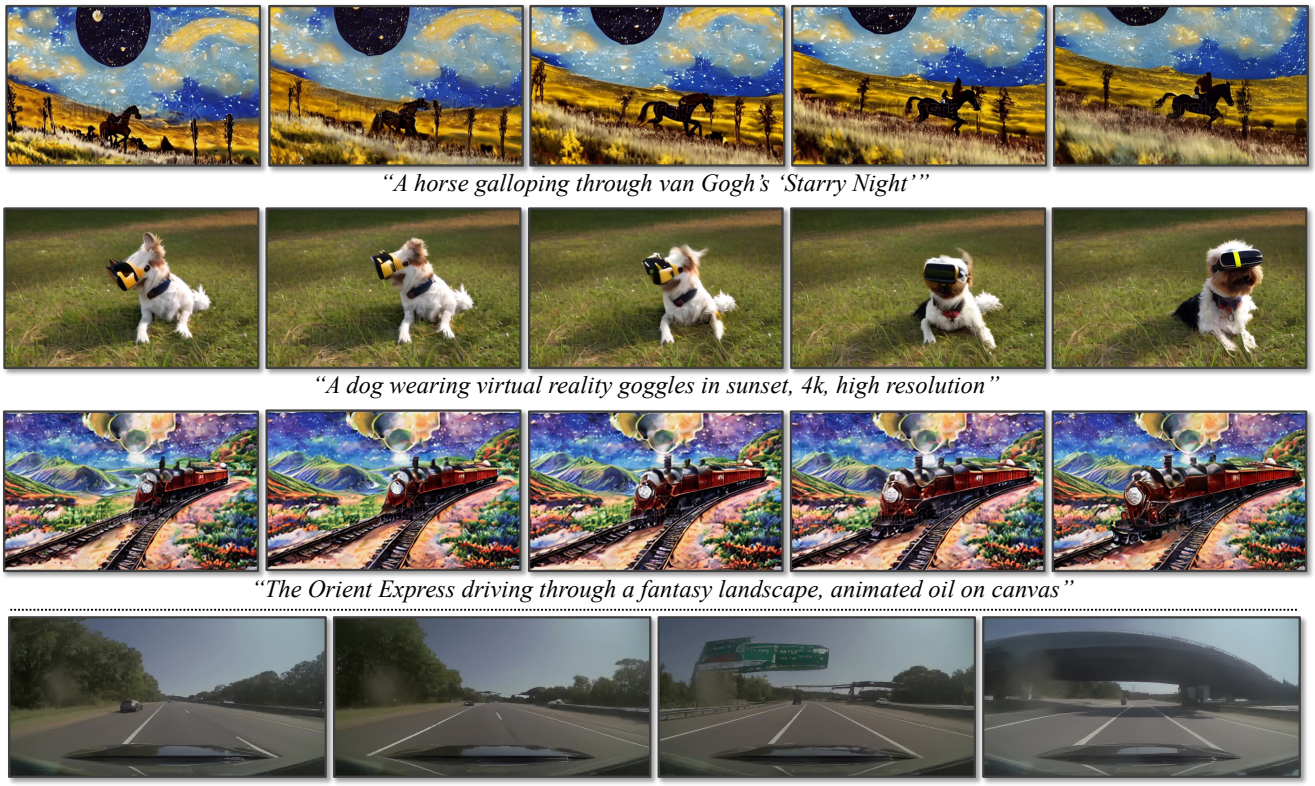

Figure 1. **Video LDM samples**. *Top:* Text-to-Video generation. *Bottom:* 512×1024 resolution real driving scene video generation.

Abstract

*Latent Diffusion Models (LDMs) enable high-quality image synthesis while avoiding excessive compute demands by training a diffusion model in a compressed lower-dimensional latent space. Here, we apply the LDM paradigm to high-resolution video generation, a particularly resource-intensive task. We first pre-train an LDM on images only; then, we turn the image generator into a video generator by introducing a temporal dimension to the latent space diffusion model and fine-tuning on encoded image sequences, i.e., videos. Similarly, we temporally align diffusion model upsamplers, turning them into temporally consistent video super resolution models. We focus on two relevant real-world applications: Simulation of in-the-wild driving data and creative content creation with text-to-video modeling. In particular, we validate our **Video LDM** on real driving videos of resolution 512×1024, achieving state-of-the-art performance. Furthermore, our approach can easily leverage off-the-shelf pre-trained image LDMs, as we only need to train a temporal alignment model in that case. Doing so, we turn the publicly available, state-of-the-art text-to-image LDM Stable Diffusion into an efficient and expressive text-to-video model with resolution up to 1280×2048. We show that the temporal layers trained in this way generalize to different fine-tuned text-to-image LDMs. Utilizing this property, we show the first results for personalized text-to-video generation, opening exciting directions for future content creation.*

*Equal contribution.
†Andreas, Robin and Tim did the work during internships at NVIDIA.

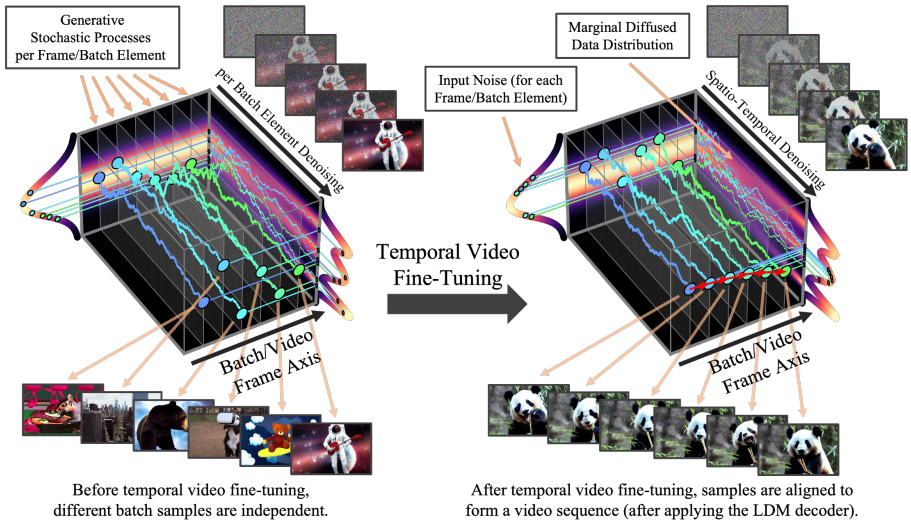

Figure 2. **Temporal Video Fine-Tuning.** We turn pre-trained image diffusion models into temporally consistent video generators. Initially, different samples of a batch synthesized by the model are independent. After temporal video fine-tuning, the samples are temporally aligned and form coherent videos. The stochastic generation process before and after fine-tuning is visualised for a diffusion model of a one-dim. toy distribution. For clarity, the figure corresponds to alignment in pixel space. In practice, we perform alignment in LDM's latent space and obtain videos after applying LDM's decoder (see Fig. 3). We also video fine-tune diffusion model upsamplers in pixel or latent space (Sec. 3.4).

1. Introduction

Generative models of images have received unprecedented attention, owing to recent breakthroughs in the underlying modeling methodology. The most powerful models today are built on generative adversarial networks [21, 38–40, 75], autoregressive transformers [15, 63, 105], and most recently diffusion models [10, 28, 29, 57, 58, 62, 65, 68, 79, 82]. Diffusion models (DMs) in particular have desirable advantages; they offer a robust and scalable training objective and are typically less parameter intensive than their transformer-based counterparts. However, while the image domain has seen great progress, *video* modeling has lagged behind—mainly due to the significant computational cost associated with training on video data, and the lack of large-scale, general, and publicly available video datasets. While there is a rich literature on video synthesis [1, 6, 8, 9, 17, 19, 22, 23, 32, 32, 37, 42, 44, 47, 51, 55, 59, 71, 78, 85, 91, 94, 97–99, 103, 106], most works, including previous video DMs [24, 31, 33, 93, 104], only generate relatively low-resolution, often short, videos. Here, we apply video models to real-world problems and generate high-resolution, long videos. Specifically, we focus on two relevant real-world video generation problems: (i) video synthesis of high-resolution real-word driving data, which has great potential as a simulation engine in the context of autonomous driving, and (ii) text-guided video synthesis for creative content generation; see Fig. 1.

To this end, we build on latent diffusion models (LDMs), which can reduce the heavy computational burden when training on high-resolution images [65]. We propose *Video LDMs* and extend LDMs to high-resolution *video* generation, a particularly compute-intensive task. In contrast to previous work on DMs for video generation [24, 31, 33, 93, 104], we first pre-train our Video LDMs on images only (or use available pre-trained image LDMs), thereby allowing us to leverage large-scale image datasets. We then transform the LDM image generator into a video generator by introducing a temporal dimension into the latent space DM and training only these temporal layers on encoded image sequences, *i.e.*, videos (Fig. 2), while fixing the pre-trained spatial layers. We similarly fine-tune LDM's decoder to achieve temporal consistency in pixel space (Fig. 3). To further enhance the spatial resolution, we also temporally align pixel-space and latent DM upsamplers [29], which are widely used for image super resolution [43, 65, 68, 69], turning them into temporally consistent video super resolution models. Building on LDMs, our method can generate globally coherent and long videos in a computationally and memory efficient manner. For synthesis at very high resolutions, the video upsampler only needs to operate locally, keeping training and computational requirements low. We ablate our method and test on 512×1024 real driving scene videos, achieving state-of-the-art video quality, and synthesize videos of several minutes length. We also video fine-tune a powerful, publicly available text-to-image LDM, *Stable Diffusion* [65], and turn it into an efficient and powerful text-to-video generator with resolution up to 1280×2048. Since we only need to train the temporal alignment layers in that case, we can use a relatively small training set of captioned videos. By transferring the trained temporal layers to differently fine-tuned text-to-image LDMs, we demonstrate personalized text-to-video generation for the first time. We hope our work opens new avenues for efficient digital content creation and autonomous driving simulation.

Contributions. *(i)* We present an efficient approach for training high-resolution, long-term consistent video generation models based on LDMs. Our key insight is to leverage pre-trained image DMs and turn them into video generators by inserting temporal layers that learn to align images in a temporally consistent manner (Figs. 2 and 3). *(ii)* We further temporally fine-tune super resolution DMs, which are ubiquitous in the literature. *(iii)* We achieve state-of-the-art high-resolution video synthesis performance on real driving scene videos, and we can generate multiple minute long

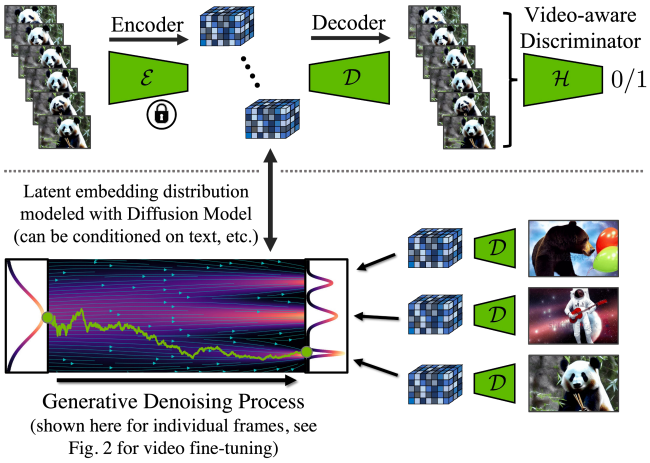

Figure 3. *Top:* During temporal decoder fine-tuning, we process video sequences with a frozen encoder, which processes frames independently, and enforce temporally coherent reconstructions across frames. We additionally employ a video-aware discriminator. *Bottom:* in LDMs, a diffusion model is trained in latent space. It synthesizes latent features, which are then transformed through the decoder into images. Note that the bottom visualization is for individual frames; see Fig. 2 for the video fine-tuning framework that generates temporally consistent frame sequences.

videos. *(iv)* We transform the publicly available *Stable Diffusion* text-to-image LDM into a powerful and expressive text-to-video LDM, and *(v)* show that the learned temporal layers can be combined with different image model checkpoints (*e.g.*, *DreamBooth* [66]).

2. Background

DMs [28, 79, 82] learn to model a data distribution $p_{\text{data}}(\mathbf{x})$ via *iterative denoising* and are trained with *denoising score matching* [28,34,50,79,81,82,92]: Given samples $\mathbf{x} \sim p_{\text{data}}$, *diffused* inputs $\mathbf{x}_\tau = \alpha_\tau \mathbf{x} + \sigma_\tau \boldsymbol{\epsilon},\ \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ are constructed; α_τ and σ_τ define a *noise schedule*, parameterized via a diffusion-time τ, such that the logarithmic signal-to-noise ratio $\lambda_\tau = \log(\alpha_\tau^2/\sigma_\tau^2)$ monotonically decreases. A denoiser model $\mathbf{f}_\theta$ (parameterized with learnable parameters θ) receives the diffused $\mathbf{x}_\tau$ as input and is optimized minimizing the denoising score matching objective

$$\mathbb{E}_{\mathbf{x}\sim p_{\text{data}},\tau\sim p_\tau,\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})}\left[\|\mathbf{y} - \mathbf{f}_\theta(\mathbf{x}_\tau; \mathbf{c}, \tau)\|_2^2\right], \quad (1)$$

where $\mathbf{c}$ is optional conditioning information, such as a text prompt, and the target vector $\mathbf{y}$ is either the random noise $\boldsymbol{\epsilon}$ or $\mathbf{v} = \alpha_\tau \boldsymbol{\epsilon} - \sigma_\tau \mathbf{x}$. The latter objective (often referred to as $\mathbf{v}$-*prediction*) has been introduced in the context of progressive distillation [73] and empirically often yields faster convergence of the model (here, we use both objectives). Furthermore, p_τ is a uniform distribution over the diffusion time τ. The forward diffusion as well as the reverse generation process in diffusion models can be described via stochastic differential equations in a continuous-time framework [82] (see Figs. 2 and 3), but in practice a fixed discretization can be used [28]. The maximum diffusion time is generally chosen such that the input data is entirely perturbed into Gaussian random noise and an iterative generative denoising process that employs the learned denoiser $\mathbf{f}_\theta$ can be initialized from such Gaussian noise to synthesize novel data. Here, we use $p_\tau \sim \mathcal{U}\{0, 1000\}$ and rely on a *variance-preserving* noise schedule [82], for which $\sigma_\tau^2 = 1 - \alpha_\tau^2$ (see Appendices F and H for details).

Latent Diffusion Models (LDMs) [65] improve in computational and memory efficiency over pixel-space DMs by first training a compression model to transform input images $\mathbf{x}\sim p_{\text{data}}$ into a spatially lower-dimensional latent space of reduced complexity, from which the original data can be reconstructed at high fidelity. In practice, this approach is implemented with a regularized autoencoder, which reconstructs inputs $\mathbf{x}$ via an encoder module $\mathcal{E}$ and a decoder $\mathcal{D}$, such that the reconstruction $\hat{\mathbf{x}}{=}\mathcal{D}(\mathcal{E}(\mathbf{x}))\approx\mathbf{x}$ (Fig. 3). To ensure photorealistic reconstructions, an adversarial objective can be added to the autoencoder training [65], which is implemented using a patch-based discriminator [35]. A DM can then be trained in the compressed latent space and $\mathbf{x}$ in Eq. (1) is replaced by its latent representation $\mathbf{z}{=}\mathcal{E}(\mathbf{x})$. This latent space DM can be typically smaller in terms of parameter count and memory consumption compared to corresponding pixel-space DMs of similar performance.

3. Latent Video Diffusion Models

Here we describe how we *video fine-tune* pre-trained image LDMs (and DM upsamplers) for high-resolution video synthesis. We assume access to a dataset p_{data} of videos, such that $\mathbf{x} \in \mathbb{R}^{T\times 3\times \tilde{H}\times \tilde{W}}$, $\mathbf{x} \sim p_{\text{data}}$ is a sequence of T RGB frames, with height and width $\tilde{H}$ and $\tilde{W}$.

3.1. Turning Latent Image into Video Generators

Our key insight for efficiently training a video generation model is to re-use a pre-trained, fixed image generation model; an LDM parameterized by parameters θ. Formally, let us denote the neural network layers that comprise the image LDM and process inputs over the pixel dimensions as *spatial* layers l_θ^i, with layer index i. However, although such a model is able to synthesize individual frames at high quality, using it directly to render a video of T consecutive frames will fail, as the model has no temporal awareness. We thus introduce additional *temporal* neural network layers l_ϕ^i, which are interleaved with the existing *spatial* layers l_θ^i and learn to align individual frames in a temporally consistent manner. These L additional temporal layers $\{l_\phi^i\}_{i=1}^L$ define the *video-aware* temporal backbone of our model, and the full model $\mathbf{f}_{\theta,\phi}$ is thus the combination of the spatial and temporal layers; see Fig. 4 for a visualization.

We start from a frame-wise encoded input video $\mathcal{E}(\mathbf{x}) = \mathbf{z} \in \mathbb{R}^{T\times C\times H\times W}$, where C is the number of latent channels

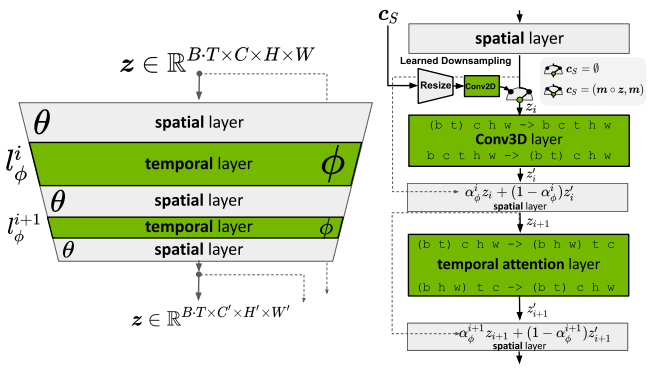

Figure 4. *Left:* We turn a pre-trained LDM into a video generator by inserting *temporal* layers that learn to align frames into temporally consistent sequences. During optimization, the image backbone θ remains fixed and only the parameters ϕ of the temporal layers l^i_ϕ are trained, *cf*. Eq. (2). *Right:* During training, the base model θ interprets the input sequence of length T as a batch of images. For the temporal layers l^i_ϕ, these batches are reshaped into video format. Their output $\mathbf{z}'$ is combined with the spatial output $\mathbf{z}$, using a learned merge parameter α. During inference, skipping the temporal layers (α^i_ϕ=1) yields the original image model. For illustration purposes, only a single U-Net Block is shown. B denotes batch size, T sequence length, C input channels and H and W the spatial dimensions of the input. $\mathbf{c}_S$ is optional context frame conditioning, when training prediction models (Sec. 3.2).

and H and W are the spatial latent dimensions. The spatial layers interpret the video as a batch of independent images (by shifting the temporal axis into the batch dimension), and for each *temporal mixing layer* l^i_ϕ, we reshape back to video dimensions as follows (using `einops` [64] notation):

$$\mathbf{z}' \leftarrow \texttt{rearrange}(\mathbf{z},\ \texttt{(b t) c h w} \rightarrow \texttt{b c t h w})$$
$$\mathbf{z}' \leftarrow l^i_\phi(\mathbf{z}', \mathbf{c})$$
$$\mathbf{z}' \leftarrow \texttt{rearrange}(\mathbf{z}',\ \texttt{b c t h w} \rightarrow \texttt{(b t) c h w}),$$

where we added the batch dimension `b` for clarity. In other words, the spatial layers treat all $B{\cdot}T$ encoded video frames independently in the batch dimension `b`, while the temporal layers $l^i_\phi(\mathbf{z}', \mathbf{c})$ process entire videos in a new temporal dimension `t`. Furthermore, $\mathbf{c}$ is (optional) conditioning information such as a text prompt. After each temporal layer, the output $\mathbf{z}'$ is combined with $\mathbf{z}$ as $\alpha^i_\phi \mathbf{z} + (1-\alpha^i_\phi)\mathbf{z}'$; $\alpha^i_\phi \in [0, 1]$ denotes a (learnable) parameter (also Appendix D).

In practice, we implement two different kinds of temporal mixing layers: (i) temporal attention and (ii) residual blocks based on 3D convolutions, *cf*. Fig. 4. We use sinusoidal embeddings [28, 89] to provide the model with a positional encoding for time.

Our video-aware temporal backbone is then trained using the same noise schedule as the underlying image model, and, importantly, we fix the spatial layers l^i_θ and *only* optimize the temporal layers l^i_ϕ via

$$\arg\min_\phi \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}, \tau\sim p_\tau, \boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})} \left[\|\mathbf{y} - \mathbf{f}_{\theta,\phi}(\mathbf{z}_\tau; \mathbf{c}, \tau)\|^2_2\right], \quad (2)$$

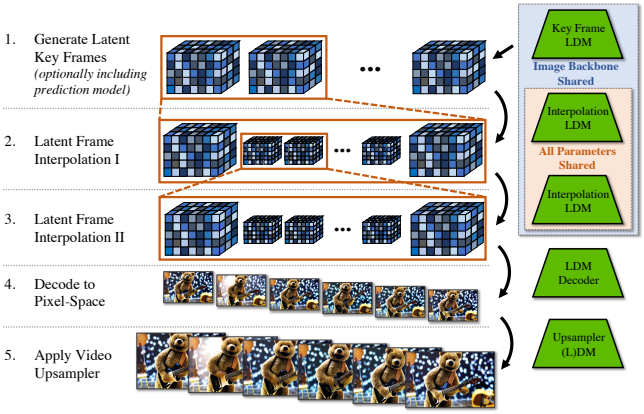

Figure 5. **Video LDM Stack.** We first generate sparse key frames. Then we temporally interpolate in two steps with the same interpolation model to achieve high frame rates. These operations are all based on latent diffusion models (LDMs) that share the same image backbone. Finally, the latent video is decoded to pixel space and optionally a video upsampler diffusion model is applied.

where $\mathbf{z}_\tau$ denotes diffused encodings $\mathbf{z} = \mathcal{E}(\mathbf{x})$. This way, we retain the native image generation capabilities by simply skipping the temporal blocks, *e.g.* by setting $\alpha^i_\phi = 1$ for each layer. A crucial advantage of our strategy is that huge image datasets can be used to pre-train the spatial layers, while the video data, which is often less widely available, can be utilized for focused training of the temporal layers.

3.1.1 Temporal Autoencoder Finetuning

Our video models build on pre-trained image LDMs. While this increases efficiency, the autoencoder of the LDM is trained on images only, causing flickering artifacts when encoding and decoding a temporally coherent sequence of images. To counteract this, we introduce additional temporal layers for the autoencoder's decoder, which we finetune on video data with a (patch-wise) temporal discriminator built from 3D convolutions, *cf*. Fig. 3. Note that the encoder remains unchanged from image training such that the image DM that operates in latent space on encoded video frames can be re-used. As demonstrated by computing reconstruction FVD [87] scores in Table 3, this step is critical for achieving good results.

3.2. Prediction Models for Long-Term Generation

Although the approach described in Sec. 3.1 is efficient for generating short video sequences, it reaches its limits when it comes to synthesizing very long videos. Therefore, we also train models as *prediction models* given a number of (first) S context frames. We implement this by introducing a temporal binary mask $\mathbf{m}_S$ which masks the $T - S$ frames the model has to predict, where T is the total sequence length as in Sec. 3.1. We feed this mask and the masked encoded video frames into the model for condition-

Figure 6. 1280×2048 resolution samples from our Stable Diffusion-based text-to-video LDM, including video fine-tuned upsampler. Prompts: "An astronaut flying in space, 4k, high resolution" and "Milk dripping into a cup of coffee, high definition, 4k".

ing. Specifically, the frames are encoded with LDM's image encoder $\mathcal{E}$, multiplied by the mask, and then fed (channel-wise concatenated with the masks) into the temporal layers l_ϕ^i after being processed with a learned downsampling operation, see Fig. 4. Let $\mathbf{c}_S = (\mathbf{m}_S \circ \mathbf{z}, \mathbf{m}_S)$ denote the concatenated spatial conditioning of masks and masked (encoded) images. Then, the objective from Eq. (2) reads

$$\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}, \mathbf{m}_S\sim p_S, \tau\sim p_\tau, \boldsymbol{\epsilon}} \left[\|\mathbf{y} - \mathbf{f}_{\theta,\phi}(\mathbf{z}_\tau; \mathbf{c}_S, \mathbf{c}, \tau)\|_2^2 \right], \quad (3)$$

where p_S represents the (categorical) mask sampling distribution. In practice, we learn prediction models that condition either on 0, 1 or 2 context frames, allowing for classifier-free guidance as discussed below.

During inference, for generating long videos, we can apply the sampling process iteratively, re-using the latest predictions as new context. The first initial sequence is generated by synthesizing a single context frame from the base image model and generating a sequence based on that; afterwards, we condition on two context frames to encode movement (details in Appendix). To stabilize this process, we found it beneficial to use *classifier-free diffusion guidance* [30], where we guide the model during sampling via

$$\mathbf{f}'_{\theta,\phi}(\mathbf{z}_\tau; \mathbf{c}_S) = \mathbf{f}_{\theta,\phi}(\mathbf{z}_\tau) + s \cdot (\mathbf{f}_{\theta,\phi}(\mathbf{z}_\tau; \mathbf{c}_S) - \mathbf{f}_{\theta,\phi}(\mathbf{z}_\tau)) \quad (4)$$

where $s \geq 1$ denotes the guidance scale and we dropped the explicit conditioning on τ and other information $\mathbf{c}$ for readability. We refer to this guidance as *context guidance*.

3.3. Temporal Interpolation for High Frame Rates

High-resolution video is characterized not only by high spatial resolution, but also by high temporal resolution, *i.e.*, a high frame rate. To achieve this, we divide the synthesis process for high-resolution video into two parts: The first is the process described in Sec. 3.1 and Sec. 3.2, which can generate *key frames* with large semantic changes, but (due to memory constraints) only at a relatively low frame rate. For the second part, we introduce an additional model whose task is to interpolate between given key frames. To implement this, we use the masking-conditioning mechanism introduced in Sec. 3.2. However, unlike the prediction task, we now mask the frames to be interpolated—otherwise, the mechanism remains the same, *i.e.*, the image model is refined into a video interpolation model. In our experiments, we predict three frames between two given key frames, thereby training a $T \to 4T$ interpolation model. To achieve even larger frame rates, we train the model simultaneously in the $T \to 4T$ and $4T \to 16T$ regimes (using videos with different fps), specified by binary conditioning.

Our training approach for prediction and interpolation models is inspired by recent works [24, 33, 93] that use similar masking techniques (also see Appendix C).

3.4. Temporal Fine-tuning of SR Models

Although the LDM mechanism already provides a good native resolution we aim to push this towards the megapixel range. We take inspiration from cascaded DMs [29] and use a DM to further scale up the Video LDM outputs by $4\times$. For our driving video synthesis experiments, we use a pixel-space DM [29] (Sec. 4.1) and scale to 512×1024; for our text-to-video models, we use an LDM upsampler [65] (Sec. 4.2) and scale to 1280×2048. We use noise augmentation with noise level conditioning [29, 68] and train the super resolution (SR) model $\mathbf{g}_{\theta,\phi}$ (on images or latents) via

$$\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}, (\tau,\tau_\gamma)\sim p_\tau, \boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})} \left[\|\mathbf{y} - \mathbf{g}_{\theta,\phi}(\mathbf{x}_\tau; \mathbf{c}_{\tau_\gamma}, \tau_\gamma, \tau)\|_2^2 \right] \quad (5)$$

where $\mathbf{c}_{\tau_\gamma} = \alpha_{\tau_\gamma}\mathbf{x} + \sigma_{\tau_\gamma}\boldsymbol{\epsilon}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, denotes a noisy low-resolution image given to the model via concatenation, and τ_γ the amount of noise added to the low-resolution image following the noise schedule α_τ, σ_τ.

Since upsampling video frames independently would result in poor temporal consistency, we also make this SR model video-aware. We follow the mechanism introduced in Sec. 3.1 with spatial layers l_θ^i and temporal layers l_ϕ^i and similarly video fine-tune the upscaler, conditioning on a low-resolution sequence of length T and concatenating low-resolution video images frame-by-frame. Since the upscaler operates locally, we conduct all upscaler training efficiently on patches only and later apply the model convolutionally.

Overall, we believe that the combination of an LDM with an upsampler DM is ideal for efficient high-resolution

Figure 7. 512 × 1024 resolution video modeling of real-world driving scenes with our Video LDM and video upsampler. *Top:* (Night time) **Driving Video Generation**. *Middle:* **Multimodal Driving Scenario Prediction**: We simulate two different scenarios given the same initial frame (red). *Bottom:* **Specific Driving Scenario Simulation**: We synthesize a scenario based on a manually designed, initial scene generated with a bounding box-conditioned Image LDM (yellow). More examples in the Appendix I.3.

video synthesis. On the one hand, the main LDM component of our Video LDM leverages a computationally efficient, compressed latent space to perform all video modeling. This allows us to use large batch sizes and jointly encode more video frames, which benefits long-term video modeling, without excessive memory demands, as all video predictions and interpolations are carried out in latent space. On the other hand, the upsampler can be trained in an efficient patch-wise manner, therefore similarly saving computational resources and reducing memory consumption, and it also does not need to capture long-term temporal correlations due to the low-resolution conditioning. Therefore, no prediction and interpolation framework is required for this component. A model overview, bringing together all components from Sec. 3.1 to Sec. 3.4, is depicted in Fig. 5.

A discussion of related work can be found in Appendix C.

4. Experiments

Datasets. Since we focus on driving scene video generation as well as text-to-video, we use two corresponding datasets/models: *(i)* An in-house dataset of real driving scene (RDS) videos. The dataset consists of 683,060 videos of 8 seconds each at resolution 512 × 1024 ($H \times W$) and frame rate up to 30 fps. Furthermore, the videos have binary night/day labels, annotations for the number of cars in a scene ("crowdedness"), and a subset of the data also has car bounding boxes. *(ii)* We use the WebVid-10M [2] dataset to turn the publicly available *Stable Diffusion* Image LDM [65] into a Video LDM. WebVid-10M consists of 10.7M video-caption pairs with a total of 52K video hours. We resize the videos into resolution 320 × 512. *(iii)* Moreover, in Appendix I.2, we show experiments on the Mountain Biking dataset by Brooks et al. [6].

Evaluation Metrics. To evaluate our models, we use frame-wise Fréchet Inception Distance (FID) [26] as well as Fréchet Video Distance (FVD) [87]. Since FVD can be unreliable (discussed, for instance, by Brooks et al. [6]), we additionally perform human evaluation. For our text-to-video experiments, we also evaluate CLIP similarity (CLIP-SIM) [98] and (video) inception score (IS) (Appendix G).

Model Architectures and Sampling. Our Image LDMs are based on Rombach et al. [65]. They use convolutional encoders and decoders, and their latent space DM architecture build on the U-Net by Dhariwal et al. [10]. Our pixel-space upsampler DMs use the same Image DM backbone [10]. DM sampling is performed using DDIM [80] in all experiments.

Further architecture, training, evaluation, sampling and dataset details can be found in the Appendix.

4.1. High-Resolution Driving Video Synthesis

We train our Video LDM pipeline, including a 4× pixel-space video upsampler, on the real driving scene (RDS) data. We condition on day/night labels and crowdedness, and randomly drop these labels during training to allow for classifier-free guidance and unconditional synthesis (we do not condition on bounding boxes here). Following the proposed training strategy above, we first train the image backbone LDM (spatial layers) on video frames independently, before we then train the temporal layers on videos. We also train Long Video GAN (LVG) [6], the previous state-of-the-art in long-term high-resolution video synthesis, on the RDS data to serve as main baseline. Table 1 (left) shows our main results for the Video LDM at 128 × 256 resolution,

Table 1. *Left:* Comparison with LVG on RDS; *Right:* Ablations.

Method	FVD	FID
LVG [6]	478	53.5
Ours	389	**31.6**
Ours (cond.)	**356**	51.9

Method	FVD	FID
Pixel-baseline	639,56	59.70
End-to-end LDM	1155.10	71.26
Attention-only	704.41	50.01
Ours	534.17	**48.26**
Ours (context-guided)	**508.82**	54.16

Table 2. User study on Driving Video Synthesis on RDS.

Method	Pref. A	Pref. B	Equal
Ours (cond.) v.s *Ours* (uncond.)	**49.33**	42.67	8.0
Ours (uncond.) v.s LVG	**54.02**	40.23	5.74
Ours (cond.) v.s LVG	**62.03**	31.65	6.33

Table 3. *Left:* Evaluating temporal fine-tuning for diffusion upsamplers on RDS data; *Right:* Video fine-tuning of the first stage decoder network leads to significantly improved consistency.

Method	FVD	FID
Ours Image Upsampler	165.98	**19.71**
Ours Video Upsampler	**45.39**	19.85

Decoder	*image-only*	*finetuned*
FVD	390.88	**32.94**
FID	**7.61**	9.17

without upsampler. We show both performance of our model with and without conditioning on crowdedness and day/night. Our Video LDM generally outperforms LVG and adding conditioning further reduces FVD. Table 2 shows our human evaluation: Our samples are generally preferred over LVG in terms of realism, and samples from our conditional model are also preferred over unconditional samples.

Next, we compare our video fine-tuned pixel-space upsampler with independent frame-wise image upsampling (Table 3), using 128×256 30 fps ground truth videos for conditioning. We find that temporal alignment of the upsampler is crucial for high performance. FVD degrades significantly, if the video frames are upsampled independently, indicating loss of temporal consistency. As expected, FID is essentially unaffected, because the individual frames are still of high quality when upsampled independently.

In Fig. 1 (bottom) and Fig. 7 (top), we show conditional samples from the combined Video LDM and video upsampler model. We observe high-quality videos. Moreover, using our prediction approach, we find that we can generate very long, temporally coherent high-resolution driving videos of multiple minutes. We validated this for up to 5 minutes; see Appendix and supplementary video for results.

4.1.1 Ablation Studies

To show the efficacy of our design choices (Sec. 3), we compare a smaller version of our Video LDM with various baselines on the RDS dataset and present the results in Table 1 (right) (for evaluation details, see Appendix G). First, using the exact same architecture as for our Video LDM, we apply our temporal finetuning strategy to a pre-trained pixel-space image diffusion model, which is clearly outperformed by ours. Further, we train an End-to-End LDM, whose entire set of parameters $\{\theta, \phi\}$ is learned on RDS videos without image pre-training of θ, leading to heavy degradations both in FID and FVD, when compared with our Video LDM. Another important architectural choice is the introduction of 3D convolutional temporal layers, since they allow us to feed the context frames $\boldsymbol{c}_S$ to the network spatially. This model achieves both lower FVD and FID scores than an attention-only temporal model, which uses the same set of spatial layers θ and has the same number of trainable parameters. Finally, we see that we can further lower FVD scores by applying *context guidance* while sacrificing a bit of visual quality indicated by increased FID scores.

Moreover, we provide an analysis on the effects of video fine-tuning the decoder of the compression model (*cf*. Sec. 3.1.1) which encompasses the LDM framework [65]. We apply our fine-tuning strategy to decoders of these compression models on the RDS dataset and compare both the obtained FVD/FID scores of reconstructed videos/image frames with those of their non-video-finetuned counterparts. Video fine-tuning leads to improvements by orders of magnitudes, as can be seen in Table 3.

4.1.2 Driving Scenario Simulation

A high-resolution video generator trained on in-the-wild driving scenes can potentially serve as a powerful simulation engine. We qualitatively explore this in Fig. 7. Given an initial frame, our video model can generate several different plausible future predictions. Furthermore, we also trained a separate, bounding box-conditioned image LDM on our data (only for image synthesis). A user can now manually create a scene composition of interest by specifying the bounding boxes of different cars, generate a corresponding image, and then use this image as initialization for our Video LDM, which can then predict different scenarios in a multimodal fashion (bottom in Fig. 7).

4.2. Text-to-Video with Stable Diffusion

Instead of first training our own Image LDM backbone, our Video LDM approach can also leverage existing Image LDMs and turn them into video generators. To demonstrate this, we turn the publicly available text-to-image LDM *Stable Diffusion* into a text-to-video generator. Specifically, using the WebVid-10M text-captioned video dataset, we train a temporally aligned version of Stable Diffusion for text-conditioned video synthesis. We briefly fine-tune Stable Diffusion's spatial layers on frames from WebVid, and then insert the temporal alignment layers and train them (at resolution 320×512). We also add text-conditioning in those alignment layers. Moreover, we further video fine-tune the publicly available latent *Stable Diffusion upsampler*, which enables $4\times$ upscaling and allows us to generate videos at resolution 1280×2048. We generate videos consisting of 113 frames, which we can render, for instance, into clips of 4.7 seconds length at 24 fps or into clips of 3.8 sec-

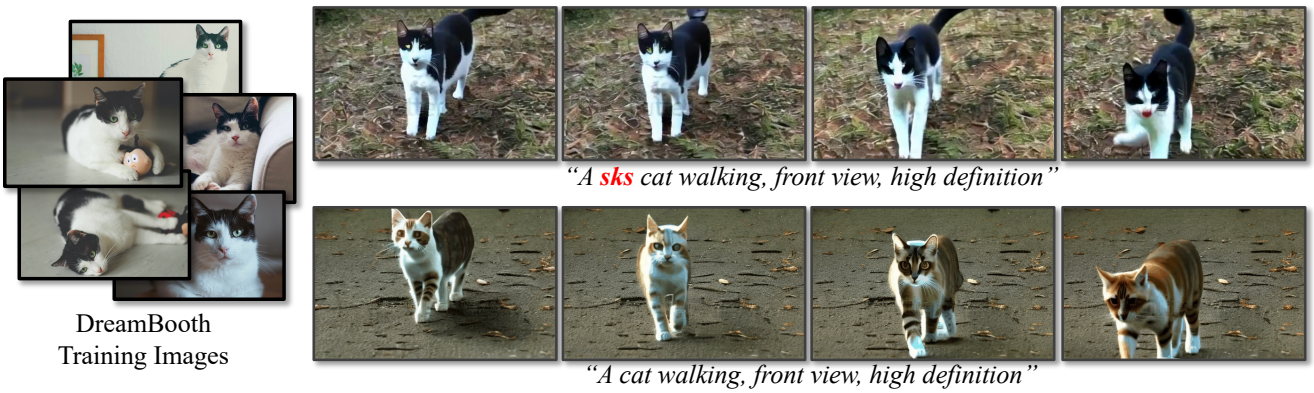

Figure 8. *Left:* DreamBooth Training Images. *Top row:* Video generated by our Video LDM with DreamBooth Image LDM backbone. *Bottom row:* Video generated without DreamBooth Image backbone. We see that the DreamBooth model preserves subject identity well.

Table 4. UCF-101 text-to-video generation.

Method	Zero-Shot	IS (↑)	FVD (↓)
CogVideo (Chinese) [32]	Yes	23.55	751.34
CogVideo (English) [32]	Yes	25.27	701.59
MagicVideo [109]	Yes	-	699.00
Make-A-Video [76]	Yes	33.00	367.23
Video LDM *(Ours)*	Yes	33.45	550.61

Table 5. MSR-VTT text-to-video generation performance.

Method	Zero-Shot	CLIPSIM (↑)
GODIVA [98]	No	0.2402
NÜWA [99]	No	0.2439
CogVideo (Chinese) [32]	Yes	0.2614
CogVideo (English) [32]	Yes	0.2631
Make-A-Video [76]	Yes	0.3049
Video LDM *(Ours)*	Yes	0.2929

onds length at 30 fps. Samples from the trained models are shown in Figs. 1 and 6. While WebVid-10M consists of photo-quality real-life videos, we are able to generate highly expressive and artistic videos beyond the video training data. This demonstrates that the general image generation capabilities of the Image LDM backbone readily translate to video generation, even though the video dataset we trained on is much smaller and limited in diversity and style. The Video LDM effectively combines the styles and expressions from the image model with the movements and temporal consistency learnt from the WebVid videos.

We evaluate zero-shot text-to-video generation on UCF-101 [83] and MSR-VTT [101] (Tabs. 4 & 5). Evaluation details in Appendix G. We significantly outperform all baselines except Make-A-Video [76], which we still surpass in IS on UCF-101. However, Make-A-Video is concurrent work, focuses entirely on text-to-video and trains with more video data than we do. We use only WebVid-10M; Make-A-Video also uses HD-VILA-100M [102].

In Appendix D, we show how we can apply our model "convolutional in time" and "convolutional in space", enabling longer and spatially-extended generation without upsampler and prediction models. More video samples shown in Appendix I.1. Experiment details in Appendix H.2.

4.2.1 Personalized Text-to-Video with Dreambooth

Since we have separate spatial and temporal layers in our Video LDM, the question arises whether the temporal layers trained on one Image LDM backbone transfer to other model checkpoints (*e.g.* fine-tuned). We test this for personalized text-to-video generation: Using DreamBooth [66], we fine-tune our Stable Diffusion spatial backbone on small sets of images of certain objects, tying their identity to a rare text token ("*sks*"). We then insert the temporal layers from the previously video-tuned Stable Diffusion (without DreamBooth) into the new DreamBooth version of the original Stable Diffusion model and generate videos using the token tied to the training images for DreamBooth (see Fig. 8 and examples in Appendix I.1.3). We find that we can generate personalized coherent videos that correctly capture the identity of the Dreambooth training images. This validates that our temporal layers generalize to other Image LDMs. To the best of our knowledge, we are the first to demonstrate personalized text-to-video generation.

Additional results and experiments in Appendix I.

5. Conclusions

We presented *Video Latent Diffusion Models* for efficient high-resolution video generation. Our key design choice is to build on pre-trained image diffusion models and to turn them into video generators by temporally video fine-tuning them with temporal alignment layers. To maintain computational efficiency, we leverage LDMs, optionally combined with a super resolution DM, which we also temporally align. Our Video LDM can synthesize high-resolution and temporally coherent driving scene videos of many minutes. We also turn the publicly available *Stable Diffusion* text-to-image LDM into an efficient text-to-video LDM and show that the learned temporal layers transfer to different model checkpoints. We leverage this for personalized text-to-video generation. We hope that our work can benefit simulators in the context of autonomous driving research and help democratize high quality video content creation (see Appendix B for broader impact and limitations).

References

[1] Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H. Campbell, and Sergey Levine. Stochastic variational video prediction. In *International Conference on Learning Representations*, 2018. 2, 15

[2] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *IEEE International Conference on Computer Vision*, 2021. 6, 17, 25

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

[4] Fan Bao, Chongxuan Li, Jun Zhu, and Bo Zhang. Analytic-dpm: an analytic estimate of the optimal reverse variance in diffusion probabilistic models. In *International Conference on Learning Representations*, 2022. 15

[5] Andreas Blattmann, Timo Milbich, Michael Dorkenwald, and Björn Ommer. ipoke: Poking a still image for controlled stochastic video synthesis. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, 2021. 15

[6] Tim Brooks, Janne Hellsten, Miika Aittala, Ting-Chun Wang, Timo Aila, Jaakko Lehtinen, Ming-Yu Liu, Alexei A Efros, and Tero Karras. Generating long videos of dynamic scenes. *arXiv:2206.03429*, 2022. 2, 6, 7, 15, 16, 17, 20, 21, 25, 26, 27

[7] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017. 20

[8] Lluis Castrejon, Nicolas Ballas, and Aaron Courville. Improved conditional vrnns for video prediction. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 2, 15

[9] Emily Denton and Rob Fergus. Stochastic video generation with a learned prior. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, 2018. 2, 15

[10] Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems*, 2021. 2, 6, 15, 17

[11] Tim Dockhorn, Arash Vahdat, and Karsten Kreis. Genie: Higher-order denoising diffusion solvers. In *Advances in Neural Information Processing Systems*, 2022. 15

[12] Tim Dockhorn, Arash Vahdat, and Karsten Kreis. Score-based generative modeling with critically-damped langevin diffusion. In *International Conference on Learning Representations (ICLR)*, 2022. 15

[13] Michael Dorkenwald, Timo Milbich, Andreas Blattmann, Robin Rombach, Konstantinos G. Derpanis, and Björn Ommer. Stochastic image-to-video synthesis using cinns. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, 2021. 15

[14] Patrick Esser, Johnathan Chiu, Parmida Atighehchian, Jonathan Granskog, and Anastasis Germanidis. Structure and content-guided video synthesis with diffusion models. *arXiv preprint arXiv:2302.03011*, 2023. 16

[15] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. *arXiv preprint arXiv:2012.09841*, 2020. 2

[16] Gereon Fox, Ayush Tewari, Mohamed Elgharib, and Christian Theobalt. Stylevideogan: A temporal generative model using a pretrained stylegan. In *British Machine Vision Conference (BMVC)*, 2021. 15, 16

[17] Jean-Yves Franceschi, Edouard Delasalles, Mickaël Chen, Sylvain Lamprier, and Patrick Gallinari. Stochastic latent residual video prediction. In *Proceedings of the 37th International Conference on Machine Learning*, 2020. 2, 15

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

[19] Songwei Ge, Thomas Hayes, Harry Yang, Xi Yin, Guan Pang, David Jacobs, Jia-Bin Huang, and Devi Parikh. Long video generation with time-agnostic vqgan and time-sensitive transformer. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, pages 102–118, Cham, 2022. Springer Nature Switzerland. 2, 15

[20] Songwei Ge, Thomas Hayes, Harry Yang, Xi Yin, Guan Pang, David Jacobs, Jia-Bin Huang, and Devi Parikh. Long video generation with time-agnostic vqgan and time-sensitive transformer. *arXiv preprint arXiv:2204.03638*, 2022. 20

[21] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 2

[22] Sonam Gupta, Arti Keshari, and Sukhendu Das. Rv-gan: Recurrent gan for unconditional video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 2024–2033, June 2022. 2, 15

[23] Tanmay Gupta, Dustin Schwenk, Ali Farhadi, Derek Hoiem, and Aniruddha Kembhavi. Imagine this! scripts to compositions to videos. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 610–626, Cham, 2018. Springer International Publishing. 2, 16

[24] William Harvey, Saeid Naderiparizi, Vaden Masrani, Christian Weilbach, and Frank Wood. Flexible diffusion modeling of long videos. *arXiv preprint arXiv:2205.11495*, 2022. 2, 5, 16

[25] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022. 15

[26] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by

a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 6, 20

[27] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P. Kingma, Ben Poole, Mohammad Norouzi, David J. Fleet, and Tim Salimans. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022. 16, 25

[28] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020. 2, 3, 4, 15, 16, 17

[29] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *arXiv preprint arXiv:2106.15282*, 2021. 2, 5, 15, 21, 23

[30] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. 5

[31] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models. *arXiv preprint arXiv:2204.03458*, 2022. 2, 16

[32] Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *arXiv:2205.15868*, 2022. 2, 8, 15, 16, 21, 24, 25

[33] Tobias Höppe, Arash Mehrjou, Stefan Bauer, Didrik Nielsen, and Andrea Dittadi. Diffusion models for video prediction and infilling. *arXiv preprint arXiv:2206.07696*, 2022. 2, 5, 16

[34] Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6:695–709, 2005. 3

[35] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 3

[36] Alexia Jolicoeur-Martineau, Ke Li, Rémi Piché-Taillefer, Tal Kachman, and Ioannis Mitliagkas. Gotta go fast when generating data with score-based models. *arXiv:2105.14080*, 2021. 15

[37] Emmanuel Kahembwe and Subramanian Ramamoorthy. Lower dimensional kernels for video discriminators. *Neural Networks*, 132:506–520, 2020. 2, 15

[38] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. In *Proc. NeurIPS*, 2021. 2, 21

[39] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019. 2

[40] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8110–8119, 2020. 2

[41] Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. *arXiv preprint arXiv:2201.11793*, 2022. 15

[42] Alex X. Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018. 2, 15

[43] Haoying Li, Yifan Yang, Meng Chang, Shiqi Chen, Huajun Feng, Zhihai Xu, Qi Li, and Yueting Chen. Srdiff: Single image super-resolution with diffusion probabilistic models. *Neurocomputing*, 479:47–59, 2022. 2, 15

[44] Yitong Li, Martin Renqiang Min, Dinghan Shen, David Carlson, and Lawrence Carin. Video generation from text. *arXiv preprint arXiv:1710.00421*, 2017. 2, 16

[45] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *International Conference on Learning Representations*, 2022. 15

[46] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *arXiv:2206.00927*, 2022. 15

[47] 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*, 2020. 2, 15

[48] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. *arXiv preprint arXiv:2201.09865*, 2022. 15

[49] Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv preprint arXiv:2101.02388*, 2021. 15

[50] Siwei Lyu. Interpretation and generalization of score matching. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, page 359–366, Arlington, Virginia, USA, 2009. AUAI Press. 3

[51] Tanya Marwah, Gaurav Mittal, and Vineeth N. Balasubramanian. Attentive semantic video generation using captions. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1435–1443, 2017. 2, 16

[52] Chenlin Meng, Ruiqi Gao, Diederik P. Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. *arXiv preprint arXiv:2210.03142*, 2022. 15

[53] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022. 15

[54] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Which training methods for gans do actually converge? In

International Conference on Machine Learning (ICML), 2018. 21

[55] Gaurav Mittal, Tanya Marwah, and Vineeth N. Balasubramanian. Sync-draw: Automatic video generation using deep recurrent attentive architectures. In *Proceedings of the 25th ACM International Conference on Multimedia*, MM '17, page 1096–1104, New York, NY, USA, 2017. Association for Computing Machinery. 2, 16

[56] Eyal Molad, Eliahu Horwitz, Dani Valevski, Alex Rav Acha, Yossi Matias, Yael Pritch, Yaniv Leviathan, and Yedid Hoshen. Dreamix: Video diffusion models are general video editors. *arXiv preprint arXiv:2302.01329*, 2023. 16

[57] 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. 2, 15

[58] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, 2021. 2, 15

[59] Yingwei Pan, Zhaofan Qiu, Ting Yao, Houqiang Li, and Tao Mei. To create what you tell: Generating videos from captions. *Proceedings of the 25th ACM international conference on Multimedia*, 2017. 2, 16

[60] Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 15

[61] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 21

[62] 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. 2, 15, 16

[63] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8821–8831. PMLR, 18–24 Jul 2021. 2

[64] Alex Rogozhnikov. Einops: Clear and reliable tensor manipulations with einstein-like notation. In *International Conference on Learning Representations*, 2022. 4

[65] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *arXiv preprint arXiv:2112.10752*, 2021. 2, 3, 5, 6, 7, 15, 16, 17, 22

[66] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image dissusion models for subject-driven generation. *arXiv preprint arXiv:2208.12242*, 2022. 3, 8, 15, 25

[67] Chitwan Saharia, William Chan, Huiwen Chang, Chris A. Lee, Jonathan Ho, Tim Salimans, David J. Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. *arXiv preprint arXiv:2111.05826*, 2021. 15

[68] 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, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022. 2, 5, 15, 16, 21, 23

[69] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *arXiv preprint arXiv:2104.07636*, 2021. 2, 15

[70] Masaki Saito, Eiichi Matsumoto, and Shunta Saito. Temporal generative adversarial nets with singular value clipping. In *ICCV*, 2017. 15

[71] 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*, May 2020. 2, 15, 21

[72] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, 2016. 21

[73] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations (ICLR)*, 2022. 3, 15

[74] Hiroshi Sasaki, Chris G. Willcocks, and Toby P. Breckon. UNIT-DDPM: Unpaired image translation with denoising diffusion probabilistic models. *arXiv preprint arXiv:2104.05358*, 2021. 15

[75] Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–10, 2022. 2

[76] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv:2209.14792*, 2022. 8, 16, 20, 21, 24, 25

[77] Abhishek Sinha, Jiaming Song, Chenlin Meng, and Stefano Ermon. D2c: Diffusion-denoising models for few-shot conditional generation. In *Advances in Neural Information Processing Systems*, 2021. 15

[78] Ivan Skorokhodov, Sergey Tulyakov, and Mohamed Elhoseiny. Stylegan-v: A continuous video generator with the price, image quality and perks of stylegan2. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3626–3636, June 2022. 2, 15

[79] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using

nonequilibrium thermodynamics. In *International Conference on Machine Learning*, 2015. 2, 3, 15

[80] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. 6, 15, 17

[81] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Proceedings of the 33rd Annual Conference on Neural Information Processing Systems*, 2019. 3

[82] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. 2, 3, 15

[83] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 8

[84] Xuan Su, Jiaming Song, Chenlin Meng, and Stefano Ermon. Dual diffusion implicit bridges for image-to-image translation. *arXiv preprint arXiv:2203.08382*, 2022. 15

[85] Yu Tian, Jian Ren, Menglei Chai, Kyle Olszewski, Xi Peng, Dimitris N. Metaxas, and Sergey Tulyakov. A good image generator is what you need for high-resolution video synthesis. In *International Conference on Learning Representations*, 2021. 2, 15, 16

[86] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015. 21

[87] 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:1812.01717*, 2018. 4, 6, 20

[88] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. In *Advances in Neural Information Processing Systems*, 2021. 15

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

[90] Ruben Villegas, Mohammad Babaeizadeh, Pieter-Jan Kindermans, Hernan Moraldo, Han Zhang, Mohammad Taghi Saffar, Santiago Castro, Julius Kunze, and Dumitru Erhan. Phenaki: Variable length video generation from open domain textual description. *arXiv:2210.02399*, 2022. 16

[91] Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. Decomposing motion and content for natural video sequence prediction. *ICLR*, 2017. 2, 15

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

[93] Vikram Voleti, Alexia Jolicoeur-Martineau, and Christopher Pal. Mcvd: Masked conditional video diffusion for prediction, generation, and interpolation. *arXiv preprint arXiv:2205.09853*, 2022. 2, 5, 16

[94] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016. 2, 15

[95] Yaohui Wang, Piotr Bilinski, Francois Bremond, and Antitza Dantcheva. G3an: Disentangling appearance and motion for video generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 15

[96] Daniel Watson, William Chan, Jonathan Ho, and Mohammad Norouzi. Learning fast samplers for diffusion models by differentiating through sample quality. In *International Conference on Learning Representations*, 2022. 15

[97] Dirk Weissenborn, Oscar Täckström, and Jakob Uszkoreit. Scaling autoregressive video models. In *International Conference on Learning Representations*, 2020. 2, 15

[98] 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:2104.14806*, 2021. 2, 6, 8, 15, 16, 21, 24

[99] 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 *European Conference on Computer Vision*, pages 720–736. Springer, 2022. 2, 8, 15, 16, 24

[100] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion GANs. In *International Conference on Learning Representations (ICLR)*, 2022. 15

[101] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 8

[102] Hongwei Xue, Tiankai Hang, Yanhong Zeng, Yuchong Sun, Bei Liu, Huan Yang, Jianlong Fu, and Baining Guo. Advancing high-resolution video-language representation with large-scale video transcriptions. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 8

[103] Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. Videogpt: Video generation using vq-vae and transformers, 2021. 2, 15

[104] Ruihan Yang, Prakhar Srivastava, and Stephan Mandt. Diffusion probabilistic modeling for video generation. *arXiv preprint arXiv:2203.09481*, 2022. 2, 16

[105] 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. 2

[106] Sihyun Yu, Jihoon Tack, Sangwoo Mo, Hyunsu Kim, Junho Kim, Jung-Woo Ha, and Jinwoo Shin. Generating videos with dynamics-aware implicit generative adversarial networks. In *International Conference on Learning Representations*, 2022. 2, 15

[107] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 15

[108] Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. *arXiv:2204.13902*, 2022. 15

[109] Daquan Zhou, Weimin Wang, Hanshu Yan, Weiwei Lv, Yizhe Zhu, and Jiashi Feng. Magicvideo: Efficient video generation with latent diffusion models. *arXiv preprint arXiv:2211.11018*, 2022. 8, 16, 24

Contents

A. Linked Videos

For fully rendered videos, we primarily refer the reader to our project page, https://research.nvidia.com/labs/toronto-ai/VideoLDM/.

Moreover, we include 5 videos in the following google drive folder: https://drive.google.com/drive/folders/1ENd9_9lzN6mI3E_HAjP52_KMWFdd0c8u:

- `driving.mp4`: This video summarizes different results for our Video LDM trained on the real-world driving data.
- `text2video.mp4`: This video shows different results for our text-to-video LDM based on Stable Diffusion.
- `mountain_biking.mp4`: This video presents further results on an additional mountain biking video dataset.
- `5_minutes_driving.mp4`: This video shows full 5 minute long generated videos for the Video LDM trained on the real-world driving data.
- `5_minutes_biking.mp4`: This video shows full 5 minute long generated videos for the Video LDM trained on the mountain biking data.

We also keep a copy of the latest version of the paper in the google drive folder, in case the paper is updated at a later point in time.

B. Broader Impact and Limitations

Powerful video generative models, like our Video LDM, have the potential to enable important content creation applications in the future and streamline and improve the creative workflow of digital artists, thereby democratizing artistic expression. Moreover, when applied for instance on videos captured from vehicles, as in our main validation experiments, generative models like ours may also serve as simulators in autonomous driving research.

Our synthesized videos are not indistinguishable from real content yet. However, enhanced versions of our model may in the future reach an even higher quality, potentially being able to generate videos that appear to be deceptively real. This has important ethical and safety implications, as state-of-the-art deep generative models can also be used for malicious purposes, and therefore generative models like ours generally need to be applied with an abundance of caution. Moreover, the data sources cited in this paper are for research purposes only and not intended for commercial application or use, and the text-to-image backbone LDMs used in this research project have been trained on large amounts of internet data. Consequently, our model is not suitable for productization. An important direction for future work is training large-scale generative models with ethically sourced, commercially viable data.

C. Related Work

Here, we present an extended discussion about related work.

Diffusion Models for Image Synthesis. Diffusion models (DMs) [28, 79, 82] have proven to be powerful image generators, yielding state-of-the art results in both unconditional and class-conditional synthesis [10, 58, 65] as well as text-to-image generation [3, 57, 62, 65, 68]. They have also been successfully used for various image editing and processing tasks [18, 25, 41, 43, 48, 53, 66, 67, 69, 74, 84].

However, despite advances in model distillation [49, 52, 73] and accelerated sampling [4, 11, 12, 36, 45, 46, 80, 96, 100, 108], DMs generally require repeated evaluations of a computationally demanding large U-Net. Thus, DMs are computationally expensive during both training and inference, especially when applied at high resolutions. To address this, *cascaded* [29] and *latent* [65, 88] diffusion models have been introduced. Both approaches divide the synthesis (and training) process into multiple stages and move the resource-intensive training and evaluation to a space of lower computational complexity. Cascaded diffusion models start out as low-resolution models and apply a series of super resolution diffusion models. Latent space models first compress the image data using an autoencoder and learn the DMs on the resulting latent space. We combine the best of these approaches for video synthesis. Our main video generator is a latent diffusion model. Additionally, some of our models use a video upsampler like in cascaded models to further increase the resolution. Variations of latent diffusion models have also been used for tasks such as controllable and semantic image generation [60, 77], and beyond image synthesis, such as 3D shape synthesis [107].

Video Synthesis. Video generation has been tackled with recurrent neural networks [1, 8, 9, 17, 42], autoregressive transformers [19, 22, 32, 97–99, 103], Normalizing Flows [5, 13], and generative adversarial networks (GANs) [6, 16, 37, 47, 70,

71, 78, 85, 91, 94, 95, 106]. In particular LongVideoGAN [6] achieves high-resolution video synthesis over relatively long time intervals, combining a low- and a high-resolution model. Moreover, the idea to insert temporal layers into pre-trained generators has been explored by the GAN-based methods MoCoGAN-HD [85] and StyleVideoGAN [16] before, but at a much smaller scale for simple object-centric videos. Another important work is CogVideo [32], which video fine-tunes a text-to-image model. However, it is a strictly autoregressive architecture building on transformers, trained in a discrete latent space. Our method, in contrast, relies on continuous DMs, is much less parameter intensive and outperforms CogVideo in text-to-video synthesis. Furthermore, our alignment layers are also implemented differently. They do not require an autoregressive masking and, for our text-to-video experiments, condition on the text prompts. Finally, we also show driving scene generation, where we employ an additional video fine-tuned upsampler module. Many more text-to-video models exist [23, 31, 44, 51, 55, 59, 98, 99].

Most related to our work are previous DMs for video synthesis: Ho et al. [31] model low-resolution videos with DMs in pixel space and train jointly on image and video data. Yang et al. [104] parameterize autoregressive video generation using a deterministic frame predictor together with a probabilistic DM. Voleti et al. [93] introduce a general DM framework that simultaneously enables video generation, prediction, and interpolation, similarly to Höppe et al. [33]. Closely related, Harvey et al. [24] synthesize long videos by generating sparse frames first, and then adding the missing frames. However, they model only low-resolution videos from small simulated toy worlds, to which the DM easily overfits, whereas we are training exclusively on diverse high-resolution real-world data. In our Video LDM, we build on these works for our video prediction and interpolation frameworks. However, in contrast to our Video LDM, all previous video DMs work directly in pixel space, synthesize low-resolution videos only, and do not directly leverage pre-trained image DMs.

Concurrent Work. Concurrently with us, Make-A-Video [76] leveraged a DALL·E 2-like text-to-image DM [62] and temporally aligned its decoder as well as one super resolution DM for consistent video generation. Furthermore, Imagen Video [27] trained a large-scale cascaded text-to-video model building on the Imagen [68] architecture. It constructs a highly demanding spatial and temporal super resolution pipeline consisting of in total 7 DMs with a total of $>$11B parameters (also see model parameters comparisons in Appendix H.2.1). In contrast to our Video LDM, both Imagen Video and Make-A-Video operate entirely in pixel space, which is less efficient, and exclusively target text-to-video, whereas we also demonstrate high-resolution driving scene generation. Furthermore, neither the large-scale DALL·E 2 and Imagen text-to-image models, nor the corresponding text-to-video models are easily reproducible without substantial GPU resources. In contrast, we are building on *Stable Diffusion* to train our text-to-video model, and construct an overall significantly more efficient pipeline. This makes our approach more user-friendly. Phenaki [90] is another strong concurrent text-to-video model that compresses videos into discrete tokens and models the token distribution via bidirectional transformers. MagicVideo [109] is a concurrent method that also leverages latent diffusion models for video generation and follows a partially related strategy compared to our Video LDM. We are outperforming MagicVideo quantitatively (see Sec. 4.2), enable higher resolution generation, show personalized video generation, and also demonstrate long driving scene and mountain bike video generation; in contrast, MagicVideo tackles text-to-video only. Other relevant concurrent works include GEN-1 [14], which also leverages a latent diffusion framework and combines it with depth conditioning for structure- and content-aware video editing and stylization, as well as Dreamix [56], a method for video editing that demonstrates personalized video generation.

D. Using Video LDM "Convolutional in Time" and "Convolutional in Space"

An intriguing property of image LDMs is their ability to generalize to spatial resolutions much larger than the ones they are trained on. This is realized by increasing the spatial size of the sampled noise and leveraging the convolutional nature of the U-Net backbone as presented in [65]. Since we use the Stable Diffusion LDM as fixed generative image backbone for our text-to-video model, our approach naturally preserves this property, which enables us to increase the spatial resolution at inference time without significant loss of image quality. We show examples at spatial resolution 512×512 generated by applying this convolutional sampling "in space" in our videos and in Figs. 12 and 13. Note that our model was trained on resolution 320×512. This convolutional application of our model for spatially extended generation essentially comes for free. We are not using the additional upsampler diffusion model in those experiments.

Furthermore, to extend convolutional sampling to the temporal dimension and, thus, to be able to generate videos much longer than those our model has been trained on, we make the following design choices regarding our temporal layers. First, we use relative sinusoidal positional encodings for our temporal attention layers similar to those used to encode the timesteps in our U-Net backbone [28]. Second, we parameterize the learned mixing factors α_ϕ^i, *cf*. Sec. 3.1, with scalars for our text-to-video model (in the other models, α_ϕ^i varies along the temporal dimension, which would prevent convolutional-in-time processing. α_ϕ^i is always constant in the spatial and channel dimensions for all models). These choices ensure that our model

can generate longer sequences by simply increasing the number of frames for the model to render. Note that when applying our model convolutionally in time, we mask the temporal attention layers such that we only attend over a maximum frame distance of 8 frames, similar to training.

We find that our Video LDM generalizes to longer sequences as shown in Fig. 14, although some degradation in quality can be observed. Furthermore, we can combine convolutional sampling in space and time leading to high-resolution videos of lengths up to 30 seconds as presented in Figs. 15 and 16 and in our videos, although our model has been trained only on sequences of 4 seconds. That said, we find that convolutional-in-time generation can be fragile, in particular when targeting long videos. Hence, we advocate training of prediction models for more robust long-term generation, as we do in our experiments on real driving scene synthesis. Also synthesizing long text-to-video samples with prediction models is left to future work.

Nevertheless, to the best of our knowledge, our approach is among the first to simultaneously generate long, high resolution *and* high (up to 30) fps videos while keeping training cost tolerable.

E. Datasets

E.1. Real Driving Scene (RDS) Video

Our RDS dataset consists of 683,060 real driving videos of 8 seconds length each at resolution 512×1024 ($H \times W$). 85,841 of these video have a frame rate of 30 fps, the rest 10 fps. Furthermore, all videos have binary night/day labels (1 for night, 0 for day) and annotations for the number of cars in a scene ("crowdedness"). Note that most of the driving videos are relatively empty highway scenes with low crowdedness.

Moreover, the data comes with an additional 100k, independent, frames that have car bounding box annotations (we only used that data for training a bounding box-conditioned image LDM to initialize video synthesis in the qualitative experiments in Sec. 4.1.2 and Fig. 7).

E.2. WebVid-10M

When turning Stable Diffusion into a text-to-video generator, we rely on WebVid-10M [2]. WebVid-10M is a large-scale dataset of short videos with textual descriptions sourced from stock footage sites. The videos are diverse and rich in their content. There are 10.7M video-caption pairs with a total of 52k video hours.

E.3. Mountain Bike

In Appendix I.2, we report additional results on a first-person mountain biking video dataset, originally introduced by Brooks et al. in Long Video GAN [6]. The dataset consists of 1,202 clips of varying, but at least 5 seconds length. The videos have a frame rate of 30 fps. The dataset is available in different resolutions, with a maximum resolution of 576×1024. More details about the dataset can be found in Brooks et al. [6] and at `https://github.com/NVlabs/long-video-gan`.

F. Architecture, Training and Sampling Details

Our Image LDMs are based on Rombach et al. [65]. They use convolutional encoders and decoders, and their latent space DM architecture builds on the U-Net by Dhariwal et al. [10]. Our pixel-space upsampler DMs use the same Image DM backbone [10]. We generally work with discretized diffusion time steps $t \in \{0, 1000\}$, and use a linear noise schedule, following the formulation of Ho et al. [28]. Also see Appendix B of Rombach et al. [65], which recapitulates the diffusion process setup as it is used in LDMs.

All architecture details, diffusion process details, as well as training hyperparameters are provided in Tables 6 to 8.

To sample from our models, we generally use the sampler from *Denoising Diffusion Implicit Models* (DDIM) [80]. The number of sampling steps, the stochasticity η, and the guidance scale vary and are also shown in Tables 6 and 7.

F.1. Description of Hyperparameters

Our hyperparameter tables (Tables 6 to 8) follow the hyperparameter tables from [65] and should be mostly self-explanatory. Here, we give some additional description for some parameters of the temporal layers:

- $\dim \alpha$ (Architecture): Dimension of the learnable skip connection parameter α that is applied after temporal layers; see Fig. 2 and Sec. 3.1.
- $\dim c_S$ (Concat Conditioning): Dimension of the concatenated spatial conditioning of masks and masked (encoded) images; see Sec. 3.2.

Table 6. Hyperparameters for our diffusion models. †: For the Text-to-Video LDMs, we do not train our own Image LDM, but use Stable Diffusion, as discussed. *Training* details correspond to Stable Diffusion 2.1-based Video LDM that was also used in quantitative evaluations in Tabs. 4 and 5. *: We trained with 80×80 patches and run at full image resolution during inference (see Appendix H.2).

Hyperparameter	**Driving (Video) LDM**	**Driving (Video) Upsampler**	**Text-to-Video (Video) LDM †**	**Text-to-Video (Video) LDM Upsampler †**	**Mountain Biking (Video) LDM**
Image (L)DM					
Architecture					
LDM	✓	✗	✓	✓	✓
f	8	-	8	4	8
z-shape	$16 \times 32 \times 4$	-	$40 \times 64 \times 4$	$80 \times 80 \times 4$ *	$16 \times 32 \times 4$
Channels	224	160	320	320	256
Depth	4	2	2	2	2
Channel multiplier	1,2,2,3,4	1,2,4,4	1,2,4,4	1,2,4,4	1,2,4
Attention resolutions	16,8,4	8	64,32,16	128,64,32	32,16,8
Head channels	32	-	-	-	-
Number of heads	-	4	8	8	4
CA Conditioning					
Embedding dimension	256	768	768	768	-
CA resolutions	16,8,4	8	64,32,16	128,64,32	-
CA sequence length	1	1	77	77	-
Training					
Parameterization	$\mathbf{v}$	$\mathbf{v}$	ϵ	ϵ	$\mathbf{v}$
# train steps	73K	84K	-	-	14K
Learning rate	10^{-4}	10^{-5}	-	-	10^{-4}
Batch size per GPU	40	4	-	-	300
# GPUs	16	8	-	-	1
GPU-type	A100-40GB	A100-80GB	-	-	A100-80GB
p_{drop}	0.1	-	-	-	-

Temporal Layers	*prediction*	*interpolation*	*upsampler*	*keyframes model*	*interpolation*	*upsampler*	*prediction*	*interpolation*
Architecture								
Depth	4	4	2	2	2	2	2	2
Attention resolutions	16,8,4	16,8,4	1, 2, 4, 4	64,32,16	64,32,16	128,64,32	32,16,8	32,16,8
Head channels	32	32	-	-	-	-	-	-
Number of heads	-	-	4	8	8	8	4	4
Input projection	✓	✓	✓	✓	✓	✓	✓	✓
Positional encoding	Sinusoidal	Sinusoidal	-	Sinusoidal	Sinusoidal	Sinusoidal	Sinusoidal	Sinusoidal
$\dim \alpha$	10	5	10	1	1	1	8	5
Temporal kernel size	3,1,1	3,1,1	3,1,1	3,1,1	3,1,1	3,1,1	3,1,1	3,1,1
Concat Conditioning								
$\dim c_S$	5	5	3	-	-	3	5	5
Context channels	128	128	128	-	-	128	128	128
CA Conditioning								
Embedding dimension	-	512	768	768	768	768	-	512
CA resolutions	-	16,4,8	8	64,32,16	64,32,16	64,32,16	-	32,16,8
CA sequence length	-	1	1	77	2	77	-	1
Training								
# train steps	146K	42K	84K	402K	95K	10K	107K	33K
Learning rate	10^{-4}	10^{-4}	10^{-4}	10^{-4}	10^{-4}	10^{-4}	10^{-4}	10^{-4}
Batch size per GPU	9	16	2	3	8	8	20	20
# GPUs	40	24	8	256	128	32	8	6
GPU-type	A100-40GB	A100-40GB	A100-80GB	A100-80GB	A100-80GB	A100-80GB	A100-40GB	A100-40GB
Sequence length	10	5	8	8	5	8	8	5
Training data FPS	2	7.5, 30	4	2	7.5, 30	2	4	7.5, 30

Diffusion Setup					
Diffusion steps	1000	1000	1000	1000	1000
Noise schedule	Linear	Linear	Linear	Linear	Linear
β_0	0.0015	1e-4	0.00085	0.00085	0.0003
β_T	0.0195	2e-2	0.0120	0.0120	0.022

Sampling Parameters								
Sampler	DDIM	DDIM	DDIM	DDIM	DDIM	DDIM	DDIM	DDIM
Steps	200	50	50	250	100	250	100	50
η	1.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0
Guidance scale	2.0	2.0	1.0	8.0	13.0	2.0	2.0	1.0

Table 7. Hyperparameters for our all models presented in the ablation study in Table 1. *: This baseline has twice as many attention layers as the other models, leading to the same number of trainable parameters.

Hyperparameter	***Ours***	**Pixel-baseline**	**End-to-End LDM**	**Attention-only***
Image (L)DM				
Architecture				
LDM	✓	✗	✓	✓
Pretrained	✓	✓	✗	✓
f	8	-	8	8
z-shape	$16 \times 32 \times 4$	$128 \times 256 \times 3$	$16 \times 32 \times 4$	$16 \times 32 \times 4$
Channels	224	128	224	224
Depth	4	4	4	4
Channel multiplier	1,2,2,3,4	1,2,2,3,4	1,2,2,3,4	1,2,2,3,4
Attention resolutions	8,4,2	16,8	8,4,2	8,4,2
Head channels	32	32	32	32
CA Conditioning				
embedding dimension	256	256	256	256
CA resolutions	16,8,4	32,16	16,8,4	16,8,4
CA sequence length	1	1	1	1
Training				
Parameterization	$\mathbf{v}$	$\mathbf{v}$	$\mathbf{v}$	$\mathbf{v}$
# train steps	73K	42K	73K	73K
Learning rate	10^{-4}	10^{-4}	10^{-4}	10^{-4}
Batch size per GPU	40	12	40	40
# GPUs	16	40	16	16
GPU-type	A100-40GB	A100-40GB	A100-40GB	A100-40GB
p_{drop}	0.1	0.1	0.1	0.1
Temporal Layers				
Architecture				
Depth	4	4	4	4
Attention resolutions	1,2,2,3,4	1,2,2,3,4	1,2,2,3,4	1,2,2,3,4
Head channels	32	32	32	32
Input projection	✓	✓	✓	✓
Positional encoding	Sinusoidal	Sinusoidal	Sinusoidal	Sinusoidal
$\dim \alpha$	10	10	10	10
Concat Conditioning				
$\dim c_S$	5	4	5	-
Context channels	128	128	128	-
Temporal kernel size	3,1,1	3,1,1	3,1,1	-
Training				
# train steps	60K	78K	62K	61K
Learning rate	10^{-4}	10^{-4}	10^{-4}	10^{-4}
Batch size per GPU	18	1	16	13
# GPUs	2	2	2	2
GPU-type	A100-80GB	A100-80GB	A100-80GB	A100-80GB
Sequence length	10	10	10	10
Training data FPS	2	2	2	2
Diffusion Setup				
Diffusion steps	1000	1000	1000	1000
Noise schedule	Linear	Linear	Linear	Linear
β_0	0.0015	10^{-4}	0.0015	0.0015
β_T	0.0195	0.02	0.0195	0.0195
Sampling Parameters				
Sampler	DDIM	DDIM	DDIM	DDIM
Steps	200	200	200	200
η	1.0	1.0	1.0	1.0

Table 8. Hyperparameters for our autoencoder models.

Hyperparameter	Driving (Video) VQAE	WebVid (Video) AE	Mountain Biking (Video) AE
Image LDM Autoencoder			
Architecture			
Regularization	*VQ*	*KL*	*KL*
Downsampling factor	8	8	8
dim z	4	4	4
Codebook size	16384	-	-
Base channels	128	128	128
Channel multiplier	1,2,2,4	1,2,2,4	1,2,2,4
Depth	2	2	2
Attention resolutions	32	-	-
Temporal Layers			
Architecture			
Base channels	128	128	128
Channel multiplier	1,2,2,4	1,2,2,4	1,2,2,4
Depth	2	2	2
Temporal kernel size	3,3,3	3,3,3	3,3,3
Training			
# train steps	41K	35K	78K
Learning rate	4.0e-5	4.0e-5	5.0e-5
Batch size per GPU	4	1	5
# GPUs	8	40	8
GPU-type	A100-40GB	A100-40GB	A100-80GB
Sequence length	6	6	6

- Context channels (Concat Conditioning): Number of output channels of the learned downsampling layers, *cf*. Sec. 3.2 and Fig. 4, which is concatenated to the input of the temporal convolutional layers.

- Temporal kernel size (Concat Conditioning): The kernel size of the 3D kernel of our temporal convolutional layers as [`time`, `height`, `width`].

G. Quantitative Evaluation

We perform quantitative evaluations on all datasets. In particular, we compute Fréchet Inception Distance (FID) and Fréchet Video Distance (FVD) metrics. Since FVD can be unreliable (discussed, for instance, in [6]), we additionally perform human evaluation. For text-to-video evaluation, we also compute (video) Inception Scores (IS) and CLIP Similarly scores (CLIPSIM).

FVD: The FVD metric measures similarity between real and generated videos [87]. We follow [6] and generate 2,048 videos (16 frames at 30 fps), except for the FVD score evaluation of our SD 2.1-based text-to-video Video LDM for the UCF-101 benchmark, where we used 10k model samples, following Make-A-Video [76]. We then extract features from a pre-trained I3D action classification model[3]. For reference statistics, we take random sequences of videos that contain at least 16 frames from the dataset. For driving scenario and mountain bike video generation, reference statistics are calculated from 2,048 videos; for text-to-video experiments, reference statistics are calculated from 10k videos. Our implementation is a simple adaption from the code provided by [20].[4]

FID: To compute FID [26] (for driving video and mountain biking video synthesis), we randomly extract 10k frames (except for mountain biking for which we extract 50k frames) from the 2,048 generated videos as well as from dataset

[3]We use the model provided by [6, 7]: `https://www.dropbox.com/s/ge9e5ujwgetktms/i3d_torchscript.pt?dl=1'`
[4]We use the official UCF FVD evaluation code provided by [20]: `https://github.com/SongweiGe/TATS/`

videos. We then extract features from a pre-trained Inception model.[5]

Human evaluation: We conduct human evaluation (user study) on Amazon Mechanical Turk to evaluate the realism of generated videos by our method in comparison to LVG [6]. For our user study, we create 100 videos, each of length 4 seconds. The user study shows pairs of videos, and each pair has one random video from our method and one from LVG. For each pair, we instruct the participants to select the favorable video in a non-forced-choice response, i.e., the participants can vote for "equally realistic". See Fig. 9 for a screenshot. Note that the A-B order of the pairs is also assigned randomly.

Each video pair was shown to four participants resulting in 400 responses per dataset. We only select workers from English-speaking countries. There are no human subjects and we do not study the participants themselves; therefore, IRB review is not applicable. We pay \$0.04 for answering a single question.

Inception Score: In our text-to-video experiments on UCF-101, we also evaluated inception scores (IS) [72]. Following previous work on video synthesis [32, 76], we used a C3D [86] model trained on UCF-101 to calculate a video version of the inception score. It is calculated from 10k samples using the official code of TGANv2 [71].[6]

CLIP Similarity (CLIPSIM): In our text-to-video experiments on MSR-VTT, we also evaluated CLIP similarity (CLIPSIM) [98]. The MSR-VTT test set contains 2990 examples and 20 descriptions/prompts per example. We generate 2990 videos (16 frames at 30 fps) by using one random prompt per example. We then average the CLIPSIM score of the 47,840 frames. We use the *ViT-B/32* [61] model to compute the CLIP score.

Note that we directly use UCF class names as text conditioning, e.g., "Basketball Dunk", in contrast to Make-A-Video [76], which manually constructs a template sentence for each class.

H. Experiment Details

In addition to the model hyperparameters above in Table 6, here we provide additional details on the experiments presented in the paper.

H.1. Details: High-Resolution Driving Video Synthesis—Sec. 4.1

We initially train our base Image LDM (both autoencoder and latent space diffusion model) on 1 fps videos from the RDS dataset at resolution 128×256. We then train the temporal layers for sparse key frame prediction with 2 fps. For that, we extracted 2 fps videos from the 10 fps driving data. As discussed in the main paper, we inform the model about the number of frames given for prediction, which is either 0, 1, or 2.

The pixel-space $4\times$ upsampler that scales the resolution to 512×1024 is trained using the same data, but at a correspondingly higher resolution. The upsampler is trained with noise augmentation and conditioning on the noise level [29, 68]. During training we randomly sample $t \in \{0, \dots, 250\}$ and perturb the low-resolution conditioning following our variance-preserving diffusion process (using the same linear noise schedule as for the main upsampling diffusion model). At inference time, we use 150 steps for perturbation.

The LDM decoder fine-tuning is performed on 30 fps videos.

The temporal interpolation model is trained using 30 fps video data, which we process for the different tasks. We train the temporal interpolation model to first scale from 1.875 fps to 7.5 fps, and then to scale from 7.5 fps to 30 fps. We are using one interpolation model with shared parameters for that, providing a conditioning label to indicate to the model which of the two temporal upsampling operations is desired.

Video Generation. For video synthesis, we first generate a single frame using the image LDM, then we run the prediction model, conditioning on the single frame, to generate a sequence of key frames. When extending the video, we again call the prediction model, but condition on two frames (which captures directional information) to produce consistent motion. Next, we optionally perform two steps of the temporal interpolation, going from 1.875 to 7.5 fps and from 7.5 to 30 fps, respectively. Also optionally, the video fine-tuned upsampler is then run over portions of 8 video frames.

LVG Baseline. Our main baseline is the state-of-the-art Long Video GAN (LVG) [6], which we trained on the RDS data at 10 fps and resolution 128×256 (comparisons to our Video LDM are carried out at this resolution and fps). LVG's low resolution component was trained with a batch size of 64 and two gradient accumulation steps and an R1 [54] penalty of 1.0. For the super resolution module, we used a batch size of 2 without gradient accumulation and the same R1 penalty. Other than that, we used LVG's default hyperparameters.

[5] We use the model provided by [38]: https://api.ngc.nvidia.com/v2/models/nvidia/research/stylegan3/versions/1/files/metrics/inception-2015-12-05.pkl.

[6] We use the official UCF IS evaluation code provided by [71]: https://github.com/pfnet-research/tgan2

Figure 9. Screenshot of instructions provided to participants for the human evaluation study.

Ablation Studies. For the ablation studies (Sec. 4.1.1), we trained a smaller version of our Video LDM as well as an "end-to-end" version of our Video LDM, which simultaneously trains the spatial and temporal layers of the latent diffusion model from scratch. The hyperparameters for trained these models are shown in Table 7.

Bounding box-conditioned image LDM. For the experiments in Sec. 4.1.2, we trained a separate bounding-box conditioned image LDM (no videos) on the 100k independent annotated frames (see Appendix E.1). It uses the same hyperparameters as our main model's image LDM backbone for training. The conditioning is implemented using cross-attention [65]: We learn embeddings of the bounding box coordinates, fuse all coordinate embeddings using a transformer, and attend to the resulting fused embeddings.

H.2. Details: Text-to-Video with Stable Diffusion—Sec. 4.2

We ran experiments with three of the publicly available Stable Diffusion (SD) checkpoints as image LDM backbones: 1.4[7], 2.0[8], and 2.1[9]. Most of the research project was conducted with the SD 1.4-based model and the SD 2.0- and SD 2.1-based Video LDMs were trained primarily for exploration purposes and additional qualitative results later (however, our quantitative evaluation for text-to-video in Tabs. 4 and 5 uses the latest SD 2.1-based model; see discussion below). Unless indicated otherwise, all shown samples in the different figures throughout the paper are from our SD 1.4-based model without upsampler. Samples from the SD 2.0-based model with upsampler (see below) are presented in Figs. 6 and 17 to 20 and in the second example in Fig. 1. Many samples from the SD 2.1-based Video LDM are shown on our project page (https://research.nvidia.com/labs/toronto-ai/VideoLDM/).

Since SD is trained on images at resolution 512×512, naively applying it to the smaller-sized videos of the WebVid-10M dataset would lead to severe degradations in image quality. We therefore first fine-tune the Stable Diffusion image backbone (spatial layers) on the WebVid-10M data. Specifically, we resize and center-crop the WebVid-10M videos to 320×512 resolution and then fine-tune the SD latent space diffusion model on independent encoded frames from the videos. We rely on standard SD training hyperparameters, with a learning rate of 10^{-4}. Note that while this (spatial layer) fine-tuning of SD on the WebVid-10M data is necessary to prevent out-of-distribution problems when modeling videos in the next step, it also slightly hurts the overall image quality compared to the original SD checkpoint, because the WebVid data is arguably of lower visual quality than the images used to train the original SD model. We assume that training with more and higher-quality video data will solve this.

Next, as described in Sec. 3 we video fine-tuned both SD's latent space diffusion model and its decoder (Sec. 3.1.1) using the WebVid-10M videos. Note that our temporal layers also consume the text conditioning (Fig. 4). We do not train a prediction model here, but only train for text-to-video generation without any additional context frames. Overall, we train our pipeline for generation of videos consisting of 113 frames (which we can render, for instance, into clips of 4.7 seconds length at 24 fps or into clips of 3.8 seconds length at 30 fps.). To synthesize longer videos, we can optionally apply our temporal layers "convolutionally in time"; similarly, to generate spatially extended higher resolution videos we can apply the model "convolutionally in space" (see Appendix D).

To reach high frame rates and enable smooth video generation, we again train an interpolation model that can temporally upsample videos from 1.875 fps to 7.5 fps as well as 7.5 fps to 30 fps. Note that in contrast to our experiments on driving scenes and mountain biking, for the text-to-video interpolation models we trained all model parameters, including the ones in the spatial layers of the image LDM backbone. For the SD 1.4/2.0-based Video LDMs, we initialized the spatial layers from the fine-tuned SD checkpoint and trained the temporal layers from scratch, whereas for SD 2.1 all layers (except for the first convolutional layer) are initialized from the trained keyframe model (including the temporal layers). For the SD 2.0-based Video LDM, we did not use text-conditioning in the interpolation model and feed empty text into SD's text inputs (the SD 1.4- and SD 2.1-based Video LDMs did use text conditioning also in their interpolation models). Since we do two rounds of interpolation, we condition the models on the fps rate to which we interpolate (by cross attention to a corresponding embedding). Moreover, for these text-to-video interpolation models we did not use the learned downsampling approach (Fig. 4), but instead concatenated the context via partially masked out frames (and the mask itself) to the channel dimension C of the U-Net input. Furthermore, we use conditioning augmentation [29] for this additional input during training (we randomly sample $t \in \{0, \dots, 250\}$ and perturb the conditioning inputs following our variance-preserving diffusion process, using the same linear noise schedule as for the main diffusion model). At test time, the conditioning augmentation is turned off, i.e., the noise level is set to zero. Finally, the interpolation models of the SD 1.4-based and SD 2.1-based Video LDMs use one conditioning frame on either side, whereas the SD 2.0-based model uses two conditioning frames on either side. Except for the training differences discussed above, the SD 1.4-based, SD 2.0-based and SD 2.1-based Video LDMs are trained with the same hyperparameters (see Table 6).

Upsampler Training: As described in Sec. 4.2, we also video fine-tune the publicly available text-guided Stable Diffusion $4\times$-upscaler[10], which is itself a latent diffusion model. We train the upsampler for temporal alignment in a patch-wise manner on 320×320 cropped videos (WebVid-10M), which are embedded into a 80×80 latent space. The 80×80 low resolution conditioning videos are concatenated to the 80×80 latents. The learnt temporal alignment layers are text-conditioned, like for our base text-to-video LDMs. Like for the driving models, the upsampler is trained with noise augmentation and conditioning on the noise level, following previous work [29, 68]. During training we randomly sample $t \in \{0, \dots, 250\}$ and perturb the

[7] https://huggingface.co/CompVis/stable-diffusion-v-1-4-original
[8] https://huggingface.co/stabilityai/stable-diffusion-2
[9] https://huggingface.co/stabilityai/stable-diffusion-2-1
[10] https://huggingface.co/stabilityai/stable-diffusion-x4-upscaler

Table 9. UCF-101 text-to-video generation performance.

Method	Zero-Shot	IS (↑)	FVD (↓)
CogVideo (Chinese) [32]	Yes	23.55	751.34
CogVideo (English) [32]	Yes	25.27	701.59
MagicVideo [109]	Yes	-	699.00
Make-A-Video [76]	Yes	33.00	367.23
Video LDM (SD 1.4) *(Ours)*	Yes	29.49	656.49
Video LDM (SD 2.1) *(Ours)*	Yes	33.45	550.61

Table 10. MSR-VTT text-to-video generation performance.

Method	Zero-Shot	CLIPSIM (↑)
GODIVA [98]	No	0.2402
NÜWA [99]	No	0.2439
CogVideo (Chinese) [32]	Yes	0.2614
CogVideo (English) [32]	Yes	0.2631
Make-A-Video [76]	Yes	0.3049
Video LDM (SD 1.4) *(Ours)*	Yes	0.2848
Video LDM (SD 2.1) *(Ours)*	Yes	0.2929

low-resolution conditioning following our variance-preserving diffusion process (using the same linear noise schedule as for the main upsampling diffusion model). At inference time, we use 30 steps for perturbation. Moreover, we apply the model at extended resolution during inference. We provide 320×512 resolution videos as low resolution input, predict 320×512 resolution latents, and decode to 1280×2048 resolution videos. Note that we did not find it necessary to temporally fine-tune the decoder of this latent diffusion model upscaler.

Results in Tabs. 4 and 5: The results reported in those tables are based on experiments with the SD 2.1-based Video LDM. In Tabs. 9 and 10, we provide extended results including an earlier, preliminary evaluation of our Video LDM based on the SD 1.4 model. Note that the FVD score was calculated with only 2,048 samples from the model in that case. The FVD score for the SD 2.1-based Video LDM was calculated with 10k generated videos, following Make-A-Video [76], for fair comparison with this work.

Overall, we find that the SD 2.1-based Video LDM performs better than the SD 1.4-based one on these benchmarks. For UCF-101, our SD 2.1-based Video LDM even slightly outperforms Make-A-Video [76] on Inception Score.

H.2.1 Number of Model Parameters

Our text-to-video LDMs that are based on Stable Diffusion have

- 84 million parameters in the autoencoder (decoder is fine-tuned).
- 860 (SD 1.4) / 865 (SD 2.0/2.1) million parameters in the image backbone LDM, this is, in the spatial layers not including the CLIP text encoder (not trained).
- 649 (SD 1.4) / 656 (SD 2.0/2.1) million parameters in the temporal layers (trained).
- 123 (SD 1.4 uses CLIP ViT-L/14) / 354 (SD 2.0/2.1 uses OpenCLIP-ViT/H) million parameters in the text encoder (not trained).
- 1,509 million parameters in the interpolation latent diffusion model (trained). We use the same interpolation model for our SD 1.4- and SD 2.0/2.1-based models.

Combined, for the SD-2.0/2.1-based Video LDMs this sums to around 3.1B parameters in the autoencoder and diffusion model components (excluding the CLIP text embedders) in the low resolution text-to-video LDM. Out of this, only around 2.2B parameters are actually trained.

Moreover, our fine-tuned text-to-video latent upsampler that is based on the public Stable Diffusion $4\times$ upscaler has

- 55 million parameters in the autoencoder (not trained).

Table 11. Effects of video fine-tuning of the decoders of the compression framework encompassing Video LDM. Here we compute reconstruction FVD and FID based on 2,048 examples from the respective datasets (WebVid data is used to fine-tune the decoder of the text-to-video LDM based on Stable Diffusion). *fine-tuned* denotes our video fine-tuned decoders.

Dataset	WebVid [2]		Mountain Biking [6]	
Method	*image-only*	*fine-tuned*	*image-only*	*fine-tuned*
FVD	35.82	**18.66**	73.78	**25.55**
FID	13.89	**11.68**	20.76	**18.65**

- 473 million parameters in the image backbone LDM (not trained).
- 449 million parameters in the temporal layers (trained).
- 354 million parameters in the OpenCLIP-ViT/H text encoder (not trained).

This sums to a total of 977 parameters in the autoencoder and diffusion model components of the upsampler, out of which only 449 were trained.

We see that compared, for instance, to Imagen Video [27], which has 11.6B parameters, our model is much smaller. Yet, it can produce high quality videos, which we attribute to the efficient LDM framework. CogVideo [32] also has much more parameters, around 9B, than our Video LDM. We suspect that Make-A-Video [76] similarly is a much larger model than ours.

H.3. Details: Personalized Text-to-Video with Dreambooth—Sec. 4.2.1

Using DreamBooth [66], we fine-tune our Stable Diffusion spatial backbone (after fine-tuning on WebVid-10M, as described above) on small sets of images of certain objects (using SD 1.4). We use 256 regularization images and train for 800 iterations using a learning rate 10^{-6}. We found it greatly beneficial to train both the U-Net as well as the CLIP text encoder. Our DreamBooth code is based on the following public codebase: `https://github.com/XavierXiao/Dreambooth-Stable-Diffusion`. After training, we insert the temporal layers from the previously video-tuned Stable Diffusion (without DreamBooth) into the new DreamBooth version of the original Stable Diffusion model. Importantly, for video generation, the spatial layers use the DreamBooth-fine-tuned CLIP text encoder whereas the temporal layers use the standard CLIP text encoder they were trained on.

I. Additional Results

Here we are showing further qualitative and quantitative results, including sampled videos from all our models.

In Appendix I.1, we show more samples from our Stable Diffusion-based text-to-video LDM. This includes samples that were generated by using the model convolutional-in-time as well as convolutional-in-space (see Appendix D). We also discuss video fine-tuning of the decoder for this text-to-video LDM.

In Appendix I.2, we show additional results from a Video LDM model we trained on a dataset of Mountain Bike videos. This includes quantitative comparisons to the previous state-of-the-art Long Video GAN baseline.

In Appendix I.3, we present more qualitative and quantitative results from our main Video LDM that was trained on real-world driving data.

I.1. Text-to-Video

I.1.1 Video-Finetuning of our Decoders

We perform a small ablation experiment over the video fine-tuning of our decoder (as described in Sec. 3.1.1). As can be seen in Table 11, video fine-tuning the decoder allows for a significant performance boost for out text-to-video model, similar to what we have observed for the driving model in the main paper.

I.1.2 More Samples

In this section, we provide additional generated samples from our text-to-video LDMs. We show generated videos at resolution 320×512 (SD 1.4-based models) and at resolution 1280×2048 resolution (SD 2.0-based models with additional

Table 12. Details for personalized text-to-Video with DreamBooth. Text in parentheses is given to the spatial layers but omitted for the temporal layers (as they are not part of the DreamBooth training). Generated samples can be found in Fig. 10.

Model	# of training images	Prompts
Building	13	"(*sks* building), 4k drone flight, high definition"
Car	10	"A (*sks*) car driving in Manhattan"
Frog	23	"A (*sks*) frog ice skating in Central Park on Christmas Eve"
Cat	11	"A (*sks*) cat walking, front view, high definition"
Tea pot	8	"A (*sks*) tea pot in the ocean"

video fine-tuned $4\times$ upscaler). Moreover, we present generated videos that are extended "convolutional in space" and/or "convolutional in time"; see Appendix D. We are able to generate long, high resolution and high frame rate, expressive and artistic videos.

- *Regular video samples from SD 1.4-based Video LDM*: Fig. 11.
- *"Convolutional in space" (SD 1.4-based Video LDM)*: Figs. 12 and 13.
- *"Convolutional in time" (SD 1.4-based Video LDM)*: Fig. 14.
- *"Convolutional in space" and "convolutional in time" (SD 1.4-based Video LDM)*: Figs. 15 and 16.
- *Regular video samples at* 1280×2048 *resolution from SD 2.0-based Video LDM with* $4\times$ *upscaler*: Figs. 17 to 20.
- *Regular video samples at* 1280×2048 *resolution from both SD 2.0-based and SD 2.1-based Video LDMs with* $4\times$ *upscaler*: All samples shown on our project page https://research.nvidia.com/labs/toronto-ai/VideoLDM/.

I.1.3 Personalized Text-to-Video with DreamBooth

We provide additional generated personalized text-to-video samples. The generated videos can be found in Fig. 10. The number of training images and the prompts used for the spatial and temporal layers can be found in Table 12. We see that we are able to successfully generate videos that faithfully include the learnt objects and capture their identity well.

I.2. Mountain Biking Video Synthesis

We conducted additional experiments on the Mountain Biking dataset [6] (see Appendix E.3) downsampled and center-cropped to resolution 256×128. We initially train our model for sparse key frame prediction at 1.875 fps. The temporal interpolation model is trained using 30 fps video data. We train the temporal interpolation model to first scale from 1.875 fps to 7.5 fps, and then to scale from 7.5 fps to 30 fps. We are using one interpolation model with shared parameters for that, providing a conditioning label to indicate to the model which of the two temporal upsampling operations is desired.

We then compare our model with the publicly available model from Long Video GAN (LVG) [6]. We report FID & FVD metrics as well as a human evaluation study in Table 13. We outperform LVG both in FID and human evaluation, but slightly underperform on FVD.

The first-person mountain biking videos have very rapidly changing background details (trees, branches, etc.). LVG cannot create these single-frame realistic details, "smoothening out" the background and therefore resulting in worse FID and also performing worse in the human evaluation study. Our method, on the other hand, has more realistic single frames; however, it slightly struggles to keep the temporal consistency of these details. The FVD metric favors short-term "smoothness" over photorealism, which explains the underperformance of our method in this metric. Generally, FVD is a metric with downsides and should be taken with a grain of salt, as discussed in detail in the Long Video GAN paper itself [6] (their Section 5.3), Overall quality and realism is best judged by human evaluators, where we outperform LVG.

We show generated 10 second (30 fps) mountain biking videos in Figs. 21 to 23.

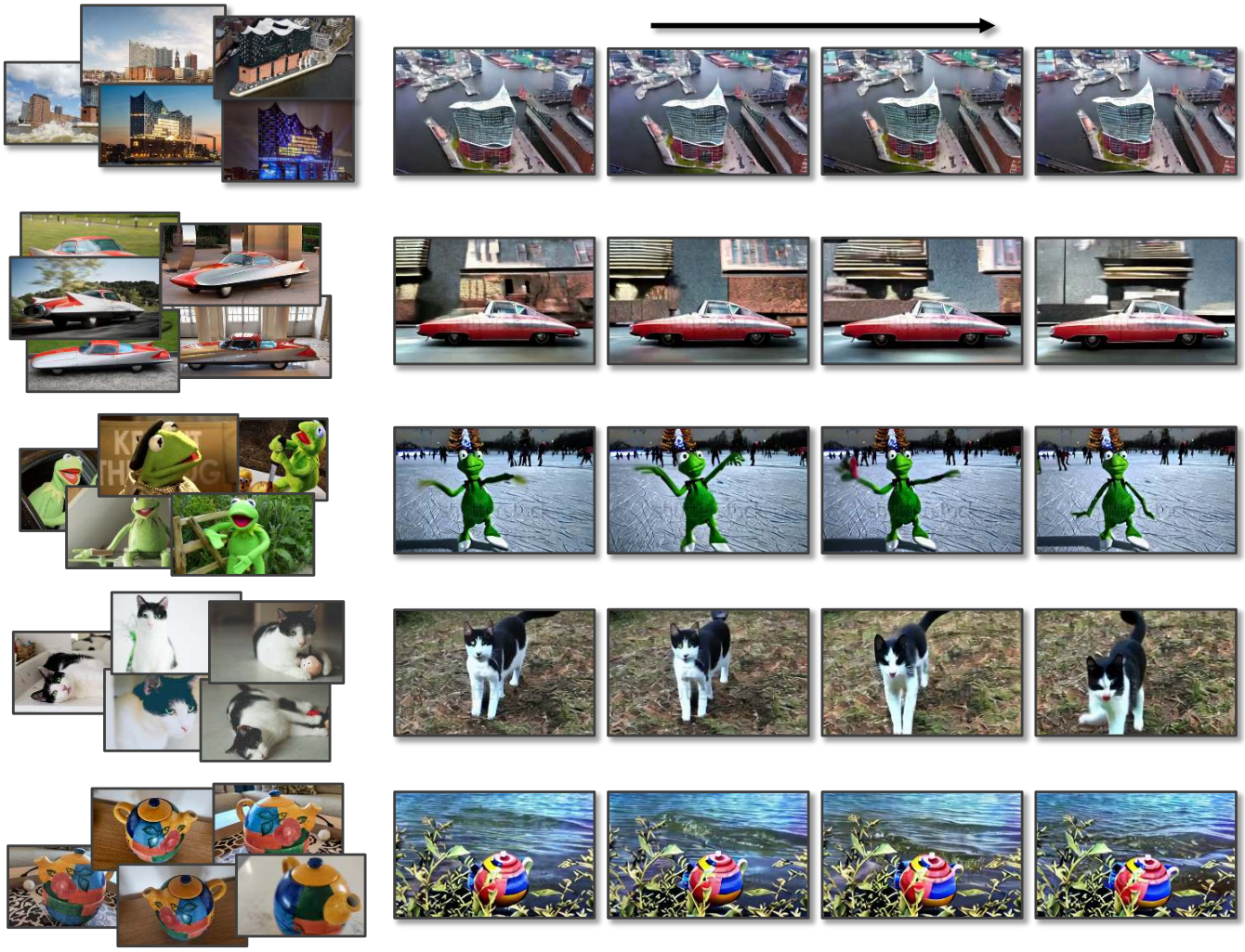

Figure 10. Generated DreamBooth-personalized videos at resolution 320×512. Text prompts of the videos can be found in Table 12. Frames are shown at 1 fps.

Table 13. Comparison with Long Video GAN (LVG) on Mountain Biking videos (human evaluation on the right).

Method	FVD	FID
LVG [6]	85.3	21.1
Video LDM *(ours)*	118	7.73

Method	Pref. A	Pref. B	Equal
Video LDM *(ours)* vs. LVG [6]	54.2	42.2	3.6

I.2.1 Video-Finetuning of our Decoders

We also perform a small ablation experiment over the video fine-tuning of the decoders (as described in Sec. 3.1.1) for the mountain biking Video LDM. As can be seen in Table 11, video fine-tuning the decoder allows for a significant performance boost on mountain biking.

I.3. Driving Video Synthesis

In this section, we provide additional generated samples from our Video LDM trained on real-world driving data. The samples are upsampled to resolution 512×1024 using our temporally aligned video upsampler; see Figs. 24 to 26.

Table 14. Decoder fine-tuning with and without additional image discriminator. We are showing reconstruction FVD and FID scores after decoder fine-tuning using our main Video LDM model for driving scenario video generation.

Method	Reconstruction FVD	Reconstruction FID
Video discriminator only	32.94	9.17
Additional image discriminator	51.01	9.04

I.3.1 Ablation on Additional Image Discriminator for Decoder Fine-Tuning

To fine-tune our decoder (see Sec. 3.1.1 and Sec. 4.1), we tested using not only a 3D-convolutional video discriminator, but to also use an additional image discriminator to maintain and possibly enhance image-level quality. Using our main driving model Video LDM, we evaluated reconstruction FVD and FID scores after decoder fine-tuning using only the video discriminator vs. with an additional image discriminator. The results are shown in Tab. 14. We found that image-level quality, as measured by FID, barely changed, while video quality, as measured by FVD, suffered considerably when an additional image discriminator was used. Consequently, we resorted to using only the video discriminator.

I.3.2 Ablation on Image-level Quality Degradation after Temporal Video Fine-Tuning

Does the image-level quality of the generated outputs of the LDM degrade when the model is fine-tuned for video synthesis? To test this, we measured image-level FID scores using independent frames generated by the image backbone model (setting $\alpha_\phi^i{=}1$) and compared to FID scores based on frames generated by the full Video LDM after learning the α_ϕ^i parameters and the temporal alignment layers on videos. For this experiment, we used the smaller version of the Video LDM for driving video generation that was used in our ablation experiments (Sec. 4.1.1). With $\alpha_\phi{=}1$, we obtain 47.00 FID; with the learnt parameters, we get 48.26 FID. We observe only a tiny degradation and conclude that image-level quality is affected only slightly when training the temporal layers for video generation.

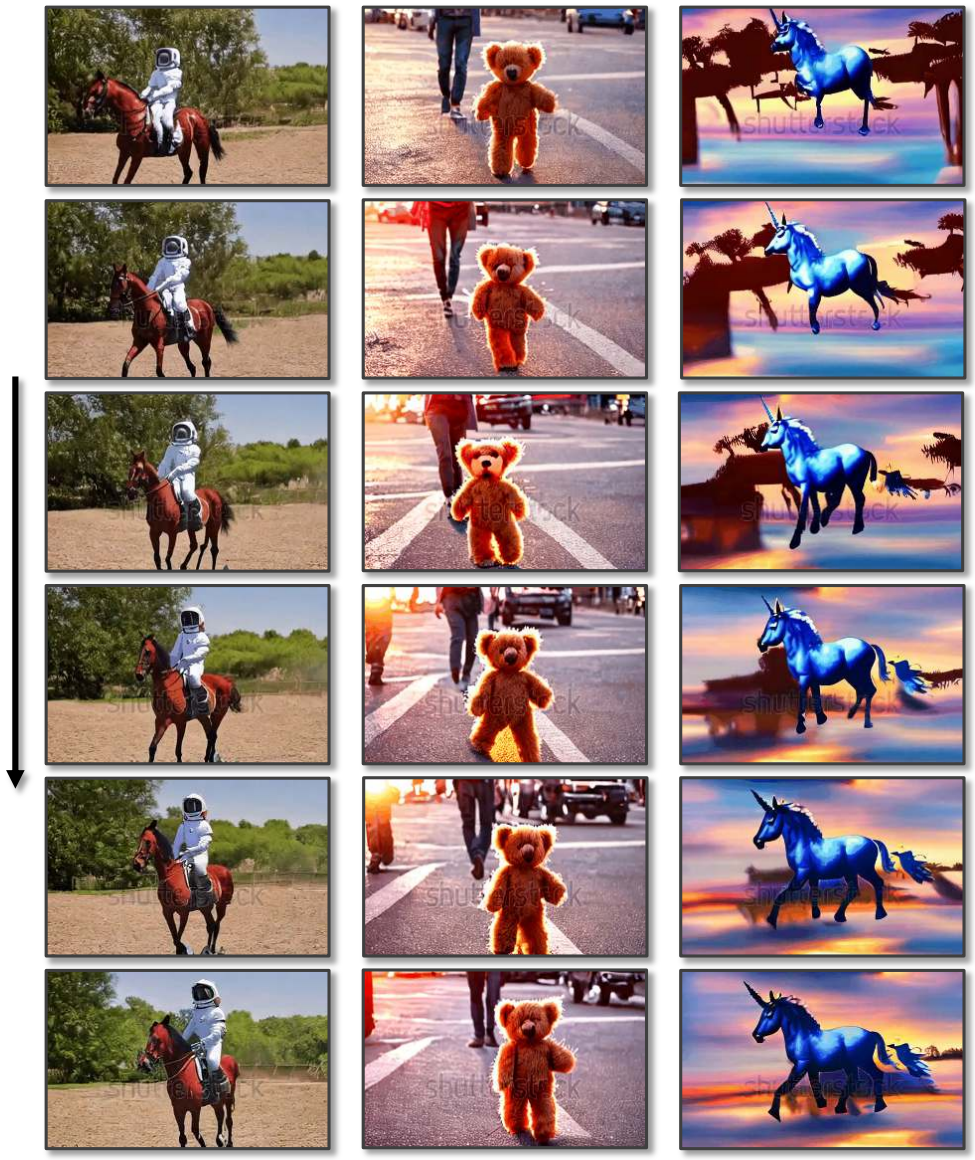

Figure 11. Generated videos at resolution 320×512. Captions from left to right are: "An astronaut riding a horse, high definition, 4k", "Teddy bear walking down 5th Avenue, front view, beautiful sunset, close up, high definition, 4k", and "A blue unicorn flying over a fantasy landscape, animated oil on canvas". Frames are shown at 2 fps.

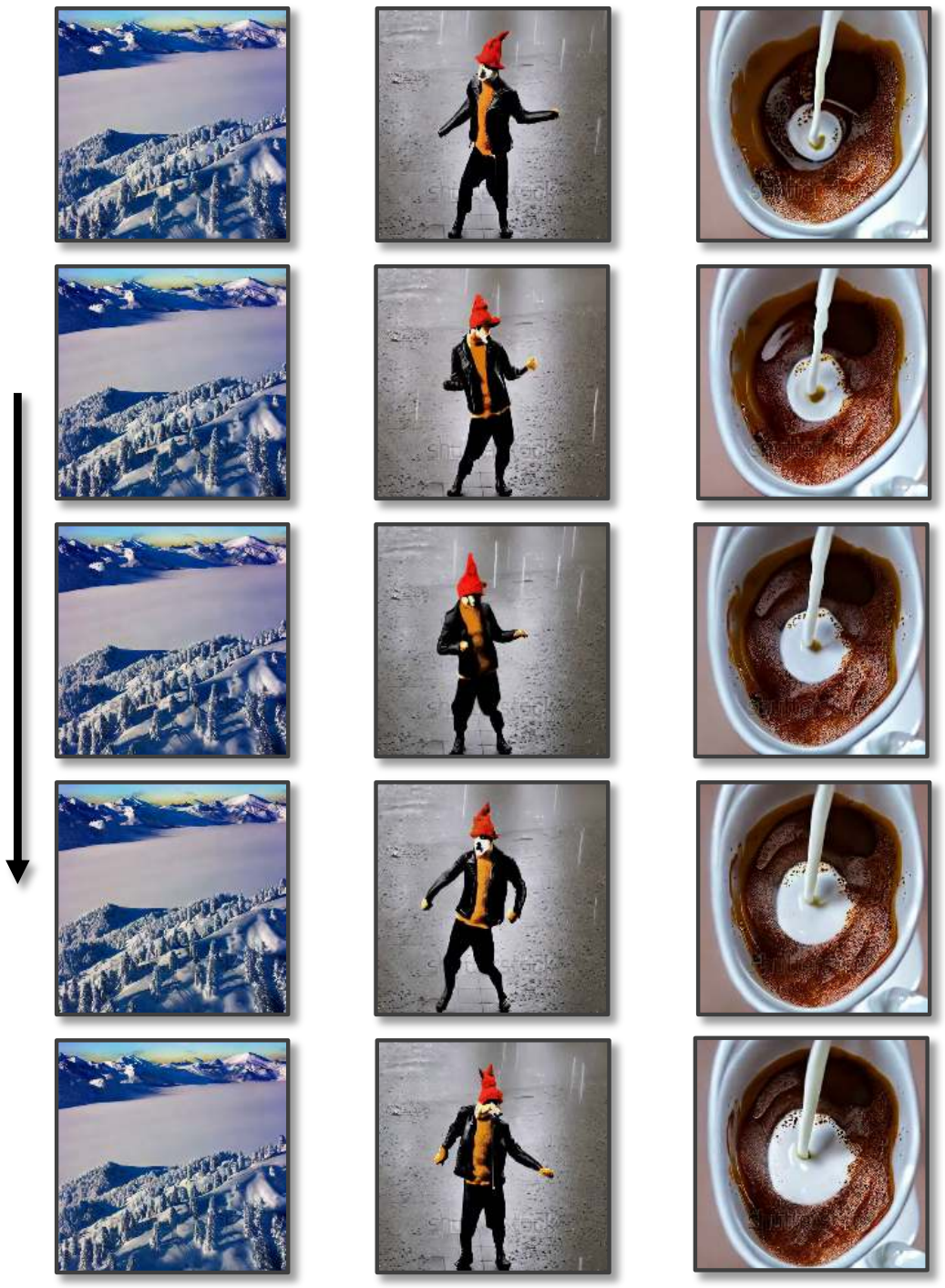

Figure 12. Generated videos at resolution 512×512 (extended "convolutional in space"; see Appendix D). Captions from left to right are: "Aerial view over snow covered mountains", "A fox wearing a red hat and a leather jacket dancing in the rain, high definition, 4k", and "Milk dripping into a cup of coffee, high definition, 4k". Frames are shown at 2 fps.

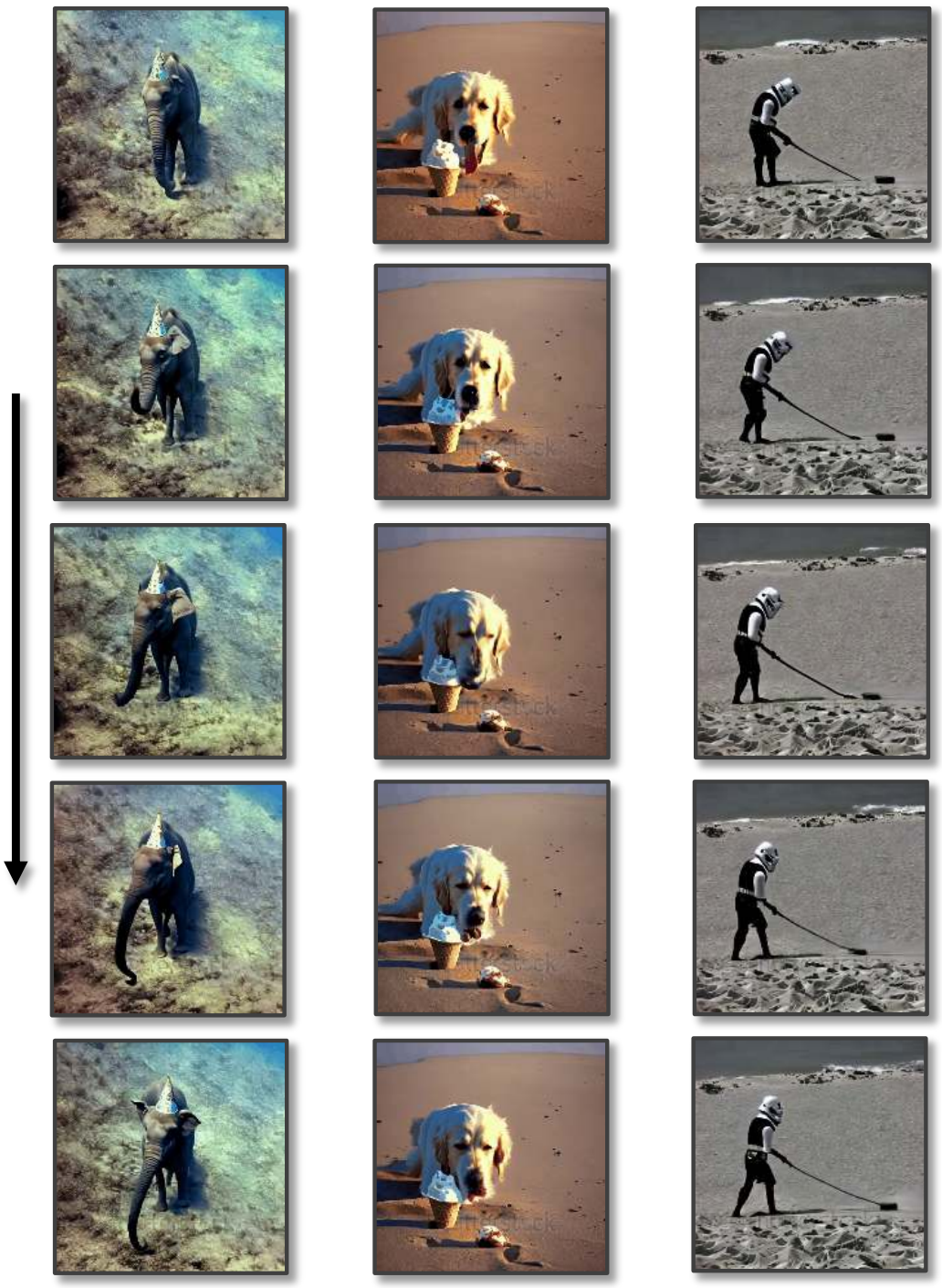

Figure 13. Generated videos at resolution 512×512 (extended "convolutional in space"; see Appendix D). Captions from left to right are: "An elephant wearing a birthday hat walking under the sea", "A golden retriever eating ice cream on a beautiful tropical beach at sunset, high resolution", and "A storm trooper vacuuming the beach". Frames are shown at 2 fps.

Figure 14. Generated videos at resolution 320×512 (extended "convolutional in time" to 8 seconds each; see Appendix D). Captions from left to right are: "A teddy bear wearing sunglasses and a leather jacket is headbanging while playing the electric guitar, high definition, 4k", "An ancient greek statue on a crowded square suddenly becomes alive and starts to walk, high definition, 4k", and "A teddy bear wearing sunglasses playing the electric guitar, high definition, 4k". Frames are shown at 1 fps.

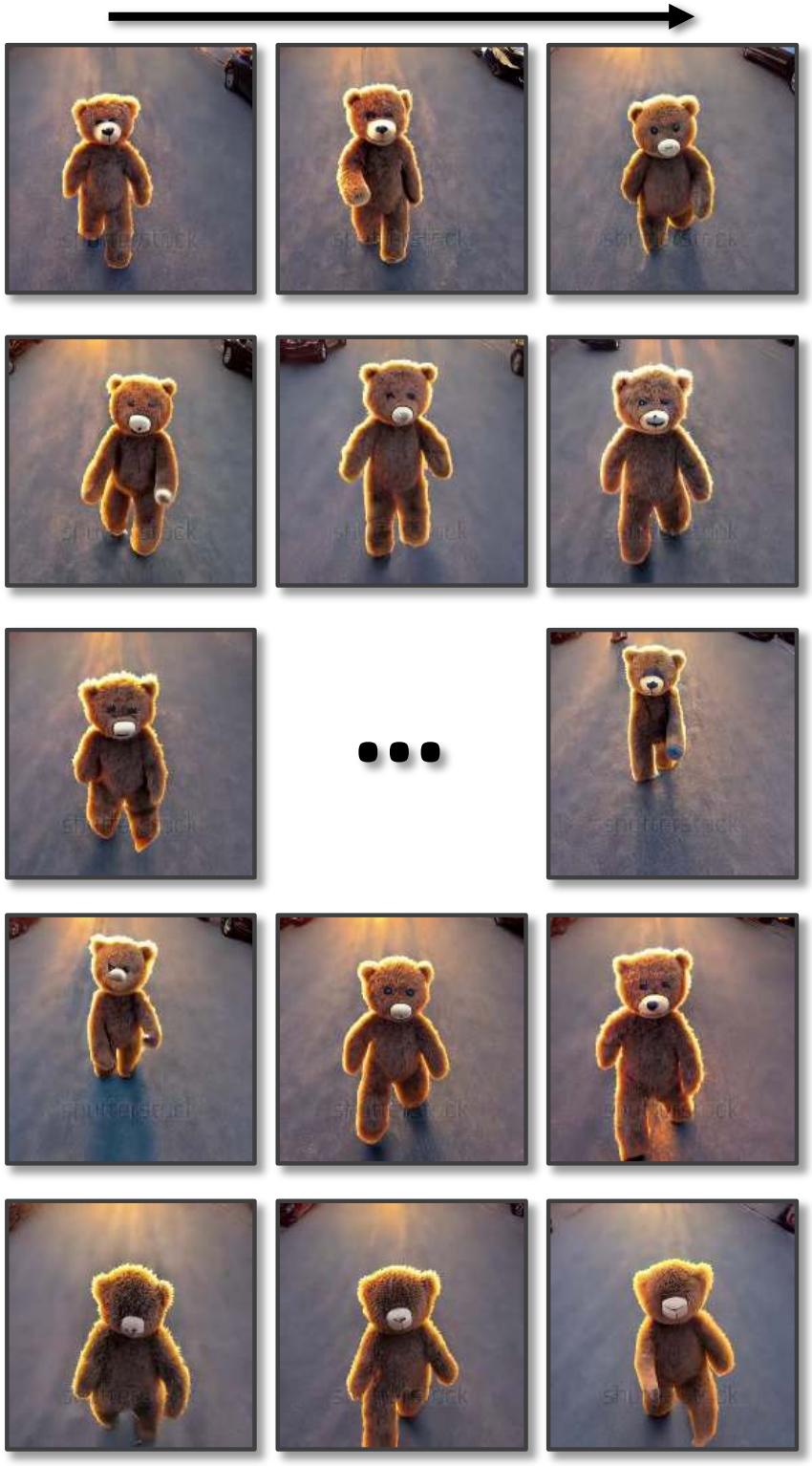

Figure 15. Generated 30 second video of "a teddy bear walking down the road in the sunset, high definition, 4k" at resolution 512×512 (extended "convolutional in space" and also "convolutional in time"; see Appendix D). Frames are shown at 1 fps.

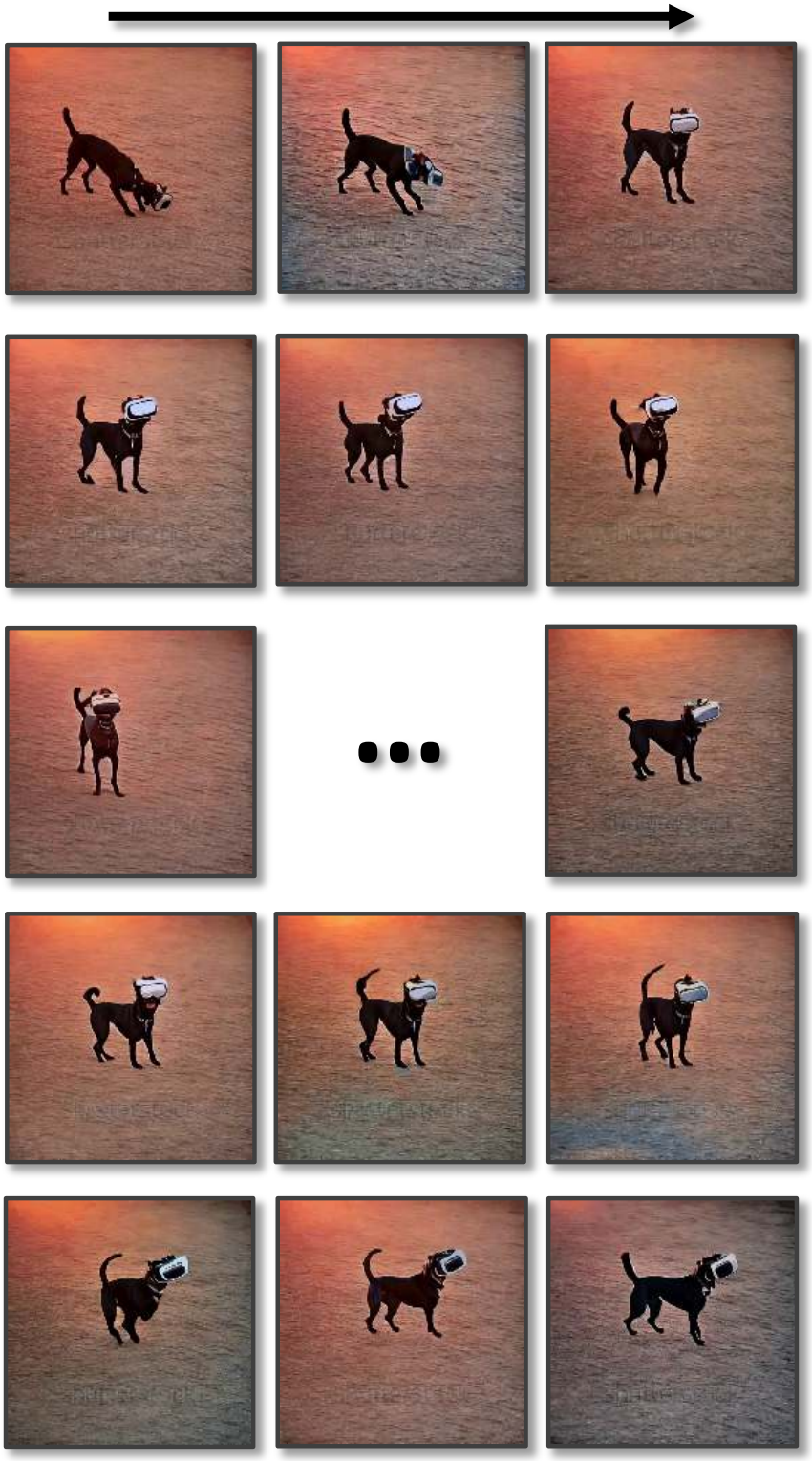

Figure 16. Generated 8 second video of "a dog wearing virtual reality goggles playing in the sun, high definition, 4k" at resolution 512×512 (extended "convolutional in space" and "convolutional in time"; see Appendix D). Frames are shown at 4 fps.

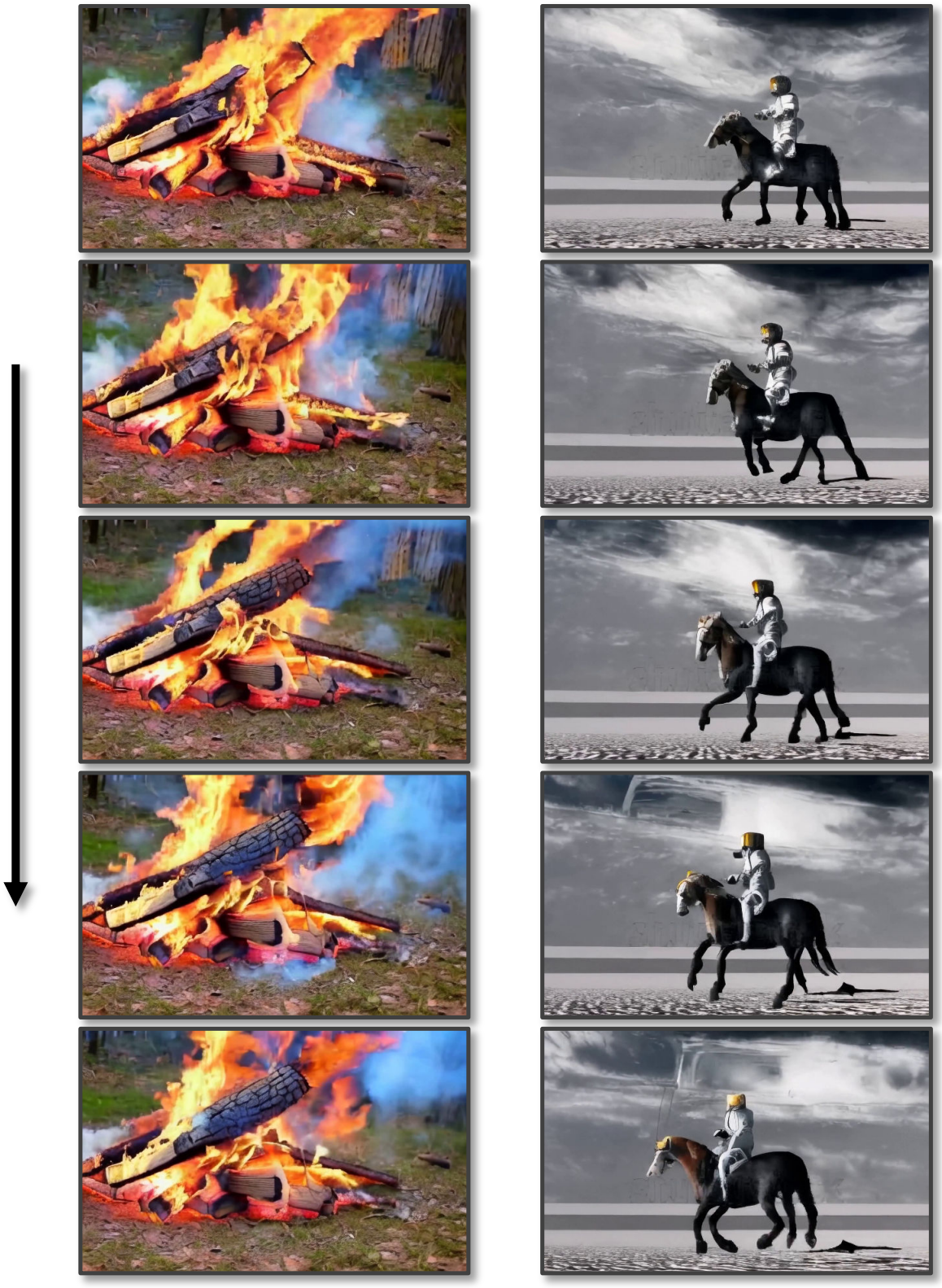

Figure 17. Generated videos at resolution 1280×2048 using our Stable Diffusion 2.0-based model and including our video fine-tuned text-to-video latent upsampler. Captions from left to right are: “Burning firewood” and “An astronaut riding a horse, 4k, high definition”. Frames are shown at 2 fps.

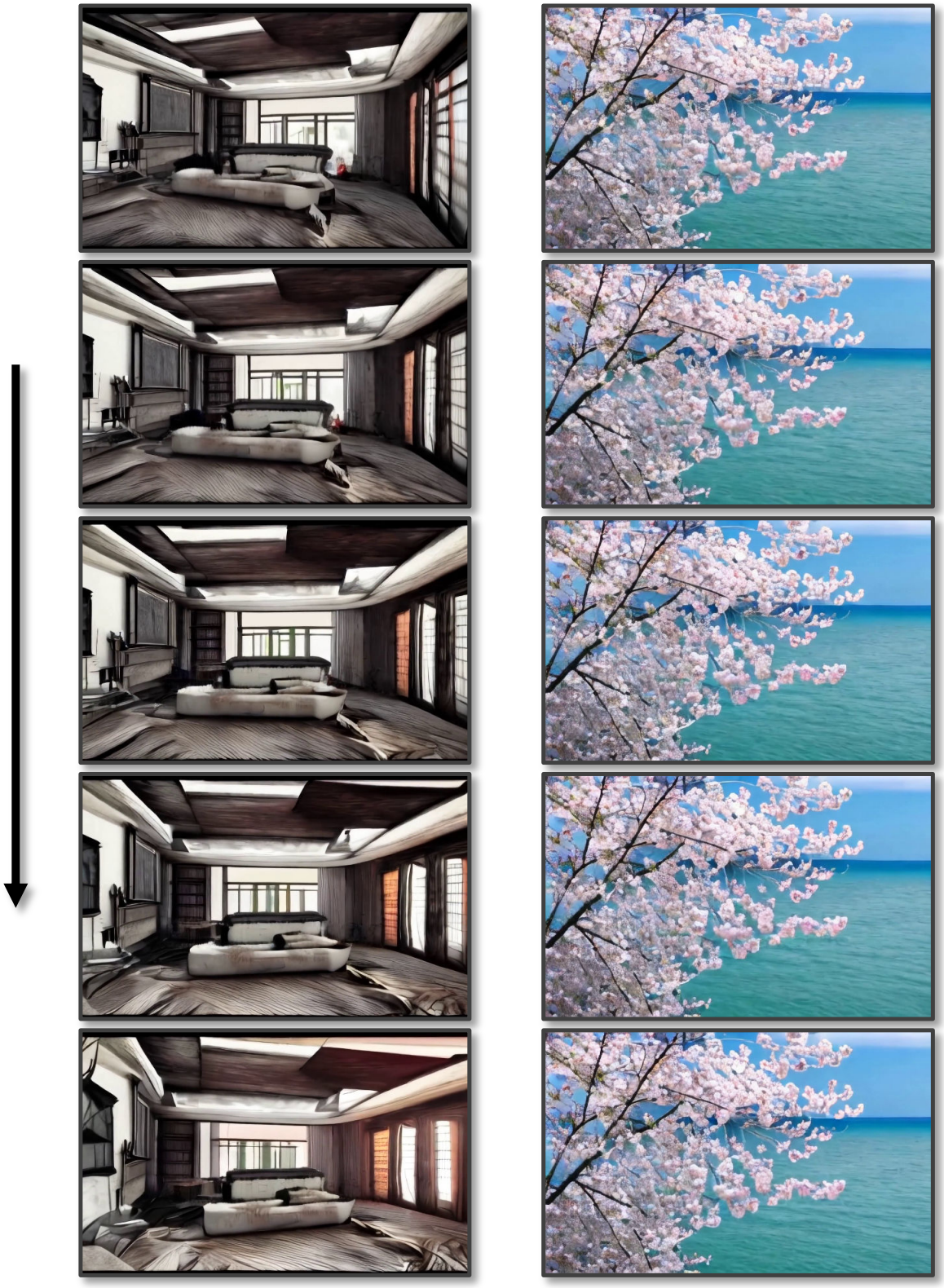

Figure 18. Generated videos at resolution 1280×2048 using our Stable Diffusion 2.0-based model and including our video fine-tuned text-to-video latent upsampler. Captions from left to right are: "horror house living room interior overview design, Moebius, Greg Rutkowski, Zabrocki, Karlkka, Jayison Devadas, Phuoc Quan, trending on Artstation, 8K, ultra wide angle, pincushion lens effect." and "Cherry blossom swing in front of ocean view, 4k, high resolution". Frames are shown at 2 fps.

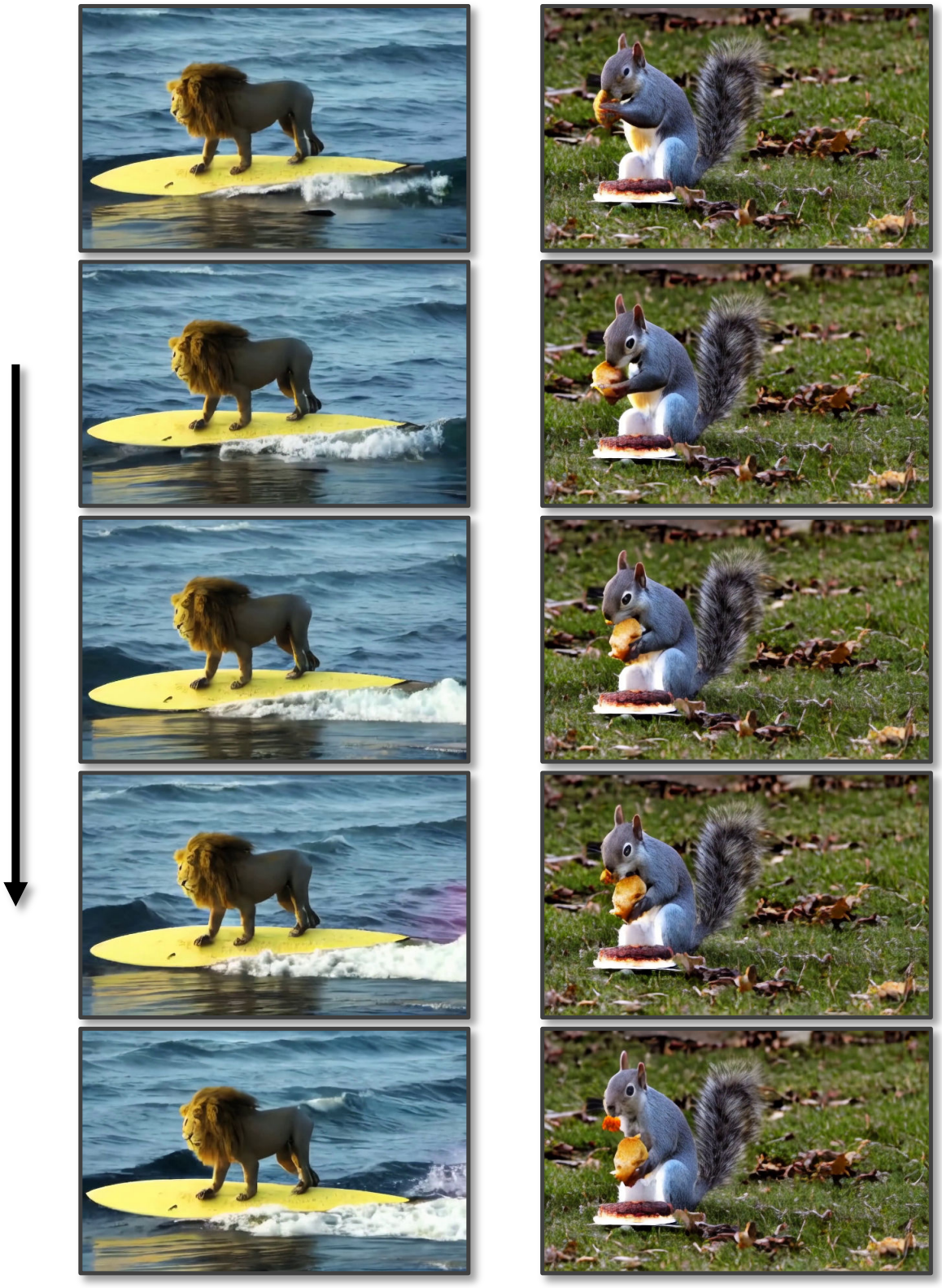

Figure 19. Generated videos at resolution 1280×2048 using our Stable Diffusion 2.0-based model and including our video fine-tuned text-to-video latent upsampler. Captions from left to right are: “A lion standing on a surfboard in the ocean in sunset, 4k, high resolution” and “A squirrel eating a burger”. Frames are shown at 2 fps.

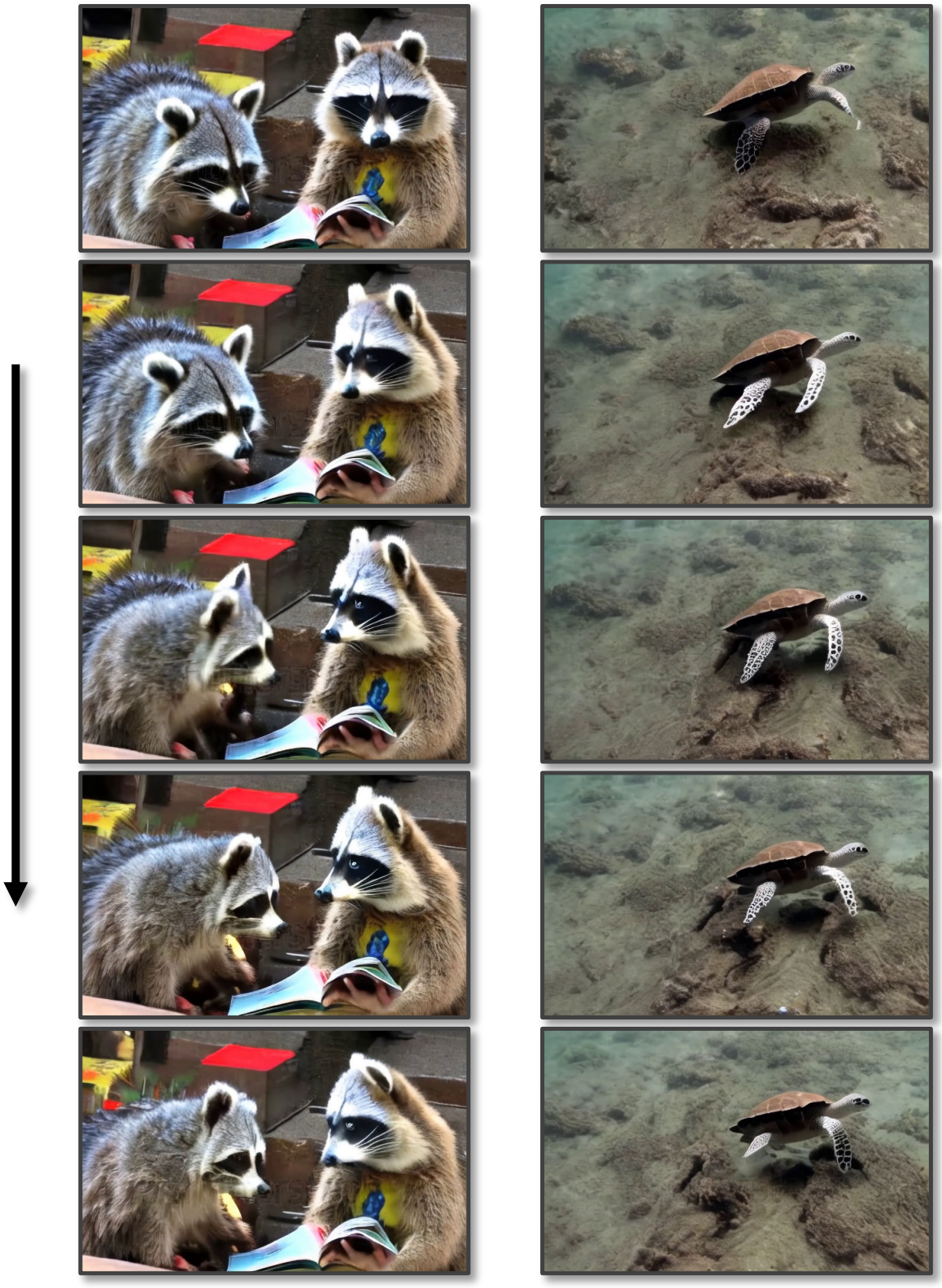

Figure 20. Generated videos at resolution 1280×2048 using our Stable Diffusion 2.0-based model and including our video fine-tuned text-to-video latent upsampler. Captions from left to right are: “Two raccoons reading books in NYC Times Square” and “Turtle swims in ocean”. Frames are shown at 2 fps.

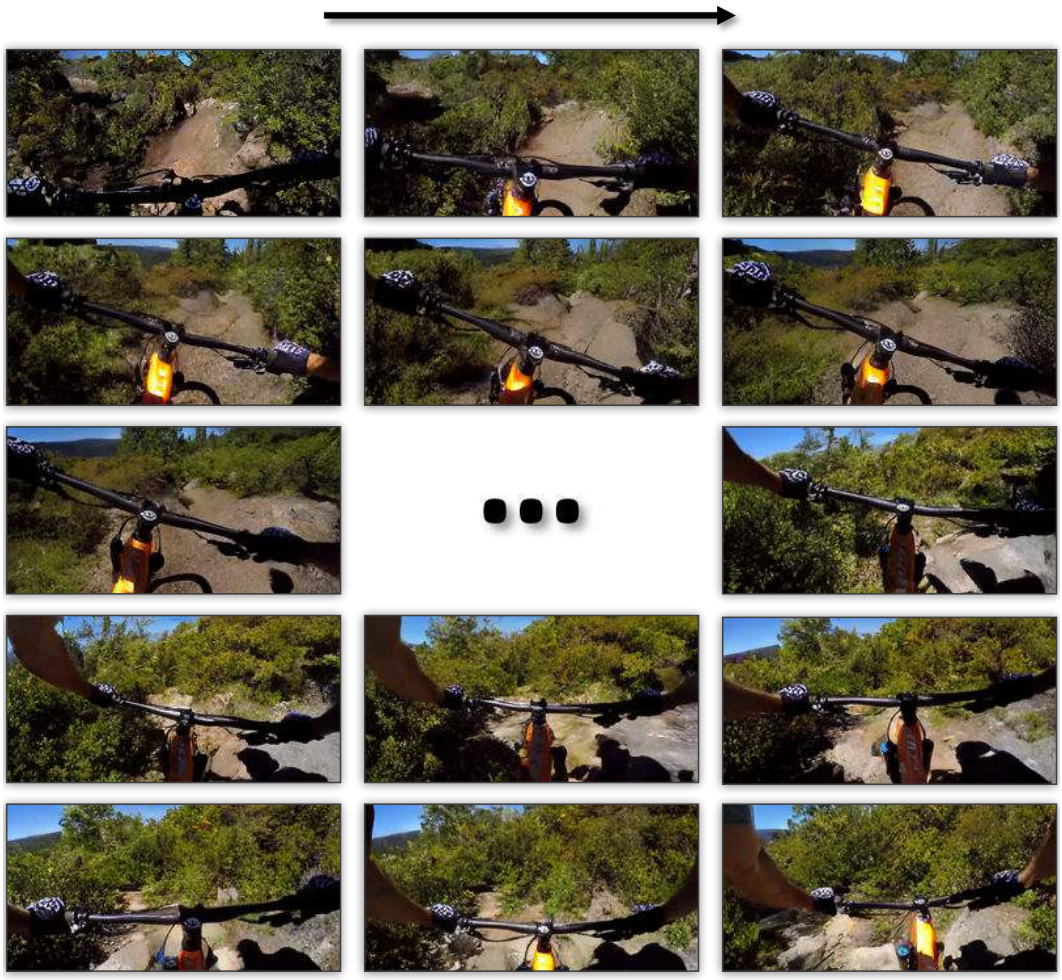

Figure 21. Generated 10 second (30 fps) Mountain Biking video at resolution 128×256. Frames are shown at 6 fps.

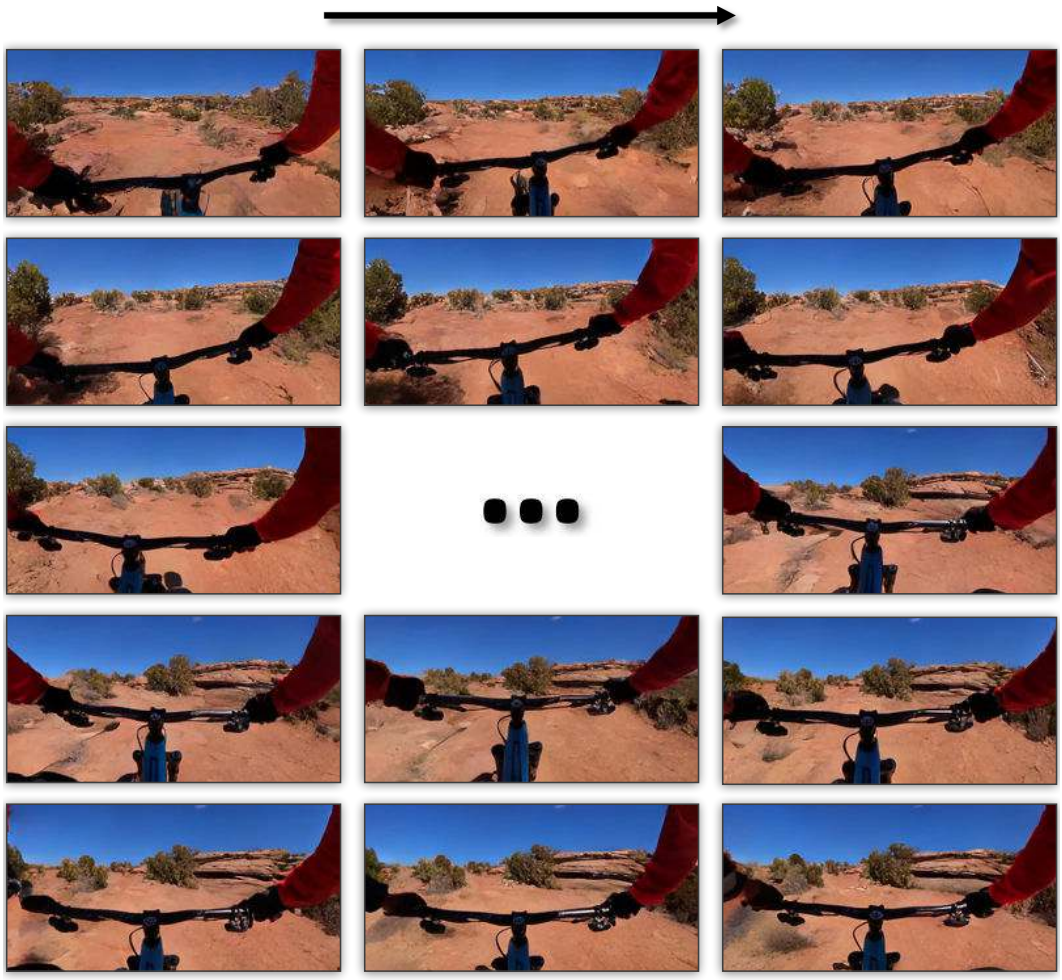

Figure 22. Generated 10 second (30 fps) Mountain Biking video at resolution 128×256. Frames are shown at 6 fps.

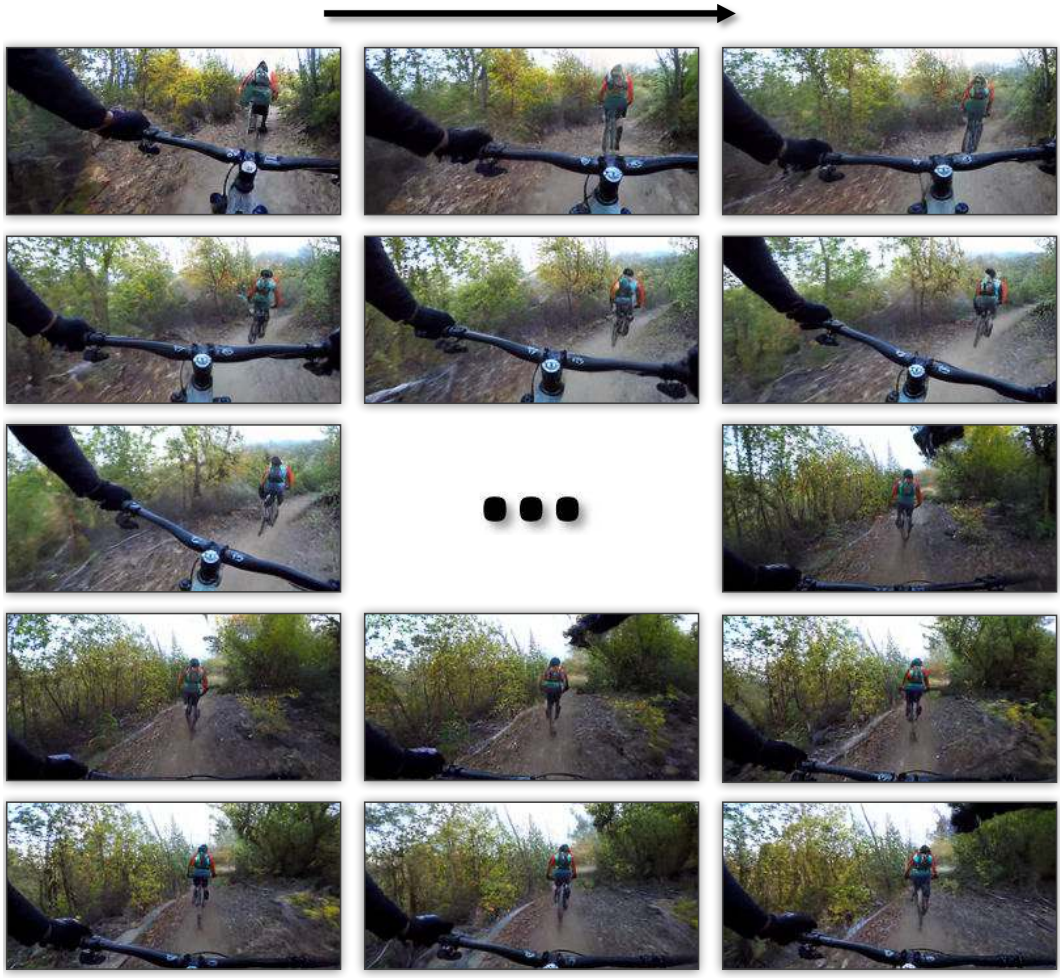

Figure 23. Generated 10 second (30 fps) Mountain Biking video at resolution 128×256. Frames are shown at 6 fps.

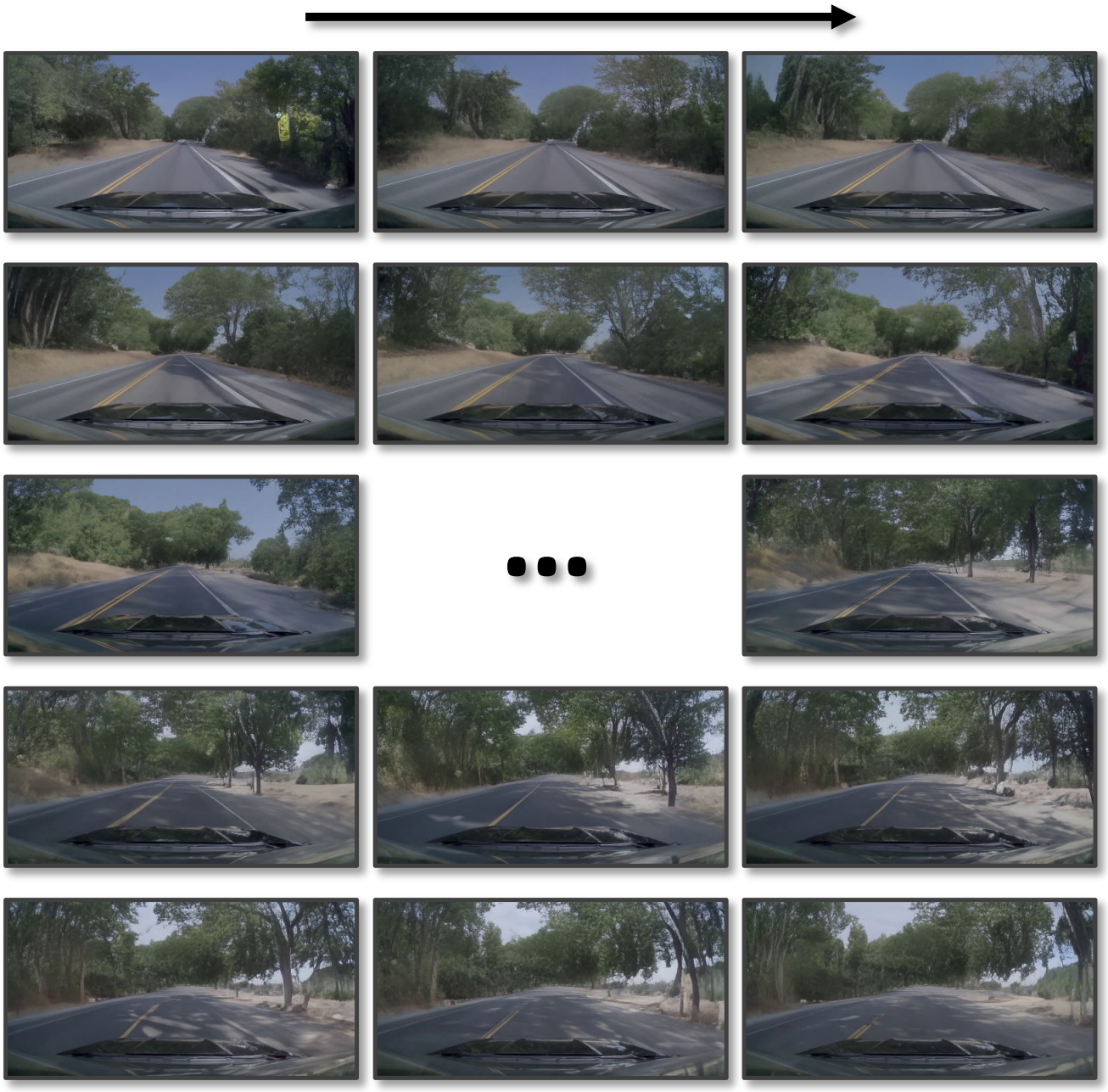

Figure 24. Generated Driving video upsampled to resolution 512×1024. Frames are shown at 2 fps.

Figure 25. Generated Driving video upsampled to resolution 512×1024. Frames are shown at 2 fps.

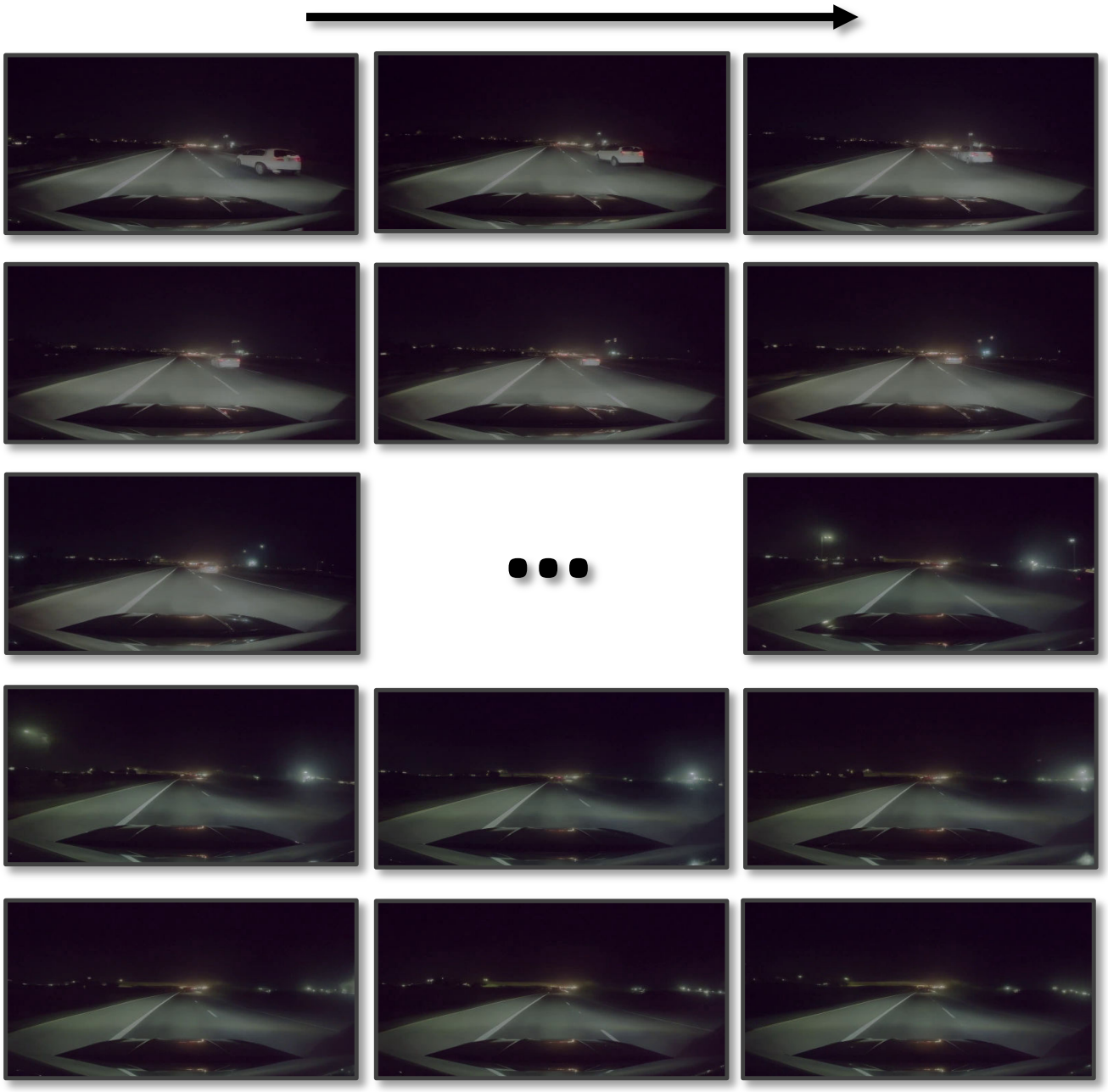

Figure 26. Generated Driving video upsampled to resolution 512×1024. Frames are shown at 2 fps.