Diffusion Models for Video Generation

This PDF conversion (v1) was generated on June 21, 2025*

Online version:
https://lilianweng.github.io/posts/2024-04-12-diffusion-video/
Date: 2024-04-12 **Estimated Reading Time:** 20 min **Author:** Lilian Weng

Diffusion models have demonstrated strong results on image synthesis in past years. Now the research community has started working on a harder task—using it for video generation. The task itself is a superset of the image case, since an image is a video of 1 frame, and it is much more challenging because:

1. It has extra requirements on temporal consistency across frames in time, which naturally demands more world knowledge to be encoded into the model.
2. In comparison to text or images, it is more difficult to collect large amounts of high-quality, high-dimensional video data, let alone text-video pairs.

Required Pre-read: Please make sure you have read the previous blog on "What are Diffusion Models?" for image generation before continuing here.

[0]This file was prepared by Sakura (bili_sakura@zju.edu.cn). Please contact me for any issues or copyright concerns. The original source of this file is https://lilianweng.github.io/posts/2024-04-12-diffusion-video/.

Contents

1 Video Generation Modeling from Scratch

First let's review approaches for designing and training diffusion video models from scratch, meaning that we do not rely on pre-trained image generators.

1.1 Parameterization & Sampling Basics

Here we use a slightly different variable definition from the previous post, but the math stays the same. Let $\mathbf{x} \sim q_{\text{real}}$ be a data point sampled from the real data distribution. Now we are adding Gaussian noise in small amount in time, creating a sequence of noisy variations of $\mathbf{x}$, denoted as $\{\mathbf{z}_t \mid t = 1 \dots, T\}$, with increasing amount of noise as t increases and the last $q(\mathbf{z}_T) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The noise-adding forward process is a Gaussian process. Let α_t, σ_t define a differentiable noise schedule of the Gaussian process:

$$q(\mathbf{z}_t|\mathbf{x}) = \mathcal{N}(\mathbf{z}_t; \alpha_t\mathbf{x}, \sigma_t^2\mathbf{I})$$

To represent $q(\mathbf{z}_t|\mathbf{z}_s)$ for $0 \le s < t \le T$, we have:

$$\begin{aligned}
\mathbf{z}_t &= \alpha_t\mathbf{x} + \sigma_t\boldsymbol{\epsilon}_t \\
\mathbf{z}_s &= \alpha_s\mathbf{x} + \sigma_s\boldsymbol{\epsilon}_s \\
\mathbf{z}_t &= \alpha_t\Big(\frac{\mathbf{z}_s - \sigma_s\boldsymbol{\epsilon}_s}{\alpha_s}\Big) + \sigma_t\boldsymbol{\epsilon}_t \\
\mathbf{z}_t &= \frac{\alpha_t}{\alpha_s}\mathbf{z}_s + \sigma_t\boldsymbol{\epsilon}_t - \frac{\alpha_t\sigma_s}{\alpha_s}\boldsymbol{\epsilon}_s \\
\text{Thus } q(\mathbf{z}_t|\mathbf{z}_s) &= \mathcal{N}\Big(\mathbf{z}_t; \frac{\alpha_t}{\alpha_s}\mathbf{z}_s, \big(1 - \frac{\alpha_t^2\sigma_s^2}{\sigma_t^2\alpha_s^2}\big)\sigma_t^2\mathbf{I}\Big)
\end{aligned}$$

Let the log signal-to-noise-ratio be $\lambda_t = \log[\alpha_t^2/\sigma_t^2]$, we can represent the DDIM (Song et al. 2020) update as:

$$q(\mathbf{z}_t|\mathbf{z}_s) = \mathcal{N}\Big(\mathbf{z}_t; \frac{\alpha_t}{\alpha_s}\mathbf{z}_s, \sigma_{t|s}^2\mathbf{I}\Big) \quad \text{where } \sigma_{t|s}^2 = (1 - e^{\lambda_t - \lambda_s})\sigma_t^2$$

There is a special $\mathbf{v}$-prediction ($\mathbf{v} = \alpha_t\boldsymbol{\epsilon} - \sigma_t\mathbf{x}$) parameterization, proposed by Salimans & Ho (2022). It has been shown to be helpful for avoiding color shift in video generation compared to $\boldsymbol{\epsilon}$-parameterization.

The $\mathbf{v}$-parameterization is derived with a trick in the angular coordinate. First, we define $\phi_t = \arctan(\sigma_t/\alpha_t)$ and thus we have $\alpha_\phi = \cos\phi, \sigma_t = \sin\phi, \mathbf{z}_\phi = \cos\phi\mathbf{x} + \sin\phi\boldsymbol{\epsilon}$. The velocity of $\mathbf{z}_\phi$ can be written as:

$$\mathbf{v}_\phi = \nabla_\phi\mathbf{z}_\phi = \frac{d\cos\phi}{d\phi}\mathbf{x} + \frac{d\sin\phi}{d\phi}\boldsymbol{\epsilon} = \cos\phi\boldsymbol{\epsilon} - \sin\phi\mathbf{x}$$

Then we can infer,

$$\begin{aligned}
\sin\phi\mathbf{x} &= \cos\phi\boldsymbol{\epsilon} - \mathbf{v}_\phi \\
&= \frac{\cos\phi}{\sin\phi}\big(\mathbf{z}_\phi - \cos\phi\mathbf{x}\big) - \mathbf{v}_\phi \\
\sin^2\phi\mathbf{x} &= \cos\phi\mathbf{z}_\phi - \cos^2\phi\mathbf{x} - \sin\phi\mathbf{v}_\phi \\
\mathbf{x} &= \cos\phi\mathbf{z}_\phi - \sin\phi\mathbf{v}_\phi \\
\text{Similarly } \boldsymbol{\epsilon} &= \sin\phi\mathbf{z}_\phi + \cos\phi\mathbf{v}_\phi
\end{aligned}$$

The DDIM update rule is updated accordingly,

$$\begin{aligned}
\mathbf{z}_{\phi_s} &= \cos\phi_s \hat{\mathbf{x}}_\theta(\mathbf{z}_{\phi_t}) + \sin\phi_s \hat{\epsilon}_\theta(\mathbf{z}_{\phi_t}) \\
&= \cos\phi_s \big(\cos\phi_t \mathbf{z}_{\phi_t} - \sin\phi_t \hat{\mathbf{v}}_\theta(\mathbf{z}_{\phi_t})\big) + \sin\phi_s \big(\sin\phi_t \mathbf{z}_{\phi_t} + \cos\phi_t \hat{\mathbf{v}}_\theta(\mathbf{z}_{\phi_t})\big) \\
&= \big(\cos\phi_s \cos\phi_t + \sin\phi_s \sin\phi_t\big)\mathbf{z}_{\phi_t} + \big(\sin\phi_s \cos\phi_t - \cos\phi_s \sin\phi_t\big)\hat{\mathbf{v}}_\theta(\mathbf{z}_{\phi_t}) \\
&= \cos(\phi_s - \phi_t)\mathbf{z}_{\phi_t} + \sin(\phi_s - \phi_t)\hat{\mathbf{v}}_\theta(\mathbf{z}_{\phi_t})
\end{aligned}$$

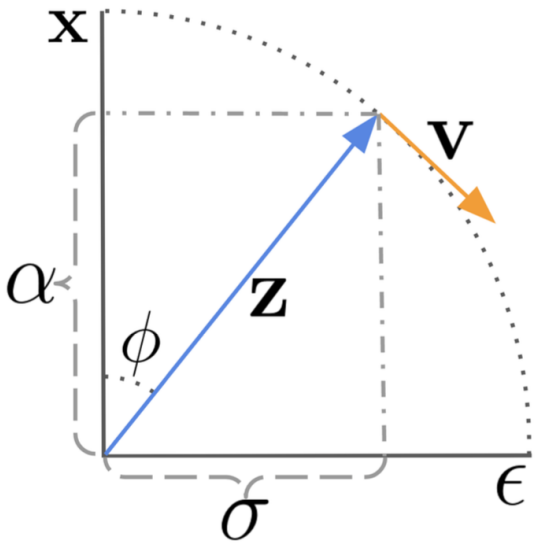

Figure 1: Visualizing how the diffusion update step works in the angular coordinate, where DDIM evolves $\mathbf{z}_{\phi_s}$ by moving it along the $-\hat{\mathbf{v}}_{\phi_t}$ direction. (Image source: Salimans & Ho, 2022)

The **v**-parameterization for the model is to predict $\mathbf{v}_\phi = \cos\phi\boldsymbol{\epsilon} - \sin\phi\mathbf{x} = \alpha_t\boldsymbol{\epsilon} - \sigma_t\mathbf{x}$.

In the case of video generation, we need the diffusion model to run multiple steps of upsampling for extending video length or increasing the frame rate. This requires the capability of sampling a second video $\mathbf{x}^b$ conditioned on the first $\mathbf{x}^a$, $\mathbf{x}^b \sim p_\theta(\mathbf{x}^b|\mathbf{x}^a)$, where $\mathbf{x}^b$ might be an autoregressive extension of $\mathbf{x}^a$ or be the missing frames in-between for a video $\mathbf{x}^a$ at a low frame rate.

The sampling of $\mathbf{x}_b$ needs to condition on $\mathbf{x}_a$ besides its own corresponding noisy variable. **Video Diffusion Models** (**VDM**; Ho & Salimans, et al. 2022) proposed the *reconstruction guidance* method using an adjusted denoising model such that the sampling of $\mathbf{x}^b$ can be properly conditioned on $\mathbf{x}^a$:

$$\begin{aligned}
\mathbb{E}_q[\mathbf{x}_b|\mathbf{z}_t, \mathbf{x}^a] &= \mathbb{E}_q[\mathbf{x}^b|\mathbf{z}_t] + \frac{\sigma_t^2}{\alpha_t}\nabla_{\mathbf{z}_t^b}\log q(\mathbf{x}^a|\mathbf{z}_t) \\
q(\mathbf{x}^a|\mathbf{z}_t) &\approx \mathcal{N}\big[\hat{\mathbf{x}}_\theta^a(\mathbf{z}_t), \frac{\sigma_t^2}{\alpha_t^2}\mathbf{I}\big] \\
\tilde{\mathbf{x}}_\theta^b(\mathbf{z}_t) &= \hat{\mathbf{x}}_\theta^b(\mathbf{z}_t) - \frac{w_r\alpha_t}{2}\nabla_{\mathbf{z}_t^b}\|\mathbf{x}^a - \hat{\mathbf{x}}_\theta^a(\mathbf{z}_t)\|_2^2
\end{aligned}$$

where $\hat{\mathbf{x}}_\theta^a(\mathbf{z}_t), \hat{\mathbf{x}}_\theta^b(\mathbf{z}_t)$ are reconstructions of $\mathbf{x}^a, \mathbf{x}^b$ provided by the denoising model. And w_r is a weighting factor and a large one $w_r > 1$ is found to improve sample quality. Note that it is also possible to simultaneously condition on low resolution videos to extend samples to be at the high resolution using the same reconstruction guidance method.

1.2 Model Architecture: 3D U-Net & DiT

Similar to text-to-image diffusion models, U-net and Transformer are still two common architecture choices. There are a series of diffusion video modeling papers from Google based on the U-net architecture and a recent Sora model from OpenAI leveraged the Transformer architecture.

VDM (Ho & Salimans, et al. 2022) adopts the standard diffusion model setup but with an altered architecture suitable for video modeling. It extends the 2D U-net to work for 3D data (Cicek et al. 2016), where each feature map represents a 4D tensor of frames × height × width × channels. This 3D U-net is factorized over space and time, meaning that each layer only operates on the space or time dimension, but not both:

- **Processing Space:**
  - Each old 2D convolution layer as in the 2D U-net is extended to be space-only 3D convolution; precisely, 3x3 convolutions become 1x3x3 convolutions.
  - Each spatial attention block remains as attention over space, where the first axis (frames) is treated as batch dimension.
- **Processing Time:**
  - A temporal attention block is added after each spatial attention block. It performs attention over the first axis (frames) and treats spatial axes as the batch dimension. The relative position embedding is used for tracking the order of frames. The temporal attention block is important for the model to capture good temporal coherence.

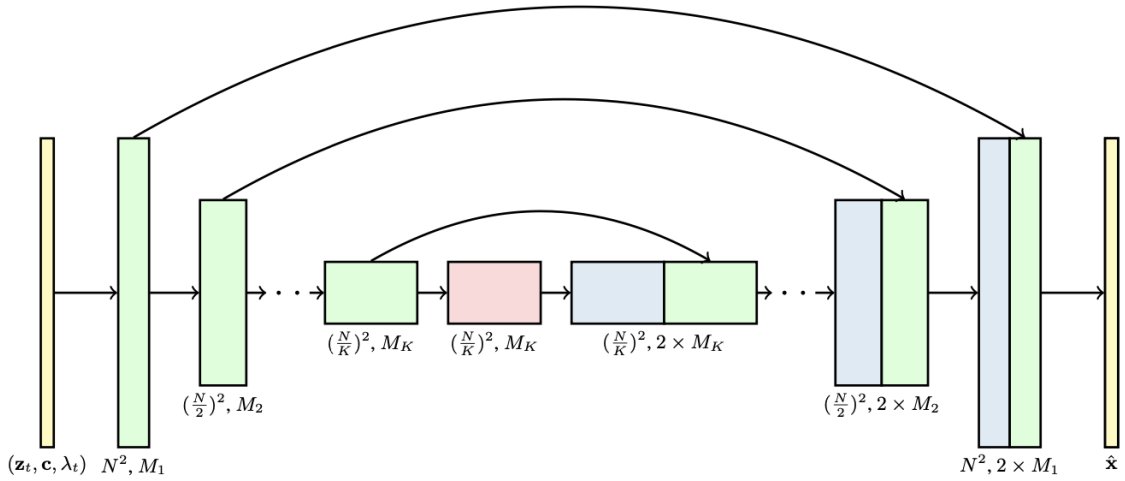

Figure 2: The 3D U-net architecture. The noisy video $\mathbf{z}_t$, conditioning information $\boldsymbol{c}$ and the log signal-to-noise ratio (log-SNR) λ_t are inputs to the network. The channel multipliers $M_1, \ldots, M_K$ represent the channel counts across layers. (Image source: Salimans & Ho, 2022)

Imagen Video (Ho, et al. 2022) is constructed on a cascade of diffusion models to enhance the video generation quality and upgrades to output 1280x768 videos at 24 fps.

The Imagen Video architecture consists of the following components, counting 7 diffusion models in total.

- A frozen T5 text encoder to provide text embedding as the conditioning input.
- A base video diffusion model.
- A cascade of interleaved spatial and temporal super-resolution diffusion models, including 3 TSR (Temporal Super-Resolution) and 3 SSR (Spatial Super-Resolution) components.

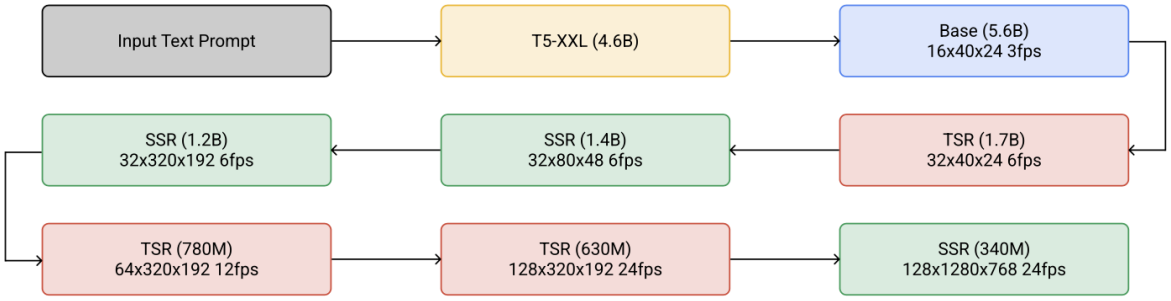

Figure 3: The cascaded sampling pipeline in Imagen Video. In practice, the text embeddings are injected into all components, not just the base model. (Image source: Ho et al. 2022)

The base denoising models performs spatial operations over all the frames with shared parameters simultaneously and then the temporal layer mixes activations across frames to better capture temporal coherence, which is found to work better than frame-autoregressive approaches.

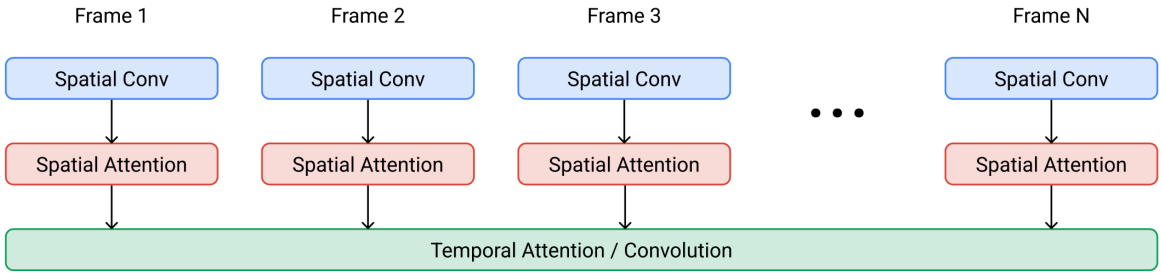

Figure 4: The architecture of one space-time separable block in the Imagen Video diffusion model. (Image source: Ho et al. 2022)

Both SSR and TSR models condition on the upsampled inputs concatenated with noisy data $\mathbf{z}_t$ channel-wise. SSR upsamples by bilinear resizing, while TSR upsamples by repeating the frames or filling in blank frames.

Imagen Video also applies progressive distillation to speed up sampling and each distillation iteration can reduce the required sampling steps by half. Their experiments

were able to distill all 7 video diffusion models down to just 8 sampling steps per model without any noticeable loss in perceptual quality.

To achieve better scaling efforts, **Sora** (Brooks et al. 2024) leverages DiT (Diffusion Transformer) architecture that operates on spacetime patches of video and image latent codes. Visual input is represented as a sequence of spacetime patches which act as Transformer input tokens.

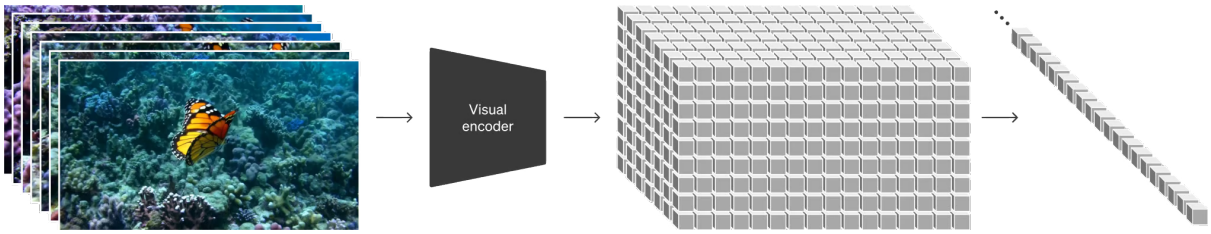

Figure 5: Sora is a diffusion transformer model. (Image source: Brooks et al. 2024)

2 Adapting Image Models to Generate Videos

Another prominent approach for diffusion video modeling is to "inflate" a pre-trained image-to-text diffusion model by inserting temporal layers and then we can choose to *only* fine-tune new layers on video data, or avoid extra training at all. The prior knowledge of text-image pairs is inherited by the new model and thus it can help alleviate the requirement on text-video pair data.

2.1 Fine-tuning on Video Data

Make-A-Video (Singer et al. 2022) extends a pre-trained diffusion image model with a temporal dimension, consisting of three key components:

1. A base text-to-image model trained on text-image pair data.
2. Spatiotemporal convolution and attention layers to extend the network to cover temporal dimension.
3. A frame interpolation network for high frame rate generation

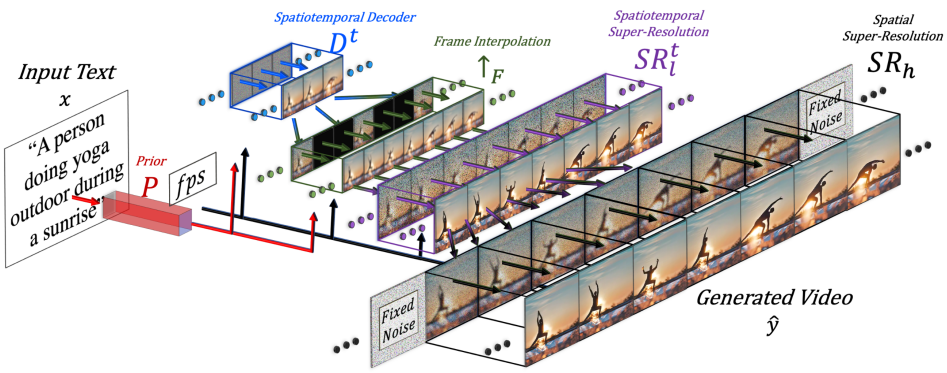

Figure 6: The illustration of Make-A-Video pipeline. (Image source: Singer et al. 2022)

The final video inference scheme can be formulated as:

$$\hat{\mathbf{y}}_t = \text{SR}_h \circ \text{SR}_l^t \circ \uparrow_F \circ D^t \circ P \circ (\hat{\mathbf{x}}, \text{CLIP}_{\text{text}}(\mathbf{x}))$$

where:

- $\mathbf{x}$ is the input text.
- $\hat{\mathbf{x}}$ is the BPE-encoded text.
- $\text{CLIP}_{\text{text}}(.)$ is the CLIP text encoder, $\mathbf{x}_e = \text{CLIP}_{\text{text}}(\mathbf{x})$.
- $P(.)$ is the prior, generating image embedding $\mathbf{y}_e$ given text embedding $\mathbf{x}_e$ and BPE encoded text $\hat{\mathbf{x}}$: $\mathbf{y}_e = P(\mathbf{x}_e, \hat{\mathbf{x}})$. This part is trained on text-image pair data and not fine-tuned on video data.
- $D^t(.)$ is the spatiotemporal decoder that generates a series of 16 frames, where each frame is a low-resolution 64x64 RGB image $\hat{\mathbf{y}}_l$.
- $\uparrow_F$ (.) is the frame interpolation network, increasing the effective frame rate by interpolating between generated frames. This is a fine-tuned model for the task of predicting masked frames for video upsampling.
- $\text{SR}_h(.), \text{SR}_l^t(.)$ are the spatial and spatiotemporal super-resolution models, increasing the image resolution to 256x256 and 768x768, respectively.
- $\hat{\mathbf{y}}_t$ is the final generated video.

Spatiotemporal SR layers contain pseudo-3D convo layers and pseudo-3D attention layers:

- Pseudo-3D convo layer: Each spatial 2D convo layer (initialized from the pre-training image model) is followed by a temporal 1D layer (initialized as the identity function). Conceptually, the convo 2D layer first generates multiple frames and then frames are reshaped to be a video clip.
- Pseudo-3D attention layer: Following each (pre-trained) spatial attention layer, a temporal attention layer is stacked and used to approximate a full spatiotemporal attention layer.

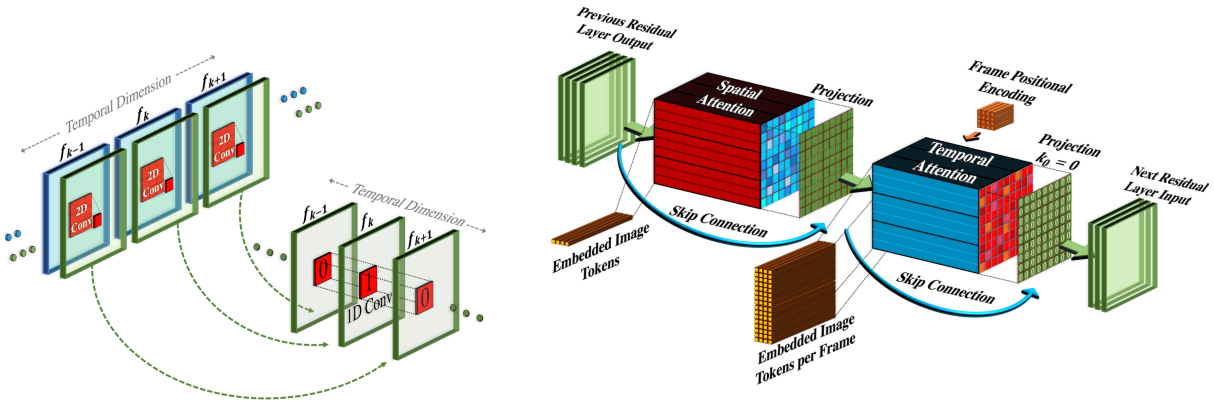

Figure 7: How pseudo-3D convolution (left) and attention (right) layers work. (Image source: Singer et al. 2022)

They can be represented as:

$$\text{Conv}_{\text{P3D}} = \text{Conv}_{\text{1D}}(\text{Conv}_{\text{2D}}(\mathbf{h}) \circ T) \circ T$$
$$\text{Attn}_{\text{P3D}} = \text{flatten}^{-1}(\text{Attn}_{\text{1D}}(\text{Attn}_{\text{2D}}(\text{flatten}(\mathbf{h})) \circ T) \circ T)$$

where an input tensor $\mathbf{h} \in \mathbb{R}^{B \times C \times F \times H \times W}$ (corresponding to batch size, channels, frames, height and weight); and $\circ T$ swaps between temporal and spatial dimensions; flatten(.) is a matrix operator to convert $\mathbf{h}$ to be $\mathbf{h}' \in \mathbb{R}^{B \times C \times F \times HW}$ and $\text{flatten}^{-1}(.)$ reverses that process.

During training, different components of Make-A-Video pipeline are trained independently.

1. Decoder D^t, prior P and two super-resolution components $\text{SR}_h, \text{SR}_l^t$ are first trained on images alone, without paired text.
2. Next the new temporal layers are added, initialized as identity function, and then fine-tuned on unlabeled video data.

Tune-A-Video (Wu et al. 2023) inflates a pre-trained image diffusion model to enable one-shot video tuning: Given a video containing m frames, $\mathcal{V} = \{v_i \mid i = 1, \dots, m\}$, paired with a descriptive prompt τ, the task is to generate a new video $\mathcal{V}^*$ based on a slightly edited & related text prompt τ^*. For example, τ = "A man is skiing" can be extended to τ^*="Spiderman is skiing on the beach". Tune-A-Video is meant to be used for object editing, background change, and style transfer.

Besides inflating the 2D convo layer, the U-Net architecture of Tune-A-Video incorporates the ST-Attention (spatiotemporal attention) block to capture temporal consistency by querying relevant positions in previous frames. Given latent features of frame v_i, previous frames v_{i-1} and the first frame v_1 are projected to query $\mathbf{Q}$, key $\mathbf{K}$ and value $\mathbf{V}$, the ST-attention is defined as:

$$\mathbf{Q} = \mathbf{W}^Q \mathbf{z}_{v_i}, \quad \mathbf{K} = \mathbf{W}^K [\mathbf{z}_{v_1}, \mathbf{z}_{v_{i-1}}], \quad \mathbf{V} = \mathbf{W}^V [\mathbf{z}_{v_1}, \mathbf{z}_{v_{i-1}}]$$
$$\mathbf{O} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}}\right) \cdot \mathbf{V}$$

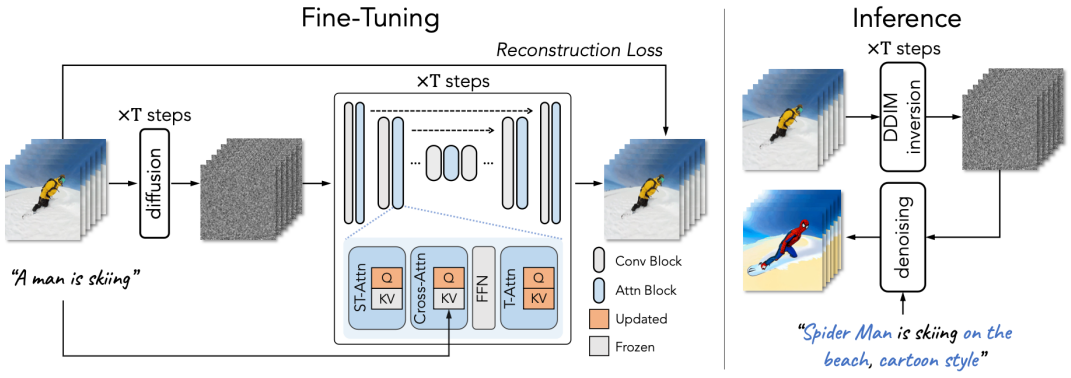

Figure 8: The Tune-A-Video architecture overview. It first runs a light-weighted fine-tuning stage on a single video before the sampling stage. Note that the entire temporal self-attention (T-Attn) layers get fine-tuned because they are newly added, but only query projections in ST-Attn and Cross-Attn are updated during fine-tuning to preserve prior text-to-image knowledge. ST-Attn improves spatial-temporal consistency, Cross-Attn refines text-video alignment. (Image source: Wu et al. 2023)

Gen-1 model (Esser et al. 2023) by Runway targets the task of editing a given video according to text inputs. It decomposes the consideration of *structure* and *content* of a video $p(\mathbf{x} \mid s, c)$ for generation conditioning. However, to do a clear decomposition of these two aspects is not easy.

- *Content* c refers to appearance and semantics of the video, that is sampled from the text for conditional editing. CLIP embedding of the frame is a good representation of content, and stays largely orthogonal to structure traits.
- *Structure* s depicts geometry and dynamics, including shapes, locations, temporal changes of objects, and s is sampled from the input video. Depth estimation or other task-specific side information (e.g. human body pose or face landmarks for human video synthesis) can be used.

The architecture changes in Gen-1 are quite standard, i.e. adding 1D temporal convo layer after each 2D spatial convo layer in its residual blocks and adding 1D temporal attention block after each 2D spatial attention block in its attention blocks. During training, the structure variable s is concatenated with the diffusion latent variable $\mathbf{z}$, where the content variable c is provided in the cross-attention layer. At inference time, the clip embedding is converted via a prior to convert CLIP text embedding to be CLIP image embedding.

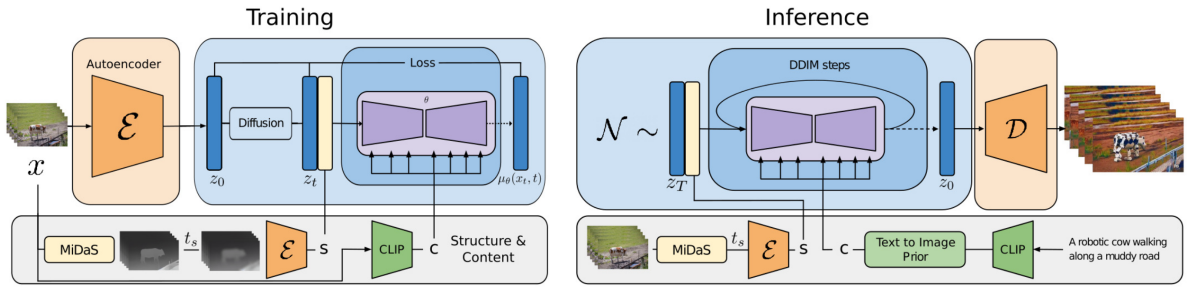

Figure 9: The overview of the Gen-1 model training pipeline. (Image source: Esser et al. 2023)

Video LDM (Blattmann et al. 2023) trains a LDM (Latent diffusion models) image generator first. Then the model is fine-tuned to produce videos with a temporal dimension added. The fine-tuning only applies to these newly added temporal layers on encoded image sequences. The temporal layers $\{l_\phi^i \mid i = 1, \dots, L\}$ in the Video LDM (See Fig. 10) are interleaved with existing spatial layers l_θ^i which stays *frozen* during fine-tuning. That's being said, we only fine-tune the new parameters ϕ but not the pre-trained image backbone model parameters θ. The pipeline of Video LDM first generates key frames at low fps and then processes through 2 steps of latent frame interpolations to increase fps.

The input sequence of length T is interpreted as a batch of images (i.e. $B \cdot T$) for the base image model θ and then gets reshaped into video format for l_ϕ^i temporal layers. There is a skip connection leads to a combination of temporal layer output $\mathbf{z}'$ and the spatial output $\mathbf{z}$ via a learned merging parameter α. There are two types of temporal mixing layers implemented in practice: (1) temporal attention and (2) residual blocks based on 3D convolutions.

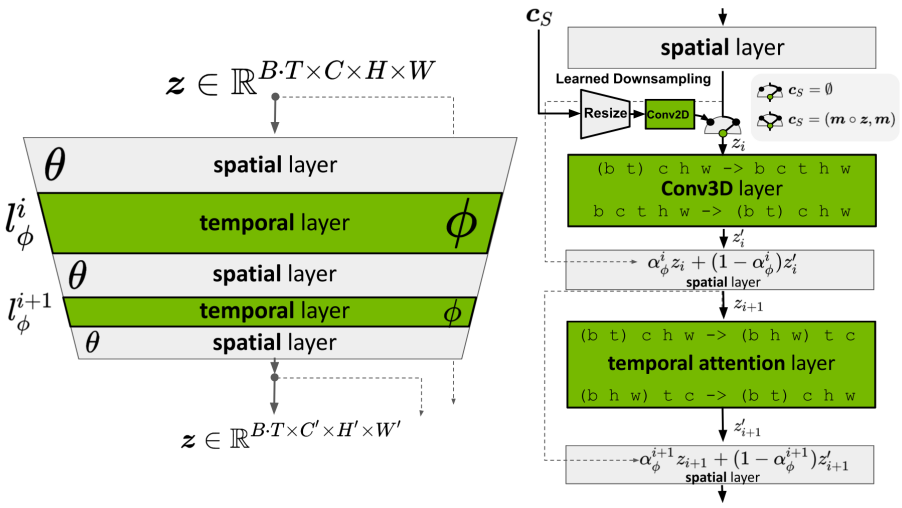

Figure 10: A pre-training LDM for image synthesis is extended to be a video generator. B, T, C, H, W are batch size, sequence length, channels, height and width, respectively. $\mathbf{c}_S$ is an optional conditioning/context frame. (Image source: Blattmann et al. 2023)

However, there is a remaining issue with LDM's pretrainined autoencoder which only sees images never videos. Naively using that for video generation can cause flickering artifacts without good temporal coherence. So Video LDM adds additional temporal layers into the decoder and fine-tuned on video data with a patch-wise temporal discriminator built from 3D convolutions, while the encoder remains unchanged so that we still can reuse the pretrained LDM. During temporal decoder fine-tuning, the frozen encoder processes each frame in the video independently, and enforce temporally coherent reconstructions across frames with a video-aware discriminator.

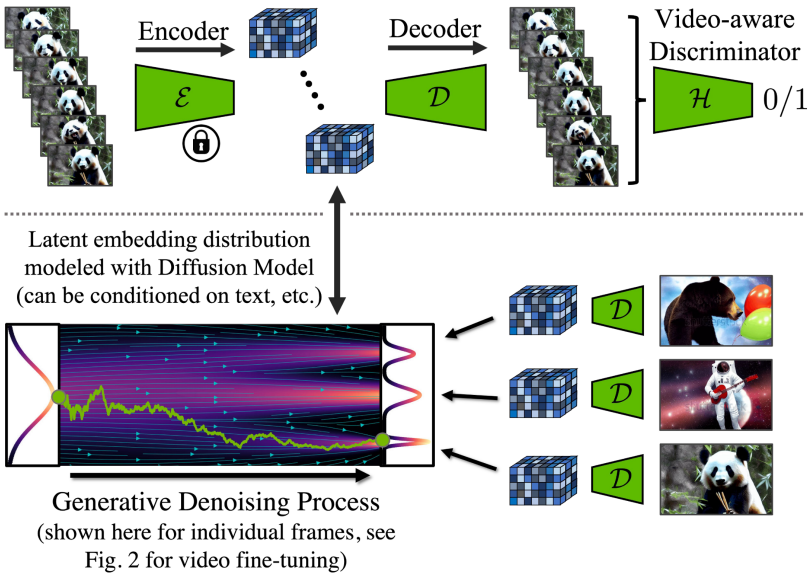

Figure 11: The training pipeline of autoencoder in video latent diffusion models. The decoder is fine-tuned to have temporal coherency with a new across-frame discriminator while the encoder stays frozen. (Image source: Blattmann et al. 2023)

Similar to Video LDM, the architecture design of **Stable Video Diffusion** (**SVD**; Blattmann et al. 2023) is also based on LDM with temporal layers inserted after every spatial convolution and attention layer, but SVD fine-tunes the entire model. There are three stages for training video LDMs:

1. *Text-to-image pretraining* is important and helps improve both quality and prompt following.
2. *Video pretraining* is beneficial to be separated and should ideally occur on a larger scale, curated dataset.
3. *High-quality video finetuning* works with a smaller, pre-captioned video of high visual fidelity.

SVD specially emphasizes the critical role of *dataset curation* in model performance. They applied a cut detection pipeline to get more cuts per video and then applied three different captioner models: (1) CoCa for mid-frame, (2) V-BLIP for a video caption, and (3) LLM based captioning based on previous two captions. Then they were able to continue to improve video datasets, by removing clips with less motion (filtered by low optical flow scores calculated at 2 fps), excessive text presence (apply optical character recognition to identify videos with lots of text), or generally low aesthetic value (annotate the first, middle, and last frames of each clip with CLIP embeddings and calculate aesthetics scores & text-image similarities). The experiments showed that a filtered, higher quality dataset leads to better model quality, even when this dataset is much smaller.

The key challenge of generating distant key frames first and then adding interpolation with temporal super-resolution is how to maintain high-quality temporal consistency. **Lumiere** (Bar-Tal et al. 2024) instead adopts a **space-time U-Net (STUNet)** architecture that generates the entire temporal duration of the video *at once* through a single pass, removing the dependency on TSR (temporal super-resolution) components. STUNet downsamples the video in both time and space dimensions and thus expensive computation happens in a compact time-space latent space.

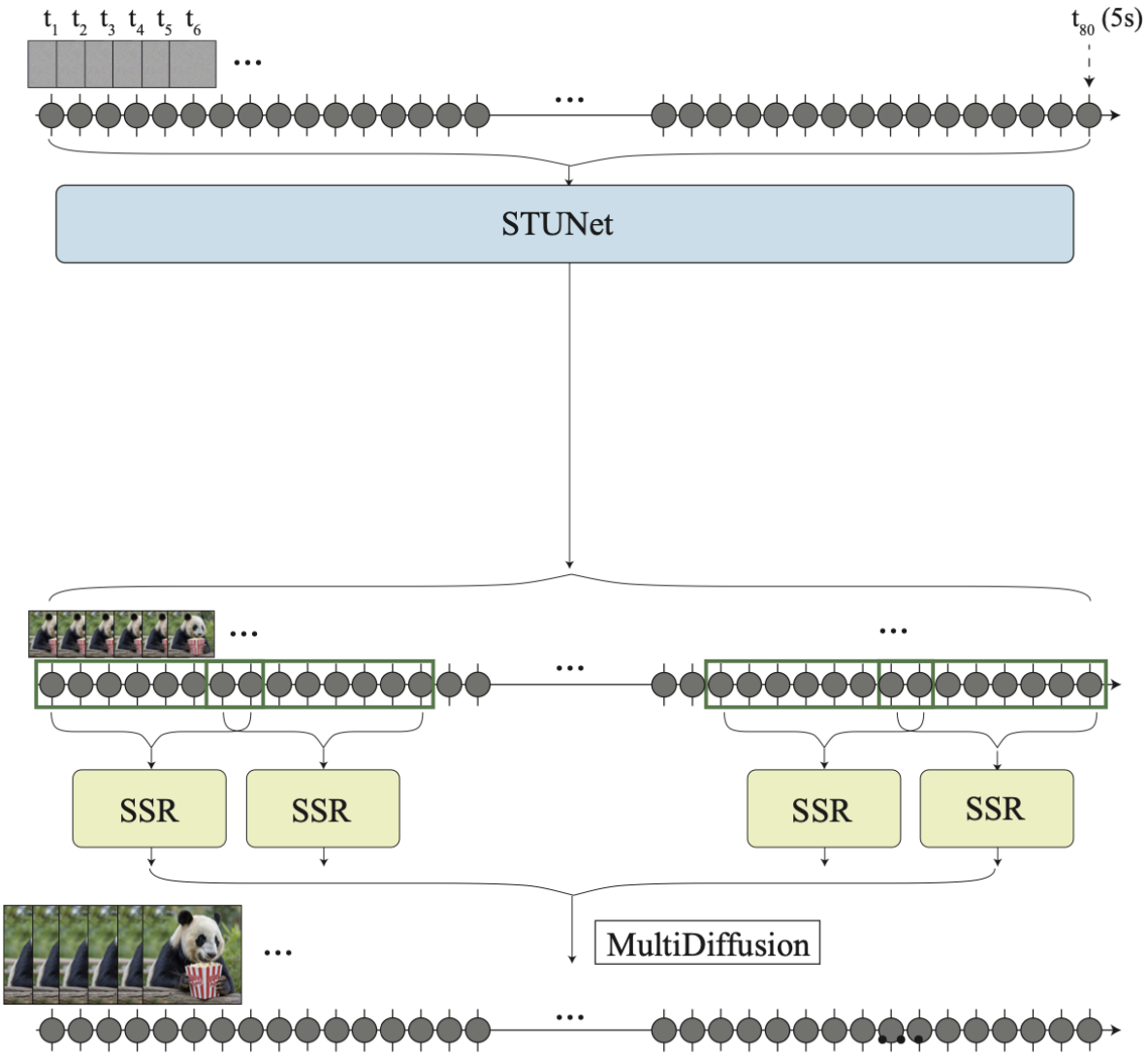

Figure 12: Lumiere removes TSR (temporal super-resolution) models. The inflated SSR network can operate only on short segments of the video due to memory constraints and thus SSR models operate on a set of shorter but overlapped video snippets. (Image source: Bar-Tal et al. 2024)

STUNet inflates a *pretrained* text-to-image U-net to be able to downsample and upsample videos at both time and space dimensions. Convo-based blocks consist of pre-trained text-to-image layers, followed by a factorized space-time convolution. And attention-based blocks at the coarsest U-Net level contains the pre-trained text-to-image, followed by temporal attention. Further training *only* happens with the newly added layers.

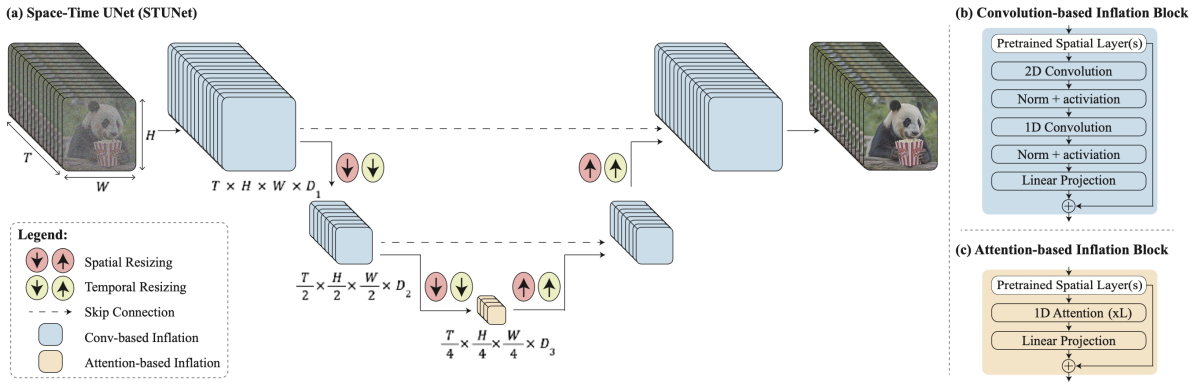

Figure 13: The architecture of (a) Space-Time U-Net (STUNet), (b) the convolution-based block, and (c) the attention-based block. (Image source: Bar-Tal et al. 2024)

2.2 Training-Free Adaptation

Somehow surprisingly, it is possible to adapt a pre-trained text-to-image model to output videos without any training 😳.

If we naively sample a sequence of latent codes at random and then construct a video of decoded corresponding images, there is no guarantee in the consistency in objects and semantics in time. **Text2Video-Zero** (Khachatryan et al. 2023) enables zero-shot, training-free video generation by enhancing a pre-trained image diffusion model with two key mechanisms for temporal consistency:

1. Sampling the sequence of latent codes with *motion dynamics* to keep the global scene and the background time consistent;
2. Reprogramming frame-level self-attention using a *new cross-frame attention* of each frame on the first frame, to preserve the context, appearance, and identity of the foreground object.

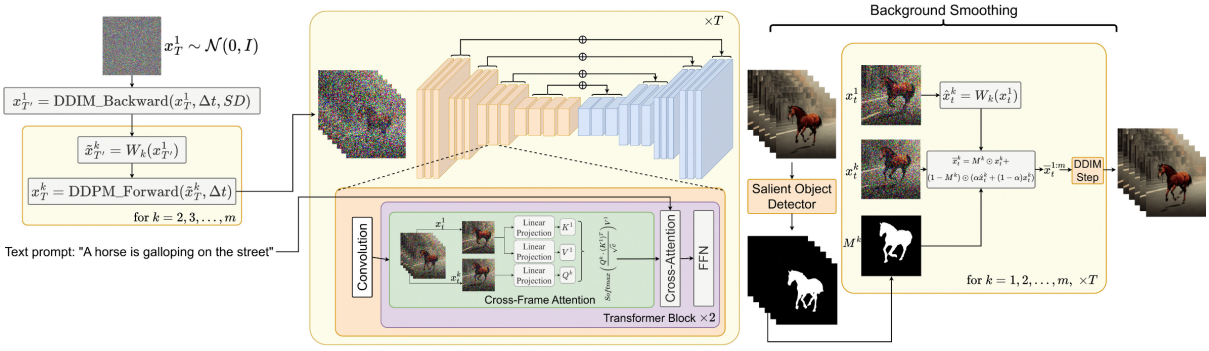

Figure 14: An overview of the Text2Video-Zero pipeline. (Image source: Khachatryan et al. 2023)

The process of sampling a sequence of latent variables, $\mathbf{x}_T^1, \ldots, \mathbf{x}_T^m$, with motion information is described as follows:

1. Define a direction $\boldsymbol{\delta} = (\delta_x, \delta_y) \in \mathbb{R}^2$ for controlling the global scene and camera motion; by default, we set $\boldsymbol{\delta} = (1, 1)$. Also define a hyperparameter $\lambda > 0$ controlling the amount of global motion.

2. First sample the latent code of the first frame at random, $\mathbf{x}_T^1 \sim \mathcal{N}(0, I)$;

3. Perform $\Delta t \geq 0$ DDIM backward update steps using the pre-trained image diffusion model, e.g. Stable Diffusion (SD) model in the paper, and obtain the corresponding latent code $\mathbf{x}_{T'}^1$ where $T' = T - \Delta t$.

4. For each frame in the latent code sequence, we apply corresponding motion translation with a warping operation defined by $\boldsymbol{\delta}^k = \lambda(k-1)\boldsymbol{\delta}$ to obtain $\tilde{\mathbf{x}}_{T'}^k$.

5. Finally apply DDIM forward steps to all $\tilde{\mathbf{x}}_{T'}^{2:m}$ to obtain $\mathbf{x}_T^{2:m}$.

$$
\begin{aligned}
\mathbf{x}_{T'}^1 &= \text{DDIM-backward}(\mathbf{x}_T^1, \Delta t) \text{ where } T' = T - \Delta t \\
W_k &\leftarrow \text{a warping operation of } \boldsymbol{\delta}^k = \lambda(k-1)\boldsymbol{\delta} \\
\tilde{\mathbf{x}}_{T'}^k &= W_k(\mathbf{x}_{T'}^1) \\
\mathbf{x}_T^k &= \text{DDIM-forward}(\tilde{\mathbf{x}}_{T'}^k, \Delta t) \text{ for } k = 2, \dots, m
\end{aligned}
$$

Besides, Text2Video-Zero replaces the self-attention layer in a pre-trained SD model with a new cross-frame attention mechanism with reference to the *first* frame. The motivation is to preserve the information about the foreground object's appearance, shape, and identity throughout the generated video.

$$
\text{Cross-Frame-Attn}(\mathbf{Q}^k, \mathbf{K}^{1:m}, \mathbf{V}^{1:m}) = \text{Softmax}\Big(\frac{\mathbf{Q}^k (\mathbf{K}^1)^\top}{\sqrt{c}}\Big)\mathbf{V}^1
$$

Optionally, the background mask can be used to further smoothen and improve background consistency. Let's say, we obtain a corresponding foreground mask $\mathbf{M}_k$ for the k-th frame using some existing method, and background smoothing merges the actual and the warped latent code at the diffusion step t, w.r.t. the background matrix:

$$
\bar{\mathbf{x}}_t^k = \mathbf{M}^k \odot \mathbf{x}_t^k + (1 - \mathbf{M}^k) \odot (\alpha \tilde{\mathbf{x}}_t^k + (1-\alpha)\mathbf{x}_t^k) \quad \text{for } k = 1, \dots, m
$$

where $\mathbf{x}_t^k$ is the actual latent code and $\tilde{\mathbf{x}}_t^k$ is the warped latent code on the background; α is a hyperparameter and the papers set $\alpha = 0.6$ in the experiments.

Text2video-zero can be combined with ControlNet where the ControlNet pretrained copy branch is applied per frame on each $\mathbf{x}_t^k$ for $k = 1, \dots, m$ in each diffusion time-step $t = T, \dots, 1$ and add the ControlNet branch outputs to the skip-connections of the main U-net.

ControlVideo (Zhang et al. 2023) aims to generate videos conditioned on text prompt τ and a motion sequence (e.g., depth or edge maps), $\mathbf{c} = \{c^i\}_{i=0}^{N-1}$. It is adapted from ControlNet with three new mechanisms added:

1. *Cross-frame attention*: Adds fully cross-frame interaction in self-attention modules. It introduces interactions between all the frames, by mapping the latent frames at *all the time steps* into $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ matrices, different from Text2Video-zero which only configures all the frames to attend to the *first* frame.

2. *Interleaved-frame smoother* is a mechanism to employ frame interpolation on alternated frames to reduce the flickering effect. At each time step t, the smoother interpolates the even or odd frames to smooth their corresponding three-frame clips. Note that the number of frames decreases in time after smoothing steps.

3. *Hierarchical sampler* utilizes a hierarchical sampler to enable long videos with time consistency under memory constraints. A long video is split into multiple short clips and each has a key frame selected. The model pre-generates these keyframes with full cross-frame attention for long-term coherency and each corresponding short clip is synthesized sequentially conditioned on the keyframes.

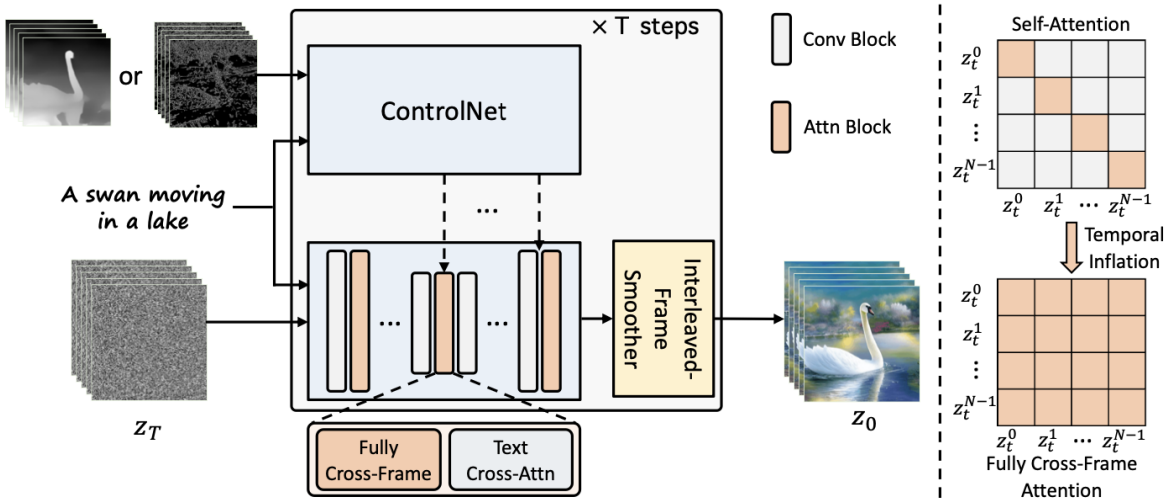

Figure 15: The overview of ControlVideo. (Image source: Zhang et al. 2023)

3 Citation

Cited as:

Weng, Lilian. (Apr 2024). Diffusion Models Video Generation. Lil'Log. https://lilianweng.github.io/posts/2024-04-12-diffusion-video/.

Or

```
@article{weng2024video,
title = "Diffusion Models Video Generation.",
author = "Weng, Lilian",
journal = "lilianweng.github.io",
year = "2024",
month = "Apr",
url = "https://lilianweng.github.io/posts/2024-04-12-diffusion-video/"
};
```

4 References

1 Cicek et al. 2016. "3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation."

2 Ho & Salimans, et al. "Video Diffusion Models." 2022 webpage

3 Bar-Tal et al. 2024 "Lumiere: A Space-Time Diffusion Model for Video Generation."

4 Brooks et al. "Video generation models as world simulators." OpenAI Blog, 2024.

5 Zhang et al. 2023 "ControlVideo: Training-free Controllable Text-to-Video Generation."

6 Khachatryan et al. 2023 "Text2Video-Zero: Text-to-image diffusion models are zero-shot video generators."

7 Ho, et al. 2022 "Imagen Video: High Definition Video Generation with Diffusion Models."

8 Singer et al. "Make-A-Video: Text-to-Video Generation without Text-Video Data." 2022.

9 Wu et al. "Tune-A-Video: One-Shot Tuning of Image Diffusion Models for Text-to-Video Generation." ICCV 2023.

10 Blattmann et al. 2023 "Align your Latents: High-Resolution Video Synthesis with Latent Diffusion Models."

11 Blattmann et al. 2023 "Stable Video Diffusion: Scaling Latent Video Diffusion Models to Large Datasets."

12 Esser et al. 2023 "Structure and Content-Guided Video Synthesis with Diffusion Models."

13 Bar-Tal et al. 2024 "Lumiere: A Space-Time Diffusion Model for Video Generation."

Tags: Generative-Model Video-Generation

(Extrinsic Hallucinations in LLMs

Thinking about High-Quality Human Data)