Lilian Weng｜视频生成的扩散模型

> 原文地址：https://lilianweng.github.io/posts/2024-04-12-diffusion-video/
>
> 编译：机器之心

视频生成任务本身是图像合成的超集，因为图像就是单帧视频。视频合成的难度要大得多，原因是：

1. 视频合成还需要不同帧之间保持时间一致性，很自然，这需要将更多世界知识编码到模型之中。
2. 相比于文本或图像，收集大量高质量、高维度的视频数据要更为困难，更遑论配对的文本 - 视频数据了。

> 如果你想了解扩散模型在图像生成方面的应用，可参读 Lilian Weng 之前发布的博文《What are Diffusion Models?》

从头建模视频生成

首先，我们先来看看如何从头设计和训练扩散视频模型，也就是说不使用已经预训练好的图像生成器。

参数化和采样

这里使用的变量定义与之前那篇文章稍有不同，但数学形式是一致的。令 ☒~☒_real 是从该真实数据分布采样的一个数据点。现在，在时间中添加少量高斯噪声，创建出 ☒ 的一个有噪声变体序列，记为：{☒_☒ | ☒ = 1..., ☒}，其中噪声随 ☒ 的增加而增长，而最后的 ☒(☒_☒)~☒(☒,☒)。这个添加噪声的前向过程是一个高斯过程。令 ☒_☒ 和 ☒_☒ 为这个高斯过程的可微噪声调度（differentiable noise schedule）：

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

为了表示 ☒(☒_☒|☒_☒)，其中 0⩽☒<☒⩽☒，有：

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

令对数信噪比为 ⊠⊠=log[⊠⊠2/⊠⊠2], ，则可将 DDIM 更新表示为：

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

Salimans & Ho (2022) 的论文《Progressive Distillation for Fast Sampling of Diffusion Models》为这里提出了一个特殊的 ⊠ 预测参数： ⊠-prediction (⊠=⊠⊠⊠−⊠⊠⊠) p。研究表明，相比于 ⊠ 参数，⊠ 参数有助于避免视频生成中出现颜色变化问题。

⊠ 的参数化是通过角坐标中的技巧导出的。首先，定义 ⊠_⊠=arctan (⊠_⊠/⊠_⊠)，由此可得到 ⊠_⊠=cos ⊠, ⊠_⊠=sin ⊠, ⊠_⊠=cos ⊠⊠+sin ⊠⊠。⊠_⊠ 的速度可以写成：

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

然后可以推导出：

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

DDIM 更新规则就可以据此更新：

$$
\begin{aligned}
\mathbf{z}_{\phi_s} &= \cos\phi_s \hat{\mathbf{x}}_\theta(\mathbf{z}_{\phi_t}) + \sin\phi_s \hat{\epsilon}_\theta(\mathbf{z}_{\phi_t}) && ; \hat{\mathbf{x}}_\theta(.), \hat{\epsilon}_\theta(.) \text{ are two models to predict } \mathbf{x}, \boldsymbol{\epsilon} \text{ based on } \mathbf{z}_{\phi_t} \\
&= \cos\phi_s \big(\cos\phi_t \mathbf{z}_{\phi_t} - \sin\phi_t \hat{\mathbf{v}}_\theta(\mathbf{z}_{\phi_t}) \big) + \sin\phi_s \big(\sin\phi_t \mathbf{z}_{\phi_t} + \cos\phi_t \hat{\mathbf{v}}_\theta(\mathbf{z}_{\phi_t}) \big) \\
&= \big(\cos\phi_s \cos\phi_t + \sin\phi_s \sin\phi_t \big) \mathbf{z}_{\phi_t} + \big(\sin\phi_s \cos\phi_t - \cos\phi_s \sin\phi_t \big) \hat{\mathbf{v}}_\theta(\mathbf{z}_{\phi_t}) \\
&= cos(\phi_s - \phi_t) \mathbf{z}_{\phi_t} + \sin(\phi_s - \phi_t) \hat{\mathbf{v}}_\theta(\mathbf{z}_{\phi_t}) && ; \text{trigonometric identity functions.}
\end{aligned}
$$

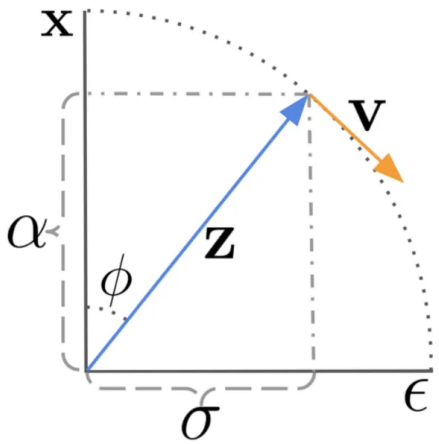

图 1：以角坐标形式展示扩散更新步骤的工作方式，其中 DDIM 通过让 z_{⊠_⊠} 沿 −⊠^⊠⊠ 的方向移动而使其不断演进。

对模型来说，⊠ 的参数化就是预测

$$
\mathbf{v}_\phi = \cos\phi\boldsymbol{\epsilon} - \sin\phi\mathbf{x} = \alpha_t\boldsymbol{\epsilon} - \sigma_t\mathbf{x}.
$$

对于视频生成任务，为了延长视频长度或提升帧率，需要扩散模型运行多个上采样步骤。这就需要基于第一个视频 ⊠^⊠ 采样第二个视频 ⊠^⊠ 的能力， ⊠⊠⊠⊠⊠(⊠⊠|⊠⊠)， 其中 ⊠^⊠ 可能是 ⊠^⊠ 的自回归扩展或是一个低帧率的视频之中缺失的帧。

除了其自身对应的有噪声变量之外，⊠^⊠ 的采样还需要基于 ⊠^⊠。Ho & Salimans 等人 2022 年的视频扩散模型（VDM）提出使用一种经过调整的去噪模型来实现重构引导方法，这样就可以很好地以 ⊠^⊠ 为基础实现 ⊠^⊠ 的采样：

$$
\begin{aligned}
\mathbb{E}_q[\mathbf{x}_b|\mathbf{z}_t, \mathbf{x}^a] &= \mathbb{E}_q[\mathbf{x}^b|\mathbf{z}_t] + \frac{\sigma_t^2}{\alpha_t} \nabla_{\mathbf{z}_t^b} \log q(\mathbf{x}^a|\mathbf{z}_t) \\
q(\mathbf{x}^a|\mathbf{z}_t) &\approx \mathcal{N}\big[\hat{\mathbf{x}}_\theta^a(\mathbf{z}_t), \frac{\sigma_t^2}{\alpha_t^2}\mathbf{I}\big] && ; \text{the closed form is unknown.} \\
\tilde{\mathbf{x}}_\theta^b(\mathbf{z}_t) &= \hat{\mathbf{x}}_\theta^b(\mathbf{z}_t) - \frac{w_r \alpha_t}{2} \nabla_{\mathbf{z}_t^b} \|\mathbf{x}^a - \hat{\mathbf{x}}_\theta^a(\mathbf{z}_t)\|_2^2 && ; \text{an adjusted denoising model for } \mathbf{x}^b
\end{aligned}
$$

其中 是 ⊠^⊠ 和 ⊠^⊠ 根据所提供的去噪模型的重构。而 ⊠_⊠ 是一个加权因子，可以寻找一个较大的 ⊠_⊠ > 1 来提升采样质量。请注意，使用同样的重建引导方法，也有可能基于低分辨率视频来扩展样本，使之变成高分辨率样本。

模型架构：3D U-Net 和 DiT

类似于文生图扩散模型，U-Net 和 Transformer 依然是常用的架构选择。谷歌基于 U-net 架构开发了一系列扩散视频建模论文，OpenAI 近期的 Sora 模型则是利用了 Transformer 架构。

VDM 采用了标准的扩散模型设置，但对架构进行了一些修改，使之更适合视频建模任务。它对 2D U-net 进行了扩展，使其能处理 3D 数据，其中每个特征图（feature map）都表示一个 4D 张量：帧数 x 高度 x 宽度 x 通道数。这个 3D U-net 是在空间和时间上进行分解，也就是说每一层都仅会操作空间或时间一个维度，而不会同时操作两者。

- 处理空间：原本和 2D U-net 中一样的 2D 卷积层会被扩展成仅针对空间的 3D 卷积，具体来说，就是 3x3 卷积变成 1x3x3 卷积。每一个空间注意力模块依然是关注空间的注意力，其中第一个轴（frames）则被当成批（batch）维度处理。
- 处理时间：每个空间注意力模块之后会添加一个时间注意力模块。其关注的是第一个轴（frames）并将空间轴视为批维度。使用这种相对位置嵌入可以跟踪帧的顺序。这个时间注意力模块可让模型获得很好的时间一致性。

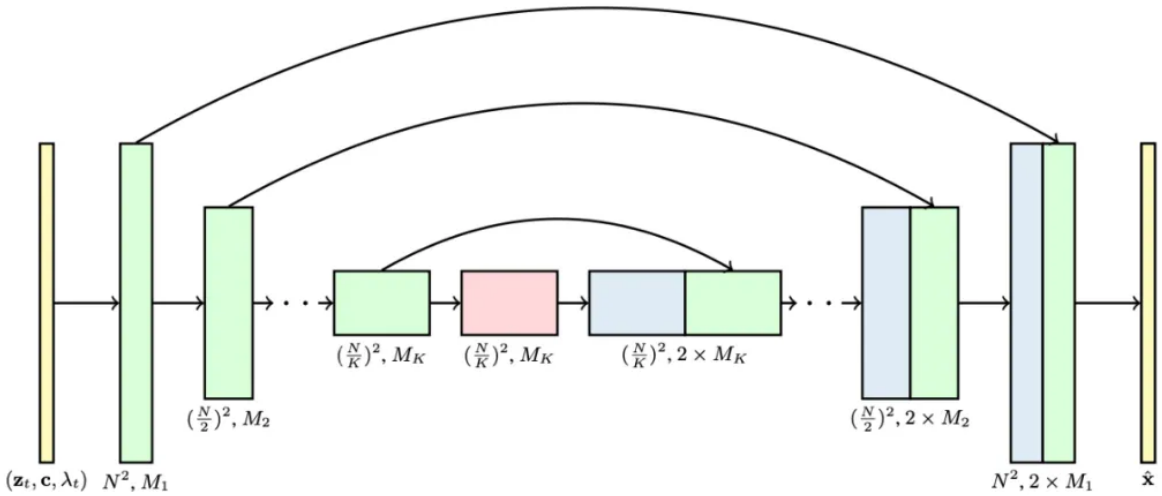

图 2：3D U-net 架构。该网络的输入是有噪声视频 z_☒、条件信息 c 和对数信噪比（log-SNR）λ_☒。通道乘数 M_1,...,M_☒ 表示各层的通道数量。

Ho, et al. 在 2022 年提出的 Imagen Video 基于一组级联的扩散模型，其能提升视频生成的质量，并将输出升级成 24 帧率的 1280x768 分辨率视频。Imagen Video 架构由以下组件构成，总计 7 个扩散模型。

- 一个冻结的 T5 文本编码器，用以提供文本嵌入作为条件输入。

- 一个基础视频扩散模型。
- 一组级联的交织放置的空间和时间超分辨率扩散模型，包含 3 个 TSR（时间超分辨率）和 3 个 SSR（空间超分辨率）组件。

图 3：Imagen Video 的级联式采样流程。在实践中，文本嵌入会被注入到所有组件中，而不只是基础模型中。

基础去噪模型使用共享的参数同时在所有帧上执行空间操作，然后时间层将各帧的激活混合起来，以更好地实现时间一致性；事实证明这种方法的效果优于帧自回归方法。

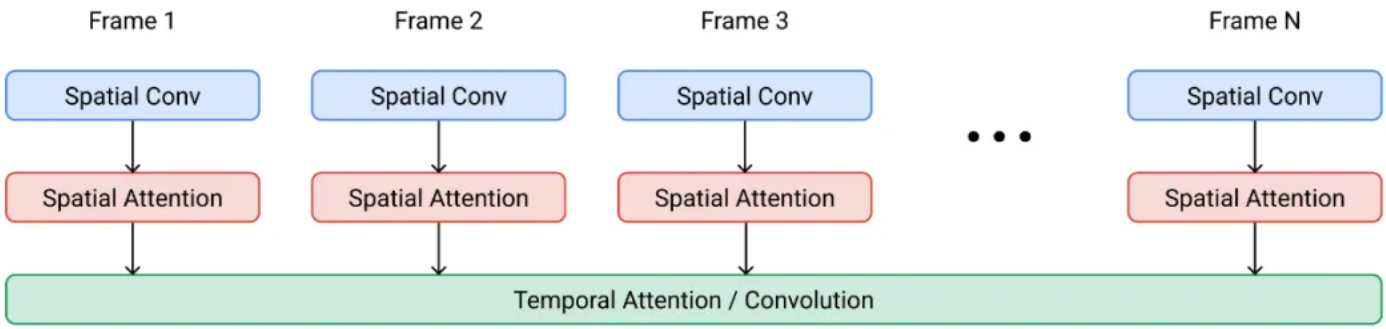

图 4：Imagen Video 扩散模型中一个空间 - 时间可分离模块的架构。

SSR 和 TSR 模型都基于在通道方面连接了有噪声数据 ☒_☒ 的上采样的输入。SSR 是通过双线性大小调整来上采样，而 TSR 则是通过重复帧或填充空白帧来上采样。

Imagen Video 还应用了渐进式蒸馏来加速采样，每次蒸馏迭代都可以将所需的采样步骤减少一半。在实验中，他们能够将所有 7 个视频扩散模型蒸馏为每个模型仅 8 个采样步骤，同时不会对感知质量造成任何明显损失。

为了更好地扩大模型规模，Sora 采用了 DiT（扩散 Transformer）架构，其操作的是视频和图像隐代码的时空块（spacetime patch）。其会将视觉输入表示成一个时空块序列，并将这些时空块用作 Transformer 输入 token。

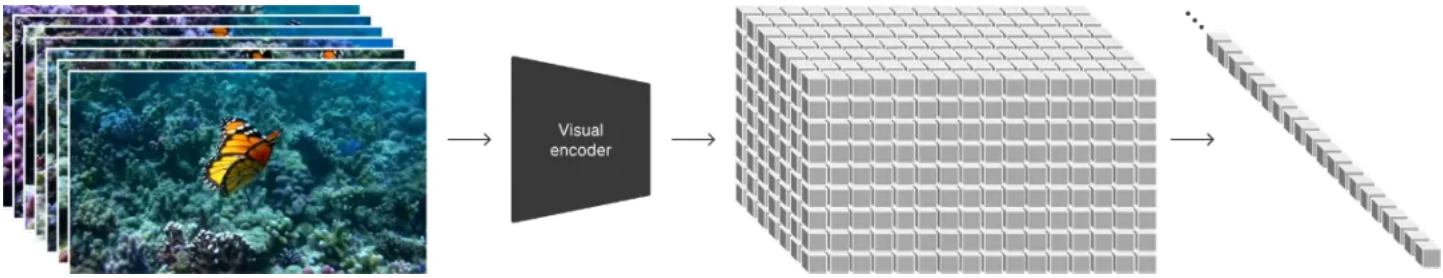

图 5：Sora 是一个扩散 Transformer 模型。

调整图像模型来生成视频

在扩散视频建模方面，另一种重要方法是通过插入时间层来「扩增」预训练的文生图扩散模型，然后就可以选择仅在视频上对新的层进行微调或完全避免进行额外的训练。这个新模型会继承文本 - 图像对的先验知识，由此可以帮助缓解对文本 - 视频对数据的需求。

在视频数据上进行微调

Singer et al. 在 2022 年提出的 Make-A-Video 是在一个预训练扩散图像模型的基础上扩展一个时间维度，其包含三个关键组件：

1. 一个在文本 - 图像对数据上训练的基础文生图模型。
2. 时空卷积和注意力层，使网络覆盖时间维度。
3. 一个帧插值网络，用于高帧率生成。

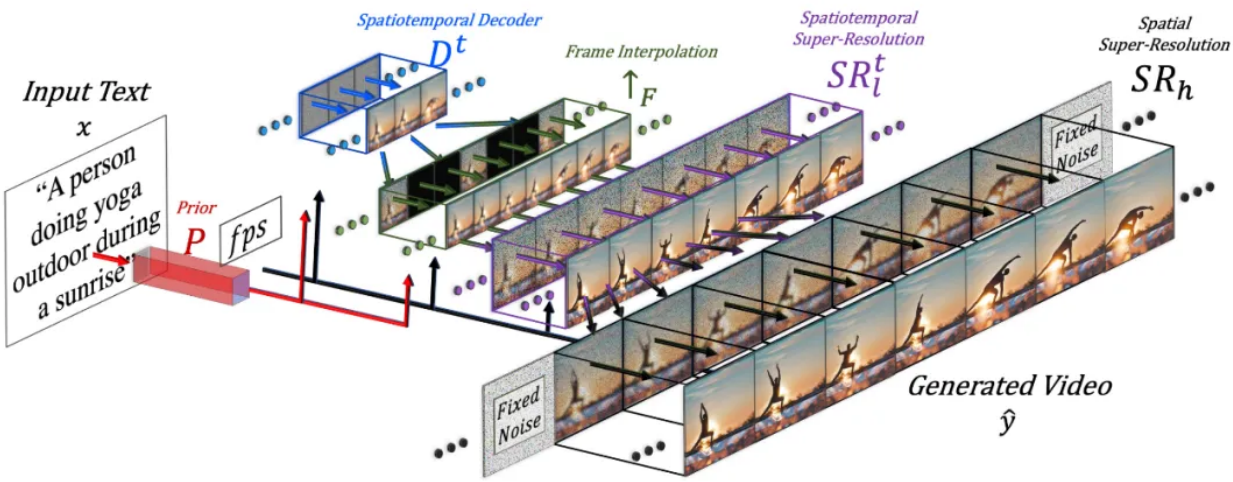

图 6：Make-A-Video 工作流程示意图。

最终的视频推理方案的数学形式可以写成这样：

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

其中：

- ☒ 是输入文本
- ☒^ 是 BPE 编码的文本
- CLIPtext(.) 是 CLIP 文本编码器，☒☒=CLIPtext(☒).
- ☒(.) 是先验，给定文本嵌入 ☒_☒ 和 BPE 编码的文本☒^ ，生成图像嵌入 ☒_☒: ☒☒=☒(☒☒,☒^) 。这部分是在文本 - 图像对数据上训练的，不会在视频数据上进行微调。
- ☒☒(.) 是时空解码器，其能生成一系列的 16 帧视频，其中每一帧都是低分辨率的 64x64 RGB 图像☒^☒.
- ↑☒(.) 是帧插值网络，可通过在生成的帧之间插值而有效提升帧率。这是一个经过微调的模型，可用于为视频上采样任务预测被掩蔽的帧。
- SR☒(.),SR☒☒(.) 是空间和时空超分辨率模型，可分别将图像分辨率提升到 256x256 和 768x768。
- ☒^☒ 是最终生成的视频。

时空超分辨率层包含伪 3D 卷积层和伪 3D 注意力层：

- 伪 3D 卷积层：每个空间 2D 卷积层（由预训练图像模型初始化）后面跟着一个时间 1D 层（由恒等函数初始化）。从概念上讲，其中的 2D 卷积层首先生成多帧，然后这些帧会被调整为一段视频。
- 伪 3D 注意力层：在每个（预训练）空间注意力层之后堆叠一个时间注意力层，从而近似得到一个完整的时空注意力层。

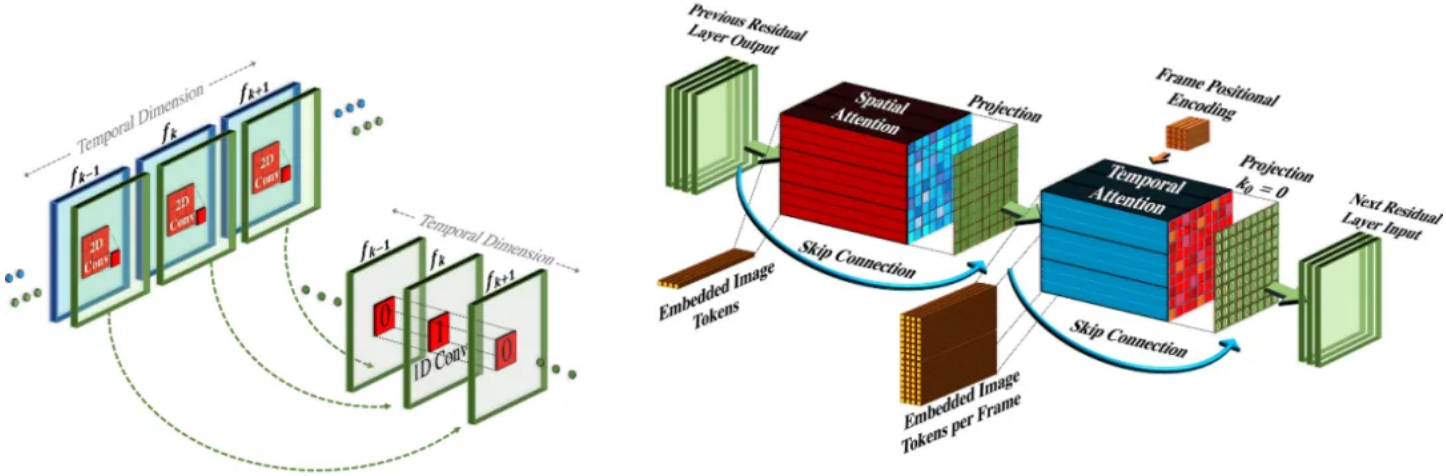

图 7：伪 3D 卷积（左）和注意力（右）层的工作方式。

它们可以表示成：

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

其中有输入张量 ☒∈☒^{☒×☒×☒×☒×☒}（对应于批量大小、通道数、帧数、高度和宽度）；

的作用是交换时间和空间维度；flatten (.) 是一个矩阵算子，可将 ☒ 转换成 ☒'∈☒^{☒×☒×☒×☒☒}，而 flatten☒¹(.) 的作用则相反。

训练时，Make-A-Video 工作流程中的不同组件是分开训练的。

1. 解码器 D^☒、先验 P 和两个超分辨率组件 SR☒,SR☒☒ 首先单独在图像上训练，而不使用配对的文本。
2. 接下来添加新的时间层，其初始化为恒等函数，然后在未标注的视频数据上进行微调。

Wu et al. 在 2023 年提出的 Tune-A-Video 是对一个预训练图像扩散模型进行扩展，使之可以实现单样本视频微调：给定一段包含 ☒ 帧的视频 ☒={ ☒_☒ | ☒=1,...,☒}，搭配上描述性的 prompt ☒，任务目标是基于经过稍有编辑且相关的文本 prompt ☒* 生成一段新视频 ☒*。举个例子，☒ = "A man is skiing" 可以扩展成 ☒* ="Spiderman is skiing on the beach"。Tune-A-Video 的设计目的是用于对象编辑、背景修改和风格迁移。

除了扩展 2D 卷积层之外，Tune-A-Video 的 U-Net 架构还整合了 ST-Attention（时空注意力）模块，可通过查询在之前帧中的相关位置来实现时间一致性。给定帧 ☒_☒、前一帧 ☒_{☒-1} 和第一帧 ☒_1 的隐含特征（它们被投射成查询 ☒、键 ☒ 和值 ☒），ST-Attention 的定义是：

$$\mathbf{Q} = \mathbf{W}^{Q}\mathbf{z}_{v_i}, \quad \mathbf{K} = \mathbf{W}^{K}[\mathbf{z}_{v_1}, \mathbf{z}_{v_{i-1}}], \quad \mathbf{V} = \mathbf{W}^{V}[\mathbf{z}_{v_1}, \mathbf{z}_{v_{i-1}}]$$

$$\mathbf{O} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^{\top}}{\sqrt{d}}\right) \cdot \mathbf{V}$$

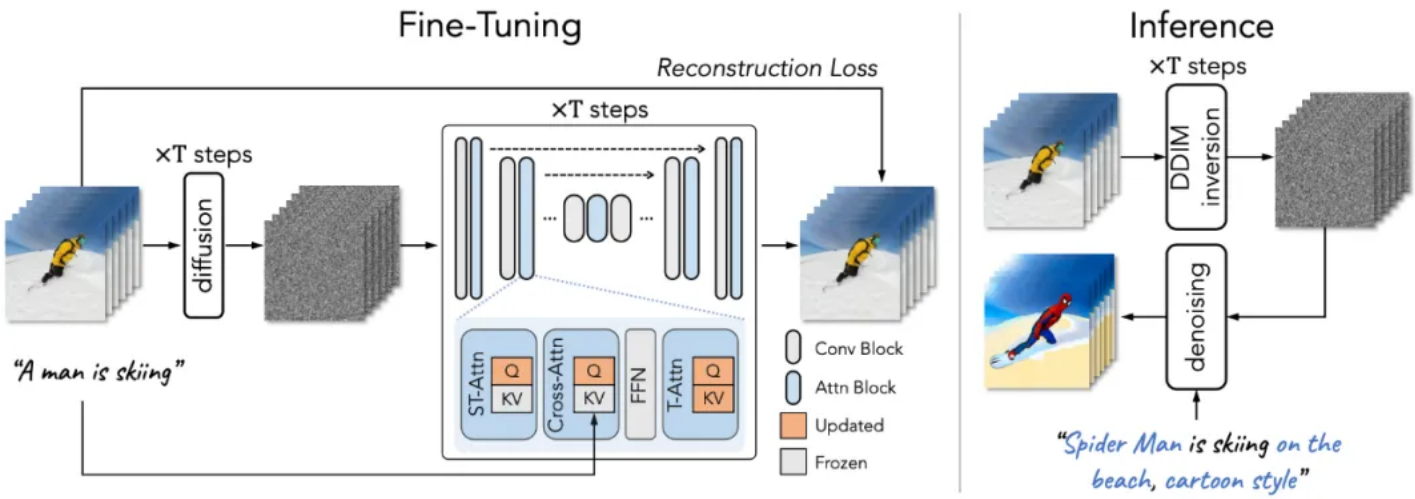

图 8：Tune-A-Video 架构概况。在采样阶段之前，它首先在单个视频上运行一个轻量加权的微调阶段。请注意整个时间自注意力（T-Attn）层都会得到微调，因为它们是新加入的，但在微调阶段，ST-Attn 和 Cross-Attn 中只有查询投射会被更新，以保留先验的文生图知识。ST-Attn 能提升时空一致性，Cross-Attn 能优化文本 - 视频的对齐。

Esser et al. 在 2023 年提出的 Gen-1 模型（Runway）针对的任务是根据文本输入编辑给定视频。它在考虑生成条件时会将视频的结构和内容分开处理：p (☒ | ☒,c)。但是，要将这两方面清楚地分开实非易事。

- 内容 ☒ 是指视频的外观和语义，其可从文本采样来进行条件化编辑。视频帧的 CLIP 嵌入能很好地表示内容，并且能在很大程度上与结构特征保持正交。
- 结构 ☒ 描述了几何性质和动态情况，包括形状、位置、物体的时间变化情况，☒ 是从输入视频采样的。可以使用深度估计或其它针对特定任务的辅助信息（比如用于人类视频合成的人体姿势或人脸标识信息）。

Gen-1 中的架构变化相当标准，即在其残差模块中的每个 2D 空间卷积层之后添加 1D 时间卷积层，在其注意力模块中的每个 2D 空间注意力模块之后添加 1D 时间注意力模块。训练期间，结构变量 ☒ 会与扩散隐变量 ☒ 连接起来，其中内容变量 ☒ 会在交叉注意力层中提供。在推理时间，会通过一个先验来转换 CLIP 嵌入 —— 将其从 CLIP 文本嵌入转换成 CLIP 图像嵌入。

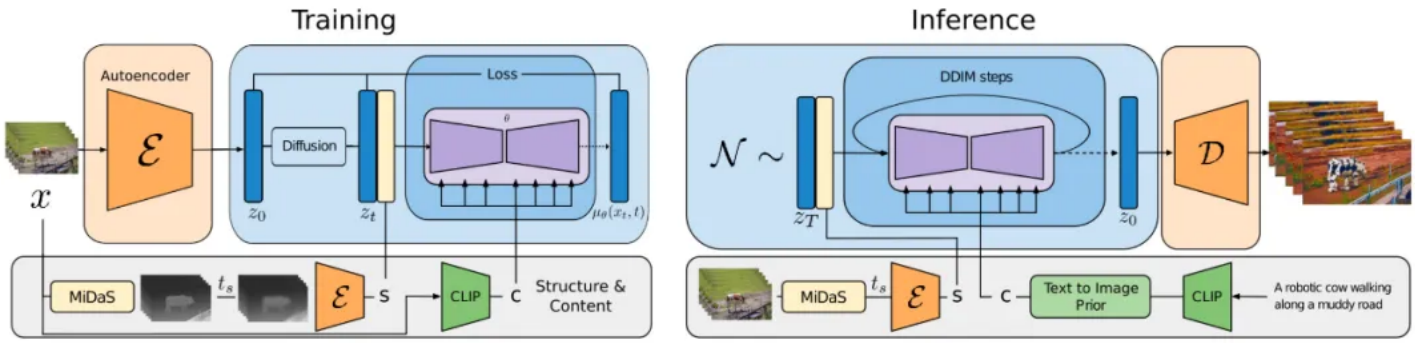

图 9：Gen-1 模型的训练流程概况。

Blattmann et al. 在 2023 年提出的 Video LDM 首先是训练一个 LDM（隐扩散模型）图像生成器。然后微调该模型，使之得到添加了时间维度的视频。这个微调过程仅用于那些在编码后的图像序列上新增加的时间层。Video LDM 中的时间层 {☒☒☒ ∣ ☒= 1,⋯,☒} （见图 10）会与已有的空间层 ☒☒☒ 交错放置，而这些空间层在微调过程中会保持冻结。也就是说，这里仅微调新参数 ☒，而不会微调预训练的图像骨干模型参数 ☒。Video LDM 的工作流程是首先生成低帧率的关键帧，然后通过 2 步隐含帧插值过程来提升帧率。

长度为 ☒ 的输入序列会被解释成用于基础图像模型 ☒ 的一批图像（即 ☒ · ☒），然后再调整为用于 ☒☒☒ 时间层的视频格式。其中有一个 skip 连接通过一个学习到的融合参数 ☒ 导向了时间层输出 ☒' 和空间输出 ☒ 的组合。在实践中，实现的时间混合层有两种：(1) 时间注意力，(2) 基于 3D 卷积的残差模块。

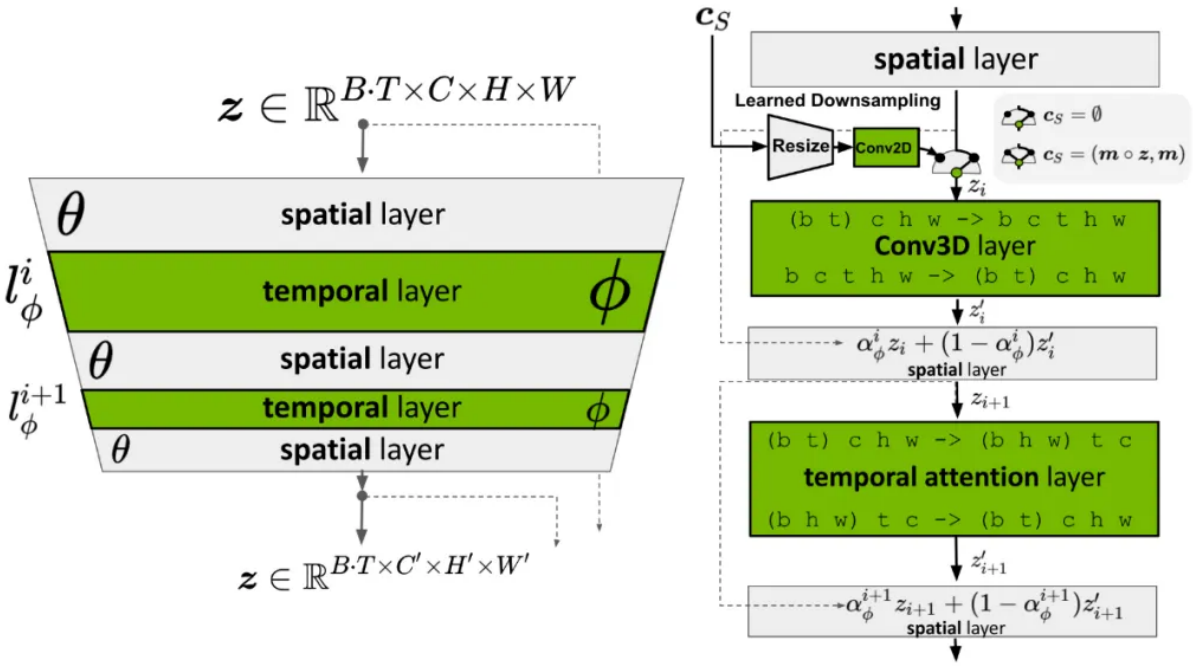

图 10：一个用于图像合成的预训练 LDM 被扩展成一个视频生成器。B、⍰、⍰、⍰、⍰ 分别是批量大小、序列长度、通道数、高度和宽度。⍰_S 是一个可选的条件 / 上下文帧。

但是，LDM 的预训练自动编码器依然还有问题：它只能看见图像，永远看不见视频。直接使用它来生成视频会产生闪动的伪影，这样的时间一致性就很差。因此 Video LDM 向解码器添加了额外的时间层，并使用一个用 3D 卷积构建的逐块时间判别器在视频数据进行微调，同时编码器保持不变，这样就依然还能复用预训练的 LDM。在时间解码器微调期间，冻结的编码器会独立地处理视频中每一帧，并使用一个视频感知型判别器强制在帧之间实现在时间上一致的重建。

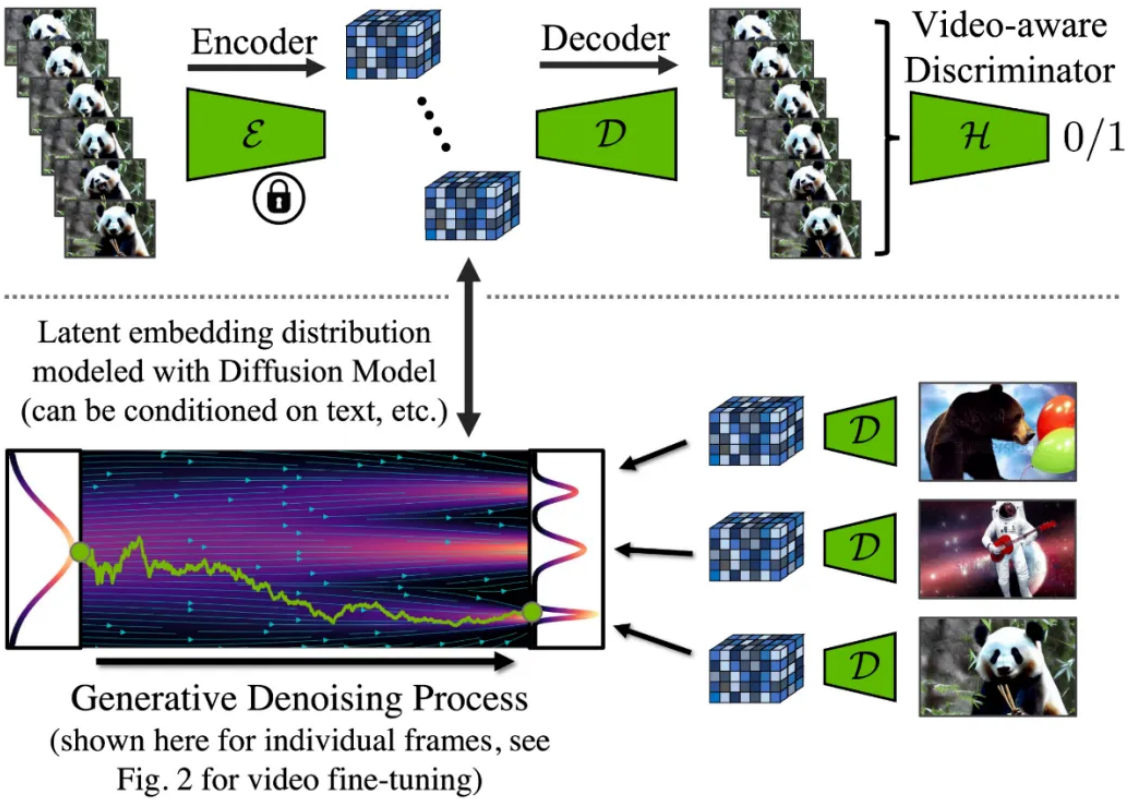

图 11：视频隐扩散模型中自动编码器的训练工作流程。其中编码器的微调目标是通过新的跨帧判别器获得时间一致性，而编码器保持不变。

类似于 Video LDM，Blattmann et al. 在 2023 年提出的 Stable Video Diffusion（SVD）的架构设计也是基于 LDM，其中每一个空间卷积和注意力层之后都插入时间层，但 SVD 是在整个模型层面上执行微调。训练视频 LDM 分为三个阶段：

1. 文生图预训练很重要，有助于提升质量以及遵从 prompt 的能力。
2. 将视频预训练分开是有利的，理想情况下应当在更大规模的经过整编的数据集上进行。
3. 使用一段更小的、高视觉保真度的预先标注了字幕的视频进行高质量视频微调。

SVD 专门强调了数据集整编对模型性能的关键作用。他们使用了一个剪辑检测流程来从每段视频获得更多剪辑，然后对其使用三个不同的字幕标注器模型：(1) 用于中间帧的 CoCa，(2) 用于视频字幕的 V-BLIP，(3) 基于前两个标注使用 LLM 来进行标注。然后他们还能继续提升视频数据集，其做法包括移除运动更少的视频片段（通过以 2 fps 速度计算低光流分数进行过滤）、清除过多的文本（使用光学字符识别来识别具有大量文本的视频）、清除看起来不够美的视频（使用 CLIP 嵌入标注每段视频的第一帧、中间帧和最后帧并计算美学分数和文本 - 图像相似度）。实验表明，使用经过过滤的更高质量的数据集能得到更好的模型质量，即便这个数据集要小得多。

对于首先生成远距离关键帧然后再使用时间超分辨率进行插值的方法，其中的关键挑战是如何维持高质量的时间一致性。Bar-Tal et al. 在 2024 年提出的 Lumiere 则是采用了一种时空 U-Net（STUNet）

架构，其可在单次通过中一次性生成整段时间上持续的视频，这样就无需依赖 TSR（时间超分辨率）组件了。STUNet 会在时间和空间维度上对视频进行下采样，因此会在一个紧凑的时间 - 空间隐空间内具备很高的计算成本。

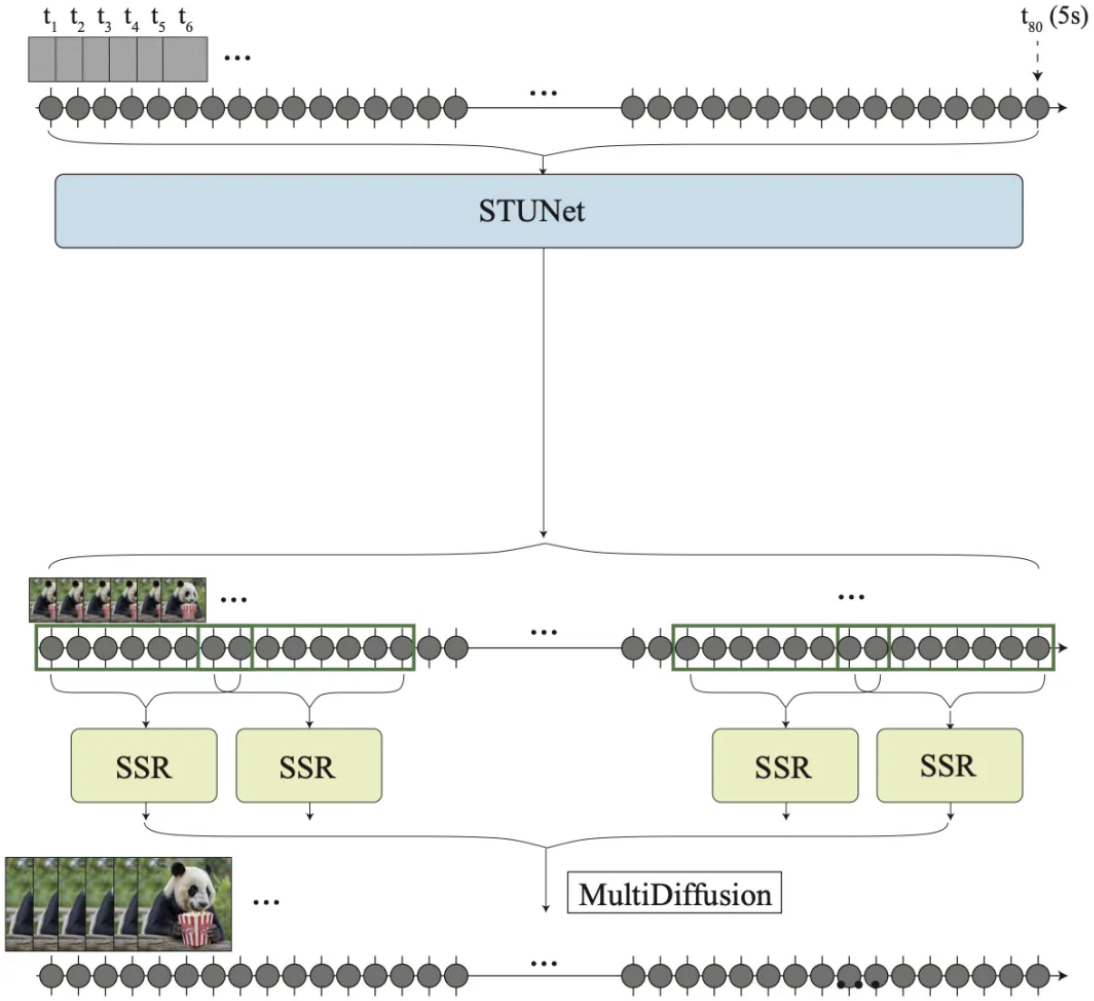

图 12：Lumiere 无需 TSR（时间超分辨率）模型。由于内存限制，经过扩展的 SSR 网络可以仅使用视频的短片段，因此 SSR 模型可以使用较短但重叠的视频片段集。

STUNet 在预训练文生图 U-Net 上扩展之后能够同时在时间和空间维度上对视频进行下采样和上采样。基于卷积的模块由预训练的文生图层构成，之后是分解的时空卷积。而在最粗粒度 U-Net 层面上的基于注意力的模块包含这个预训练文生图模块，之后是时间注意力。只有新添加的层需要进一步训练。

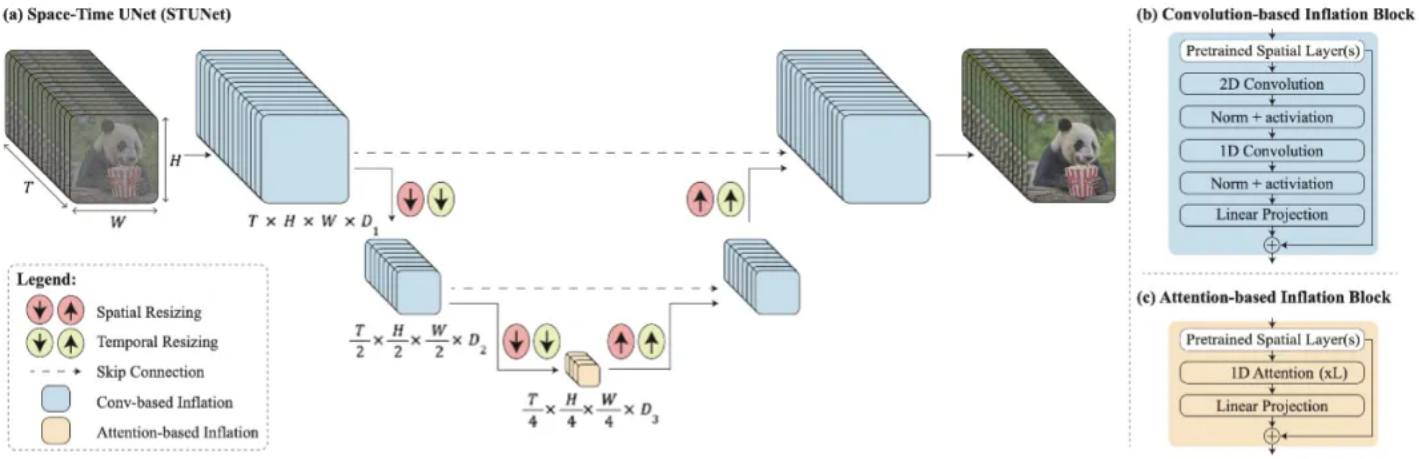

图 13：架构示意图：(a) 时空 U-Net（STUNet）、(b) 基于卷积的模块、(c) 基于注意力的模块。

无训练适应

也有可能不使用任何训练就让预训练的文生图模型输出视频，这多少有点让人惊讶。

如果我们直接简单地随机采样一个隐含代码的序列，然后用解码出来的对应图像构建一段视频，那么无法保证物体和语义在时间上的一致性。Khachatryan et al. 在 2023 年提出的 Text2Video-Zero 可实现零样本无训练的视频生成，其做法是让预训练的图像扩散模型具备用于时间一致性的两个关键机制。

1. 采样具有运动动态的隐含代码序列，以保证全局场景和背景的时间一致性。
2. 使用一个新的跨帧注意力（每一帧在第一帧上的注意力）重新编程帧层面的自注意力，以保证前景事物的上下文、外观和身份信息的一致性。

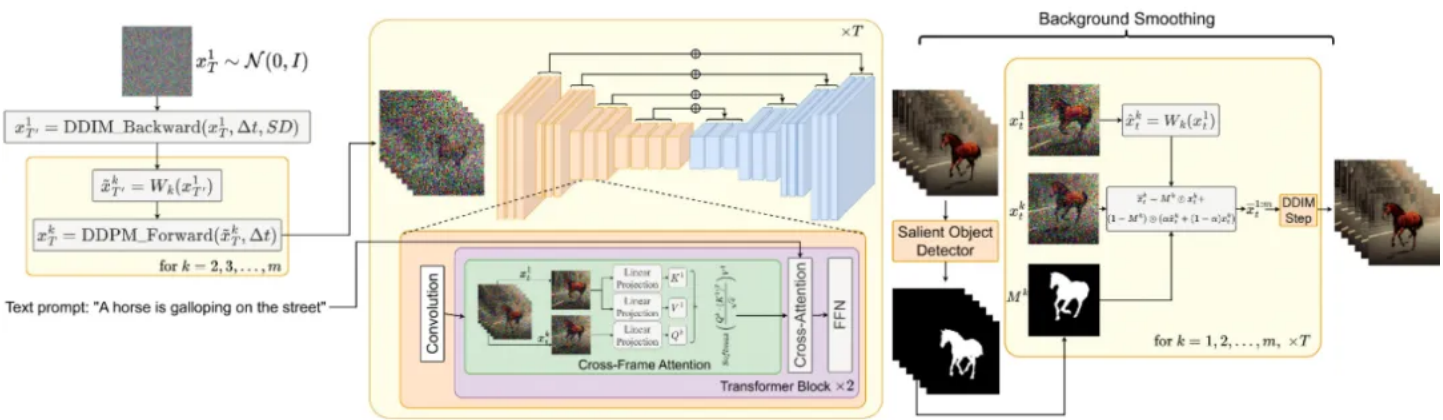

图 14：Text2Video-Zero 工作流程示意图。

下面用数学描述这个采样带有运动信息的隐含变量序列的过程：

1. 定义一个方向 ☒=(☒_☒, ☒_☒)∈☒² 来控制全局场景和相机运动；默认情况下，设定 ☒=(1, 1)。再定义一个超参数 λ>0，控制全局运动的数量。
2. 首先随机采样第一帧的隐含代码 ☒☒1☒☒(0,☒);
3. 使用预训练的图像扩散模型（例如论文中的 Stable Diffusion（SD）模型）执行 Δ☒≥0 DDIM 后向更新步骤，得到相应的隐含代码 ☒☒′1 ，其中 ☒'=☒-Δ☒。
4. 对于该隐含代码序列中的每一帧，使用一个扭曲操作（其定义为 ☒^☒=λ(☒-1)☒ ）执行相应的运动平移，得到
5. 最后对所有 ☒~☒′2:☒ 使用 DDIM 前向步骤，得到 ☒☒2:☒.

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

此外，Text2Video-Zero 还换掉了预训练 SD 模型中的自注意力层，并代之以一种参考第一帧的新型跨帧注意力机制。其目标是在生成的视频结果中保留前景事物的上下文、外观和身份信息。

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

另外还可以选择使用背景掩码，以让视频背景过渡更平滑并进一步提升背景一致性。假设我们已经使用某种方法得到了第 ☒ 帧相应的前景掩码 ☒_☒，然后背景平滑操作可根据以下背景矩阵将实际隐含代码和扩散步骤 ☒ 扭曲过的隐含代码融合起来：

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

其中☒☒☒ 是实际的隐含代码，☒~☒☒ 是在背景上扭曲的隐含代码，☒ 是一个超参数，该论文的实验中设定 ☒=0.6。

Text2Video-Zero 可与 ControlNet 结合起来，其中在每个扩散时间步骤 ⍰=⍰,...,1，每一帧都会在 ⍰⍰⍰ （⍰=1,...,⍰）上使用 ControlNet 预训练的副本分支，并将该 ControlNet 分支的输出添加到主 U-Net 的 skip 连接。

Zhang et al. 在 2023 年提出的 ControlVideo 的目标是基于文本 prompt 和运动序列（例如深度或边缘图） ⍰={⍰⍰}⍰=0⍰−1. 生成视频。该模型是基于 ControlNet 调整得到的，其中新增了三个机制：

1. 跨帧注意力：在自注意力模块中添加完整的跨帧交互。它引入了所有帧之间的交互，其做法是将所有时间步骤的隐含帧映射到 ⍰、⍰、⍰ 矩阵，这不同于 Text2Video-Zero（其是让所有帧都关注第一帧）。
2. 交替式帧平滑器（interleaved-frame smoother）机制是通过在交替帧上采用帧插值来减少闪烁效应。在每个时间步骤 ⍰，该平滑器会插值偶数或奇数帧，以平滑其相应的三帧剪辑。请注意，平滑步骤后帧数会随时间推移而减少。
3. 分层式采样器能在内存限制下保证长视频的时间一致性。一段长视频会被分割成多段短视频，其中每一段短视频都会选出一帧关键帧。该模型会使用完全跨帧注意力预生成这些关键帧以实现长期一致性，而每段相应的短视频都基于这些关键帧按顺序合成。

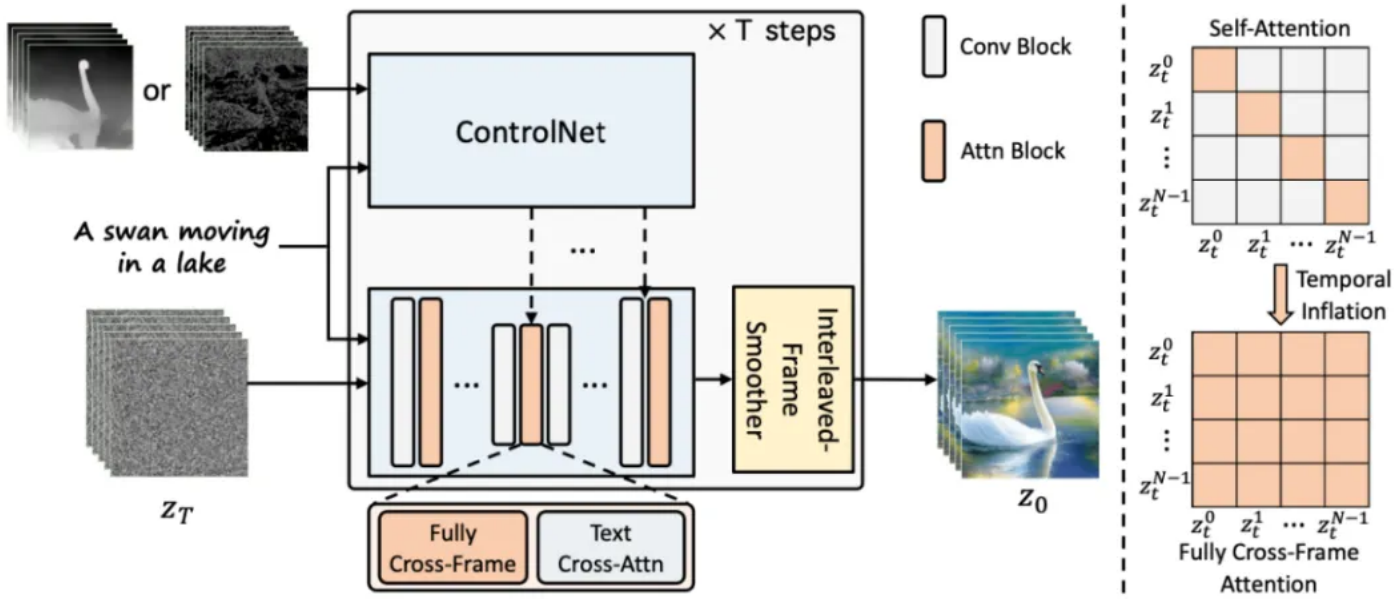

图 15：ControlVideo 概览。