文生视频大模型：从概率生成建模到时空扩散 Transformer（上册）

统一符号、数学基础、模型组件、MLLM Planner 与评测体系

面向具备深度学习基础的计算机博士生

2026-07-12

目录

前言：这不是模型名单，而是一套可推导、可实现、可评测的知识体系

文生视频（Text-to-Video, T2V）并不是“把文生图多生成几帧”。它要求模型同时解决四个彼此耦合的问题：

1. **语义建模**：理解实体、属性、动作、空间关系、时间顺序、镜头语言与风格；
2. **视觉生成**：产生高分辨率、纹理自然、构图合理的每一帧；
3. **动力学建模**：让物体、人物、相机和背景在时间轴上连续、可解释并尽量符合物理规律；
4. **计算系统设计**：在数量巨大的时空 token 上完成训练、并行和低成本推理。

因此，一份真正可用于入门科研的材料，需要把概率生成模型、视频压缩、扩散与流、Transformer、文本/多模态条件、数据工程和评测协议放进同一套符号体系中。本册的目标正是如此。

本资料按三册规划：

分册	核心内容	读完后应具备的能力
上册（本册）	任务定义、统一符号、Video VAE、扩散、Flow Matching、Video DiT、MLLM Planner、评价体系	能读懂主流视频生成论文的方法部分与评测部分；能从公式推到训练/采样伪代码
中册	主流模型谱系与工程细节，重点剖析 Wan、Bernini、CogVideoX、HunyuanVideo、LTX、Sora 类系统；数据、训练规模、分布式训练、推理成本	能复现开源模型、估算算力与显存、理解不同架构取舍
下册	长视频、可控生成、编辑、蒸馏、实时生成、音视频联合、世界模型、安全与研究选题；完整实验项目	能设计研究课题、搭建评测与消融实验、形成论文级研究方案

本册默认读者已经理解：线性代数、概率论、最大似然估计、反向传播、卷积网络、Transformer、自注意力和基本优化方法。对扩散模型没有前置要求。

> **资料时点**：文献与开源生态更新至 2026 年 7 月。模型产品的在线能力会变化；本册重点放在稳定的理论结构和可复现实验原则，而不是短期排行榜。

学习目标

完成本册后，你应当能够独立回答以下问题：

- 文生视频究竟在估计什么条件分布？T2V、I2V、V2V、R2V 的统一形式是什么？
- 为什么必须使用视频 VAE？压缩率如何决定 Transformer 序列长度与训练成本？
- DDPM 的噪声回归损失从何而来？为什么存在 ϵ、x_0、v 三种参数化？
- Flow Matching 与 Rectified Flow 在数学上学习什么？为什么现代大模型常用速度场预测？
- Video DiT 中 full attention、factorized attention、window attention 分别有什么复杂度与偏置？
- MLLM planner 与普通 prompt extension 有何本质区别？语义计划如何连接到 DiT renderer？
- FVD、IS、FID、CLIP-based alignment、VBench、MLLM-as-a-judge 各测什么，又遗漏什么？
- 如何设计一个不会被单一指标误导的 T2V 实验协议？

统一符号与阅读约定

视频生成论文常见的困难不是单个公式，而是不同论文对“干净样本”“噪声端点”“扩散时间”和“潜变量”使用相反记号。本册采用下列约定。

基本对象

符号	含义
$\mathbf{x} \in [0,1]^{T\times H\times W\times C}$	像素视频；T 帧，空间分辨率 $H \times W$，通常 $C = 3$
$\mathbf{y}$	原始文本提示或编辑指令
$\mathbf{c}$	条件的统称，可含文本、图像、视频、姿态、深度、轨迹、音频等
E_ϕ, D_ψ	Video VAE 编码器和解码器
$\mathbf{z}_\star = E_\phi(\mathbf{x})$	干净的数据潜变量；星号表示“数据端点”
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$	高斯噪声
$\tau \in [0,1]$	连续噪声/流时间；本册尽量不用 t 同时表示帧索引和扩散时间

符号	含义
$\mathbf{z}_\tau$	时间 τ 的中间潜变量
f_θ	通用生成网络；具体可为 U-Net、DiT 或其他时空网络
$v_\theta(\mathbf{z}_\tau, \tau, \mathbf{c})$	速度场或 flow prediction 网络
$s_\theta(\mathbf{z}_\tau, \tau, \mathbf{c})$	score 网络，近似 $\nabla_{\mathbf{z}} \log p_\tau(\mathbf{z} \mid \mathbf{c})$
N	送入 Transformer 的视频 token 数
d	Transformer 隐藏维度
L	Transformer 层数
n_h	注意力头数

时间轴与流端点

- 对视频内容时间，使用帧下标 $i \in \{0, 1, \dots, T-1\}$。
- 对扩散/流时间，使用 $\tau \in [0, 1]$。
- 在**扩散**章节中，$\tau = 0$ 通常代表数据、$\tau = 1$ 代表近似高斯噪声。
- 在 **Rectified Flow** 章节中，为符合运输直觉，定义 $\mathbf{z}^{(0)} \sim p_{\text{noise}}$、$\mathbf{z}^{(1)} \sim p_{\text{data}}$，并从 0 积分到 1。

两种方向可以互换，关键是实现中保持端点、目标速度和求解方向一致。

常见误区

看到论文写 x_0 时，不要立即认定它是数据。在 DDPM 文献中 x_0 通常是干净样本；在最优传输或 Rectified Flow 文献中，x_0 常是源分布，可能就是噪声。阅读时先查端点定义，再看损失。

1. 文生视频任务：从自然语言到条件视频分布

1.1 数学定义

给定文本提示 $\mathbf{y}$，文生视频模型学习条件分布

$$p_\theta(\mathbf{x} \mid \mathbf{y}),$$

其中 $\mathbf{x}$ 是一段视频，而不是唯一确定的标签。相同提示“金毛犬在海边追逐红色飞盘，手持摄影”可以对应无数合法视频：犬的外观、沙滩、天气、运动轨迹和镜头细节都可不同。因此，目标不是回归条件均值

$$\hat{\mathbf{x}} = \mathbb{E}[\mathbf{x} \mid \mathbf{y}],$$

因为多峰分布的均值通常表现为模糊且不自然的“平均视频”，而是学习一个可采样的高维条件分布。

现代系统通常在潜空间中建模：

$$\mathbf{z}_\star = E_\phi(\mathbf{x}), \qquad \mathbf{z}_\star \sim p_\theta(\mathbf{z} \mid \mathbf{y}), \qquad \hat{\mathbf{x}} = D_\psi(\hat{\mathbf{z}}_\star).$$

如果包含图像、源视频或结构控制，则统一写为

$$p_\theta(\mathbf{x}_{\text{target}} \mid \mathbf{c}), \quad \mathbf{c} = \{\mathbf{y}, \mathbf{x}_{\text{source}}, \mathbf{r}_1, \dots, \mathbf{r}_K\}.$$

这里 $\mathbf{r}_k$ 可以是参考主体、深度图、人体姿态、相机轨迹、分割掩码或音频。

1.2 主要任务及其条件形式

文生视频（Text-to-Video, T2V）

$$\mathbf{c} = \mathbf{y}, \qquad \hat{\mathbf{x}} \sim p_\theta(\mathbf{x} \mid \mathbf{y}).$$

这是最纯粹的开放域生成任务。文本给出语义约束，但没有像素锚点，因此多样性最大、主体一致性也最难。

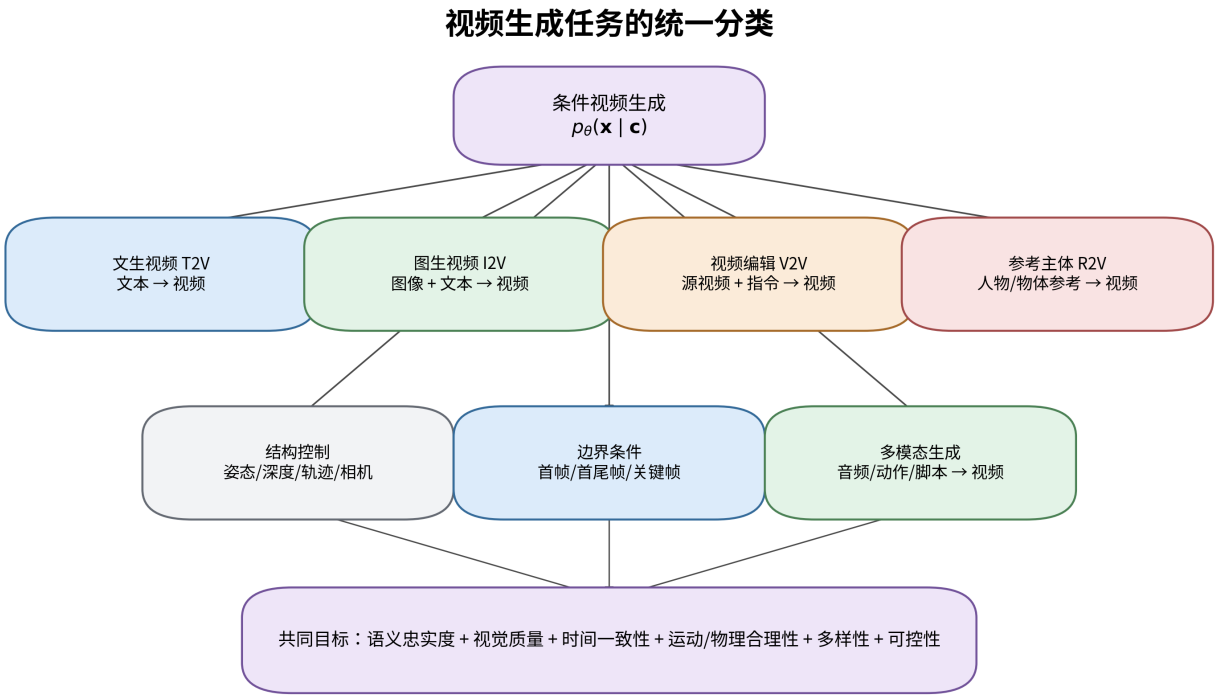

图 1: 视频生成任务的统一分类

图生视频（Image-to-Video, I2V）

$$\mathbf{c} = \{\mathbf{y}, \mathbf{I}_0\},$$

其中 $\mathbf{I}_0$ 常作为首帧或参考画面。I2V 把外观和构图固定下来，主要学习“如何动”。它通常比 T2V 更容易产生视觉上精致的结果，但动作可能弱、相机运动与对象运动容易纠缠。

首尾帧/关键帧生成（FLF2V / Keyframe-to-Video）

$$\mathbf{c} = \{\mathbf{y}, \mathbf{I}_0, \mathbf{I}_{T-1}\},$$

目标是生成满足边界条件的中间轨迹。难点类似桥接分布（bridge distribution）：不仅要从首帧自然演化，还必须抵达尾帧。

视频到视频编辑（Video-to-Video, V2V）

$$\hat{\mathbf{x}} \sim p_\theta(\mathbf{x}_{\text{target}} \mid \mathbf{x}_{\text{source}}, \mathbf{y}_{\text{edit}}).$$

编辑要求同时满足“改变什么”和“保持什么”。如果只使用高层语义条件，容易丢失身份、背景和运动；如果过度注入源视频低层特征，又可能无法执行编辑。

参考主体到视频（Subject/Reference-to-Video, R2V/S2V）

输入一张或多张人物、动物、商品或风格参考图，并要求在新场景中保持主体身份。它强调跨姿态、跨视角、跨光照的身份一致性。

结构可控视频生成

控制条件可能包括：

- 人体姿态序列 $\mathbf{P}_{0:T-1}$；
- 深度或表面法线 $\mathbf{D}_{0:T-1}$；
- 光流、点轨迹或物体路径 $\mathbf{R}$；
- 相机外参/内参或相机运动描述；
- 语义分割、边缘、草图、3D 场景信息。

其统一目标仍为 $p_\theta(\mathbf{x} \mid \mathbf{y}, \mathbf{R})$，区别只在条件编码与注入方式。

联合音视频生成

当模型同时生成视频 $\mathbf{x}$ 和音频 $\mathbf{a}$ 时，需要建模

$$p_\theta(\mathbf{x}, \mathbf{a} \mid \mathbf{y}) = p_\theta(\mathbf{x} \mid \mathbf{y}) p_\theta(\mathbf{a} \mid \mathbf{x}, \mathbf{y})$$

或使用共享连续时间、联合 latent/token 序列进行并行建模。关键不只是音质，而是口型、碰撞声、节奏和场景事件的同步。

1.3 “生成一段视频”隐含的六个目标

给定条件 $\mathbf{c}$，一个理想样本至少同时满足：

1. **语义忠实度（semantic faithfulness）**：主体、动作、属性、关系、数量和顺序符合提示；
2. **单帧质量（spatial fidelity）**：清晰、纹理合理、无明显结构畸变；
3. **时间一致性（temporal consistency）**：身份、外观、背景和光照不随机漂移；
4. **运动质量（motion quality）**：动作幅度、速度、加速度和相机运动自然；
5. **分布多样性（diversity）**：不同随机种子产生不同但合法的样本，而非模式坍塌；
6. **物理与因果合理性（physical/causal plausibility）**：物体持续存在、接触关系和动力学不明显违背常识。

这些目标并非总是同向：强 classifier-free guidance 往往提高文本一致性，却可能牺牲多样性并放大伪影；强首帧约束提高外观一致性，却可能压低动作幅度。

1.4 为什么视频比图像难得多

维度爆炸

一张 1024×1024 RGB 图像约有 3.1 百万标量；一段 5 秒、24 fps、1280×720 的视频有

$$120 \times 1280 \times 720 \times 3 \approx 3.32 \times 10^8$$

个像素标量。即使采用潜空间压缩，token 数仍可能达到数万甚至十万。

时间维不是独立复制

若逐帧独立采样

$$p(\mathbf{x}_{0:T-1} \mid \mathbf{y}) \overset{\text{错误近似}}{=} \prod_{i=0}^{T-1} p(\mathbf{x}_i \mid \mathbf{y}),$$

每帧都可能单独很好看，但人物服装、背景细节和物体位置会闪烁。正确模型必须捕获

$$p(\mathbf{x}_{0:T-1} \mid \mathbf{y})$$

中的高阶时空相关性。

文本中动作词的监督弱

互联网 caption 往往描述“有什么”，不精确描述“何时发生、如何发生”。例如“一个人做饭”可能覆盖切菜、搅拌、倒油、装盘等多阶段事件。视频数据量虽大，高质量动作-时间对齐标注却稀缺。

相机运动造成观测混淆

像素运动可以来自：物体运动、非刚体形变、相机平移/旋转/变焦、遮挡显露或光照变化。模型若不进行因子分解，常把“相机环绕静止雕像”误学成“雕像变形”。

长期一致性要求隐式记忆

短片中看不见的对象可能稍后重新出现。模型需要某种状态或记忆机制维持对象身份、场景布局与未观测属性。普通固定上下文 Transformer 仅有隐式记忆，序列变长后容易漂移。

1.5 技术路线的历史压缩图

可以把视频生成的发展粗略分成五类思想：

路线	代表性思想	主要优点	主要瓶颈
自回归像素/token	按时间或离散视觉 token 逐个预测	似然清晰，适合序列建模	解码慢，误差累积，长序列昂贵
GAN	生成器与判别器对抗	单次前向快，早期视频清晰度较好	训练不稳、模式坍塌、覆盖不足
VAE	学习连续潜空间并最大化 ELBO	压缩、可插值、训练稳定	单独使用时细节偏平滑
扩散模型	从数据逐步加噪，再学习反向去噪	覆盖好、质量高、条件控制灵活	采样多步、算力大
Flow Matching / Rectified Flow	直接学习噪声到数据的连续速度场	目标简洁、训练稳定、利于少步采样	仍依赖高质量网络与数值求解

现代大型 T2V 系统通常是组合：**Video VAE + 文本/多模态编码器 + Video DiT + 扩散或 Flow Matching + CFG/蒸馏/并行推理**。

现代文生视频模型的典型“条件—生成—解码”管线

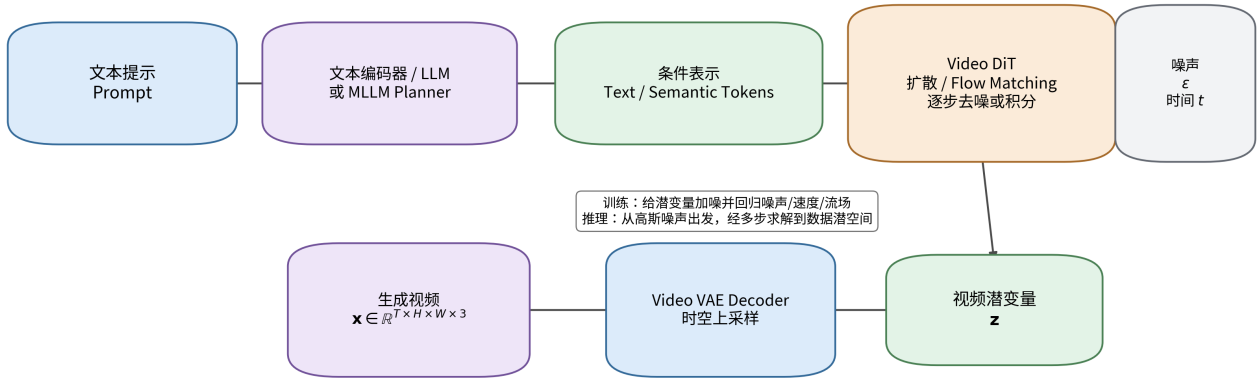

图 2: 现代文生视频模型的典型管线

核心结论

今天阅读一个新视频模型时，先不要被模型名吸引。先定位六件事：视频如何压缩、token 如何构造、生成目标是扩散还是流、条件如何注入、时空注意力如何降复杂度、评测协议是否可信。大多数模型创新都可映射到这六个问题。

2. 视频表示、潜空间与计算规模

2.1 视频张量、帧率与时长

视频可写为

$$\mathbf{x} = [\mathbf{x}_0, \dots, \mathbf{x}_{T-1}], \qquad \mathbf{x}_i \in \mathbb{R}^{H\times W\times C}.$$

若帧率为 r fps、时长为 S 秒，则理论帧数约为 $T = rS$。但训练数据常因抽帧、变帧率、解码误差和端点约定出现 $T \neq rS$。工程上必须记录真实时间戳，而不是只相信容器元数据。

帧率不是单纯的“质量参数”

帧率决定：

- 邻帧位移大小与运动可学习性；
- 时间 VAE 的压缩难度；
- DiT 的 token 数；
- 评测中 optical flow、flicker 和 FVD 的输入分布；
- 输出播放时的动作速度。

例如，训练用 16 fps、推理后按 24 fps 播放，会让动作整体加速 1.5 倍，除非通过插帧或时间条件校正。

2.2 三种建模空间

像素空间

直接对 $\mathbf{x}$ 建模保留全部信息，但维度过高。像素扩散在高分辨率视频上极其昂贵。

连续潜空间

Video VAE 将视频编码为

$$\mathbf{z}_{\star} = E_{\phi}(\mathbf{x}) \in \mathbb{R}^{T' \times H' \times W' \times C_z},$$

其中

$$T' \approx \left\lceil \frac{T}{f_t} \right\rceil, \quad H' \approx \left\lceil \frac{H}{f_h} \right\rceil, \quad W' \approx \left\lceil \frac{W}{f_w} \right\rceil.$$

f_t, f_h, f_w 分别为时间与空间压缩因子。连续 latent 适合扩散和流模型，因为可直接加高斯噪声并回归连续向量场。

离散视觉 token

VQ-VAE/VQGAN 类模型把 latent 量化到码本：

$$k_i = \arg\min_k \|\mathbf{h}_i - \mathbf{e}_k\|_2,$$

然后用自回归或掩码模型预测离散索引。优点是可复用语言模型式目标，缺点是量化误差、码本利用率和极长 token 序列。

2.3 压缩率与 token 数

若潜变量再以 3D patch (p_t, p_h, p_w) 划分，则 token 数为

$$N = \left\lceil \frac{T'}{p_t} \right\rceil \left\lceil \frac{H'}{p_h} \right\rceil \left\lceil \frac{W'}{p_w} \right\rceil.$$

每个 patch 展平后经线性层映射到 d 维：

$$\mathbf{h}_0 = \mathrm{PatchEmbed}(\mathbf{z}_{\tau}) \in \mathbb{R}^{N \times d}.$$

数值例子

考虑 81 帧、720×1280 视频。假设 VAE 压缩为 $(f_t, f_h, f_w) = (4, 8, 8)$，近似得到

$$T' = 21, \qquad H' = 90, \qquad W' = 160.$$

若 patch 为 $(1, 2, 2)$，则

$$N = 21 \times 45 \times 80 = 75{,}600.$$

这远大于常见语言模型的几千 token。单层全局注意力的分数矩阵有

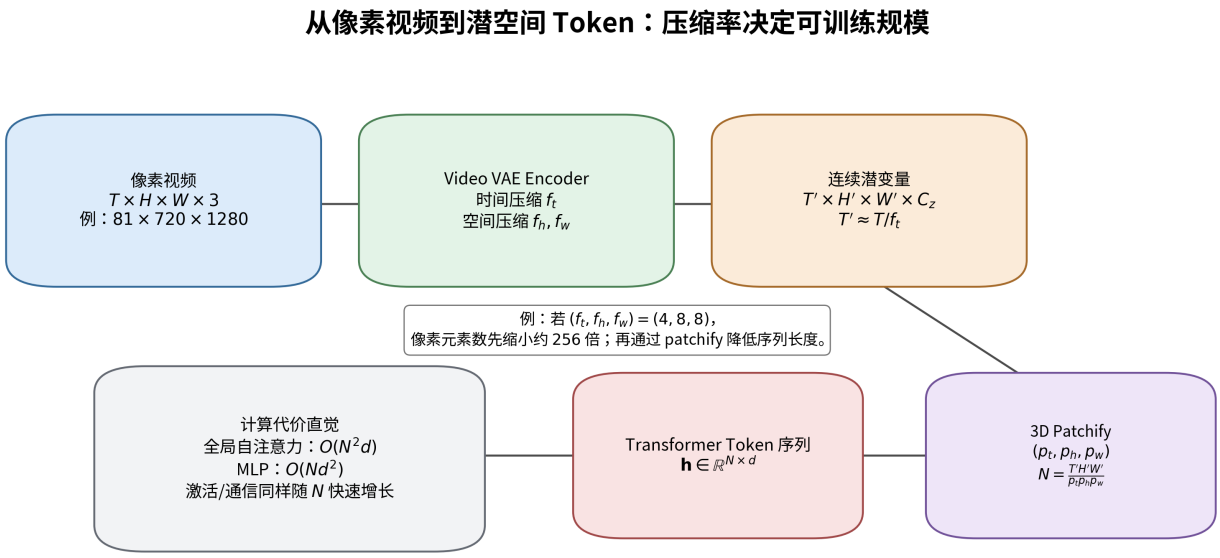

图 3: 视频潜空间与 token 化

$$N^2 \approx 5.72 \times 10^9$$

个元素；即使每个元素只占 2 字节，也超过 10 GB，尚未计入多头、梯度和其他激活。因此，大型视频模型必须依赖 FlashAttention、序列并行、分块/稀疏注意力、进一步压缩或更大的 patch。

空间压缩的平方收益

把空间压缩因子从 8 提升到 16，$H'W'$ 约缩小 4 倍，token 数缩小 4 倍，而全局注意力矩阵近似缩小 16 倍。这解释了高压缩 Video VAE 对 720p/1080p 生成的重要性。但压缩过强会损害文字、面部、手指和快速运动细节。

2.4 Transformer 计算复杂度

对 N 个 token、隐藏维度 d，一个标准 Transformer block 的主要 FLOPs 近似为：

- QKV 与输出投影：$O(Nd^2)$；
- 注意力相关矩阵：$O(N^2 d)$；
- 两层 MLP，扩张比 r：$O(rNd^2)$。

可粗略写为

$$\text{FLOPs}_{\text{block}} \approx aNd^2 + bN^2 d,$$

其中常数 a, b 与实现有关。短序列大宽度时 MLP/投影占主导；超长视频 token 时 $N^2 d$ 很快成为瓶颈。

显存不只来自参数

训练显存包括：

$$M \approx M_{\text{param}} + M_{\text{grad}} + M_{\text{optimizer}} + M_{\text{activation}} + M_{\text{communication buffer}}.$$

以 AdamW 和 bf16 参数为例，若不分片，参数、梯度、fp32 master weights 和两个动量状态可能使每参数占用约 16–20 字节。14B 参数模型仅模型状态就可能超过 200 GB；视频长序列还让激活显存极高。因此需 ZeRO/FSDP、激活检查点、混合精度和序列并行。

2.5 位置、时间和帧率条件

视频 token 至少有三维坐标 (i, h, w)。常见位置编码：

- 可学习绝对位置嵌入；
- 1D 展平位置编码；
- 分离的时间、行、列嵌入相加；
- 3D RoPE，将隐藏维度切分为时间/高度/宽度旋转子空间；
- 相对位置偏置；
- 可外推的连续坐标或归一化坐标。

还应显式提供 fps、时长或时间步距，否则模型可能把相同帧序列在不同播放速度下视为同一运动。

2.6 数据预处理的最小正确集

训练前至少应完成：

1. **解码审计**：检测损坏帧、黑屏、重复帧、音画不同步；
2. **镜头切分**：避免一个训练 clip 跨越硬切镜头，除非专门训练多镜头能力；
3. **时间采样**：固定帧数、随机时间跨度或多 fps bucket；
4. **分辨率与宽高比 bucket**：减少过度裁剪和 padding；
5. **caption 清洗/重写**：包含主体、动作、场景、镜头与风格；
6. **质量过滤**：清晰度、美学、压缩伪影、水印、字幕、NSFW；
7. **运动过滤**：去掉完全静止、剧烈抖动、重复循环或异常变速样本；
8. **去重**：视频级、帧级和语义级近重复；
9. **授权与隐私审计**：版权、肖像、敏感内容和可追溯性。

常见误区

“数据量”不能只按视频条数报告。一个 10 秒 clip 和一部长片不是同一统计单位。至少同时报告：clip 数、总小时数、平均时长、分辨率分布、fps 分布、去重后规模，以及各训练阶段实际看到的 token/帧数。

3. 条件生成建模基础：似然、潜变量、自回归与对抗学习

3.1 最大似然与条件分布

给定数据集

$$\mathcal{D} = \{(\mathbf{x}^{(n)}, \mathbf{c}^{(n)})\}_{n=1}^{M},$$

理想目标是最大化条件对数似然：

$$\max_{\theta} \sum_{n=1}^{M} \log p_\theta(\mathbf{x}^{(n)} \mid \mathbf{c}^{(n)}).$$

等价地，最小化真实条件分布与模型分布的前向 KL：

$$\mathbb{E}_{p_{\text{data}}(\mathbf{c})} D_{\text{KL}}(p_{\text{data}}(\mathbf{x} \mid \mathbf{c}) \| p_\theta(\mathbf{x} \mid \mathbf{c}))\,.$$

前向 KL 对遗漏真实模式惩罚较强，因此最大似然类方法倾向覆盖数据分布；但在超高维视频上直接计算似然通常不可行。

3.2 潜变量模型

引入潜变量 $\mathbf{z}$：

$$p_\theta(\mathbf{x} \mid \mathbf{c}) = \int p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c}) p_\theta(\mathbf{z} \mid \mathbf{c})\, d\mathbf{z}.$$

Video VAE 负责构建紧凑连续潜空间，扩散/流模型负责学习 $p_\theta(\mathbf{z} \mid \mathbf{c})$。这种分工把“高保真压缩”与“多模态分布生成”拆开。

3.3 变分下界（ELBO）

由于真实后验 $p_\theta(\mathbf{z} \mid \mathbf{x})$ 难以计算，引入近似后验 $q_\phi(\mathbf{z} \mid \mathbf{x})$：

$$\begin{aligned}\log p_\theta(\mathbf{x}) &= \log \int q_\phi(\mathbf{z} \mid \mathbf{x}) \frac{p_\theta(\mathbf{x}, \mathbf{z})}{q_\phi(\mathbf{z} \mid \mathbf{x})} d\mathbf{z} \\ &\geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x} \mid \mathbf{z})] - D_{\mathrm{KL}}(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})).\end{aligned}$$

第一项鼓励重建，第二项约束潜空间接近先验。实际高保真 Video VAE 常加入感知损失、对抗损失和时间一致性损失，因此不再是纯粹的标准 VAE 目标。

3.4 自回归分解

把离散视频 token 序列写为 $u_{1:N}$，则

$$p_\theta(u_{1:N} \mid \mathbf{c}) = \prod_{j=1}^{N} p_\theta(u_j \mid u_{<j}, \mathbf{c}).$$

训练使用 teacher forcing，目标是交叉熵：

$$\mathcal{L}_{\mathrm{AR}} = -\sum_{j=1}^{N} \log p_\theta(u_j^{\mathrm{gt}} \mid u_{<j}^{\mathrm{gt}}, \mathbf{c}).$$

优点是目标简单、可与 LLM 统一；缺点是：

- 视频 token 极多，串行解码慢；
- 训练看到真前缀，推理看到自身错误前缀，产生 exposure bias；
- 固定 raster order 未必符合视频的双向时空结构。

掩码生成模型（MaskGIT 类）通过并行预测被 mask token、迭代填充，缓解串行瓶颈。Bernini 的语义 planner 也采用了连续 ViT embedding 上的掩码迭代思想，但它预测的是语义计划，不是最终像素 token。

3.5 GAN 目标及其遗产

条件 GAN 的经典目标为

$$\min_G \max_D \mathbb{E}_{\mathbf{x} \sim p_{\mathrm{data}}}[\log D(\mathbf{x}, \mathbf{c})] + \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})}[\log(1 - D(G(\mathbf{z}, \mathbf{c}), \mathbf{c}))].$$

GAN 的生成只需一次前向传播，适合实时任务；但视频 GAN 的对抗博弈更不稳定，而且判别器可能只关注单帧纹理，忽略长期动力学。尽管大型开放域 T2V 已以扩散/流为主，对抗损失仍常用于训练 VAE decoder，以提高重建锐度。

3.6 为什么扩散/流成为主流

扩散和 Flow Matching 的共同优势：

- 训练目标接近标准监督回归，梯度稳定；
- 不需要同时训练一个强判别器；
- 容易注入文本、图像、控制信号；
- 对多峰分布覆盖通常优于 GAN；
- 可通过 guidance 在质量、语义一致性与多样性之间调节；
- 与大规模 Transformer 的 scaling 特性相容。

代价是推理需要多次网络求值。后续的蒸馏、一致性模型、少步流和缓存方法，主要都在解决这一问题。

3.7 条件注入的四种基本方式

1. **拼接（concatenation）**：把条件 token 与视频 token 放在同一序列；
2. **交叉注意力（cross-attention）**：视频 query 读取文本/参考 token 的 key-value；
3. **归一化调制（AdaLN/FiLM）**：由时间和条件产生 scale、shift、gate；
4. **额外残差分支（ControlNet/adapter）**：控制编码器输出逐层加到主干。

这些方式可组合。大型 DiT 往往用 AdaLN 注入噪声时间，用 cross-attention 或联合注意力融合文本，用通道拼接/额外 token 注入首帧或源视频 latent。

练习与自检

1. 证明最大化条件似然等价于最小化对真实条件分布的前向 KL，指出被省略的常数项。2. 对 $T = 81, H = 720, W = 1280$，分别计算 $(f_t, f_h, f_w) = (4, 8, 8)$ 与 $(4, 16, 16)$、patch $(1, 2, 2)$ 下的 token 数及全局注意力矩阵元素数。3. 解释为什么对逐帧生成结果计算很高的平均 CLIP 相似度，仍不能证明视频在时间上连续。

4. Video VAE：把不可承受的像素视频压缩为可生成的潜变量

4.1 为什么 VAE 是视频大模型的“地基”

对现代潜空间视频生成模型，Video VAE 不是附属模块。它同时决定：

- DiT 看到的序列长度；
- 可恢复的最高空间与时间频率；
- 文字、面部、手部和快速运动是否在生成前已经丢失；
- 解码时是否出现闪烁、拖影、色彩漂移；
- 能否分块编码/解码超长视频；
- T2I 与 T2V 是否能共享同一潜空间。

生成器不可能恢复 VAE 已系统性删除的信息。因此，比较两个视频生成模型时，如果它们使用不同 VAE，不能把全部差异归因于 DiT。

4.2 基本概率模型

标准高斯 VAE 编码器输出均值和对数方差：

$$\boldsymbol{\mu}_\phi(\mathbf{x}), \qquad \log \boldsymbol{\sigma}^2_\phi(\mathbf{x}).$$

通过重参数化采样

$$\mathbf{z}_\star = \boldsymbol{\mu}_\phi(\mathbf{x}) + \boldsymbol{\sigma}_\phi(\mathbf{x}) \odot \boldsymbol{\eta}, \qquad \boldsymbol{\eta} \sim \mathcal{N}(0, I).$$

解码器给出重建 $\hat{\mathbf{x}} = D_\psi(\mathbf{z}_\star)$。基础损失可写为

$$\mathcal{L}_{\mathrm{VAE}} = \lambda_{\mathrm{rec}} \mathcal{L}_{\mathrm{rec}} + \lambda_{\mathrm{KL}} D_{\mathrm{KL}}(q_\phi(\mathbf{z} \mid \mathbf{x}) \| \mathcal{N}(0, I)).$$

对对角高斯，KL 有闭式：

$$D_{\mathrm{KL}} = \frac{1}{2} \sum_j \left(\mu_j^2 + \sigma_j^2 - \log \sigma_j^2 - 1 \right).$$

实际视频 VAE 往往把 λ_{KL} 设得较小，以优先保证重建，而由后续 latent 标准化和生成模型适配潜分布。

4.3 重建损失不应只有像素 MSE

仅使用

$$\mathcal{L}_2 = \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2$$

会鼓励条件均值，导致高频纹理模糊。常见组合为

$$\mathcal{L}_{\mathrm{rec}} = \lambda_1 \|\mathbf{x} - \hat{\mathbf{x}}\|_1 + \lambda_p \mathcal{L}_{\mathrm{perceptual}} + \lambda_t \mathcal{L}_{\mathrm{temporal}} + \lambda_g \mathcal{L}_{\mathrm{GAN}}.$$

感知损失

用冻结图像/视频网络 Φ_l 的中间特征比较：

$$\mathcal{L}_{\text{perceptual}} = \sum_l w_l \|\Phi_l(\mathbf{x}) - \Phi_l(\hat{\mathbf{x}})\|_1.$$

图像感知网络有利于纹理，却可能忽视运动。视频 VAE 更理想的做法是加入 3D/视频特征或显式时间约束。

时间差分损失

一阶差分约束：

$$\mathcal{L}_{\Delta} = \sum_{i=1}^{T-1} \|(\hat{\mathbf{x}}_i - \hat{\mathbf{x}}_{i-1}) - (\mathbf{x}_i - \mathbf{x}_{i-1})\|_1.$$

也可用光流把前一重建帧 warp 到后一帧：

$$\mathcal{L}_{\text{warp}} = \sum_i \|\hat{\mathbf{x}}_{i+1} - \mathcal{W}(\hat{\mathbf{x}}_i, \mathbf{F}_{i\to i+1})\|_1.$$

注意遮挡区域不满足简单 warp，需要可靠性 mask。

对抗损失

空间判别器关注单帧锐度，时间判别器或 3D 判别器关注短期运动。其风险是引入 hallucination：重建看起来更真实，却不再忠实于输入细节。对于编辑和身份保持任务，忠实度往往比“想象出的锐度”更重要。

4.4 2D、3D 与因果 Video VAE

逐帧 2D VAE

对每帧独立编码：

$$\mathbf{z}_i = E_{2D}(\mathbf{x}_i).$$

优点是可直接复用成熟图像 VAE，缺点是没有时间压缩，并可能造成逐帧编码抖动。

3D VAE

使用 3D convolution/attention 在 (T, H, W) 上联合编码，能进行时间压缩并捕获局部运动。但普通 3D 卷积在编码第 i 帧时可能读取未来帧，难以流式处理。

因果 Video VAE

时间卷积只读取当前及过去：

$$\mathbf{h}_i = \sum_{k=0}^{K-1} W_k *_{2D} \mathbf{h}_{i-k}.$$

因果结构支持 chunk-by-chunk 编码/解码和任意长度视频，但每个 chunk 边界需要缓存历史状态；如果缓存丢失，会在边界产生跳变。

2D+1D 分解

将空间卷积与时间卷积分开：

$$\mathbf{h}' = \text{Conv}_{2D}(\mathbf{h}), \qquad \mathbf{h}'' = \text{Conv}_{1D\text{-time}}(\mathbf{h}').$$

它显著降低 3D 卷积成本，是实用设计。也可以只在少数层执行时间下采样，把大部分网络保留为图像友好的 2D 结构。

4.5 时间压缩的边界问题

若时间下采样因子为 $f_t = 4$，常见长度关系不一定是严格 $T' = T/4$。带 causal padding 的卷积可能满足

$$T' = \left\lfloor \frac{T-1}{f_t} \right\rfloor + 1.$$

这解释了为什么一些模型偏好 $T = 4k+1$，例如 81 帧：压缩后得到 21 个时间 latent。推理时若给出不兼容帧数，代码通常会裁剪、padding 或产生错误的最后几帧。

4.6 Latent 标准化

VAE latent 的每通道方差通常不等于 1。为使噪声日程和优化稳定，可使用标量或通道级缩放：

$$\tilde{\mathbf{z}}_\star = \frac{\mathbf{z}_\star - \boldsymbol{\mu}_z}{\boldsymbol{\sigma}_z}.$$

若训练与推理使用不同 scaling factor，结果会严重偏色或崩坏。开源权重中的 `scaling_factor`、`shift_factor`、latent mean/std 必须与 VAE 配套。

4.7 Tiling、chunking 与任意长度解码

空间 tiling

把 latent 划成重叠块，分别解码，再以窗口加权融合。若没有 overlap，卷积感受野在块边界被截断，出现网格缝。

时间 chunking

对长视频按时间块解码，并缓存 causal convolution 状态。对非因果模型，可采用重叠时间窗口和 cross-fade，但增加重复计算。

峰值显存估计

即使 DiT latent 能放入显存，VAE 解码到像素空间时的高分辨率激活也可能 OOM。推理系统应分别测量：文本编码、DiT 去噪、VAE 解码三个阶段的峰值，而不是只报模型权重大小。

4.8 VAE 的典型失败模式

现象	可能原因	建议诊断
所有模型输出都有相同轻微闪烁	VAE 重建本身不稳定	对真实视频做 encode-decode，不经过 DiT
小字必然糊掉	空间压缩过强或感知损失忽视文字	用合成文字视频测 OCR 准确率
快速运动有拖影	时间压缩/时间卷积低通效应	按运动速度分桶测重建
chunk 边界跳变	causal cache 或 overlap 处理错误	对同一视频整段/分块解码做像素差
颜色整体偏移	latent scaling 或 decoder 归一化不匹配	检查均值方差、权重版本和精度
面部锐利但身份变化	adversarial loss 过强	增加 identity/feature reconstruction

核心结论

训练视频生成主干前，应先冻结一个通过“重建上限测试”的 Video VAE。至少报告 PSNR/SSIM/LPIPS、视频感知指标、时间差分误差、快速运动子集、文字/人脸子集，以及长视频分块一致性。否则主干模型会为 VAE 缺陷背锅。

4.9 Video VAE 训练伪代码

```
# x: [B, C, T, H, W]
mu, logvar = encoder(x)
std = torch.exp(0.5 * logvar)
z = mu + std * torch.randn_like(std)
x_hat = decoder(z)
```

```
loss_l1 = (x - x_hat).abs().mean()
loss_kl = 0.5 * (mu.square() + logvar.exp() - logvar - 1).mean()
loss_perc = perceptual_distance(x_hat, x)
loss_temp = temporal_difference_loss(x_hat, x)
loss_gen = generator_adversarial_loss(video_discriminator, x_hat)

loss = (
    lambda_l1 * loss_l1
    + lambda_kl * loss_kl
    + lambda_perc * loss_perc
    + lambda_temp * loss_temp
    + lambda_adv * loss_gen
)
loss.backward()
optimizer.step()
```

真实系统还需要判别器交替更新、混合精度、分布式归一化、随机时长/分辨率 bucket 和 EMA。

5. 扩散模型：从前向加噪到反向生成

5.1 离散 DDPM 前向过程

令干净 latent 为 $\mathbf{z}_0 \equiv \mathbf{z}_\star$。定义一条固定马尔可夫加噪链：

$$q(\mathbf{z}_k \mid \mathbf{z}_{k-1}) = \mathcal{N}(\sqrt{1-\beta_k}\,\mathbf{z}_{k-1}, \beta_k I), \quad k = 1, \dots, K.$$

记

$$\alpha_k = 1 - \beta_k, \qquad \bar{\alpha}_k = \prod_{j=1}^{k} \alpha_j.$$

利用高斯组合，可直接从干净样本采样任意噪声级：

$$q(\mathbf{z}_k \mid \mathbf{z}_0) = \mathcal{N}(\sqrt{\bar{\alpha}_k}\mathbf{z}_0, (1-\bar{\alpha}_k)I),$$

即

$$\boxed{\mathbf{z}_k = \sqrt{\bar{\alpha}_k}\mathbf{z}_0 + \sqrt{1-\bar{\alpha}_k}\boldsymbol{\epsilon}}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(0, I).$$

这条闭式公式非常关键：训练时无需真的执行 k 次加噪，只需随机采样 k 和一份噪声。

5.2 反向过程

希望学习

$$p_\theta(\mathbf{z}_{k-1} \mid \mathbf{z}_k, \mathbf{c}) = \mathcal{N}(\boldsymbol{\mu}_\theta(\mathbf{z}_k, k, \mathbf{c}), \boldsymbol{\Sigma}_k).$$

真实后验在给定 $\mathbf{z}_0$ 时有闭式：

$$q(\mathbf{z}_{k-1} \mid \mathbf{z}_k, \mathbf{z}_0) = \mathcal{N}(\tilde{\boldsymbol{\mu}}_k, \tilde{\beta}_k I),$$

其中

$$\tilde{\boldsymbol{\mu}}_k = \frac{\sqrt{\bar{\alpha}_{k-1}}\beta_k}{1-\bar{\alpha}_k}\mathbf{z}_0 + \frac{\sqrt{\alpha_k}(1-\bar{\alpha}_{k-1})}{1-\bar{\alpha}_k}\mathbf{z}_k.$$

如果网络能从 $\mathbf{z}_k$ 预测 $\mathbf{z}_0$ 或噪声 $\boldsymbol{\epsilon}$，就能构造反向均值。

5.3 从 ELBO 到噪声回归

DDPM 的变分目标可分解为一系列 KL:

$$-\log p_\theta(\mathbf{z}_0) \le \mathbb{E}_q \left[D_{\mathrm{KL}}(q(\mathbf{z}_K \mid \mathbf{z}_0) \| p(\mathbf{z}_K)) + \sum_{k=2}^{K} D_{\mathrm{KL}}(q(\mathbf{z}_{k-1} \mid \mathbf{z}_k, \mathbf{z}_0) \| p_\theta(\mathbf{z}_{k-1} \mid \mathbf{z}_k)) - \log p_\theta(\mathbf{z}_0 \mid \mathbf{z}_1) \right].$$

在固定方差、适当参数化后，中间 KL 等价于带权噪声 MSE。经典简化目标为

$$\boxed{\mathcal{L}_\epsilon = \mathbb{E}_{\mathbf{z}_0, \boldsymbol{\epsilon}, k, \mathbf{c}} \left[\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{z}_k, k, \mathbf{c}) \|_2^2 \right]}.$$

这就是“给干净 latent 加噪，网络预测所加噪声”。

5.4 三种常见参数化

给定连续形式

$$\mathbf{z}_\tau = \alpha_\tau \mathbf{z}_\star + \sigma_\tau \boldsymbol{\epsilon},$$

其中常设 $\alpha_\tau^2 + \sigma_\tau^2 = 1$。

噪声预测 ϵ-prediction

网络输出 $\hat{\boldsymbol{\epsilon}}$，恢复数据:

$$\hat{\mathbf{z}}_\star = \frac{\mathbf{z}_\tau - \sigma_\tau \hat{\boldsymbol{\epsilon}}}{\alpha_\tau}.$$

高噪声端 $\alpha_\tau \to 0$ 时，数据恢复对误差很敏感。

数据预测 x_0-prediction

网络直接输出 $\hat{\mathbf{z}}_\star$。低噪声端直观，但不同噪声级目标尺度差异可能较大。

v-prediction

定义

$$\mathbf{v} = \alpha_\tau \boldsymbol{\epsilon} - \sigma_\tau \mathbf{z}_\star.$$

则

$$\mathbf{z}_\star = \alpha_\tau \mathbf{z}_\tau - \sigma_\tau \mathbf{v}, \qquad \boldsymbol{\epsilon} = \sigma_\tau \mathbf{z}_\tau + \alpha_\tau \mathbf{v}.$$

v-prediction 在高低噪声区间有较均衡的尺度，因此在很多 latent diffusion 系统中稳定。

数学要点

ϵ、x_0 和 v 并不是三个不同生成理论，而是同一 $\mathbf{z}_\tau = \alpha_\tau \mathbf{z}_\star + \sigma_\tau \epsilon$ 关系下的可逆参数化。真正不同的是损失权重、时间采样、噪声日程和求解器。

5.5 Score 视角

扰动分布为 $p_\tau(\mathbf{z})$，score 定义为

$$\mathbf{s}_\tau(\mathbf{z}) = \nabla_{\mathbf{z}} \log p_\tau(\mathbf{z}).$$

对条件高斯 $q(\mathbf{z}_\tau \mid \mathbf{z}_\star)$，其条件 score 为

$$\nabla_{\mathbf{z}_\tau} \log q(\mathbf{z}_\tau \mid \mathbf{z}_\star) = -\frac{\mathbf{z}_\tau - \alpha_\tau \mathbf{z}_\star}{\sigma_\tau^2} = -\frac{\boldsymbol{\epsilon}}{\sigma_\tau}.$$

因此噪声预测与 score matching 直接相关:

$$\mathbf{s}_\theta(\mathbf{z}_\tau, \tau, \mathbf{c}) \approx -\frac{\boldsymbol{\epsilon}_\theta(\mathbf{z}_\tau, \tau, \mathbf{c})}{\sigma_\tau}.$$

在连续时间中，前向 SDE

$$d\mathbf{z} = f(\mathbf{z}, \tau)d\tau + g(\tau)d\mathbf{w}$$

有反向时间 SDE

$$d\mathbf{z} = \left[f(\mathbf{z}, \tau) - g(\tau)^2 \nabla_{\mathbf{z}} \log p_\tau(\mathbf{z})\right] d\tau + g(\tau)d\bar{\mathbf{w}}.$$

还存在具有相同边缘分布的 probability flow ODE:

$$\frac{d\mathbf{z}}{d\tau} = f(\mathbf{z}, \tau) - \frac{1}{2} g(\tau)^2 \nabla_{\mathbf{z}} \log p_\tau(\mathbf{z}).$$

这建立了扩散、score 和 ODE 采样之间的桥梁。

5.6 信噪比与损失加权

定义

$$\mathrm{SNR}(\tau) = \frac{\alpha_\tau^2}{\sigma_\tau^2}.$$

低噪声区 SNR 高，目标主要是修复细节；高噪声区 SNR 低，目标主要是确定全局语义和构图。若均匀采样时间并使用普通 MSE, 不同区间对训练的有效贡献并不均衡。

常见策略包括:

- 按 log-SNR 设计 noise schedule;
- Min-SNR weighting，截断过高 SNR 的权重;
- 对分辨率或时长调整时间分布;
- EDM 风格预条件，把输入、输出和 skip connection 按噪声尺度归一化。

5.7 Classifier-Free Guidance（CFG）

训练时以概率 p_{drop} 丢弃条件，令同一网络同时学习条件与无条件预测:

$$f_{\theta,\mathrm{cond}} = f_\theta(\mathbf{z}_\tau, \tau, \mathbf{c}), \qquad f_{\theta,\mathrm{uncond}} = f_\theta(\mathbf{z}_\tau, \tau, \emptyset).$$

推理时

$$\boxed{\hat{f} = f_{\mathrm{uncond}} + w(f_{\mathrm{cond}} - f_{\mathrm{uncond}})}$$

其中 $w \geq 1$ 为 guidance scale。几何上，它沿“条件相对于无条件的增量方向”外推。

- w 小: 多样性高，但可能忽略 prompt;
- w 大: 文本更贴合，但易过饱和、运动僵硬、结构畸变;
- 视频中 guidance 可随时间/噪声变化，例如高噪声阶段强引导语义，低噪声阶段减弱以保护细节。

负面提示常作为“无条件”分支的替代输入，但它不是严格无条件分布，而是另一个条件分布，效果依赖训练是否见过类似负向描述。

5.8 采样器

DDPM ancestral sampling

逐步采样并注入随机噪声，随机性强、步数多。

DDIM

构造非马尔可夫但保持相同训练边缘的采样过程；$\eta = 0$ 时确定性，可显著减少步数并支持 latent inversion。

Euler / Heun

对连续 ODE/SDE 数值积分。一阶 Euler 每步一次网络求值；二阶 Heun 通常每步两次求值但误差更小。

DPM-Solver 类

利用扩散 ODE 的半线性结构做高阶求解，常能在 10–30 步获得较好质量。求解器与模型参数化、噪声 schedule 必须匹配。

5.9 视频扩散中的噪声相关结构

若每帧噪声完全独立，模型需要同时去除大量时间不一致噪声；若所有帧共享同一噪声，又会压制独立运动。可构造

$$\boldsymbol{\epsilon}_i = \sqrt{\rho}\,\boldsymbol{\epsilon}_{\text{shared}} + \sqrt{1-\rho}\,\boldsymbol{\epsilon}_i^{\text{ind}},$$

其中 ρ 控制跨帧相关性。类似思想可用于 noise initialization、长视频窗口拼接或 motion prior，但必须避免产生静态复制。

5.10 训练伪代码

```
# z0: clean video latent; cond: text/reference conditions
k = torch.randint(1, K + 1, (batch_size,), device=z0.device)
eps = torch.randn_like(z0)
a = sqrt_alpha_bar[k].view(B, 1, 1, 1, 1)
s = sqrt_one_minus_alpha_bar[k].view(B, 1, 1, 1, 1)
zk = a * z0 + s * eps

# classifier-free condition dropout
cond_in = drop_condition(cond, probability=p_drop)
eps_hat = video_dit(zk, timestep=k, condition=cond_in)
loss = ((eps_hat - eps) ** 2 * weight[k]).mean()
loss.backward()
optimizer.step()
```

练习与自检

1. 从 $q(z_k|z_{k-1})$ 递推证明 $q(z_k|z_0)$ 的闭式。2. 从 $z_\tau = \alpha z_\star + \sigma\epsilon$ 与 $v = \alpha\epsilon - \sigma z_\star$ 推导 $z_\star$ 和 ϵ 的反解。3. 解释为什么 CFG 必须在同一噪声状态、同一时间步上比较条件与无条件输出。4. 设计实验分离“采样步数不足”和“Video VAE 重建上限”两类问题。

6. Flow Matching 与 Rectified Flow：直接学习噪声到数据的速度场

6.1 连续归一化流（CNF）

定义时间相关 ODE

$$\frac{d\mathbf{z}_\tau}{d\tau} = \mathbf{v}_\theta(\mathbf{z}_\tau, \tau, \mathbf{c}), \qquad \tau \in [0,1].$$

若初始分布 p_0 为高斯噪声，ODE 的流映射把 p_0 推送到 p_1。推理为

$$\mathbf{z}_1 = \mathbf{z}_0 + \int_0^1 \mathbf{v}_\theta(\mathbf{z}_\tau, \tau, \mathbf{c})d\tau.$$

理论上 CNF 可通过瞬时变量变换计算似然:

$$\frac{d}{d\tau}\log p_\tau(\mathbf{z}_\tau) = -\nabla\cdot\mathbf{v}_\theta(\mathbf{z}_\tau,\tau),$$

但大规模生成通常不显式优化散度似然，而使用 Flow Matching 的无需轨迹模拟（simulation-free）回归目标。

6.2 概率路径与条件路径

我们希望构造边缘分布路径 $p_\tau(\mathbf{z})$，满足

$$p_0 = p_{\text{noise}}, \qquad p_1 = p_{\text{data}}.$$

直接获得边缘速度场困难。Flow Matching 先对每对端点 $(\mathbf{z}^{(0)},\mathbf{z}^{(1)})$ 定义条件路径

$$p_\tau(\mathbf{z}\mid\mathbf{z}^{(0)},\mathbf{z}^{(1)})$$

及其可计算条件速度 $\mathbf{u}_\tau$，然后回归:

$$\boxed{\mathcal{L}_{\text{FM}} = \mathbb{E}_{\tau,\mathbf{z}^{(0)},\mathbf{z}^{(1)},\mathbf{z}_\tau}\left[\|\mathbf{v}_\theta(\mathbf{z}_\tau,\tau,\mathbf{c}) - \mathbf{u}_\tau(\mathbf{z}_\tau\mid\mathbf{z}^{(0)},\mathbf{z}^{(1)})\|_2^2\right].}$$

关键定理表明，在平方损失下，最优预测是条件速度的后验均值:

$$\mathbf{v}^*(\mathbf{z},\tau) = \mathbb{E}[\mathbf{u}_\tau\mid\mathbf{z}_\tau=\mathbf{z}].$$

这个边缘向量场能生成所设计的边缘概率路径。

6.3 最简单的 Rectified Flow 路径

采样

$$\mathbf{z}^{(0)}\sim\mathcal{N}(0,I), \qquad \mathbf{z}^{(1)}\sim p_{\text{data}}.$$

定义直线插值

$$\boxed{\mathbf{z}_\tau = (1-\tau)\mathbf{z}^{(0)} + \tau\mathbf{z}^{(1)}}$$

则单对端点的速度为常数:

$$\boxed{\mathbf{u}_\tau = \frac{d\mathbf{z}_\tau}{d\tau} = \mathbf{z}^{(1)} - \mathbf{z}^{(0)}.}$$

训练目标因此极其简单:

$$\mathcal{L}_{\text{RF}} = \mathbb{E}\left[\|\mathbf{v}_\theta(\mathbf{z}_\tau,\tau,\mathbf{c}) - (\mathbf{z}^{(1)}-\mathbf{z}^{(0)})\|_2^2\right].$$

注意：虽然每对样本的条件路径是直线，边缘最优速度 $\mathbf{v}^*(z,\tau)$ 是对所有可能端点配对的条件平均，生成轨迹不一定完全直线。更好的端点 coupling 或 reflow 可让轨迹更直，从而减少数值积分步数。

6.4 一般仿射概率路径

更一般地，令

$$\mathbf{z}_\tau = a_\tau\mathbf{z}^{(1)} + b_\tau\mathbf{z}^{(0)},$$

满足 $a_0=0, b_0=1, a_1=1, b_1=0$。速度为

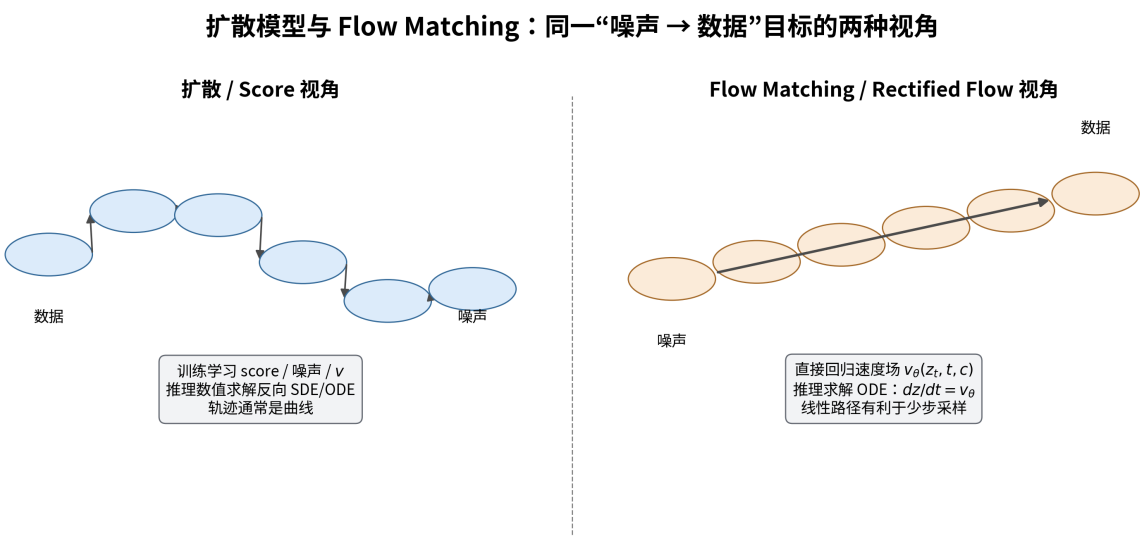

图 4: 扩散与 Flow Matching 的两种视角

$$\mathbf{u}_\tau = \dot{a}_\tau \mathbf{z}^{(1)} + \dot{b}_\tau \mathbf{z}^{(0)}.$$

扩散式高斯路径也是 Flow Matching 可支持的路径。区别不在“是否加噪”，而在训练目标如何表达和推理采用何种动态系统。

6.5 与扩散参数化的关系

对

$$\mathbf{z}_\tau = \alpha_\tau \mathbf{z}_\star + \sigma_\tau \epsilon,$$

沿固定端点微分：

$$\frac{d\mathbf{z}_\tau}{d\tau} = \dot{\alpha}_\tau \mathbf{z}_\star + \dot{\sigma}_\tau \epsilon.$$

因此 flow velocity 是数据和噪声的线性组合。给定特定 schedule，可在 x_0、ϵ、v 和 flow prediction 之间变换。实践中名字容易混乱：某些代码把“flow prediction”也命名为 `v_prediction`，但它未必等于扩散文献定义的 $v = \alpha\epsilon - \sigma x_0$。必须查看 scheduler 的公式。

6.6 推理：数值积分

从噪声 $\mathbf{z}_0$ 出发，用 Euler：

$$\mathbf{z}_{n+1} = \mathbf{z}_n + \Delta\tau_n \mathbf{v}_\theta(\mathbf{z}_n, \tau_n, \mathbf{c}).$$

Heun 先预测：

$$\tilde{\mathbf{z}}_{n+1} = \mathbf{z}_n + \Delta\tau_n \mathbf{v}_\theta(\mathbf{z}_n, \tau_n),$$

再校正：

$$\mathbf{z}_{n+1} = \mathbf{z}_n + \frac{\Delta\tau_n}{2}\left[\mathbf{v}_\theta(\mathbf{z}_n, \tau_n) + \mathbf{v}_\theta(\tilde{\mathbf{z}}_{n+1}, \tau_{n+1})\right].$$

网络函数求值次数（Number of Function Evaluations, NFE）比“步数”更准确：Heun 的 20 步约需 40 次网络调用。

6.7 时间采样与 flow shift

若均匀采样 $\tau \sim U(0,1)$，不同分辨率/序列长度的有效难度可能不均衡。大型 latent video 模型常对时间做非线性变换：

$$\tilde{\tau} = g_s(\tau),$$

其中 s 是 shift 参数，使训练或采样更关注高噪声/中噪声区。不同 scheduler 的 `flow_shift` 定义不统一，不能只比较数值大小。

直觉上，高分辨率 latent 具有更复杂的高频结构，适当调整时间密度可改变模型在全局布局与细节恢复之间的计算分配。

6.8 CFG 在 flow 中

同样可对速度场做 CFG：

$$\hat{\mathbf{v}} = \mathbf{v}_{\text{uncond}} + w(\mathbf{v}_{\text{cond}} - \mathbf{v}_{\text{uncond}}).$$

如果使用正负提示、参考图和多控制条件，可分别做多分支 guidance，但每增加一个分支就增加一次前向计算。工程上常采用：

- 条件批处理，把 cond/uncond 合并在 batch 维；
- guidance distillation，把双分支行为蒸馏到单分支；
- 随时间变化的 guidance；
- 只在关键层或关键步启用某些控制。

6.9 为什么 Flow Matching 适合视频大模型

1. 训练目标为直接的连续向量回归，易于大规模分布式训练；
2. 可选较直概率路径，利于少步采样；
3. 与 DiT、连续 VAE latent 和条件 cross-attention 自然兼容；
4. 能统一图像、视频、编辑等不同端点运输任务；
5. ODE 视角便于研究 trajectory、蒸馏与缓存。

但它不是自动的“更快”：如果向量场很弯、网络误差大或 CFG 过强，仍需很多 NFE。少步生成质量取决于训练路径、模型容量、时间参数化和求解器共同作用。

6.10 Flow Matching 训练伪代码

```
# z_data: encoded clean video latent
z_noise = torch.randn_like(z_data)
tau = torch.rand(batch_size, device=z_data.device)
t = tau.view(B, 1, 1, 1, 1)

z_t = (1.0 - t) * z_noise + t * z_data
velocity_target = z_data - z_noise

cond_in = drop_condition(cond, probability=p_drop)
velocity_pred = video_dit(z_t, continuous_time=tau, condition=cond_in)
loss = weighted_mse(velocity_pred, velocity_target, tau)
loss.backward()
optimizer.step()
```

采样：

```
z = torch.randn(latent_shape, device=device)
for tau, tau_next in time_grid:
    v_cond = model(z, tau, cond)
    v_uncond = model(z, tau, null_cond)
    v = v_uncond + guidance_scale * (v_cond - v_uncond)
    z = z + (tau_next - tau) * v  # Euler; production may use Heun/UniPC
video = vae.decode(z)
```

练习与自检

1. 证明平方损失最优回归函数为条件期望 $E[u_\tau \mid z_\tau]$。2. 对一维双峰数据分布，画出随机独立 coupling 下的条件直线与边缘速度，解释“条件路径直”不等于“生成轨迹必直”。3. 在相同 NFE 下比较 Euler 与 Heun；在相同“步数”下比较会有

什么不公平？4. 查阅任一开源模型 scheduler，明确其端点方向、prediction type、time shift 与 CFG 公式。

7. Video DiT：在超长时空 token 上学习去噪器或速度场

扩散或 Flow Matching 只规定“学什么目标”；真正决定模型容量、计算复杂度和条件控制能力的是参数化网络。早期视频扩散多采用 3D U-Net，当前大规模系统则广泛采用 Diffusion Transformer（DiT）及其视频扩展。

7.1 从 U-Net 到 DiT

U-Net 的归纳偏置

卷积 U-Net 通过下采样、上采样和 skip connection 在多尺度上处理视觉特征。它的优势是：

- 局部性强，对小数据和局部纹理友好；
- 天然形成多尺度金字塔；
- 在中等分辨率下计算规律明确。

但视频大模型面临三个问题：

1. 文本和参考条件越来越多，卷积模块的条件注入较零散；
2. 长程时空关系需要很深网络才能传播；
3. Transformer 更容易沿“参数、数据、计算”三轴扩展，并复用语言/多模态模型中的并行和稳定化技术。

DiT 的基本思想

DiT 将噪声潜变量切成 patch token，用 Transformer 预测噪声、数据或速度：

$$\hat{\mathbf{u}} = f_\theta(\mathrm{Patchify}(\mathbf{z}_\tau), \tau, \mathbf{c}),$$

再将 token 输出 unpatchify 回与 $\mathbf{z}_\tau$ 相同的张量形状。这里 $\mathbf{u}$ 可代表 ϵ、x_0、扩散 v 或 flow velocity；网络结构本身并不决定 prediction type。

核心结论

“DiT” 是网络骨干，“DDPM/Flow Matching” 是训练目标，“Euler/Heun/DPM-Solver” 是数值采样器。这三层必须分开理解。一个 Video DiT 可以用 Flow Matching 训练，也可以用扩散噪声回归训练。

7.2 3D Patchify 与输出头

令 Video VAE latent 为

$$\mathbf{z}_\tau \in \mathbb{R}^{B\times C_z\times T'\times H'\times W'}.$$

使用 patch 大小 (p_t, p_h, p_w) 后，token 数

$$N = T_p H_p W_p, \quad T_p = \left\lceil \frac{T'}{p_t} \right\rceil, \quad H_p = \left\lceil \frac{H'}{p_h} \right\rceil, \quad W_p = \left\lceil \frac{W'}{p_w} \right\rceil.$$

若任一维度不能整除 patch size，实现必须显式 padding，并在 unpatchify 后裁掉 padding；否则 $\lceil\cdot\rceil$ 只是形式记号，张量 reshape 会失败。

每个 patch 含 $C_z p_t p_h p_w$ 个标量，经线性投影：

$$\mathbf{H}^{(0)} = \mathbf{Z}_{\mathrm{patch}}\mathbf{W}_{\mathrm{in}} + \mathbf{b}_{\mathrm{in}} \in \mathbb{R}^{B\times N\times d}.$$

最终输出头映射回 patch 维度：

$$\mathbf{U}_{\mathrm{patch}} = \mathbf{H}^{(L)}\mathbf{W}_{\mathrm{out}} + \mathbf{b}_{\mathrm{out}},$$

再通过 unpatchify 得到

$$\hat{\mathbf{u}} \in \mathbb{R}^{B\times C_z\times T'\times H'\times W'}.$$

Patch 大小的三重权衡

- 更大 patch：token 更少，注意力更便宜，但细粒度运动和小物体变差；
- 更小 patch：表示更精细，但序列急剧增长；
- 时间 patch $p_t > 1$：可显著降序列长度，但会把若干 latent 帧绑在同一 token 中，降低快速运动的时间分辨率。

因此，比较两个模型的参数量时，必须同时比较 VAE 压缩率和 patch 大小；只看“几 B 参数”不能反映一次前向的真实成本。

7.3 自注意力与多头注意力

给定 token 矩阵 $\mathbf{H} \in \mathbb{R}^{N\times d}$，单头注意力为

$$\mathbf{Q} = \mathbf{H}\mathbf{W}_Q, \quad \mathbf{K} = \mathbf{H}\mathbf{W}_K, \quad \mathbf{V} = \mathbf{H}\mathbf{W}_V,$$

$$\mathrm{Attn}(\mathbf{H}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_h}} + \mathbf{B}\right)\mathbf{V},$$

其中 $d_h = d/n_h$，$\mathbf{B}$ 可包含 mask 或相对位置偏置。多头注意力将不同头的输出拼接后投影：

$$\mathrm{MHA}(\mathbf{H}) = \mathrm{Concat}(\mathbf{O}_1, \dots, \mathbf{O}_{n_h})\mathbf{W}_O.$$

视频中的注意力并不只是在“看相邻帧”。全局 self-attention 允许任意时空 token 直接交换信息，因此可维护远距离主体身份、遮挡前后的对应关系和全局镜头布局；代价是 $O(N^2)$。

7.4 时间条件：从 sinusoidal embedding 到 AdaLN-Zero

生成网络必须知道当前噪声/流时间 τ。常先构造 Fourier 或 sinusoidal embedding:

$$\gamma(\tau) = [\sin(\omega_1\tau), \cos(\omega_1\tau), \dots, \sin(\omega_m\tau), \cos(\omega_m\tau)],$$

再经 MLP 得到时间向量

$$\mathbf{e}_\tau = \mathrm{MLP}(\gamma(\tau)).$$

DiT 常通过 adaptive LayerNorm（AdaLN）调制每层:

$$\mathrm{AdaLN}(\mathbf{h}; \mathbf{e}) = (1 + \mathbf{s}(\mathbf{e})) \odot \mathrm{LN}(\mathbf{h}) + \mathbf{b}(\mathbf{e}).$$

带门控的 block 可写为

$$\tilde{\mathbf{h}} = \mathbf{h} + \mathbf{g}_{\mathrm{attn}}(\mathbf{e}) \odot \mathrm{MHA}(\mathrm{AdaLN}(\mathbf{h}; \mathbf{e})),$$

$$\mathbf{h}^+ = \tilde{\mathbf{h}} + \mathbf{g}_{\mathrm{mlp}}(\mathbf{e}) \odot \mathrm{MLP}(\mathrm{AdaLN}(\tilde{\mathbf{h}}; \mathbf{e})).$$

AdaLN-Zero 将门控或末层投影近零初始化，使初始网络接近恒等映射，有利于深层扩散 Transformer 稳定训练。

条件向量 $\mathbf{e}$ 可组合时间、分辨率、帧率、宽高比、运动强度和 pooled text embedding:

$$\mathbf{e} = \mathbf{e}_\tau + \mathbf{e}_{\mathrm{fps}} + \mathbf{e}_{\mathrm{size}} + \mathbf{e}_{\mathrm{ratio}} + \mathbf{e}_{\mathrm{text,pool}}.$$

7.5 时空注意力的主要设计

令 $F = T_p$ 为时间 token 数，$S = H_p W_p$ 为空间 token 数，$N = FS$。

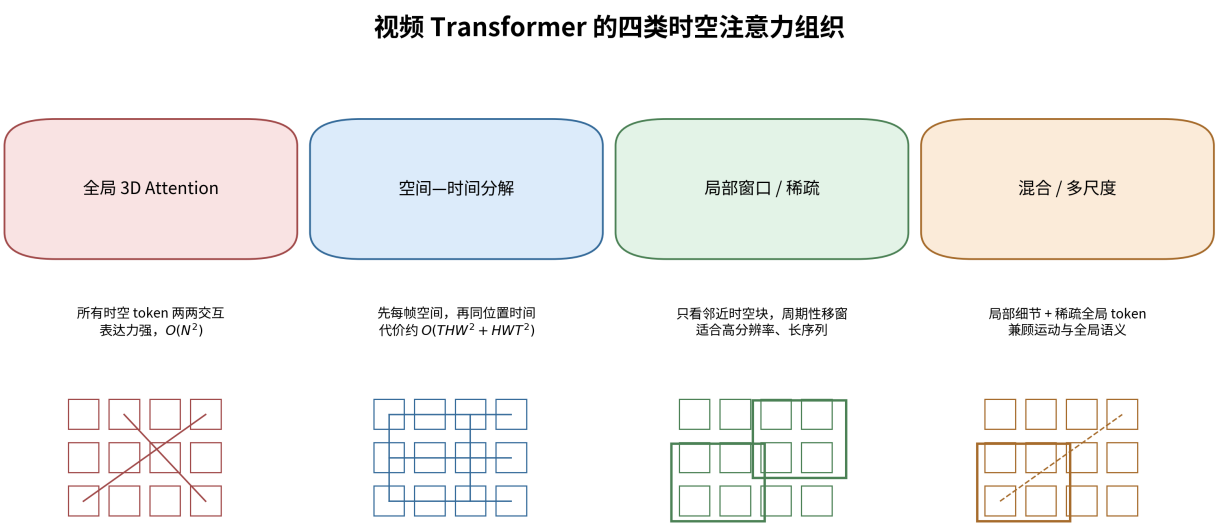

图 5: Video DiT 中常见的时空注意力模式

全局 3D 注意力（full spatiotemporal attention）

将所有 token 展平后一次注意力：

$$\text{Cost}_{\text{full}} = O((FS)^2 d) = O(F^2 S^2 d).$$

优点是表达力强、实现简单、无人工分解；缺点是长视频或高分辨率下极昂贵。

空间-时间分解注意力（factorized attention）

先在每帧内做空间注意力，再在每个空间位置跨时间做注意力：

$$\text{Cost}_{\text{fact}} = O(FS^2 d) + O(SF^2 d).$$

当 $S \gg F$ 时，第一项仍较大，但远低于 $F^2 S^2$。其归纳偏置是先整合帧内结构，再传播时间信息；缺点是一次 block 内的任意跨时空交互需要两步完成。

轴向注意力（axial attention）

分别沿时间、高度、宽度轴做注意力。复杂度近似

$$O(N(F + H_p + W_p)d).$$

它更省计算，但对斜向或大范围二维结构的直接建模较弱。

窗口或局部注意力

每个 token 只关注 $K = w_t w_h w_w$ 个局部 token：

$$\text{Cost}_{\text{window}} = O(NKd).$$

通过 shifted window、周期性全局层、稀疏全局 token 或分层结构恢复远程通信。窗口设计必须覆盖合理的最大位移，否则快速运动对象跨窗后难以关联。

块稀疏与混合注意力

一种常见折衷是：大部分层使用局部/分解注意力，少数层使用全局注意力；或对低分辨率全局 token 做全局通信，对高分辨率 token 做局部更新。

因果与双向注意力

- 离线 T2V 通常使用双向时间注意力，因为整段视频同时去噪；
- 流式生成或自回归延长需要因果 mask；
- 编辑任务可让 target token 关注全部 source token，但限制 source token 不被 target 污染。

7.6 3D RoPE：把时间、高度和宽度写进相位

对一对隐藏维度 (q_{2j}, q_{2j+1})，1D RoPE 施加旋转：

$$\begin{bmatrix} q'_{2j} \\ q'_{2j+1} \end{bmatrix} = \begin{bmatrix} \cos\varphi_j(p) & -\sin\varphi_j(p) \\ \sin\varphi_j(p) & \cos\varphi_j(p) \end{bmatrix} \begin{bmatrix} q_{2j} \\ q_{2j+1} \end{bmatrix}.$$

3D RoPE 将 head 维度划分给时间、行和列坐标：

$$\mathbf{q}' = R_t(i) \oplus R_h(h) \oplus R_w(w)\ \mathbf{q},$$

对 $\mathbf{k}$ 做相同旋转。内积自然依赖相对坐标差，因此有利于变分辨率和变时长外推。

3D RoPE 的工程细节

- 三个轴分到的维度比例不必相等；空间通常占更多维度；
- fps 改变时，相同帧索引代表不同真实时间，可把时间坐标设为 i/r；
- 超出训练长度时需处理频率外推、位置缩放或 NTK-style scaling；
- 不同宽高比 bucket 应使用真实二维坐标，而非简单把所有 token 当作一维序列。

7.7 文本条件：cross-attention、joint attention 与 MMDiT

令文本编码器输出

$$\mathbf{C}_y \in \mathbb{R}^{M \times d_c}.$$

Cross-attention

视频 token 作为 query，文本 token 作为 key/value：

$$\mathrm{XAttn}(\mathbf{H}, \mathbf{C}_y) = \mathrm{softmax}\left(\frac{(\mathbf{H}\mathbf{W}_Q)(\mathbf{C}_y\mathbf{W}_K)^\top}{\sqrt{d_h}} \right) (\mathbf{C}_y\mathbf{W}_V).$$

复杂度为 $O(NMd)$，通常远小于视频 self-attention。token-level cross-attention 能把不同视频区域动态对齐到“红色”“向左跑”“背景中的塔”等词。

Joint attention / multimodal Transformer

把文本 token 与视频 token 拼接或放入两个可交互流中：

$$[\mathbf{H}_v; \mathbf{H}_y] \xrightarrow{\text{joint attention}} [\mathbf{H}'_v; \mathbf{H}'_y].$$

MMDiT 风格通常让两种模态有独立投影和归一化，但共享注意力矩阵或相互通信。其优势是条件融合更深；代价是文本 token 也参与多层计算，且实现/并行更复杂。

Pooled modulation 与 token-level condition 的分工

- pooled embedding：适合全局风格、整体语义、时间/分辨率调制；
- token sequence：适合实体-属性绑定、关系和细粒度词对齐；
- 二者同时使用通常优于只用其中一种。

7.8 图像、视频和结构条件如何注入

通道拼接

把噪声 target latent、source latent、mask 等在 channel 维拼接：

$$\tilde{\mathbf{z}}_\tau = \mathrm{Concat}_C(\mathbf{z}_{\tau,\mathrm{target}}, \mathbf{z}_{\mathrm{source}}, \mathbf{m}).$$

简单有效，但改变输入层，并要求所有条件与 target 在时空网格上对齐。

Token 拼接

把参考图像/视频编码为视觉 token，作为额外序列。可通过 mask 指定 source/target 的可见关系。

Cross-attention

source feature 作为 key/value，target token 作为 query。适合不同分辨率、不同长度或多参考图。

残差控制分支

类似 ControlNet：复制部分 block 或建立轻量 condition encoder，输出残差

$$\mathbf{h}_l^+ = \mathbf{h}_l + \lambda_l \Delta \mathbf{h}_l^{\mathrm{ctrl}}.$$

零初始化可让新增控制分支从“不影响原模型”开始训练。

Feature replacement / anchoring

在去噪过程中将已知区域替换为按同一噪声水平加噪后的 source latent：

$$\mathbf{z}_\tau \leftarrow \mathbf{m} \odot \mathbf{z}_{\tau,\mathrm{known}} + (1-\mathbf{m}) \odot \mathbf{z}_{\tau,\mathrm{gen}}.$$

这是 inpainting、首帧保持和局部编辑的常见原则。

7.9 现代 Video DiT 的稳定化组件

QK-Norm

在注意力前归一化 query/key：

$$\tilde{\mathbf{q}} = \frac{\mathbf{q}}{\|\mathbf{q}\|_2}, \qquad \tilde{\mathbf{k}} = \frac{\mathbf{k}}{\|\mathbf{k}\|_2},$$

并使用可学习温度，减少超深网络中 attention logits 爆炸。

RMSNorm 与 LayerNorm

LayerNorm 同时去均值和缩放；RMSNorm 只按均方根缩放：

$$\mathrm{RMSNorm}(\mathbf{h}) = \frac{\mathbf{h}}{\sqrt{d^{-1}\sum_j h_j^2 + \epsilon}} \odot \mathbf{g}.$$

RMSNorm 更省计算，但是否优于 LayerNorm 取决于整体参数化。

SwiGLU / GEGLU

门控 MLP：

$$\mathrm{SwiGLU}(\mathbf{h}) = (\mathrm{SiLU}(\mathbf{h}\mathbf{W}_1) \odot \mathbf{h}\mathbf{W}_2)\mathbf{W}_3.$$

零初始化与残差缩放

输出投影近零初始化、AdaLN-Zero gate、按深度缩放残差，均可让大模型初始接近稳定恒等映射。

训练时数值检查

至少监控：

- loss 按时间区间分桶；
- gradient norm、update norm、参数 RMS；
- attention logit/max、Q/K norm；
- latent 与 target velocity 的均值/方差；
- bf16/fp16 overflow、NaN 首发层；
- 不同分辨率 bucket 的 loss 和吞吐。

7.10 长序列训练的并行方式

数据并行

不同 GPU 处理不同样本；通信梯度。最简单，但单个视频样本必须装入一张卡或一组模型并行卡。

FSDP / ZeRO

将参数、梯度和优化器状态分片。它解决模型状态显存，不直接解决单层长序列 activation 和 attention 显存。

Tensor parallel

把线性层和注意力头沿隐藏维度分片。适合超宽模型。

Sequence/context parallel

把 N 个视频 token 沿序列维分到多卡。注意力需 all-to-all 或 ring communication 交换 K/V 或局部结果。视频模型中，它往往比语言模型更关键，因为 N 极大。

Pipeline parallel

把层切成若干 stage。对超大模型有效，但 microbatch 太少会产生 pipeline bubble；视频单样本很大时需和 sequence parallel 联合。

Activation checkpointing

前向只保存部分边界 activation，反向时重算。显存显著下降，代价是增加 FLOPs。

数学要点

FSDP 解决“模型状态太大”，sequence parallel 解决“一个样本的 token 太长”，activation checkpointing 解决“中间激活太多”。三者作用对象不同，不能互相替代。

7.11 一个抽象的 Video DiT block

```
class VideoDiTBlock(nn.Module):
    def forward(self, video_tokens, time_cond, text_tokens, rope_cache):
        # time_cond may also include fps, resolution and pooled text
        shift1, scale1, gate1, shift2, scale2, gate2 = self.modulation(time_cond)

        h = self.norm1(video_tokens)
        h = h * (1.0 + scale1[:, None, :]) + shift1[:, None, :]
        h = self.self_attention(                  # RoPE rotates Q/K, not raw tokens
```

```
            h, rope_cache=rope_cache
        )                                           # full/factorized/windowed
        video_tokens = video_tokens + gate1[:, None, :] * h

        # Optional token-level semantic conditioning
        video_tokens = video_tokens + self.cross_attention(
            query=self.norm_cross(video_tokens),
            key_value=text_tokens,
        )

        h = self.norm2(video_tokens)
        h = h * (1.0 + scale2[:, None, :]) + shift2[:, None, :]
        h = self.mlp(h)
        video_tokens = video_tokens + gate2[:, None, :] * h
        return video_tokens
```

整网前向:

```
def video_dit(z_t, tau, text_tokens, meta):
    tokens, grid = patchify_3d(z_t)
    tokens = input_projection(tokens)
    rope_cache = build_3d_rope(grid, fps=meta.fps)

    cond = time_mlp(time_embedding(tau))
    cond = cond + metadata_embedding(meta) + pooled_text(text_tokens)

    for block in blocks:
        tokens = block(tokens, cond, text_tokens, rope_cache)

    velocity_tokens = output_head(final_norm(tokens, cond))
    return unpatchify_3d(velocity_tokens, grid)
```

7.12 阅读 Video DiT 论文时的核对表

1. VAE 压缩率和 latent channel 是多少?
2. 3D patch 大小是多少? 真实 token 数是多少?
3. attention 是 full、factorized、window、sparse 还是混合?
4. 文本是 cross-attention、joint attention 还是仅 pooled modulation?
5. 位置编码是否包含真实 fps/时间坐标?
6. prediction type 是 ϵ、x_0、扩散 v 还是 flow velocity?
7. 条件 dropout 和 CFG 如何实现?
8. 使用了哪些并行维度? 论文报告的是 GPU 数还是总 GPU-hours?
9. 训练分辨率/时长是否分阶段?
10. 推理成本是否包含 text encoder、VAE decode、CFG 双分支和后处理?

练习与自检

1. 对 $F = 21, S = 3600$，计算 full attention 与 factorized attention 的注意力元素数量比。2. 设计一种“每四层一次全局注意力、其余为窗口注意力”的 block pattern，说明信息跨窗传播所需层数。3. 解释为什么只增加 VAE 空间压缩率，可能同时改善 DiT 吞吐并恶化文字/人脸。4. 对一段变帧率视频，提出一种连续时间坐标编码，使真实 0.5 秒间隔而非帧索引决定 RoPE 相位。

8. 从文本条件到 MLLM Planner：语义理解、规划与像素渲染

传统 T2V 把文本 embedding 直接送入去噪器。随着提示包含多主体、复杂动作、镜头切换和长时间因果关系，仅靠一次 text encoding 很难形成稳定的“未来视频计划”。因此，现代系统开始把生成分成两层：先在语言或高层视觉语义空间规划，再在 VAE latent 空间渲染。

8.1 文本提示到底包含哪些变量

可把一个提示解析为结构化集合：

$$\mathbf{y} \mapsto \{\mathcal{O}, \mathcal{A}, \mathcal{R}_s, \mathcal{R}_t, \mathcal{C}, \mathcal{S}, \mathcal{M}\},$$

其中：

- $\mathcal{O}$：对象与人物；
- $\mathcal{A}$：属性，如颜色、材质、服装、数量；
- $\mathcal{R}_s$：空间关系；
- $\mathcal{R}_t$：时间关系与事件顺序；
- $\mathcal{C}$：相机、镜头、景别和运镜；
- $\mathcal{S}$：风格、光照、色调；
- $\mathcal{M}$：运动方式、速度、幅度和交互。

模型失败常不是“没理解整个句子”，而是某个绑定约束没有传到对应时空 token。例如：

- 红球和蓝立方体都出现，但颜色绑定互换；
- “先开门再坐下”中的两个事件都有，但顺序反了；
- “相机静止，汽车向右移动”被生成成汽车静止、相机左移。

8.2 文本编码器的三类选择

CLIP 类双塔编码器

训练目标对齐全局图文表示，优点是视觉语义强、推理快；缺点是长文本、计数、复杂语法和时间关系较弱。可使用 token-level hidden states，而不只使用最终 pooled vector。

T5/UL2 类编码器

来自大规模语言建模，长文本理解、句法和组合性通常更好。其表示未必天然与视觉空间对齐，需要生成模型在训练中学习映射。

Decoder-only LLM / MLLM

可进行指令理解、提示扩写、镜头规划和多轮控制。MLLM 还可读取参考图/源视频，输出文本计划、离散视觉 token 或连续视觉语义表示。代价是计算大、训练接口更复杂，也可能产生语言上合理但视觉上不可实现的计划。

双编码器或多编码器融合

系统可同时使用 CLIP 的视觉对齐和 T5/LLM 的语言理解：

$$\mathbf{C} = \mathrm{Fuse}(\mathbf{C}_{\mathrm{CLIP}}, \mathbf{C}_{\mathrm{T5/LLM}}).$$

融合方式包括 token 拼接、独立 cross-attention、门控加权或在不同层使用不同条件。

8.3 Prompt rewriting 不等于 planning

Prompt rewriting

把用户短提示扩成更详细文本：

$$\tilde{\mathbf{y}} = g_{\mathrm{LLM}}(\mathbf{y}).$$

例如补充镜头、光照、材质和动作副词。它仍把全部计划压缩在自然语言中，通常不包含与目标视频逐时空位置对齐的连续表示。

Semantic planning

引入中间语义变量 $\mathbf{s}$:

$$q_\omega(\mathbf{s} \mid \mathbf{y}, \mathbf{r}), \qquad p_\theta(\mathbf{z}_\star \mid \mathbf{s}, \mathbf{y}, \mathbf{r}),$$

从而

$$p(\mathbf{z}_\star \mid \mathbf{y}, \mathbf{r}) = \int p_\theta(\mathbf{z}_\star \mid \mathbf{s}, \mathbf{y}, \mathbf{r}) q_\omega(\mathbf{s} \mid \mathbf{y}, \mathbf{r}) d\mathbf{s}.$$

$\mathbf{r}$ 表示参考图或源视频。计划 $\mathbf{s}$ 可以是:

- 镜头列表和事件时间线;
- 关键帧或低帧率 storyboard;
- 3D/轨迹/姿态表示;
- 离散视觉 token;
- 来自 ViT/MLLM 的连续高层视觉 embedding。

真正的 planner 应满足：它的输出比原始文本更接近目标视频结构，并可被 renderer 直接消费。

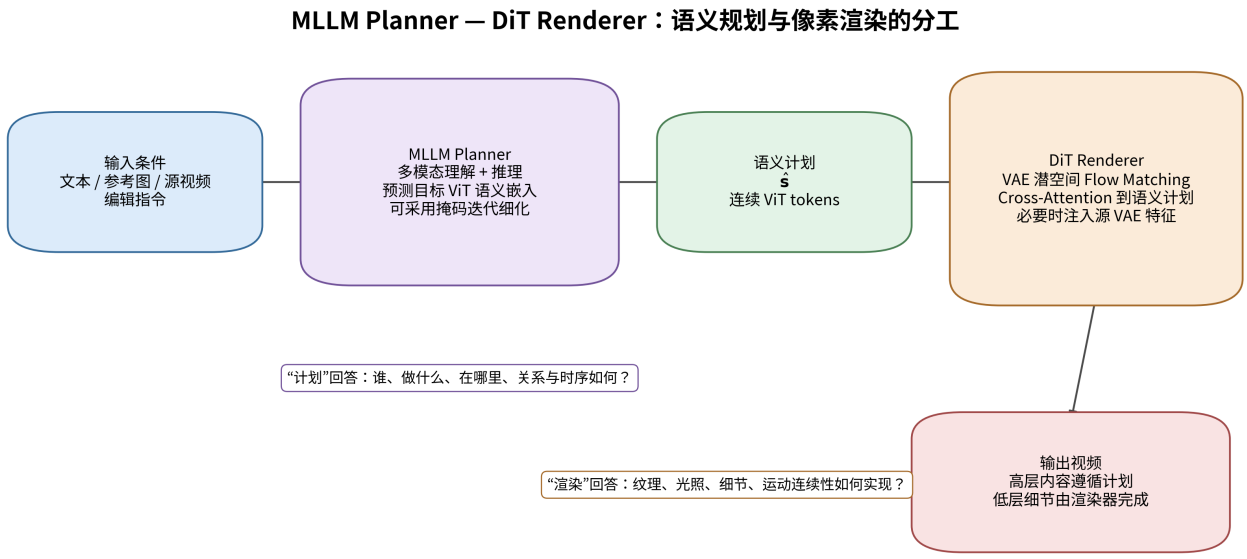

图 6: MLLM Planner 与 DiT Renderer 的两阶段语义接口

8.4 为什么要把 planning 与 rendering 分开

令 $\mathbf{S}$ 表示高层语义计划。条件互信息恒等式为

$$I(\mathbf{Z}; \mathbf{S} \mid \mathbf{Y}) = H(\mathbf{Z} \mid \mathbf{Y}) - H(\mathbf{Z} \mid \mathbf{S}, \mathbf{Y}).$$

因此，只要计划 $\mathbf{S}$ 确实携带与目标视频相关、而文本 $\mathbf{Y}$ 尚未给出的信息，就有

$$H(\mathbf{Z} \mid \mathbf{S}, \mathbf{Y}) < H(\mathbf{Z} \mid \mathbf{Y}),$$

即 renderer 面对的条件不确定性下降。等价的链式恒等式是

$$H(\mathbf{Z} \mid \mathbf{Y}) = H(\mathbf{S} \mid \mathbf{Y}) + H(\mathbf{Z} \mid \mathbf{S}, \mathbf{Y}) - H(\mathbf{S} \mid \mathbf{Z}, \mathbf{Y}).$$

直观上，直接 renderer 必须同时决定:

- 谁在何处；
- 何时做什么；
- 相机如何运动；
- 每个像素的纹理和噪声细节。

planner 先降低“高层结构”的条件熵，renderer 再专注于连续 VAE latent 的高保真生成。这类似语言生成中先列提纲再写正文，也类似机器人中 task planning 与 motion control 的分层。

但分层不是免费收益：若 planner 给出错误或过强的计划，renderer 可能无法纠正；若 $\mathbf{s}$ 信息不足，renderer 仍会漂移；若计划空间和 VAE latent 不对齐，条件会被忽略。

8.5 Bernini 式连续语义规划

Bernini 的核心抽象是：MLLM planner 不只输出自然语言，而是在预训练视觉编码器的连续语义空间中预测目标视频的表示；随后 Video DiT renderer 在 VAE latent 空间生成最终视频。

设视觉语义编码器为 F_{sem}，目标视频语义为

$$\mathbf{s}_\star = F_{\text{sem}}(\mathbf{x}_{\text{target}}).$$

planner 接收文本和可选 source/reference 条件：

$$\hat{\mathbf{s}}_\star = P_\omega(\mathbf{y}, \mathbf{s}_{\text{source}}, \text{instruction}).$$

Bernini 的公开方法具有一个容易混淆的**两层生成结构**：

1. 外层采用 masked generative modeling。训练时随机遮蔽一部分目标 ViT token；推理时从全 mask 开始，按 mask-ratio schedule 逐轮填充目标语义序列；
2. 内层的轻量 ViT-embedding decoder 在每个待恢复位置上，以 Flow Matching 预测连续 ViT embedding，而不是做离散码本分类。

对一个被遮蔽的目标语义 token，可用直线路径表示其连续解码目标：

$$\mathbf{s}_\tau = (1-\tau)\boldsymbol{\xi} + \tau \mathbf{s}_\star, \qquad \boldsymbol{\xi} \sim \mathcal{N}(0, I),$$

$$\mathcal{L}_{\text{plan}} = \mathbb{E}\left[\left\|\mathbf{u}_\omega(\mathbf{s}_\tau, \tau, \mathbf{h}_{\text{MLLM}}) - (\mathbf{s}_\star - \boldsymbol{\xi})\right\|_2^2\right],$$

其中 $\mathbf{h}_{\text{MLLM}}$ 是由文本、源视觉输入、当前已填充目标 token 与 mask pattern 联合产生的上下文化 hidden state。这里的 Flow Matching 是**连续 token 解码器的目标**；外层仍是逐轮减少 mask 的并行迭代，而不是对整张计划只积分一条全局 ODE。

renderer 在 VAE latent 空间学习：

$$\mathbf{z}_\tau = (1-\tau)\boldsymbol{\epsilon} + \tau \mathbf{z}_\star,$$

$$\mathcal{L}_{\text{render}} = \mathbb{E}\left[\left\|\mathbf{v}_\theta(\mathbf{z}_\tau, \tau, \mathbf{y}, \hat{\mathbf{s}}_\star, \mathbf{z}_{\text{source}}) - (\mathbf{z}_\star - \boldsymbol{\epsilon})\right\|_2^2\right].$$

若 planner 同时保留语言建模能力，总目标可抽象为

$$\mathcal{L} = \lambda_{\text{NTP}}\mathcal{L}_{\text{NTP}} + \lambda_{\text{plan}}\mathcal{L}_{\text{plan}} + \lambda_{\text{render}}\mathcal{L}_{\text{render}}.$$

其中 $\mathcal{L}_{\text{NTP}}$ 是 next-token prediction 或指令建模损失。

连续 ViT 语义空间的作用

与 VAE latent 相比，ViT 高层表示更强调对象、动作和场景，弱化精确像素；与自然语言相比，它保留更丰富的视觉布局和时空模式。因此它适合作为 planner 与 renderer 的中间接口。

统一生成与编辑

对 T2V，source 条件为空；对参考生成或编辑，planner 读取 source 语义，renderer 也可读取 source VAE feature。于是可统一表示为

$$(\mathbf{y}, \mathbf{x}_{\text{source}}) \xrightarrow{P_\omega} \mathbf{s}_{\text{target}} \xrightarrow{R_\theta} \mathbf{x}_{\text{target}}.$$

高层 source feature 保持身份/语义，低层 source VAE feature 保持纹理、布局或未编辑区域。

Segment-Aware 3D RoPE（SA-3D RoPE）

Bernini 在统一序列中可同时放入多张参考图、源视频和目标视频。不同视觉片段中的 token 可能拥有相同的局部坐标 (t, h, w)；仅使用普通 3D RoPE 时，注意力难以区分“参考图左上角”和“目标视频左上角”。SA-3D RoPE 为每个视觉 segment 再引入片段索引 g，可抽象写为

$$R_{\text{SA}}(g, t, h, w) = R_g(g)\, R_t(t)\, R_h(h)\, R_w(w),$$

即在原有时空相位上叠加 segment-dependent phase。这样既保留片段内部的相对时空几何，又能区分不同来源和目标片段。

这与“planner 网格到 renderer 网格的分辨率对齐”是两个问题。后者通常由投影层、cross-attention、插值或归一化坐标处理；SA-3D RoPE 主要解决多视觉片段的身份消歧。

8.6 Planner 的三种训练策略

独立预训练

先固定视觉语义编码器，训练 planner 预测真实语义；再训练 renderer 条件于真实或预测语义。

优点：模块可诊断；缺点：训练-推理分布差异，即 renderer 训练看真计划、推理看有误差的计划。

Teacher forcing + 噪声计划

训练 renderer 时混合：

$$\tilde{\mathbf{s}} = \rho \mathbf{s}_\star + (1-\rho)\hat{\mathbf{s}}_\star + \boldsymbol{\eta},$$

让其适应 planner 误差。

联合或轻量协同训练

允许 renderer loss 的部分梯度影响 planner，或周期性更新 planner。优点是接口适配；风险是 renderer 追求像素损失，破坏 planner 已有语言/语义能力。

8.7 计划的粒度

全局计划

一个或少数 token 描述整体内容。成本低，但难以表达事件顺序和局部轨迹。

帧级/时间段计划

$$\mathbf{s} = [\mathbf{s}_1, \dots, \mathbf{s}_K],$$

每个 token 对应一帧或一个时间段。适合动作变化和长视频，但序列更长。

时空网格计划

$$\mathbf{s} \in \mathbb{R}^{T_s \times H_s \times W_s \times d_s}.$$

可表达布局和局部语义，是最强但最昂贵的形式。renderer 可通过 cross-attention 从高层语义网格查询。

分镜/镜头层级计划

先预测 shot 数和镜头级语义，再对每个 shot 预测局部计划：

$$p(\mathbf{s}) = p(K) \prod_{k=1}^{K} p(\mathbf{s}_k \mid \mathbf{s}_{<k}, \mathbf{y}).$$

这更接近电影生成，但会引入镜头边界、跨镜头身份和全局叙事一致性问题。

8.8 规划器的误差传播与可诊断性

可把最终误差粗略写为

$$\|\hat{\mathbf{z}} - \mathbf{z}_\star\| \le L_R \|\hat{\mathbf{s}} - \mathbf{s}_\star\| + \epsilon_R,$$

其中 L_R 是 renderer 对计划扰动的局部 Lipschitz 常数，ϵ_R 是给定真计划时的渲染误差。于是有两条研究路线：

1. 降低 planner error；
2. 让 renderer 对合理计划误差更鲁棒，即降低有效 L_R。

诊断实验

- **Oracle-plan**：renderer 使用真实视频的 semantic embedding；
- **Predicted-plan**：使用 planner 输出；
- **No-plan**：移除计划，只用文本；
- **Shuffled-plan**：打乱时间或空间位置；
- **Low-rank-plan**：压缩计划，测试信息瓶颈；
- **Plan corruption**：注入噪声，画质量-扰动曲线。

这组实验能分离“planner 没规划好”和“renderer 没利用好计划”。

8.9 Classifier-Free Guidance 的条件解释

在 score 形式中，由 Bayes 公式：

$$\nabla_{\mathbf{z}} \log p(\mathbf{z} \mid \mathbf{c}) = \nabla_{\mathbf{z}} \log p(\mathbf{z}) + \nabla_{\mathbf{z}} \log p(\mathbf{c} \mid \mathbf{z}) + \text{const}.$$

条件与无条件 score 的差近似提供条件似然梯度：

$$\mathbf{s}_{\text{cond}} - \mathbf{s}_{\text{uncond}} \approx \nabla_{\mathbf{z}} \log p(\mathbf{c} \mid \mathbf{z}).$$

CFG 用

$$\hat{\mathbf{s}} = \mathbf{s}_{\text{uncond}} + w(\mathbf{s}_{\text{cond}} - \mathbf{s}_{\text{uncond}})$$

放大条件方向。对 flow velocity 也使用同样线性组合，但其严格概率解释取决于所用参数化和概率路径。

多条件 CFG

有文本、参考图和计划时，可写成

$$\hat{\mathbf{v}} = \mathbf{v}_\emptyset + w_y(\mathbf{v}_y - \mathbf{v}_\emptyset) + w_r(\mathbf{v}_{y,r} - \mathbf{v}_y) + w_s(\mathbf{v}_{y,r,s} - \mathbf{v}_{y,r}).$$

这允许分别控制文本、参考和计划强度，但需多次前向，且各方向并不正交。实际模型常通过训练时随机 drop 不同条件，获得可组合分支。

8.10 Guidance 的典型副作用

- w 太小：文本被忽略、构图随机；
- w 太大：饱和、过锐、运动僵硬、重复纹理；
- 长提示中少数高权重词压制其他约束；
- 参考 guidance 过强：视频像“动起来的照片”，动作幅度不足；
- planner guidance 过强：renderer 机械复制高层计划，细节缺乏多样性。

可采用随时间变化的 guidance：

$$w(\tau) = w_{\min} + (w_{\max} - w_{\min})h(\tau).$$

高噪声阶段更影响全局语义，低噪声阶段更影响纹理；合理 schedule 应通过消融确定，而不是默认越大越好。

8.11 提示词工程的结构化模板

在研究中，提示模板的目的不是“写得华丽”，而是控制变量。一个可评测模板可写为：

> [主体与数量] + [稳定属性] + [动作与方向] + [交互对象] + [场景] + [时间顺序] + [相机运动] + [景别/焦段] + [光照与风格] + [速度/时长]

例如：

> Two identical yellow toy cars start side by side. The left car moves forward first; one second later the right car turns clockwise. Static overhead camera, matte gray floor, uniform studio lighting, five-second continuous shot.

该提示明确了数量、身份、顺序、方向、相机和连续镜头，适合作为组合性/时间性测试。模糊的“two cars moving cinematically”无法定位失败原因。

8.12 Planner-Renderer 推理伪代码

```
# Stage 1: masked semantic planning in continuous ViT space
text_tokens = mllm.encode_text(prompt)
source_sem = sem_encoder(source_video) if source_video is not None else None
plan = full_mask_tokens(semantic_plan_shape, device=device)

for refine_step, next_mask_ratio in enumerate(mask_schedule):
    # The MLLM reasons over text, source segments and the partially filled target.
    hidden = mllm(
        text_tokens=text_tokens,
        source_semantics=source_sem,
        target_plan_tokens=plan,
        target_mask=is_masked(plan),
    )

    masked_idx = is_masked(plan)
    # Each masked position is decoded to a continuous ViT embedding by
    # a small flow-matching decoder conditioned on its MLLM hidden state.
    proposals, confidence = vit_embedding_decoder.sample_flow(
        condition=hidden[masked_idx],
        num_steps=planner_flow_steps,
    )
    # `confidence` is indexed within the masked subset. Map local proposal
    # indices back to their global spatiotemporal token positions.
    chosen_global, chosen_local = select_masked_positions(
        masked_idx, confidence, next_mask_ratio
    )
    plan[chosen_global] = proposals[chosen_local]

semantic_plan = plan

# Stage 2: flow-matching renderer in VAE latent space
source_latent = vae.encode(source_video) if source_video is not None else None
z = torch.randn(target_latent_shape, device=device)
for tau, tau_next in renderer_time_grid:
```

```
    v_cond = renderer(z, tau, text_tokens, semantic_plan, source_latent)
    v_null = renderer(z, tau, null_text, null_plan, source_latent)
    v = v_null + guidance_scale * (v_cond - v_null)
    z = ode_step(z, v, tau, tau_next)

video = vae.decode(z)
```

8.13 Planner 研究的关键问题

1. 计划空间是否包含时间和空间结构，而不只是全局 embedding?
2. planner 是语言模型、视觉生成模型，还是二者的统一多模态模型?
3. 计划是否可解释、可编辑、可缓存?
4. 训练时用真实计划还是预测计划? 如何消除 exposure gap?
5. renderer 在多大程度上依赖计划? 是否出现 condition collapse?
6. 对长视频，计划是一次生成还是滚动更新? 如何维护全局状态?
7. 计划错误能否由 verifier 或 reward model 发现并重采样?
8. 规划带来的质量提升是否抵消额外 MLLM 推理成本?

练习与自检

1. 写出带离散分镜变量 K 和连续语义计划 S 的分层生成分布。2. 设计 oracle-plan / predicted-plan / no-plan 三组实验，说明可分别估计哪些误差。3. 解释为何 prompt rewrite 可以提高语义覆盖，却不保证时间一致性。4. 对“红球先穿过蓝环，随后绿立方体落下”设计原子约束和自动问答式评测。

9. 视频生成评价体系：没有一个标量可以代表“好视频”

视频生成评价比训练损失更难。训练时有明确的噪声或速度 target；推理时却没有唯一正确视频。一个结果可能画质很好但与提示无关，也可能语义正确但运动僵硬。可靠评价必须分解目标，并明确每个指标的输入、特征提取器、统计估计和盲区。

9.1 六类评价问题

对生成视频 $\hat{\mathbf{x}}$、提示 $\mathbf{y}$ 和可选真实/源视频，至少区分：

1. **分布真实性与多样性**：生成集合是否像真实视频集合；
2. **文本-视频一致性**：内容是否满足提示中的原子约束；
3. **单帧视觉质量**：清晰度、美学、结构和伪影；
4. **时间与运动质量**：是否闪烁、漂移、卡顿、运动不足或不连续；
5. **物理/世界知识**：交互、守恒、因果、材料和常识是否合理；
6. **条件保持性**：I2V/V2V/编辑中，身份、背景、结构和未编辑区域是否保持。

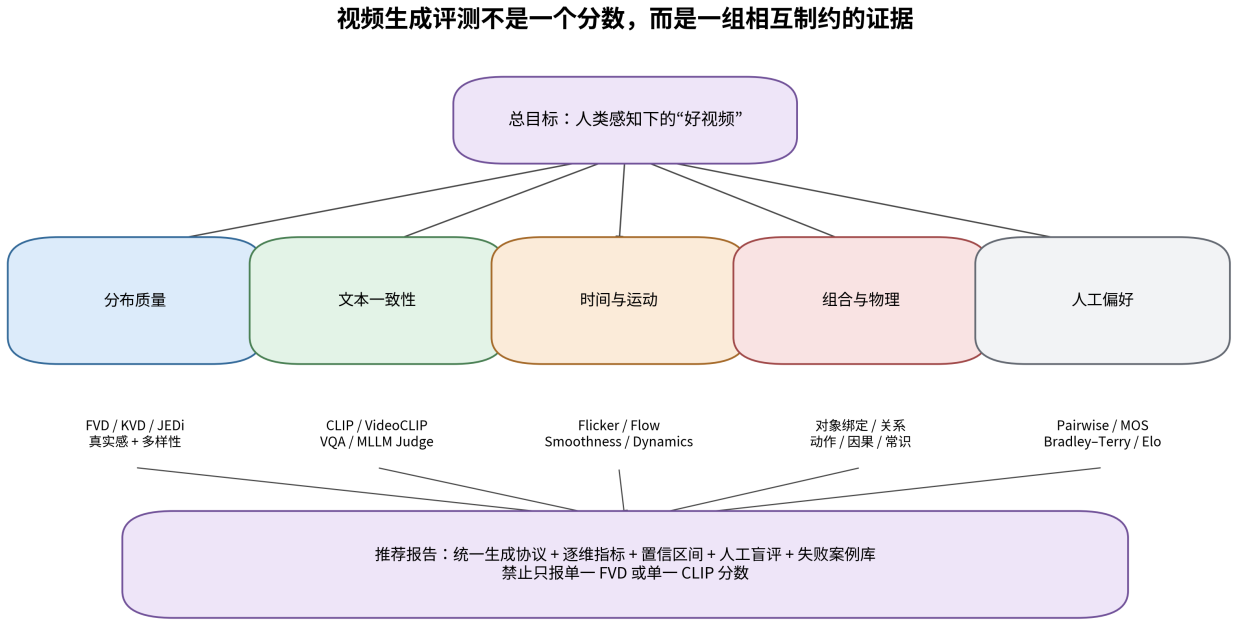

图 7: 视频生成评价指标的分层地图

核心结论

FVD 主要是集合级分布距离；CLIPScore 主要是粗粒度语义对齐；IS 主要反映分类器认为样本是否清晰且类别分布是否多样。三者都不能单独证明动作正确、时间稳定或符合物理规律。

9.2 评价对象与统计记号

设真实视频集合

$$\mathcal{R} = \{\mathbf{x}_i^{(r)}\}_{i=1}^{n_r},$$

生成集合

$$\mathcal{G} = \{\mathbf{x}_j^{(g)}\}_{j=1}^{n_g}.$$

若有 P 个提示、每提示 K 个随机种子，写为

$$\mathbf{x}_{p,k}^{(g)} \sim p_\theta(\mathbf{x} \mid \mathbf{y}_p), \quad p = 1, \dots, P, \quad k = 1, \dots, K.$$

特征提取器记为

$$\phi(\mathbf{x}) \in \mathbb{R}^d.$$

它可以是图像分类网络、I3D、视频自监督模型、CLIP/VideoCLIP 或 MLLM。评价结果不仅由生成模型决定，也由 ϕ、视频采样方式和预处理决定。

9.3 人类评价：最终标准，但不是天然无偏

绝对评分

让标注者按 1–5 或 0–100 分评价:

- 文本一致性;
- 画质;
- 时间连贯;
- 动作自然;
- 物理合理;
- 总体偏好。

优点是可得到多维解释；缺点是不同标注者尺度不一致，且“总体质量”容易混入品牌、风格和审美偏好。

成对比较

给同一提示下模型 A/B 的视频，询问哪个更好。成对比较通常比绝对分更稳定，因为人更擅长相对判断。

对模型 i, j，Bradley-Terry 模型写为

$$P(i \succ j) = \frac{\exp(s_i)}{\exp(s_i) + \exp(s_j)},$$

其中 s_i 是潜在质量分。可由所有 pairwise wins 最大似然估计排行榜和不确定性。

人评协议的最低要求

- 同一提示、相同分辨率/时长/播放速度;
- 模型名匿名，左右位置随机;
- 不允许视频自动循环次数不同;
- 分开问语义、运动、画质，最后才问总体偏好;
- 每对至少多个独立标注者;
- 报告标注人数、有效样本、重复一致性和置信区间;
- 对长视频允许完整播放，不能只看首帧或缩略图;
- 明确是否允许“平局/都差/无法判断”。

标注者一致性

可报告 Fleiss’ kappa、Krippendorff’ s alpha 或 pairwise agreement。低一致性不一定表示评价无效，也可能说明维度定义太模糊或样本差异太小。

9.4 Inception Score（IS）

给定预训练分类器输出 $p(y \mid \mathbf{x})$，边缘类别分布为

$$p(y) = \mathbb{E}_{\mathbf{x}\sim p_g} p(y \mid \mathbf{x}).$$

Inception Score 定义为

$$\boxed{\mathrm{IS} = \exp\left(\mathbb{E}_{\mathbf{x}\sim p_g} D_{\mathrm{KL}}(p(y \mid \mathbf{x}) \| p(y))\right).}$$

两个相反方向的熵

利用

$$\mathbb{E}_x D_{\mathrm{KL}}(p(y \mid x) \| p(y)) = H[p(y)] - \mathbb{E}_x H[p(y \mid x)],$$

高 IS 要求：

- 单个样本的分类器预测熵低，即“看起来像明确类别”；
- 全部样本的边缘类别熵高，即“覆盖多个类别”。

IS 的局限

1. 不需要真实数据，因此不能检测与目标数据分布的偏差；
2. 只关心分类器标签空间，不关心背景、动作和文本提示；
3. 生成相同的高置信类别原型可能得到不错分数；
4. 用逐帧 Inception 计算时几乎不测时间质量；
5. 不同实现的帧采样、分类器和 split 数不同时不能直接比较。

在现代开放域 T2V 中，IS 适合作为历史参考，而不是主结论。

9.5 Fréchet Inception Distance（FID）

提取真实和生成样本的特征，近似为高斯：

$$\phi(\mathcal{R}) \sim \mathcal{N}(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r), \qquad \phi(\mathcal{G}) \sim \mathcal{N}(\boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g).$$

FID 是两个高斯的平方 2-Wasserstein 距离：

$$\boxed{\mathrm{FID} = \|\boldsymbol{\mu}_r - \boldsymbol{\mu}_g\|_2^2 + \mathrm{Tr}\big(\boldsymbol{\Sigma}_r + \boldsymbol{\Sigma}_g - 2(\boldsymbol{\Sigma}_r^{1/2}\boldsymbol{\Sigma}_g\boldsymbol{\Sigma}_r^{1/2})^{1/2}\big).}$$

越低越好。第一项比较均值，第二项比较协方差。

视频中如何使用 FID

- **Frame FID**：从视频抽帧，当作图像集合；测单帧分布，不测时间；
- **First-frame FID**：只比较首帧；适合某些 I2V 分析；
- **Per-frame-position FID**：分别比较第 i 帧，诊断后期质量衰减；
- **Latent/semantic FID**：更换特征提取器后本质仍是 Fréchet 距离，但不应继续含糊称为同一 FID。

有限样本偏差

FID 的样本估计有偏，样本数越少通常越不稳定。比较模型时应使用相同 n_g, n_r；最好画出 score 随样本数的收敛曲线或用 bootstrap 给区间。

9.6 Fréchet Video Distance（FVD）

FVD 把整段 clip 输入视频特征网络（经典实现常用 I3D），得到

$$\phi_v(\mathbf{x}) \in \mathbb{R}^{d_v}.$$

然后与 FID 使用同一 Fréchet 公式：

$$\boxed{\mathrm{FVD} = \|\boldsymbol{\mu}_r - \boldsymbol{\mu}_g\|_2^2 + \mathrm{Tr}\big(\boldsymbol{\Sigma}_r + \boldsymbol{\Sigma}_g - 2(\boldsymbol{\Sigma}_r^{1/2}\boldsymbol{\Sigma}_g\boldsymbol{\Sigma}_r^{1/2})^{1/2}\big).}$$

区别在于 $\boldsymbol{\mu}, \boldsymbol{\Sigma}$ 来自视频 clip feature，而不是单帧 Inception feature。

FVD 测到了什么

如果视频 backbone 对动作和时间模式敏感，FVD 能同时反映：

- 外观分布；
- 动作类别和粗粒度运动；
- 视频级时空统计；
- 多样性和模式覆盖。

FVD 没有测什么

- 不显式比较每个提示是否满足；
- 不保证物理正确；
- 可能对局部闪烁、手指畸变或文字错误不敏感；
- backbone 的训练域可能偏向动作类别；
- 高质量但与参考数据域不同的风格可能被惩罚。

实践中的敏感变量

1. clip 帧数和抽样 fps；
2. resize/crop 方式；
3. 是否归一化到相同值域；
4. I3D/其他 backbone 的具体 checkpoint 和层；
5. 真实样本数与生成样本数；
6. 一个长视频切成几个 clip；
7. 同一提示多 seed 是否被当作独立样本。

为什么不能只报 FVD

FVD 建立在“特征近似高斯”和“所用 backbone 对重要失真敏感”两个强假设上。后续研究表明，传统 I3D 特征可能对某些时间扰动不够敏感，且稳定估计需要较多样本。因此，FVD 应与语义、细粒度时间和人评共同报告。

9.7 KVD、MMD 与 JEDi 类距离

最大均值差异（Maximum Mean Discrepancy, MMD）比较两个分布在核 Hilbert 空间的均值：

$$\mathrm{MMD}^2(P,Q) = \mathbb{E}_{x,x'\sim P}k(x,x') + \mathbb{E}_{y,y'\sim Q}k(y,y') - 2\mathbb{E}_{x\sim P,y\sim Q}k(x,y).$$

对视频特征使用多项式核或 RBF 核，可得到 Kernel Video Distance（KVD）类指标。它不需要把特征分布近似为高斯，但结果高度依赖 kernel 和 bandwidth。

JEDi 一类方法使用视频自监督/预测式表示（例如 JEPA embedding）和 MMD，目标是让特征对时间失真更敏感，并降低 FVD 的样本复杂度。使用此类新指标时，应同时报告 backbone、kernel、clip 预处理和与人评的验证，而不是只引用一个新名字。

9.8 CLIP-based 文本-视频对齐

CLIP 产生归一化文本和图像 embedding：

$$\mathbf{e}_y = \frac{f_T(\mathbf{y})}{\|f_T(\mathbf{y})\|_2}, \qquad \mathbf{e}_i = \frac{f_I(\mathbf{x}_i)}{\|f_I(\mathbf{x}_i)\|_2}.$$

一种简单视频 CLIPScore 为

$$\boxed{S_{\mathrm{CLIP}} = \frac{1}{T}\sum_{i=0}^{T-1}\max(0,\mathbf{e}_y^\top\mathbf{e}_i).}$$

也可用均匀/关键帧采样、最大值、时间加权，或先聚合视频特征再做余弦相似度。

它擅长的内容

- 主体、场景、显著属性；
- 大致风格和图文相关性；
- 快速、可扩展地筛选大量结果。

它不擅长的内容

- “先 A 后 B” 的时间顺序；
- 细粒度动作方向和速度；
- 数量、否定和复杂属性绑定；
- 相机运动与主体运动区分；
- 逐帧身份漂移；
- 物理和因果关系。

逐帧平均还会让静态但语义正确的视频取得高分。因此 CLIPScore 应配合 dynamic degree、动作对齐和时间指标。

9.9 Video-text embedding 指标

用视频-文本对比学习模型直接编码 clip：

$$S_{\mathrm{VT}} = \cos(f_V(\mathbf{x}), f_T(\mathbf{y})).$$

相较逐帧 CLIP，它可能更敏感于动作和时间，但仍受训练数据 caption 偏差影响。不同 video-text backbone 的分数不可直接横向比较。

原子语义分解

把提示拆为 J 个原子命题 a_j：

$$\mathbf{y} \to \{a_1, \dots, a_J\}.$$

每个原子独立评估：

$$S_{\mathrm{atomic}} = \frac{\sum_{j=1}^{J} w_j s_j}{\sum_{j=1}^{J} w_j}.$$

原子可以是：

- object presence;
- attribute binding;
- count;
- spatial relation;
- action;
- motion direction;
- temporal order;
- camera constraint;
- negation。

这种分解比一个全局余弦分数更可诊断。

9.10 MLLM/VLM-as-a-Judge

一种常见流程：

1. LLM 将提示解析成 scene graph 或原子问题；
2. 视频 MLLM 读取整段视频或采样帧；
3. 对每个问题输出 yes/no、证据时间段和置信度；
4. 聚合为维度分数。

例如提示“红球先穿过蓝环，随后绿方块落下”，问题可为：

- 是否出现一个红球?
- 是否出现蓝色环?
- 红球是否穿过蓝环?
- 绿方块是否在穿环事件之后下降?
- 相机是否保持静止?

优点

- 可处理开放词汇和复杂关系；
- 能提供错误解释；
- 适合低频、难以训练专门检测器的概念。

风险

- judge 可能看不懂快速动作或细小对象；
- 抽帧可能丢失事件顺序；
- 语言先验会让 judge“脑补”不存在的内容；
- 商业 API 更新会造成基准漂移；
- 被评模型与 judge 共享训练数据或模型家族，可能有偏；
- prompt injection、水印和屏幕文字可能干扰 judge。

因此必须用人工标注验证相关性，并固定 judge 版本、提示模板、帧采样和随机性。

ETVA 型问答评估

ETVA 等方法通过细粒度问题生成与视频问答来评估对齐，强调把复杂提示分解为可验证命题。其价值主要在可诊断性，而不是宣称一个 MLLM 分数就是客观真值。

9.11 T2VScore 类综合指标

T2VScore 将评价拆为:

- **T2VScore-A**: Text-Video Alignment;
- **T2VScore-Q**: Video Quality，常由多个质量专家融合。

可抽象为

$$S_{\mathrm{T2V}} = F(S_{\mathrm{align}}, S_{\mathrm{spatial}}, S_{\mathrm{temporal}}, S_{\mathrm{aesthetic}}).$$

融合权重若通过人评拟合，可提高总体相关性，但会继承标注集的人群、提示域和模型分布。论文中应同时报告子分数，不能只报融合总分。

9.12 单帧质量指标

无参考图像质量（NR-IQA）

利用美学/质量模型直接预测：清晰度、曝光、构图、压缩伪影等。常见问题是对生成特有畸变（多手指、局部融化）不一定敏感。

人脸、手部与文字专项

可使用人脸检测置信度、关键点完整率、OCR 字符准确率和手部姿态检测。但这些检测器自身有域偏差；检测失败不等价于生成失败，反之亦然。

Aesthetic score

美学模型通常从人类偏好数据学习。它适合筛选“视觉悦目”，但可能偏好浅景深、电影色调、中心构图等常见风格，不应替代真实性或提示一致性。

9.13 全参考重建指标：PSNR、SSIM、LPIPS

这些指标要求生成结果与目标在像素或内容上对齐，适用于 Video VAE 重建、视频超分、确定性编辑或带 ground truth 的预测任务；不适合开放域 T2V 的一对一评价。

MSE 与 PSNR

$$\mathrm{MSE} = \frac{1}{THWC}\|\hat{\mathbf{x}} - \mathbf{x}\|_2^2,$$

若像素最大值为 L：

$$\mathrm{PSNR} = 10\log_{10}\frac{L^2}{\mathrm{MSE}}.$$

高 PSNR 表示像素接近，但对感知结构不敏感。

SSIM

局部窗口中：

$$\mathrm{SSIM}(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}.$$

它比较亮度、对比度和结构，但仍不完全对应感知质量。

LPIPS

在预训练网络多层特征中比较归一化距离：

$$\mathrm{LPIPS}(x,y) = \sum_l \frac{1}{H_l W_l}\sum_{h,w}\left\|\mathbf{w}_l \odot (\hat{\phi}_l(x)_{h,w} - \hat{\phi}_l(y)_{h,w})\right\|_2^2.$$

LPIPS 更接近感知差异，但对时间一致性仍需逐帧/运动补偿扩展。

9.14 时间一致性与运动指标

相邻帧特征一致性

$$S_{\mathrm{adj}} = \frac{1}{T-1}\sum_{i=0}^{T-2}\cos(\phi_I(\mathbf{x}_i), \phi_I(\mathbf{x}_{i+1})).$$

高分可能表示主体稳定，也可能只是视频静止，因此必须与 motion magnitude 联合解释。

Optical-flow warping error

估计前向光流 $\mathbf{f}_i$，把第 i 帧 warp 到第 $i+1$ 帧：

$$E_{\mathrm{warp}} = \frac{1}{T-1}\sum_i \left\|\mathbf{x}_{i+1} - \mathcal{W}(\mathbf{x}_i, \mathbf{f}_i)\right\|_1.$$

应使用遮挡 mask：新显露区域本来就无法由前一帧 warp 得到。光流估计器若在生成伪影上失效，指标也会失真。

Temporal LPIPS

可比较运动补偿后的感知差异：

$$\mathrm{tLPIPS} = \frac{1}{T-1}\sum_i \mathrm{LPIPS}(\mathbf{x}_{i+1}, \mathcal{W}(\mathbf{x}_i, \mathbf{f}_i)).$$

Flicker / temporal high-frequency energy

对静态区域或运动补偿后残差

$$\mathbf{r}_i = \mathbf{x}_{i+1} - \mathcal{W}(\mathbf{x}_i, \mathbf{f}_i),$$

测量时间高频能量，可发现亮度闪烁、纹理 boiling 和随机细节变化。

Motion smoothness

设光流或关键点轨迹为 $\mathbf{p}_i$，速度与加速度：

$$\mathbf{v}_i = \mathbf{p}_{i+1} - \mathbf{p}_i, \qquad \mathbf{a}_i = \mathbf{v}_{i+1} - \mathbf{v}_i.$$

可用 jerk

$$\mathbf{j}_i = \mathbf{a}_{i+1} - \mathbf{a}_i$$

的统计量衡量突变。但真实碰撞、切镜和快速动作本来就可能有大加速度，必须按场景解释。

Dynamic degree

估计运动幅度，例如

$$D_{\text{motion}} = \frac{1}{(T-1)HW} \sum_{i,h,w} \mathbb{I}\left(\|\mathbf{f}_i(h,w)\|_2 > \delta\right).$$

它回答“动得多不多”，不回答“动得对不对”。一个剧烈闪烁的视频可能 dynamic degree 很高。

Subject/background consistency

检测或分割主体，计算跨帧 identity/appearance feature 相似度；背景可在去除前景和相机运动后计算。需要把相机造成的视角变化与身份漂移区分开。

9.15 相机运动与对象运动的分解评价

像素光流可分解为

$$\mathbf{f}_i \approx \mathbf{f}_i^{\text{camera}} + \mathbf{f}_i^{\text{object}} + \mathbf{f}_i^{\text{nonrigid}}.$$

可通过特征匹配和单应/基础矩阵估计全局相机运动，再在残差中测对象运动。对于“静态相机”提示，应检查全局变换是否接近单位；对于“dolly in / pan left / orbit”提示，应检查估计轨迹方向。

这类指标比纯 optical-flow magnitude 更接近摄影语言，但在非平面场景、强视差和大遮挡下估计困难。

9.16 编辑与条件保持指标

对源视频 $\mathbf{x}_s$、编辑结果 $\mathbf{x}_e$：

未编辑区域保持

有 mask $\mathbf{m}$ 时：

$$E_{\text{preserve}} = \frac{\|(1-\mathbf{m}) \odot (\mathbf{x}_e - \mathbf{x}_s)\|_1}{\|(1-\mathbf{m})\|_1}.$$

身份保持

对人脸/主体 embedding：

$$S_{\text{id}} = \frac{1}{T} \sum_i \cos(f_{\text{id}}(x_{s,i}), f_{\text{id}}(x_{e,i})).$$

运动保持

比较 source/result optical flow 或轨迹：

$$E_{\text{motion-preserve}} = \frac{1}{T-1}\sum_i \|\mathbf{f}_i^{(s)} - \mathbf{f}_i^{(e)}\|_1.$$

编辑成功率

用 CLIP/MLLM/检测器测结果是否满足编辑指令。保持与编辑通常是 Pareto trade-off，应用报告二维曲线，而不是只优化一个加权总分。

9.17 VBench 的 16 个维度

VBench 将评价分成视频质量与条件一致性多个维度。常用 16 项如下。

维度	主要问题
Subject Consistency	主体身份和外观是否跨帧稳定
Background Consistency	背景结构、纹理和场景是否稳定
Temporal Flickering	是否出现非运动导致的高频闪烁
Motion Smoothness	运动是否连续、无异常跳变
Dynamic Degree	视频是否具有足够可见运动
Aesthetic Quality	构图、色彩、视觉审美
Imaging Quality	清晰度、曝光、伪影等技术画质
Object Class	指定对象是否出现
Multiple Objects	多对象是否同时正确出现
Human Action	人类动作是否符合提示
Color	指定颜色是否正确绑定
Spatial Relationship	左右、上下、前后等关系是否正确
Scene	场景/环境是否匹配
Temporal Style	慢动作、延时、节奏等时间风格
Appearance Style	油画、动画、电影等外观风格
Overall Consistency	整体文本-视频语义一致性

如何正确使用 VBench

- 报告完整维度，不只报 total score；
- 固定 prompt suite、生成时长、分辨率和 seed 数；
- 查明每个维度具体 evaluator 和版本；
- dynamic degree 与 consistency 要一起看；
- 某模型若针对 benchmark prompt 调参，需说明；
- total score 的权重是人为选择，不代表所有应用的效用函数。

VBench++

VBench++ 将评价扩展到更多任务和能力，包括 I2V、长视频及可信/安全相关维度。使用扩展基准时应明确具体 track，不能把不同 track 的总分混为一个排行。

9.18 代表性专项 benchmark

EvalCrafter

以多样化提示和多组客观指标，从 visual quality、content、motion 和 text-video alignment 等方面评价，并用人评拟合综合分。它提醒我们：大型开放域模型需要多维评价，而非只靠 FVD/IS。

T2V-CompBench

聚焦组合性，覆盖：

- consistent attribute binding;
- dynamic attribute binding;

- spatial relationships;
- motion binding;
- action binding;
- object interactions;
- generative numeracy。

其指标结合 MLLM、检测和 tracking。研究复杂提示遵循能力时，这类分类结果比一个 CLIPScore 更有价值。

VideoPhy / VideoPhy-2

评价真实活动中的物理常识，涉及材料交互、动作和守恒规律。通常同时检查 caption adherence 与 physical commonsense；语义正确但物理错误不能算成功。

ChronoMagic-Bench

聚焦延时/形态变化视频，如生长、融化、天气和化学变化，评价 metamorphic amplitude 与 temporal coherence。它测试普通短动作指标难以覆盖的长程状态变化。

T2VTextBench

专门评价视频中可读文字的准确性和跨帧一致性。OCR 正确率、字符稳定性和文字随物体运动的一致性是独立难题。

T2VWorldBench

聚焦世界知识、事实和因果，包括物理、自然、活动、文化等类别。其核心是把“看起来合理”与“事实上正确”区分开。

ETVA

把提示解析成原子问题，通过细粒度视频问答评估文本-视频对齐，适合分析漏对象、错误关系和事件顺序。

GenEval 的边界

GenEval 主要是文生图组合性 benchmark，关注对象、数量、颜色和空间位置。其思想可迁移到视频的逐帧组合性，但它本身不评价时间顺序、动作、运动平滑和跨帧一致性。视频论文若报告“GenEval”，必须说明是首帧、逐帧还是自定义扩展，避免让读者误认为它是完整 T2V 指标。

9.19 相关性：自动指标是否像人

给自动分数 a_i 和人评分 h_i:

Pearson 相关

$$\rho_P = \frac{\sum_i (a_i - \bar{a})(h_i - \bar{h})}{\sqrt{\sum_i (a_i - \bar{a})^2}\sqrt{\sum_i (h_i - \bar{h})^2}}.$$

测线性关系，对异常值敏感。

Spearman 相关

对分数取秩后计算 Pearson，测单调关系。评价模型排行时常更适合。

Kendall’ s tau

在没有 ties 的基本形式下，基于样本对的 concordant/discordant 关系:

$$\tau_K = \frac{N_{\text{concordant}} - N_{\text{discordant}}}{\binom{n}{2}}.$$

相关性不能替代绝对有效性

一个指标在已有模型范围内能排对顺序，不代表它能识别新型失败；模型迭代后可能出现 evaluator 未见过的伪影。因此需持续更新验证集并做 adversarial stress test。

9.20 统计显著性与置信区间

以提示为统计单位

若每个提示有多个 seed，先对 seed 平均：

$$\bar{s}_p = \frac{1}{K}\sum_{k=1}^{K} s_{p,k},$$

再对提示平均：

$$\bar{s} = \frac{1}{P}\sum_{p=1}^{P} \bar{s}_p.$$

把每个 seed 当完全独立样本会低估同一提示内的相关性。

配对比较

对同一提示的模型 A/B：

$$d_p = \bar{s}_p^{(A)} - \bar{s}_p^{(B)}.$$

对 $\{d_p\}$ 做 bootstrap 或配对检验，比独立样本检验更有统计功效。

Bootstrap

从提示集合有放回采样，重复计算均值/差值，取 2.5% 与 97.5% 分位数作为 95% 区间。若 prompts 按类别分层，应做 stratified bootstrap。

多重比较

同时比较很多模型/指标会提高假阳性。研究主结论应预先指定主要指标，或采用 Holm/Benjamini-Hochberg 等校正，并报告 effect size，而不只报 p 值。

9.21 多样性评价

Intra-prompt diversity

同一提示多个 seed 的特征距离：

$$D_{\text{intra}} = \frac{2}{K(K-1)}\sum_{k<l} \|\phi(\mathbf{x}_{p,k}) - \phi(\mathbf{x}_{p,l})\|_2.$$

但高距离可能来自不稳定和错误，而非有意义多样性。应只在满足提示的样本中测 conditional diversity。

Coverage 与 precision

在特征流形中：

- precision：生成样本有多少落在真实数据邻域；
- recall/coverage：真实数据模式有多少被生成覆盖。

高 precision 低 recall 表示保守、模式少；低 precision 高 recall 可能表示多样但不真实。

多样性-一致性的 Pareto 曲线

改变 CFG 或 temperature，画

$$(S_{\text{alignment}}(w), D_{\text{diversity}}(w))$$

而非只选择一个有利的 guidance scale。不同应用可在 Pareto front 上选择工作点。

9.22 一份可信的 T2V 评价协议

Prompt 集

至少分层覆盖：

- 单主体与多主体；
- 属性绑定、数量、空间关系；
- 人体动作、物体运动、交互；
- 时间顺序和状态变化；
- 相机静止与指定运镜；
- 物理/世界知识；
- 文字、风格、长尾实体；
- 易、中、难三级。

生成设置

固定并报告：

- 模型 checkpoint；
- prompt rewrite/planner 是否开启；
- resolution、frames、fps、duration；
- sampler、NFE、time shift；
- CFG、negative prompt；
- seed 数；
- VAE decode 和插帧/超分后处理；
- 每视频总延迟与峰值显存。

指标最小组合

一个通用实验至少应有：

1. 分布质量：FVD 或更合适的 video feature distance；
2. 文本对齐：video-text/atomic QA；
3. 时间质量：flicker、subject/background consistency、motion smoothness；
4. 运动幅度：dynamic degree；
5. 细分 benchmark：组合性或物理；
6. 人类成对偏好；
7. 效率：NFE、延迟、峰值显存和吞吐。

报告方式

- 主表放关键维度和 95% CI；
- 附录给完整 prompt-level 结果；
- 同时展示成功和失败案例；
- 避免只挑最好 seed；
- 公开评测代码、配置、prompt 与生成 metadata；
- 自动 judge 的原始回答应可审计。

9.23 指标选择决策表

研究问题	首选指标	必须补充
新生成骨干是否改善总体分布	FVD/KVD/JEDi 类	人评、语义、时间分项
文本遵循是否改善	原子 QA、T2V-CompBench、video-text score	画质与多样性
时间模块是否有效	flicker、flow-warp、subject consistency	dynamic degree，防止静态投机
物理能力是否提升	VideoPhy 类、人评规则检查	文本一致性，防止“没执行动作”
Video VAE 是否更好	PSNR/SSIM/LPIPS、重建 FVD、tLPIPS	压缩率、decode 成本
编辑模型是否更好	edit success + preservation + identity	人评、运动保持

研究问题	首选指标	必须补充
蒸馏/加速是否成功	质量-NFE/延迟 Pareto	同硬件、同分辨率、同 CFG
长视频是否更稳定	分段质量曲线、身份漂移、事件完成率	总时长、窗口重叠、内存成本

常见误区

一个指标一旦成为优化目标，就可能被模型“钻空子”。高 temporal consistency 可由静态视频获得；高 dynamic degree 可由抖动获得；高 CLIPScore 可由让主体占满画面获得；低 FVD 可由复制训练分布中的常见动作获得。多指标和人工审计不是形式要求，而是防止 Goodhart's law。

练习与自检

1. 构造两个模型：A 生成清晰静态图，B 生成稍模糊但动作正确的视频。预测 IS、CLIPScore、dynamic degree、motion smoothness 的相对结果。2. 证明 IS 的对数等于 $H[p(y)] - E_x H[p(y|x)]$。3. 说明为什么 FVD 的真实/生成样本数不一致会影响可比性。4. 为“两个玻璃球从斜坡滚下并碰撞，碰撞后速度改变但球不消失”设计自动与人工联合评价。5. 设计一个 prompt-level bootstrap，比较两个模型的 VBench subject consistency，给出伪代码。

10. 核心困难、失败模式与因果诊断

视频生成研究中最浪费时间的做法，是看到坏样本后只调 prompt、CFG 或随机种子。一个伪影可能来自数据、Video VAE、生成骨干、时间/位置编码、采样器、条件接口或后处理。有效诊断需要先定位模块，再解释机制。

10.1 从生成链路看误差来源

完整链路可写为

$$\mathbf{y}, \mathbf{r} \xrightarrow{\text{text/MLLM}} \mathbf{c}, \mathbf{s} \xrightarrow{\text{diffusion/flow DiT}} \hat{\mathbf{z}}_\star \xrightarrow{D_\psi} \hat{\mathbf{x}} \xrightarrow{\text{postprocess}} \tilde{\mathbf{x}}.$$

最终误差可粗略分解为

$$\mathcal{E}_{\text{total}} \approx \mathcal{E}_{\text{condition}} + \mathcal{E}_{\text{generation}} + \mathcal{E}_{\text{VAE}} + \mathcal{E}_{\text{sampling}} + \mathcal{E}_{\text{post}}.$$

这不是严格可加的独立误差，但提供诊断框架。

10.2 语义失败

对象遗漏与额外对象

现象：提示中的对象缺失，或模型生成未要求的对象。

常见原因：

- caption 数据常只列显著主体；
- 文本 token 在 cross-attention 中竞争；
- CFG 强化最显著词而压制次要约束；
- planner 或 prompt rewrite 自行补全常见场景；
- 多对象训练样本较少。

诊断：

- 用原子 object presence 检测；
- 逐层观察 cross-attention 或条件 ablation；
- 固定 seed，逐步增加对象，画成功率随对象数变化；
- 比较短提示、结构化提示和 planner 输出。

属性绑定错误

提示“红球在蓝盒子左侧”，模型可能生成蓝球和红盒子。这是模型知道所有概念，但没有维护实体-属性绑定。

可将语义图表示为

$$G = (V, E),$$

其中节点是实体，边是属性/关系。生成模型需保持

$$\text{bind}(o_i, a_i), \qquad \text{rel}(o_i, o_j, r_{ij}).$$

改善方向包括：更强 token-level condition、对象级计划、区域/轨迹监督、组合性数据和原子 reward。

数量错误

Transformer 的连续 attention 和互联网 caption 并不天然实现精确计数。对象重叠、出画和身份合并使视频计数更难。评价时应使用 tracking 后的 unique identity 数，而非逐帧 detection 数简单平均。

动作或方向错误

“向左跑”生成向右，可能来自：

- 数据增强中的水平翻转未同步修改 caption；

- 文本只描述动作类别，不描述方向；
- 相机运动与对象运动混淆；
- 空间位置编码弱或坐标系不统一。

事件顺序错误

对于事件 A、B，提示要求

$$t(A) < t(B).$$

模型可能同时发生、顺序颠倒或只完成一个。原因是 clip 太短、caption 无时间戳、全局文本 embedding 缺少事件对齐。可采用时间段计划、事件边界标注或按原子事件生成 verifier。

10.3 主体一致性与身份漂移

外观漂移

现象：人物脸、衣服、动物花纹、商品细节随帧变化。

机制：

- 模型每个时间位置都需恢复高频细节，而这些细节在高噪声阶段不确定；
- 时间 attention 范围不足；
- VAE 将身份细节压缩掉；
- 训练 clip 中主体本身被遮挡或切镜；
- 参考图条件只提供全局 embedding，缺少局部身份 token。

诊断：

1. 用 VAE 对真实视频重建；若已漂移，先修 VAE；
2. 比较 T2V 与同一首帧的 I2V；若 I2V 显著好，主要是外观锚点不足；
3. 计算人脸/主体 embedding 随时间曲线；
4. 把视频分为前、中、后段，检查漂移是否累积；
5. 增加 temporal attention 范围或 reference cross-attention，做对照。

拓扑不稳定

手指数目、眼镜框、车轮、杯柄等局部结构会出现/消失。它既可能来自单帧生成能力，也可能来自时间对应失败。可比较单帧随机抽样质量与连续帧拓扑变化来分离。

10.4 背景漂移、纹理 boiling 与闪烁

背景漂移

静态墙面纹理、窗户位置和地平线缓慢变化。常见原因是：

- 全局坐标与相机模型不足；
- 局部窗口 attention 无法维持大尺度地图；
- 数据中大量手持抖动；
- sampler 在低噪声阶段持续改变高频细节。

Texture boiling

物体整体稳定，但毛发、草地、砖墙等纹理像“沸腾”。这通常是每帧高频噪声恢复不一致。改进包括：

- 更强时间 VAE；
- 共享/相关噪声结构；
- motion-compensated temporal loss；
- 在低噪声阶段加强跨帧 feature coupling；
- 降低过强 CFG 或 sharpen 后处理。

亮度/色彩闪烁

可能由 VAE normalization、分块解码、bf16 数值、后处理 tone mapping 或每帧独立超分引起。应在 latent、VAE 原始输出、后处理输出三个层级分别测时间残差。

10.5 运动失败

Motion collapse

视频几乎静止，仅有眨眼、呼吸或相机微动。

常见原因：

- 静态/低运动训练数据占比过高；
- I2V 的首帧条件过强；
- 高 CFG 更偏向稳定语义而抑制运动多样性；
- 模型在有限容量下优先优化单帧质量；
- dynamic caption 不足；
- 训练时抽帧间隔太小，邻帧几乎相同。

诊断时同时报告 dynamic degree 和语义正确率。简单增加 optical-flow loss 可能导致无意义抖动。

运动过度或随机抖动

高 dynamic degree 不代表好。随机相机抖动、局部变形和纹理闪烁都能提高运动幅度。需要 motion smoothness、camera decomposition 和 object tracking 联合判断。

速度不一致

同一物体无原因忽快忽慢，常来自：

- fps metadata 缺失；
- 训练 clip 时间跨度混杂；
- 时间位置编码按帧号而非真实时间；
- 采样后插帧改变节奏。

接触与交互失败

手抓杯子、球撞墙、脚踩地等需要精确接触约束。模型常出现穿透、悬浮、接触后对象融化。单纯视觉相似度难以监督接触动力学，需要细粒度数据、3D/轨迹条件或物理 verifier。

10.6 相机-对象纠缠

对场景点 $\mathbf{X}$，像素位置由相机投影

$$\mathbf{u}_i \sim \mathbf{K}_i[\mathbf{R}_i \mid \mathbf{t}_i]\mathbf{X}_i.$$

像素变化同时依赖对象状态 $\mathbf{X}_i$ 和相机 $(\mathbf{R}_i, \mathbf{t}_i)$。只在像素空间学习时，这两种解释可互换。

典型失败：

- 提示“相机环绕雕像”，模型让雕像本身旋转；
- 提示“静态相机，汽车前进”，模型用背景后退模拟；
- zoom 被误成物体膨胀；
- pan 中背景局部变形而非刚性移动。

研究方向包括：显式相机 token、相机轨迹监督、3D-aware latent、深度/点轨迹条件，以及把 camera motion 与 object motion 分开评价。

10.7 物理与因果失败

对象持久性

对象被遮挡后应保持身份和数量。生成模型可能在遮挡期间“忘记”对象，重新出现时改变颜色或形状。

守恒规律

质量、动量、能量并不直接由视觉损失强制。常见错误：液体凭空增加、碰撞后物体消失、影子方向不一致。

材料属性

玻璃、布料、液体、烟雾和刚体具有不同动力学。互联网 caption 很少显式标注材料参数，模型可能只学到外观，不学到状态方程。

因果先后

事件 A 导致 B 应满足：

$$A \to B, \qquad t(A) < t(B),$$

并具有合理中间状态。模型可能仅生成两个相关视觉片段，却没有因果过渡。

为什么“更多数据”不自动解决

被动互联网视频主要提供观测相关性，而非受控干预。相机剪辑、不可见外力和 caption 缺失让因果变量不可辨识。增强物理能力可能需要：

- 仿真和真实数据混合；
- 物体/接触/轨迹标注；
- 3D 或状态空间建模；
- action-conditioned world model；
- 规则/能量约束或可学习 verifier；
- 生成后搜索和拒绝采样。

10.8 长视频的误差累积

固定窗口扩展

若模型一次生成 T_0 帧，长视频通常用窗口滚动：

$$\mathbf{x}_{kT_s:kT_s+T_0} \sim p_\theta(\mathbf{x} \mid \mathbf{x}_{<kT_s}, \mathbf{y}).$$

窗口 overlap 能平滑边界，但会出现：

- 语义漂移；
- 身份逐段改变；
- 场景布局累积误差；
- 重复动作或循环；
- overlap 区过度平滑；
- 每段重新采样导致风格变化。

长期状态

可显式维护状态

$$\mathbf{m}_{k+1} = F(\mathbf{m}_k, \mathbf{x}_k, \mathbf{y}),$$

其中 $\mathbf{m}$ 可包含角色表、场景地图、事件完成状态、相机状态和关键帧。MLLM planner 特别适合维护符号/语义状态，但仍需与像素 renderer 对齐。

多镜头叙事

单镜头时间一致性与多镜头故事一致性不同。硬切本来允许背景和视角突变，但角色身份、服装、道具和叙事状态应保持。评价必须先检测 shot boundary，再区分 shot 内连续性与 shot 间语义连续性。

10.9 数据问题：规模、质量与可学习信号

Caption 偏差

普通 caption 倾向描述名词，忽略：

- 动作阶段；
- 速度与方向；

- 相机；
- 对象进入/离开画面；
- 因果；
- 不可见状态。

因此高质量 recaption 通常要输出主体、动作、场景、镜头、风格和时间顺序，并与 clip 边界匹配。

切镜污染

一个 clip 内含硬切，模型可能学到无条件瞬移。若目标是单镜头生成，应严格过滤；若目标含多镜头，应显式提供 shot token 或边界条件。

重复与泄漏

近重复视频会：

- 夸大有效数据规模；
- 让 benchmark prompt/视频泄漏；
- 造成 memorization；
- 使 FVD 偏低。

需要帧哈希、视频 embedding、音频指纹和语义近重复联合去重。

质量过滤的反作用

过度美学过滤会缩窄分布，使模型只会电影化画面；过度运动过滤会移除静态场景；过度去水印可能损伤真实文字场景。应保留分层数据并按训练阶段/采样权重使用。

版权、隐私与安全

数据管线需要可追溯许可证、人物隐私和敏感内容审计。技术上还应测试：训练样本复现、名人身份生成、受版权风格模仿、色情/暴力和错误信息能力。

10.10 VAE、DiT 与采样器的定位实验

第一步：VAE oracle reconstruction

对真实视频直接

$$\mathbf{x} \xrightarrow{E_\phi} \mathbf{z}_\star \xrightarrow{D_\psi} \tilde{\mathbf{x}}.$$

若文字、手部、快速运动或颜色已经失真，生成模型无法超过 VAE 信息上限。

第二步：真实 latent 加小噪声再去噪

从低噪声 $\tau \ll 1$ 开始，检查模型能否恢复细节。失败指向低噪声训练权重、输出参数化或 VAE latent scaling。

第三步：文本条件 ablation

比较：full prompt、删除动作词、删除相机词、null condition。若动作词几乎不改变输出，说明条件接口或数据监督弱。

第四步：固定 latent noise

固定 seed，只改变 CFG、sampler、NFE 和 flow shift。这样能把随机内容差异与采样超参数影响分开。

第五步：中间状态 decode

将若干 $\mathbf{z}_\tau$ 映射为估计 $\hat{\mathbf{z}}_\star$ 再 decode，观察：

- 高噪声阶段：布局和对象何时确定；
- 中噪声阶段：动作和相机何时形成；
- 低噪声阶段：是否引入闪烁/过锐。

第六步：oracle condition

对 planner 模型使用真实 semantic plan；对 I2V 使用真实首帧；对 trajectory 模型使用真实轨迹。逐级替换为预测条件，量化接口误差。

10.11 失败模式-根因-修复矩阵

失败现象	首先检查	常见修复方向
单帧就模糊	VAE 重建、训练分辨率	改善 VAE、低噪声权重、高分辨率阶段
纹理跨帧闪烁	VAE 时间重建、低噪声采样	时序 VAE、motion-aware loss、降低过强 guidance
主体脸逐渐变化	reference condition、temporal attention	身份 token、长程注意力、reference cross-attention
几乎不动	数据运动分布、首帧强度、CFG	动态采样、动作 caption、条件 dropout/权重
无意义抖动	optical flow、后处理、窗口边界	motion smoothness、相机稳定、temporal overlap
左右方向反了	flip augmentation/caption	同步改写方向词、坐标条件
事件顺序错	clip/caption 时间标注	时间段计划、事件监督、MLLM verifier
物体穿透/消失	物理数据、遮挡状态	轨迹/深度/3D 条件、物理 reward
长视频重复	滚动窗口状态	显式 memory、全局计划、事件完成表
编辑改变了背景	source feature 和 mask 注入	多尺度 source anchoring、保持损失
推理步数减少后崩坏	轨迹曲率、蒸馏目标	更好 solver、consistency/distillation、reflow
分块 decode 有接缝	VAE tile overlap	加 overlap/window blending、统一 normalization

10.12 从第一性原理看尚未解决的挑战

表示瓶颈

视频需同时压缩外观、运动、3D 结构和长期状态。单一连续 latent 是否足够，仍是开放问题。

监督瓶颈

文本 caption 不等价于动作、相机、物理和因果标注。模型规模扩大不能创造不存在的监督信号。

计算瓶颈

token 数随时长和分辨率乘法增长，全局交互则平方增长。高压缩、稀疏计算和分层生成必须共同发展。

生成目标瓶颈

MSE 型去噪/速度损失是局部回归，最终人类偏好是高度结构化、非局部的。如何引入 reward 而不破坏多样性和稳定性，是关键问题。

评价瓶颈

自动 evaluator 本身不可靠；人评昂贵且主观。模型一旦优化公开指标，指标失效会加速。

世界建模瓶颈

“看起来像视频”与“能预测可干预世界”不同。真正 world model 需要动作、状态、可逆性、反事实和长期记忆，而不只是文本条件视觉生成。

练习与自检

给定一个症状：“720p T2V 中人物前 2 秒稳定，随后脸逐渐变化；VAE 重建真实视频正常，I2V 比 T2V 好很多，增加 NFE 无改善。”请按证据排序最可能根因，并设计三个最小消融实验。

11. 端到端训练与推理 Pipeline

本章把前述数学组件连接成可执行系统。具体模型会改变模块，但一个严谨项目应明确数据流、训练阶段、损失、并行、checkpoint 和评价元数据。

11.1 总体模块图

训练系统通常包含：

1. 视频数据管线；
2. caption/MLLM 数据管线；
3. Video VAE；
4. 文本编码器/MLLM planner；
5. Video DiT；
6. diffusion/flow scheduler；
7. 分布式优化和 checkpoint；
8. 离线生成与评价集群。

可将一批样本表示为

$$\mathcal{B} = \{(\mathbf{x}_b, \mathbf{y}_b, \mathbf{r}_b, \mathbf{m}_b, \boldsymbol{\eta}_b)\}_{b=1}^{B},$$

其中 $\boldsymbol{\eta}_b$ 存放 fps、时长、分辨率、宽高比、shot boundary、质量分等 metadata。

11.2 数据准备流水线

Step 1：采集与授权记录

为每条原始视频保存：来源、许可证、时间戳、内容哈希、原始分辨率/fps、音频状态和删除请求映射。

Step 2：解码与基础审计

- 解码成功率；
- 黑帧/纯色帧；
- 重复帧比例；
- 编码损坏；
- 真实时间戳单调性；
- 音视频时长差；
- 旋转 metadata。

Step 3：镜头切分

检测硬切、渐变和闪白。对单镜头训练，clip 不跨 boundary；对多镜头训练，保留 boundary token。

Step 4：clip 采样

可随机选择：

- 固定帧数 + 固定 fps；
- 固定真实时长 + 变帧数；
- 多帧率 bucket；
- 多时长 curriculum。

设 clip 起点 t_0、间隔 Δt：

$$\mathbf{x}_i = V(t_0 + i\Delta t), \quad i = 0, \dots, T-1.$$

Step 5：空间 bucket

保持宽高比，先缩放再裁剪或 padding。bucket 应使同 batch tensor 形状一致，同时减少内容裁掉。

Step 6：质量与内容过滤

组合规则/模型：

$$q(\mathbf{x}) = w_a q_{\text{aesthetic}} + w_i q_{\text{imaging}} + w_m q_{\text{motion}} - w_w q_{\text{watermark}} - \cdots.$$

不要只按一个总分硬阈值；可分层采样保留多样性。

Step 7：caption 与结构标注

理想 caption 包含：

- 主体、属性、数量；
- 动作、方向、速度；
- 交互和事件顺序；
- 场景和时间；
- 相机运动、景别；
- 风格和光照；
- 是否切镜。

可同时保存短 caption、详细 caption、结构化 scene graph 和 QA 对。

Step 8：去重和泄漏审计

训练集内部去重，并与验证/benchmark 按视觉、文本和视频指纹交叉去重。

Step 9：离线 latent/text cache

若 VAE/text encoder 冻结，可预计算：

$$\mathbf{z}_\star = E_\phi(\mathbf{x}), \qquad \mathbf{C}_y = F_T(\mathbf{y}).$$

优点是训练吞吐更高；缺点是占存储、数据增强受限、编码器升级需重算。必须保存编码器版本和 latent scaling。

11.3 Video VAE 的独立训练阶段

一个典型目标：

$$\mathcal{L}_{\text{VAE}} = \lambda_{\text{pix}}\mathcal{L}_{\text{pix}} + \lambda_{\text{perc}}\mathcal{L}_{\text{perc}} + \lambda_{\text{temp}}\mathcal{L}_{\text{temp}} + \lambda_{\text{KL}}\mathcal{L}_{\text{KL}} + \lambda_{\text{adv}}\mathcal{L}_{\text{adv}}.$$

课程式训练

1. 低分辨率、短 clip 稳定重建；
2. 增加空间分辨率；
3. 增加时间长度与运动；
4. 加 perceptual/adversarial loss；
5. 最后调 tiling/chunking 一致性。

冻结标准

在训练 DiT 前，应确认：

- latent 均值/方差稳定；
- 人脸、文字、小物体和快速运动重建可接受；
- 不同长度和 bucket 无明显边界伪影；
- encode/decode 可分块且结果一致；
- checkpoint、scaling factor 和 preprocessing 固定。

11.4 生成模型的分阶段预训练

图像预训练

把图像视为 $T = 1$ 视频，学习强空间先验。优点是高质量图文数据多；缺点是时间模块未训练。

短视频低分辨率预训练

学习基本运动和时间一致性。可冻结部分空间层，训练 temporal layer，或全参数联合训练。

多分辨率/多时长联合训练

按 token budget 采样 bucket。若每样本 token 数 N_b 不同，可让 batch 满足

$$\sum_{b\in\mathcal{B}} N_b \le N_{\text{budget}}.$$

这比固定样本数更公平。

高质量微调

使用更严格筛选的高分辨率数据，较小学习率，增强美学、动作和文本对齐。需防止分布变窄和遗忘长尾。

指令/编辑联合训练

将 T2V、I2V、V2V、R2V 统一为不同 condition mask:

$$\mathbf{c} = (\mathbf{y}, \mathbf{z}_{\text{source}}, \mathbf{m}_{\text{task}}, \mathbf{r}).$$

训练时随机选择任务，避免只为每个任务维护完全独立模型。

偏好优化与蒸馏

可在基础生成能力稳定后进行:

- reward-weighted regression;
- preference optimization;
- rejection sampling fine-tuning;
- consistency/trajectory distillation;
- adversarial distillation。

偏好阶段必须监控多样性、身份和物理，不可只优化美学分。

11.5 Flow Matching 训练批次

对每个样本:

1. 编码 $\mathbf{z}_\star = E_\phi(\mathbf{x})$;
2. 采样噪声 $\mathbf{z}^{(0)} \sim \mathcal{N}(0, I)$;
3. 采样时间 $\tau \sim p(\tau)$;
4. 构造路径 $\mathbf{z}_\tau$;
5. 随机 drop 条件;
6. DiT 预测 velocity;
7. 按时间/bucket 加权 MSE。

直线路径:

$$\mathbf{z}_\tau = (1-\tau)\mathbf{z}^{(0)} + \tau\mathbf{z}_\star,$$

$$\mathbf{u}_\tau = \mathbf{z}_\star - \mathbf{z}^{(0)}.$$

总损失可写为

$$\mathcal{L} = \mathbb{E}\left[\lambda(\tau, N, \boldsymbol{\eta})\|\mathbf{v}_\theta(\mathbf{z}_\tau, \tau, \mathbf{c}) - \mathbf{u}_\tau\|_2^2\right].$$

λ 可按噪声时间、分辨率、mask 或有效 token 调整。

11.6 条件 dropout 设计

为了 CFG，训练时以概率 p_{drop} 将文本替换为空条件：

$$\tilde{\mathbf{c}} = \begin{cases} \emptyset, & u < p_{\text{drop}}, \\ \mathbf{c}, & \text{otherwise.} \end{cases}$$

多条件时，可定义独立或组合 drop：

文本	参考	计划	用途
保留	保留	保留	完整条件模型
丢弃	丢弃	丢弃	无条件分支
保留	丢弃	丢弃	文本-only guidance
保留	保留	丢弃	参考增量
保留	保留	保留/扰动	计划鲁棒性

drop 概率过低，无条件分支学不好；过高会浪费条件训练容量。

11.7 优化器与学习率

常用 AdamW：

$$\mathbf{m}_t = \beta_1 \mathbf{m}_{t-1} + (1-\beta_1)\mathbf{g}_t,$$

$$\mathbf{v}_t = \beta_2 \mathbf{v}_{t-1} + (1-\beta_2)\mathbf{g}_t^2,$$

$$\theta_{t+1} = \theta_t - \eta_t \frac{\hat{\mathbf{m}}_t}{\sqrt{\hat{\mathbf{v}}_t} + \epsilon} - \eta_t \lambda_{\text{wd}} \theta_t.$$

需要记录的超参数

- base learning rate 与按 global batch scaling 规则；
- warmup tokens/steps；
- cosine/constant schedule；
- $\beta_1, \beta_2, \epsilon$；
- weight decay 排除项；
- gradient clipping；
- EMA decay；
- bf16/fp16/fp8 策略；
- loss scaling；
- gradient accumulation。

Global batch 的定义

样本数并不足够。至少报告：

$$B_{\text{global}} = B_{\text{micro}} \times n_{\text{DP}} \times n_{\text{accum}},$$

以及每步 video latent tokens：

$$N_{\text{step}} = \sum_{b=1}^{B_{\text{global}}} N_b.$$

训练规模最好报告总 seen tokens、总 frames/hours 和 GPU-hours。

11.8 混合精度与数值稳定

bf16

指数范围接近 fp32，通常比 fp16 更稳；尾数较短。大模型训练常优先 bf16。

fp16

精度略高但指数范围小，易 overflow，需要 dynamic loss scaling。

fp8

可显著提高吞吐和降低显存，但需 per-tensor/per-channel scaling、amax history 和敏感层保留高精度。视频长序列的 attention、normalization 和 VAE decode 需单独验证。

保留 fp32 的常见部分

- optimizer states;
- loss reduction;
- 某些 normalization/statistics;
- scheduler/time computation;
- FID/FVD covariance 与矩阵平方根。

11.9 分布式训练组合

一个大规模配置可写为

$$N_{\text{GPU}} = n_{\text{DP}} \times n_{\text{TP}} \times n_{\text{SP}} \times n_{\text{PP}}.$$

其中：

- DP：数据并行；
- TP：张量并行；
- SP/CP：序列/上下文并行；
- PP：流水并行。

通信与计算平衡

- 全局 attention 的 sequence parallel 通信大；
- window attention 更局部，但跨卡窗口划分复杂；
- VAE encode/decode 可与 DiT pipeline 异步；
- text encoder 若冻结可离线 cache；
- 多 bucket 会导致负载不均，需要按 token 数调度。

Checkpoint 内容

必须保存：

- model/EMA weights;
- optimizer、LR scheduler;
- gradient scaler;
- global step 与 seen tokens;
- RNG states;
- dataloader sampler state;
- VAE/text encoder hash;
- dataset manifest/version;
- distributed topology;
- git commit 与 config。

否则“续训”可能改变数据顺序或 scheduler，难以复现。

11.10 训练监控与在线样本

标量监控

- total loss 与按 τ 分桶 loss;
- 各分辨率/时长 bucket loss;
- gradient/update norm;
- throughput：samples/s、frames/s、tokens/s;
- MFU/HFU;
- data loading wait;
- all-reduce/all-to-all 时间;
- GPU memory 与 OOM retry;
- NaN/overflow 计数。

固定验证提示

使用固定 prompts、seeds、sampler 和 CFG 周期生成，覆盖：

- 人脸/手;
- 多对象;
- 快速运动;
- 静态相机;
- 指定运镜;
- 文字;
- 物理交互;
- 长尾风格。

只看训练 loss 无法发现条件忽略、运动坍塌和 VAE 闪烁。

11.11 推理 Pipeline

Step 1：输入解析

- 安全检查;
- prompt rewrite 或 MLLM planning;
- 解析分辨率、时长、fps、任务类型;
- 参考图/视频预处理。

Step 2：条件编码

$$\mathbf{C}_y = F_T(\mathbf{y}), \quad \mathbf{s} = P_\omega(\mathbf{y}, \mathbf{r}), \quad \mathbf{z}_s = E_\phi(\mathbf{x}_s).$$

Step 3：确定 latent shape

给定输出 T, H, W，按 VAE 压缩和 padding 规则得到

$$(T', H', W').$$

必须记录裁剪/补齐，否则 decode 后时长或分辨率可能偏一格。

Step 4：初始化噪声

$$\mathbf{z}_0 \sim \mathcal{N}(0, I).$$

长视频窗口可使用共享噪声、overlap latent 或确定性 seed 派生，减少边界变化。

Step 5：数值采样

对时间网格 $0 = \tau_0 < \cdots < \tau_K = 1$：

$$\mathbf{v}_k = \mathrm{CFG}(\mathbf{v}_\theta(\mathbf{z}_k, \tau_k, \mathbf{c})),$$

$$\mathbf{z}_{k+1} = \mathrm{ODEStep}(\mathbf{z}_k, \mathbf{v}_k, \tau_k, \tau_{k+1}).$$

Step 6: VAE decode

必要时 spatial tile、temporal chunk，使用 overlap blending。记录是否启用 stochastic decode。

Step 7: 后处理

可能包括：

- 色彩空间转换；
- 去噪/锐化；
- 插帧；
- 超分；
- 音频生成与同步；
- 水印/内容凭证；
- 视频编码。

评价模型本体时，应同时提供 raw decode 与 postprocessed 结果，防止把外部超分/插帧贡献归给生成模型。

11.12 推理成本分解

总延迟近似为

$$L_{\text{total}} = L_{\text{text/planner}} + K_{\text{NFE}} L_{\text{DiT}} + L_{\text{VAE}} + L_{\text{post}}.$$

若 CFG 需要 cond/uncond 两次前向：

$$K_{\text{effective}} \approx 2K$$

（批处理可降低 wall-clock，但 FLOPs 仍接近双倍）。Heun 每步两次函数评估，也要按 NFE 计算。

Real-time factor

对输出时长 S_{video}、生成延迟 L：

$$\mathrm{RTF} = \frac{L}{S_{\text{video}}}.$$

$\mathrm{RTF} < 1$ 表示生成快于播放时长。还应报告首帧延迟、峰值显存、吞吐和硬件。

每像素/每帧成本

不同分辨率与时长比较时，可报告：

$$\mathrm{LatencyPerMPFrame} = \frac{L}{T \cdot H \cdot W / 10^6}.$$

它仍不能完全消除注意力非线性，但比只报“生成一次几秒”更透明。

11.13 端到端训练伪代码

```
for batch in dataloader:
    video, prompt, metadata = batch.video, batch.text, batch.meta

    # Frozen components can be cached offline.
    with torch.no_grad():
        z_data = vae.encode(video) * latent_scale
        text_tokens = text_encoder(prompt)
        semantic_target = sem_encoder(video) if use_planner else None
```

```
# Optional semantic planner objective.
if use_planner:
    s_noise = torch.randn_like(semantic_target)
    tau_s = sample_time(batch_size, schedule=planner_schedule)
    s_t = interpolate(s_noise, semantic_target, tau_s)
    u_s = semantic_target - s_noise
    u_s_pred = planner(s_t, tau_s, text_tokens)
    planner_loss = weighted_mse(u_s_pred, u_s, tau_s)
else:
    planner_loss = 0.0

# Video latent flow objective.
z_noise = torch.randn_like(z_data)
tau = sample_time(batch_size, schedule=video_schedule)
z_t = interpolate(z_noise, z_data, tau)
u_target = z_data - z_noise

cond = condition_dropout(
    text_tokens=text_tokens,
    semantic_plan=semantic_target,
    metadata=metadata,
)
u_pred = video_dit(z_t, tau, cond)
flow_loss = weighted_mse(u_pred, u_target, tau, valid_mask=batch.mask)

loss = flow_loss + planner_weight * planner_loss
loss = loss / grad_accum_steps
scaler.scale(loss).backward()

if ready_to_step():
    scaler.unscale_(optimizer)
    clip_grad_norm_(model.parameters(), max_norm)
    scaler.step(optimizer)
    scaler.update()
    optimizer.zero_grad(set_to_none=True)
    lr_scheduler.step()
    ema.update(model)
```

11.14 复现实验卡（建议直接放论文附录）

数据

- 数据集版本、clip 数、总小时；
- resolution/fps/duration 分布；
- 去重、过滤、caption 方法；
- 训练/验证泄漏检查；
- 许可与不可公开部分说明。

模型

- VAE 架构与压缩率；
- latent scaling；
- patch size、hidden size、layers、heads；
- attention 类型与位置编码；
- text/MLLM 版本；
- 参数量（总/可训练）。

训练

- objective/prediction type；
- time distribution/shift；
- optimizer/LR/global token batch；
- total steps/tokens/frames；
- mixed precision；

- DP/TP/SP/PP/FSDP；
- GPU 型号、数量、GPU-hours；
- checkpoint 选择规则。

推理

- sampler、NFE；
- CFG 与 schedule；
- prompt rewrite/planner；
- resolution/frames/fps；
- VAE tiling；
- 后处理；
- latency/VRAM/hardware。

评价

- prompts 与 seeds；
- metric 版本/backbone/preprocessing；
- sample count；
- 置信区间；
- 人评协议；
- 失败样本与原始输出。

练习与自检

1. 对变分辨率训练，设计一个按 latent token budget 动态组 batch 的 sampler。2. 给定 64 张 GPU，提出 DP×TP×SP 的两种拓扑，分别适合“模型状态过大”和“单视频序列过长”的情形。3. 写出完整 latency accounting，说明为何“20-step Heun + CFG”不能简单称为 20 次前向。4. 设计一张训练 dashboard，将数据、数值、模型质量和系统吞吐指标分组。

12. 从零上手：学习路线、最小实验与论文阅读方法

本章面向已经掌握深度学习、但第一次进入视频生成的研究者。目标不是先运行最大的模型，而是用一组可控实验建立“表示-目标-网络-采样-评价”的因果直觉。

12.1 知识依赖图

建议按以下顺序掌握：

1. 条件概率生成与潜变量；
2. VAE/感知重建；
3. DDPM、score、CFG；
4. Flow Matching/Rectified Flow 与 ODE；
5. DiT、AdaLN、RoPE；
6. 视频 token 化与时空 attention；
7. 文本/多模态条件；
8. 数据和分布式系统；
9. 多维评价；
10. 具体模型与研究前沿。

不要先背模型名。能从一个开源配置反推出 latent shape、token 数、prediction target 和 NFE，才算真正入门。

12.2 六周学习计划

第 1 周：生成建模与扩散基础

- 推导 VAE ELBO；
- 推导 DDPM 的 $q(x_t|x_0)$ 和 posterior；
- 实现二维 toy data 的 DDPM 或 score model；
- 理解 ϵ、x_0、v 互换；
- 实验 CFG 对质量和多样性的影响。

产出：一份 3–5 页推导笔记和一个二维高斯混合采样 notebook。

第 2 周：Flow Matching 与数值求解

- 实现直线 Rectified Flow；
- 比较 Euler、Heun、RK4 的误差和 NFE；
- 可视化学习到的向量场；
- 研究 time sampling/shift；
- 理解 coupling 与 trajectory curvature。

产出：二维向量场图、NFE-质量曲线和一页误差分析。

第 3 周：Video VAE 与视频数据

- 构建小型 Moving-MNIST/Shapes 数据集；
- 训练 2D frame VAE 与 3D/2+1D VAE；
- 比较空间/时间压缩率；
- 测 PSNR、LPIPS、tLPIPS；
- 检查 tile/chunk boundary。

产出：VAE 重建诊断表和压缩率-质量-速度 Pareto 图。

第 4 周：Video DiT

- 在 latent 上实现最小 Video DiT；
- 比较 full、factorized 和 window attention；
- 实现 3D position encoding；
- 计算 token/FLOPs/显存；
- 在同一 toy dataset 上训练 flow model。

产出：三种 attention 的复杂度、吞吐和时间一致性对比。

第 5 周：条件生成与评价

- 增加文本/类别/轨迹条件；
- 实现 condition dropout 和 CFG；
- 构建 50–100 个可诊断 prompts；
- 实现 CLIP/video-text、flow-warp、dynamic degree；
- 做小规模 pairwise 人评。

产出：一张多维 radar/table，而不是一个总分。

第 6 周：复现开源模型与研究提案

- 阅读一个开源 T2V repo 的 config、scheduler 和 pipeline；
- 在有限 GPU 上运行低分辨率/短时推理；
- 做一个 LoRA/adapter 或 sampler 消融；
- 建立可复现 evaluation card；
- 提出一个可证伪的研究假设。

产出：复现实验报告、失败样本册和 2 页 research proposal。

12.3 最小项目 A：Video VAE 审计

数据

选择含以下场景的小数据集：

- 静态背景 + 平移物体；
- 快速小物体；
- 周期运动；
- 遮挡后重现；
- 文字或细线；
- 相机平移。

实验变量

- 2D VAE vs 3D VAE；
- 时间压缩 $f_t \in \{1, 2, 4\}$；
- 空间压缩 $f_s \in \{4, 8, 16\}$；
- pixel/perceptual/temporal loss；
- causal vs non-causal；
- full decode vs chunk decode。

评价

维度	指标
像素重建	PSNR, SSIM
感知重建	LPIPS
时间稳定	tLPIPS, warping error
小物体	detection/keypoint recall
文字	OCR accuracy
系统	encode/decode latency, VRAM, bitrate

关键结论

只有当 VAE oracle reconstruction 达到可接受质量，才值得训练更大的 DiT。否则生成结果上限被压缩器锁死。

12.4 最小项目 B：Moving Shapes 上的 Latent Rectified Flow

数据生成

每个视频包含 1–3 个不同颜色形状，随机初始位置和恒定速度，在边界弹回。条件文本可结构化为：

red circle moves right; blue square moves down; static camera.

数据生成器提供真值：对象 ID、轨迹、速度、碰撞和遮挡。

模型

- 小型 Video VAE；
- 6–12 层 Video DiT；
- hidden size 256–512；
- full 或 factorized attention；
- Flow Matching 直线路径；
- 文本可先用 learnable condition token，后换小型 text encoder。

可证伪实验

1. 去掉 temporal attention，身份一致性是否下降?
2. 训练时随机水平翻转但不改方向词，会出现多大方向错误?
3. CFG 从 1 到 10，语义、运动和多样性如何变化?
4. 时间压缩从 1 到 4，碰撞时刻误差如何变化?
5. full attention 与 factorized attention 在相同 FLOPs 下谁更好?

轨迹评价

检测生成形状中心 $\hat{\mathbf{p}}_{i,t}$，与最佳 identity matching 后的真值比较：

$$E_{\text{traj}} = \frac{1}{KT} \sum_{i,t} \|\hat{\mathbf{p}}_{i,t} - \mathbf{p}_{i,t}\|_2.$$

事件顺序和碰撞可直接用真值时间戳评价。这一 toy 项目比先看开放域主观视频更容易建立因果认识。

12.5 最小项目 C：开源 T2V 推理审计

对任一公开 checkpoint，不急于改代码，先完成：

配置反推

- VAE temporal/spatial compression；
- latent channel；
- patch size；
- token count；
- text encoder；
- DiT layers/hidden/heads；
- prediction type；
- scheduler endpoints；
- CFG 分支；
- default resolution/fps/frames；
- VAE tiling 与 offload。

单变量实验

固定 prompt 和 seed，分别改变：

- NFE；
- sampler；
- CFG；
- time shift；
- resolution；
- frames/fps；
- negative prompt；
- prompt rewrite；
- VAE tiling/offload。

每次只改变一个变量，并保存原始 metadata。把结果画成质量-延迟/显存曲线。

复现陷阱

- repo 默认会自动扩写 prompt；
- demo 使用私有 negative prompt；
- 输出视频容器 fps 与模型条件 fps 不同；
- 首次运行包含模型加载/编译时间；
- num_inference_steps 不等于 NFE；
- CFG cond/uncond 可能 batch 合并；
- CPU offload 降显存但显著增延迟；
- VAE decode 可能是主要显存峰值；
- 发布样例可能经过超分、插帧或挑 seed。

12.6 最小项目 D：评价工具箱

建议实现统一接口：

```
class VideoMetric:
    name: str
    higher_is_better: bool

    def prepare(self, device): ...
    def update(self, prompt, generated_video, real_video=None, source_video=None): ...
    def compute(self) -> dict[str, float]: ...
```

至少包含：

- metadata validator；
- frame/video-text alignment；
- FVD 或替代分布距离；
- subject/background feature consistency；
- optical-flow dynamic degree；
- motion-compensated flicker；
- atomic prompt QA；
- bootstrap CI；
- per-prompt CSV/JSON 输出。

评价目录建议

```
evaluation/
|-- prompts.jsonl
|-- generations/
|   |-- model_a/<prompt_id>/<seed>.mp4
|   `-- model_b/<prompt_id>/<seed>.mp4
|-- metadata/
|   `-- generation_config.json
|-- metrics/
|-- cache/
|   |-- video_features/
|   `-- optical_flow/
|-- reports/
|   |-- summary.csv
|   |-- prompt_level.parquet
|   `-- failures.html
`-- human_eval/
```

12.7 论文方法部分的阅读顺序

面对一篇新论文，按以下顺序，而不是从摘要中的宣传词开始：

1. 任务与输出规格

T2V/I2V/编辑？几秒、几帧、多少 fps、什么分辨率？单镜头还是多镜头？

2. 表示

Video VAE 还是离散 tokenizer? 压缩率、latent channel、因果性、是否公开?

3. 目标

DDPM、EDM、Flow Matching、Rectified Flow、autoregressive 或混合? prediction type 和时间方向?

4. 骨干

U-Net/DiT? patch、attention、位置编码、条件方式?

5. 数据

来源、小时数、caption、filter、去重、分阶段数据混合?

6. 训练规模

参数量、总 tokens/frames、GPU-hours、精度、并行拓扑?

7. 推理

sampler、NFE、CFG、planner、后处理、显存和延迟?

8. 评价

样本数、prompt、seed、metric 版本、人评、是否公平比较?

9. 消融

创新模块是否在相同 compute/data 下单独验证?

10. 失败案例

论文是否展示不利类别? 失败是否与声称解决的问题直接相关?

12.8 论文公式与代码的对齐方法

建立四列映射:

论文概念	数学符号	配置字段	代码入口
数据 latent	$\mathbf{z}_\star$	`latent_channels`, `vae_scale`	`vae.encode`
流时间	τ	`flow_shift`, `timestep_sampling`	scheduler
velocity target	$\mathbf{z}_\star - \epsilon$	`prediction_type`	loss function
3D patch	(p_t, p_h, p_w)	`patch_size`	patch embed
CFG	w	`guidance_scale`	pipeline loop
NFE	K	steps × calls/step	sampler

不要相信变量名。某 repo 的 `v_prediction` 可能是扩散 v，也可能是 flow velocity；某 `timestep` 可能从 data 到 noise 递增，也可能相反。以实际公式和 scheduler update 为准。

12.9 如何提出一个可研究的问题

一个合格问题应包含:

1. 可观测失败;
2. 可解释机制;
3. 最小干预;
4. 可证伪实验;
5. 不会被单一指标投机的评价。

示例：高压缩 VAE 是否是快速运动失败的主因?

- **观察**：快速小物体运动模糊、轨迹断裂；
- **假设**：时间/空间压缩丢失高频位移信息；
- **干预**：保持 DiT compute 近似相同，比较不同 VAE 压缩与 patch；
- **预测**：oracle VAE reconstruction 和生成轨迹误差同时改善；
- **反证**：若 oracle 重建正常但生成仍失败，主要根因不在 VAE；
- **评价**：LPIPS、tLPIPS、tracking error、FVD、吞吐。

示例：MLLM planner 是否改善事件顺序，而非只改善 prompt 长度?

- prompt rewrite baseline 使用同一 MLLM 输出详细文本；
- continuous-plan 模型输出时序语义 token；
- oracle plan 上界；
- time-shuffled plan 对照；
- 用原子事件完成率和顺序准确率评价；
- 匹配额外参数和推理 FLOPs。

12.10 建议的阅读主线

概率与扩散

1. VAE 与 ELBO；
2. DDPM；
3. score-based SDE；
4. classifier-free guidance；
5. EDM/噪声参数化；
6. Flow Matching；
7. Rectified Flow。

架构

1. Vision Transformer；
2. DiT；
3. 3D/分解时空 attention；
4. latent video diffusion；
5. 3D RoPE、MMDiT、序列并行。

视频生成

1. 早期 GAN/VAE/自回归视频生成；
2. Video Diffusion Models；
3. Video LDM；
4. 文生视频开放模型；
5. 大规模 Video DiT/Flow 系统；
6. planner-renderer、长视频和 world model。

评价

1. IS/FID；
2. FVD；
3. CLIPScore；
4. VBench/VBench++；
5. EvalCrafter/T2VScore；
6. compositional、physical、long-video 专项 benchmark；
7. FVD 局限与新 video representation distance。

具体模型谱系、Wan 系列、Bernini 完整实现、数据规模和算力将在中册系统展开。

12.11 自检题

1. 为什么 T2V 不能用像素回归的条件均值解决?
2. 给定 VAE 压缩和 patch，如何计算 Video DiT token 数?

3. ϵ-prediction 与 score 有何关系?
4. Rectified Flow 直线路径的 target velocity 是什么?
5. 为什么条件路径直不保证边缘生成轨迹直?
6. full 与 factorized attention 的复杂度分别是多少?
7. 3D RoPE 为什么应考虑 fps?
8. prompt rewrite 与 continuous semantic planner 的本质区别是什么?
9. CFG 为什么可能降低多样性和运动?
10. FVD 为什么不能评价 prompt adherence?
11. static video 为什么可能同时获得高 temporal consistency 和高 CLIPScore?
12. 如何区分 VAE 伪影与 DiT 伪影?
13. 为什么 `num_inference_steps` 不一定等于 NFE?
14. 如何以提示为单位构建置信区间?
15. 怎样证明一个 planner 模块确实被 renderer 使用?

核心结论

入门的完成标准不是“能调用 pipeline”，而是：给定一个结果，你能提出模块级根因；给定一个配置，你能估算 token/显存/NFE；给定一张排行榜，你能指出每个指标的含义和盲区；给定一个新想法，你能设计可证伪消融。

附录 A：核心公式速查表

A.1 条件生成

$$p_\theta(\mathbf{x} \mid \mathbf{c}), \qquad \hat{\mathbf{x}} \sim p_\theta(\mathbf{x} \mid \mathbf{c}).$$

潜空间生成：

$$\mathbf{z}_\star = E_\phi(\mathbf{x}), \qquad \hat{\mathbf{z}}_\star \sim p_\theta(\mathbf{z} \mid \mathbf{c}), \qquad \hat{\mathbf{x}} = D_\psi(\hat{\mathbf{z}}_\star).$$

A.2 VAE 与 ELBO

$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x} \mid \mathbf{z})] - D_{\mathrm{KL}}(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})).$$

重参数化：

$$\mathbf{z} = \boldsymbol{\mu}_\phi(\mathbf{x}) + \boldsymbol{\sigma}_\phi(\mathbf{x}) \odot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, I).$$

A.3 DDPM

前向一步：

$$q(\mathbf{z}_k \mid \mathbf{z}_{k-1}) = \mathcal{N}(\sqrt{\alpha_k}\mathbf{z}_{k-1}, (1-\alpha_k)I).$$

闭式：

$$\mathbf{z}_k = \sqrt{\bar{\alpha}_k}\mathbf{z}_\star + \sqrt{1-\bar{\alpha}_k}\boldsymbol{\epsilon}.$$

噪声损失：

$$\mathcal{L}_\epsilon = \mathbb{E}\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{z}_k, k, \mathbf{c})\|_2^2.$$

数据恢复：

$$\hat{\mathbf{z}}_\star = \frac{\mathbf{z}_k - \sqrt{1-\bar{\alpha}_k}\hat{\boldsymbol{\epsilon}}}{\sqrt{\bar{\alpha}_k}}.$$

A.4 连续扩散参数化

$$\mathbf{z}_\tau = \alpha_\tau \mathbf{z}_\star + \sigma_\tau \boldsymbol{\epsilon}.$$

扩散 v：

$$\mathbf{v} = \alpha_\tau \boldsymbol{\epsilon} - \sigma_\tau \mathbf{z}_\star.$$

反解：

$$\mathbf{z}_\star = \alpha_\tau \mathbf{z}_\tau - \sigma_\tau \mathbf{v},$$

$$\boldsymbol{\epsilon} = \sigma_\tau \mathbf{z}_\tau + \alpha_\tau \mathbf{v},$$

在 $\alpha_\tau^2 + \sigma_\tau^2 = 1$ 时成立。

score：

$$\mathbf{s}_\theta \approx -\frac{\boldsymbol{\epsilon}_\theta}{\sigma_\tau}.$$

A.5 Classifier-Free Guidance

$$\hat{\mathbf{u}} = \mathbf{u}_{\text{uncond}} + w(\mathbf{u}_{\text{cond}} - \mathbf{u}_{\text{uncond}}).$$

$\mathbf{u}$ 可为 ϵ、score、扩散 v 或 flow velocity，但必须在同一 $\mathbf{z}_\tau, \tau$ 上组合。

A.6 Flow Matching / Rectified Flow

连续 ODE:

$$\frac{d\mathbf{z}_\tau}{d\tau} = \mathbf{v}_\theta(\mathbf{z}_\tau, \tau, \mathbf{c}).$$

直线路径:

$$\mathbf{z}_\tau = (1-\tau)\mathbf{z}^{(0)} + \tau\mathbf{z}^{(1)}.$$

条件速度:

$$\mathbf{u}_\tau = \mathbf{z}^{(1)} - \mathbf{z}^{(0)}.$$

训练:

$$\mathcal{L}_{\text{RF}} = \mathbb{E}\left[\|\mathbf{v}_\theta(\mathbf{z}_\tau, \tau, \mathbf{c}) - (\mathbf{z}^{(1)} - \mathbf{z}^{(0)})\|_2^2\right].$$

Euler:

$$\mathbf{z}_{k+1} = \mathbf{z}_k + \Delta\tau_k \mathbf{v}_\theta(\mathbf{z}_k, \tau_k, \mathbf{c}).$$

Heun:

$$\tilde{\mathbf{z}}_{k+1} = \mathbf{z}_k + \Delta\tau_k \mathbf{v}_k,$$

$$\mathbf{z}_{k+1} = \mathbf{z}_k + \frac{\Delta\tau_k}{2}(\mathbf{v}_k + \tilde{\mathbf{v}}_{k+1}).$$

A.7 Video token 数

VAE 后:

$$T' = \left\lceil \frac{T}{f_t} \right\rceil, \quad H' = \left\lceil \frac{H}{f_h} \right\rceil, \quad W' = \left\lceil \frac{W}{f_w} \right\rceil.$$

Patch 后:

$$N = \left\lceil \frac{T'}{p_t} \right\rceil \left\lceil \frac{H'}{p_h} \right\rceil \left\lceil \frac{W'}{p_w} \right\rceil.$$

A.8 Attention

$$\text{Attn}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_h}}\right)V.$$

若 $N = FS$:

$$C_{\text{full}} = O(F^2 S^2 d),$$

$$C_{\text{factorized}} = O(FS^2d + SF^2d),$$

$$C_{\text{window}} = O(NKd).$$

A.9 AdaLN

$$\text{AdaLN}(h; e) = (1 + s(e)) \odot \text{LN}(h) + b(e).$$

门控残差：

$$h^+ = h + g(e) \odot F(\text{AdaLN}(h; e)).$$

A.10 Planner-Renderer

$$q_\omega(\mathbf{s} \mid \mathbf{y}, \mathbf{r}), \qquad p_\theta(\mathbf{z} \mid \mathbf{s}, \mathbf{y}, \mathbf{r}).$$

$$p(\mathbf{z} \mid \mathbf{y}, \mathbf{r}) = \int p_\theta(\mathbf{z} \mid \mathbf{s}, \mathbf{y}, \mathbf{r}) q_\omega(\mathbf{s} \mid \mathbf{y}, \mathbf{r}) d\mathbf{s}.$$

A.11 IS

$$\text{IS} = \exp\left(\mathbb{E}_x D_{\text{KL}}(p(y \mid x) \| p(y))\right).$$

A.12 FID/FVD

$$D_F = \|\mu_r - \mu_g\|_2^2 + \text{Tr}\left(\Sigma_r + \Sigma_g - 2(\Sigma_r^{1/2}\Sigma_g\Sigma_r^{1/2})^{1/2}\right).$$

FID 使用图像特征；FVD 使用视频 clip 特征。

A.13 CLIP/video-text alignment

$$S_{\text{align}} = \cos(f_V(\mathbf{x}), f_T(\mathbf{y})).$$

逐帧版本：

$$S = \frac{1}{T} \sum_i \cos(f_I(\mathbf{x}_i), f_T(\mathbf{y})).$$

A.14 PSNR、warping 与动态程度

$$\text{PSNR} = 10 \log_{10} \frac{L^2}{\text{MSE}}.$$

$$E_{\text{warp}} = \frac{1}{T-1} \sum_i \|x_{i+1} - \mathcal{W}(x_i, f_i)\|_1.$$

$$D_{\text{motion}} = \frac{1}{(T-1)HW} \sum_{i,h,w} \mathbb{I}(\|f_i(h, w)\| > \delta).$$

附录 B：术语表

术语	中文与精确定义
T2V	Text-to-Video；由文本条件采样视频分布
I2V	Image-to-Video；以首帧/参考图约束外观和构图
V2V	Video-to-Video；以源视频和指令生成目标视频
R2V/S2V	Reference/Subject-to-Video；保持参考主体身份
FLF2V	First-Last-Frame-to-Video；给定首尾帧生成中间过程
Video VAE	将像素视频压缩为连续时空 latent 的变分自编码器
Latent Diffusion	在 VAE latent 而非像素空间执行扩散
Causal VAE	时间编码/解码只依赖当前及过去帧，可支持流式处理
Temporal compression	VAE 在时间轴上的下采样比例 f_t
Spatial compression	VAE 在高宽轴上的下采样比例 f_h, f_w
Patchify	将 latent 切成 3D patch 并映射为 Transformer token
DiT	Diffusion Transformer；用于预测扩散/流目标的 Transformer
Video DiT	在时空 token 上运行的 DiT，含视频位置与 attention 设计
Full attention	所有时空 token 两两交互，复杂度 $O(N^2)$
Factorized attention	将空间注意力和时间注意力分开计算
Axial attention	分别沿时间、高度、宽度轴做注意力
Window attention	每个 token 只关注局部时空窗口
Sparse attention	只计算预定义或动态稀疏连接
FlashAttention	IO-aware 的精确 attention 实现，减少 HBM 读写和显存
RoPE	Rotary Position Embedding；用相位旋转编码相对位置
3D RoPE	分别对时间、高度、宽度坐标应用 RoPE
AdaLN	Adaptive LayerNorm；由时间/条件产生 scale 和 shift
AdaLN-Zero	近零初始化门控/投影，使初始 block 接近恒等
QK-Norm	对 query/key 做归一化以控制 attention logits
Cross-attention	一类 token 作 query，条件 token 作 key/value
Joint attention	文本和视觉 token 在共享/交互注意力中共同更新
MMDiT	Multimodal DiT；多模态 token 有独立投影并深度交互
DDPM	Denoising Diffusion Probabilistic Model
Noise schedule	随扩散时间规定信号/噪声比例的函数
Score	$\nabla_x \log p_t(x)$，概率密度对输入的梯度
SDE	Stochastic Differential Equation；连续随机扩散过程
Probability-flow ODE	与扩散 SDE 具有相同边缘分布的确定性 ODE
ϵ-prediction	预测加入样本的高斯噪声
x_0-prediction	直接预测干净数据
v-prediction	扩散文献中的信号-噪声线性组合参数化
Flow Matching	通过回归条件速度学习目标概率路径的向量场
Rectified Flow	常用直线概率路径并学习噪声到数据运输的 flow 方法
Vector field	在每个 (z, t) 位置给出瞬时速度的函数
Coupling	源分布样本与目标分布样本的联合配对分布
ODE solver	用有限 NFE 积分速度场的数值方法
Euler	一阶显式 ODE 步进方法
Heun	二阶 predictor-corrector 方法；每步通常两次函数评估
NFE	Number of Function Evaluations；网络前向调用次数
CFG	Classifier-Free Guidance；放大条件与无条件预测差
Condition dropout	训练时随机移除条件，使同一网络学习无条件分支
Guidance distillation	把多分支 CFG 行为蒸馏到更少前向的学生模型
Time shift / flow shift	对训练/采样时间分布做非线性变换
MLLM	Multimodal Large Language Model；能处理语言和视觉输入
MLLM planner	先预测结构化/连续高层视频语义，再交给 renderer
Prompt rewriting	用 LLM 扩写文本；不必产生视频对齐的中间计划
Semantic plan	介于文本与像素之间的高层视觉/时间表示
Renderer	根据文本、计划和参考条件生成 VAE latent/像素的模型
ViT embedding	Vision Transformer 的连续视觉语义表示
Masked generative modeling	对被 mask 的 token/连续表示进行条件生成
Teacher forcing	训练下游模块时使用真实上游输出

术语	中文与精确定义
Exposure gap	训练看真实条件、推理看预测条件造成的分布差异
FVD	Fréchet Video Distance；视频特征高斯的 Fréchet 距离
FID	Fréchet Inception Distance；图像特征分布距离
IS	Inception Score；分类置信度与类别多样性的组合
CLIPScore	基于 CLIP 文图 embedding 相似度的对齐分数
KVD	Kernel Video Distance；视频特征上的核分布距离
MMD	Maximum Mean Discrepancy；核均值嵌入的分布差异
JEDi	使用 JEPA 类视频 embedding 与 MMD 的评价方法
VBench	将视频质量和文本一致性分成 16 个维度的基准
VBench++	扩展到 I2V、长视频、可信等任务的评价框架
T2VScore	分离文本对齐和视频质量的综合评价方法
Dynamic degree	视频中显著运动的幅度/比例，不等价于运动正确性
Temporal flicker	非真实运动导致的帧间高频变化
tLPIPS	时间/运动补偿后的感知差异指标
Warping error	用光流 warp 后相邻帧的重建残差
Subject consistency	主体身份/外观在时间上的稳定性
Background consistency	背景结构和纹理在时间上的稳定性
Compositionality	正确组合对象、属性、关系、动作、数量与时间约束
Physical commonsense	对材料、接触、守恒和真实动作规律的遵循
Shot	两次剪辑边界之间的连续镜头
Scene	语义上同一地点/事件的一组镜头；不等同于 shot
Long-video drift	滚动生成中身份、场景或叙事状态逐步偏离
Sequence parallel	沿 token 序列维分片 Transformer 计算
Tensor parallel	沿隐藏维度/注意力头分片线性计算
FSDP/ZeRO	分片参数、梯度和优化器状态的训练技术
Activation checkpointing	反向重算部分前向以换取更低激活显存
Latent cache	预计算冻结 VAE/text encoder 输出以提高训练吞吐
RTF	Real-Time Factor；生成时间除以输出视频时长
MFU	Model FLOPs Utilization；实际有效模型 FLOPs 与硬件峰值之比
LoRA	用低秩矩阵增量微调冻结权重的方法
Distillation	用教师轨迹/输出训练更快或更小学生模型
Reflow	用模型生成 coupling/trajectory 重新训练 flow，使路径更直
Consistency model	学习不同噪声时间状态映射一致性，以支持少步生成
World model	显式/隐式建模环境状态、动作和未来转移的生成模型

附录 C：论文与实验检查清单

C.1 方法正确性

- □ 数据端点与噪声端点定义无歧义；
- □ prediction type 与 scheduler update 一致；
- □ 训练/推理的时间方向一致；
- □ latent scaling 在 encode、训练、decode 中一致；
- □ fps、frames、duration 语义一致；
- □ condition dropout 能覆盖所需 CFG 分支；
- □ source mask/known region 在每个噪声水平正确加噪；
- □ padding token 不进入 loss/attention；
- □ 多分辨率坐标和 RoPE 正确；
- □ 变帧率视频使用真实时间或明确重采样。

C.2 计算与系统

- □ 参数量区分总参数与可训练参数；
- □ 报告 latent token 数而非只报像素分辨率；
- □ 报告 NFE 而非只报 sampler steps；
- □ 延迟含 text/planner、DiT、VAE 和后处理；
- □ 峰值显存包含 VAE decode；
- □ 首次编译/模型加载与稳态推理解耦；
- □ 相同硬件、精度和 batch 比较速度；
- □ GPU 数和 GPU-hours 均报告；
- □ 数据加载和通信瓶颈被监控；
- □ checkpoint 可恢复 RNG 和 dataloader state。

C.3 评价

- □ 同一 prompt、seed 集和输出规格；
- □ FVD/FID 使用相同样本数和特征版本；
- □ CLIP-based 指标说明帧采样/聚合；
- □ 时间一致性与 dynamic degree 联合报告；
- □ 组合性/物理至少一个专项 benchmark；
- □ 自动 judge 经人评验证；
- □ prompt-level bootstrap 与 95% CI；
- □ 公开完整维度，不只报 total；
- □ 展示随机或预注册样本，不只挑最好 seed；
- □ raw decode 与后处理结果区分。

C.4 Planner 模型

- □ prompt rewriting baseline；
- □ no-plan baseline；
- □ oracle-plan 上界；
- □ predicted-plan 主结果；
- □ shuffled/corrupted-plan 诊断；
- □ renderer 是否真正依赖 plan；
- □ planner 额外参数、NFE、延迟；
- □ 计划表示的时间/空间分辨率；
- □ 真计划与预测计划的 exposure gap；
- □ 长视频状态维护方式。

C.5 数据透明度

- □ clip 数、总小时、帧数/token 数；
- □ 分辨率/fps/时长分布；
- □ caption 来源和重写模型；
- □ motion/quality/aesthetic filter；
- □ shot cut 处理；

- ☐ 去重与 benchmark 泄漏；
- ☐ 版权、隐私和删除机制；
- ☐ 不同训练阶段的数据混合；
- ☐ 数据版本可追踪；
- ☐ 训练样本复现/记忆测试。

附录 D：代表性文献与继续阅读

以下按主题给出学习入口。中册会把这些工作放入模型谱系，并补充 Wan、CogVideoX、HunyuanVideo、LTX、Open-Sora、Bernini 等实现与训练规模。

D.1 概率生成、VAE 与 GAN

1. Kingma, D. P., and Welling, M. **Auto-Encoding Variational Bayes**. ICLR 2014.
2. Goodfellow, I. et al. **Generative Adversarial Nets**. NeurIPS 2014.
3. Esser, P., Rombach, R., and Ommer, B. **Taming Transformers for High-Resolution Image Synthesis**. CVPR 2021.
4. Rombach, R. et al. **High-Resolution Image Synthesis with Latent Diffusion Models**. CVPR 2022.

D.2 扩散、score 与 guidance

5. Sohl-Dickstein, J. et al. **Deep Unsupervised Learning using Nonequilibrium Thermodynamics**. ICML 2015.
6. Ho, J., Jain, A., and Abbeel, P. **Denoising Diffusion Probabilistic Models**. NeurIPS 2020.
7. Song, Y. et al. **Score-Based Generative Modeling through Stochastic Differential Equations**. ICLR 2021.
8. Nichol, A., and Dhariwal, P. **Improved Denoising Diffusion Probabilistic Models**. ICML 2021.
9. Ho, J., and Salimans, T. **Classifier-Free Diffusion Guidance**. 2021.
10. Karras, T. et al. **Elucidating the Design Space of Diffusion-Based Generative Models**. NeurIPS 2022.
11. Salimans, T., and Ho, J. **Progressive Distillation for Fast Sampling of Diffusion Models**. ICLR 2022.
12. Lu, C. et al. **DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps**. NeurIPS 2022.

D.3 Flow Matching 与 Rectified Flow

13. Lipman, Y. et al. **Flow Matching for Generative Modeling**. ICLR 2023.
14. Liu, X., Gong, C., and Liu, Q. **Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow**. ICLR 2023.
15. Albergo, M. S., and Vanden-Eijnden, E. **Building Normalizing Flows with Stochastic Interpolants**. ICLR 2023.
16. Tong, A. et al. **Improving and Generalizing Flow-Based Generative Models with Minibatch Optimal Transport**. 2023.
17. Zheng, Q. et al. **Guided Flows for Generative Modeling and Decision Making**. 2023.

D.4 Transformer、DiT 与位置编码

18. Vaswani, A. et al. **Attention Is All You Need**. NeurIPS 2017.
19. Dosovitskiy, A. et al. **An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale**. ICLR 2021.
20. Peebles, W., and Xie, S. **Scalable Diffusion Models with Transformers**. ICCV 2023.
21. Su, J. et al. **RoFormer: Enhanced Transformer with Rotary Position Embedding**. Neurocomputing 2024; preprint 2021.
22. Dao, T. et al. **FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness**. NeurIPS 2022.
23. Dao, T. **FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning**. ICLR 2024.

D.5 视频生成基础

24. Vondrick, C., Pirsiavash, H., and Torralba, A. **Generating Videos with Scene Dynamics**. NeurIPS 2016.
25. Tulyakov, S. et al. **MoCoGAN: Decomposing Motion and Content for Video Generation**. CVPR 2018.
26. Clark, A., Donahue, J., and Simonyan, K. **Efficient Video Generation on Complex Datasets**. 2019.
27. Ho, J. et al. **Video Diffusion Models**. NeurIPS 2022.
28. Singer, U. et al. **Make-A-Video: Text-to-Video Generation without Text-Video Data**. ICLR 2023.

29. Ho, J. et al. **Imagen Video: High Definition Video Generation with Diffusion Models**. 2022.
30. Villegas, R. et al. **Phenaki: Variable Length Video Generation from Open Domain Textual Descriptions**. ICLR 2023.
31. Blattmann, A. et al. **Align Your Latents: High-Resolution Video Synthesis with Latent Diffusion Models**. CVPR 2023.
32. Wang, X. et al. **VideoComposer: Compositional Video Synthesis with Motion Controllability**. NeurIPS 2023.
33. Chen, H. et al. **VideoCrafter1: Open Diffusion Models for High-Quality Video Generation**. 2023.
34. Chen, H. et al. **VideoCrafter2: Overcoming Data Limitations for High-Quality Video Diffusion Models**. CVPR 2024.
35. Wang, J. et al. **ModelScope Text-to-Video Technical Report**. 2023.
36. Yang, Z. et al. **CogVideo: Large-scale Pretraining for Text-to-Video Generation via Transformers**. ICLR 2023.
37. Yang, Z. et al. **CogVideoX: Text-to-Video Diffusion Models with An Expert Transformer**. 2024.
38. Kong, W. et al. **HunyuanVideo: A Systematic Framework for Large Video Generative Models**. 2024.
39. Wan Team. **Wan: Open and Advanced Large-Scale Video Generative Models**. arXiv:2503.20314, 2025.
40. Bernini Team. **Bernini: Latent Semantic Planning for Video Diffusion**. arXiv:2605.22344, 2026.

D.6 数据集与 caption

41. Bain, M. et al. **Frozen in Time: A Joint Video and Image Encoder for End-to-End Retrieval**; introduces WebVid-2M. ICCV 2021.
42. Xue, H. et al. **Advancing High-Resolution Video-Language Representation with Large-Scale Video Transcriptions**; HD-VILA-100M. CVPR 2022.
43. Chen, T.-S. et al. **Panda-70M: Captioning 70M Videos with Multiple Cross-Modality Teachers**. CVPR 2024.
44. Wang, Y. et al. **InternVid: A Large-scale Video-Text Dataset for Multimodal Understanding and Generation**. ICLR 2024.
45. OpenVid contributors. **OpenVid-1M: A Large-Scale High-Quality Dataset for Text-to-Video Generation**. 2024.
46. VidCapBench authors. **VidCapBench: A Comprehensive Benchmark of Video Captioning for Controllable Text-to-Video Generation**. 2025.

D.7 评价指标与 benchmark

47. Salimans, T. et al. **Improved Techniques for Training GANs**; introduces Inception Score. NeurIPS 2016.
48. Heusel, M. et al. **GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium**; introduces FID. NeurIPS 2017.
49. Unterthiner, T. et al. **Towards Accurate Generative Models of Video: A New Metric and Challenges**; introduces FVD. 2018.
50. Hessel, J. et al. **CLIPScore: A Reference-free Evaluation Metric for Image Captioning**. EMNLP 2021.
51. Huang, Z. et al. **VBench: Comprehensive Benchmark Suite for Video Generative Models**. CVPR 2024.
52. Huang, Z. et al. **VBench++: Comprehensive and Versatile Benchmark Suite for Video Generative Models**. 2024.
53. Liu, Y. et al. **EvalCrafter: Benchmarking and Evaluating Large Video Generation Models**. CVPR 2024.
54. Wu, J. Z. et al. **Towards A Better Metric for Text-to-Video Generation**; T2VScore. 2024.
55. Sun, K. et al. **T2V-CompBench: A Comprehensive Benchmark for Compositional Text-to-Video Generation**. 2024.
56. Bansal, H. et al. **VideoPhy: Evaluating Physical Commonsense for Video Generation**. 2024.
57. Bansal, H. et al. **VideoPhy-2: A Challenging Action-Centric Physical Commonsense Evaluation in Video Generation**. 2025.
58. Yuan, S. et al. **ChronoMagic-Bench: A Benchmark for Metamorphic Evaluation of Text-to-Time-lapse Video Generation**. 2024.
59. Luo, G. Y. et al. **Beyond FVD: Enhanced Evaluation Metrics for Video Generation Quality**; proposes JEDi. 2024.

60. Guan, K. et al. **ETVA: Evaluation of Text-to-Video Alignment via Fine-grained Question Generation and Answering**. 2025.
61. Guo, X. et al. **T2VTextBench: A Human Evaluation Benchmark for Textual Control in Video Generation Models**. 2025.
62. Chen, Y. et al. **T2VWorldBench: A Benchmark for Evaluating World Knowledge in Text-to-Video Generation**. 2025.
63. Ghosh, D. et al. **GenEval: An Object-Focused Framework for Evaluating Text-to-Image Alignment**. NeurIPS 2023 Workshop / arXiv 2023.

D.8 建议的阅读动作

对每篇论文完成四件事:

1. 用本册符号重写其生成目标;
2. 从配置计算实际 latent token 数;
3. 把训练和推理成本拆成 VAE、条件编码、DiT NFE 和后处理;
4. 按第 9 章判断其评价是否覆盖声称解决的问题。

结语：上册的知识闭环

本册建立了一个统一闭环：

视频/文本数据 → Video VAE 表示 → 扩散或 Flow Matching 目标 → Video DiT 时空计算 → 文本/MLLM 规划条件 → 数值采样与解码

掌握这条链路后，模型名只是具体设计点的组合。中册将沿这一框架逐一剖析主流系统，重点回答：Wan 各版本具体如何压缩视频、构造 DiT、组织数据和训练；Bernini 的 MLLM planner、连续语义空间、renderer 与统一生成/编辑如何落到实现；CogVideoX、HunyuanVideo、LTX、Open-Sora 等为什么做出不同工程取舍；公开信息能支持怎样的预训练规模、GPU-hours、显存和推理成本估算。