Institute of Trustworthy Embodied AI
Fudan University

FlashMotion：轨迹引导的少步可控视频生成

Quanhao Li[1,2] ZhenXing[1,2] Rui Wang[1,2] Haidong Cao[1,2]
QiDai[3] Daoguo Dong[1,2] ZuxuanWu[1,2,†]

[1]可信具身人工智能研究院、复旦大学，[2]上海多模态具身人工智能重点实验室，
[3]微软亚洲研究院

arXiv:2603.12146v1 [cs.CV] 12 Mar 2026

摘要

近年来，轨迹可控视频生成取得了显著进展。以往方法主要采用基于适配器的架构，沿预定义轨迹实现精确运动控制。然而，这些方法都依赖多步去噪过程，造成大量时间冗余和计算开销。现有视频蒸馏方法虽能成功将多步生成器蒸馏为少步生成器，但直接将这些方法用于轨迹可控视频生成，会导致视频质量和轨迹准确度均明显下降。为弥合这一差距，我们提出**FlashMotion**，一种专为少步轨迹可控视频生成设计的新型训练框架。我们首先在多步视频生成器上训练轨迹适配器，以实现精确的轨迹控制。随后，将生成器蒸馏为少步版本，以加速视频生成。最后，我们采用结合扩散与对抗目标的混合策略对适配器进行微调，使其与少步生成器对齐，从而生成高质量且轨迹准确的视频。为进行评估，我们推出**FlashBench**，这是一个面向长序列轨迹可控视频生成的基准测试，可衡量不同前景物体数量下的视频质量和轨迹准确度。在两种适配器架构上的实验表明，无论是视觉质量还是轨迹一致性，FlashMotion均优于现有视频蒸馏方法和以往的多步模型。

代码： https://github.com/quanhaol/FlashMotion**网站：**
https://quanhaol.github.io/flashmotion-site/

1 引言

扩散模型的兴起 [12, 35, 36] 显著推动了视频生成领域的发展，使近期模型 [8, 16, 40, 50, 52, 53, 55, 64] 能够直接根据文本或视觉输入合成高质量视频。在这些进展的基础上，轨迹可控视频生成进一步引入了用户定义的运动控制，使视频能够按照指定的轨迹模式生成 [5, 18, 26, 41, 57, 62, 63]。尽管生成能力令人瞩目，以往的方法仍需要多个去噪步，而如图1 （a）所示，直接减少步数可能会导致严重的模糊伪影。

为解决这一高　　计算负担，近期提出的视频蒸馏方法旨在将　　ll

†通讯作者。

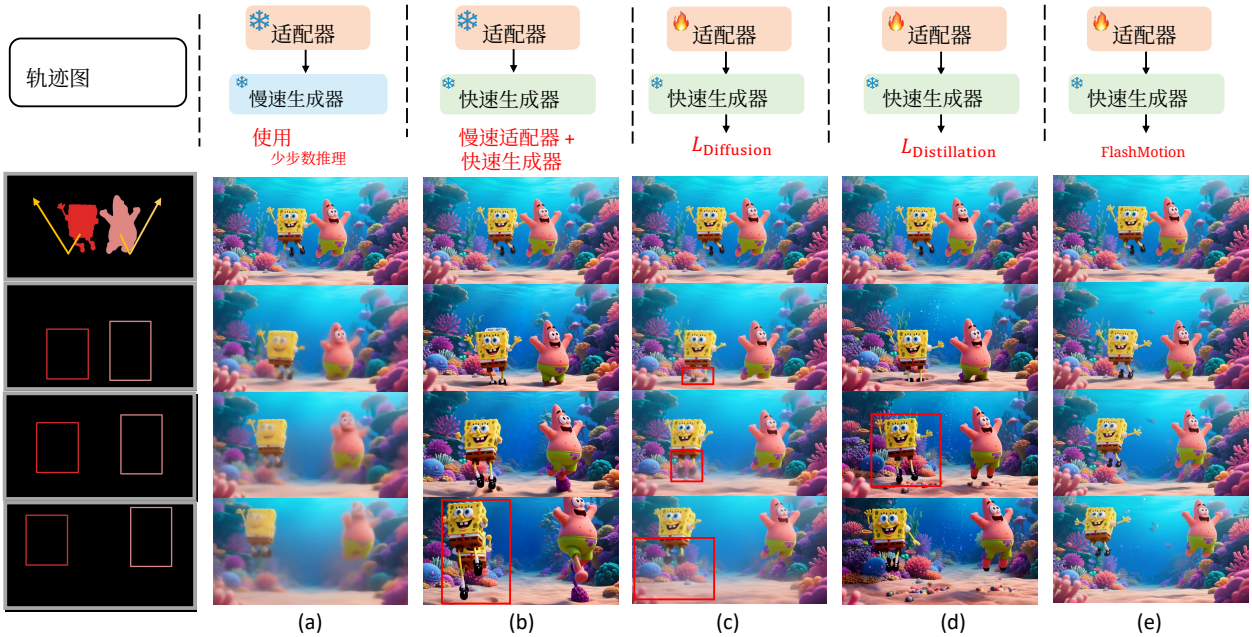

图 1 展示了**FlashMotion**的研究动机和能力。我们将*SlowGenerator* 定义为多步视频模型，将*FastGenerator* 定义为其少步蒸馏版本。*SlowAdapter*与*SlowGenerator*一起训练，而*FastAdapter* 则针对*FastGenerator*进行微调。(a) 在少步推理中将 *SlowAdapter* 与*SlowGenerator* 搭配使用会导致输出模糊。(b) 将*SlowAdapter*应用于*FastGenerator*会降低视频质量和轨迹准确度。(c) 仅使用扩散损失微调适配器仍会产生模糊伪影。(d) 使用现有蒸馏方法微调适配器，视频质量和轨迹控制效果仍不理想。(e) FlashMotion实现了高质量、准确的少步轨迹可控视频生成。

多步教师模型蒸馏为少步学生模型，从而显著加快视频生成过程 [2, 13, 20, 21, 23, 33, 38, 43, 60]。然而，将这些方法直接应用于轨迹可控视频生成可能会导致效果不佳（图1 （d）），而轨迹可控视频生成的加速仍基本尚未得到探索。

一种直接明了的方法是，利用现有策略将训练良好的多步视频生成器（*SlowGenerator*）蒸馏为少步学生模型（CogVideoX [55],等），例如 Wan [40],*FastGenerator*），同时保持原有的轨迹适配器（*SlowAdapter*）不变。然而，如图 1(b)所示，这会显著降低视频质量和轨迹准确度，表明SlowAdapter 与*FastGenerator*并不直接兼容。这种不兼容源于SlowAdapter 专为*SlowGenerator*的多步去噪过程设计，在此过程中，轨迹条件通过渐进式细化缓慢引导初始噪声。相比之下，*FastGenerator* 仅需少数几个去噪步即可合成视频，因此其去噪路径截然不同。

本文提出了**FlashMotion**，这是一种新颖的训练框架，通过在*SlowAdapter*之上适配*FastGenerator*，实现少步数的轨迹可控视频生成。我们发现，直接微调*SlowAdapter* 以适配*FastGenerator* 并使用标准扩散损失，虽能实现合理的轨迹对齐，但生成的视频存在严重的模糊伪影（图1（c））。这是因为扩散损失仅提供像素级监督，而不强制实现分布级一致性，导致生成（伪）数据分布与真实数据分布之间存在偏差。

为缓解这一问题，FlashMotion 引入扩散判别器，以引导轨迹适配器的优化，弥合生成视频分布与真实视频分布之间的差距。具体而言，我们采用混合训练策略对 *SlowAdapter* 进行微调，同时优化扩散与对抗目标。扩散判别器经过训练，用于区分带噪的真实视频潜变量与生成视频潜变量，从而

使二者的底层数据分布保持一致。同时，扩散损失提供像素级监督，促使模型生成轨迹对齐视频。为平衡这两个目标并确保优化稳定，我们进一步引入动态扩散损失缩放机制，在训练过程中自适应地调整损失权重。此外，得益于 *SlowAdapter*提供的强先验，该训练阶段仅需在 4 个 A100 GPU 上进行 1000步的轻量级微调，从而将训练成本降至最低。

除了训练框架之外，我们也迫切需要一个全面的基准测试。现有的轨迹可控视频生成基准测试 [18, 25, 28] 受限于视频时长较短和轨迹标注有限。为克服这些限制，我们推出了**FlashBench**，这是一个大规模、全面的基准测试，为长视频序列提供轨迹标注。FlashBench 还根据前景物体的数量将视频分为六类，并按照 [18]，从视觉质量和轨迹控制准确度两方面评估各类别中的模型。总而言之，我们的主要贡献如下：

- 据我们所知，FlashMotion 是首个研究少步轨迹可控视频生成的工作。我们提出并系统考察了一系列有潜力的方法，并进行了深入分析与比较。
- 我们提出了一种新颖的三阶段训练框架，融合扩散与对抗目标，从而能够在少步视频扩散模型之上有效训练轨迹适配器。FlashMotion 大幅加速了视频生成，同时提升了视觉保真度和轨迹准确度。
- 我们提出了FlashBench，这是一个由带有详细轨迹标注的长视频序列构成的大规模基准测试。大量实验表明，FlashMotion取得了更优的性能，同时优于少步蒸馏方法和多步轨迹引导视频生成方法。

2 相关工作

轨迹可控视频生成轨迹可控视频生成因其能够在视频生成过程中精确控制前景物体的运动轨迹，近来受到了广泛关注。一些免训练方法试图通过直接操纵特定空间区域内的注意力图值来实现轨迹控制 [14, 24, 29, 54]。然而，由于缺乏显式的轨迹监督，这类方法往往难以实现一致且时间连贯的运动控制。近期基于训练的方法引入了用于轨迹控制的可学习模块，从而能够将各种轨迹表示用作条件信号 [4, 7, 18, 41, 42, 45–48, 56, 62, 65]。通过这些结构化条件显式建模轨迹，轨迹适配器能够有效地将细粒度时空控制注入视频生成过程。尽管可控性有所提升，这些方法仍依赖于包含数十次乃至数百次去噪迭代的多步扩散推理，从而导致显著的延迟和计算开销。相比之下，FlashMotion提出了一种少步轨迹可控视频生成模型，在保持视觉质量和轨迹可控性的同时大幅减少去噪迭代次数。

视频扩散蒸馏 步数蒸馏是加速扩散模型的一种常见且有效的方法。现有的视频蒸馏方法主要改编自图像蒸馏方法，大致可分为三类：一致性蒸馏、分数蒸馏和对抗蒸馏。一致性蒸馏 [22, 37] 通过将概率流轨迹上的任意点直接映射回其起点，从而实现单步生成。VideoLCM[43], 、T2V-Turbo[17], 和DCM[23]等方法将这一概念扩展到视频领域，从而以极少的采样步数实现高效的视频合成。分数蒸馏 [58, 59] 侧重于最小化学生模型与教师模型的分数估计之间的差异。近期的视频方法，如POSE [2], 、MagicDistillation[33], 、CausVid [60],和Self-Forcing[13] ，采用分数蒸馏目标，旨在逼近多步扩散教师模型的同一分布。对抗蒸馏[6, 31, 32] 则使用判别器来

缩小真实样本与生成样本之间的分布差异。在视频领域，APT[20] 和APT2 [21]利用这一策略进行单步对抗蒸馏，训练一个判别器来区分真实视频与蒸馏生成器合成的视频。尽管效率提升显著，现有的视频蒸馏方法并非专门为轨迹可控视频生成而设计，直接应用于该任务时，往往会导致视觉质量和轨迹准确度下降。

3 方法

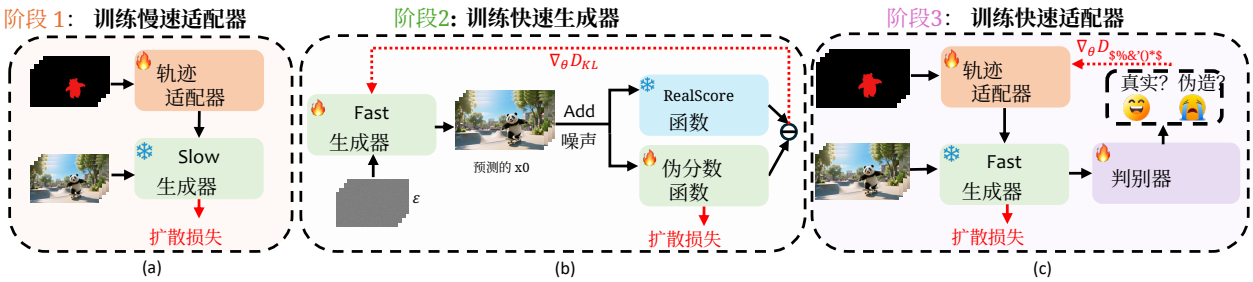

图 2 FlashMotion 训练流程概览。FlashMotion 分三个阶段进行训练：(1) 首先，在*SlowGenerator* 上使用扩散损失训练*SlowAdapter* ； (2) 在分布匹配 [59] 损失的监督下，从*SlowGenerator*蒸馏得到*FastGenerator* ； (3) 最后，采用结合对抗损失与扩散损失的混合训练策略，对*SlowAdapter* 进行微调，使其与*FastGenerator* 对齐。

3.1 概述

我们提出了**FlashMotion**，这是一种可通过轨迹控制的图像到视频框架，能够以较少的去噪步生成高质量且轨迹一致的视频，同时实现可控性与效率。如图2所示，FlashMotion 通过三阶段训练过程实现这一目标。在第3.2节中，我们详细介绍*SlowAdapter*的训练，包括其模型架构和渐进式训练流程。在第3.3节中，我们详细介绍*FastGenerator*的训练，该训练通过将多步教师模型蒸馏为少步学生模型来实现。在第3.4节中，我们说明如何通过同时采用扩散与对抗目标的混合训练方案，将*SlowAdapter* 适配为*FastAdapter* 。最后，我们将在第3.5节介绍 FlashBench，这是专为评估长时视频序列而设计的综合基准测试。

3.2 训练 SlowAdapter

如图2 (a) 所示，FlashMotion 首先在*SlowGenerator*上使用标准扩散损失训练轨迹适配器。接下来，我们将介绍其架构和训练过程。

轨迹适配器架构 我们设计了两种不同的轨迹适配器架构，用于评估 FlashMotion 的泛化能力：基于ControlNet的适配器 [61] 和轻量级基于ResNet的适配器 [9]。具体而言，我们的轨迹适配器中的块数量与Wan2.2-TI2V-5B [40]中的 DiT [27]块数量保持一致。我们使用预训练的 3D VAE [15] 编码器将轨迹图编码到潜在空间 $Z_{trajectory} \in R^{\frac{T}{4} \times \frac{H}{16} \times \frac{W}{16} \times 48}$中，随后将其作为轨迹适配器的输入。每个轨迹适配器块的输出随后经过零初始化的卷积层，并与基础模型 [51, 61],中的对应 DiT 块相加，从而提供轨迹引导。

训练流程 遵循 MagicMotion [18],，我们采用由密到疏训练策略，逐步提升适配器的轨迹理解能力。首先使用分割掩码作为稠密轨迹条件训练适配器，随后使用边界框作为稀疏轨迹条件对其进行微调。通过这一两阶段训练过程，我们获得了*SlowAdapter* ，它能够为*SlowGenerator*提供轨迹引导。

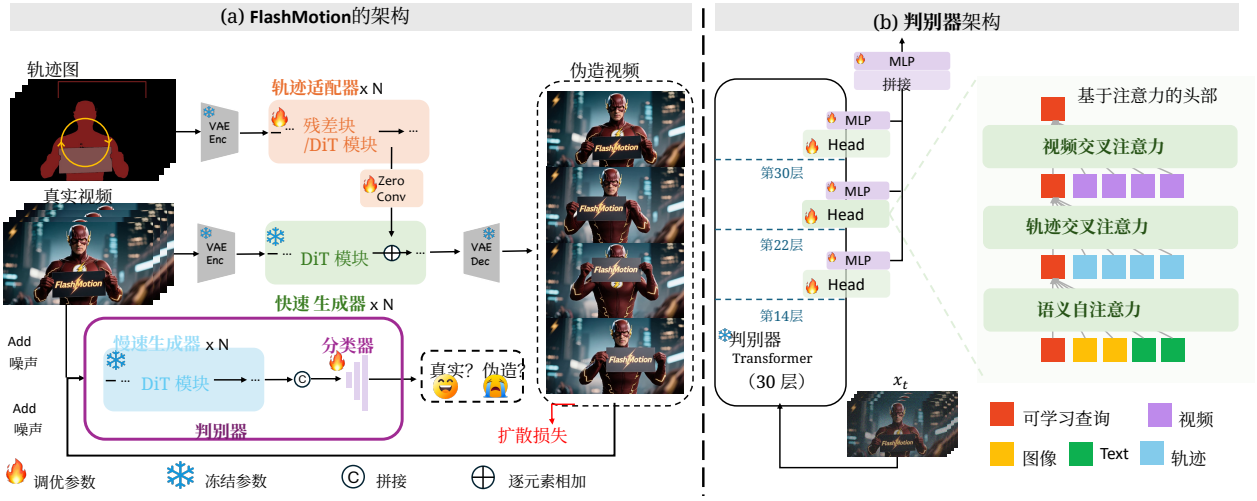

图3 (a) **FlashMotion**的架构。轨迹适配器在*FastGenerator*上进行微调，采用同时结合扩散与对抗目标的混合策略。(b) 我们的扩散判别器架构的详细图示。判别器采用从*SlowGenerator*克隆而来的DiT骨干网络，并将其DiT模块的若干中间特征输入基于注意力的分类器，以区分真实视频与生成视频。

3.3 训练FastGenerator

我们的目标是将*SlowGenerator* 蒸馏为*FastGenerator*，使其仅用几个去噪步就能生成高质量的视频序列。具体而言，我们采用 Wan2.2-TI2V-5B [40] 作为*SlowGenerator*。它基于 DiT [27] 架构构建，并使用一系列 Transformer [39] 模块进行迭代去噪。

对于蒸馏，我们采用DMD [59], ，一种分数蒸馏方法，通过对齐教师与学生视频分布 p_{real} 和 p_{fake} ，最小化它们的KL散度。我们在此考虑三个组件：少步学生生成器 G_θ、真实分数模型 μ_{real}和虚假分数模型 μ_{fake}，均由Wan2.2-TI2V-5B [40]的权重初始化。如图2(b)所示，我们首先使用 G_θ 执行少步推理过程，将纯高斯噪声 $\epsilon \sim \mathrm{N}\,0, I$ 映射为干净视频样本 x_0。这些干净样本随后受到不同幅度的加性高斯噪声扰动，以生成扩散视频 x_t。这些扰动样本随后被送入真实分数模型 μ_{real} 和虚假分数模型 μ_{fake}，它们分别估计真实视频分布与生成视频分布的分数，定义为$s_{\text{real}}x_t, t = \nabla_x \log\ p_{\text{real}}x_t, t$, $s_{\text{fake}}x_t, t = \nabla_x \log\ p_{\text{fake}}x_t, t$。

最后，我们的学生 生成器模型 G_θ 可以通过以下分布匹配梯度 t:

$$\begin{aligned}\nabla\mathcal{L}_{\text{DMD}} &= \mathbb{E}_t\left(\nabla_\theta \,\mathrm{KL}\left(p_{\text{fake}} \| p_{\text{real}}\right)\right) \\ &= \mathop{\mathbb{E}}_{\epsilon\sim N0;I}\left[-\left(s_{\text{real}}\ (x_t, t) - s_{\text{fake}}\ (x_t, t)\right)\frac{dG_\theta}{d\theta}\right]\end{aligned} \tag{1}$$

训练期间，我们冻结真实分数模型 μ_{real} ，将其作为目标分布。此外，我们通过最小化标准扩散损失，动态更新虚假分数模型 μ_{fake} ，以跟踪学生生成器 G_θ生成的不断变化的样本分布。

$$\mathcal{L}_{\text{fake}} = \mathbb{E}\left[\|\mu_{\text{fake}}x_t, t - x_0\|_2^2\right] \tag{2}$$

其中， x_0 表示由 G_θ生成的虚假视频样本。

3.4 训练 FastAdapter

如图1(b)所示，直接在*SlowAdapter* 上使用*FastGenerator* 会导致视觉质量下降和轨迹准确度较低。因此，亟需一种简单有效的方法，将*SlowAdapter* 微调为*FastAdapter*。我们采用一种结合扩散目标和对抗目标的混合训练方案，使模型能够保持轨迹准确度并避免视觉质量下降（图2(c)）。

扩散损失 我们首先使用在阶段1训练的*SlowAdapter* 的参数初始化轨迹适配器的权重（详情见第3.2 节）。训练过程中，如图3 (a)所示，预训练的 3D VAE编码器 [40] 将轨迹图和真实视频映射到潜在空间，分别记为 z_{traj} 和x_0^{real}，并将其作为轨迹适配器和视频生成器的输入。每个适配器块生成的轨迹特征通过零初始化卷积层注入快速生成器的对应模块，从而引导合成（伪造）视频潜变量$x_0^{\text{fake}} = G_\theta x_t,\ t$。随后，我们使用标准扩散损失优化轨迹适配器：

$$\mathcal{L}_{diffusion} = \left\| G_\theta x_t, t - x_0^{\text{real}} \right\|_2^2 \tag{3}$$

对抗训练 然而，如图1(c)所示，仅使用扩散损失微调*SlowAdapter*通常会导致生成视频中出现明显的模糊伪影。由于扩散损失仅强制实现像素级对齐，因此会造成真实视频与生成视频的分布不匹配。为此，我们引入扩散判别器来弥合这种分布差异。

受APT [20], 启发，我们使用真实视频潜变量与伪造视频潜变量的扩散版本（记为 x_t^{fake}和x_t^{real}）作为扩散判别器的输入；该判别器被训练以产生能够有效区分真实视频与生成（伪造）视频的logit。我们使用 Wan2.2-TI2V-5B [40], 的权重初始化判别器骨干网络，并将基于注意力的分类器纳入扩散Transformer以产生logit。为提高内存效率并加快收敛，我们冻结扩散判别器的骨干网络，仅训练新添加的分类器。

如图3 (b)所示，分类器被附加到原始DiT骨干网络的选定层上。每个分类器包含一个基于注意力的头部，其后接一个输出单个token的MLP层。随后，所有分类器的token被拼接，并经过另一个MLP层以产生最终 logit，用于指示输入视频是真实还是伪造。

具体而言，如图3 (b)所示，每个分类器块通过三个连续的注意力层处理一个可学习查询token。语义自注意力层融合首帧图像与文本信息，以增强语义表示。在该层中，可学习查询token q 与首帧图像嵌入 e_i 和文本嵌入 e_{text}拼接，然后通过自注意力操作进行处理，使查询token能够跨多个语义模态进行注意力计算。随后，所得token被传递到轨迹交叉注意力层，在该层中它作为查询，并对轨迹图tokene_{traj}进行注意力计算，这些token在注意力计算中用作键和值 [39]。最后，该token由视频交叉注意力层处理，并对视频token e_{video}进行注意力计算。每个注意力层之后都对可学习token施加残差连接，为清晰起见，该连接在图3 (b)中省略。

因此，我们采用以下损失函数，以交替方案微调轨迹适配器和扩散判别器。

$$\mathcal{L}_{\mathcal{G}} = \min_{\theta} \mathbb{E}_{t\sim 0,T}\left[f\left(-\mathcal{D}_\phi\left(x_t^{\text{fake}}, t \right) \right) \right] \tag{4}$$

$$\mathcal{L}_{\mathcal{D}} = \min_{\phi} \mathbb{E}_{t\sim 0,T}[f\left(-\mathcal{D}_\phi\left(x_t^{\text{real}}, t \right) \right)\ f\left(\mathcal{D}_\phi\left(x_t^{\text{fake}}, t \right) \right)] \tag{5}$$

其中， f 表示 softplus函数 [3]， $T = 1000$， $\mathcal{D}\phi$ 表示扩散判别器， θ 和 ϕ 分别表示轨迹适配器和分类器的参数。

动态扩散损失缩放扩散损失约束生成视频在像素级遵循用户指定的轨迹，而 GAN 损失则弥合生成视频与真实视频之间的分布差异。因此，我们结合这两个目标函数对轨迹适配器进行联合训练，具体形式如下：

$$\mathcal{L} = \mathcal{L}_{\mathcal{G}} \ \lambda \mathcal{L}_{diffusion} \tag{6}$$

然而，我们观察到，在训练初期，扩散损失 $\mathcal{L}_{diffusion}$ 的梯度远大于 GAN 损失 $\mathcal{L}_{\mathcal{G}}$的梯度，直接将两者结合仍会导致结果模糊。为缓解这种不平衡，我们为系数 λ引入了一种动态加权方案，定义如下：

$$\lambda = \frac{1}{4} \times 10^{-3} \times step \ 0.1 \tag{7}$$

其中， $step$ 表示当前训练迭代。

3.5 FlashBench

以往的轨迹可控视频生成研究 [18, 19, 26, 34, 44, 49, 62, 65] 主要在 DAVIS [28], VIPSeg [25], 和 MagicBench [18]上进行评估。现有基准测试侧重于较短的视频序列，而 FlashMotion 能够生成长达 121 帧的视频。这种差异使得我们无法全面评估 FlashMotion 的长期时间一致性和轨迹可控性。因此，亟需一个面向长序列轨迹可控视频生成的公开基准测试。

遵循 MagicMotion [18], 中介绍的数据流程，我们通过扩展 MagicBench，构建了 FlashBench，并为所有帧添加了全面的轨迹标注。为便于详细分析，FlashBench 还根据前景物体的数量划分为六组：一至五个，以及五个以上。

4 实验

我们首先在第 4.1 节介绍实验设置，包括数据集、实现细节、评估指标和对比基线。随后，第4.2 节报告定量和定性结果，并与现有方法进行全面比较。最后，第 4.3 节通过消融研究进一步分析 FlashMotion 各个组件的贡献和有效性。

4.1 实验设置

数据集。我们使用 MagicData [18] 作为三个训练阶段的训练数据集，其中包含 2.3 万个高质量视频，并附有文本和轨迹标注，包括分割掩码和边界框。评估方面，我们在三个不同的基准测试上进行实验：FlashBench、MagicBench [18]和 DAVIS [28]。

实现细节。 在阶段 1 中，我们采用两种轨迹适配器架构：ResNet [10] 和 ControlNet [61]。ResNet 适配器从头开始训练，而 ControlNet 适配器则使用主 DiT 的权重进行初始化。两者首先使用分割掩码作为轨迹条件训练 4.6K 步，然后再使用边界框微调 5.4K 步。训练使用 16 张 A100 GPU，每张 GPU 的批量大小为 1，学习率为 2×10^{-6}。在阶段 2 中，*FastGenerator* 通过将 Wan2.2-TI2V-5B [40] 蒸馏为一个四步图生视频生成器而获得。所有参数均在 16 张 A100 GPU 上微调 5.5K 步，每张 GPU 的批量大小为 1。训练期间，生成器和虚假分数模型分别以 5×10^{-7} 和 1×10^{-7}的学习率进行优化，更新比例为 1:5。在阶段 3 中，轨迹适配器和判别器也以 2×10^{-6} 的学习率进行优化，更新比例为 1:5。扩散损失的权重根据 $\lambda = \frac{1}{4} \times 10^{-3} \times$ *step* 0.1逐渐增加，其中*step* 表示当前训练迭代。本阶段使用 4 张 A100 GPU 训练 1000步，每张 GPU 的批量大小为 1。

表 1 在 **FlashBench**、**MagicBench** 和 **DAVIS** 上的定量结果。我们报告 ResNet 和 ControlNet 适配器的 FID、FVD 和掩码/框交并比（%）。每项指标的最佳结果以**粗体**突出显示，次佳结果加下划线。去噪时间按在一块 A100 GPU 上生成 121 帧所需的时间计算。

方法	FlashBench			MagicBench			DAVIS			去噪时间（秒）	参数 (B)
	FID(↓)	FVD(↓)	M/B 交并比(↑)	FID(↓)	FVD(↓)	M/B 交并比(↑)	FID(↓)	FVD(↓)	M/B交并比(↑)		
MultiSteps（50 步）											
MagicMotion [18]	20.03	138.83	68.10/73.68	15.17	107.21	76.61/81.45	50.36	760.95	53.94/72.84	1158.63	11.53
Wan2.2（ResNet） [40]	19.03	139.61	52.19/57.76	21.72	140.41	62.09/67.85	46.44	703.15	31.22/42.74	333.00	5.02
Wan2.2（ControlNet） [40]	16.93	152.04	65.41/71.28	20.05	157.98	72.80/78.46	**43.70**	791.80	52.76/71.20	664.53	10.28
DragAnything [49]	34.93	267.56	58.54/61.72	31.36	253.40	66.30/70.85	70.70	1166.22	40.13/53.60	589.07	2.21
SG-I2V [26]	28.52	252.49	50.20/55.72	32.60	168.82	68.78/74.39	90.93	1170.60	37.36/50.96	1277.15	1.52
Tora [62]	31.79	315.11	48.17/53.70	26.27	245.23	58.95/64.03	51.75	766.76	37.98/50.90	691.13	6.32
LeviTor [41]	64.58	335.47	36.36/39.81	38.32	194.53	39.96/46.36	97.98	922.68	25.24/31.42	80.08	2.21
FewSteps（4 步）— 适配器：ResNet											
DMD [58]	24.38	228.33	43.24/52.61	25.27	206.57	49.69/59.44	51.75	1058.35	33.08/49.78	11.72	5.02
生成对抗网络 [6]	31.32	208.06	43.78/49.99	33.31	209.93	56.60/63.10	66.31	1143.14	30.49/42.80	11.72	5.02
潜在一致性模型 [22]	26.79	462.09	55.31/60.80	28.24	398.06	64.98/70.83	63.07	1075.61	42.56/58.52	11.72	5.02
FlashMotion	15.81	108.96	63.96/70.01	14.16	109.20	72.34/77.92	50.58	786.42	46.74/64.00	11.72	5.02
FewSteps（4 步）— 适配器：ControlNet											
DMD [58]/ 生成对抗网络 [6]	OOM										–
潜在一致性模型 [22]	28.34	340.29	61.29/64.83	25.87	261.87	70.55/74.57	62.25	1164.75	45.94/61.27	24.44	10.28
FlashMotion	**14.35**	**96.08**	**69.15/75.38**	**12.49**	**99.30**	**76.92/82.17**	45.66	**690.13**	**54.54/74.37**	24.44	10.28

评估指标。 评估时，我们遵循以往的研究 [18, 41, 45, 49, 65]，采用 FID [11] 和 FVD [30]来衡量视觉质量。此外，我们遵循 MagicMotion [18]，并采用 Mask_IoU 和 Box_IoU 来量化轨迹准确度。

对比基线。 我们将 FlashMotion 与多种最先进的轨迹可控视频生成方法进行比较，包括 MagicMotion [18], Tora [62], DragAnything [49], SGI2V [26], LeviTor [41],以及搭配*SlowAdapter*的 Wan2.2-TI2V-5B [40]。由于现有方法均不支持少步轨迹可控视频生成，我们基于现有视频蒸馏方法 [6, 22, 58] 设计了若干对比基线。在这些方法中，我们将*SlowAdapter*与*SlowGenerator*组合而成的模型定义为教师模型，而学生模型则由适配器与*FastGenerator*配对组成。由于 DMD [58] 和生成对抗网络（GAN） [6] 在 ControlNet 架构下会出现 CUDA 显存溢出错误，因此我们仅报告它们使用 ResNet 时的结果。

4.2 与其他方法的比较

定量比较

我们在 FlashBench、MagicBench [18], 和 DAVIS [28], 上将 FlashMotion 与现有方法进行比较，同时评估视觉质量和轨迹准确度。在 FlashBench 中，我们将每个视频的前 121 帧作为真实标注。由于若干先前方法 [18, 41, 49, 62] 无法生成这么长的视频，我们从这 121 帧中均匀采样 N 帧，其中 N 对应于各方法支持的最大视频长度。在 MagicBench [18] 和 DAVIS [28], 上，我们遵循 MagicMotion 的做法，使用每个生成视频的前 49 帧进行评估 [18]。如表 1所示，在不同适配器架构下，FlashMotion 在视觉质量和轨迹准确度方面均优于所有现有少步蒸馏方法 [6, 22, 58]。当采用 ControlNet 作为适配器时，FlashMotion 在保持少步采样效率的同时，进一步超越了所有先前的多步基线方法，速度较此前最先进水平提升 47× 倍 [18]。

定性比较

定性比较结果见图 4，其中展示了相应的输入图像、提示词和轨迹。我们展示了所有少步基线方法以及四种具有代表性的基于 DiT 的多步基线方法：MagicMotion[18], Tora [62] 和 Wan [40] + *SlowGenerator*。如图 4所示，FlashMotion



图4 定性比较结果。FlashMotion展现出更优的定性表现，优于以往的多步轨迹可控方法和少步蒸馏基线。

在视觉质量和轨迹准确度方面均优于所有这些方法。

4.3 消融研究

由于篇幅有限，正文中仅展示了在 FlashBench 上的消融结果；更多关于 Mag- icBench [18] 和 DAVIS [28] 的结果，请参阅补充材料。为确保公平比较，所有实验均采用与 FlashMotion Stage3 相同的训练配置。

快速适配器。 为验证 *FastAdapter* 的必要性，我们计算了直接将*SlowAdapter* 应用于 *FastGenerator* 的定量性能。如表 2 所示，移除 *FastAdapter* 训练阶段会导致视觉质量和轨迹准确度显著下降。图 5 中的结果还表明，移除该训练阶段会导致视频出现严重的色偏。这表明 *SlowAdapter* 无法直接控制 *FastGenerator* 的生成过程，凸显了 *FastAdapter* 训练阶段的必要性。

表2 关于*FastAdapter* 训练阶段、扩散损失、GAN 损失及动态损失缩放策略的消融研究。

方法	FID↓	FVD↓	M IoU↑	边界框交并比↑
适配器类型：*ResNet*				
慢速适配器	22.75	168.46	49.79	56.62
不使用扩散损失	18.87	161.07	52.04	58.04
不使用 GAN 损失	22.74	206.75	**65.82**	**70.60**
不使用动态尺度	26.32	210.93	65.54	69.77
FlashMotion	**15.81**	**108.96**	63.96	70.01
适配器类型：*ControlNet*				
慢速适配器	19.44	171.83	62.72	69.38
不使用扩散损失	21.21	172.04	55.91	61.59
不使用 GAN 损失	28.82	265.46	**71.56**	75.48
不使用动态尺度	19.93	155.55	70.46	**75.89**
FlashMotion	**14.35**	**96.08**	69.15	75.38

扩散损失。 我们通过在训练过程中移除扩散损失，评估其影响。如表 2 和图 5所示，移除扩散损失后，生成视频的轨迹准确度显著降低，生成视频与用户提供的轨迹之间出现明显偏差。此外，移除扩散损失也会导致视觉质量下降。

GAN 损失。 我们对 GAN 损失进行了消融研究，如表2所示。移除对抗目标后，虽然轨迹准确性略有提升，但视觉

质量却骤降了近 90%，并引入了如图 5所示的严重模糊伪影。

动态扩散损失缩放。 我们进一步通过在训练期间将损失缩放固定为 1，评估动态扩散损失缩放策略的有效性。如表 2所示，禁用动态缩放机制会导致视觉质量明显下降，并再次产生显著的模糊伪影，如图 5所示。 **判别器架构**。 为验证判别器的设计，我们针对四种不同的判别器架构进行了实验。如表 3所示，仅使用 视频交叉注意力 层时，视觉质量和轨迹准确度最差。相比之下，加入语义自注意力模块可以提升模型的语义理解能力，从而提高生成视频的视觉质量；而轨迹交叉注意力 模块则能有效增强轨迹控制准确度。完整的判别器架构在所有指标上均取得最佳表现。

表 3 在**FlashBench**上对判别器架构进行的消融研究。VC 表示视频交叉注意力层，SS 表示语义自注意力层，TC 表示轨迹交叉注意力层。

方法	FID(↓)	FVD(↓)	M IoU(↑)	边界框交并比(↑)
适配器类型：ResNet				
仅 VC	16.76	110.83	62.07	67.76
语义自注意力+视频交叉注意力	16.31	109.02	62.54	68.05
轨迹交叉注意力+视频交叉注意力	16.64	110.01	62.99	69.36
FlashMotion	**15.81**	**108.96**	**63.96**	**70.01**
适配器类型：ControlNet				
仅 VC	15.56	115.72	63.04	71.73
语义自注意力+VC	15.37	99.24	65.84	72.35
轨迹交叉注意力+VC	15.70	101.06	68.78	73.85
FlashMotion	**14.35**	**96.08**	**69.15**	**75.38**



图5 关于*FastAdapter* 训练阶段、扩散损失、GAN 损失和动态损失缩放策略的消融研究。

5 结论

在这项工作中，我们提出了 FlashMotion，这是一种新颖的框架，通过三阶段训练范式实现少步轨迹可控视频生成。首先，我们在多步视频生成器上训练轨迹适配器，以实现精确的轨迹控制。接着，我们将多步生成器蒸馏为少步版本，以加速视频合成。最后，我们采用结合扩散与对抗目标的混合策略对轨迹适配器进行微调，使其与少步生成器对齐，从而实现少步轨迹可控视频生成。此外，我们还提出了 FlashBench，这是一个针对长序列轨迹可控视频生成的综合基准测试，用于评估视觉质量和轨迹准确度。大量实验表明，FlashMotion 不仅超越了现有的少步蒸馏方法，在视觉保真度和轨迹一致性方面也优于以往的多步轨迹可控视频生成模型。

致谢 本研究得到国家自然科学基金委员会（项目编号：62472098）和上海市科学技术委员会（项目编号：25511106100）的资助。

参考文献

[1] Andreas Blattmann、Tim Dockhorn、Sumith Kulal、Daniel Mendelevitch、Maciej Kilian、Dominik Lorenz、Yam Levi、Zion English、Vikram Voleti、Adam Letts 等。《稳定视频扩散：将潜视频扩散模型扩展至大型数据集》。arXiv 预印本 arXiv:2311.15127，2023年。[2] Jiaxiang Cheng、Bing Ma、Xuhua Ren、Hongyi Jin、Kai Yu、Peng Zhang、Wenyue Li、Yuan Zhou、Tianxiang Zheng 和 Qinglin Lu。《POSE：视频扩散模型的分阶段单步对抗均衡》。arXiv 预印本 arXiv:2508.21019，2025年。[3] Charles Dugas、Yoshua Bengio、François Bélisle、Claude Nadeau 和 René Garcia。《融入二阶函数知识以改进期权定价》。载于 T. Leen、T. Dietterich 和 V. Tresp 编，《神经信息处理系统进展》，2000年。[4] Xiao Fu、Xian Liu、Xintao Wang、Sida Peng、Menghan Xia、Xiaoyu Shi、Ziyang Yuan、Pengfei Wan、Di Zhang 和 Dahua Lin。《3dtrajmaster：掌握三维轨迹，实现视频生成中的多实体运动》。载于国际学习表征会议，2025年。[5] Daniel Geng、Charles Herrmann、Junhwa Hur、Forrester Cole、Serena Zhang、Tobias Pfaff、Tatiana Lopez-Guevara、Carl Doersch、Yusuf Aytar、Michael Rubinstein、Chen Sun、Oliver Wang、Andrew Owens 和 Deqing Sun。《运动提示：利用运动轨迹控制视频生成》。arXiv 预印本 arXiv:2412.02700，2024年。[6] Ian Goodfellow、Jean Pouget-Abadie、Mehdi Mirza、Bing Xu、David Warde-Farley、Sherjil Ozair、Aaron Courville 和 Yoshua Bengio。《生成对抗网络》。NeurIPS，2014年。[7] Zekai Gu、Rui Yan、Jiahao Lu、Peng Li、Zhiyang Dou、Chenyang Si、Zhen Dong、Qifeng Liu、Cheng Lin、Ziwei Liu、Wenping Wang 和 Yuan Liu。《作为着色器的扩散：用于灵活视频生成控制的三维感知视频扩散》。arXiv 预印本 arXiv:2501.03847，2025年。[8] Yoav HaCohen、Nisan Chiprut、Benny Brazowski、Daniel Shalem、Dudu Moshe、Eitan Richardson、Eran Levin、Guy Shiran、Nir Zabari、Ori Gordon、Poriya Panet、Sapir Weissbuch、Victor Kulikov、Yaki Bitterman、Zeev Melumian 和 Ofir Bibi。《Ltx-video：实时视频潜扩散》。arXiv 预印本 arXiv:2501.00103，2024年。[9] Kaiming He、Xiangyu Zhang、Shaoqing Ren 和 Jian Sun。《用于图像识别的深度残差学习》。载于 CVPR，2016年。[10] Kaiming He、Xiangyu Zhang、Shaoqing Ren 和 Jian Sun。《用于图像识别的深度残差学习》。载于 IEEE计算机视觉与模式识别会议论文集，2016年。[11] Martin Heusel、Hubert Ramsauer、Thomas Unterthiner、Bernhard Nessler 和 Sepp Hochreiter。《通过双时间尺度更新规则训练的生成对抗网络会收敛至局部纳什均衡》。NeurIPS，2017年。[12] Jonathan Ho、Ajay Jain 和 Pieter Abbeel。《去噪扩散概率模型》。NeurIPS，2020年。[13] Xun Huang、Zhengqi Li、Guande He、Mingyuan Zhou 和 Eli Shechtman。《自强制：弥合自回归视频扩散中的训练测试差距》。arXiv 预印本 arXiv:2506.08009，2025年。

[14] Yash Jain、Anshul Nasery、Vibhav Vineet 和 Harkirat Behl。《Peekaboo：通过掩码扩散实现交互式视频生成》。arXiv 预印本 arXiv:2312.07509，2023年。[15] Diederik P Kingma 和 Max Welling。《自编码变分贝叶斯》。arXiv 预印本 arXiv:1312.6114，2013年。[16] Weijie Kong、Qi Tian、Zijian Zhang、Rox Min、Zuozhuo Dai、Jin Zhou、Jiangfeng Xiong、Xin Li、Bo Wu、Jianwei Zhang 等。《混元视频：大型视频生成模型的系统化框架》。arXiv 预印本 arXiv:2412.03603，2024年。[17] Jiachen Li、Weixi Feng、Tsu-Jui Fu、Xinyi Wang、Sugato Basu、Wenhu Chen 和 William Yang Wang。《T2V-Turbo：通过混合奖励反馈突破视频一致性模型的质量瓶颈》。神经信息处理系统进展，2024年。[18] Quanhao Li、Zhen Xing、Rui Wang、Hui Zhang、Qi Dai 和 Zuxuan Wu。《MagicMotion：通过稠密到稀疏轨迹引导实现可控视频生成》。载于 ICCV，2025年。[19] Yaowei Li、Xintao Wang、Zhaoyang Zhang、Zhouxia Wang、Ziyang Yuan、Liangbin Xie、Ying Shan 和 Yuexian Zou。《Image Conductor：交互式视频合成的精确控制》。载于 AAAI，2025年。[20] Shanchuan Lin、Xin Xia、Yuxi Ren、Ceyuan Yang、Xuefeng Xiao 和 Lu Jiang。《单步视频生成的扩散对抗后训练》。arXiv 预印本 arXiv:2501.08316，2025年。[21] Shanchuan Lin、Ceyuan Yang、Hao He、Jianwen Jiang、Yuxi Ren、Xin Xia、Yang Zhao、Xuefeng Xiao 和 Lu Jiang。《面向实时交互式视频生成的自回归对抗后训练》。arXiv 预印本 arXiv:2506.09350，2025年。[22] Simian Luo、Yiqin Tan、Longbo Huang、Jian Li 和 Hang Zhao。《潜在一致性模型：通过少步推理合成高分辨率图像》。arXiv 预印本 arXiv:2310.04378，2023年。[23] Zhengyao Lv、Chenyang Si、Tianlin Pan、Zhaoxi Chen、Kwan-Yee K Wong、Yu Qiao 和 Ziwei Liu。《DCM：用于高效高质量视频生成的双专家一致性模型》。arXiv 预印本 arXiv:2506.03123，2025年。[24] Wan-Duo Kurt Ma、John P Lewis 和 W Bastiaan Kleijn。《Trailblazer：基于扩散的视频生成的轨迹控制》。载于2024年亚洲计算机图形学国际会议论文集，2024年。[25] Jiaxu Miao、Yunchao Wei、Yu Wu、Chen Liang、Guangrui Li 和 Yi Yang。《VSPW：用于野外视频场景解析的大规模数据集》。载于 CVPR，2021年。[26] Koichi Namekata、Sherwin Bahmani、Ziyi Wu、Yash Kant、Igor Gilitschenski 和 David B. Lindell。《SG-I2V：图像到视频生成中的自引导轨迹控制》。载于国际学习表征会议，2025年。网址 https://openreview.net/forum?id=uQjySppU9x。[27] William Peebles 和 Saining Xie。《基于 Transformer 的可扩展扩散模型》。arXiv 预印本 arXiv:2212.09748，2022年。[28] F. Perazzi、J. Pont-Tuset、B. McWilliams、L. Van Gool、M. Gross 和 A. Sorkine-Hornung。《用于视频目标分割的基准数据集与评估方法》。载于 CVPR，2016年。[29] Haonan Qiu、Zhaoxi Chen、Zhouxia Wang、Yingqing He、Menghan Xia 和 Ziwei Liu。《FreeTraj：视频扩散模型中的免调优轨迹控制》。arXiv 预印本 arXiv:2406.16863，2024年。[30] René Ranftl、Katrin Lasinger、David Hafner、Konrad Schindler 和 Vladlen Koltun。《迈向稳健的单目深度估计：混合数据集实现零样本跨数据集迁移》。IEEE模式分析与机器智能汇刊，2020年。[31] Axel Sauer、Frederic Boesel、Tim Dockhorn、Andreas Blattmann、Patrick Esser 和 Robin Rombach。《利用潜在对抗扩散蒸馏实现快速高分辨率图像合成》。载于2024年亚洲计算机图形学国际会议论文集，2024年。[32] Axel Sauer、Dominik Lorenz、Andreas Blattmann 和 Robin Rombach。《对抗扩散蒸馏》。载于欧洲计算机视觉会议，2024年。[33] Shitong Shao、Hongwei Yi、Hanzhong Guo、Tian Ye、Daquan Zhou、Michael Lingelbach、Zhiqiang Xu 和 Zeke Xie。《MagicDistillation：面向大规模少步合成的弱到强视频蒸馏》。arXiv 预印本 arXiv:2503.13319，2025年。

[34] Xiaoyu Shi、Zhaoyang Huang、Fu-Yun Wang、Weikang Bian、Dasong Li、Yi Zhang、Manyuan Zhang、Ka Chun Cheung、Simon See、Hongwei Qin 等。Motion-i2v：通过显式运动建模实现一致且可控的图像到视频生成。SIGGRAPH，2024年。[35] Jiaming Song、Chenlin Meng 和 Stefano Ermon。去噪扩散隐式模型。arXiv预印本 arXiv:2010.02502，2020年。[36] Yang Song、Jascha Sohl-Dickstein、Diederik P Kingma、Abhishek Kumar、Stefano Ermon 和 Ben Poole。通过随机微分方程实现基于得分的生成建模。arXiv预印本 arXiv:2011.13456，2020年。[37] Yang Song、Prafulla Dhariwal、Mark Chen 和 Ilya Sutskever。一致性模型。arXiv预印本 arXiv:2303.01469，2023年。[38] Yanxiao Sun、Jiafu Wu、Yun Cao、Chengming Xu、Yabiao Wang、Weijian Cao、Donghao Luo、Chengjie Wang 和 Yanwei Fu。Swiftvideo：通过轨迹分布对齐实现少步视频生成的统一框架。arXiv预印本 arXiv:2508.06082，2025年。[39] Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N Gomez、Łukasz Kaiser 和 Illia Polosukhin。Attention Is All You Need。NeurIPS，2017年。[40] Team Wan、Ang Wang、Baole Ai、Bin Wen、Chaojie Mao、Chen-Wei Xie、Di Chen、Feiwu Yu、Haiming Zhao、Jianxiao Yang、Jianyuan Zeng、Jiayu Wang、Jingfeng Zhang、Jingren Zhou、Jinkai Wang、Jixuan Chen、Kai Zhu、Kang Zhao、Keyu Yan、Lianghua Huang、Mengyang Feng、Ningyi Zhang、Pandeng Li、Pingyu Wu、Ruihang Chu、Ruili Feng、Shiwei Zhang、Siyang Sun、Tao Fang、Tianxing Wang、Tianyi Gui、Tingyu Weng、Tong Shen、Wei Lin、Wei Wang、Wei Wang、Wenmeng Zhou、Wente Wang、Wenting Shen、Wenyuan Yu、Xianzhong Shi、Xiaoming Huang、Xin Xu、Yan Kou、Yangyu Lv、Yifei Li、Yijing Liu、Yiming Wang、Yingya Zhang、Yitong Huang、Yong Li、You Wu、Yu Liu、Yulin Pan、Yun Zheng、Yuntao Hong、Yupeng Shi、Yutong Feng、Zeyinzi Jiang、Zhen Han、Zhi-Fan Wu 和 Ziyu Liu。Wan：开放且先进的大规模视频生成模型。arXiv预印本 arXiv:2503.20314，2025年。[41] Hanlin Wang、Hao Ouyang、Qiuyu Wang、Wen Wang、Ka Leong Cheng、Qifeng Chen、Yujun Shen 和 Limin Wang。LeviTor：面向三维轨迹的图像到视频合成。发表于 CVPR，第12490–12500页，2025年。[42] Qinghe Wang、Yawen Luo、Xiaoyu Shi、Xu Jia、Huchuan Lu、Tianfan Xue、Xintao Wang、Pengfei Wan、Di Zhang 和 Kun Gai。Cinemaster：用于电影级文本到视频生成的三维感知可控框架。发表于 SIGGRAPH，2025年。[43] Xiang Wang、Shiwei Zhang、Han Zhang、Yu Liu、Yingya Zhang、Changxin Gao 和 Nong Sang。VideoLCM：视频潜在一致性模型。arXiv预印本 arXiv:2312.09109，2023年。[44] Zhouxia Wang、Yushi Lan、Shangchen Zhou 和 Chen Change Loy。ObjCtrl-2.5D：无需训练、利用相机姿态控制物体。发表于 arXiv预印本 arXiv:2412.07721，2024年。[45] Zhouxia Wang、Ziyang Yuan、Xintao Wang、Yaowei Li、Tianshui Chen、Menghan Xia、Ping Luo 和 Ying Shan。Motionctrl：用于视频生成的统一灵活运动控制器。发表于 SIGGRAPH，2024年。[46] Yujie Wei、Shiwei Zhang、Zhiwu Qing、Hangjie Yuan、Zhiheng Liu、Yu Liu、Yingya Zhang、Jingren Zhou 和 Hongming Shan。Dreamvideo：通过定制主体与运动，创作梦想视频。发表于 IEEE/CVF计算机视觉与模式识别会议论文集，2024年。[47] Yujie Wei、Shiwei Zhang、Hangjie Yuan、Xiang Wang、Haonan Qiu、Rui Zhao、Yutong Feng、Feng Liu、Zhizhong Huang、Jiaxin Ye 等。Dreamvideo-2：通过精确运动控制实现零样本主体驱动视频定制。arXiv预印本 arXiv:2410.13830，2024年。[48] Yujie Wei、Shiwei Zhang、Hangjie Yuan、Biao Gong、Longxiang Tang、Xiang Wang、Haonan Qiu、Hengjia Li、Shuai Tan、Yingya Zhang 等。Dreamrelation：关系中心视频定制。发表于 IEEE/CVF国际计算机视觉会议论文集，2025年。[49] Weijia Wu、Zhuang Li、Yuchao Gu、Rui Zhao、Yefei He、David Junhao Zhang、Mike Zheng Shou、Yan Li、Tingting Gao 和 Di Zhang。DragAnything：利用实体表示实现任意对象的运动控制。发表于 ECCV。Springer，2024年。[50] Zhen Xing、Qi Dai、Zihao Zhang、Hui Zhang、Han Hu、Zuxuan Wu 和 Yu-Gang Jiang。Vidiff：利用扩散模型通过多模态指令转换视频。arXiv预印本 arXiv:2311.18837，2023年。

[51] Zhen Xing、Qi Dai、Han Hu、Zuxuan Wu 和 Yu-Gang Jiang。Simda：用于高效视频生成的简易扩散适配器。发表于 CVPR，2024年。[52] Zhen Xing、Qi Dai、Zejia Weng、Zuxuan Wu 和 Yu-Gang Jiang。Aid：将图生视频扩散模型适配于指令引导视频预测。arXiv预印本 arXiv:2406.06465，2024年。[53] Zhen Xing、Qijun Feng、Haoran Chen、Qi Dai、Han Hu、Hang Xu、Zuxuan Wu 和 Yu-Gang Jiang。视频扩散模型综述。ACM计算调查，2024年。[54] Shiyuan Yang、Liang Hou、Haibin Huang、Chongyang Ma、Pengfei Wan、Di Zhang、Xiaodong Chen 和 Jing Liao。Direct-a-video：通过用户指定的相机运动和物体运动实现定制化视频生成。发表于 SIGGRAPH，2024年。[55] Zhuoyi Yang、Jiayan Teng、Wendi Zheng、Ming Ding、Shiyu Huang、Jiazheng Xu、Yuanming Yang、Wenyi Hong、Xiaohan Zhang、Guanyu Feng 等。CogVideoX：采用专家Transformer的文生视频扩散模型。arXiv预印本 arXiv:2408.06072，2024年。[56] Guy Yariv、Yuval Kirstain、Amit Zohar、Shelly Sheynin、Yaniv Taigman、Yossi Adi、Sagie Benaim 和 Adam Polyak。Through-the-mask：用于图像到视频生成的基于掩码的运动轨迹。发表于 CVPR，2025年。[57] Shengming Yin、Chenfei Wu、Jian Liang、Jie Shi、Houqiang Li、Gong Ming 和 Nan Duan。Dragnuwa：融合文本、图像和轨迹，实现视频生成的细粒度控制。arXiv预印本 arXiv:2308.08089，2023年。[58] Tianwei Yin、Michaël Gharbi、Taesung Park、Richard Zhang、Eli Shechtman、Fredo Durand 和 William T Freeman。改进用于快速图像合成的分布匹配蒸馏。发表于 NeurIPS，2024年。[59] Tianwei Yin、Michaël Gharbi、Richard Zhang、Eli Shechtman、Frédo Durand、William T Freeman 和 Taesung Park。通过分布匹配蒸馏实现单步扩散。发表于 CVPR，2024年。[60] Tianwei Yin、Qiang Zhang、Richard Zhang、William T Freeman、Fredo Durand、Eli Shechtman 和 Xun Huang。从缓慢的双向视频扩散模型到快速的自回归视频扩散模型。发表于 CVPR，2025年。[61] Lvmin Zhang、Anyi Rao 和 Maneesh Agrawala。为文生图扩散模型添加条件控制。发表于 ICCV，2023年。[62] Zhenghao Zhang、Junchao Liao、Menghao Li、Zuozhuo Dai、Bingxue Qiu、Siyu Zhu、Long Qin 和 Weizhi Wang。Tora：面向视频生成的轨迹导向扩散Transformer。发表于 CVPR，2025年。[63] Zhenghao Zhang、Junchao Liao、Xiangyu Meng、Long Qin 和 Weizhi Wang。Tora2：用于多实体视频生成的运动与外观定制化扩散Transformer。arXiv预印本 arXiv:2507.05963，2025年。[64] Zangwei Zheng、Xiangyu Peng、Tianji Yang、Chenhui Shen、Shenggui Li、Hongxin Liu、Yukun Zhou、Tianyi Li 和 Yang You。Open-sora：让所有人都能高效制作视频。arXiv预印本 arXiv:2412.20404，2024年。[65] Haitao Zhou、Chuang Wang、Rui Nie、Jinlin Liu、Dongdong Yu、Qian Yu 和 Changhu Wang。Trackgo：一种灵活高效的可控视频生成方法。发表于 AAAI，2025年。

附录

6 更多消融实验结果

6.1 定量结果

这里，我们提供了三个基准测试（包括 FlashBench、MagicBench [18], 和 DAVIS [28] ）上的完整定量结果，见表 4 和表 5。所有消融研究均在 4 块英伟达 A100 GPU 上训练 1000 步，其他训练配置与 FlashMotion 第三阶段保持一致。

表 4 FlashMotion 的全面消融研究。我们分析了两种适配器变体（ResNet 和 ControlNet），逐步移除关键组件，包括*FastAdapter*训练阶段、扩散损失、GAN 损失以及动态扩散损失缩放策略。结果表明，每个组件对于维持高视频质量和精准的运动对齐都至关重要。

方法	FlashBench			MagicBench			DAVIS		
	FID(↓)	FVD(↓)	M/B 交并比%(↑)	FID(↓)	FVD(↓)	M/B交并比%(↑)	FID(↓)	FVD(↓)	M/B交并比%(↑)
适配器类型：ResNet									
慢速适配器	22.75	168.46	49.79 / 56.62	21.59	162.93	60.24 / 67.23	52.01	992.26	36.33 / 51.37
不使用扩散损失	18.87	161.07	52.04 / 58.04	21.95	162.31	63.14 / 69.02	55.28	983.91	37.22 / 52.47
不含 GAN 损失	22.74	206.75	**65.82 / 70.60**	30.51	167.91	**73.86 / 78.48**	66.46	1015.81	**47.13**/ 62.58
不含动态尺度	26.32	210.93	65.54 / 69.77	21.90	167.00	73.60 / 78.15	73.12	998.85	47.01 / 60.12
FlashMotion	**15.81**	**108.96**	63.96 / 70.01	**14.16**	**109.20**	72.34 / 77.92	**50.58**	**786.42**	46.74 / **64.00**
适配器类型：ControlNet									
慢速适配器	19.44	171.83	62.72 / 69.38	21.19	161.80	70.20 / 76.54	46.42	875.37	50.52 / 70.83
不使用扩散损失	21.21	172.04	55.91 / 61.59	22.36	176.01	66.25 / 71.82	49.27	882.81	42.46 / 59.01
不使用 GAN 损失	28.82	265.46	**71.56** /75.48	26.33	192.85	**78.26**/ 82.15	75.42	1131.65	**55.87**/ 68.59
不使用动态尺度	19.93	155.55	70.46 / **75.89**	16.83	131.59	77.49 / **82.29**	61.47	958.22	55.51 / 70.13
FlashMotion	**14.35**	**96.08**	69.15 / 75.38	**12.49**	**99.30**	76.92 / 82.17	**45.66**	**690.13**	54.54 / **74.37**

表 5 判别器架构的消融研究。VC 表示视频交叉注意力层，SS 表示语义自注意力 层，TC 表示轨迹交叉注意力 层。结果表明，我们的判别器设计在所有基准测试和指标上均实现了最佳综合性能。

方法	FlashBench			MagicBench			DAVIS		
	FID(↓)	FVD(↓)	M/B IoU%(↑)	FID(↓)	FVD(↓)	M/B IoU%(↑)	FID(↓)	FVD(↓)	M/B IoU%(↑)
适配器类型：ResNet									
仅 VC	16.76	110.83	62.07 / 67.76	14.73	114.61	71.00 / 75.86	53.22	800.50	43.97 / 60.16
语义自注意力+视频交叉注意力	16.31	109.02	62.54 / 68.05	14.44	113.88	71.16 / 76.28	52.34	830.14	44.61 / 62.50
轨迹交叉注意力+视频交叉注意力	16.64	110.01	62.99 / 69.36	14.87	114.11	71.70 / 77.31	53.16	830.57	45.11 / 62.56
FlashMotion	**15.81**	**108.96**	**63.96 / 70.01**	**14.16**	**109.20**	**72.34 / 77.92**	**50.58**	**786.42**	**46.74 / 64.00**
适配器类型：ControlNet									
仅 VC	15.56	115.72	63.04 / 71.73	13.71	120.22	75.78 / 81.33	49.39	798.79	51.48 / 69.00
语义自注意力+视频交叉注意力	15.37	99.24	65.84 / 72.35	13.42	101.58	75.35 / 81.06	46.24	711.82	53.33 / 71.99
轨迹交叉注意力+视频交叉注意力	15.70	101.06	68.78 / 73.85	13.96	105.49	76.48 / 82.15	48.50	758.96	53.91 / 72.90
FlashMotion	**14.35**	**96.08**	**69.15 / 75.38**	**12.49**	**99.30**	**76.92 / 82.17**	**45.66**	**690.13**	**54.54 / 74.37**

快速适配器 为评估*FastAdapter* 训练阶段的重要性，我们评估直接将*SlowAdapter*应用于*FastGenerator* 在全部三个基准测试上的性能。如表4所示，去除

该*FastAdapter* 阶段会导致所有基准测试中视频质量与轨迹准确度持续下降，凸显了额外*FastAdapter* 训练阶段的必要性。

扩散损失 为评估扩散损失的作用，我们在训练期间将其移除，并在所有基准测试上测量模型性能。如表 4所示，对于两种适配器架构，移除扩散损失都会导致轨迹对齐程度明显下降。这表明，扩散损失对于维持生成动作与用户指定轨迹之间的轨迹一致性至关重要。此外，移除该损失还会降低图像和视频质量。

GAN 损失 我们对 GAN 损失进行了消融研究，结果总结于表 4中。移除对抗目标虽能略微提高轨迹准确度，却会使图像和视频质量均下降约 90%，并引入严重的模糊伪影。

动态扩散损失缩放 我们进一步通过在训练期间将损失缩放系数固定为 1，验证所提出的动态扩散损失缩放策略的有效性。如表 4所示，禁用动态缩放会导致所有三个基准测试中的图像和视频质量明显下降，并再次产生显著的模糊伪影。

判别器架构最后，我们评估了不同判别器架构的影响，如表5所示。仅使用视频交叉注意力层时，视觉质量和轨迹准确度均为最低。相比之下，引入语义自注意力模块能够增强模型的语义理解，提升视觉真实感；而轨迹交叉注意力模块则能够强化轨迹控制准确度。总体而言，我们完整的判别器架构在所有评估指标和基准测试中均取得了最佳结果。

6.2 更多定性结果

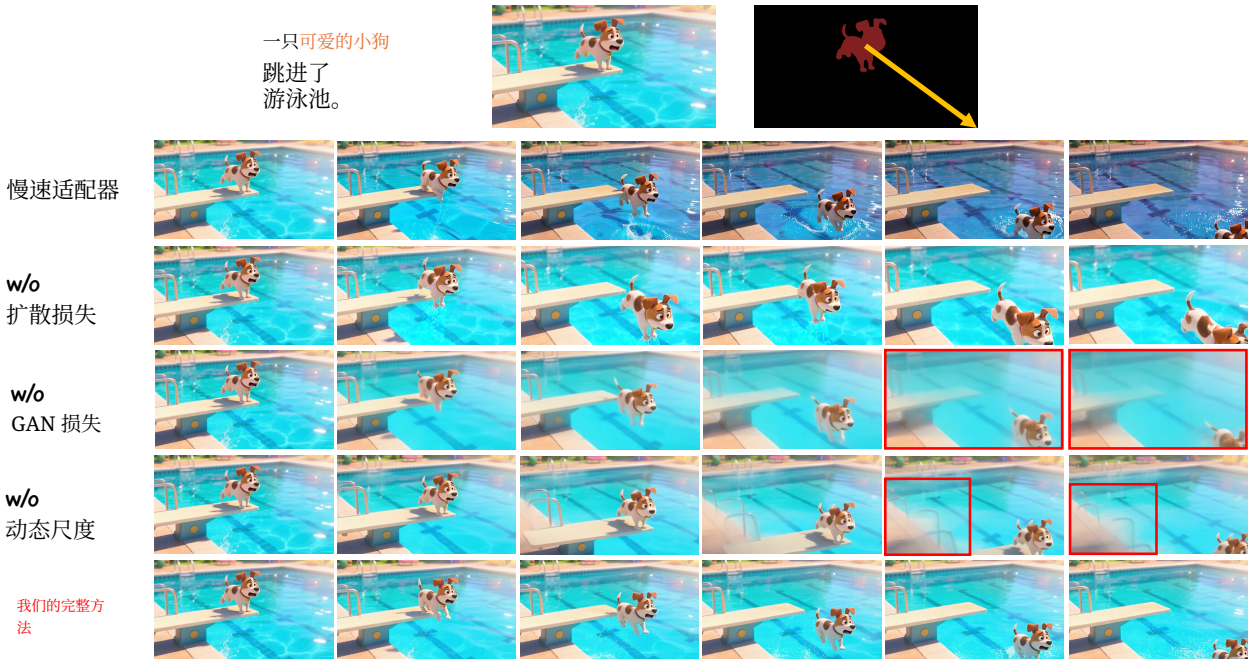

图 6 更多消融研究结果。只有我们的完整方法才能生成兼具高视觉质量和高轨迹准确度的视频。



图 7 更多消融研究结果。只有我们完整的方法才能生成兼具高视觉质量和轨迹准确度的视频。

详细的定性消融结果见图6、图7和图8。如图所示，直接将*SlowAdapter* 应用于*FastGenerator* 会产生明显的伪影——例如图6 和图8中的颜色漂移，以及图7中的物体形状扭曲。此外，在训练过程中移除扩散损失会显著降低轨迹保真度：物体（例如狗或公交车）会偏离预定路径；在图8所示的极端情况下，一个海绵宝宝还会被错误地复制成两个。最后，去除 GAN 损失或动态缩放策略中的任一项，都会引入严重的模糊伪影。

7 更多比较结果

7.1 骨干网络比较

如表6所示，我们全面比较了不同方法所采用的骨干架构。该表总结了各方法支持的视频长度和空间分辨率，以及相应的去噪延迟和参数总量。值得注意的是，FlashMotion 在基于 ControlNet 和基于 ResNet 的适配器中都实现了最快的去噪速度，同时支持最高分辨率和最长生成长度。用户可根据需求灵活选择 FlashMotion 的 ResNet 或 ControlNet 版本，在生成速度、视频质量和轨迹准确度之间取得平衡。

7.2 不同物体数量下的结果

受篇幅限制，正文仅报告了在 FlashBench 上的整体定量比较。此处，我们针对不同数量的受控物体进行详细评估，涵盖前景物体数量从 1–5 个到 5 个以上的情况。如表 7 和表 8所示，FlashMotion 的 ControlNet 变体在所有指标上始终优于所有竞争方法，在视觉质量和轨迹准确度方面均胜过多步和少步基线方法。使用基于 ResNet 的轨迹适配器时，FlashMotion 的视觉质量也优于先前最先进水平的方法 MagicMotion [18], ，不过由于参数容量有限，其轨迹准确度仍略逊一筹。



图 8 更多消融研究结果。只有我们的完整方法才能生成兼具高视觉质量和高轨迹准确度的视频。

7.3 更多定性结果

本节中，我们展示与先前方法的更多定性比较。如图 9–15所示，FlashMotion 能够准确控制物体轨迹并生成高质量视频，而其他方法则会产生明显的伪影和不一致性。完整视频结果请参阅补充材料中的“Supplementary video.mp4”。

8 案例研究

8.1 不同风格

如图 16所示，FlashMotion 支持生成具有多种视觉风格的视频，包括梦幻现实主义、超现实微缩摄影、三维卡通渲染和东方水墨画。为了更好地展示模型的鲁棒性，以及其在复杂布局中保持运动一致性的能力，我们特意选择纵向图像而非横向图像。这些示例共同展现了 FlashMotion 对不同艺术领域的强大适应能力，同时保持了结构与运动的连贯性。

8.2 相机控制

FlashMotion 支持拉近和拉远等相机控制操作。如图17所示，可以通过操控前景物体（如杯子或女性面具）的边界框大小来调整相机运动。此外，如图18所示，用户可以通过控制恐龙、猛犸象或工业搅拌机等物体的边界框，在面包店或博物馆等场景中移动。

表6 模型配置与骨干架构对比，包括支持的视频长度、空间分辨率、去噪延迟和总参数量。FlashMotion 在支持最高分辨率和最长生成时长的同时，实现了最快的去噪速度。

方法	视频长度	视频分辨率	去噪延迟（秒）	总参数量（十亿）	基础模型
LeviTor [41]	16	288×512	80.08	2.21	SVD [1]
DragAnything [49]	14	320×576	589.07	2.21	SVD [1]
SG-I2V [26]	14	576×1024	1277.15	1.52	SVD [1]
Tora [62]	49	480×720	691.13	6.32	CogVideoX [55]
MagicMotion [18]	49	480×720	1158.63	11.53	CogVideoX [55]
Wan+ResNet [40]	121	704×1280	333.00	5.02	Wan2.2 [40]
Wan+ControlNet [40]	121	704×1280	664.53	10.28	Wan2.2 [40]
FlashMotion (ResNet)	121	704×1280	11.72	5.02	Wan2.2 [40]
FlashMotion (ControlNet)	121	704×1280	24.44	10.28	Wan2.2 [40]

表7 包含1、2和3个受控物体的场景在FlashBench上的定量比较结果。详细评估表明，采用基于ControlNet的适配器的FlashMotion在所有指标上始终优于所有竞争方法，而基于ResNet的适配器在视觉质量方面也优于以往工作。

方法	物体_数量=1			物体_数量=2			物体_数量=3		
	FID(↓)	FVD(↓)	M/B 交并比(↑)	FID(↓)	FVD(↓)	M/B 交并比(↑)	FID(↓)	FVD(↓)	M/B IoU(↑)
MultiSteps（50 步）									
MagicMotion [18]	53.62	741.91	67.93/83.46	59.37	697.50	61.05/73.47	52.44	563.38	66.13/72.92
Wan2.2（ResNet） [40]	49.01	599.93	61.10/76.34	56.19	582.42	51.49/62.07	57.39	566.53	50.06/56.75
Wan2.2（ControlNet） [40]	50.04	594.54	66.07/83.98	51.20	591.56	59.64/73.18	49.49	547.90	62.64/70.01
DragAnything [49]	76.28	1076.20	62.70/74.88	91.08	1196.46	53.34/63.06	89.26	1099.45	54.01/57.55
SG-I2V [26]	70.20	984.94	64.09/76.45	78.93	926.79	47.16/57.04	73.08	891.52	48.31/54.25
Tora [62]	73.15	902.55	58.24/69.00	80.27	939.72	46.45/57.47	82.54	869.43	46.80/52.66
LeviTor [41]	128.25	1318.56	49.63/59.73	127.24	1124.07	38.09/44.82	131.60	1252.00	35.65/39.08
FewSteps（4 步）— 适配器：ResNet									
DMD [58]	64.71	709.74	55.34/74.30	63.28	687.09	45.21/59.62	64.03	636.34	43.08/53.14
生成对抗网络 [6]	79.73	728.35	54.58/66.52	77.25	700.88	41.38/51.34	74.52	673.58	41.46/48.80
潜在一致性模型 [22]	58.97	875.26	64.61/80.06	72.26	1032.56	56.40/68.58	65.52	1033.12	53.52/59.67
FlashMotion	46.64	509.36	68.02/**84.86**	51.21	497.62	60.27/73.08	44.41	433.60	63.40/71.59
FewSteps（4 步）— 适配器：ControlNet									
DMD [58]/ 生成对抗网络 [6]	OOM								
潜在一致性模型 [22]	61.13	851.48	62.83/76.15	76.41	929.77	56.68/66.79	69.65	831.79	57.86/63.51
FlashMotion	**44.97**	**465.86**	**68.44**/84.51	**46.16**	**437.18**	**63.87/76.99**	**42.20**	**422.16**	**66.45/73.91**

9 FlashBench 的更多详情

FlashBench 包含 600 个视频，依据前景物体数量分为六类（从 1–5 个到多于 5 个）。为了更全面地分析该数据集，我们进一步可视化了视频长度的分布，如图19所示，这表明它支持长视频生成评估。

表8 在包含4个、5个及超过5个受控物体的场景中，FlashBench上的定量比较结果。详细评估表明，采用基于ControlNet的适配器的FlashMotion在所有指标上始终优于所有竞争方法，而基于ResNet的适配器在视觉质量方面也优于以往工作。

方法	物体_数量=4			物体_数量=5			物体_数量>5		
	FID(↓)	FVD(↓)	M/B IoU(↑)	FID(↓)	FVD(↓)	M/B IoU(↑)	FID(↓)	FVD(↓)	M/B IoU(↑)
MultiSteps（50 步）									
MagicMotion [18]	45.67	546.40	70.29/73.21	44.41	450.10	73.86/76.93	44.41	409.25	69.29/62.35
Wan2.2（ResNet） [40]	61.69	575.65	50.89/53.93	52.04	476.04	55.56/56.98	41.59	453.60	44.31/41.03
Wan2.2（ControlNet） [40]	49.25	503.03	66.15/68.65	43.58	409.57	70.70/70.94	37.11	406.06	67.27/61.05
DragAnything [49]	75.00	997.03	59.97/60.23	83.35	812.67	62.92/61.48	97.48	1006.25	56.95/49.51
SG-I2V [26]	64.83	861.49	50.87/55.46	65.91	713.41	54.21/55.83	66.22	828.14	36.75/35.52
Tora [62]	65.25	737.03	46.28/51.05	73.88	714.65	52.76/54.55	93.60	1073.05	37.98/36.89
LeviTor [41]	167.97	1774.66	35.10/34.02	185.75	2015.57	33.33/30.34	135.75	1287.67	24.23/23.41
FewSteps（4 步）— 适配器：ResNet									
DMD [58]	66.08	749.99	41.38/48.44	67.03	697.32	42.02/47.94	52.62	671.65	32.74/32.74
生成对抗网络 [6]	69.55	571.76	45.87/50.43	65.83	500.22	48.67/52.49	59.83	584.86	31.00/30.78
潜在一致性模型 [22]	62.24	959.97	57.30/57.89	58.71	869.45	56.78/58.98	49.66	780.45	43.51/40.21
FlashMotion	38.47	411.71	66.58/67.87	39.53	326.98	68.67/79.78	37.07	384.06	56.92/52.02
FewSteps（4 步）— 适配器：ControlNet									
DMD [58]/ 生成对抗网络 [6]	OOM								
潜在一致性模型 [22]	60.28	752.48	63.18/63.38	56.29	637.06	66.48/65.66	53.64	541.55	60.79/53.62
FlashMotion	**36.62**	**367.24**	**71.81/75.49**	**35.19**	**294.47**	**74.94/76.98**	**32.72**	**305.68**	**69.43/64.50**



图9 不同方法的定性比较结果。

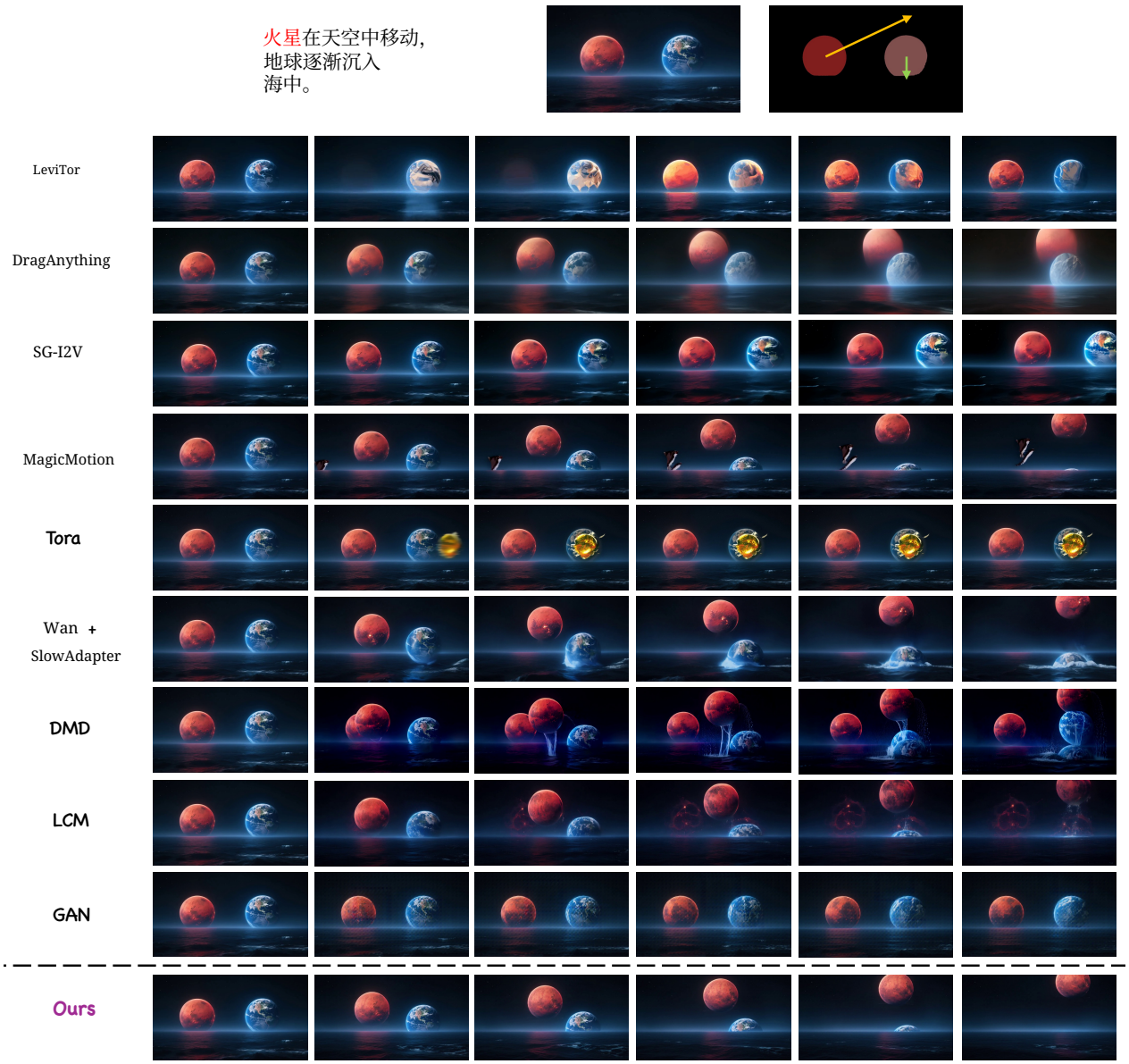

图 10 不同方法的定性比较结果。

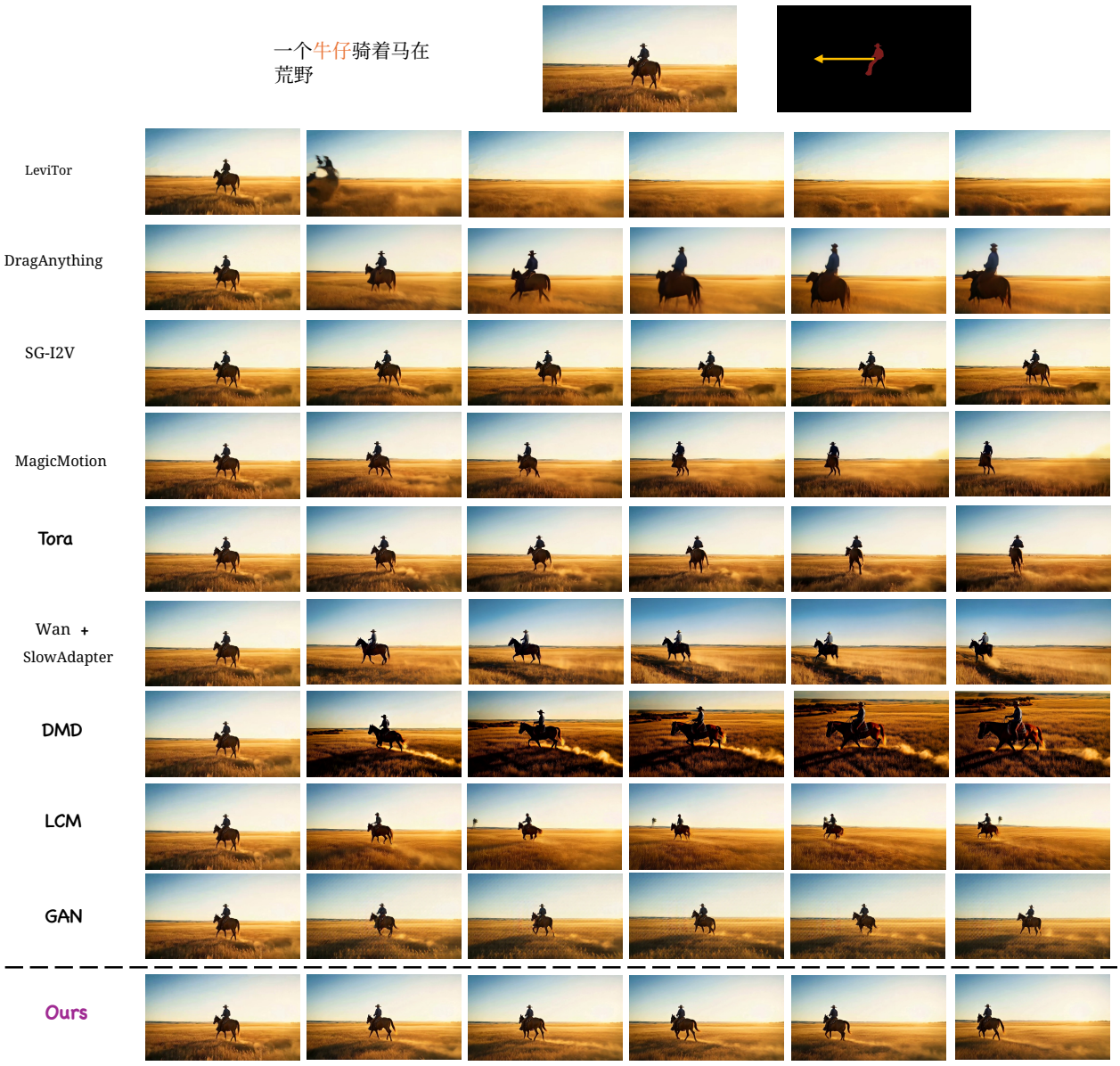

图11 不同方法的定性比较结果。

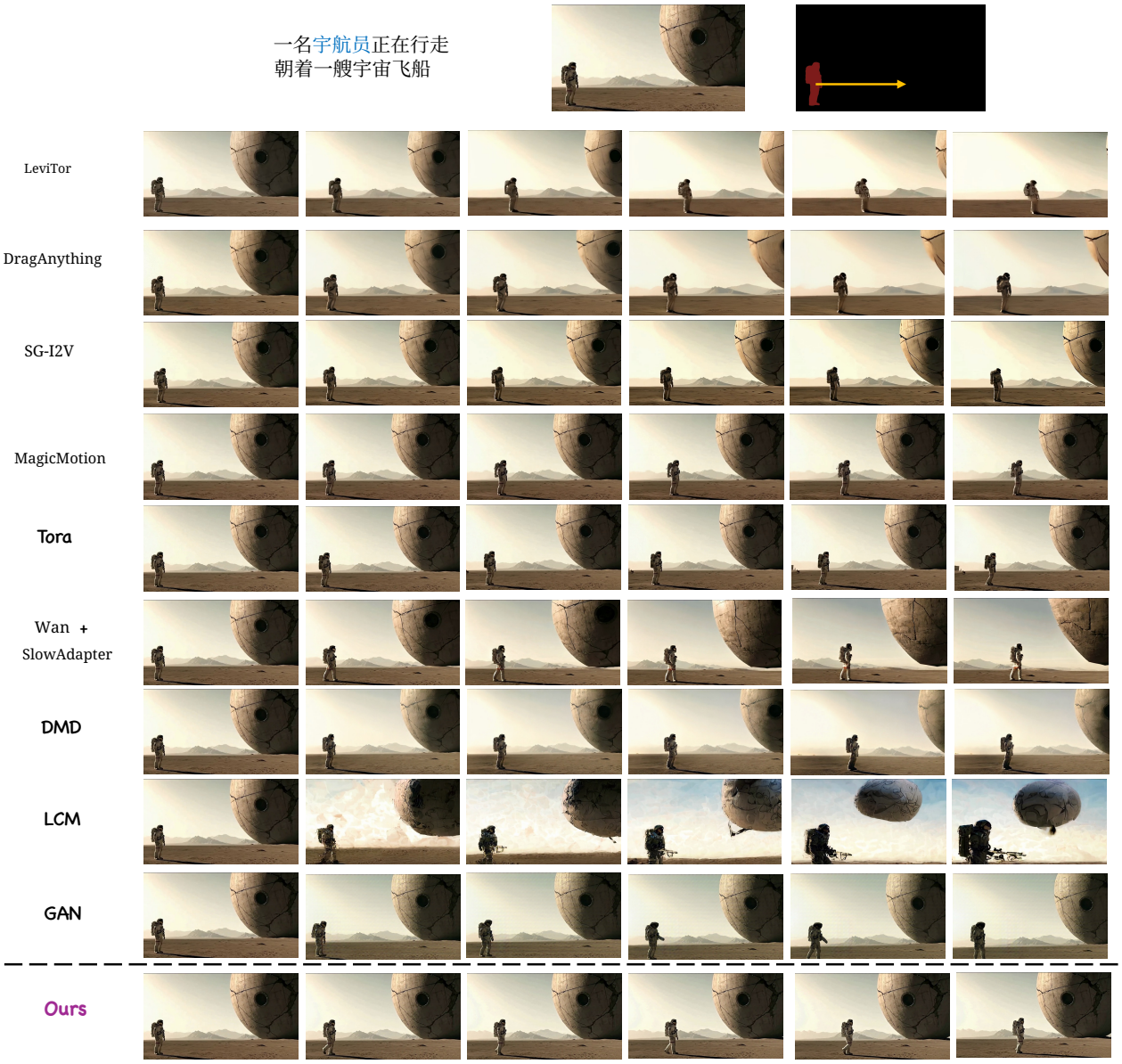

图12 不同方法的定性比较结果。

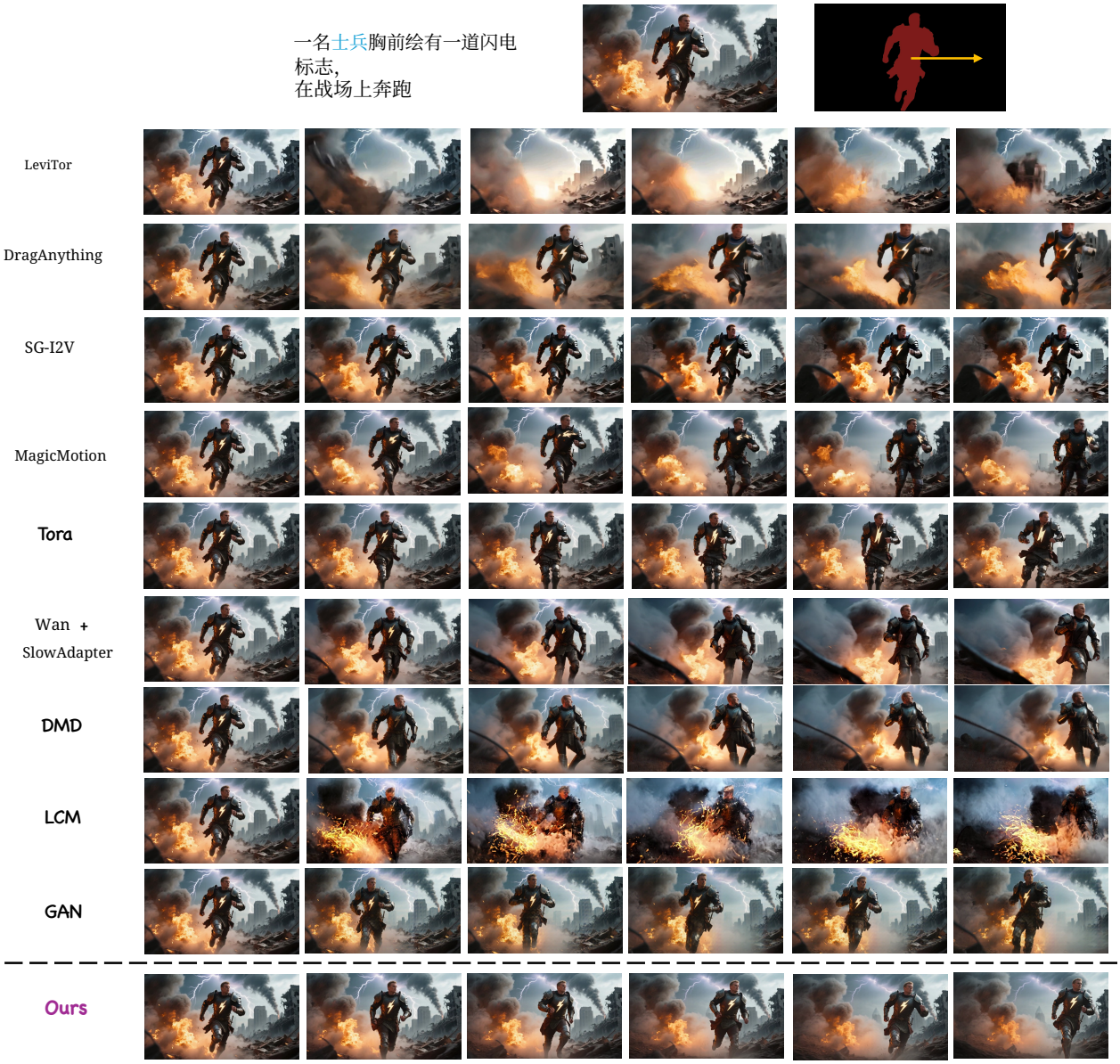

图13 不同方法的定性比较结果。

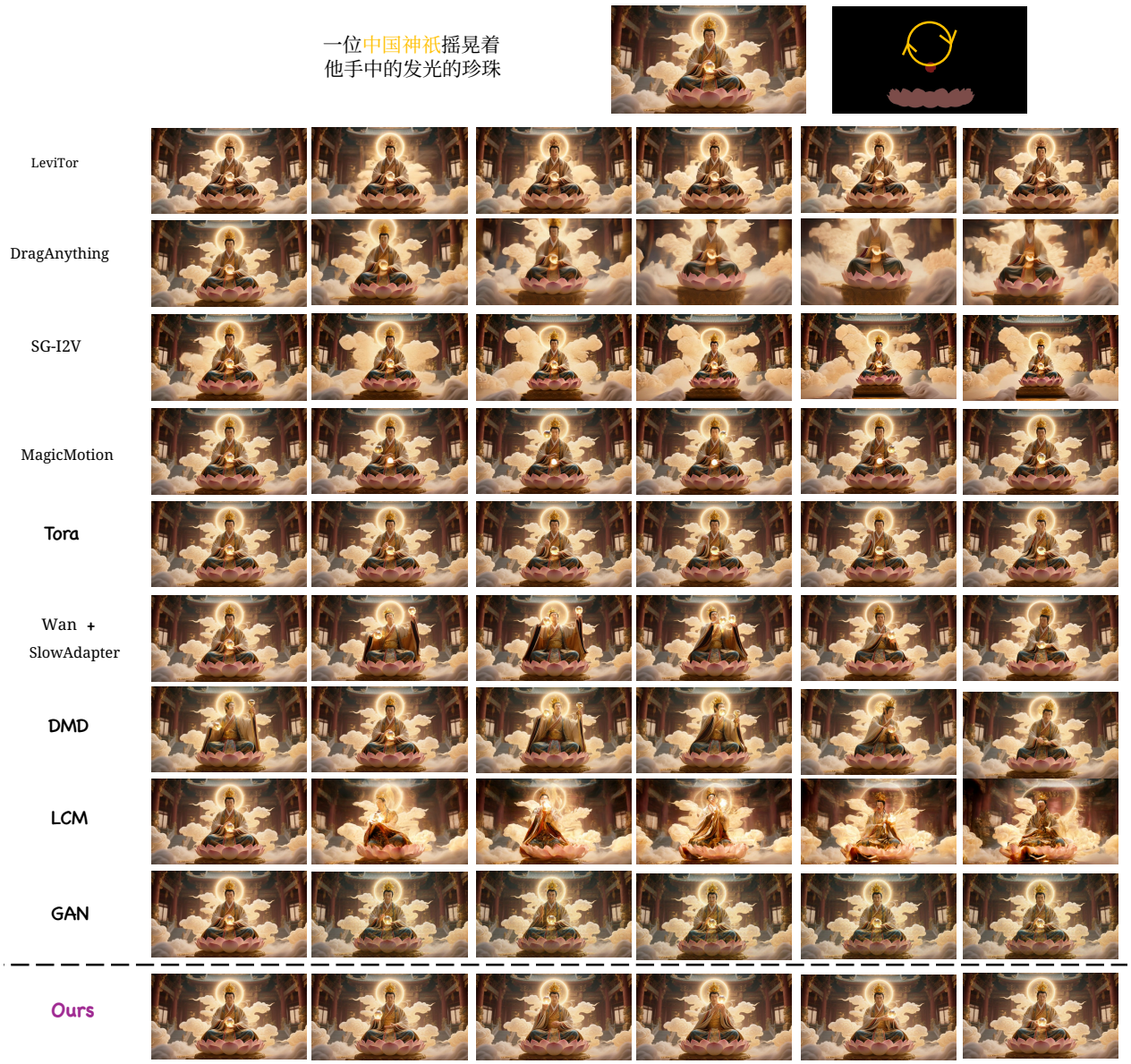

图14 不同方法的定性比较结果。

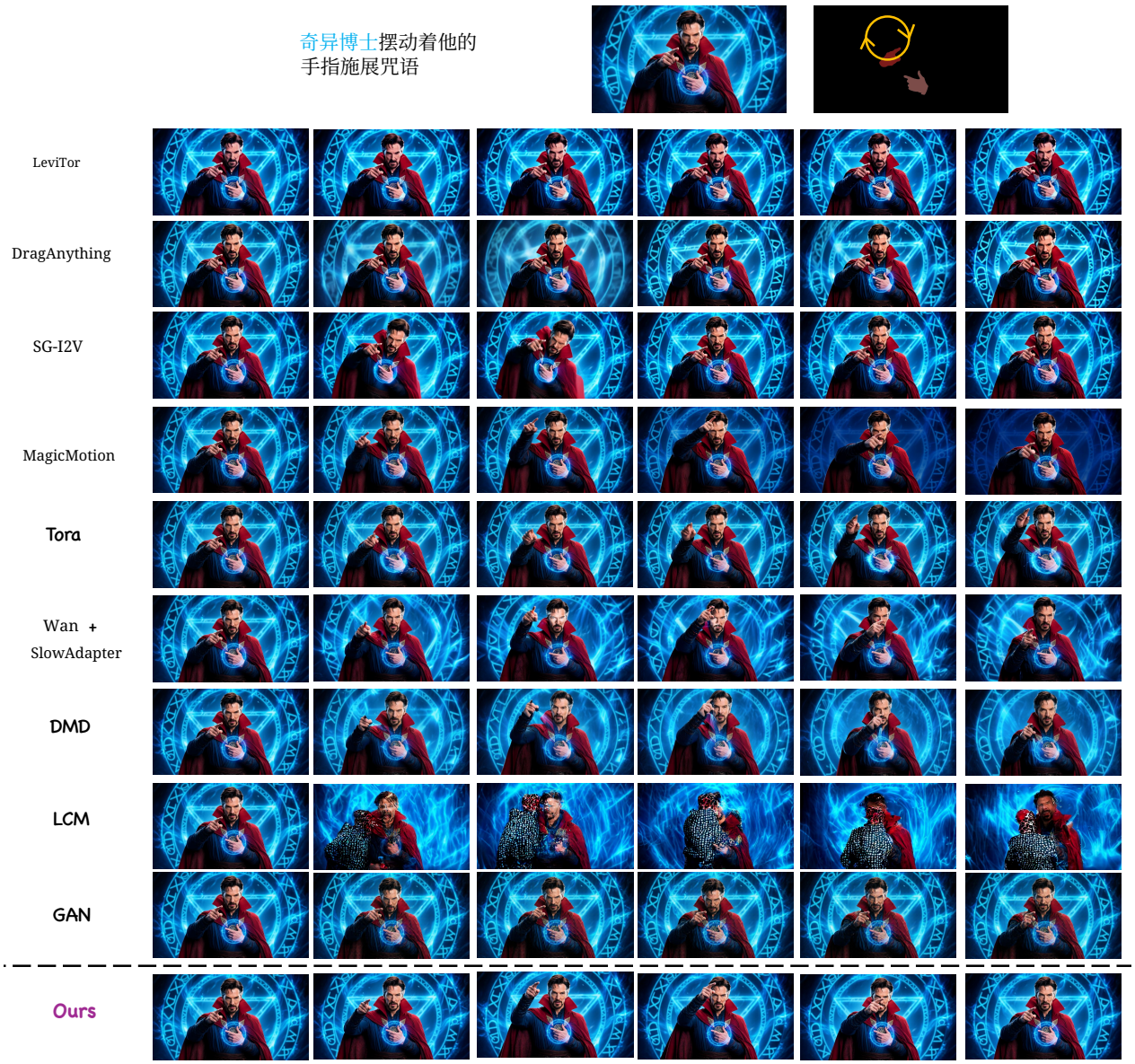

图15 不同方法的定性比较结果。

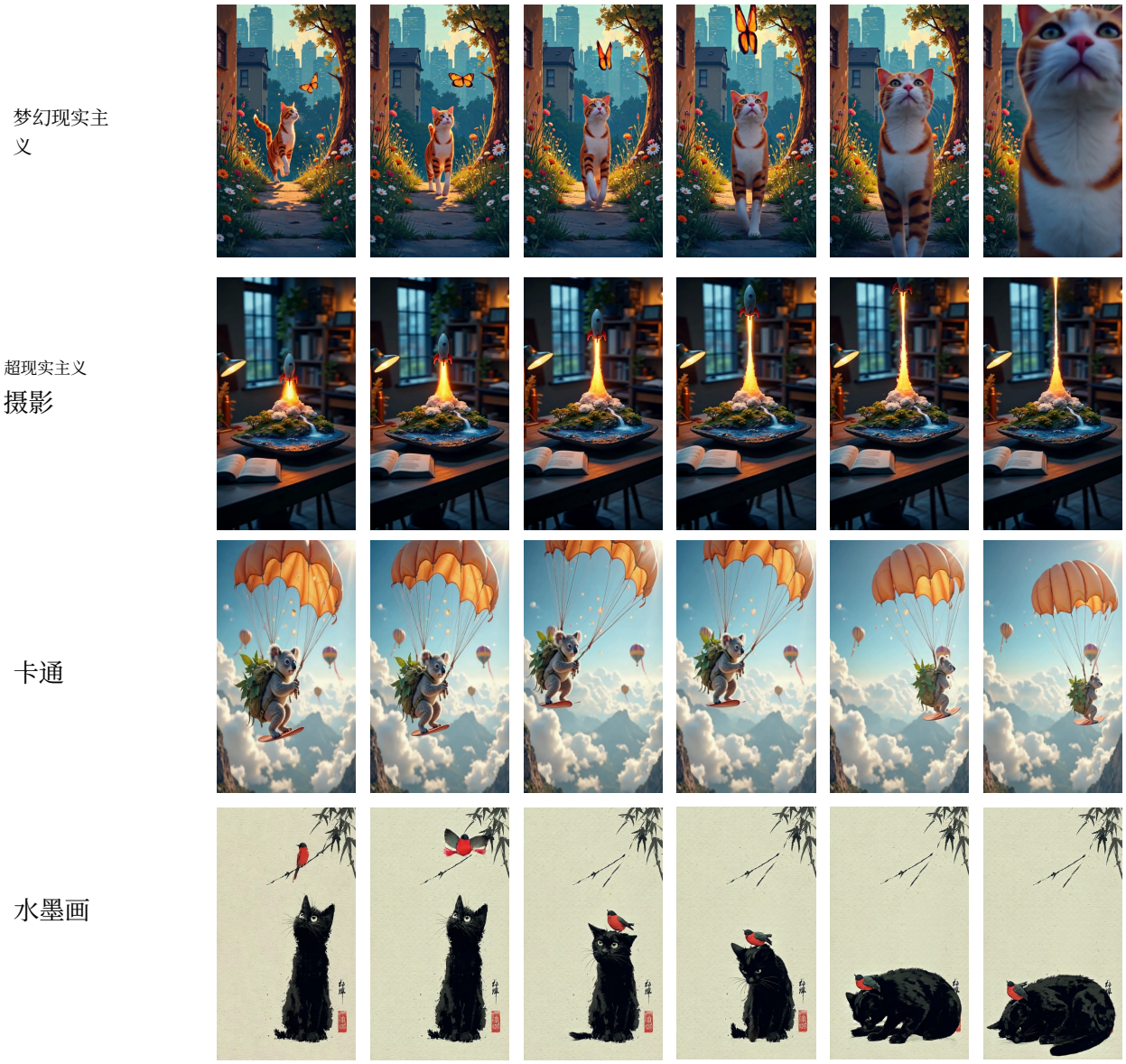

图16 FlashMotion支持生成不同风格的视频。

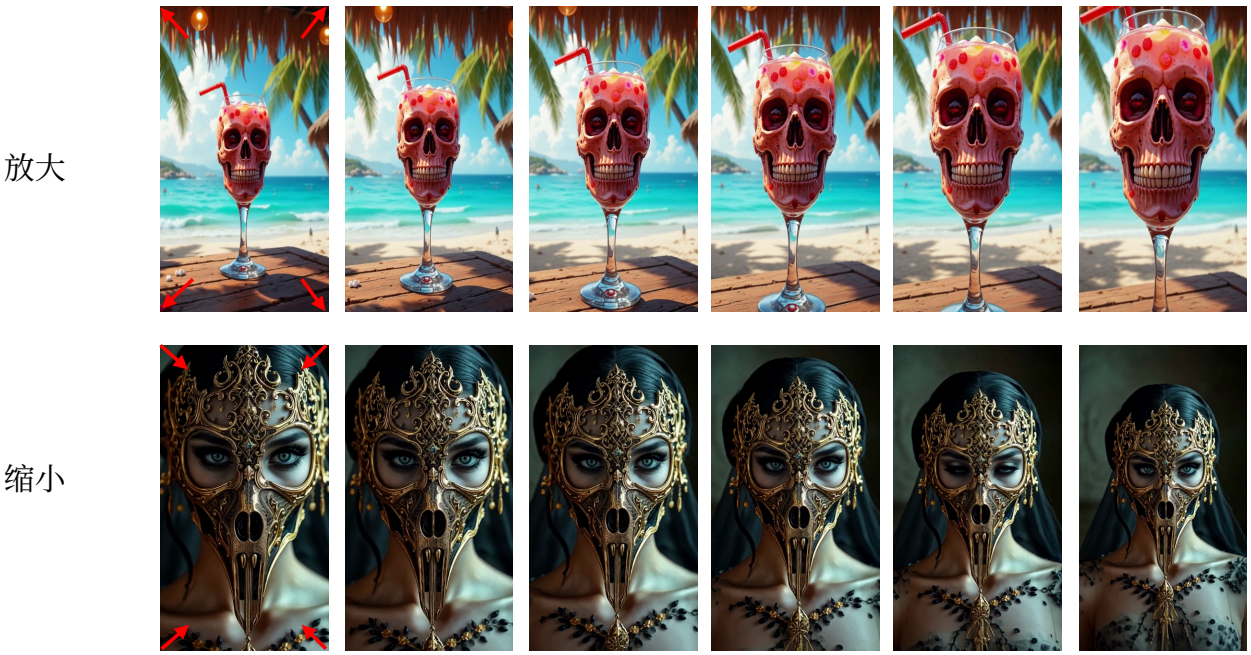

图17 FlashMotion 通过调整前景物体（例如杯子或女性面具）的边界框大小，实现可控的摄像机运动，例如拉近或拉远。

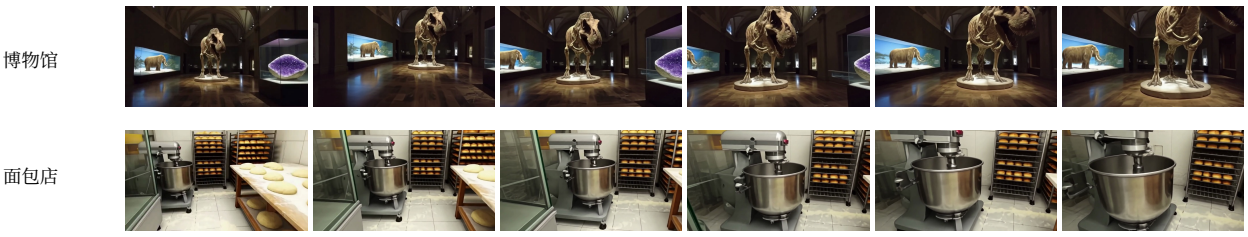

图18 FlashMotion 通过操控关键物体（包括恐龙、猛犸象和工业搅拌机）的边界框，支持在各种环境中进行场景导航，例如面包店或博物馆。

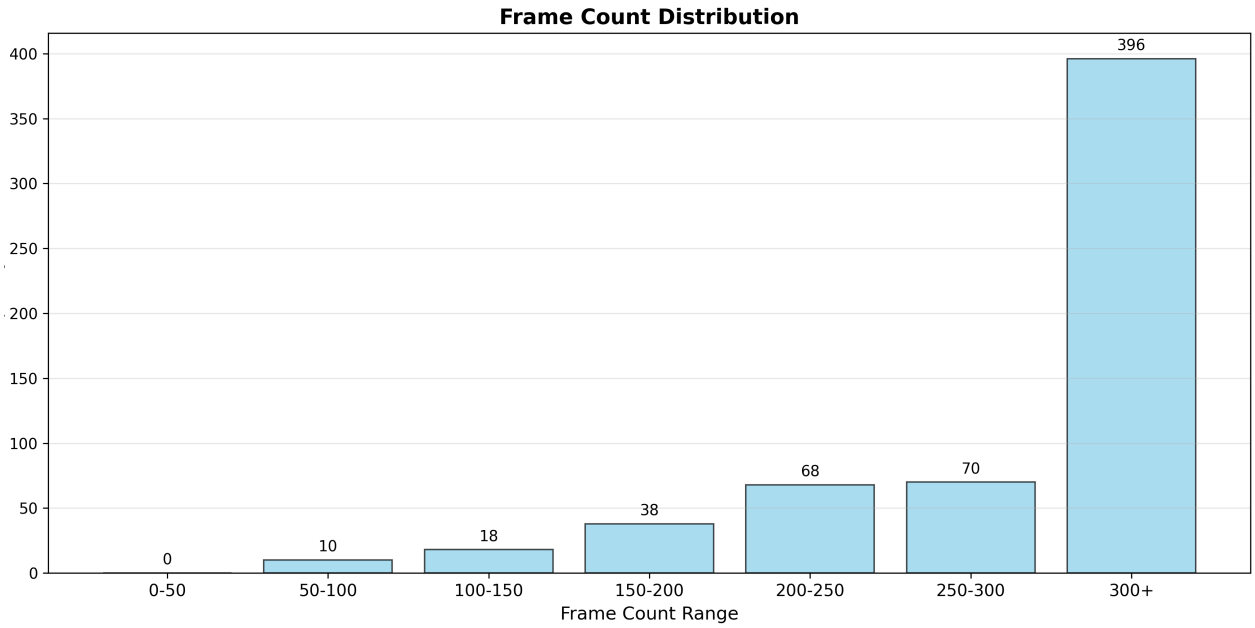

图19 FlashBench中视频帧数的分布，展示了其对长视频生成评估的支持。