Taming Video Models for 3D and 4D Generation via Zero-Shot Camera Control

Supplementary Material

Contents

1. Proof of The Equivalence between Diffusion and Flow Models

We consider Flow Matching [16, 18] as a special case of diffusion modeling [6, 12]. In the following, we will first outline the formulation of diffusion models and then substitute the specific parameterization of Flow Matching to demonstrate their compatibility.

Given a random variable $\mathbf{x}_0$ drawn from an unknown data distribution $q_0(\mathbf{x}_0)$, a Diffusion Probabilistic Model (DPM) [9, 19, 23] defines a forward process that gradually transforms the data into a simple prior distribution, typically a Gaussian distribution. The conditional distribution of the noised variable $\mathbf{x}_t$ at time t given the initial data $\mathbf{x}_0$ is defined as a Gaussian transition kernel [13]:

$$q_t(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t|\alpha_t\mathbf{x}_0, \sigma_t^2\mathbf{I}). \tag{1}$$

Equivalently, a sample $\mathbf{x}_t$ at any time $t \in [0, T]$ can be expressed through a reparameterization [6, 13]:

$$\mathbf{x}_t = \alpha_t\mathbf{x}_0 + \sigma_t\boldsymbol{\epsilon},\ \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{2}$$

Here, α_t and σ_t are scalar functions of time, known as the noise schedule, that control the signal-to-noise ratio. Typically, α_t decreases over time while σ_t increases, satisfying a condition such as $\alpha_t^2 + \sigma_t^2 = 1$ in Variance Preserving (VP) SDEs [9, 23]. Kingma et al. [13] proves that the following stochastic differential equation (SDE) has the same transition distribution in Eq. (1) for any $t \in [0, T]$:

$$\mathrm{d}\mathbf{x}_t = f(t)\mathbf{x}_t\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}_t,\ \mathbf{x}_0 \sim q_0(\mathbf{x}_0), \tag{3}$$

where $\mathbf{w}_t$ is a standard Wiener process. The drift coefficient $f(t)$ and the diffusion coefficient $g(t)$ can be derived using schedule parameters α_t and σ_t [13]:

$$f(t) = \frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t},\quad g^2(t) = \frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t} - 2\frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\sigma_t^2. \tag{4}$$

The generative process of diffusion models involves reversing this forward process. This can be achieved via a corresponding reverse-time SDE [23]. For more efficient generation, one can utilize the associated probability flow ordinary differential equation (PF-ODE), which shares the same marginal distributions as at each time t as that of the SDE [23]. This PF-ODE is given by:

$$\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = f(t)\mathbf{x}_t - \frac{1}{2}g^2(t)\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t). \tag{5}$$

By relating the score function $\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t)$ to the noise term via $\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t) \approx -\frac{\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)}{\sigma_t}$, where $\boldsymbol{\epsilon}_\theta$ is a neural network trained to predict the noise, the ODE becomes [11, 34]:

$$\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = f(t)\mathbf{x}_t + \frac{g^2(t)}{2\sigma_t}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t). \tag{6}$$

Now, let us consider the forward process in Flow Matching [16, 18]. The path from a data point $\mathbf{x}_0$ to a noise sample $\boldsymbol{\epsilon}$ is defined by a simple linear interpolation:

$$\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\cdot\boldsymbol{\epsilon},\ \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{7}$$

where $t \in [0, 1]$. By comparing Eq. (7) with the general form of the diffusion forward process in Eq. (2), we can establish a direct correspondence by setting the diffusion schedule parameters as:

$$\alpha_t = 1 - t \quad \text{and} \quad \sigma_t = t.$$

Substituting this specific parameterization into the definitions for $f(t)$ and $g(t)$ in Eq. (4), we derive the corresponding coefficients for this Flow Matching SDE:

$$f_{\mathrm{FM}}(t) = \frac{\mathrm{d}\log(1-t)}{\mathrm{d}t} = \frac{-1}{1-t}, \tag{8}$$

$$g^2_{\mathrm{FM}}(t) = \frac{\mathrm{d}(t^2)}{\mathrm{d}t} - 2\frac{-1}{1-t}t^2 = \frac{2t}{1-t}. \tag{9}$$

Next, we insert these specific coefficients $f_{\mathrm{FM}}(t)$ and $g^2_{\mathrm{FM}}(t)$ into the PF-ODE formulation from Eq. (6). To analyze the underlying dynamics, we consider the ideal case where the score is perfectly known, which is equivalent to replacing the model prediction $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$ with the ground-truth noise $\boldsymbol{\epsilon}$. This yields:

$$\begin{aligned}
\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} &= f_{\mathrm{FM}}(t)\mathbf{x}_t + \frac{g^2_{\mathrm{FM}}(t)}{2\sigma_t}\boldsymbol{\epsilon} \\
&= \frac{-1}{1-t}\mathbf{x}_t + \frac{2t}{2t\cdot(1-t)}\boldsymbol{\epsilon} \\
&= \frac{\boldsymbol{\epsilon} - \mathbf{x}_t}{1-t} \\
&= \frac{\boldsymbol{\epsilon} - [(1-t)\mathbf{x}_0 + t\cdot\boldsymbol{\epsilon}]}{1-t} \\
&= \frac{(1-t)\boldsymbol{\epsilon} - (1-t)\mathbf{x}_0}{1-t} \\
&= \boldsymbol{\epsilon} - \mathbf{x}_0.
\end{aligned} \tag{10}$$

This resultant vector field, $\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = \boldsymbol{\epsilon} - \mathbf{x}_0$, is precisely the time derivative of the Flow Matching path defined in Eq. (7). This equivalence demonstrates that the process prescribed by Flow Matching is a specific instance of the diffusion models, corresponding to the linear noise schedule $\alpha_t = 1 - t$ and $\sigma_t = t$. Therefore, Flow Matching can be formally viewed as a subset of the broader diffusion modeling framework [6, 12].

2. Evaluation Metrics

We employ seven complementary metrics to comprehensively evaluate video generation quality: FID and $\mathrm{CLIP}_{\mathrm{sim}}$ similarity for static scenes, FVD and $\mathrm{CLIP\text{-}V}_{\mathrm{sim}}$ for dynamic scenes, and ATE, RPE-T, and RPE-R for camera trajectory consistency. These metrics provide objective quantitative assessment across multiple dimensions including image realism, semantic consistency, temporal coherence, and camera motion fidelity.

2.1. Static Scene Evaluation

Fréchet Inception Distance (FID). FID [7] measures image generation quality by comparing the distribution of real and generated images in the Inception-V3 feature space. We use an ImageNet-pretrained Inception-V3 [24] model and extract 2048-dimensional features from the pool3 layer. The FID score is computed as:

$$\mathrm{FID} = \|\mu_r - \mu_g\|^2 + \mathrm{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}) \tag{11}$$

where μ_r and μ_g are the mean vectors of real and generated image features, and Σ_r and Σ_g are the corresponding covariance matrices.

CLIP Similarity. CLIP similarity [22] evaluates the semantic similarity between generated and real images using vision-language pre-trained representations. We employ the CLIP ViT-B/32 model trained on 400 million image-text pairs. The similarity score is calculated as:

$$\mathrm{CLIP}_{\mathrm{sim}} = \frac{1}{N}\sum_{i=1}^{N}\cos(f_{r,i}, f_{g,i}) \tag{12}$$

where $f_{r,i}$ and $f_{g,i}$ are the L2-normalized 512-dimensional CLIP features of the i-th real and generated image pair.

2.2. Dynamic Scene Evaluation

Fréchet Video Distance (FVD). FVD [26] measures distributional differences between real and generated video using pretrained spatio-temporal features. We use an I3D (Inflated 3D ConvNet) pretrained on Kinetics [3] and extract 1024-D features from the global average pooling layer for each video clip. Following FID, we compute the Fréchet distance between the Gaussian fits of real and generated I3D features:

$$\mathrm{FVD} = \|\mu_r - \mu_g\|_2^2 + \mathrm{Tr}\Big(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}\Big), \tag{13}$$

with μ_r, μ_g and Σ_r, Σ_g estimated over clip-level I3D features.

Video CLIP Similarity ($\mathrm{CLIP\text{-}V}_{\mathrm{sim}}$). $\mathrm{CLIP\text{-}V}_{\mathrm{sim}}$ extends CLIP similarity to the temporal domain by computing frame-level semantic consistency between generated and real videos. The score is calculated as:

$$\mathrm{CLIP\text{-}V}_{\mathrm{sim}} = \frac{1}{M}\sum_{j=1}^{M}\left[\frac{1}{T_j}\sum_{t=1}^{T_j}\cos(f_{r,j,t}, f_{g,j,t})\right] \tag{14}$$

where M is the number of video pairs, T_j is the frame count of the j-th video pair, and $f_{r,j,t}$, $f_{g,j,t}$ are the CLIP features of the t-th frame in the j-th video pair.

2.3. Camera Trajectory Evaluation

Absolute Trajectory Error (ATE). Before evaluation, we align the estimated trajectory to the reference by a global Sim3 transform (scale, rotation, translation). Let the aligned pose components be $\tilde{\mathbf{t}}_{\mathrm{est},i}$ and $\tilde{\mathbf{R}}_{\mathrm{est},i}$. ATE measures global consistency by the Euclidean distance between corresponding camera positions:

$$\begin{aligned}
\mathrm{ATE}_i &= \left\|\mathbf{t}_{\mathrm{ref},i} - \tilde{\mathbf{t}}_{\mathrm{est},i}\right\|_2, \\
\mathrm{ATE} &= \sqrt{\frac{1}{n}\sum_{i=1}^{n}\mathrm{ATE}_i^2}.
\end{aligned} \tag{15}$$

Relative Pose Error — Translation (RPE-T). RPE-T evaluates local translation accuracy between consecutive

frames. Define relative motions via poses (index gap $\Delta = 1$):

$$\Delta\mathbf{T}_{\text{ref},i} = \mathbf{T}_{\text{ref},i}^{-1}\mathbf{T}_{\text{ref},i+1}, \qquad \Delta\mathbf{T}_{\text{est},i} = \tilde{\mathbf{T}}_{\text{est},i}^{-1}\tilde{\mathbf{T}}_{\text{est},i+1}. \quad (16)$$

Let $\Delta\mathbf{t}_{\text{ref},i}$ and $\Delta\mathbf{t}_{\text{est},i}$ be the translation parts of these relative transforms. The per-step error and RMSE are:

$$\begin{aligned} \text{RPE-T}_i &= \left\|\Delta\mathbf{t}_{\text{ref},i} - \Delta\mathbf{t}_{\text{est},i}\right\|_2, \\ \text{RPE-T} &= \sqrt{\frac{1}{n-1}\sum_{i=1}^{n-1}\text{RPE-T}_i^2}. \end{aligned} \quad (17)$$

Relative Pose Error — Rotation (RPE-R). RPE-R assesses the accuracy of orientation changes between consecutive frames. Let the relative rotations be

$$\Delta\mathbf{R}_{\text{ref},i} = \mathbf{R}_{\text{ref},i}^{-1}\mathbf{R}_{\text{ref},i+1}, \qquad \Delta\mathbf{R}_{\text{est},i} = \tilde{\mathbf{R}}_{\text{est},i}^{-1}\tilde{\mathbf{R}}_{\text{est},i+1}. \quad (18)$$

The per-step angular error (degrees) and RMSE are:

$$\begin{aligned} \text{RPE-R}_i &= \arccos\left(\frac{\operatorname{trace}\left(\Delta\mathbf{R}_{\text{ref},i}^{\top}\Delta\mathbf{R}_{\text{est},i}\right) - 1}{2}\right)\cdot\frac{180}{\pi}, \\ \text{RPE-R} &= \sqrt{\frac{1}{n-1}\sum_{i=1}^{n-1}\text{RPE-R}_i^2}. \end{aligned} \quad (19)$$

2.4. Evaluation Details

Preprocessing. For FID, images are resized to 299×299 and fed to Inception-V3 with standard ImageNet normalization. For FVD and CLIP-based metrics, frames are resized to 224×224 with the respective model normalizations. To align with I3D input requirements for FVD, we uniformly downsample the generated videos to 16 frames while strictly preserving the start and end frames to maintain boundary constraints. A similar temporal sampling strategy is applied for CLIP-based video metrics. For camera trajectory evaluation, images are resized to 720×480 and uniformly sampled to 20 frames.

Evaluation Protocol. To ensure robust feature covariance estimation for distributional metrics, we compute statistics over the aggregated evaluation set. Specifically, for FID on static scenes, we aggregate approximately $1,200$ GT images from $40+$ scenes as the reference distribution, comparing them against $\approx 2,200$ generated novel views (40∼120 views per scene). For wild scenes without ground truth, we report the average FID over 5 representative scene groups using manually curated reference sets (20∼40 images per scene) to ensure content consistency. For FVD, we evaluate ≈ 700 generated video clips from $50+$ scenes against an equal number of reference clips extracted from DAVIS and cinematic sequences.

All baseline methods are evaluated under this identical protocol. This standardized comparison ensures that the observed relative performance gaps reliably reflect the intrinsic differences in generation quality, verifying the superiority of our method across the evaluated metrics.

For trajectories, poses are recovered by SfM, the estimated trajectory is aligned to the reference by Sim3 to resolve scale, and metrics are computed using `evo` with alignment and scale correction enabled.

3. Implementation Details

This section provides a detailed breakdown of the Flow-Gated Latent Fusion (FLF) module,as introduced in Section 3.3 of the *main* paper. The goal of FLF is to identify and selectively update latent channels that are highly relevant to motion, thereby preserving visual details encoded in appearance-focused channels. To achieve this, at each denoising step i, FLF computes a motion similarity score $S^{(t,c)}$ for each latent channel c. Below, we detail how this score is calculated.

3.1. Details for FLF Estimation and Motion Scoring

Optical Flow Estimation At each denoising step i, we compute optical flow maps for each channel c of both the predicted latent $\hat{\mathbf{x}}_0^{(t)}$ and the target trajectory latent $\mathbf{x}_{\text{traj}}$. The computation is performed frame-by-frame; that is, for each latent tensor, we calculate the dense optical flow between consecutive temporal frames using the Farnebäck algorithm [5]. This process yields a predicted flow map, $\mathcal{F}_{\text{pred}}^{(t,c)}$, and a ground-truth (GT) flow map, $\mathcal{F}_{\text{gt}}^{(t,c)}$. At each pixel, the flow is a 2D vector (u_*, v_*) representing horizontal and vertical displacement. All subsequent metric calculations are performed over the set of valid (i.e., non-occluded) pixels, defined as $\Omega^{(t,c)} = \{(x, y, \tau) \mid \mathbf{M}^{(c)}(x, y, \tau) = 1\}$, where (x, y) are pixel coordinates and τ is the frame index. Since optical flow is computed between adjacent frames, for a latent tensor with T_l total frames, the index τ ranges from 1 to $T_l - 1$.

Metric Calculation The motion score $S^{(t,c)}$ is derived from three standard optical flow metrics that quantify the error between the predicted flow $\mathcal{F}_{\text{pred}}^{(t,c)}$ and the ground-truth flow $\mathcal{F}_{\text{gt}}^{(t,c)}$ at each step i.

- **Masked End-point Error (M-EPE)** measures the average Euclidean distance between the predicted and GT flow vectors over all valid pixels. Let $err(x, y, \tau)$ be the Euclidean error at a specific pixel:

$$err(x, y, \tau) = \left\|\mathcal{F}_{\text{pred}}^{(t,c)}(x, y, \tau) - \mathcal{F}_{\text{gt}}^{(t,c)}(x, y, \tau)\right\|_2. \quad (20)$$

  The M-EPE is then calculated as:

$$\text{M-EPE}^{(t,c)} = \frac{1}{|\Omega^{(t,c)}|}\sum_{(x,y,\tau)\in\Omega^{(t,c)}} err(x, y, \tau). \quad (21)$$

- **Masked Angular Error (M-AE)** calculates the average angular difference. We first define the cosine similarity

$sim(x, y, \tau)$ between the flow vectors:

$$sim(x, y, \tau) = \frac{\mathcal{F}_{\text{pred}}^{(t,c)}(x, y, \tau) \cdot \mathcal{F}_{\text{gt}}^{(t,c)}(x, y, \tau)}{\|\mathcal{F}_{\text{pred}}^{(t,c)}(x, y, \tau)\| \cdot \|\mathcal{F}_{\text{gt}}^{(t,c)}(x, y, \tau)\|}. \quad (22)$$

The M-AE is derived by averaging the arccosine of this similarity:

$$\text{M-AE}^{(t,c)} = \frac{1}{|\Omega^{(t,c)}|} \sum_{(x,y,\tau)\in\Omega^{(t,c)}} \arccos\left(sim(x, y, \tau)\right). \quad (23)$$

- **Outlier Percentage (Fl-all)** is the percentage of pixels in $\Omega^{(t,c)}$ where the flow estimation is considered erroneous. Following standard benchmarks, a pixel is flagged as an outlier if its M-EPE exceeds 3 pixels or if its relative error, $\|\mathcal{F}_{\text{pred}}^{(t,c)} - \mathcal{F}_{\text{gt}}^{(t,c)}\|_2 / \|\mathcal{F}_{\text{gt}}^{(t,c)}\|_2$, is greater than 5%. We denote this outlier percentage as $\text{F}^{(t,c)}$.

Normalization and Weighting The three metrics exist on different scales, so we first normalize each to the range $[0, 1]$ before combining them. This corresponds to the ${\text{Norm}_k}^{(t,c)}$ terms used in the main text:

$$\begin{aligned} {\text{Norm}_{\text{E}}}^{(t,c)} &= \min(\text{M-EPE}^{(t,c)}/n_{\text{E}}, 1), \\ {\text{Norm}_{\text{A}}}^{(t,c)} &= \min(\text{M-AE}^{(t,c)}/n_{\text{A}}, 1), \\ {\text{Norm}_{\text{F}}}^{(t,c)} &= \min(\text{F}^{(t,c)}/n_{\text{F}}, 1), \end{aligned} \quad (24)$$

where $n_{\text{E}}, n_{\text{A}}$, and n_{F} are normalization constants chosen to reflect typical value ranges for each metric. The final motion score $S^{(t,c)}$ is a weighted sum of the inverted normalized errors, as defined in Eq. (25) (corresponding to Eq. (6) in the *main text*):

$$S^{(t,c)} = \sum_{k\in\{\text{E, A, F}\}} \gamma_k \left(1 - {\text{Norm}_k}^{(t,c)}\right), \quad (25)$$

where the weights γ_k (where $k \in \{\text{E}, \text{A}, \text{F}\}$ and $\sum_k \gamma_k = 1$) and the normalization constants are set based on common practices in optical flow evaluation to balance each metric's contribution. In our experiments, we set $n_{\text{E}} = 10$, $n_{\text{A}} = 30$, and $n_{\text{F}} = 0.5$. The weights in Eq. (25) are set to $(\gamma_{\text{E}}, \gamma_{\text{A}}, \gamma_{\text{F}}) = (0.4, 0.4, 0.2)$.

3.2. Hyperparameter Settings

To facilitate reproducibility, we provide the default hyperparameter settings used in our experiments, as listed in Table 1. These values are based on the implementation with the Wan 2.1 backbone. In practical applications, users may fine-tune these coefficients according to specific scene requirements to achieve optimal results.

Beyond the coefficients listed above, we specify several key operational parameters. Taking the Wan 2.1 model (which uses 50 sampling steps) as an example, the Intra-Step Recursive Refinement (IRR) is applied by default during the first 20 sampling steps. For optical flow estimation within the FLF module, we utilize the classic Farneback algorithm [5] with its default settings ('pyr_scale'=0.5, 'levels'=3, 'winsize'=15).

Table 1. Default coefficient settings used in our experiments (taking Wan 2.1 implementation as an example). While these values serve as a robust baseline, users can fine-tune them for specific scenes to maximize generation quality.

	Reference	Value	Description
γ_k	Eq. (6) in *main* paper	$[0.4, 0.4, 0.2]$	Weights for optical flow metrics
$\lambda^{(t)}$	Eq. (6) in *main* paper	0.65	Threshold for FLF channel filtering
ρ	Eq. (8) in *main* paper	4.0	DSG guidance scale

Critically, regarding the channel filtering in FLF, in addition to the threshold $\lambda^{(t)}$, we implement a progressively relaxed channel selection strategy to balance structural guidance and detail preservation: **Phase 1 (Steps 0–5):** Channel filtering is disabled. All channels are injected with guidance information. This is because the early-stage latent states are too noisy for reliable optical flow calculation, necessitating full guidance injection to establish initial structure. **Phase 2 (Steps 6–10):** We enforce a strict limit where a maximum of 2 channels are allowed to retain the original prediction (i.e., at least 14 channels are replaced with guided features). This ensures sufficient structural information is injected during the formative stages of generation. **Phase 3 (Steps $\geq$ 11):** The constraint is relaxed to allow a maximum of 6 channels to retain the original prediction. This looser constraint prevents excessive guidance from compromising fine texture details in the later stages.

4. More Experimental Results

This section provides additional experiments and results that complement the findings presented in the main paper. We include a detailed efficiency analysis, further ablation studies, and more qualitative examples to fully demonstrate the capabilities and robustness of our framework.

4.1. Efficiency and Runtime Analysis

Our framework is training-free and operates entirely at inference time. Table 2 provides a detailed comparison of inference efficiency against several state-of-the-art methods on a single NVIDIA A100 GPU.

Our method incurs zero training cost, offering a significant advantage over resource-intensive fine-tuning approaches. The primary computational overhead stems from the IRR module, which effectively adds an extra sampling process, taking approximately the same time as a single standard sampling step. In contrast, the DSG module in-

Figure 1. Qualitative ablation study of the DSG method. Substituting DSG with a standard CFG formulation fails to handle the large angular disparity between the two velocity fields, resulting in significant visual artifacts and errors. Removing the adaptive weighting factor β_t (denoted as DSG w/o β_t) compromises guidance stability and introduces inconsistencies. In contrast, our full DSG framework stably generates high-fidelity and structurally consistent results.

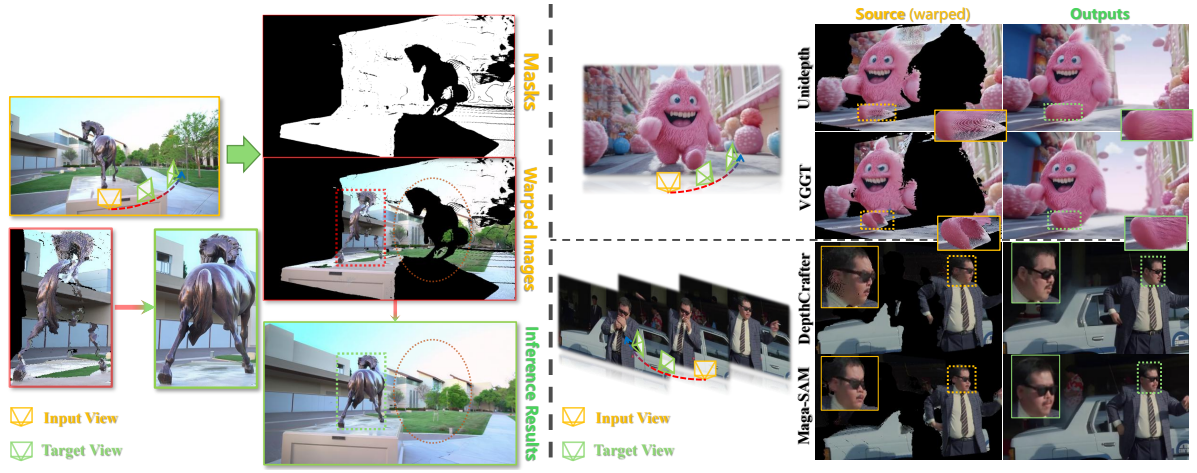

Figure 2. Depth-models ablation. Our method leverages the inherent world knowledge of VDMs to correct errors and fill missing regions even under challenging inputs (left). This strong self-correction ability ensures broad compatibility with different depth estimators (right). Despite variations or noise in depth-based warping, it reliably compensates through learned priors and produces realistic, high-quality results.

volves only a simple matrix operation per step, incurring negligible temporal cost. Similarly, the FLF module's channel-wise optical flow estimation presents a substantially lower computational burden than the backbone model inference. A detailed breakdown of these component costs is reported in Table 3. Note that while absolute times may vary across hardware configurations, the relative temporal relationships remain reliable. Despite these additions, our framework maintains inference speeds comparable to, and often faster than, existing methods, as evidenced in Table 2.

This demonstrates that our framework achieves robust controllability without prohibitive computational costs, offering an efficient alternative to training-intensive pipelines.

4.2. Ablation of DSG and Naive CFG

Through extensive experiments, we observe that the difference between our trajectory-guided velocity $\mathbf{v}_t^{\text{traj}}$ and the unguided velocity $\mathbf{v}_t^{\text{ori}}$ is far greater than that between the conditional $\mathbf{v}_{\text{con}}$ and unconditional $\mathbf{v}_{\text{uncon}}$ estimates in standard CFG [8]. Specifically, empirical analysis reveals that

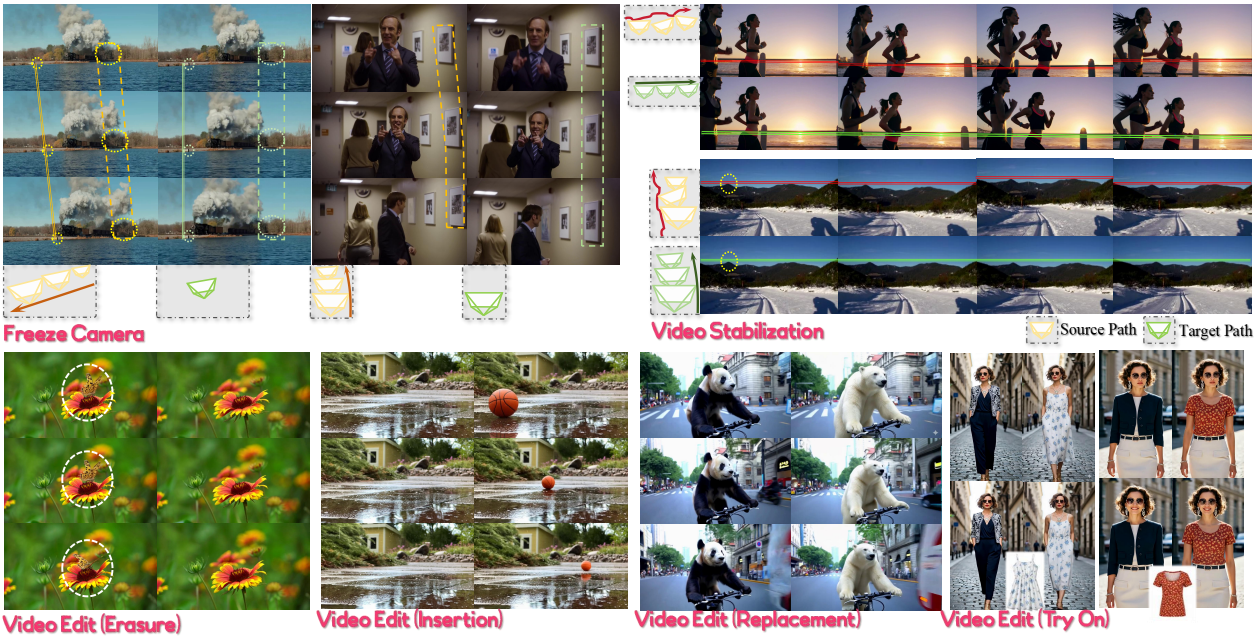

Figure 3. Other video effects enabled by our method. Beyond video re-cam, our flexible depth-based warping also supports various video editing operations, such as freezing the camera, stabilizing video, and editing video content. These extensions further broaden the practical scope of our approach.

Table 2. Efficiency comparison. We measure inference throughput on a single NVIDIA A100 across methods built on SVD [2], Wan 2.1 [27], CogVideoX [30], LongCat [25], and custom backbones. Our approach achieves competitive generation speed compared to prior approaches while avoiding any training overhead. ***SR*** denotes LongCat's built-in super-resolution model, and ***D*** represents its distilled version. The **best** and second-best results are highlighted in bold and underlined, respectively.

	Resolution	Inference Speed (frames/min)	Base Video Model	Training -Free
See3D [20]	576 × 1024	14.71	Custom	✗
ViewCrafter [33]	576 × 1024	13.89	Custom	✗
ViewExtrapolator [17]	576 × 1024	15.63	SVD	✓
TrajectoryAttention [29]	576 × 1024	4.55	SVD	✗
TrajectoryCrafter [32]	384 × 672	14.71	CogVideoX	✗
NVS-Solver [31]	576 × 1024	2.69	SVD	✓
ReCamMaster [1]	480 × 832	5.55	Wan 2.1 T2V	✗
WorldForge, on Wan 2.1 [27]	720 × 1280	1.45	Wan 2.1 I2V	✓
WorldForge, on Wan 2.1 [27]	480 × 832	3.68	Wan 2.1 I2V	✓
WorldForge, on SVD [2]	576 × 1024	**19.23**	SVD	✓
WorldForge, on LongCat [25]	480 × 832	3.85	Longcat	✓
WorldForge, on LongCat [25]	720 × 1280	3.33	Longcat (***SR***)	✓
WorldForge, on LongCat [25]	480 × 832	<u>16.67</u>	Longcat (***D***)	✓
WorldForge, on LongCat [25]	720 × 1280	10.20	Longcat (***D+SR***)	✓

Table 3. Computational cost breakdown of a single generation step. We report the runtime of each component on an NVIDIA A100, taking the generation of a 49-frame video at 832 × 480 resolution as an example. By default, we apply our guidance during the first 20 sampling steps. The primary overhead comes from the IRR module, while the DSG module incurs negligible cost.

Component	Time (s)
VAE Encoding & Decoding	4.5
Backbone Inference (Transformer)	8.4
FLF Module (Ours)	1.2
DSG Module (Ours)	≈ 0
IRR Module (Ours)	14.1
Total Runtime	**29.0**

the angular difference in our setting typically ranges from 50° to 70°, significantly exceeding the $< 5°$ divergence found in typical CFG scenarios. To mitigate the adverse effects caused by this large angular discrepancy, we propose Dual-Path Self-Corrective Guidance (DSG). A direct visual comparison is presented in Fig. 1, where we compare our full framework against a standard CFG implementation and an ablated version without the adaptive weight β_t. The results demonstrate that replacing DSG with a naive CFG formulation leads to severe visual artifacts and structural distortions, while removing the adaptive weighting factor reduces guidance stability. In contrast, our full DSG framework successfully maintains structural integrity and high perceptual quality while closely adhering to the intended camera path.

4.3. Ablation on Video Diffusion Models

To evaluate the transferability of our proposed guidance mechanism and its performance across models of varying parameter scales, we conducted ablation studies by porting our entire framework to the U-Net-based Stable Video Diffusion (SVD) [2], which possesses fewer parameters, and the recently released LongCat model [25]. We performed minor hyperparameter fine-tuning to adapt our method to their respective architectures and sampling strategies. Subsequently, we conducted a fair comparison using identical inputs. Quantitative results are presented in Table 4. It is worth noting that due to SVD's constraints in parameter count and architecture, it encapsulates fewer inherent world priors, which prevents it from fully exploiting the potential of our guidance algorithm. Comprehensive visualization results demonstrating the capabilities across these models are shown in Fig. 10 through Fig. 14.

Table 4. Quantitative comparison across different backbones. Using single-view 3D scene generation as a benchmark, we evaluate our method on SVD [2], Wan 2.1 [27], and LongCat [25]. The results demonstrate the scalability of our approach and its ability to generalize across different VDM architectures. Furthermore, the performance gains on advanced backbones indicate that our method effectively leverages the capabilities of the underlying model, promising improved generation quality as base models continue to evolve.

	CLIP ↑	ATE ↓	RPE-T ↓	RPE-R ↓
WorldForge (on SVD [2])	0.910	0.265	0.316	0.444
WorldForge (on Wan 2.1 [27])	0.948	**0.077**	0.086	**0.221**
WorldForge (on LongCat [25])	**0.949**	0.095	**0.076**	0.230

4.4. Ablation on Depth Estimation Models

Our framework operates on a warp-and-repaint strategy. To assess the robustness and flexibility of our approach regarding depth estimation, we evaluated its performance using several state-of-the-art depth estimators: VGGT [28], UniDepth [21], Mega-SaM [15], and DepthCrafter [10]. As illustrated in Fig. 2, our method demonstrates broad compatibility, maintaining consistently high performance across all tested models. Even when depth-based warping yields challenging inputs characterized by noise, errors, or significant disocclusion regions, our framework effectively compensates for these imperfections. This resilience stems from the strong generative world priors inherent in the underlying VDM, which our guidance modules leverage to correct artifacts and plausibly inpaint missing areas during the repainting stage. This self-correction capability confirms that our framework functions in a plug-and-play manner with various depth estimation techniques and naturally scales with improvements in depth estimation performance.

4.5. Applications in Video Editing

Beyond trajectory-controlled generation, our framework's flexibility makes it a powerful tool for various video post-production and editing tasks. This includes effects like video stabilization, camera freezing, and dynamic viewpoint switching.

Furthermore, by incorporating a flexible masking strategy, our framework can perform diverse content edits such as object removal, addition, subject replacement, and virtual try-on seamlessly. The general process for these edits involves first segmenting the target region in each frame using a tool like SAM [14]. The desired edit is then applied to the first frame (e.g., using Gemini [4]). Finally, this edited frame and the corresponding masks are processed by our pipeline to render a temporally consistent result. For adding new objects where none exist in the source video, a simple bounding box can be provided to guide the placement. Fig. 3 shows several qualitative examples of these video editing effects.

4.6. Generation on Challenging Scenes

Our approach demonstrates robust performance in difficult cases where other methods may falter. We highlight two such scenarios: human-centric scenes and single-image 360° view generation.

Human-Centric Scenes. Human-centric scenes are challenging for novel view synthesis due to the need for high structural and temporal consistency. As shown in Fig. 4, some methods can struggle with these cases, sometimes introducing artifacts, unintended motion, or difficulty rendering plausible facial features. For instance, TrajectoryCrafter [32] may recover the coarse structure, but can introduce unnatural facial deformations. In contrast, our method's use of strong generative priors and precise trajectory guidance helps maintain scene stationarity and consistency, producing more natural renderings that better preserve the subject's appearance.

Large Camera Movements and 360° View Generation. Generating large camera movements (e.g., a 180° turn) or full 360° orbit views from a single image in a single pass is highly challenging for existing methods. It risks hallucination in invisible regions due to the limited field-of-view of the source observation. Our method effectively resolves this problem via an iterative multi-clip generation strategy. By using the last frame of the previous clip as the prior for the next, we successfully achieve large-range scene generation, such as 180° turns (as shown in Fig. 5). Furthermore, combined with our framework's precise trajectory control, it enables the creation of coherent, object-centric orbit views of complex scenes (Fig. 6). We achieve the full 360° loop by generating a sequence where the final frame seamlessly connects to the first. This is made possible by our precise guidance, which maintains high image quality

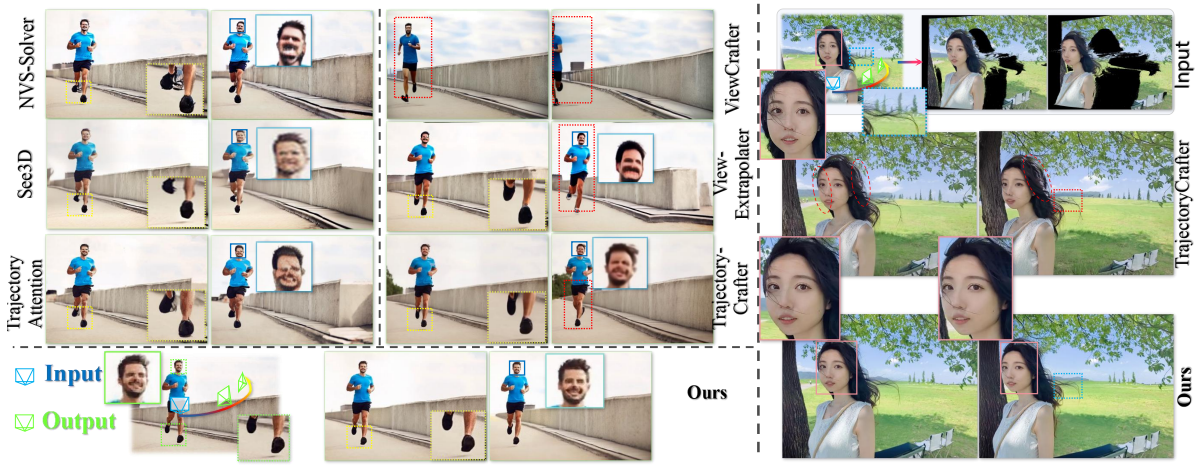

Figure 4. Static 3D generation on human-centric scenes. Existing methods struggle, particularly with motion-prone shots (left) and portrait close-ups (right). On the left, baselines introduce artifacts and unintended motion. On the right, most fail to produce plausible results; TrajectoryCrafter [32] recovers coarse structure but lacks detail and visual appeal. In contrast, our method maintains scene stationarity under trajectory guidance and produces natural, faithful renderings, achieving both precise control and high perceptual quality.

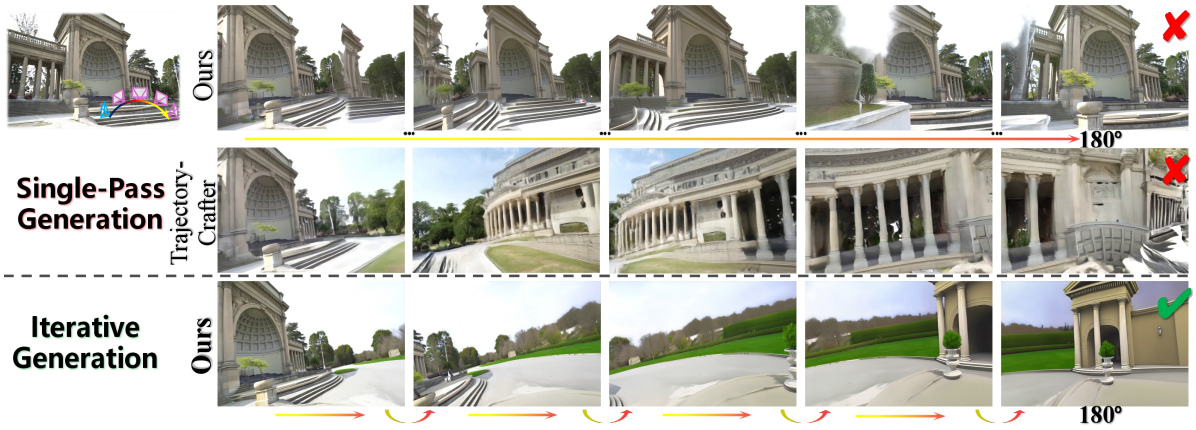

Figure 5. Large camera movements (e.g., 180°). Single-pass generation of large angles often suffers from poor quality. Our method effectively resolves this problem via iterative generation.

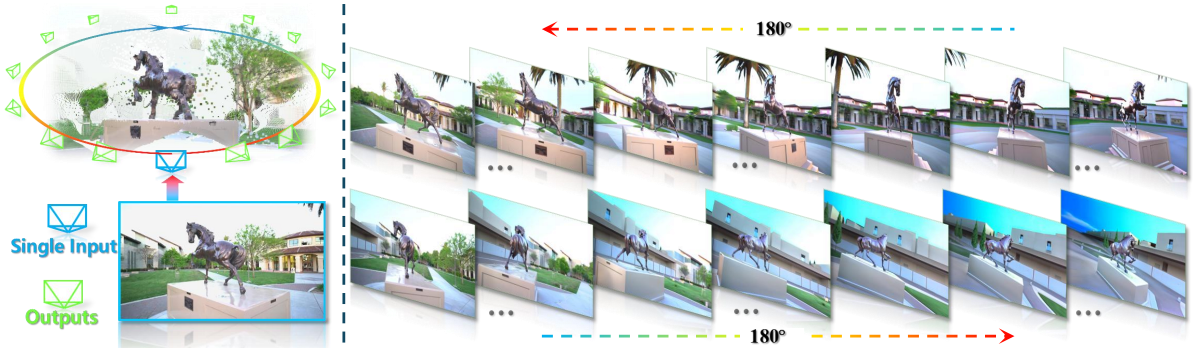

Figure 6. 360° orbit views from a single real-world outdoor image. With precise trajectory control and realistic rendering, our method overcomes the viewpoint limitation of single-image generation and produces ultra-wide views of complex real scenes. Unlike panorama-based approaches, it directly supports object-centric trajectories and achieves higher visual quality.

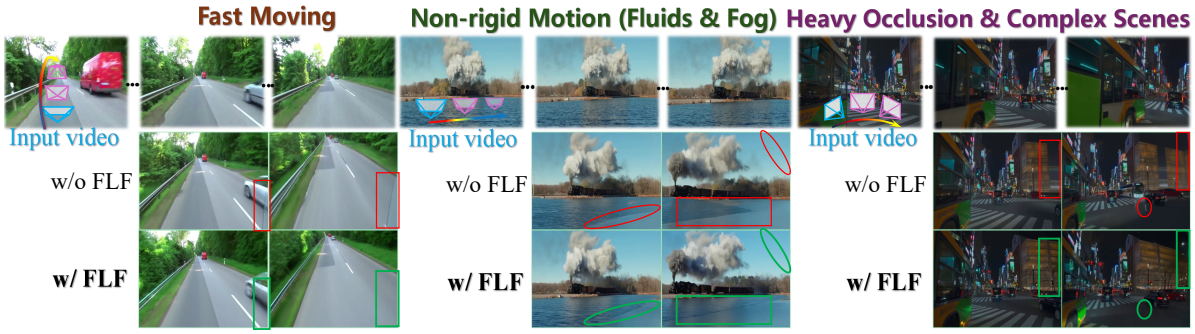

Figure 7. Robustness in challenging scenarios. Our framework maintains structural integrity even under fast motion and complex occlusions.

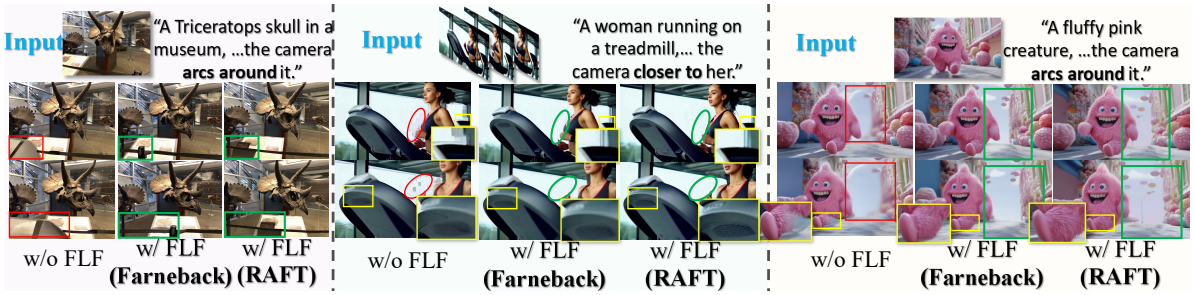

Figure 8. Robustness across optical flow estimators. FLF consistently enhances quality with both Farnebäck [5] and RAFT, validating its flexibility and robustness.

and prevents the accumulation of errors over the entire long-range trajectory, thereby avoiding a common point of failure in other methods. Unlike traditional panoramic approaches, our method directly generates a continuous view along a given trajectory, offering more flexibility and strong visual quality, particularly for object-centric paths.

Challenging Scenarios. We also demonstrate the robustness against fast motion, occlusions, and non-rigid dynamics. As illustrated in Fig. 7, even when local flow precision drops, enabling FLF consistently yields better results than disabling it. This robustness relies on our adaptive designs: when flow is unstable (e.g., high variance), our dynamic threshold automatically adjusts. This allows more channels to receive control signals, preventing misclassification and maintaining structural stability.

4.7. Robustness across Optical Flow Estimators

A potential concern is whether our framework heavily relies on a specific optical flow algorithm. To validate this, we replaced the lightweight Farnebäck algorithm [5] with the learning-based RAFT model. As shown in Fig. 8, our Flow-Gated Latent Fusion (FLF) consistently enhances generation quality regardless of the flow backbone. Furthermore, our multi-metric scoring system (evaluating magnitude, direction, and reliability) mitigates single-metric noise, ensuring that the lightweight Farnebäck choice is highly sufficient for our pipeline.

4.8. Limitations and Failure Cases

While our framework achieves precise zero-shot camera control, we acknowledge that, similar to other depth-warping-based methods (e.g., TrajectoryCrafter [32]), our performance is inherently bottlenecked by the quality of the underlying depth estimation.

As illustrated in Fig. 9, in scenarios involving extremely complex scene dynamics or severe depth errors, structural distortions may occur. However, it is worth noting that our dynamic gating mechanism automatically reduces the number of filtered channels in such chaotic cases, ensuring that the outputs are typically no worse than the baseline model without FLF guidance. In future work, introducing explicit camera pose encoding or semantic priors could help resolve these fundamental ambiguities and further enhance robustness against depth failures.

4.9. More Cases

To provide a more comprehensive evaluation of our method's performance across different backbone architectures, we present additional qualitative results in Fig. 10, Fig. 11, Fig. 12, Fig. 13 and Fig. 14. As illustrated, our approach achieves superior visual fidelity and structural plausibility, consistently delivering state-of-the-art performance on both Wan 2.1 [27] and LongCat [25] models.

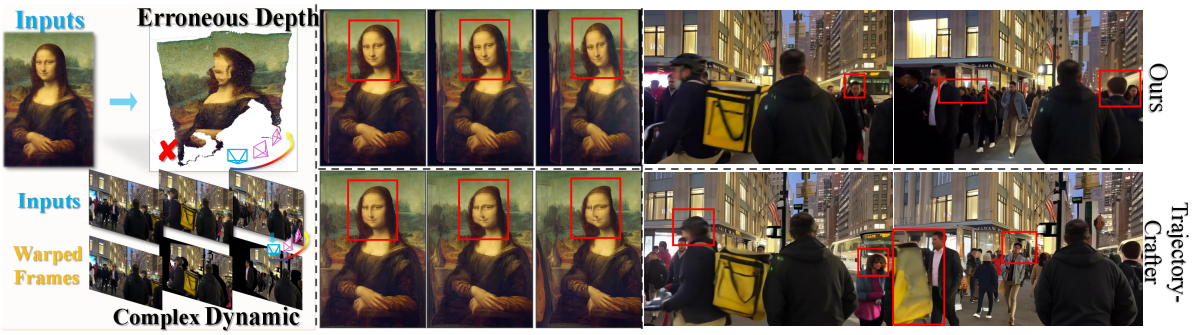

Figure 9. Failure cases. Erroneous depth estimation in highly complex scenes can diminish the control accuracy, leading to artifacts.

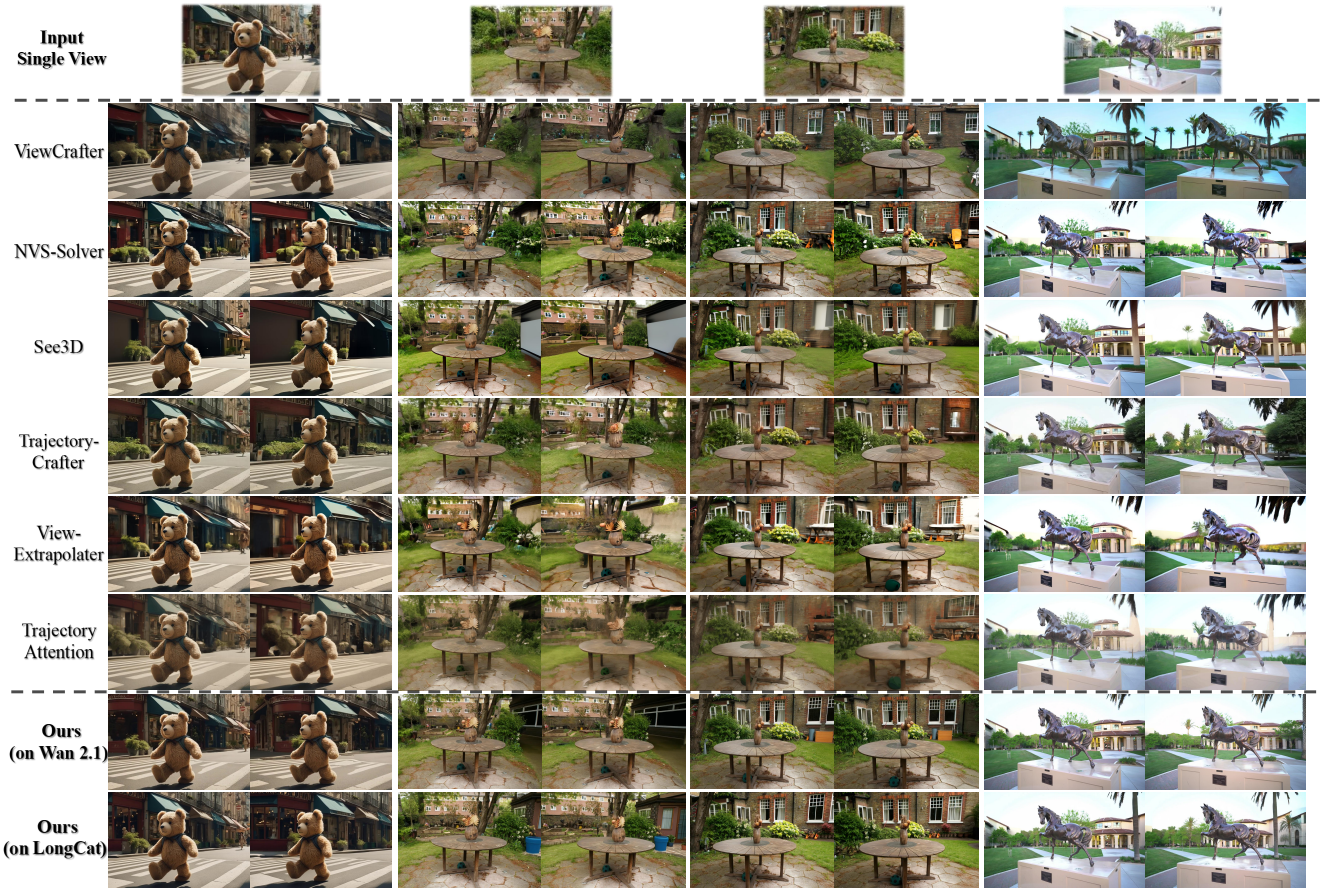

Figure 10. Additional qualitative results for single-view 3D scene generation (Case 1). Validated on Wan 2.1 and LongCat architectures, our method consistently produces 3D-consistent novel views with high visual fidelity.

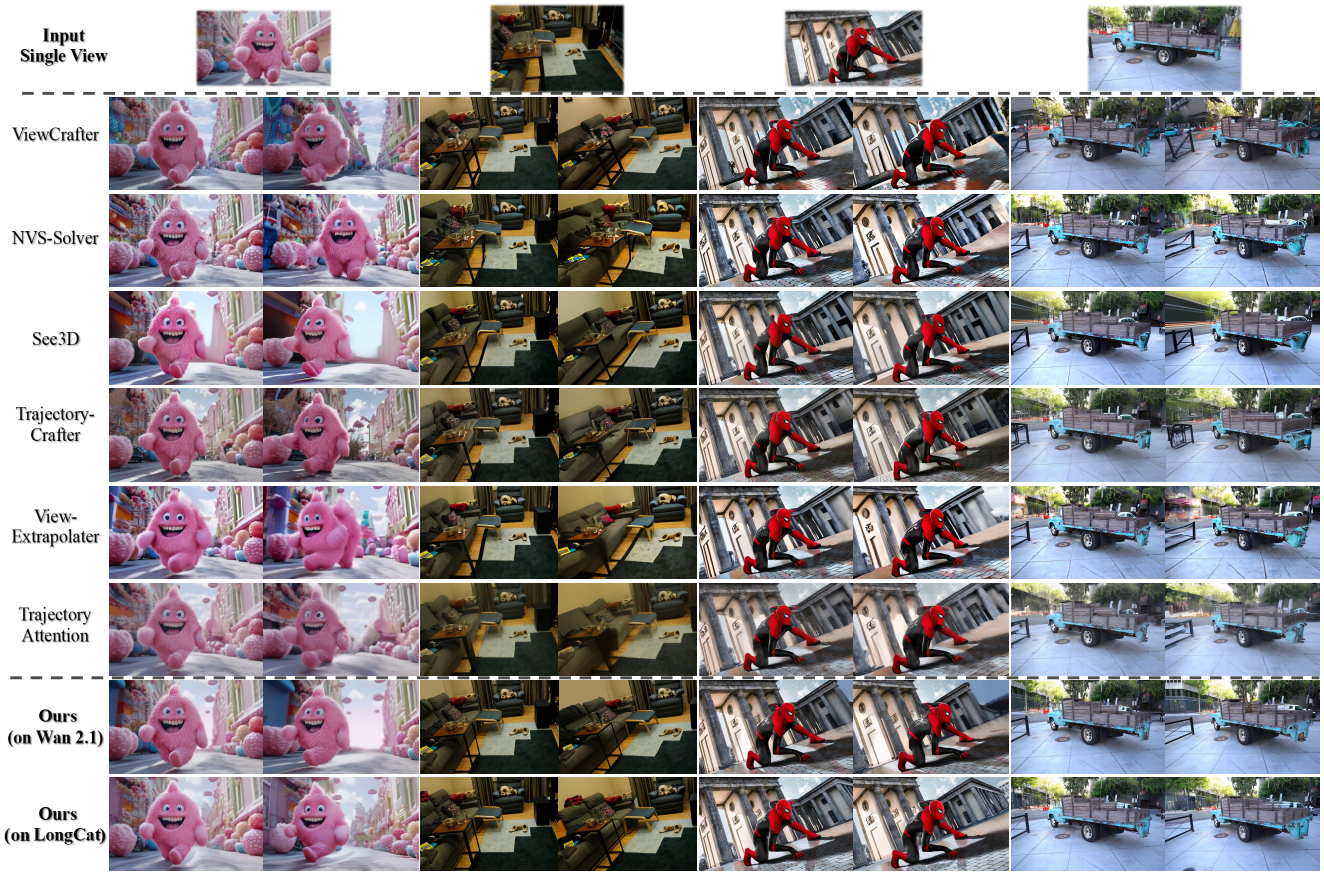

Figure 11. Additional qualitative results for single-view 3D scene generation (Case 2). Validated on Wan 2.1 and LongCat architectures, our method consistently produces 3D-consistent novel views with high visual fidelity.

Figure 12. Additional qualitative results for dynamic video re-filming (Case 1). Validated on Wan 2.1 and LongCat architectures, our method enables effective camera control with superior realism and temporal smoothness.

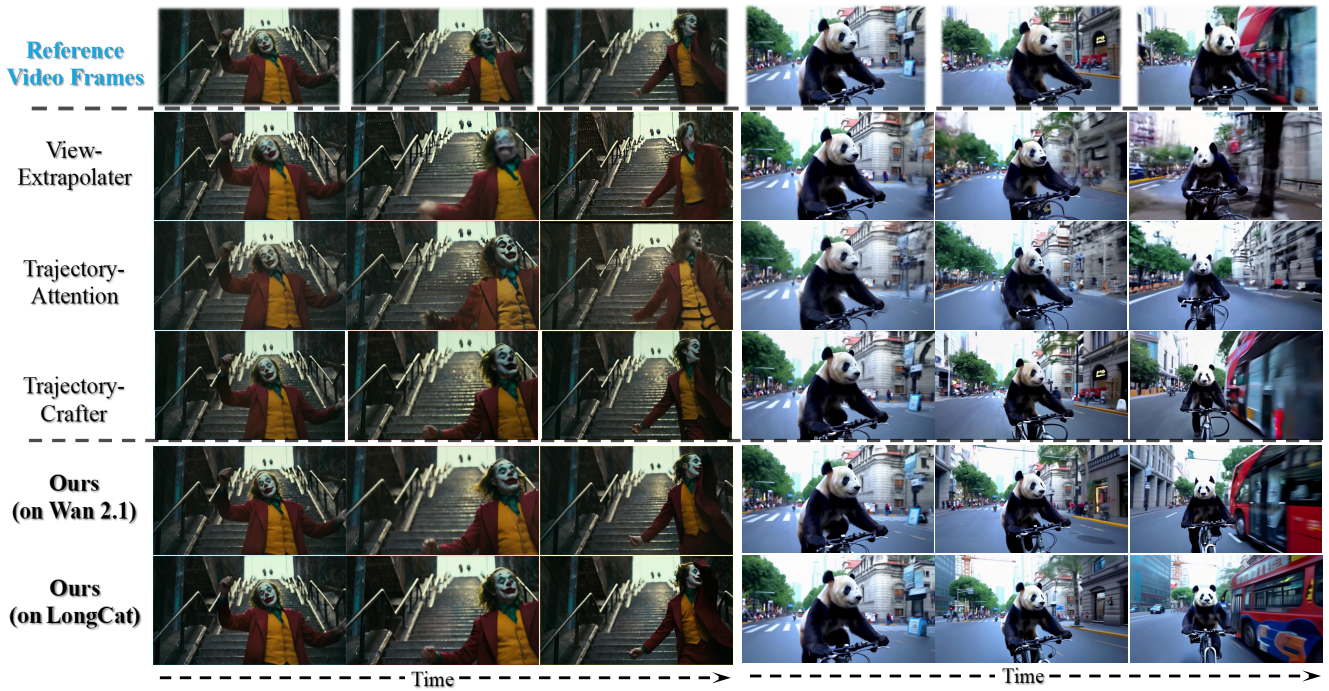

Figure 13. Additional qualitative results for dynamic video re-filming (Case 2). Validated on Wan 2.1 and LongCat architectures, our method enables effective camera control with superior realism and temporal smoothness.

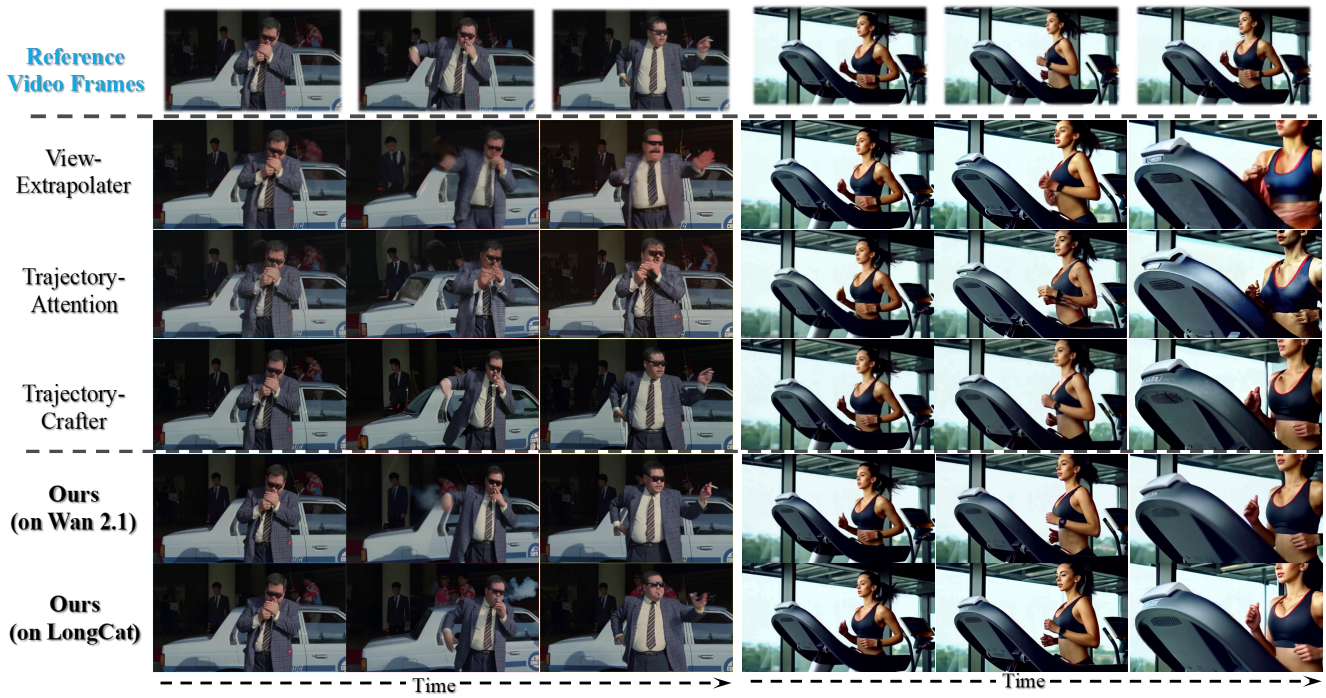

Figure 14. Additional qualitative results for dynamic video re-filming (Case 3). Validated on Wan 2.1 and LongCat architectures, our method enables effective camera control with superior realism and temporal smoothness.

References

[1] Jianhong Bai, Menghan Xia, Xiao Fu, Xintao Wang, Lianrui Mu, Jinwen Cao, Zuozhu Liu, Haoji Hu, Xiang Bai, Pengfei Wan, et al. Recammaster: Camera-controlled generative rendering from a single video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14834–14844, 2025. 6

[2] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023. 6, 7

[3] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6299–6308, 2017. 2

[4] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. 7

[5] Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Scandinavian conference on Image analysis*, pages 363–370, 2003. 3, 4, 9

[6] Ruiqi Gao, Emiel Hoogeboom, Jonathan Heek, Valentin De Bortoli, Kevin Patrick Murphy, and Tim Salimans. Diffusion models and gaussian flow matching: Two sides of the same coin. In *The Fourth Blogpost Track at ICLR 2025*, 2025. 1, 2

[7] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems (NeurIPS)*, 30, 2017. 2

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

[9] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:6840–6851, 2020. 1

[10] Wenbo Hu, Xiangjun Gao, Xiaoyu Li, Sijie Zhao, Xiaodong Cun, Yong Zhang, Long Quan, and Ying Shan. Depthcrafter: Generating consistent long depth sequences for open-world videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2005–2015, 2025. 7

[11] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems (NeurIPS)*, 35:26565–26577, 2022. 1

[12] Diederik Kingma and Ruiqi Gao. Understanding diffusion objectives as the elbo with simple data augmentation. *Advances in Neural Information Processing Systems (NeurIPS)*, 36:65484–65516, 2023. 1, 2

[13] Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:21696–21707, 2021. 1

[14] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4015–4026, 2023. 7

[15] Zhengqi Li, Richard Tucker, Forrester Cole, Qianqian Wang, Linyi Jin, Vickie Ye, Angjoo Kanazawa, Aleksander Holynski, and Noah Snavely. Megasam: Accurate, fast and robust structure and motion from casual dynamic videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10486–10496, 2025. 7

[16] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022. 1

[17] Kunhao Liu, Ling Shao, and Shijian Lu. Novel view extrapolation with video diffusion priors. *arXiv preprint arXiv:2411.14208*, 2024. 6

[18] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022. 1

[19] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:5775–5787, 2022. 1

[20] Baorui Ma, Huachen Gao, Haoge Deng, Zhengxiong Luo, Tiejun Huang, Lulu Tang, and Xinlong Wang. You see it, you got it: Learning 3d creation on pose-free videos at scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2016–2029, 2025. 6

[21] Luigi Piccinelli, Yung-Hsu Yang, Christos Sakaridis, Mattia Segu, Siyuan Li, Luc Van Gool, and Fisher Yu. Unidepth: Universal monocular metric depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10106–10116, 2024. 7

[22] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Mearning (ICML)*, pages 8748–8763, 2021. 2

[23] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. 1

[24] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, 2016. 2

[25] Meituan LongCat Team, Xunliang Cai, Qilong Huang, Zhuoliang Kang, Hongyu Li, Shijun Liang, Liya Ma, Siyu

Ren, Xiaoming Wei, Rixu Xie, et al. Longcat-video technical report. *arXiv preprint arXiv:2510.22200*, 2025. 6, 7, 9

[26] Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018. 2

[27] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 6, 7, 9

[28] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5294–5306, 2025. 7

[29] Zeqi Xiao, Wenqi Ouyang, Yifan Zhou, Shuai Yang, Lei Yang, Jianlou Si, and Xingang Pan. Trajectory attention for fine-grained video motion control. In *International Conference on Learning Representations (ICLR)*, 2025. 6

[30] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024. 6

[31] Meng You, Zhiyu Zhu, Hui Liu, and Junhui Hou. Nvs-solver: Video diffusion model as zero-shot novel view synthesizer. In *International Conference on Learning Representations (ICLR)*, 2025. 6

[32] Mark Yu, Wenbo Hu, Jinbo Xing, and Ying Shan. Trajectorycrafter: Redirecting camera trajectory for monocular videos via diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 100–111, 2025. 6, 7, 8, 9

[33] Wangbo Yu, Jinbo Xing, Li Yuan, Wenbo Hu, Xiaoyu Li, Zhipeng Huang, Xiangjun Gao, Tien-Tsin Wong, Ying Shan, and Yonghong Tian. Viewcrafter: Taming video diffusion models for high-fidelity novel view synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–18, 2025. 6

[34] Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. *Advances in Neural Information Processing Systems (NeurIPS)*, 36:49842–49869, 2023. 1